

Сервис тестирования корректности настройки SSL на сервере Qualys SSL Labs – SSL Server Test

Агафонова Оксана

8 июня 2015 г.

Содержание

1 Цель работы

Изучить сервис тестирования корректности настройки SSL на сервере Qualys SSL Labs – SSL Server Test

2 Ход работы

2.1 Изучить лучшие практики по развертыванию SSL/TLS

SSL/TLS прост в развертывании, просто работает, не обеспечивая достаточного уровня безопасности. Но основная проблема заключается в том, что SSL/TLS нелегко правильно развернуть. Для того чтобы TLS обеспечивал необходимый уровень безопасности, системные администраторы и разработчики должны приложить дополнительные усилия в настройке своих серверов и в разработке приложений.

1. Приватный ключ и сертификат

Качество защиты, обеспечиваемой TLS полностью зависит от секретного ключа, закладывающего основу безопасности, и сертификата, который сообщает о подлинности сервера для его посетителей.

(a) Используйте 2048-битные закрытые ключи

Используйте 2048-битный RSA или 256-битные ECDSA закрытые ключи для всех ваших серверов. Ключи такой крепости безопасны и будут оставаться безопасными в течение значительного периода времени. Если у вас есть 1024-битные RSA ключи, то следует заменить их более сильными ключами как можно скорее.

(b) Защитите закрытый ключ

Относитесь к закрытым ключам как к важным активам, предоставляя доступ к как можно меньшей группе сотрудников. Рекомендуемые меры:

- Генерируйте закрытые ключи и запросы на сертификат (CSRs) на доверенном компьютере. Некоторые CA предлагают генерацию ключей и CSRs для вас, но это нецелесообразно.
- Используйте парольную защиту закрытых ключей, чтобы предотвратить их компрометацию в тех случаях, когда они хранятся в резервных системах. Парольная защита закрытых ключей не помогает на промышленном сервере, потому что злоумышленник может получить ключи из процесса памяти. Есть аппаратные устройства, которые могут защитить секретные ключи даже в случае компрометации сервера, но они стоят дорого и, таким образом, оправданы только в организациях с высокими требованиями безопасности.
- После компрометации отзывайте старые сертификаты и генерируйте новые ключи.
- Обновляйте сертификаты каждый год и всегда с новыми закрытыми ключами.

(c) Обеспечьте охват всех используемых доменных имен

Убедитесь, что ваши сертификаты охватывают все доменные имена, которые вы хотите использовать на сайте. Например, у вас есть главный домен `www.example.com`,

но вы также используете домен `www.example.net`. Ваша цель — избежать предупреждения о недействительности сертификата, которое будет путать ваших пользователей и ослаблять их доверие.

Даже тогда, когда на сервере настроено только одно доменное имя, нужно иметь в виду, что вы не можете контролировать, как пользователи приходят к вам на сайт или какие ссылки на него указывают. В большинстве случаев, вы должны убедиться, что сертификат работает с и без WWW (например, как для `example.com` и `www.example.com`). Безопасный веб-сервер должен иметь сертификат, действительный для каждого настроенного доменного имени. Сертификаты на весь домен (Wildcard) имеют свое преимущество, но следует избегать их, если их использование означает предоставление закрытого ключа большой группе людей, например, системным администраторам разных организации. Кроме того, имейте в виду, что Wildcard сертификаты могут быть использованы злоумышленниками для передачи уязвимости от одного веб-сайта на все другие сайты, которые используют один и тот же сертификат.

(d) Приобретайте сертификаты у надежного удостоверяющего центра

Выбирайте надежный удостоверяющий центр (CA), который заботиться о своем бизнесе и безопасности. Рассмотрим следующие критерии при выборе CA:

Отношение к безопасности

Все CA проходят регулярный аудит (иначе они не имели бы право работать как CA), но некоторые из них более серьезно относятся к безопасности, чем другие. Выяснить, какие из них лучше в этом отношении нелегко, но один способ заключается в изучении их истории инцидентов безопасности и выявлении того, как они реагировали на компрометации и инциденты безопасности и учились ли они на своих ошибках.

Основное направление деятельности

CA, у которых выпуск сертификатов является основным направлением деятельности, потеряют бизнес, если они сделают что-то ужасно неправильно, и они, вероятно, не будут пренебрегать разделением сертификатов, преследуя потенциально более прибыльные возможности в других местах.

Предлагаемые услуги

Как минимум, выбранный CA должен обеспечивать поддержку списка отозванных сертификатов (CRL) и протокола OCSP.

Инструменты управления сертификатами

Если вам нужно большое количество сертификатов, то выберите центр сертификации, который даст вам хорошие инструменты для управления ими.

Поддержка

Выберите центр сертификации, который предоставляет хорошую поддержку, когда это необходимо.

(e) Используйте надежные алгоритмы подписи сертификата

Безопасность сертификата зависит от длины закрытого ключа и прочности используемой функции хеширования. Сегодня большинство сертификатов используют алгоритм SHA1, который считается слабым.

2. Конфигурация

Если вы правильно настроили на сервере TLS, то можете быть уверены, что данные вашего сайта корректно отображаются для посетителей сайта, используются только безопасные алгоритмы и все известные уязвимости устранены.

(a) Используйте безопасные протоколы

Существует пять версий протоколов в SSL/TLS семейства: SSL v2, SSL v3, TLS v1.0, TLS v1.1 и TLS v1.2. Из них:

- SSL v2 является небезопасным и не должен быть использован.
- SSL v3 является небезопасным при использовании с HTTP и слабым при использовании с другими протоколами. Эта версия также устарела, поэтому она не должна использоваться.
- TLS v1.0 до сих пор является безопасным протоколом. При использовании с HTTP этот протокол обеспечивает безопасность, но только при тщательной конфигурации.
- TLS v1.1 и v1.2 не имеют известных проблем безопасности.

TLS v1.2 должен быть вашим основным протоколом. Эта версия лучше, потому что она поддерживает важные функции, которые недоступны в более ранних версиях. Если ваш сервер (или любое промежуточное устройство) не поддерживает TLS v1.2, то планируйте его модернизацию в ускоренном режиме. Если ваши поставщики услуг не поддерживают TLS v1.2, требуйте, чтобы они модернизировали свою систему.

Для поддержки более старых клиентов вы должны продолжать поддерживать TLS v1.0 и TLS v1.1 еще некоторое время. С некоторыми обходными путями эти протоколы еще можно считать достаточно безопасными для большинства веб-сайтов.

(b) Используйте безопасные алгоритмы шифрования

Для безопасного обмена данными вы должны сначала убедиться, что вы общаетесь непосредственно с нужным абонентом (и не через кого-то, кто будет подслушивать). В SSL и TLS алгоритмы шифрования используются для определения, насколько безопасно происходит обмен данными. Они состоят из различных строительных блоков. Если в одном из строительных блоков наблюдается слабая безопасность, то вы должны быть в состоянии переключиться на другой. Ваша цель — использовать только те алгоритмы шифрования, которые обеспечивают аутентификацию и шифрование в 128 бит или более. Всего остального следует избегать:

- Наборы со слабыми алгоритмами шифрования (как правило, от 40 до 56 бит) могут быть легко взломаны
- RC4 также сейчас считается слабым. Вы должны убрать поддержку этого алгоритма как можно раньше, но только после проверки потенциального негативного воздействия на совместимость.
- 3DES обеспечивает около 112 бит безопасности. Это ниже рекомендованного минимума 128 бит, но это все еще достаточно сильный алгоритм. Большая практическая проблема в том, что 3DES гораздо медленнее, чем альтернативные варианты. Таким образом, мы не рекомендуем его для повышения производительности.

(c) Контроль за выбором алгоритма шифрования

В SSL версии 3 и более поздних версиях протокола, клиенты отправляют список алгоритмов шифрования, которые они поддерживают, и сервер выбирает один из них для организации безопасного канала связи. Не все сервера могут делать это хорошо, так как некоторые выбирают первый поддерживаемый алгоритм из списка. Таким образом, выбор правильного алгоритма шифрования является критически важным для безопасности.

(d) Поддержка Forward Secrecy

Forward Secrecy — это особенность протокола, который обеспечивает безопасный обмен данными, он не зависит от закрытого ключа сервера. С алгоритмами шифрования, которые не поддерживают Forward Secrecy, возможно расшифровать ранее зашифрованные разговоры с помощью закрытого ключа сервера. Нужно поддерживать и предпочитать ECDHE (аббревиатура ECDHE расшифровывается как «эффемерный алгоритм Диффи-Хеллмана с использованием эллиптических кривых») алгоритмы шифрования. Для поддержки более широкого круга клиентов, вы должны также использовать DHE, как запасной вариант после ECDHE.

(e) Отключите Renegotiation по инициативе клиента

В SSL / TLS renegotiation позволяет сторонам остановить обмен данными, с тем чтобы повторно инициировать его для обеспечения безопасности. Есть некоторые случаи, в которых renegotiation должен быть инициирован сервером, но нет никакой известной необходимости позволять инициировать renegotiation клиентом. Кроме того это может облегчить организацию DDoS-атаки на ваши сервера.

(f) Снижение известных проблем

В какой-то момент могут возникнуть проблемы с безопасностью с любым продуктом. Хорошо, если вы всегда в курсе событий в мире информационной безопасности. По крайней мере, вы должны следить за релизами безопасности продуктов, которые используете, и устанавливать их, как только они становятся доступными.

Следите за тем, что происходит в мире безопасности и адаптируйтесь к ситуации, когда это необходимо. По крайней мере, вы должны сразу устанавливать патчи, закрывающие обнаруженные уязвимости, как только они становятся доступными. Обратите внимание на следующие вопросы:

— *Отключите TLS compression*

В 2012 году CRIME attack показал, как TLS сжатие может быть использовано злоумышленниками для выявления деталей конфиденциальных данных (например, сессионные куки). Очень немногие клиенты поддерживали TLS сжатие тогда (и в настоящее время), так что маловероятно, что вы будете испытывать какие-либо проблемы с производительностью после отключения TLS сжатия на серверах.

— *Отключите RC4*

Алгоритм RC4 является небезопасным и должен быть отключен. В настоящее время мы знаем, что для взлома RC4 требуются миллионы запросов, много пропускной способности и времени. Таким образом, риск все еще относительно невелик, но вполне возможно, что атаки будут масштабнее в будущем. Перед снятием RC4

проверьте, будут ли ваши существующие пользователи затронуты; другими словами, проверить, если у вас есть клиенты, которые поддерживают только RC4.

— *Будьте в курсе атаки BEAST*

Успешная атака BEAST похожа на взлом сессии. К сожалению, для смягчения угрозы со стороны сервера требуется RC4, который больше не рекомендуется. Из-за этого, а также из-за того что атака BEAST теперь в значительной степени уменьшается на стороне клиента, мы больше не рекомендуем смягчения на сервере путем использования RC4. В некоторых ситуациях, когда есть большое количество старых клиентов, уязвимых для атаки BEAST, более безопасно использовать RC4 с TLS 1.0 и более ранние версии протокола. Принимать это решение следует осторожно и только после полного понимания окружающей среды и модели ее угроз.

— *Отключить SSL v3*

SSL v3 уязвим против POODLE атаки, которая была обнаружена в октябре 2014. Лучший способ устранения уязвимости POODLE атаки — это отключить SSL v3, который в большинстве сайтов можно сделать безопасно.

2.2 Изучить основные уязвимости и атаки на SSL последнего времени — POODLE, HeartBleed

POODLE

Вводная информация

14 октября 2014 года в протоколе шифрования SSL версии 3 была выявлена уязвимость. Эта уязвимость, названная POODLE (Padding Oracle On Downgraded Legacy Encryption), позволяет злоумышленнику читать информацию, зашифрованную этой версией протокола, используя атаку man-in-the-middle. Также SSLv3 является очень старой версией протокола, но тем не менее многие приложения поддерживают его и используют SSLv3 в случаях, когда недоступны другие более новые и лучшие варианты шифрования. Что важно, злоумышленник может нарочно требовать использования только SSLv3 на обеих сторонах соединения. Уязвимости POODLE подвержены любые сервисы или клиенты, которые могут соединяться, используя SSLv3.

Что такое уязвимость POODLE?

Этой уязвимости подвержен SSL-протокол версии 3, который позволяет перехватить содержимое зашифрованное с помощью SSLv3.

Кто подвержен этой уязвимости?

Этой уязвимости подвержено любое программное обеспечение, использующее для шифрования соединения SSLv3. Это веб-браузеры, веб-серверы, почтовые серверы и тому подобное.

Как это работает?

Если кратко, то уязвимость POODLE присутствует, потому что протокол SSLv3 некорректно проверяет содержимое, пересылаемое в зашифрованном виде. Благодаря этому не происходит верификации со стороны получателя и атакующий может подменять данные и передавать к месту получения. При определенных условиях модифицированные данные могут быть приняты получателем без каких-либо предупреждений. В среднем, каждый 256-й запрос будет принят получателем и позволит злоумышленнику расшифровать один байт. Это может быть повторено нужное количество раз. Любой злоумышленник, участвуя таким образом в пересылке данных с помощью этого протокола, сможет получить ключ к расшифровке данных за очень короткое время.

Как защититься?

Должны быть предприняты действия, которые не позволят использовать SSLv3 ни в случае использования клиентских приложений, ни в случае серверных. И серверы и клиенты должны отключить полностью поддержку SSLv3.

HeartBleed

Heartbleed — ошибка (переполнение буфера) в криптографическом программном обеспечении OpenSSL, позволяющая несанкционированно читать память на сервере или на клиенте, в том числе для извлечения закрытого ключа сервера. Информация об уязвимости была опубликована в апреле 2014 года, ошибка существовала с конца 2011 года.

Основная особенность уязвимости заключается в том, что атакующий может прочитать определенный диапазон адресов памяти (длиной 64KB) в процессе на сервере, который использует эту библиотеку. Используя эту уязвимость злоумышленники путем отправки специальным образом сформированного запроса могут:

Украсть пароли/логины от ваших сервисов. Получить доступ к конфиденциальным cookie. Украсть приватный SSL/TLS ключ сервера, с которым вы работаете по HTTPS (скомпрометировать HTTPS). Украсть любую секретную информацию, которую защищает HTTPS (прочитать письма, сообщения на сервере и т. д.).

Злоумышленники могут скомпрометировать HTTPS и позднее (через известную атаку типа MitM), имея у себя на руках закрытый SSL/TLS-ключ (представиться сервером).

В качестве примера успешной эксплуатации можно привести Yahoo, которая была подвержена этой уязвимости. С использованием Heartbleed можно было быстро получить доступ к логинам и паролям пользователей в открытом виде. [уязвимость исправлена и сертификат для HTTPS был перевыпущен].

2.3 Практическое задание

Выбрать со стартовой страницы SSL Server Test один домен из списка Recent Best и один домен из списка Recent Worst – изучить отчеты, интерпретировать результаты в разделе Summary

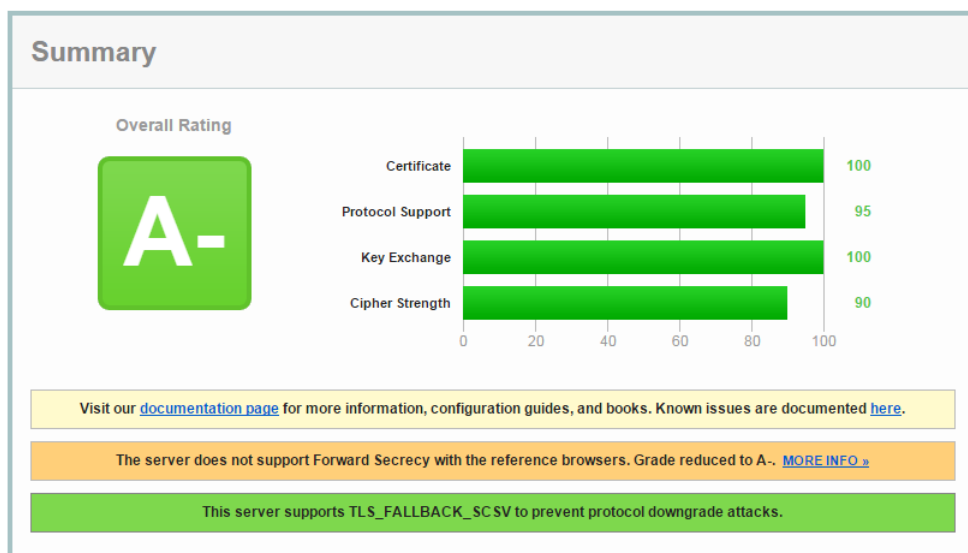


Рис. 1: SSL Report: www2.mizunousa.com (50.205.43.98)

Summary

Сервер должен обеспечить гарантии того, что сессионные ключи, полученные при помощи набора ключей долговременного пользования, не будут скомпрометированы при компрометации одного из долговременных ключей.

Этот сервер поддерживает TLS-FALLBACK-SCSV, чтобы предотвратить возврат к предыдущей версии протокола атаки. Это механизм, который решает проблемы, вызванные неудачными повторными соединениями, и тем самым закрывает возможность злоумышленникам инициировать использование SSL 3.0. Это также предотвращает понижение с TLS 1.2 до 1.1 или 1.0 и тем самым может помочь предотвратить будущие атаки.

Configuration

Configuration		
	Protocols	
	TLS 1.2	Yes
	TLS 1.1	No
	TLS 1.0	Yes
	SSL 3	No
	SSL 2	No
	Cipher Suites (SSL 3+ suites in server-preferred order; deprecated and SSL 2 suites always at the end)	
	TLS_RSA_WITH_AES_256_GCM_SHA384 (0x9d)	256
	TLS_RSA_WITH_AES_128_GCM_SHA256 (0x9c)	128
	TLS_RSA_WITH_AES_256_CBC_SHA256 (0x3d)	256
	TLS_RSA_WITH_AES_256_CBC_SHA (0x35)	256
	TLS_RSA_WITH_AES_128_CBC_SHA256 (0x3c)	128
	TLS_RSA_WITH_AES_128_CBC_SHA (0x2f)	128
	TLS_RSA_WITH_3DES_EDE_CBC_SHA (0xa)	112

Рис. 2: Configuration

TLS - Transport Layer Security. Транспортный уровень безопасности

SSL - Secure Sockets Layer. Протокол защищенных сокетов (протокол, гарантирующий безопасную передачу данных по сети; комбинирует криптографическую систему с открытым ключом и блочное шифрование данных)

RSA - аббревиатура от фамилий Rivest, Shamir и Adleman.

RC4 - Rivest cipher Ron's code 4. Поточковый шифр, широко применяющийся в различных системах защиты информации в компьютерных сетях (например, в протоколах SSL и TLS, алгоритмах обеспечения безопасности беспроводных сетей WEP и WPA).

SHA - Secure Hash Algorithm. Безопасный алгоритм хэширования.

AES - Advanced Encryption Standard.

CBC - Cipher Block Chaining. Улучшенный стандарт шифрования.

3DES - Triple Data Encryption Standard. Утроенный стандарт шифрования данных.

SNI - Server Name Indication. Указание названия сервера


NPN - Next Protocol Negotiation. Протокол согласования.

HSTS - HTTP Strict Transport Security. механизм, активирующий форсированное защищённое соединение по HTTPS. Данная политика безопасности позволяет сразу же устанавливать безопасное соединение, вместо использования HTTP.

HPKP - HTTP Public Key Pinning. HTTP-расширение для механизма привязки открытых ключей

HTTP - HyperText Transfer Protocol. Протокол прикладного уровня передачи данных

Прокомментировать большинство позиций в разделе Protocol Details



Protocol Details	
Secure Renegotiation	Supported
Secure Client-Initiated Renegotiation	No
Insecure Client-Initiated Renegotiation	No
BEAST attack	Not mitigated server-side (more info) TLS 1.0: 0x35
POODLE (SSLv3)	No, SSL 3 not supported (more info)
POODLE (TLS)	No (more info)
Downgrade attack prevention	Yes, TLS_FALLBACK_SCSV supported (more info)
TLS compression	No
RC4	No
Heartbeat (extension)	No
Heartbleed (vulnerability)	No (more info)
OpenSSL CCS vuln. (CVE-2014-0224)	No (more info)
Forward Secrecy	No WEAK (more info)
Next Protocol Negotiation (NPN)	No
Session resumption (caching)	Yes
Session resumption (tickets)	No
OCSP stapling	No
Strict Transport Security (HSTS)	Disabled max-age=0
Public Key Pinning (HPKP)	No
Long handshake intolerance	No
TLS extension intolerance	No
TLS version intolerance	TLS 1.98 TLS 2.98
Incorrect SNI alerts	-

Рис. 3: Protocol Details

Secure Renegotiation - возобновление подключения TLS.

BEAST attack - атака утилитой BEAST (Browser Exploit Against SSL/TLS).

POODLE - уязвимость, позволяющая расшифровать содержимое защищённого канала коммуникации.

Downgrade attack - атака, при которой пользователя вынуждают использовать менее безопасные протоколы, которые всё ещё поддерживаются из соображений совместимости.

TLS compression - В 2012 году CRIME attack показал, как TLS сжатие может быть использовано злоумышленниками для выявления деталей конфиденциальных данных (например, сессионные куки).

Heartbleed - ошибка в OpenSSL, позволяющая несанкционированно читать память на сервере до 64 килобайт за один запрос. Атаку можно производить бесконечное количество раз.

Forward Secrecy - особенность протокола, который обеспечивает безопасный обмен данными, он не зависит от закрытого ключа сервера. С алгоритмами шифрования, которые не поддерживают Forward Secrecy, возможно расшифровать ранее зашифрованные разговоры с помощью закрытого ключа сервера.

Next Protocol Negotiation - клиент сообщает серверу по каким протоколам он бы хотел общаться и сервер может ответить наиболее предпочтительным из тех, которые он знает.

Strict Transport Security - механизм, активирующий форсированное защищённое со-

единение по HTTPS. Данная политика безопасности позволяет сразу же устанавливать безопасное соединение, вместо использования HTTP. Механизм использует особый заголовок HTTP Strict-Transport-Security, для переключения пользователя, зашедшего по HTTP, на HTTPS-сервер.

2.4 Выводы

Сервер использует доверенный сертификат и защищен от следующих атак: heart-bleed, downgrade, beast, и не содержит уязвимость pooodle. Кроме того, сервер не поддерживает forward secresu для всех браузеров, что является дополнительной угрозой. В качестве итога, можно сказать, что в случае необходимости специалист способен нанести значительный урон данному сервису.