Безопасность HTML5 – часть первая

Автор: Michael Schmidt (Compas Security)

Источник: http://www.securitylab.ru/analytics/419068.php

Данный документ содержит выдержки из магистерской диссертации Майкла Шмидта. Здесь отражены рассмотренные в диссертации аспекты HTML5, касающиеся безопасности.

1. Введение

1.1 История HTML5 и текущая модель Всемирной паутины

На данный момент действующим стандартом HTML (Hypertext Markup Language), установленным W3C (World Wide Web Consortium или Консорциумом Всемирной паутины), является HTML 4.01 [1]. Этот стандарт определяет, как должен использоваться HTML для описания веб-страниц. XHTML 1.0 и XHTML 1.1 обладают в основном той же функциональностью, что и HTML 4.01, за исключением нескольких ограничений и расширений HTML. Однако, эти два стандарта основаны на XML (Extensible Markup Language), а не на SGML (Standard Generalized Markup Language) [2].

HTML5 (HTML версии 5) является потомком HTML 4.01, XHTML 1.0 и XHTML 1.1. Производители браузеров: Apple Computer, Inc., Mozilla Foundation и Opera Software ASA основали сообщество WHATWG (Web Hypertext Application Technology Working Group или Рабочую группу по развитию технологии гипертекстовых веб-приложений) в 2004 году с намерением развивать и расширять новые веб-технологии сперва под знаком Web Application 1.0, а позднее под именем HTML5. Одна из основных причин создания WHATWG заключалась в том, что среди производителей барузеров росло беспокойство по поводу предложенной W3C концепции XHTML2. W3C в это время разрабатывало стандарт XHTML2, но прекратило над ним работу в 2009 для ускорения разработки HTML5 [6]. С этого момента и W3C, и WHATWG работают над HTML5, но поддерживают свои версии спецификаций, которые немного отличаются в некоторых вопросах. Однако, главный автор, Ян Хиксон, работает над версией WHATWG. Поскольку

разработка HTML5 определяется в основном WHATWG, некоторые критикуют HTML5 за то, что он испытывает слишком большое влияние со стороны производителей браузеров и слишком малое со стороны пользователей Всемирной паутины [8]. Это может отразиться на безопасности, что и показывает данный документ.

HTML5 по версии WHATWG имеет статус "Living Standard" [9], а по версии W3C - "Working Draft" (Рабочий проект) [3]. Несколько производителей браузеров уже реализовали множество возможностей HTML5. Статус кандидата в рекомендации планируется к 2012, а статус официальной рекомендации к 2022 [5]. Для тестирования поддержки браузером тех или иных возможностей HTML5 существуют веб-сайты вроде [10]. Тем не менее, в октябре 2010 года W3C официально заявил, что HTML5 еще не готов для использования в современных веб-приложениях из-за проблем с возможностями взаимодействия [11]. По этой причине положения, дающиеся в данном документе, могут измениться, то есть необходимо внимательно рассматривать контекст, в котором описаны атаки и средства противодействия им. Изменения в спецификации HTML5 могут как смягчить описанные атаки, так и породить новые уязвимости.

HTML5 предоставляет новые возможности веб-приложениям, но также рождает новые проблемы безопасности. Один из известных примеров некорректного использования HTML5 – вечные куки [12], которые открыто обсуждались в новостях безопасности [13]. С помощью этих вечных куки и нескольких технологий можно попытаться скоррелировать пользовательские сессии. Кроме того, новые технологии HTML5 вроде Web Storage используются для хранения уникальных идентификаторов в браузере на стороне клиента. При обсуждении реализации веб-приложений на основе HTML5 нужно учитывать эти проблемы безопасности, равно как и новые возможности.

На рисунке 1 показана высокоуровневая модель Всемирной паутины, на которой основана вторая глава. Рамка обозначает поставщика веб-приложений,

который состоит из веб-сервера, обслуживающего по крайней мере один сайт и хранящего данные в базе. Сайт в ответ на запросы предоставляет ресурсы клиентам через Интернет.

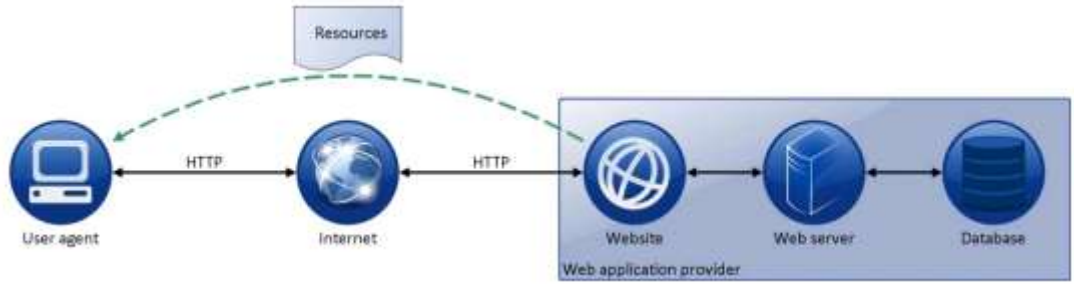

Рисунок 1. Стандартная модель Всемирной паутины

Вот более детальное описание показанных на рисунке сущностей:

- **Ресурсы:** Ресурсы – любые сетевые данные или сервисы, доступные через Интернет. Их местоположение определяется через URI (унифицированный идентификатор ресурса) [14]. Примерами ресурсов являются веб-страницы, содержащие HTML/CSS и JavaScript код, а также ссылки на дополнительные ресурсы вроде рисунков и видео.
- **Пользовательский агент** (ПА, User-Agent, UA): ПА – это потребитель веб-приложений, который запрашивает ресурсы у поставщика веб-приложений. Ресурсы обрабатываются ПА, рендерятся и отображаются для конечного пользователя. ПА имеет возможность установления HTTP (Hypertext Transfer Protocol) соединений с веб-серверами для корректного рендеринга html/css и выполнения JavaScript. Более того, в ПА реализованы стандарты HTML 4.01 и HTML 5 и соответствующие возможности вроде Geolocation API (раздел 2.8) или Web Storage (раздел 2.3).
- **Веб-приложение** – общий термин, обозначающий поставщика веб-ресурсов, который состоит из трех главных частей:
- **Веб-сайт**. Веб-сайт состоит из нескольких отдельных веб-ресурсов и доступен по своему URI.
- **Веб-сервер**. Веб-сервер предоставляет хостинг для по меньшей мере одного веб-сайта. HTTP(S)-соединение устанавливается между ПА и веб-сервером. Помимо хостинга сайтов, веб-сервер также предоставляет дополнительные ресурсы. Другие соединения, вроде Web Socket API (раздел 2.7), также устанавливаются между ПА и веб-сервером.
- **База данных**. Базы данных хранят любые данные, необходимые для веб-приложений, например, персональную информацию о пользователях.

- **Интернет**. ПА получает доступ к веб-приложениям через Интернет. ПА может соединяться с любым веб-приложением и не ограничен в целях. Веб-приложения, в свою очередь, доступны из любой точки Интернета. На Рисунке 1 представлена глобальная модель, рассматриваемая в главе 2. Однако, конкретные модели, используемые в дальнейших разделах, могут немного отличаться в зависимости от описываемого сценария. Например, один из сценариев описывает Интранет-приложение, доступ к которому возможен только в пределах внутренней корпоративной сети. Эти изменения будут указаны явно в соответствующих разделах.

1.2 Мотивация

С тех пор, как Джон фон Нейман опубликовал свою теорию самовоспроизводящихся программ в 1949, атаки на компьютерные системы эволюционировали, как и атаки на веб-приложения. Одной из первых описанных крупных атак на веб-приложения была DDoS-атака (distributed Denial-Of-Service) на Yahoo, eBay, Amazon, и несколько других веб-сайтов в 2000 [16].

Веб-сервера являются постоянными целями для атак. Они, как правило, доступны 24 часа 7 дней в неделю и 365 дней в году. Это делает возможными хорошо спланированные ручные и автоматизированные атаки на веб-сервера. Консорциум безопасности веб-приложений провел в 2008 году исследование, которое показало, что 97554 из 12186 протестированных веб-сайтов (87.5%) имеет уязвимости [17]. Компания WhiteHat Security протестировала около 2.000 сайтов в своем исследовании, которое выявило, что средний сайт имеет 13 уязвимостей [18]. Компания Verizon в отчете за 2010 год по исследованию «Data Breach Investigations» пишет, что за 6 лет было выявлено более 900 инцидентов в ходе которых было скомпрометировано более 900 миллионов записей (по данным, предоставленным Секретной Службой США) [19].

Конечные пользователи также являются целью многих атак. Лаборатория Касперского в своем бюллетене безопасности за 2010 год сообщает, что число

drive-by атак исчисляется десятками миллионов и что на пользователей сети Kaspersky Security Network было проведено 73,619,767 атак. Secunia пишет, что в приложениях третьих фирм было обнаружено гораздо больше уязвимостей, чем в программах Microsoft [21]. Эта закономерность особенно интересна в контексте веб-браузеров: количество найденных уязвимостей в Internet Explorer равно 51 [22], когда для Mozilla Firefox оно составляет 95 [23] (но необходимо учитывать, что не все уязвимости одинаково критичные). Symantec пишет в своем ежегодном отчете за 2010 год, что было зарегистрировано 339.600 различных штаммов вредоносного ПО, найденных в электронных письмах, заблокировано более 188.6 миллионов фишинговых писем и обнаружено 42.926 различных доменов с вредоносным контентом, причем 90% из них являются легитимными, но скомпрометированными сайтами [24]. Всего Лаборатория Касперского зафиксировала 327.598.028 атак на компьютеры пользователей только в первом квартале 2010 года [25].

Как видно, существует множество атак на веб-приложения (судя по 2010 году), и необходимость обеспечения безопасности в Интернет растет. Помимо удобств, предоставляемых Всемирной паутиной, критичными моментами, нуждающимися в рассмотрении, являются вопросы безопасности. Это справедливо как для нынешних, так и для будущих веб-приложений. Угрозы для веб-приложений, описанные в данном разделе, необходимо иметь в виду при рассмотрении проблем безопасности HTML5.

2. Проблемы безопасности HTML5.

HTML5 вносит в HTML несколько технологических изменений. В данной главе технически изложены последствия для безопасности, которые повлекут за собой эти изменения.

2.1 Введение

В процессе создания спецификации HTML5 соображения, связанные с безопасностью, фиксировались с самого начала. Каждая часть спецификации имеет свой подраздел, касающийся безопасности. Эти подразделы покрывают

моменты, которые нужно хорошенько продумать при реализации соответствующих частей. В подразделах описываются уязвимости, которые могут возникнуть при реализации соответствующей возможности, а также безопасный способ ее реализации производителями браузеров. Например, авторы спецификации HTML5 выделили уязвимость под названием Information leakage (Утечка информации) для элемента canvas, которая возникает, если скрипты могут получать доступ к информации из разных источников. Далее в спецификации дается тщательное описание того, как избежать этой уязвимости в безопасной реализации (соответствующая выдержка из спецификации canvas для HTML5 находится в разделе 5.5.1 данного документа).

Помимо инструкций по безопасной реализации функционала, стандарт HTML5 содержит инновационные возможности, позволяющие решать существующие проблемы безопасности HTML:

- Web Messaging: делает возможным безопасное общение между элементами с различных доменов и снимает необходимость в небезопасных хаках (см. раздел 2.5).
- Inline Frame (Iframe) Sandboxing (Песочница для Iframe): Встроенные элементы Iframe теперь могут быть ограничены в возможностях. Можно, например, запретить запускать JavaScript [26] (см. раздел 2.9.3).

Вот еще пара примеров решения существующих проблем веб-приложений:

- Сокрытие реферера: Добавление в ссылку атрибута rel со значением noreferrer предотвращает утечку информации через поле запроса referrer при переходе по ссылке. Это особенно полезно в почтовых веб-приложениях (POC приложение есть в разделе 5.2.11).
- Безопасный контент-снифинг: Процедура определения типа ресурса теперь определена строго, что смягчает атаки на алгоритм контент-снифинга, т. е. автоматического определения типа контента (описано в [27]). Выдержки из спецификации HTML5, которые описывают правила определения типа контента, приведены в разделе 5.5.2.

Следующий список дает представление о возможностях HTML5, обсуждаемых в данной главе. Каждая возможность, описанная в этом списке, будет проанализирована более детально в своем подразделе. Таким образом, сначала дается представление о возможности, а затем описываются уязвимости и угрозы, возможные сценарии атаки и (если существуют) меры противодействия им. В главе рассматриваются следующие возможности HTML5:

- **Cross-Origin Resource Sharing** (CORS, Междоменное разделение ресурсов)[28]: CORS позволяет клиентам делать междоменные запросы через XMLHttpRequest. В HTML5 ослаблена политика ограничения домена, которая изолирует документы с различных доменов друг от друга [29]. При определенных условиях в HTML5 возможно запрашивать ресурсы с других доменов и делиться с ними информацией.
- **Web Storage** (Веб-хранилище) [30]: С помощью технологии Web Storage веб-приложения могут обходить ограничения количества хранимой на стороне клиента информации. Используя Web Storage, веб-приложения могут хранить около пяти мегабайт данных на стороне клиента и получать к ним доступ через JavaScript в ходе последующих сессий.
- **Offline Web Application** (Оффлайновые веб-приложения) [31]: Используя эту технологию HTML5, веб-приложения могут работать в оффлайн-режиме. Сначала, в онлайн-режиме, веб-приложение посылает ПА инструкции, в результате которых необходимая информация сохраняется в оффлайновом кэше веб-приложения. Впоследствии приложение можно использовать в оффлайн-режиме, не нуждаясь в Интернет-соединении.
- **Web Messaging** [32]: Iframe-ы одного веб-приложения, соответствующие разным доменам, могут общаться друг с другом через HTML5 Web Messaging. При разработке Iframe можно заложить возможность принимать сообщения от других Iframe-ов.
- **Custom scheme and content handlers** (Специальные обработчики схемы и контента) [3]: HTML5 позволяет веб-приложениям регистрироваться в качестве обработчиков URI-схем и типов контента. Например, веб-

приложение может зарегистрироваться как обработчик почтовых ссылок (mailto): на какую бы ссылку типа mailto на каком бы домене не нажал пользователь, он будет перенаправлен на зарегистрированное веб-приложение.

- **Web Sockets API** [33]: Этот прикладной программный интерфейс HTML5 предоставляет способ установления полнодуплексного канала между сервером и ПА. Через этот канал возможен асинхронный обмен данными. Приемы AJAX (Asynchronous JavaScript and XML) больше необязательны для установления асинхронного соединения.
- **Geolocation API** [34]: Используя Geolocation API, веб-приложения могут определять местоположение пользователя. Это позволяет веб-приложениям предоставлять своим потребителям услуги в зависимости от их местоположения. Эта возможность особенно интересна для мобильных пользователей.
- **Неявные особенности HTML5**, касающиеся безопасности: В данном подразделе будут описаны некоторые особенности HTML5, которые не являются прямыми источниками новых уязвимостей, но могут быть использованы опосредованно для запуска атак. В подразделе раскрываются эти особенности, а также объясняется их связь с другими уязвимостями.

На рисунке 2 показана высокоуровневая диаграмма, дающая представление о возможностях HTML5 и о том, как они соотносятся друг с другом в контексте веб-браузера. DomainA.csnc.ch представляет домен загруженного веб-сайта, в который встроены три Iframe с различными источниками. Iframe, загруженный из untrusted.csnc.ch, запускается в песочнице и не имеет разрешений на запуск JavaScript. Iframe-ы, загруженные из anydomainA.csnc.ch и anydomainB.csnc.ch, общаются между собой посредством Web Messaging. Специальные обработчики схемы и контента зарегистрированы приложением с домена domainB.csnc.ch, к которому происходит обращение, когда пользователь запрашивает подходящий тип ресурса. С домена domainC.csnc.ch загружаются

дополнительные ресурсы через CORS. Geolcation API, Offline Web Application, Web Storage и Web Workers представляют возможности HTML5 на стороне клиента (реализованы в ПА), которые могут использоваться веб-сайтами. В данном примере подразумевается, что anydomainB.csnc.ch использует все эти возможности.

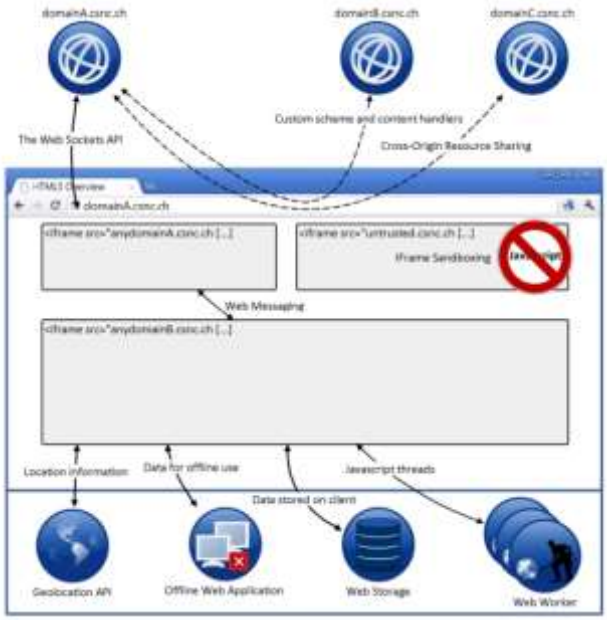

Рисунок 2. Обзор технологий HTML5

Список уязвимостей и атак из данной главы не является всеобъемлющим. Он не покрывает все возможные уязвимости, угрозы и атаки, характерные для HTML5. По мнению автора, список ограничивается наиболее важными и критичными пунктами. Для большинства атак разработаны приложения, демонстрирующие возможность их проведения (так называемые Proof-Of-Concept или POC-приложения). Эти приложения собраны в главе 5 и имеют ссылки в соответствующих разделах. Некоторые атаки продемонстрированы сторонними приложениями, ссылки на которые также приведены в соответствующих разделах.

Список источников:

[1] World Wide Web Consortium (W3C). (1999, Dec.) HTML 4.01 Specification, W3C Recommendation . [Online]. http://www.w3.org/TR/1999/REC-html401-19991224/

[2] The World Wide Web Consortium (W3C) . (2000, Jan.) XHTML 1.0 The Extensible HyperText Markup Language (Second Edition). [Online]. http://www.w3.org/TR/xhtml1/

[3] The World Wide Web Consortium (W3C). (2011, Jan.) HTML5 - A vocabulary and associated APIs for HTML and XHTML. [Online]. http://www.w3.org/TR/html5/

[4] M. Pilgrim, HTML5: Up and Running. Sebastopol: O'Reilly Media, 2010.

[5] Web Hypertext Application Technology Working Group (WHATWG). (2011, Jan.) FAQ - What is the WHATWG?. [Online]. http://wiki.whatwg.org/wiki/FAQ

[6] World Wide Web Consortium (W3C). (2009, Jul.) XHTML 2 Working Group Expected to Stop Work End of 2009, W3C to Increase Resources on HTML 5. [Online]. http://www.w3.org/News/2009#entry-6601

[7] M. Schäfer. (2010, Dec.) Übersicht über HTML5-Spezifikationen und -Literatur. [Online]. http://molily.de/weblog/html5-specs

[8] P. Kröner, HTML5 Webseiten innovativ und zukunftssicher. München: Open Source Press, 2010. [9] Web Hypertext Application Technology Working Group (WHATWG). (2011, Feb.) HTML - Living Standard. [Online]. http://www.whatwg.org/specs/web-apps/current-work/multipage/

[10] N. Leenheer. (2010, Jun.) THE HTML5 TEST – HOW WELL DOES YOUR BROWSER SUPPORT HTML5?. [Online]. http://html5test.com/

[11] P. Krill and P. L. Hegaret. (2010, Oct.) W3C: Hold off on deploying HTML5 in websites. [Online]. http://www.infoworld.com/d/developer-world/w3c-hold-html5-in-websites-041

[12] S. Kamkar. (2010, Oct.) Evercookie - virtually irrevocable persistent cookies. [Online]. http://samy.pl/evercookie/

[13] J. Bager. (2010, Sep.) Das Zombie-Cookie, Heise Security. [Online]. http://www.heise.de/security/meldung/Das-Zombie-Cookie-1094770.html

[14] Internet Engineering Task Force - The Internet Society. (2005) Uniform Resource Identifier (URI): Generic Syntax. [Online]. http://tools.ietf.org/html/rfc3986