электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК 004.453

Некоторые особенности метапоиска в защищенных информационных системах

Лабун И.В.

Студент, кафедра «Компьютерные системы и сети»
МГТУ им. Баумана, г. Москва, Россия

Научный руководитель: Самарев Р.С., к.т.н.,
доцент кафедры «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана

МГТУ им. Н.Э. Баумана
samarev@acm.org

В процессе работы сотруднику предприятия часто требуется искать информацию в защищенных (тем или иным способом) информационных системах. При разработке метапоисковой системы, то есть системы, объединяющей данные из различних источников, следует учитывать этот факт. Важно обеспечить доступ метапоисковой системы к таким ресурсам, в зависимости от текущих прав пользователя. В данной статье рассмотрены вопросы обеспечения метапоиска информации в защищенных системах с разграничением доступа.

Исходные данные задачи следующие: имеется предприятие с определенным количеством внутренних информационных систем: система документооборота, система управления проектами, файловое хранилище и т.п. Среди данных ресурсов есть как общедоступные ресурсы, так и закрытые, доступ к которым предоставляется по логину и паролю. Требуется обеспечить централизованный поиск по всем этим ресурсам.

Для обеспечения поиска было решено разработать метапоисковую систему, так как из-за необходимости учёта и непостоянства прав доступа пользователей к информации, централизованное хранение индексов для всех информационных ресурсов было

невозможно. При этом в процессе поиска должно проводиться разграничение доступа пользователей к информации, т.е. для каждого пользователя (или группы пользователей) должны устанавливаться те ресурсы, в которых он может (или не может) производить поиск. Также параллельно проводились работы по созданию единой точки входа во все эти информационные ресурсы (Single Sign-On system - SSO), которая основывалась на использовании технологии OpenID. Данная технология хорошо подошла для решения проблемы метапоиска в защищенных системах.

Анализ вариантов авторизированного метапоиска

В виду того, что в защищенных системах необходимо проведение аутентификации и авторизации пользователей, назовём способ сбора информации из них авторизированным метапоиском. Рассмотрим несколько способов его организации. Первый способ — модификация отдельных информационно-поисковых систем (ИПС), с целью создания специального API для авторизированного поиска. Данный способ недостаточно универсален и имеет ограниченную применимость, т.к. пригоден только для систем с открытым исходным кодом и затрудняет добавление новых отдельных ИПС.

Второй способ — научить метапоисковую систему имитировать поведение браузера пользователя во время авторизации. Для этого нужно, чтобы метапоисковая система получила данные авторизации от браузера пользователя и от сервера, а затем использовала их каждый раз при осуществлении поиска. Это проще всего сделать, используя метапоисковую систему в качестве прокси-сервера. Т.е. метапоисковая система будет находиться между пользователем и отдельными ИПС в тракте передачи данных. Получается, что требуется воспроизвести классическую атаку «Человек посередине» (Man in the middle), в которой метапоисковая система выполняет роль «злоумышленника». Данный способ предпочтительней первого, т.к. во-первых, не требует дополнительной модификации отдельных ИПС, во-вторых, при добавлении нового компонентного поискового движка (т.е. ИПС, являющейся источником информации для метапоисковой системы) не требуется изменять ни саму метапоисковую систему, ни поисковые системы нижнего уровня, т.е. свойства расширяемости метапоисковой системы не ухудшаются.

Методы аутентификации в Интернете

Применительно к веб-приложениям, аутентификация — это проверка подлинности предъявленного пользователем идентификатора (логина). После аутентификации по логину веб-приложение определяет полномочия пользователя (происходит процесс авторизации), после чего решается, какую информацию предоставлять пользователю в ответ на его запрос. Самый распространенный способ аутентификации — это проверка подлинности имени пользователя секретной фразой (паролем).

На данный момент существуют три основных протокола обмена данными аутентификации (логин и пароль) в веб-приложениях:

1. базовая аутентификация (basic access authentication);
2. дайджест аутентификация (digest access authentication);
3. аутентификация по cookie (cookie authentication).

Базовая аутентификация — наиболее простой протокол обмена идентификационной информацией. Протокол базовой аутентификации следующий. Когда сервер хочет, чтобы клиент предоставил ему данные аутентификации, он включает в HTTP-заголовок ответа поле **WWW-Authenticate: Basic realm="realm"** и выставляет код ответа 401 (Unauthorized). Realm – строка, идентифицирующая группу ресурсов, для которых запрашиваются данные аутентификации. Для каждой группы ресурсов идентификационные данные могут отличаться для одного и того же пользователя. Далее клиент при следующем запросе к ресурсу добавляет в HTTP-заголовок запроса поле **Authorization: Basic <данные аутентификации>**. Данные аутентификации — закодированная в Base64 строка, полученная при последовательной конкатенации имени пользователя, двоеточия («:») и пароля. При неудачной аутентификации сервер снова отправляет ответ с кодом 401 и выставляет в HTTP-заголовок **WWW-Authenticate: Basic realm="realm"**. В случае же, когда клиент предоставил верные данные аутентификации, сервер отправляет в ответ запрашиваемые данные. Таким образом, для того, чтобы получить данные, для доступа к которым требуется Basic аутентификация клиент (браузер пользователя) должен каждый раз включать в заголовок запроса поле Authorization со своими идентификационными данными.

Базовая аутентификация имеет один существенный недостаток — она совершенно беззащитна к атакам типа прослушивания сетевого трафика. Идентификационные данные никак не шифруются (они всего лишь кодируются в Base64) и не изменяются во времени, поэтому злоумышленнику их очень легко перехватить и использовать для доступа к конфиденциальной информации.

Дайджест аутентификация лишена данного недостатка, т.к. предусматривает передачу идентификационных данных в зашифрованном виде (применяется широко известный алгоритм MD5), причем каждый раз при составлении зашифрованной строки применяются случайным образом сгенерированные сервером данные, что предотвращает повторное использование кодовой строки. Рассмотрим данный протокол подробней.

При запросе ресурса, требующего аутентификации, клиент получает ответ с кодом 401 (Unauthorized) и указанием способа аутентификации в заголовке (WWW-Authenticate: Digest <данные для составления кодовой строки>). Кодовая строка составляется из имени

пользователя, пароля и данных, полученных от сервера. Эти данные включают в себя реалм (realm), домен (domain), случайный одноразовый код (nonce) и т.п. Далее полученные данные комбинируются с информацией, введенной пользователем, шифруются (обычно с применением алгоритма MD5, но также возможны другие варианты) и отправляются на сервер при следующем запросе в заголовке Authorization.

Приведем пример процесса аутентификации с использованием дайджест аутентификации.

1. Клиент запрашивает данные у сервера

GET /dir/index.html HTTP/1.0

Host: localhost

2. Сервер отправляет ответ с кодом 401 и информацией о методе авторизации

Date: Sun, 10 Apr 2012 20:26:47 GMT

WWW-Authenticate: Digest realm="testrealm@host.com",

qop="auth,auth-int",

nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",

opaque="5ccc069c403ebaf9f0171e9517f40e41"

Content-Type: text/html

Content-Length: 311

3. Пользователь вводит имя пользователя и пароль в браузере

4. Сервер посылает ответ с кодом 200 (OK)

HTTP/1.0 200 OK

Server: HTTPd/0.9

Date: Sun, 10 Apr 2005 20:27:03 GMT

Content-Type: text/html

Content-Length: 7984

Аутентификация с помощью куки (cookie) является самой распространенной схемой аутентификации в Интернете. Куки – данные, которые хранятся на клиенте и отправляются серверу с каждым запросом. Куки передаются клиенту в HTTP-заголовке Set-Cookie и имеют следующий формат:

Set-Cookie: NAME=VALUE; expires=DATE; path=PATH; domain=DOMAIN_NAME; secure

Обязательными полями являются только NAME и VALUE, которые представляют название и значение куки соответственно. После получения клиентом куки, они отправляются с каждым запросом серверу в заголовке Cookie.

После предоставление пользователем данных аутентификации (обычно путем

отправки данных веб-формы), сервер создает сессию, присваивает ей идентификатор и отправляет его в качестве куки. Дальше, при каждом следующем обращении пользователя к веб-приложению сервер проверяет куки, прикрепленные к запросу и проводит аутентификацию и дальнейшую авторизацию пользователей.

Рассмотренные выше схемы аутентификации в Интернете уязвимы к атакам типа «человек посередине» (Man in the middle). В случае с основной (Basic) аутентификацией атака сводится к получению имени пользователя и пароля и повторное их использование при дальнейших запросах. Самый простой способ произвести данную атаку на схему дайджест аутентификации – это подменить ответ сервера, устанавливающий схему авторизации (код 401) таким образом, чтобы заменить схему аутентификации на основную (Basic). Далее все сводится к ситуации, описанной выше. При использовании схемы аутентификации с помощью куки атака заключается в перехвате куки и использовании их атакующем.

Для защиты обмена информацией между клиентом и сервером от прослушивания трафика очень часто используется протокол HTTPS. При использовании этого протокола данные протокола HTTP шифруются. Таким образом, если злоумышленник попытается прослушать трафик, передаваемый между сервером и клиентом максимум, что он сможет увидеть – это только заголовки TCP пакетов. Поэтому перехват трафика напрямую в качестве атаки на схему аутентификации работать не будет. Однако протокол HTTPS (а точнее SSL, который является основой HTTPS) также уязвим к атакам типа «человек посередине». Для проведения этой атаки на протокол HTTPS атакующий должен включить в разрыв свой сервер, который позволит создать две независимые SSL сессии для каждого TCP соединения клиента с сервером: клиент устанавливает SSL соединение со включенным в разрыв сервером, а тот в свою очередь с сервером приложения. Т.е. на стороне атакующего вставленного сервера данные сначала расшифровываются (первое соединение), затем зашифровываются заново и отправляются серверу (второе соединение). Единственное требование – наличие подписанных сертификатов безопасности.

Предложенное решение

Выше было показано, что все основные схемы авторизации в Интернете подвержены атаке «человек посередине», поэтому технически возможно для решения задачи авторизованного подключения использовать метапоисковую систему в качестве «злоумышленника».

В процессе поиска информации важно обеспечить единый доступ пользователя ко всем поисковым ресурсом, т.е. проще говоря, пользователь перед произведением поиска

должен только один раз ввести свои идентификационные данные (логин и пароль) и получить доступ ко всем ресурсам, к которым он имеет право обращаться. Для решения этой задачи на предприятии используется протокол OpenID. Его суть заключается в том, что существует центральный сервер авторизации, к которому обращаются другие информационные системы для аутентификации пользователя (авторизация выполняется на стороне клиентских информационных систем). Данный протокол также подвержен вышеупомянутой атаке.

В итоге общий алгоритм метапоиска с разграничением доступа пользователей к отдельным ИПС системам выглядит следующим образом. Перед началом поиска пользователь вводит поисковый запрос и свои идентификационные данные для аутентификации по OpenID (имя пользователя и пароль), эти данные перехватывает метапоисковая система. Далее метапоисковая система начинает производить поиск по отдельным ИПС, при этом инициируя процесс аутентификации, имитируя поведение браузера пользователя.

Для того, чтобы метапоисковая система смогла перехватывать идентификационные данные пользователей требуется, чтобы все обращения к информационным ресурсам, в которых осуществляется поиск проходили через саму метапоисковую систему. Для этого была предложена и реализована схема, описанная ниже. Схема взаимодействия метапоисковой системы, пользователя и ИПС изображена на рис. 1. Стрелками обозначен порядок возникновения информационных потоков.

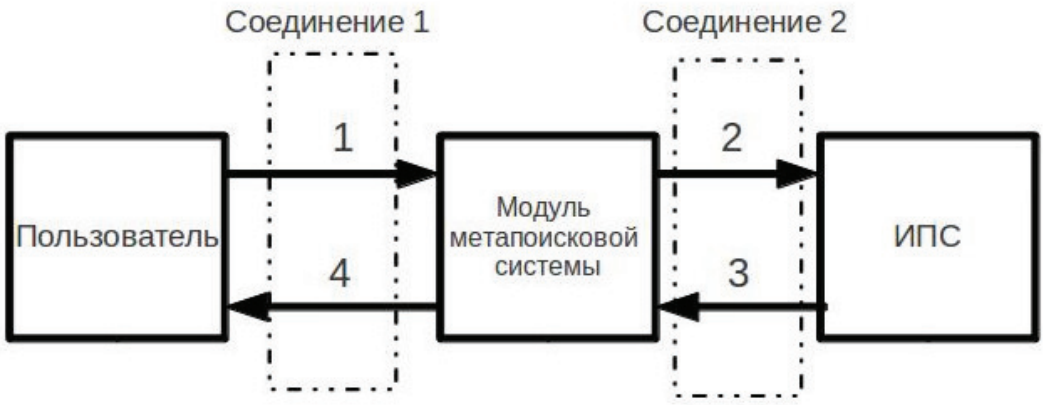

Рис. 1. Схема взаимодействия метапоисковой системы

С точки зрения пользователя все отдельные ИПС имеют URL, относящиеся к одному хосту, т.е. следуют следующему формату: http://<адрес метапоисковой системы>/<компонентная поисковая система>/<относительный путь к документу

компонентной поисковой системы>. Запросы по URL такого вида (стрелка №1 на рисунке) обрабатывает специальный модуль метапоисковой системы. Данный модуль проверяет, есть ли в HTTP-заголовке запроса аутентификационные данные пользователя, и если есть, то сохраняет их. Затем преобразует данный «виртуальный» адрес в фактический URL той информационной системы, к которой обращался пользователь, переадресует HTTP-запрос по этому адресу (стрелка №2 на рисунке) и получает ответ (стрелка №3 на рисунке). Т.е. происходит преобразование адресов вида http://metasearchengine/resource1/path/to/file в адреса вида http://realaddressofresource1/path/to/file. Далее полученный ответ информационной системы отправляется пользователю (стрелка №4 на рисунке), при этом в html коде полученной страницы изменяется для всех гиперссылок формат URL, в случае если тот указывает на документ в информационной системе, которая является компонентной для метапоисковой системы (т.е. является для нее источником информации). Т.е. выполняется преобразование адресов, обратное описанному выше. Данное преобразование выполняется для того, чтобы все http-запросы пользователей к таким системам гарантировано обрабатывались специальным модулем метапоисковой системы. По сути пользователь не должен иметь прямого доступа к компонентным информационным системам – все обращения к ним должны проходить через систему метапоиска.

При передаче данных от пользователя к отдельной ИПС создаются два TCP-соединения – первое между пользователем и модулем метапоисковой системы (соединение №1), второе – между модулем метапоисковой системы и отдельной ИПС (соединение №2). Рассмотрим процесс передачи данных от пользователя к отдельной ИПС. В случае с соединением №1 отправляемые данные будут зашифровываться на стороне пользователя, передаваться в зашифрованном виде по каналу связи, а затем расшифровываться на стороне модуля метапоисковой системы. В случае с соединением №2 данные будут зашифровываться на стороне модуля метапоисковой системы, а расшифровываться на стороне получателя. Поэтому для самой метапоисковой системы данные будут представлены в расшифрованном (оригинальном) виде и даже при использовании протокола HTTPS возможно изменение содержимого передаваемого сообщения с целью преобразования URL. В отличие от большинства реальных атак типа «Man in the Middle» на протокол HTTPS, на «атакующей» (т. е. на стороне метапоисковой системы) устанавливаются сертификаты всех отдельных информационно-поисковых систем, поэтому пользователь даже не заметит того факта того, что с его стороны соединение установлено не с фактическим источником данных. В большинстве случаев злоумышленник не обладает оригинальными сертификатами, о чем пользователя

предупреждает браузер.

Заключение

Предложенный способ обладает рядом достоинств. Во-первых, он не требует изменения компонентных информационных систем. Во-вторых, данный способ создает дополнительную защиту компонентных информационных систем, т.к. пользователи не имеют к ним прямого доступа и могут даже не догадываться о том, что пользуются несколькими информационными системами. В-третьих, система очень хорошо масштабируется – при добавлении в систему нового модуля для организации поиска в защищенных системах, увеличение накладных расходов на обработку данных минимально, т.к. оба модуля полностью независимы друг от друга.

Список литературы

1. К. Маннинг, П.Рагхаван, Х.Шютце. Введение в информационный поиск.: Пер. с англ. - М.: ООО «И.Д. Вильямс», 2011, 528 с.
2. Э. Смит. Аутентификация: от паролей до открытых ключей.: Пер. с англ. – М.: «Вильямс», 202, 432 с.
3. RFC 2617. HTTP Authentication: Basic and Digest Access Authentication.
4. RFC 2109. HTTP State Management Mechanism.
5. Спецификация протокола OpenID 2.0 (OpenID Authentication 2.0 – Final) http://openid.net/specs/openid-authentication-2_0.html (дата обращения 15.09.2013)
6. W. Meng, C. Yu, K. Liu. Building Efficient and Effective Metasearch Engines
   Y. Lu, W. Meng, L. Shu, C. Yu, K. Liu. Evaluation of Result Merging Strategies for Metasearch Engines