С. В. Гордейчик

ОЦЕНКА ЗАЩИЩЕННОСТИ ИНТЕРНЕТ-ПОЛЬЗОВАТЕЛЕЙ

Введение

В современном мире интернет-безопасности защита клиентов является одной из наиболее актуальных задач. Развитие средств защиты, эффективное управление обновлениями резко снизили вероятность успешной и незаметной атаки на серверные системы. В результате киберпреступнику гораздо выгоднее проводить атаки на пользовательские компьютеры, где уровень защиты традиционно ниже. Даже распространенные на данный момент атаки на веб-приложения, как правило, используются не для дальнейшей компрометации сетевой инфраструктуры, а для заражения рабочих станций агентами бот-сетей, осуществляющих сбор банковской и приватной информации, организации рассылки спама и DDOS-атак.

Наиболее распространенным вектором атак на рабочие места является использование неустраненных уязвимостей реализации в браузере пользователя. Ежегодно в этих программах обнаруживаются сотни уязвимостей различного типа, причем значительная часть (до 36 %) не устраняется производителем в приемлемые сроки [1].

Еще одну группу приложений, широко используемых злоумышленниками, составляют расширения для браузеров, такие как Java, Flash, компоненты ActiveX и другие. В группу риска также попадают программы, обрабатывающие популярные в Интернете форматы данных, такие как изображения, документы в формате PDF и т. д.

Результаты экспериментов в реальных условиях [2] показывают, что своевременное устранение уязвимостей реализации и обновление версий ПО являются одними из наиболее эффективных методов обеспечения безопасности. Однако отсутствие в ОС Windows (как наиболее распространенной ОС) эффективного механизма обновления ПО сторонних производителей снижает вероятность адекватного управления уязвимостями на компьютерах интернет-пользователей практически до нуля.

Современные средства проактивной защиты, теоретически позволяющие компенсировать данный недостаток, к сожалению, не вполне эффективно справляются с атаками, использующими уязвимости в ПО. По некоторым оценкам [3], наивысшая эффективность представителей данного класса средств защиты составляет около 30 %.

В связи с этим требуется механизм, который позволял бы решать следующие задачи:

- повышение осведомленности интернет-пользователей в области безопасности;
- анализ ОС и прикладного ПО на наличие распространенных уязвимостей;
- управление процессом устранения уязвимостей (рекомендации, автоматизированная установка обновлений и т. д.).

Традиционным средством защиты в подобных ситуациях являются средства анализа защищенности (сканеры безопасности).

1. Безагентное сканирование

К сожалению, ни один из распространенных механизмов оценки защищенности не подходит для решения задачи оперативной проверки безопасности интернет-приложений пользователя. Идеальная реализация концепции анализа для решения описанных задач подразумевает комбинацию свойств сетевых и системных сканеров, а именно: возможность анализа узла посредством сетевого взаимодействия без установки дополнительного ПО (режим Pentest) и широкий доступ к ресурсам компьютера, реализуемый системными сканерами (режим Audit). Максимально близки к ней системные сканеры, устанавливающие агента через браузер при посещении веб-сайта с использованием Java или элементов управления ActiveX. Однако такие сканеры требуют взаимодействия с пользователем для установки ПО и работают достаточно медленно.

Резюмируя требования к системам оценки защищенности интернет-пользователей, можно выдвинуть два основных тезиса: не должна требоваться в явном виде установка ПО, скорость работы должна быть сопоставима со временем просмотра веб-страницы, т. е. порядка единиц секунд.

Для реализации систем оценки защищенности, которые удовлетворяли бы указанным требованиям, из распространенных в Интернете технологий в наибольшей степени подходит технология AJAX. Рассмотрим подходы к реализации оценки защищенности на основе AJAX.

2. Идентификация версий приложений

Одним из базовых механизмов средств анализа защищенности являются так называемые «баннерные проверки», основанные на идентификации версий приложений. Полученный список установленных приложений и их версий сопоставляется со списком уязвимостей, содержащимся в базе знаний браузера. Данный механизм легко реализуем с помощью технологии AJAX и может использоваться для проверки уязвимостей ПО, механизм обновлений которого основан на изменении версии. Примерами подобных программ являются браузеры Mozilla Firefox и Opera, приложения Java и Adobe Flash.

Как правило, баннер для браузера передается в HTTP-заголовке User-Agent. Для уточнения этой информации можно применять различные методы идентификации, например характерные отклонения от стандарта HTTP, использование заглавных и строчных букв, порядок следования заголовков [4] и т. д. Более точный метод — интерактивные вызовы свойств JavaScript navigator.appName и navigator.appVersion, в ответ на которые браузер возвращает свою версию.

Этой информации достаточно для того, чтобы идентифицировать тип и версию браузера и задействовать соответствующие механизмы для проведения дальнейших проверок. Кроме того, для браузеров, в которых устранение уязвимостей приводит к изменению версий, этой информации достаточно для идентификации актуальных проблем безопасности приложения.

Баннерный метод также может применяться для идентификации ряда вредоносных программ, модифицирующих заголовки HTTP-запросов. Примером такой программы является Trojan.Win32.Dialer.hc [5].

Кроме того, браузеры, которые поддерживают метод navigator.plugins, дают возможность получить список установленных расширений, и в ряде случаев данный список содержит номера версий ПО. Таким методом в Firefox и Opera можно идентифицировать версии расширений Flash и Java, а также ряда других приложений. Однако оба указанных расширения могут быть определены более точным методом.

3. Определение уязвимостей Flash и Java

Язык ActiveScript, используемый технологией Flash, содержит переменную $version, в которой указывается версия используемого Flash-плеера. Результат вызова этого метода может быть получен Javascript и передан серверу для анализа.

Версия расширения Java определятся аналогично с использованием свойств java.version и java.vendor объекта java.lang.System.

Кроме того, как уже говорилось выше, некоторые браузеры позволяют получить версию Java и Flash из списка установленных расширений.

Plugins	Java(TM) Platform SE 6 U12 Mozilla Default Plug-in Adobe Acrobat 2007 Microsoft Office system Shockwave Flash: 10.0.22 Java(TM) Platform SE 6 U12

Рис. 1. Список расширений Firefox

4. Идентификация версий ActiveX

Последние несколько лет элементы управления ActiveX широко используются злоумышленниками для проведения атак на системы пользователей [6]. Со временем ситуация стала настолько напряженной, что компания Microsoft приступила к выпуску обновлений безопасности, блокирующих компоненты ActiveX сторонних производителей [7]: для компонентов устанавливается значение параметра «Kill Bit», которое отключает использование компонента в браузере.

Стандартным подходом к определению версии ActiveX является обратная разработка компонента с целью определения наличия и формата вызова метода, возвращающего его версию. Однако такой подход трудно назвать универсальным, поскольку он требует существенных трудозатрат для каждого из компонентов, что затрудняет добавление новых проверок. Кроме того, при отсутствии у компонента метода, возвращающего номер версии, проверки таким способом выполнить невозможно.

Для решения этой задачи может использоваться механизм обратной совместимости ActiveX-компонентов. Дело в том, что при попытке создания компонента с явным указанием номера версии компонент будет создан даже в случае, если в системе установлен ActiveX старшей версии. В обратной ситуации, когда при создании компонента указывается версия, номер которой больше номера версии компонента, существующего в системе, вызов приведет к ошибке.

Таким образом, последовательно пытаясь создавать компоненты ActiveX с версиями, номера которых больше номеров последних уязвимых версий, можно определить, какие версии компонента установлены в системе и являются ли они уязвимыми.

Использование данного подхода дает возможность проверять наличие известных уязвимостей реализации в произвольных компонентах ActiveX без существенных затрат на добавление новых проверок.

5. Уязвимости в Internet Explorer

Поскольку компания Microsoft использует для устранения уязвимостей бинарные исправления («патчи»), установка которых не изменяет версии ПО, баннерный метод в данном случае не применим. Методы AJAX не позволяют получить доступ к реестру или файловой системе, и традиционный подход определения списка установленных в системе исправлений не работает. Чтобы обойти это ограничение, можно применять хорошо зарекомендовавший себя в сетевых сканерах метод проверок типа «эксплойт». Этот подход использует разницу в поведении систем с установленным обновлением и без него.

Проверки этого типа являются наиболее сложными и ресурсоемкими с точки зрения реализации, поскольку требуют индивидуального исследования каждой уязвимости. Более того, в данной реализации необходима особая осторожность при проведении проверки, поскольку сбои в работе пользовательского браузера в ходе проверок неприемлемы. С целью демонстрации данного подхода рассмотрим методы определения одного из обновлений в Internet Explorer.

Первый из них, касающийся AJAX в Internet Explorer, позволяет с помощью объекта XMLHTTPRequest создавать HTTP-запросы с модифицированными заголовками путем внедрения в поля запроса символов перевода строки. Таким образом, если в IE не установлено соответствующее обновление (MS07-042 [8]), то попытка отправить подобный запрос будет удачна и сервер вернет соответствующий статус. В противном случае будет возвращен код 404 или другой код ошибки.

Другой метод определения обновлений связан с версиями компонентов ActiveX. Дело в том, что установка обновлений Internet Explorer в ряде случаев приводит к изменению версий используемых в системе ActiveX-компонентов. Таким образом, контролируя версии установленных

компонентов, можно идентифицировать наличие или отсутствие обновлений браузера, а также других приложений, например Microsoft Office. Ряд обновлений используется для блокировки возможности создания небезопасных ActiveX. В этом случае ошибка при попытке активации компонента будет явно указывать на наличие обновления.

6. Практическая реализация

В настоящее время практическая реализация данной концепции представлена в нескольких продуктах, таких как Mozilla Plugin Check [9] и SurfPatrol.Ru [10].

Система Mozilla Plugin Check реализует только один из описанных в данной публикации методов и позволяет оценивать защищенность только браузеров, реализующих технологию расширений (plugins), что ограничивает практическое применение.

Система SurfPatrol, в которой реализованы описанные в данной статье подходы, позволяет оперативно проводить экспресс-анализ безопасности компьютера пользователя без установки дополнительного программного обеспечения. Система спроектирована для использования на интернет- и интранет-порталах и может предоставляться с различным уровнем детализации, например с индикатором («баннером»), указывающим на общую степень риска, или с полным списком обнаруженных проблем.

Система реализована с использованием технологии AJAX и позволяет оперативно оценивать защищенность следующих функций системы пользователя:

- проверка наличия обновлений безопасности браузера (Internet Explorer, Mozilla Firefox, Opera и т. д.);
- проверка наличия обновлений безопасности расширений браузера (Adobe Flash, Java и др.);
- проверка наличия небезопасных версий компонентов ActiveX.

Подобные решения могут использоваться для размещения на пользовательских интернет-порталах (например, в системах интернет-банкинга), в системах карантина интернет-доступа (например, Network Access Control) и т. д.

7. Ограничения метода

Безагентные решения являются самым молодым из подходов к решению задачи оценки защищенности пользователей. В этом случае пользователю не требуется устанавливать дополнительные программы и весь процесс анализа системы осуществляется с использованием веб-технологий.

Преимуществами решений на основе веб-агента являются:

- возможность кроссплатформенной реализации;
- автоматическая проверка через веб-сайт;
- высокая скорость работы (единицы секунд);
- минимальные привилегии, необходимые на компьютере пользователя.

К недостаткам можно отнести:

- небольшой охват анализируемых приложений (браузеры, расширения браузеров);
- зависимость детализации получаемой информации от используемого браузера.

СПИСОК ЛИТЕРАТУРЫ:

1. *Cenzic*. Web Application Security Trends Report Q3-Q4, 2009. URL: http://www.cenzic.com/downloads/Cenzic_AppsecTrends_Q3-Q4-2009.pdf.
2. The Honeynet Project. Know Your Enemy: Malicious Web Servers. URL: http://www.honeynet.org/papers/mws.

3. *Secunia*. Internet Security Suite test October 2008. URL: http://secunia.com/gfx/Secunia_Exploit-vs-AV_test-Oct-2008.pdf.
4. *Дубровин В*. Утечка данных через служебную информацию и сетевой протокол в клиентском приложении. URL: http://securityvulns.ru/articles/xs/client_information_leak.asp.
5. Threat Expert. Dialer.HC. URL: http://www.threatexpert.com/report.aspx?md5=47c376a6d8afa73d45d23e16edef1ea0.
6. Month of ActiveX Bug. URL: http://moaxb.blogspot.com.
7. Microsoft. Update Rollup for ActiveX Kill Bits. URL: http://www.microsoft.com/technet/security/advisory/960715.mspx.
8. Microsoft. Описание обновления для системы безопасности служб MSXML 3.0. URL: http://support.microsoft.com/kb/936021.
9. Mozilla. Mozilla Plugin Check. URL: http://www.mozilla.com/en-US/plugincheck.
10. Positive Technologies. SurfPatrol.Ru. URL: http://www.surfpatrol.ru.