16. Шифрование

Обзор

Zabbix поддерживает шифрование соединений между Zabbix сервером, Zabbix прокси, Zabbix агентом, zabbix_sender и zabbix_get утилитами с использованием Transport Layer Security (TLS) протокола v.1.2. Шифрование поддерживается начиная с Zabbix 3.0. Поддерживаются шифрования на основе сертификата и на основе pre-shared ключа.

Шифрование опционально и настраивается для отдельных компонентов (например, некоторые прокси и агенты можно настроить на использование шифрования с сервером на основе сертификатов, в то время как другие могут использовать шифрование на основе pre-shared ключа, а остальные могут продолжать использовать незашифрованные соединения как и ранее).

Сервер (прокси) может использовать различные настройки с разными узлами сети.

Программы Zabbix демонов слушают один порт для шифрованных и незашифрованных входящих подключений. Добавление шифрования не потребует открывать новые порты на брандмауэрах.

Ограничения

- Приватные ключи хранятся в формате обычного текста в файлах, которые Zabbix компоненты считывают в процессе запуска.
- Введенные pre-shared ключи в веб-интерфейсе Zabbix хранятся в базе данных Zabbix в виде обычного текста.
- Встроенное шифрование не защищает коммуникации:
  - между веб-сервером с веб-интерфейсом Zabbix и веб-браузером на строне пользователя,
  - между Zabbix веб-интерфейсом и Zabbix сервером,
  - между Zabbix сервером (прокси) и базой данных Zabbix.
- В настоящее время каждое незашифрованное соединение открывается с полными TLS переговорами, кэширование сессий и билеты не реализованы.
- Добавление шифрования увеличивает время проверок и действий, в зависимости от сетевых задержек.
  Например, если пакет опаздывает на 100мс, тогда открытие TCP соединение и отправка незашифрованного запроса займет около 200мс.
  При наличии шифрования на установку TLS соединения добавится около 1000 мс.
  Возможно потребуется увеличить время ожидания, в противном случае некоторые элементы данных и действия, выполняющие удаленные скрипты на агентах смогут работать с незашифрованными соединениями, но не смогут при шифрованном соединении (будет превышено время ожидания).
- Шифрование не поддерживается сетевым обнаружением. Zabbix агент проверки выполняемые сетевым обнаружением будут незашифрованными и, если Zabbix агент настроен на отклонение незашифрованных соединений, то такие проверки не будут успешными.

Компиляция Zabbix с поддержкой шифрования

Для поддержки шифрования Zabbix должен быть скомпилировать и связан с по крайней мере одной из четырёх крипто библиотекой:

- *GnuTLS* (с версии 3.1.18)
- *OpenSSL* (версии 1.0.1, 1.0.2, 1.1.0). *OpenSSL* 1.1.1 поддерживается начиная с версии Zabbix 3.0.23.
- *LibreSSL* (протестировано с версиями 2.7.4, 2.8.2) поддерживается с Zabbix версии 3.0.26. *LibreSSL* 2.6.x не поддерживается. *LibreSSL* поддерживается как совместимая замена *OpenSSL*, новые API функции `tls_*()` специфичные для *LibreSSL* не используются. Zabbix компоненты скомпилированные с *LibreSSL* не будут поддерживать PSK, можно использовать только сертификаты.
- *mbed TLS* (ранее *PolarSSL*)(версия 1.3.9 и выше 1.3.x). *mbed TLS* 2.x в настоящее время не поддерживается, это не простая замена ветки 1.3, Zabbix не скомпилируется с *mbed TLS* 2.x.

Библиотека выбирается при помощи опции в скрипте “configure”:

- `--with-gnutls[=DIR]`
- `--with-openssl[=DIR]` (также используется и для *LibreSSL*)
- `--with-mbedtls[=DIR]`

Например, чтобы сконфигурировать исходные коды сервера и агента с *OpenSSL*, вы можете использовать что-то вроде:

```
./configure --enable-server --enable-agent --with-mysql --enable-ipv6 --
with-net-snmp --with-libcurl --with-libxml2 --with-openssl
```

Можно скомпилировать разные компоненты Zabbix с различными крипто библиотеками (например, сервер с *OpenSSL*, агент с *GnuTLS*).

В наших тестах *OpenSSL* был самым быстрым, далее *GnuTLS*.

Если вы планируете использовать pre-shared ключи (PSK) рассмотрите возможность использования библиотек *GnuTLS*, более новую *OpenSSL* (начиная с 1.1.0) или *mbed TLS* с компонентами Zabbix, использующие PSK. Эти библиотеки поддерживают наборы шифров PSK с Совершенной прямой секретностью (Perfect forward secrecy). Более старые версии *OpenSSL* библиотеки (версии 1.0.1, 1.0.2c) поддерживают PSK, но доступные наборы шифров PSK не обеспечивают Совершенную прямую секретность.

Управление зашированными соединениями

Соединения в Zabbix могут использовать:

- без шифрования (по умолчанию)
- RSA шифрование на основе сертификатов
- шифрование на основе PSK

Имеется два важных параметра, которые используются, чтобы указать шифрование между компонентами Zabbix:

- TLSConnect
- TLSAccept

TLSConnect задает какое использовать шифрование и может принимать одно из 3 значений (unencrypted, PSK, certificate). TLSConnect используется в файлах конфигурации Zabbix прокси (в активном режиме задает только подключения к серверу) и Zabbix agentd (при активных проверках). В веб-интерфейсе Zabbix параметр TLSConnect является эквивалентом поля *Подключения к узлу сети* с вкладки *Настройка→Узлы сети→<какой-то узел сети>→Шифрование* и поля *Подключения к прокси* с вкладки *Администрирование→Прокси→<какой-то прокси>→Шифрование*. Если настроенный тип шифрования для соединения завершится неудачей, другие типы шифрования не будут опробованы.

TLSAccept задает какой тип соединений разрешен при входящих подключениях. Тип подключений: unencrypted, PSK, certificate. Можно указать одно или более значений. TLSAccept используется в файлах конфигурации Zabbix прокси (в пассивном режиме задает только соединения с сервера) и Zabbix agentd (при пассивных проверках). В веб-интерфейсе Zabbix параметр TLSAccept является эквивалентом поля *Соединения с узла сети* с вкладки *Настройка→Узлы сети→<какой-то узел сети>→Шифрование* и поля “Соединения с прокси” с вкладки *Администрирование→Прокси→<какой-то прокси>→Шифрование*.

Как правило, вы настраиваете только один тип шифрования для входящих подключений. Но вы можете захотите переключить режим шифрования, например с незашированного на основанный на сертификатах с минимальным временем простоя и с возможностью отката. Для этого вы можете задать TLSAccept=unencrypted,cert в файле конфигурации agentd и перезапустить агента Zabbix.
Затем вы можете протестировать подключение от zabbix_get к агенту, используя сертификат. Если подключение работает, вы можете перенастроить шифрование у этого агента в Zabbix веб-интерфейсе на вкладке *Настройка→Узлы сети→<какой-то узел сети>→Шифрование*, переключив настройку *Подключения к узлу сети* на “Сертификат”.
Когда кэш конфигурации сервера обновится (и конфигурация прокси обновится, если узел сети наблюдается через прокси), тогда подключения к этому агенту будут зашифрованы.
Если всё работает как ожидается, вы можете задать TLSAccept=cert в файле конфигурации агента и перезапустить Zabbix агента.
Теперь агент будет принимать только зашифрованные подключения на основе сертификатов. Незашифрованные и основанные на PSK подключения будут отклонены.

Шифрование на сервере и прокси работает аналогичным образом. Если в веб-интерфейсе Zabbix в настройке узла сети *Соединения с узла сети* задано равным “Сертификат”, тогда от агента (активные проверки) и zabbix_sender (траппер элементы данных) будут приниматься только зашифрованные соединения на основе сертификатов.

Скорее всего вы настроите входящие и исходящие соединения на использование одного типа шифрования или без шифрования вовсе. Но, технически, имеется возможность настроить шифрование асимметрично, например, шифрование на основе сертификатов для входящих подключений и на основе PSK для исходящих подключений.

Обзорные настройки шифрования отображаются в веб-интерфейсе Zabbix *Настройка→Узлы сети* по каждому узлу сети по правой стороне, в колонке *ШИФРОВАНИЕ АГЕНТА*. Примеры отображения настроек:

Пример	Подключения К узлу сети	Разрешенные подключения ОТ узла сети	Отклоненные подключения С узла сети
NONE	Незашифровано	Незашифровано	Зашифровано на основе сертификата и PSK
CERT NONE PSK CERT	Зашифровано, на основе сертификата	Зашифровано, на основе сертификата	Незашифровано и на основе PSK
PSK NONE PSK CERT	Зашифровано на основе PSK	Зашифровано на основе PSK	Незашифровано и на основе сертификата
PSK NONE PSK CERT	Зашифровано на основе PSK	Незашифровано и зашифровано на основе PSK	На основе сертификата
CERT NONE PSK CERT	Зашифровано на основе сертификата	Незашифровано на основе PSK или зашифровано на основе сертификата	-

По умолчанию используются незашифрованные подключения. Шифрование необходимо настраивать по каждому узлу сети и прокси отдельно.

zabbix_get и zabbix_sender с шифрованием

Смотрите страницы помощи zabbix_get и zabbix_sender по использованию этих утилит при наличии шифрования.

Алгоритмы шифрования

Алгоритмы конфигурируются внутри в процессе запуска Zabbix и зависят от крипто библиотеки, в настоящее время алгоритмы нельзя настраивать пользователями.

Настроенные алгоритмы шифрования по типу библитеки с более высокого уровня к низкому уровню:

Библиотека	Алгоритмы шифрования сертификатов	Алгоритмы шифрования PSK
mbed TLS (PolarSSL) 1.3.9	TLS-ECDHE-RSA-WITH-AES-128-GCM-SHA256 TLS-ECDHE-RSA-WITH-AES-128-CBC-SHA256 TLS-ECDHE-RSA-WITH-AES-128-CBC-SHA TLS-RSA-WITH-AES-128-GCM-SHA256 TLS-RSA-WITH-AES-128-CBC-SHA256 TLS-RSA-WITH-AES-128-CBC-SHA	TLS-ECDHE-PSK-WITH-AES-128-CBC-SHA256 TLS-ECDHE-PSK-WITH-AES-128-CBC-SHA TLS-PSK-WITH-AES-128-GCM-SHA256 TLS-PSK-WITH-AES-128-CBC-SHA256 TLS-PSK-WITH-AES-128-CBC-SHA
GnuTLS 3.1.18	TLS_ECDHE_RSA_AES_128_GCM_SHA256 TLS_ECDHE_RSA_AES_128_CBC_SHA256 TLS_ECDHE_RSA_AES_128_CBC_SHA1 TLS_RSA_AES_128_GCM_SHA256 TLS_RSA_AES_128_CBC_SHA256 TLS_RSA_AES_128_CBC_SHA1	TLS_ECDHE_PSK_AES_128_CBC_SHA256 TLS_ECDHE_PSK_AES_128_CBC_SHA1 TLS_PSK_AES_128_GCM_SHA256 TLS_PSK_AES_128_CBC_SHA256 TLS_PSK_AES_128_CBC_SHA1

Библиотека	Алгоритмы шифрования сертификатов	Алгоритмы шифрования PSK
OpenSSL 1.0.2c	ECDHE-RSA-AES128-GCM-SHA256 ECDHE-RSA-AES128-SHA256 ECDHE-RSA-AES128-SHA AES128-GCM-SHA256 AES128-SHA256 AES128-SHA	PSK-AES128-CBC-SHA
OpenSSL 1.1.0	ECDHE-RSA-AES128-GCM-SHA256 ECDHE-RSA-AES128-SHA256 ECDHE-RSA-AES128-SHA AES128-GCM-SHA256 AES128-CCM8 AES128-CCM AES128-SHA256 AES128-SHA	ECDHE-PSK-AES128-CBC-SHA256 ECDHE-PSK-AES128-CBC-SHA PSK-AES128-GCM-SHA256 PSK-AES128-CCM8 PSK-AES128-CCM PSK-AES128-CBC-SHA256 PSK-AES128-CBC-SHA

Алгоритмы шифрования при использовании сертификатов:

	TLS сервер		
TLS клиент	*mbed TLS (PolarSSL)*	*GnuTLS*	*OpenSSL 1.0.2*
mbed TLS (PolarSSL)	TLS-ECDHE-RSA-WITH-AES-128-GCM-SHA256	TLS-ECDHE-RSA-WITH-AES-128-GCM-SHA256	TLS-ECDHE-RSA-WITH-AES-128-GCM-SHA256
GnuTLS	TLS-ECDHE-RSA-WITH-AES-128-GCM-SHA256	TLS-ECDHE-RSA-WITH-AES-128-GCM-SHA256	TLS-ECDHE-RSA-WITH-AES-128-GCM-SHA256
OpenSSL 1.0.2	TLS-ECDHE-RSA-WITH-AES-128-GCM-SHA256	TLS-ECDHE-RSA-WITH-AES-128-GCM-SHA256	TLS-ECDHE-RSA-WITH-AES-128-GCM-SHA256

Алгоритмы шифрования при использовании PSK:

	TLS сервер		
TLS клиент	*mbed TLS (PolarSSL)*	*GnuTLS*	*OpenSSL 1.0.2*
mbed TLS (PolarSSL)	TLS-ECDHE-PSK-WITH-AES-128-CBC-SHA256	TLS-ECDHE-PSK-WITH-AES-128-CBC-SHA256	TLS-PSK-WITH-AES-128-CBC-SHA
GnuTLS	TLS-ECDHE-PSK-WITH-AES-128-CBC-SHA256	TLS-ECDHE-PSK-WITH-AES-128-CBC-SHA256	TLS-PSK-WITH-AES-128-CBC-SHA
OpenSSL 1.0.2	TLS-PSK-WITH-AES-128-CBC-SHA	TLS-PSK-WITH-AES-128-CBC-SHA	TLS-PSK-WITH-AES-128-CBC-SHA