1 Использование сертификатов

Обзор

Zabbix может использовать RSA сертификаты в формате PEM, подписанные публичным или внутренним центром сертификации (CA). Проверка сертификата выполняется в отношении с заранее подготовленным CA сертификатом. Опционально можно использовать списки отзывов сертификатов (CRL). Каждый компонент Zabbix может иметь только один настроенный сертификат.

Для получения более подробной информации о том как настроить и управлять внутренним CA, как генерировать запросы на сертификаты и подписывать их, как отзывать сертификаты, всё это вы можете найти в большом количестве различных руководств в сети, например, OpenSSL PKI Tutorial v1.1.

Тщательно продумывайте и тестируйте ваши расширения сертификатов - смотри Ограничения при использовании расширений X.509 v3 сертификатов.

Параметры настройки сертификатов

Параметр	Обязателен	Описание
TLSCAFile	*	Абсолютный путь к файлу, который содержит сертификаты верхнего уровня CA(и) для верификации сертификата узла. При наличии цепочки сертификатов с несколькими членами, они должны быть отсортированы: сначала следуют сертификаты CA низкого уровня за сертификатами более высокого уровня CA(и). Сертификаты из нескольких CA(и) можно включать в один файл.
TLSCRLFile		Абсолютный путь к файлу, который содержит списки отозванных сертификатов. Смотрите заметки в Списки отозванных сертификатов (CRL).
TLSCertFile	*	Абсолютный путь к файлу, который содержит сертификат (цепочку сертификатов). В случае цепочки сертификатов с несколькими членами они должны быть отсортированы: сначала сервер, прокси или агент, с последующими CA сертификатами низкого уровня и затем CA сертификаты более высокого уровня.
TLSKeyFile	*	Абсолютный путь к файлу, который содержит приватный ключ. Задайте права доступа к этому файлу - он должен быть доступен для чтения только пользователю Zabbix.
TLSServerCertIssuer		Разрешенный эмитент сертификата сервера.
TLSServerCertSubject		Разрешенный субъект сертификата сервера.

Настройка сертификата на Zabbix сервере

1. Для того, чтобы проверять сертификаты хостов, Zabbix сервер должен иметь доступ к файлу с их корневыми верхнего уровня самоподписными CA сертификатами. Например, если мы ожидаем сертификаты от двух независимых корневых CA, мы можем поместить их сертификаты в файл `/home/zabbix/zabbix_ca_file`, примерно следующим образом:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
    Signature Algorithm: sha1WithRSAEncryption
        Issuer: DC=com, DC=zabbix, O=Zabbix SIA, OU=Development group,
CN=Root1 CA
            ...
        Subject: DC=com, DC=zabbix, O=Zabbix SIA, OU=Development group,
CN=Root1 CA
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
            ...
        X509v3 extensions:
            X509v3 Key Usage: critical
                Certificate Sign, CRL Sign
            X509v3 Basic Constraints: critical
                CA:TRUE
            ...
-----BEGIN CERTIFICATE-----
MIID2jCCAsKgAwIBAgIBATANBgkqhkiG9w0BAQUFADB+MRMwEQYKCZImiZPyLGQB
....
9wEzdN8uTrqoyU78gi12npLj08LegRKjb5hFTVmO
-----END CERTIFICATE-----
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
    Signature Algorithm: sha1WithRSAEncryption
        Issuer: DC=com, DC=zabbix, O=Zabbix SIA, OU=Development group,
CN=Root2 CA
            ...
        Subject: DC=com, DC=zabbix, O=Zabbix SIA, OU=Development group,
CN=Root2 CA
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
            ....
        X509v3 extensions:
            X509v3 Key Usage: critical
                Certificate Sign, CRL Sign
            X509v3 Basic Constraints: critical
                CA:TRUE
            ....
-----BEGIN CERTIFICATE-----
MIID3DCCAsSgAwIBAgIBATANBgkqhkiG9w0BAQUFADB/MRMwEQYKCZImiZPyLGQB
...
vdGNYoSfvu41GQAR5Vj5FnRJRzv5XQOZ3B6894GY1zY=
```

```
-----END CERTIFICATE-----
```

2. Поместите цепочку сертификатов Zabbix сервера в файл, например, `/home/zabbix/zabbix_server.crt`:

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
    Signature Algorithm: sha1WithRSAEncryption
        Issuer: DC=com, DC=zabbix, O=Zabbix SIA, OU=Development group,
CN=Signing CA
        ...
        Subject: DC=com, DC=zabbix, O=Zabbix SIA, OU=Development group,
CN=Zabbix server
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                ...
        X509v3 extensions:
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Basic Constraints:
                CA:FALSE
            ...
-----BEGIN CERTIFICATE-----
MIIECDCCAvCgAwIBAgIBATANBgkqhkiG9w0BAQUFADCBgTETMBEGCgmSJomT8ixk
...
h02u1GHiy46GI+xfR3LsPwFKlkTaaLaL/6aaoQ==
-----END CERTIFICATE-----
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 2 (0x2)
    Signature Algorithm: sha1WithRSAEncryption
        Issuer: DC=com, DC=zabbix, O=Zabbix SIA, OU=Development group,
CN=Root1 CA
        ...
        Subject: DC=com, DC=zabbix, O=Zabbix SIA, OU=Development group,
CN=Signing CA
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
            ...
        X509v3 extensions:
            X509v3 Key Usage: critical
                Certificate Sign, CRL Sign
            X509v3 Basic Constraints: critical
                CA:TRUE, pathlen:0
        ...
-----BEGIN CERTIFICATE-----
```

```
MIID4TCCAsmgAwIBAgIBAjANBgkqhkiG9w0BAQUFADB+MRMwEQYKCZImiZPyLGQB
...
dyCeWnvL7u5sd6ffo8iRny0QzbHKmQt/wUtcVIvWXdMIFJM0Hw==
-----END CERTIFICATE-----
```

Здесь первым является сертификат Zabbix сервера, за ним промежуточные CA сертификат.

3. Поместите приватный ключ Zabbix сервера в файл, например, `/home/zabbix/zabbix_server.key`:

```
-----BEGIN PRIVATE KEY-----
MIIEwAIBADANBgkqhkiG9w0BAQEFAASCBKowggSmAgEAAoIBAQC9tIXIJoVnNXDl
...
IJLkhbybBYEf47MLhffWa7XvZTY=
-----END PRIVATE KEY-----
```

4. Измените параметры TLS в файле конфигурации Zabbix сервера, примерно так:

```
TLSCAFile=/home/zabbix/zabbix_ca_file
TLSCertFile=/home/zabbix/zabbix_server.crt
TLSKeyFile=/home/zabbix/zabbix_server.key
```

Настройка шифрования для Zabbix прокси на основе сертификата

1. Подготовьте файлы с CA сертификатами верхнего уровня, сертификатом (цепочкой) прокси и приватным ключем, как описано в Настройке сертификата на Zabbix сервере. Измените параметры `TLSCAFile`, `TLSCertFile`, `TLSKeyFile` в файле конфигурации прокси соответственно.

2. При активном прокси измените `TLSConnect` параметр:

```
TLSConnect=cert
```

При пассивном прокси измените `TLSAccept` параметр:

```
TLSAccept=cert
```

3. Теперь у вас есть минимальная настройка прокси на основе сертификата. Вы возможно захотите улучшить безопасность прокси, указав параметры `TLSServerCertIssuer` и `TLSServerCertSubject` (смотри Ограничение разрешенных Эмитента и Субъекта сертификата).

4. В конечном итоге параметры TLS в файле конфигурации прокси могут выглядеть следующим образом:

```
TLSConnect=cert
TLSAccept=cert
TLSCAFile=/home/zabbix/zabbix_ca_file
```

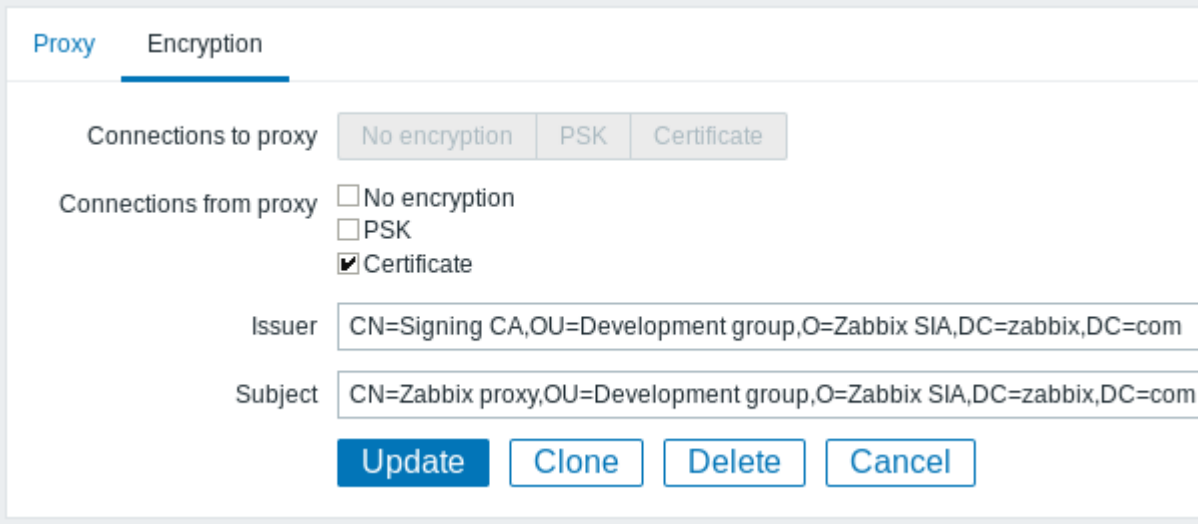
```
TLSServerCertIssuer=CN=Signing CA,OU=Development group,O=Zabbix
SIA,DC=zabbix,DC=com
TLSServerCertSubject=CN=Zabbix server,OU=Development group,O=Zabbix
SIA,DC=zabbix,DC=com
TLSCertFile=/home/zabbix/zabbix_proxy.crt
TLSKeyFile=/home/zabbix/zabbix_proxy.key
```

5. Настройте шифрование этому прокси в веб-интерфейсе Zabbix:

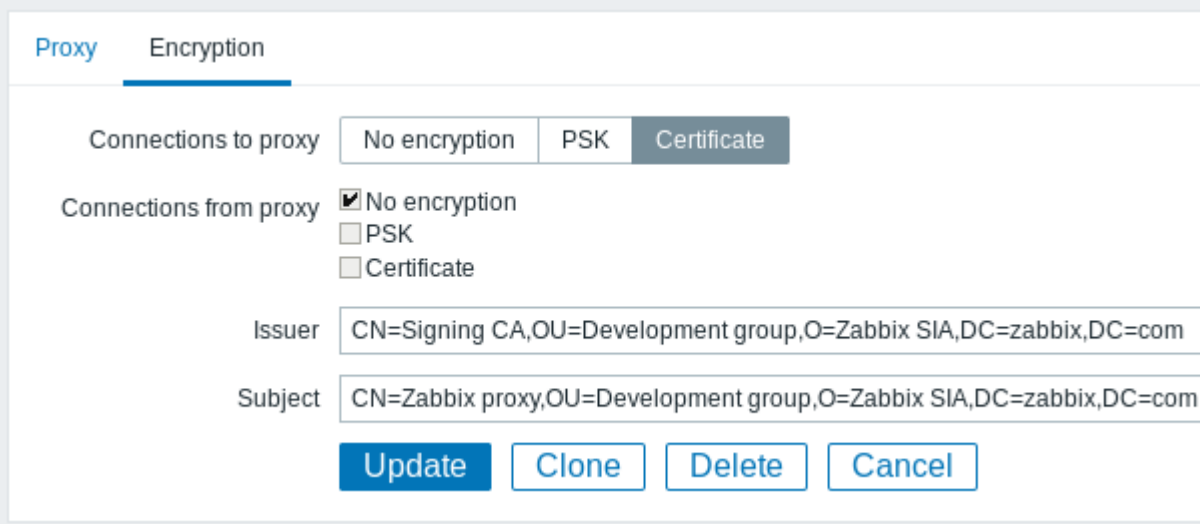
- Перейдите в: *Администрирование* → *Прокси*
- Выберите прокси и нажмите на вкладку **Шифрование**

В примере ниже поля Эмитент и Субъект заполнены - смотрите Ограничение разрешенных Эмитента и Субъекта сертификата о том, как использовать эти поля.

При активном прокси



При пассивном прокси



Настройка шифрования для Zabbix агента на основе сертификата

1. Подготовьте файлы с CA сертификатами верхнего уровня, сертификатом (цепочкой) агента и приватным ключем, как описано в Настройке сертификата на Zabbix сервере. Измените параметры TLSCAFile, TLSCertFile, TLSKeyFile в файле конфигурации агента соответственно.

2. При активных проверках измените TLSConnect параметр:

```
TLSConnect=cert
```

При пассивных проверках измените TLSAccept параметр:

```
TLSAccept=cert
```

3. Теперь у вас есть минимальная настройка агента на основе сертификата. Вы возможно захотите улучшить безопасность агента, указав параметры TLSServerCertIssuer и TLSServerCertSubject.(смотри Ограничение разрешенных Эмитента и Субъекта сертификата).

4. В конечном итоге параметры TLS в файле конфигурации агента могут выглядеть следующим образом:

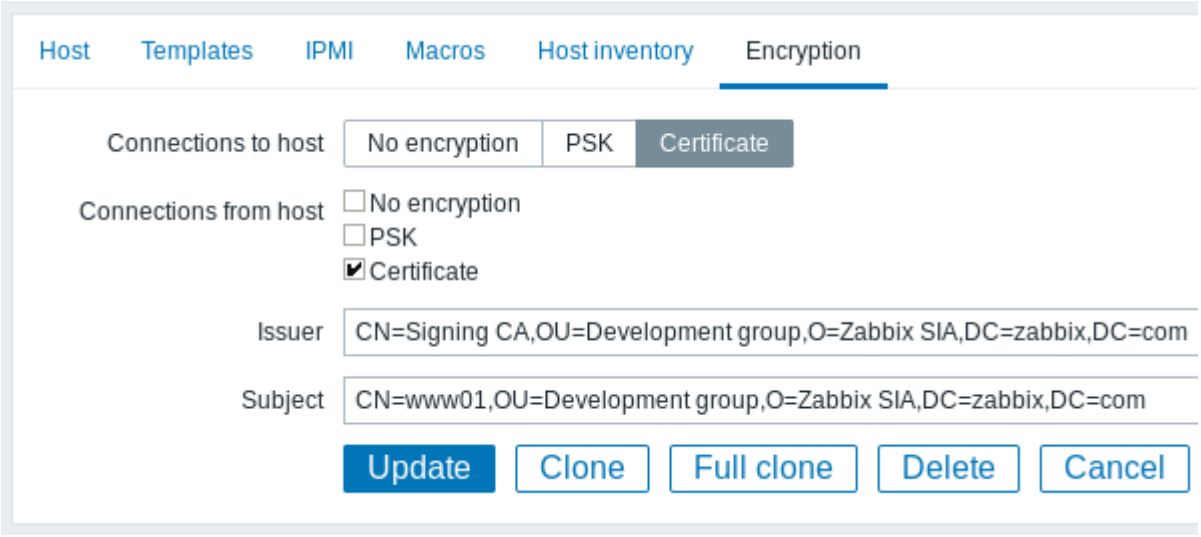
```
TLSConnect=cert
TLSAccept=cert
TLSCAFile=/home/zabbix/zabbix_ca_file
TLSServerCertIssuer=CN=Signing CA,OU=Development group,O=Zabbix
SIA,DC=zabbix,DC=com
TLSServerCertSubject=CN=Zabbix proxy,OU=Development group,O=Zabbix
SIA,DC=zabbix,DC=com
TLSCertFile=/home/zabbix/zabbix_agentd.crt
TLSKeyFile=/home/zabbix/zabbix_agentd.key
```

(Пример предполагает, что хост наблюдается через прокси, отсюда Субъект сертификата прокси.)

5. Настройте шифрование этому агенту в веб-интерфейсе Zabbix:

- Перейдите в: *Настройка → Узлы сети*
- Выберите узел сети и нажмите на вкладку **Шифрование**

В примере ниже поля Эмитент и Субъект заполнены - смотрите Ограничение разрешенных Эмитента и Субъекта сертификата о том, как использовать эти поля.



Ограничение разрешенных Эмитента и Субъекта сертификата

Когда два компонента Zabbix (например, сервер и агент) устанавливают TLS соединение, они оба проверяют сертификаты друг друга. Если сертификат узла подписан доверенным CA (с предварительно подготовленным сертификатом верхнего уровня в `TLSCAFile`), является действительным, он не истёк и проходит некоторые другие проверки, тогда коммуникация может продолжаться. Эмитент и субъект сертификата в этом простом случае не проверяется.

Здесь имеется риск - кто-угодно при наличии действительного сертификата может выдавать себя за другого (например, сертификат хоста можно использовать, чтобы выдавать себя за сервер). Такое поведение может быть приемлемо в небольших средах, где сертификаты подписываются специализированного внутреннего CA и риск действий от чужого имени является минимальным.

Если ваш CA верхнего уровня используется для выдачи других сертификатов, которые не должны приниматься Zabbix или вы хотите снизить риск действий от чужого имени, вы можете ограничить разрешенные сертификаты, указав их строки Эмитента и Субъекта.

Например, вы можете записать в файл конфигурации Zabbix прокси:

```
TLSServerCertIssuer=CN=Signing CA,OU=Development group,O=Zabbix
SIA,DC=zabbix,DC=com
TLSServerCertSubject=CN=Zabbix server,OU=Development group,O=Zabbix
SIA,DC=zabbix,DC=com
```

При наличии этих настроек активный прокси не будет разговаривать с Zabbix сервером с другими строками Эмитента и Субъекта в сертификате, пассивный прокси не пример запросы от такого сервера.

Несколько заметок о соответствии строк Эмитента и Субъекта:

1. Строки Эмитента и Субъекта проверяются независимо. Обе строки опциональны.
2. Допустимы символы UTF-8.
3. Не указанная строка означает, что принимается любая строка.
4. Строки сравниваются “как-есть”, они должны в точности быть такими же.
5. Шаблоны и регулярные выражения при проверке соответствия не поддерживаются.

6. Реализованы только некоторые требования из RFC 4514 Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names:
    1. управляющие символы '"' (U+0022), '+' U+002B, ',' U+002C, ';' U+003B, '<' U+003C, '>' U+003E, '\' U+005C в любом месте в строке.
    2. управляющие символы пробела (' ' U+0020) или символ решетки ('#' U+0023) в начале строки.
    3. управляющий символ пробела (' ' U+0020) в конце строки.
7. Совпадения не будет, если встречается нулевой символ (U+0000) (RFC 4514 позволяет это).
8. Требования RFC 4517 Lightweight Directory Access Protocol (LDAP): Syntaxes and Matching Rules и RFC 4518 Lightweight Directory Access Protocol (LDAP): Internationalized String Preparation не поддерживаются по причине необходимого объема работы.

Очередность полей в Эмитента и Субъекта строках и форматирование очень важны! Zabbix следует RFC 4514 рекомендации и использует “обратный” порядок этих полей.

Обратный порядок можно продемонстрировать в примере:

```
TLSServerCertIssuer=CN=Signing CA,OU=Development group,O=Zabbix
SIA,DC=zabbix,DC=com
TLSServerCertSubject=CN=Zabbix proxy,OU=Development group,O=Zabbix
SIA,DC=zabbix,DC=com
```

Обратите внимание, что он начинается с верхнего уровня (CN), переходит к среднему уровню (OU, O) и заканчивается полями верхнего уровня (DC).

По умолчанию *OpenSSL* отображает поля Эмитента и Субъекта в “нормальном” порядке, в зависимости от использованных дополнительных опций:

```
$ openssl x509 -noout -in /home/zabbix/zabbix_proxy.crt -issuer -subject
issuer= /DC=com/DC=zabbix/O=Zabbix SIA/OU=Development group/CN=Signing CA
subject= /DC=com/DC=zabbix/O=Zabbix SIA/OU=Development group/CN=Zabbix proxy

$ openssl x509 -noout -text -in /home/zabbix/zabbix_proxy.crt
Certificate:
        ...
        Issuer: DC=com, DC=zabbix, O=Zabbix SIA, OU=Development group,
CN=Signing CA
      ...
        Subject: DC=com, DC=zabbix, O=Zabbix SIA, OU=Development group,
CN=Zabbix proxy
```

Здесь строки Эмитента и Субъекта начинаются с верхнего уровня (DC) и заканчиываются полем нижнего уровня (CN), пробелы и разделители полей зависят от используемых опций. Ни одно из этих значений не будет совпадать в Zabbix полях Эмитента и Субъекта!

Для получения надлежащих строк Эмитента и Субъекта, допустмых в Zabbix, вызовите OpenSSL со специальными опциями
`-nameopt esc_2253,esc_ctrl,utf8,dump_nostr,dump_unknown,dump_der,sep_comma_plus,dn_rev`

,sname:

```
$ openssl x509 -noout -issuer -subject -nameopt
esc_2253,esc_ctrl,utf8,dump_nostr,dump_unknown,dump_der,sep_comma_plus,dn_re
v,sname -in /home/zabbix/zabbix_proxy.crt
issuer= CN=Signing CA,OU=Development group,O=Zabbix SIA,DC=zabbix,DC=com
subject= CN=Zabbix proxy,OU=Development group,O=Zabbix SIA,DC=zabbix,DC=com
```

Теперь строковые поля находятся в обратном" порядке, поля разделены запятой, строки можно использовать в файлах конфигурации Zabbix и в веб-интерфейсе.

Ограничения при использовании расширений X.509 v3 сертификатов

- Расширение **Альтернативное имя субъекта (*subjectAltName*)**.
  Альтернативные имена субъектов из *subjectAltName* расширения (такие как IP адрес, e-mail адрес) не поддерживаются Zabbix. В Zabbix проверяется только значение поля "Субъект" (смотри Ограничение разрешенных Эмитента и Субъекта сертификата).
  Если сертификат использует *subjectAltName* расширение, тогда результат зависит от конкретной комбинации наборов инструментов криптографии с которыми скомпилированы компоненты Zabbix (это расширение может работать, а может и не работать, Zabbix может отказаться принимать такие сертификаты от узлов).
- Расширение **Использование Расширенного Ключа**.
  Если используется, то, как правило, необходимо указывать как *clientAuth* (TLS WWW аутентификация клиента), так и *serverAuth* (TLS WWW аутентификация сервера).
  Например, при пассивных проверках Zabbix агент выступает в роли TLS сервера, таким образом необходимо указать *serverAuth* в сертификате агента. При активных проверках в сертификате агента необходимо задать *clientAuth*.
  GnuTLS выводит предупреждение в случае нарушения использования ключа, но разрешает продолжение соединения.
- Расширение **Ограничения Имени**.
  Не все наборы инструментов криптографии поддерживают его. Это расширение может помешать Zabbix в загрузке CA сертификатов, где этот раздел промаркирован как *критический* (зависит от конкретного набора инструментов криптографии).

Списки отозванных сертификатов (CRL)

Если сертификат скомпрометирован, CA может отозвать его, включив в CRL. Списки CRL можно настраивать в файлах конфигурации сервера, прокси и агента, используя параметр TLSCRLFile. Например:

```
TLSCRLFile=/home/zabbix/zabbix_crl_file
```

где zabbix_crl_file может содержать списки CRL от нескольких CA и может выглядеть следующим образом:

```
-----BEGIN X509 CRL-----
MIIB/DCB5QIBATANBgkqhkiG9w0BAQUFADCBgTETMBEGCgmSJomT8ixkARkWA2Nv
...
treZeUPjb7LSmZ3K2hpbZN7SoOZcAoHQ3GWd9npuctg=
```

```
-----END X509 CRL-----
-----BEGIN X509 CRL-----
MIIB+TCB4gIBATANBgkqhkiG9w0BAQUFADB/MRMwEQYKCZImiZPyLGQBGRYDY29t
...
CAEebS2CND3ShBedZ8YSil59O6JvaDP61lR5lNs=
-----END X509 CRL-----
```

CRL файл загружается только при запуске Zabbix. При обновлении CRL требуется перезапуск.

Если компонент Zabbix скомпилирован с *OpenSSL* и используются списки CRL, тогда каждый сертификат верхнего и промежуточного уровней CA в цепочках сертификатов должен иметь соответствующий список CRL (может быть пустым) в TLSCRLFile.

Ограничения в использовании CRL расширений

- Расширение **Идентификатор Ключа Полномочий**.
  CRL для CA с идентичными именами могут не работает в случае *mbedTLS* (*PolarSSL*), даже с расширением “Идентификатор Ключа Полномочий” (“Authority Key Identifier”).