Issues of installing HTTPS for two or more virtual hosts at one server

Vladimirov Sergey, 111 group
2005, 23rd of April.

Особенности установки HTTPS для нескольких виртуальных хостов

Владимиров Сергей, 111 группа,
23 апреля 2005 года

Описание проблемы.

При попытке настройки HTTPS для любого WEB-сервера у администраторов не возникает проблем с разборов примеров, которые приведены в инструкциях. Следуя пошаговым инструкциям, они без проблем настраивают WEB-сервера для работы с тестовым или даже одним реальным доменным именем. Однако при попытке подключения к WEB-серверу второго доменного имени, не важно, в качества синонима (alias) или в качестве виртуального хоста (virtual host) стандартные браузеры, в том числе Internet Explorer, отказываются признавать соединение надёжным, ссылаясь на испорченный или неверных сертификат.

Причина возникновения проблемы.

Виртуальные хосты.

Понятие виртуального хоста связано с тем, что один и тот же физический сервер может отвечать на запросы, направленные на разные доменные имена. Например, один и тот же сервер может принимать запросы, если они посланы на www.mipt.ru или www.abitu.ru. Это происходит из-за следующих особенностей настройки:

- Служба доменных имён (Domain Name Service, DNS) утверждает, что оба доменных имени связано с одним и тем же IP-адресом
- На компьютере, который владеет данным IP-адресом, WEB-сервер настроен таким образом, что распределяет запросы, которые начинаются с разных доменных имён, на разные каталоги. Он может это сделать, т.к. в HTTP-запросе содержится полный URL, включая имя хоста.

Добавление защиты.

Защищённым HTTP-соединением называют соединение по протоколу HTTPS. Это является связкой двух протоколов – SSL (Secure Socket Layer) и HTTP. Сначала устанавливается соединение по протоколу SSL, потом, уже по установленному безопасному каналу, идёт обмен данными в стандарте HTTP.
Рассмотрим подробнее процесс установления связи по протоколу SSL.

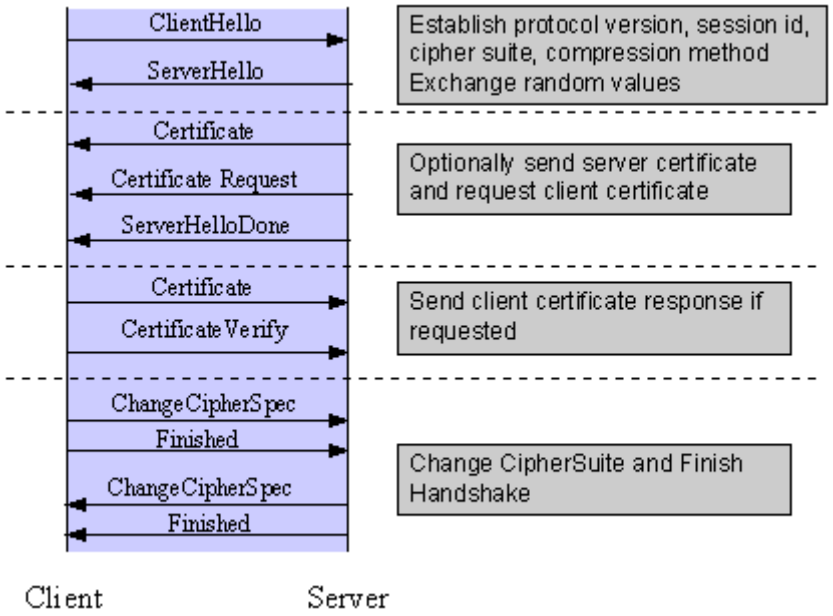

Рисунок 1: Simplified SSL Handshake Sequence

1. Клиент и сервер обмениваются приветствиями, версиями протоколов, случайными числами.
2. Сервер отсылает (опционально, но именно так в большинстве случаев) свой сертификат клиенту. Возможно – хотя редко – запрашивает сертификат клиента.
3. Клиент отсылает, если был запрошен, свой сертификат и удостоверение, что он является владельцем сертификата.
4. На последнем этапе окончательно генерируется сессионный ключ и завершается установление соединения.

Как видно из изложенного выше, клиент не посылает серверу никаких данных, специфичных для HTTP-протокола. Более того, при установлении соединения не известно, какой протокол будет использоваться приложением – это может быть протокол передачи почты, протокол для входа в корпоративную сеть или, например, HTTP. Более того, клиент даже не передаёт серверу имя того сервера, к которому он обратился. Таким образом, у сервера имеется следующая информация о соединении:

- IP-адрес, по которому обратился пользователь.
- порт, на который пользователь отправил запрос
- другие поля, не имеющие значения для рассматриваемого вопроса

Если на сервере установлено несколько виртуальных хостов, основанных на том, что один IP-адрес, может иметь несколько имён, то при обращении пользователя к ним WEB-сервер не может до передачи сертификата узнать, к какому именно из виртуальных хостов обратился пользователь. А значит, для всех виртуальных хостов сервер будет устанавливать соединения, используя один и тот же сертификат.

Причина отказа принятия сертификата браузером.

Рассмотрим подробнее список полей, которые содержатся в сертификате:

- Версия (1, 2 или 3). Сертификат первой версии может содержать только обязательный поля без всяких расширений или дополнительной информации.

уникальный Subject и идентификаторы объекта были добавлены во второй версии. Эти поля позволяют использовать одинаковые имена несколько раз. Однако использовать эти поля не рекомендуется. Расширения были добавлены в третей версии. Они позволяют конкретизировать область применения сертификата и некоторые другие параметры, относящиеся к владельцу сертификата. Цифра в данном поле на единицу меньше фактической версии, т.к. нумерация начинается с нуля.

- Серийный номер – уникальный идентификатор сертификата, выдаваемый центром сертификации (certificate authority).
- Идентификатор алгоритма цифровой подписи, используемый центром сертификации для подписи сертификата.
- Издатель – идентифицирует центр сертификации, который выдал сертификат.
- Subject – информация о владельце сертификата. Если одному и тому же лицу требуется несколько сертификатов, подписанных одним центром сертификации, эти сертификаты могут иметь одно и тоже значение данного поля. Иногда данное поле может быть пустым (например, когда ключ ассоциирован с адресом электронной почты). В этом случае должно быть использовано расширение "subjectAltName".
- Validity – срок действия – начало и окончания периода действия сертификата.
- Subject Public Key Info – открытый ключ владельца – открытый ключ, для которого и был создан сертификат. Также ключ сам содержит идентификатор алгоритма и параметры, с которым его надо использовать.
- Issuer Unique ID и Subject Unique ID. Эти поля уникально идентифицируют издателя и владельца сертификата. Они используются когда поля "Issuer" и "Subject" содержат одинаковые значения, но сертификат не самоподписанный.
- Расширения – набор из одного или более дополнительный полей. Каждое расширение может быть критическим или нет, стандартизованным или специальным.
- Алгоритм подписи – содержит идентификатор криптографического алгоритма, использованный центром сертификации для подписи данного сертификата.
- Значение подписи – подпись центра сертификации для подтверждения правильности информации.

Для протокола HTTPS используется поля Issued To – в него необходимо записать имя хоста, для которого выдан сертификат.

Алгоритм действий браузера и сервера

Из введённого пользователя URL взять имя сервера и номер порта (если порт не указан – использовать стандартный порт для HTTPS соединения)	
Обратиться к DNS серверу за получением IP-адреса	
По данному IP-адресу обратиться на взятый ранее порт. *Прим.: протокол TCP/IP открывает соединение используя цифровой адрес и порт, но не имя хоста и порт, хотя последнее может использоваться как оболочка.*	Сервер открывает соединение и передаёт управление модулю протокола SSL для установления связи.

Сервер и клиент выполняют "рукопожатие" для установления связи:

```
client-hello          C -> S: challenge, cipher_specs
server-hello          S -> C: connection-
                        id,server_certificate,cipher_specs
client-master-key     C -> S: {master_key}server_public_key
client-finish         C -> S: {connection-id}client_write_key
server-verify         S -> C: {challenge}server_write_key
server-finish         S -> C: {new_session_id}server_write_key
```

Это описанный в стандарте алгоритм при отсутствии ранее установленной сессии.
Как видно, уже на втором шаге сервер посылает свой сертификат. Все строчки строго прописаны – длины, возможные значения во всех случаях, константы. В них нет лишнего места, чтобы поместить дополнительную информацию, не жертвуя при этом совместимостью.

Более того, часто соединение по протоколу SSL работает отдельно от сервера – его выполняет специальный модуль, которому уже заранее в качестве параметра передаётся сертификат сервера.

Таким образом, сервер передаёт браузеру сертификат, который, в случае наличия нескольких виртуальных хостов, будет содержать неверное имя хоста в поле Issued To, т.к. на этом IP-адресе и порту "висят" несколько виртуальных хостов.

Браузер получает сертификат, и даже если он верный, правильно подписан и срок действия не закончился и т.д., и т.п., браузер должен его отклонить, т.к. пользователь запросил не то имя хоста, которое указано.	

Варианты решения данной проблемы.

Изменение стандарта.

Одним из вариантов решения проблемы может стать изменения стандарта SSL, введением возможности указания клиентом необходимого сертификата при подключении. Дать возможность клиенту указать, какую именно услугу он хочет получить и в зависимости от этого передавать ему различные сертификаты.
Это потребует изменений в программном обеспечении в большом количестве, что возможно только внутри замкнутой корпоративной сети. Однако, внутри такой сети не представляет сложности получить ещё пару-другую IP-адресов или даже целую подсеть, что будет гораздо проще.

Без изменений стандарта.

Второй вариант требует жертв в удобстве использования браузера, которые, тем не менее, чаще всего будут ложиться на плечи WEB-дизайнера и администратора сервера, а не пользователя.

Основная идея состоит в использовании для каждого виртуального хоста отдельного номера порта, всё ещё оставаясь на одном IP-адресе. То есть каждый виртуальный хост для HTTPS соединений использует свой собственный порт и свой собственный сертификат.

К недостаткам данного способа можно отнести:

- Необходимость пользователю ввести кроме идентификатора протокола "https", имени хоста ещё и номер порта, т.к. он будет отличаться от стандартного.
- Корпоративные маршрутизаторы и фильтры могут не пропускать исходящие соединения на большинство портов, кроме стандартных HTTP и HTTPS.

Второй недостаток относится к серверам, которые размещены в Internet, хотя он не важен в корпоративной сети.

Первый недостаток не так важен по трём причинам:

- Самые частые адреса хранятся в истории браузера либо в закладках, поэтому пользователю набирать адрес не приходится.
- Ссылки на эти хосты уже могут содержать необходимые номера портов. А в печатной литературе HTTPS адреса не используются, т.к. они не предназначены для широкого доступа.
- Если протокол HTTPS используется только для аутентификации клиента только для передачи имени и пароля в момент доступа в закрытую область, и сервер самостоятельно пересылает клиента на адрес, начинающийся с HTTPS (для того, чтобы злоумышленник не подсмотрел пароль), то сервер может и подставить адрес.

Использованная литература:

1. SSL/TLS Strong Encryption: An Introduction, The Apache Software Foundatio, http://www.apache.org/
2. Introduction to X.509 certificates, EldoS Corporation, http://www.eldos.org/docs/cert_intro_common.html
3. RFC 2616 "Hypertext Transfer Protocol - HTTP/1.1", Network Working Group, http://www.faqs.org/rfcs/rfc2616.html
4. SSL2, Kipp E.B. Hickman, The SSL Protocol, 1995. See http://www.netscape.com/eng/security/SSL_2.html.
5. SSL3, Alan O. Freier, Philip Karlton, Paul C. Kocher, The SSL Protocol Version 3.0, 1996. See http://www.netscape.com/eng/ssl3/draft302.txt.
6. TLS1, Tim Dierks, Christopher Allen, The TLS Protocol Version 1.0, 1999. See http://ietf.org/rfc/rfc2246.txt.