# Руководство по настройке работы сервисов оркестратора с RabbitMQ через SSL

**Оглавление:**

1. [Заведение OpenSSL сертификатов для работы через SSL](#Заведение_сертификатов).
2. [Настройка RabbitMQ на Linux для работы через SSL](#Настройка_RabbitMQ_Linux).
3. [Настройка RabbitMQ на Windows для работы через SSL](#Настройка_RabbitMQ_Windows).
4. [Настройка сервисов оркестратора для работы с RabbitMQ через SSL.](#Настройка_сервисов_оркестратора)
5. [Полезные ссылки](#Полезные_ссылки).
6. **Заведение OpenSSL сертификатов для работы через SSL.**
   1. Устанавливаем **OpenSSL** либо с официального сайта (<https://www.openssl.org/>), либо используем ту версию, что идет в составе одной из утилит (например можно использовать ту, которая идет в составе **git**).
   2. Для того, чтобы в скриптах каждый раз не использовать полный путь к файлу **openssl.exe**, рекомендуется добавить этот путь в настройки системной переменной **Path** (если же Вы предпочитаете работать через **PowerShell**, а не через командную строку, и не желаете изменять системную переменную **Path**, то можно использовать сеансовые алиасы).
   3. Далее в данном руководстве предполагается, что полный путь прописан в системной переменной **Path**.
   4. Прописать путь в **Path**, если Вы работаете в **Windows** можно таким образом:

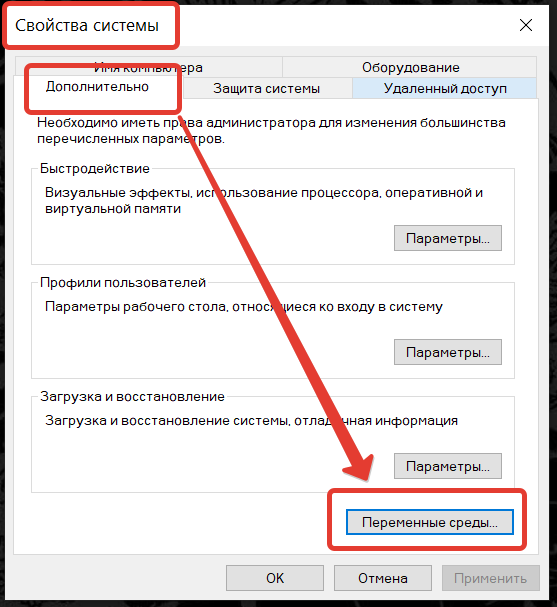
Открываем указанное ниже диалоговое окно (и далее выбираем соответствующие вкладки / нажимаем указанные кнопки / выбираем указанные строки из списков):

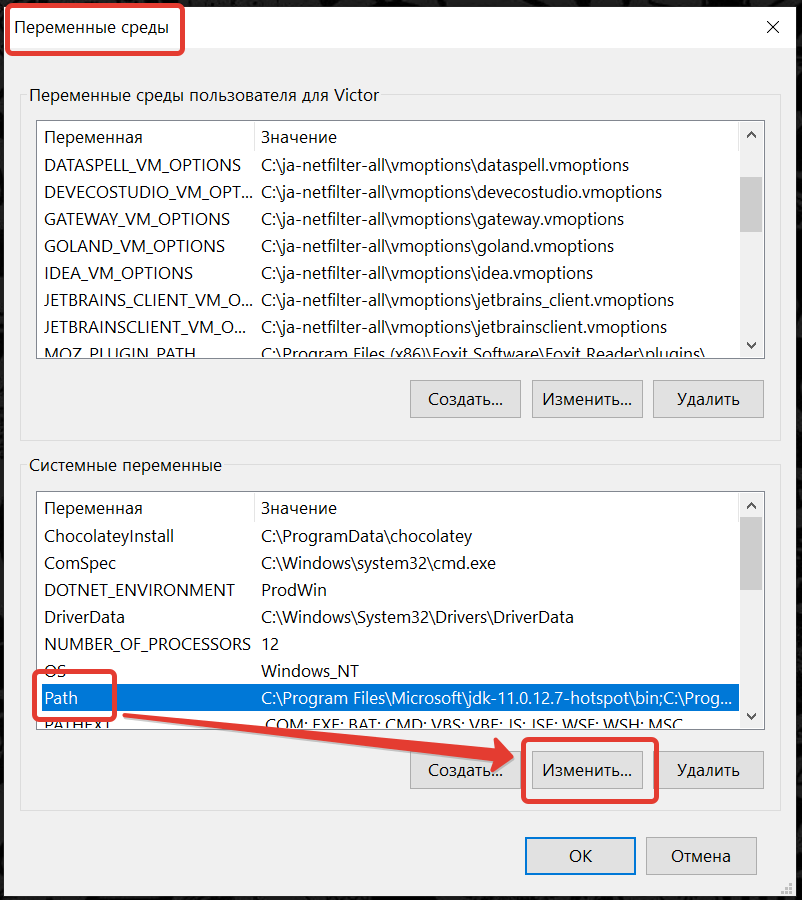
- В нерусифицированной версии Windows:

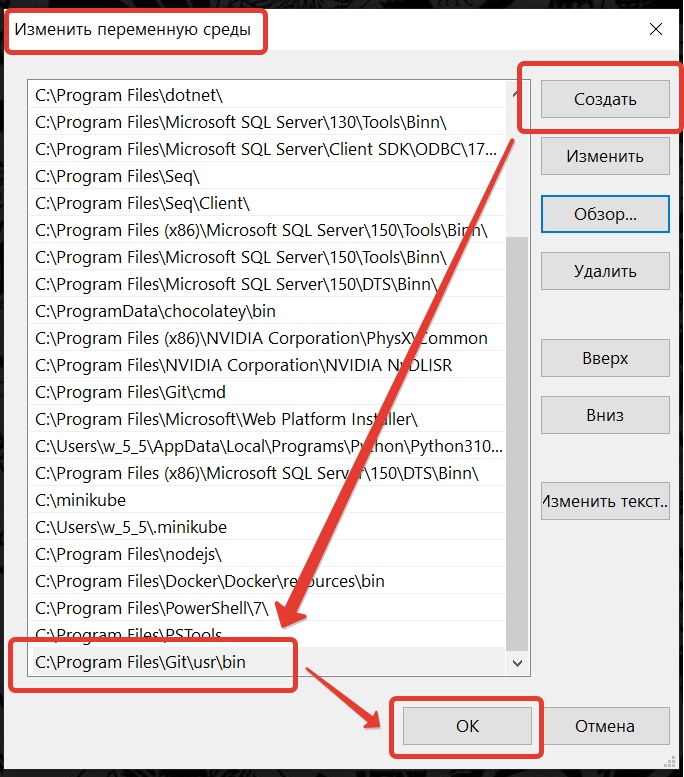
***System Properties* > Advanced > *Environment Variables…* > *System variables* > *Path > Edit… > New > [Тут указываем полный путь к файлу openssl.exe]***:

- В русской версии Windows:

***Свойства системы* > *Дополнительно* > *Переменные среды…* > *Системные переменные* > *Path > Изменить… > Создать > [Тут указываем полный путь к файлу openssl.exe]***:







* 1. Далее создаем каталог, в котором будут размещаться как сами сертификаты, так и необходимые для их создания файлы и переходим в него.
  2. **RabbitMQ** должен при запуске иметь доступ к **корневому сертификату СА**, **сертификату сервера** и **приватному ключу** сертификата сервера. Создадим эти файлы.
  3. Первым делом создаем **корневой ключ CA** и затем создаем **корневой сертификат CA**.

|  |
| --- |
| openssl genrsa -out RMQ-CA-Key.pem  openssl req -new -key RMQ-CA-Key.pem -x509 -days 10000 -out RMQ-CA-cert.pem |

Для этого открываем командную строку и выполняем в ней такие 2 команды:

Заполняем необходимые поля для сертификата. Имена выбираем произвольно. На этом **корневой сертификат CA** создан.

|  |
| --- |
| openssl genrsa -out RMQ-server-key.pem 2048 |

* 1. Генерируем **ключ для сервера**:

|  |
| --- |
| openssl req -new -key RMQ-server-key.pem -out RMQ-signingrequest.csr |

* 1. Теперь создаем **запрос на** **сертификат сервера**.

При заполнении полей, в поле **Common Name** важно указать имя сервера: **домен** или **IP** адрес **сервера** (например домен: **db.example.com** или IP-адрес: **192.168.0.116**):

* 1. Генерируем **самоподписанный x509-сертификат сервера** используя запрос на сертификат сервера, сертификат СА и ключ сертификата СА:

|  |
| --- |
| openssl x509 -req -days 1000 -in RMQ-signingrequest.csr -CA RMQ-CA-cert.pem -CAkey RMQ-CA-Key.pem -CAcreateserial -out RMQ-server-cert.pem |

* 1. Теперь таким же образом (как описано в трех предыдущих пунктах) создаем связку ключ-сертификат для клиента. Имена файлов соответственно меняем (например: **RMQ-client-key.pem, RMQ-client-signingrequest.csr, RMQ-client-cert.pem**) :

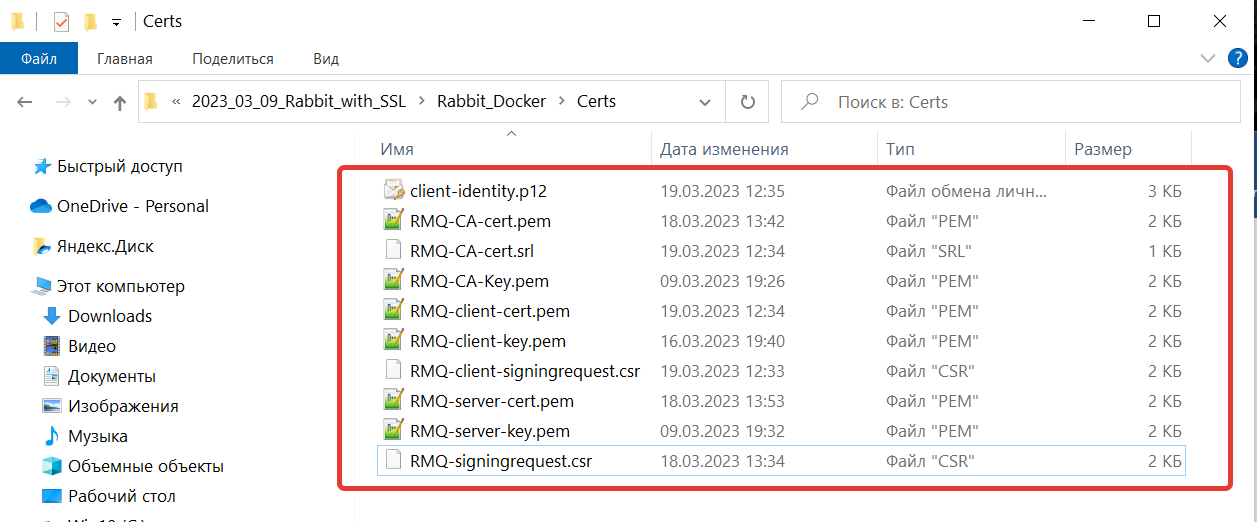
|  |
| --- |
| openssl genrsa -out RMQ-client-key.pem 2048  openssl req -new -sha256 -key RMQ-client-key.pem -out RMQ-client-signingrequest.csr  openssl x509 -req -days 1000 -in RMQ-client-signingrequest.csr -CA RMQ-CA-cert.pem -CAkey RMQ-CA-Key.pem -CAcreateserial -out RMQ-client-cert.pem |

При заполнении полей сертификата клиента, в поле **Common Name** указываем **логин** под которым будет выполняться подключение к RabbitMQ (по умолчанию для сервисов Оркестратора это имя: **admin** – далее во всех скриптах в этом руководстве подразумевается, что было задано именно такое имя для этого параметра).

* 1. Для работы с **RabbitMQ** клиентам требуется специальный тип сертификата **PKCS12**, создаем его такой командой:

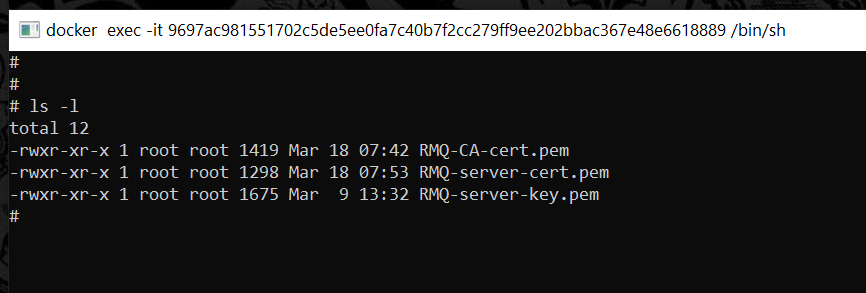
|  |
| --- |
| openssl pkcs12 -export -out client-identity.p12 -inkey RMQ-client-key.pem -in RMQ-client-cert.pem |

* 1. При этом запоминаем (записываем) **парольную фразу**, требуемую для данного типа сертификатов. Она нам пригодится в дальнейшем при конфигурировании работы сервисов оркестратора.
  2. В итоге в нашей рабочей папке должно быть 10 таких файлов:



* 1. Теперь переходим к настройке **RabbitMQ** и сервисов оркестратора для работы через **SSL**, как описано далее в настоящем руководстве.

1. **Настройка RabbitMQ на Linux для работы через SSL.**
   1. Копируем **корневой сертификат CA**, **ключ** и **сертификат сервера** (созданные ранее в **п.1.** настоящего руководства) в **удобный для Вас каталог** (допустим он расположен по такому пути: **/etc/pki/tls**).
   2. В итоге в этом каталоге должны быть установлены указанные выше файлы с такими правами:



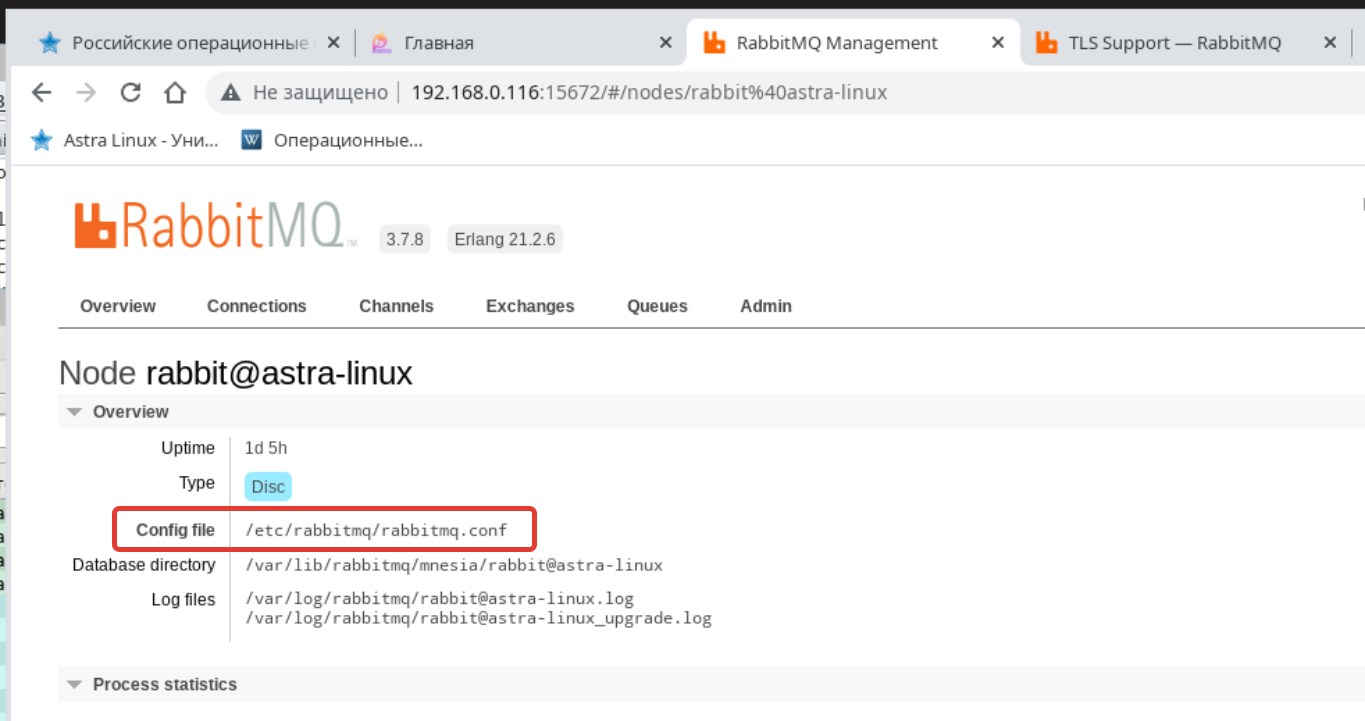
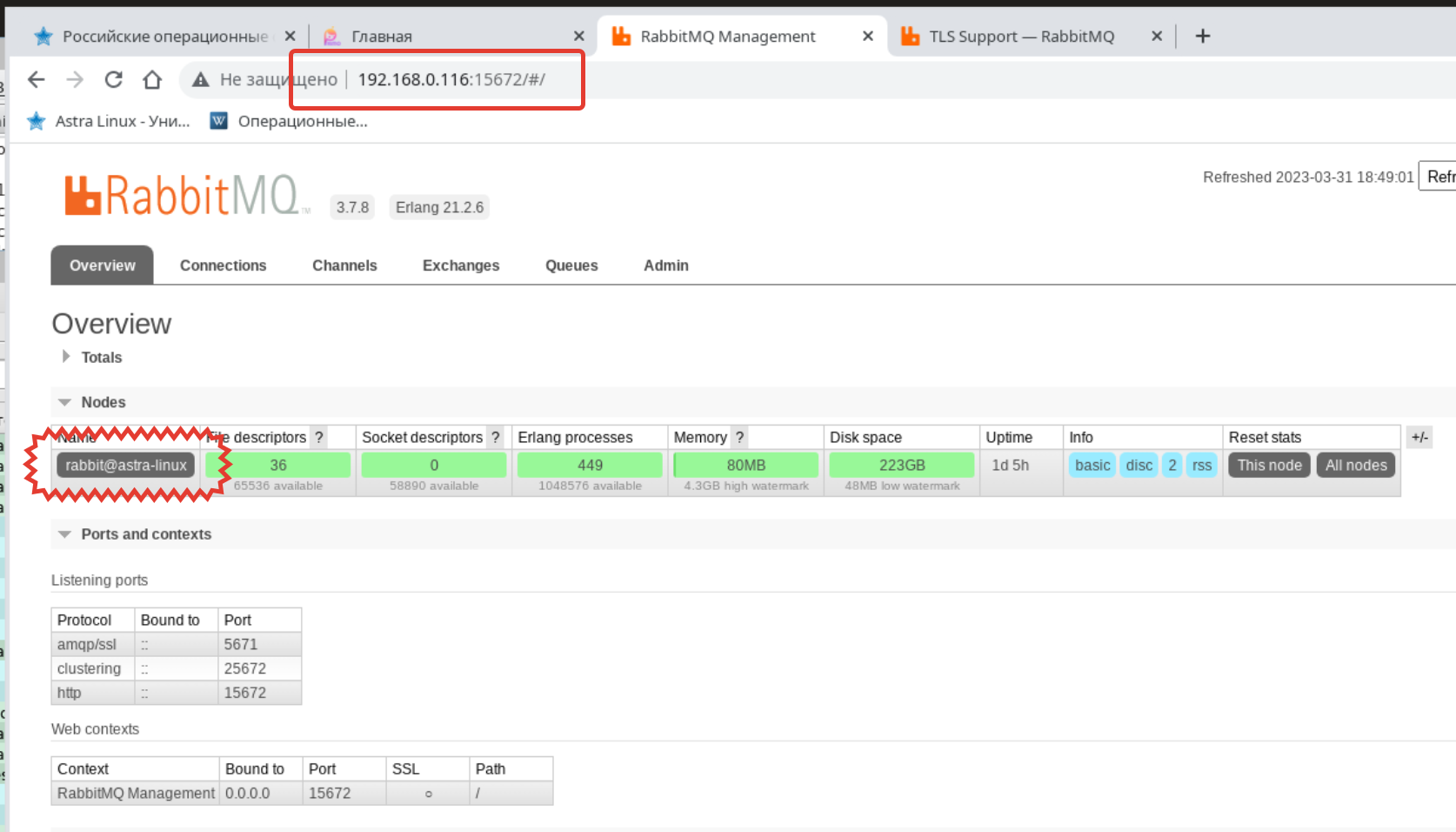
* 1. Теперь переходим к правке файла конфигурации **RabbitMQ**. Он должен быть собран с поддержкой **SSL/TLS**. Этот файл может по разному именоваться и может иметь разное расширение.
  2. Расположение всех конфигурационых файлов можно найти запустив в командной строке команду:

|  |
| --- |
| rabbitmq-diagnostics status |

Конфигурационные файлы будут перечислены в секции **Config files**:



* 1. Расположение конфигурационных файлов также можно найти, если кликнуть по кнопке с названием текущего узла в **RabbitMQ management UI** (в примере, показанном на картинке, следует кликнуть по кнопке с наименованием rabbit@astra-linux) :



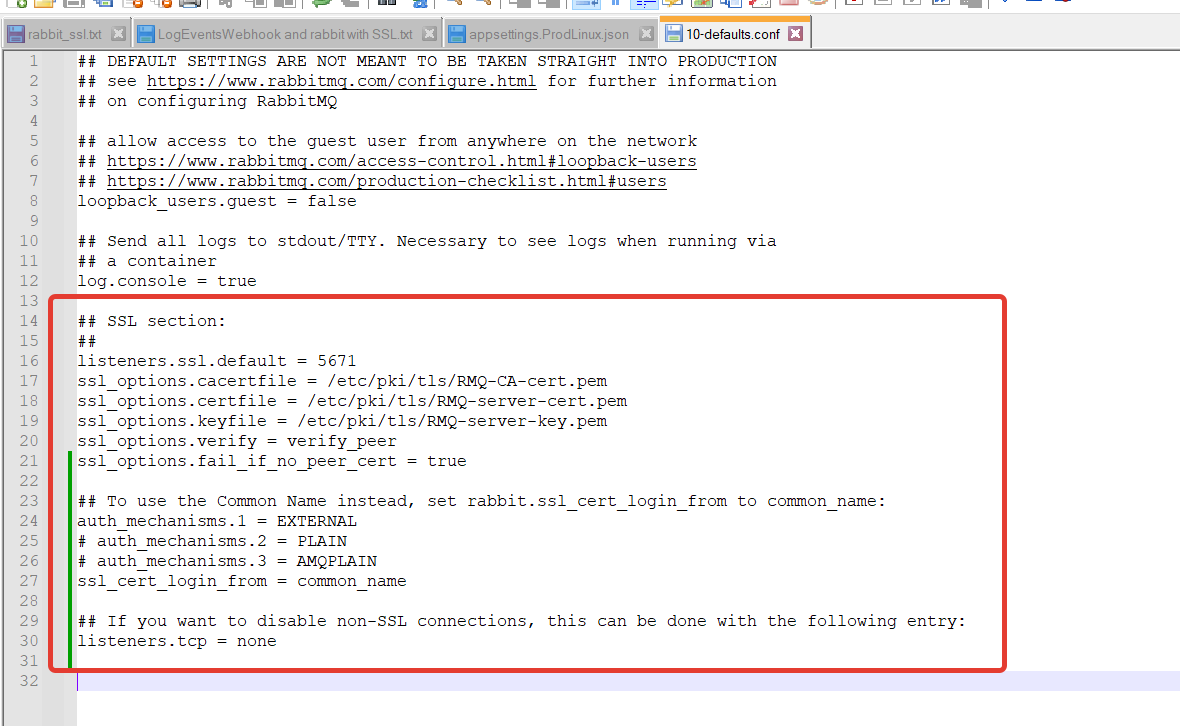
* 1. Если конфигурационный файл имеет расширение **\*.config**, то изменяем его в соответствии с описанием по настройке **SSL** в **RabbitMQ** для операционной системы **Windows** данного руководства.
  2. Если этот файл имеет расширение \*.**conf**, то для работы **RabbitMQ** через **SSL/TLS** вносим в него такие строки, подставив актуальные пути к сертификатам и ключу:

|  |
| --- |
| listeners.ssl.default = 5671  ssl\_options.cacertfile = /path/to/ca\_certificate.pem  ssl\_options.certfile = /path/to/server\_certificate.pem  ssl\_options.keyfile = /path/to/server\_key.pem  ssl\_options.verify = verify\_peer  ssl\_options.fail\_if\_no\_peer\_cert = true  auth\_mechanisms.1 = EXTERNAL  auth\_mechanisms.2 = PLAIN  auth\_mechanisms.3 = AMQPLAIN  ssl\_cert\_login\_from = common\_name |

Если нам необходимо, чтобы **RabbitMQ** работал исключительно только через **SSL/TLS**, то следует внести в конфиг такие строки:

|  |
| --- |
| listeners.ssl.default = 5671  ssl\_options.cacertfile = /path/to/ca\_certificate.pem  ssl\_options.certfile = /path/to/server\_certificate.pem  ssl\_options.keyfile = /path/to/server\_key.pem  ssl\_options.verify = verify\_peer  ssl\_options.fail\_if\_no\_peer\_cert = true  auth\_mechanisms.1 = EXTERNAL  ssl\_cert\_login\_from = common\_name  listeners.tcp = none |

В моем случае эта секция конфига выглядит таким образом:



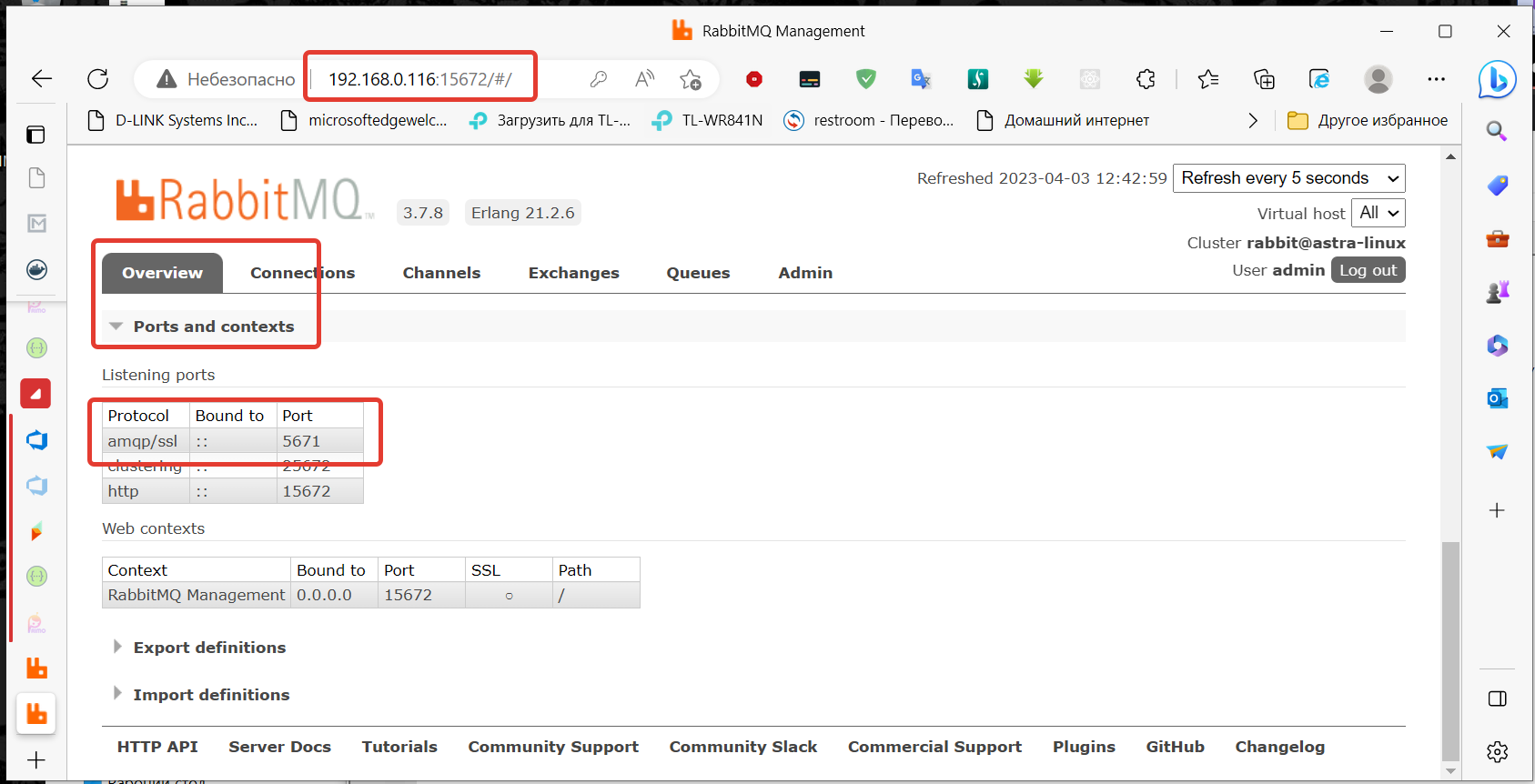
* 1. Следующий шаг: необходимо активировать встроенный в **RabbitMQ** плагин для аутентификации через **SSL**. Сделать это можно выполнив в командной строке такую команду:

|  |
| --- |
| rabbitmq-plugins enable rabbitmq\_auth\_mechanism\_ssl |

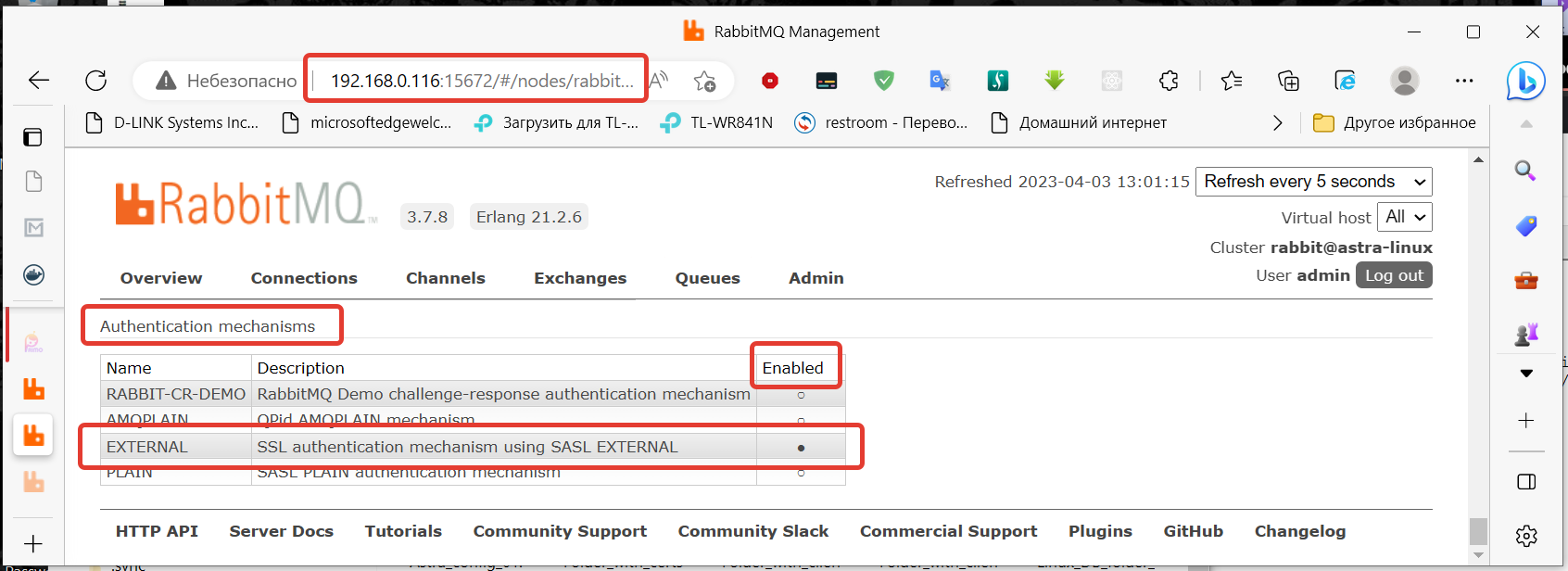
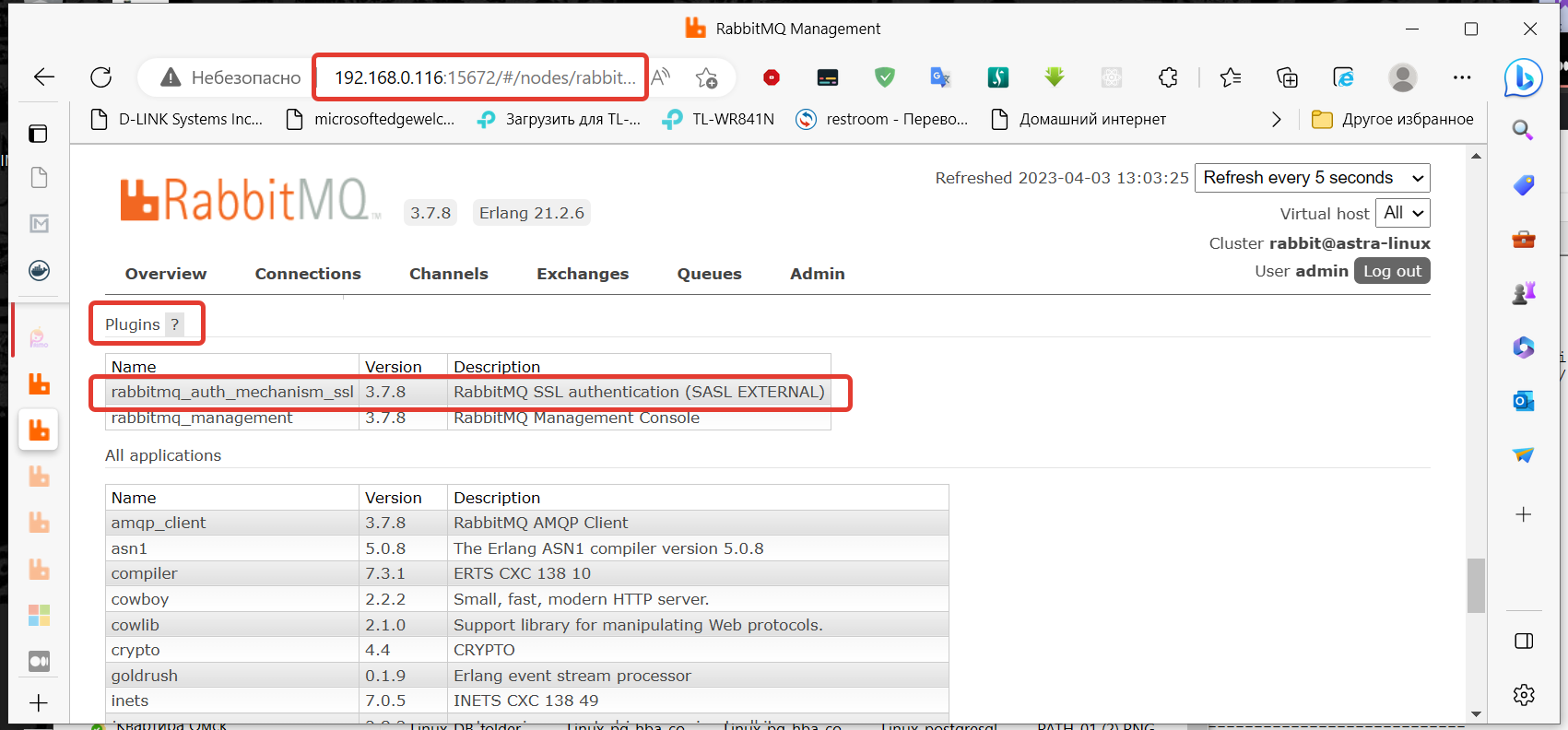
* 1. После этого необходимо перезагрузить сервис **RabbitMQ**. Сделать это можно выполнив в командной строке такие 2 команды:

|  |
| --- |
| rabbitmq-service stop  rabbitmq-service start |

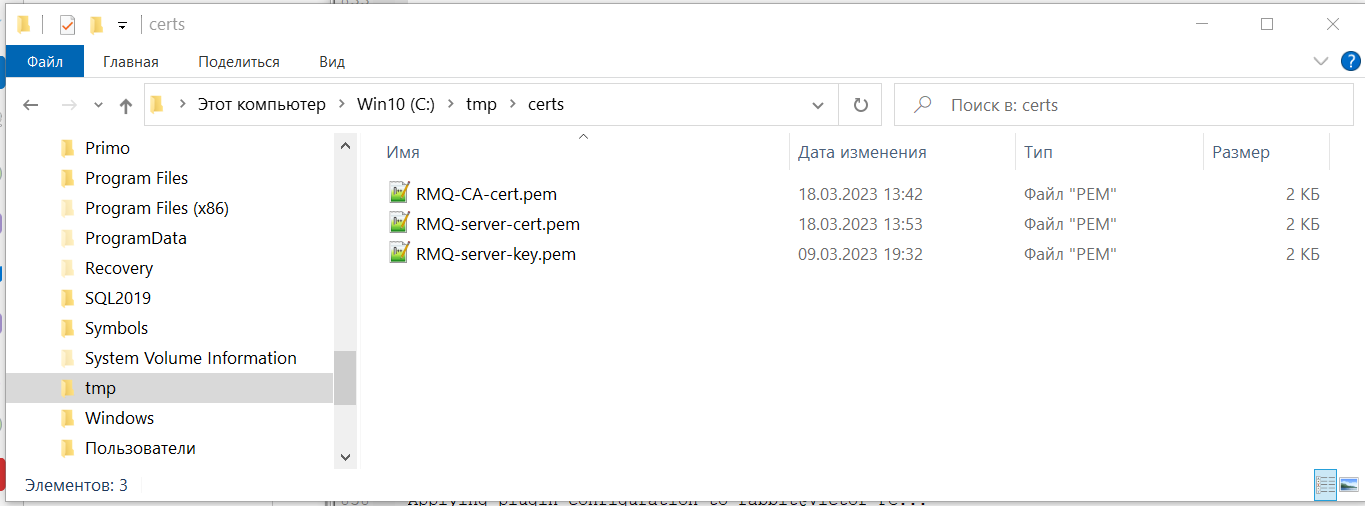
* 1. Теперь проверяем, что мы все сделали правильно и сервис **RabbitMQ** может работать через **SSL/TLS**. Запускаем **RabbitMQ management UI** и проверяем, что запущен SSL-порт **5671**:



Затем кликнув по кнопке с названием текущего узла, проверяем, что запущен плагин для работы через **SSL** и включен механизм аутентификации **EXTERNAL** :



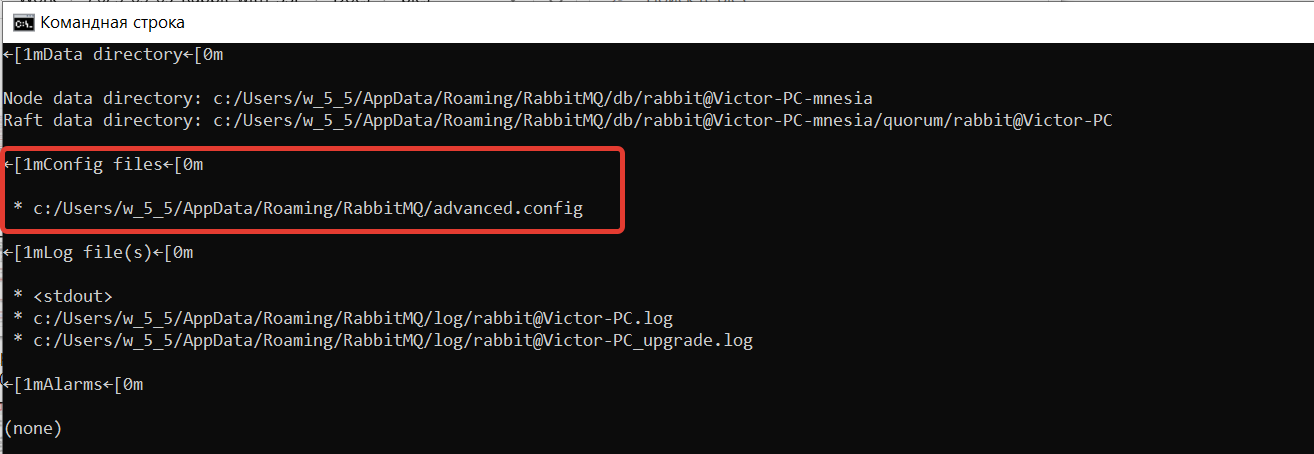
1. **Настройка RabbitMQ на Windows для работы через SSL.**
   1. Копируем **корневой сертификат CA**, **ключ** и **сертификат сервера** (созданные ранее в **п.1.** настоящего руководства) в **удобный для Вас каталог** (допустим он расположен по такому пути: **C:/tmp/certs**).
   2. В итоге в этом каталоге должны быть установлены указанные выше файлы:



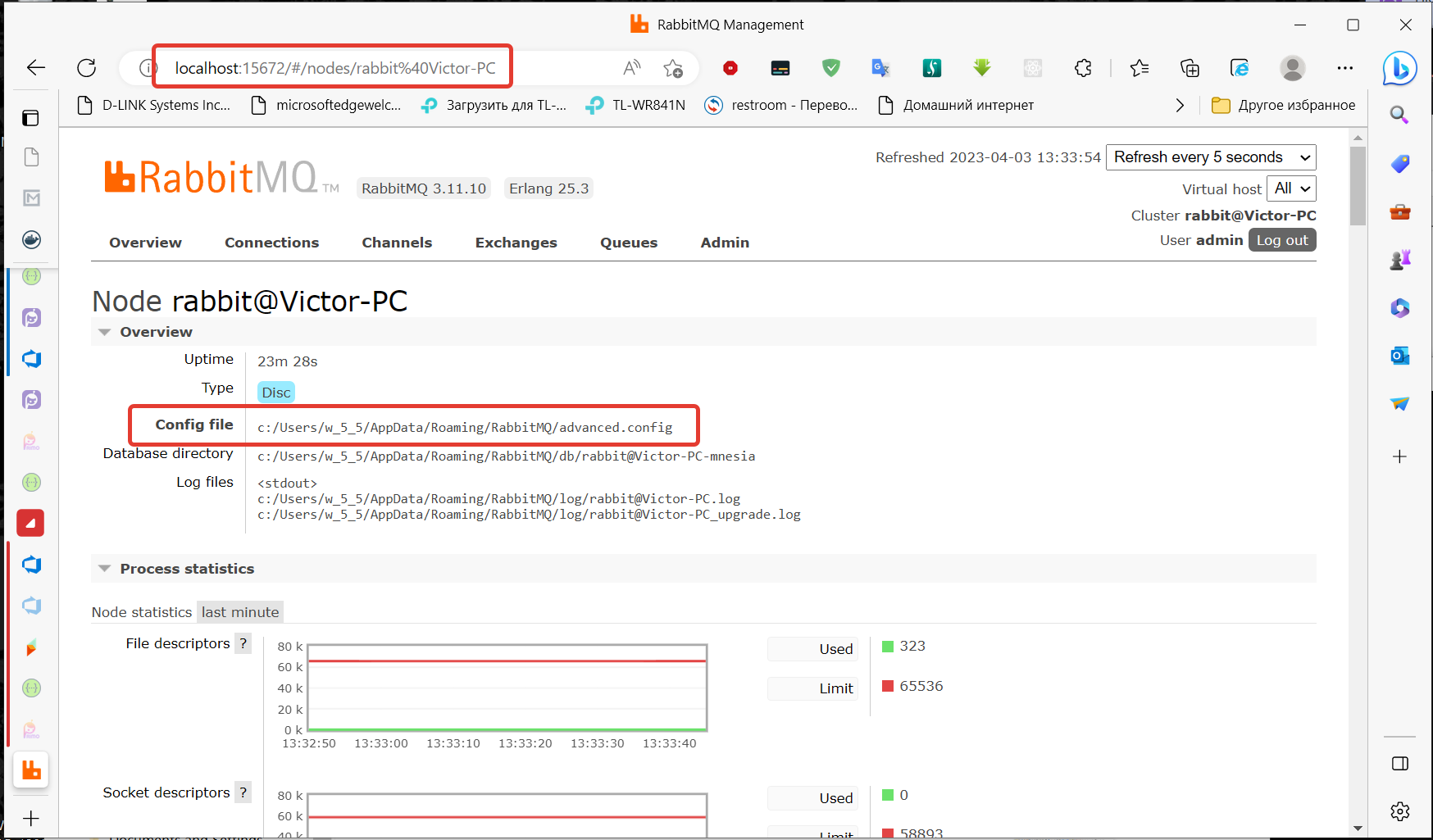
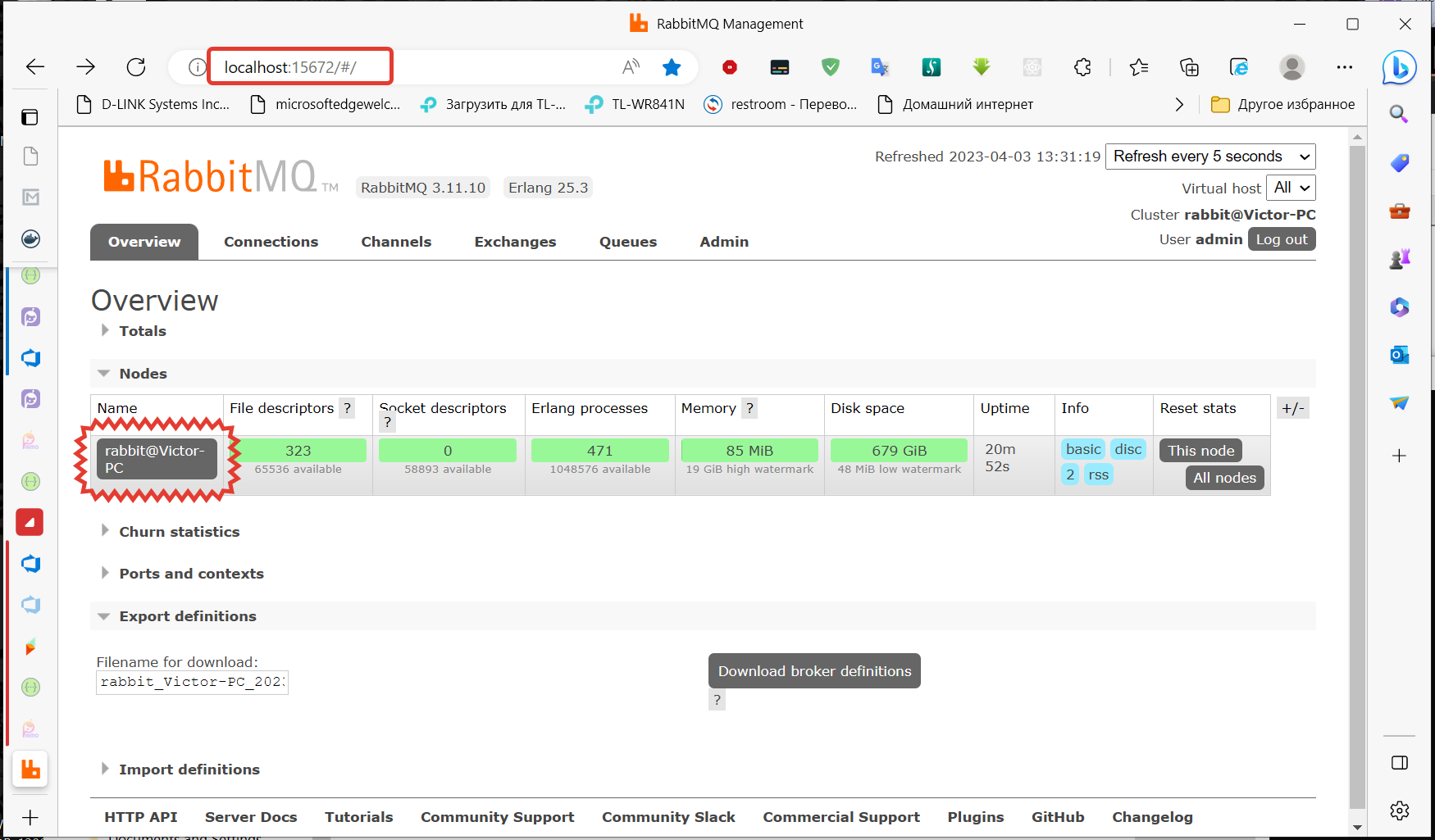
* 1. Теперь переходим к правке файла конфигурации **RabbitMQ**. Он должен быть собран с поддержкой **SSL/TLS**. Этот файл может по разному именоваться и может иметь разное расширение.
  2. Расположение всех конфигурационых файлов можно найти запустив в командной строке команду:

|  |
| --- |
| rabbitmq-diagnostics status |

Конфигурационные файлы будут перечислены в секции **Config files**:



* 1. Расположение конфигурационных файлов также можно найти, если кликнуть по кнопке с названием текущего узла в **RabbitMQ management UI** (в примере, показанном на картинке, следует кликнуть по кнопке с наименованием rabbit@Victor-PC):



* 1. Если конфигурационный файл имеет расширение **\*.conf**, то изменяем его в соответствии с описанием по настройке **SSL** в **RabbitMQ** для операционной системы **Linux** данного руководства.
  2. Если этот файл имеет расширение \*.config, то для работы **RabbitMQ** через **SSL/TLS** вносим в него такие строки, подставив актуальные пути к сертификатам и ключу:

|  |
| --- |
| [  {ssl, [{versions, ["tlsv1.2", "tlsv1.1"]}]},  {rabbit, [  {auth\_backends, [rabbit\_auth\_backend\_internal]},  {auth\_mechanisms, ['PLAIN', 'AMQPLAIN', 'EXTERNAL']},  {ssl\_listeners, [5671]},  {ssl\_cert\_login\_from, common\_name},  {tcp\_listeners, [5672]},  {ssl\_options, [  {cacertfile, "C:/tmp/certs/RMQ-CA-cert.pem"},  {certfile, "C:/tmp/certs/RMQ-server-cert.pem"},  {keyfile, "C:/tmp/certs/RMQ-server-key.pem"},  {verify, verify\_peer},  {fail\_if\_no\_peer\_cert, true},  {depth, 2}  ]}  ]}  ]. |

Если нам необходимо, чтобы **RabbitMQ** работал исключительно только через **SSL/TLS**, то следует внести в конфиг такие строки:

|  |
| --- |
| [  {ssl, [{versions, ["tlsv1.2", "tlsv1.1"]}]},  {rabbit, [  {auth\_backends, [rabbit\_auth\_backend\_internal]},  {auth\_mechanisms, ['EXTERNAL']},  {ssl\_listeners, [5671]},  {ssl\_cert\_login\_from, common\_name},  {tcp\_listeners, []},  {ssl\_options, [  {cacertfile, "C:/tmp/certs/RMQ-CA-cert.pem"},  {certfile, "C:/tmp/certs/RMQ-server-cert.pem"},  {keyfile, "C:/tmp/certs/RMQ-server-key.pem"},  {verify, verify\_peer},  {fail\_if\_no\_peer\_cert, true},  {depth, 2}  ]}  ]}  ]. |

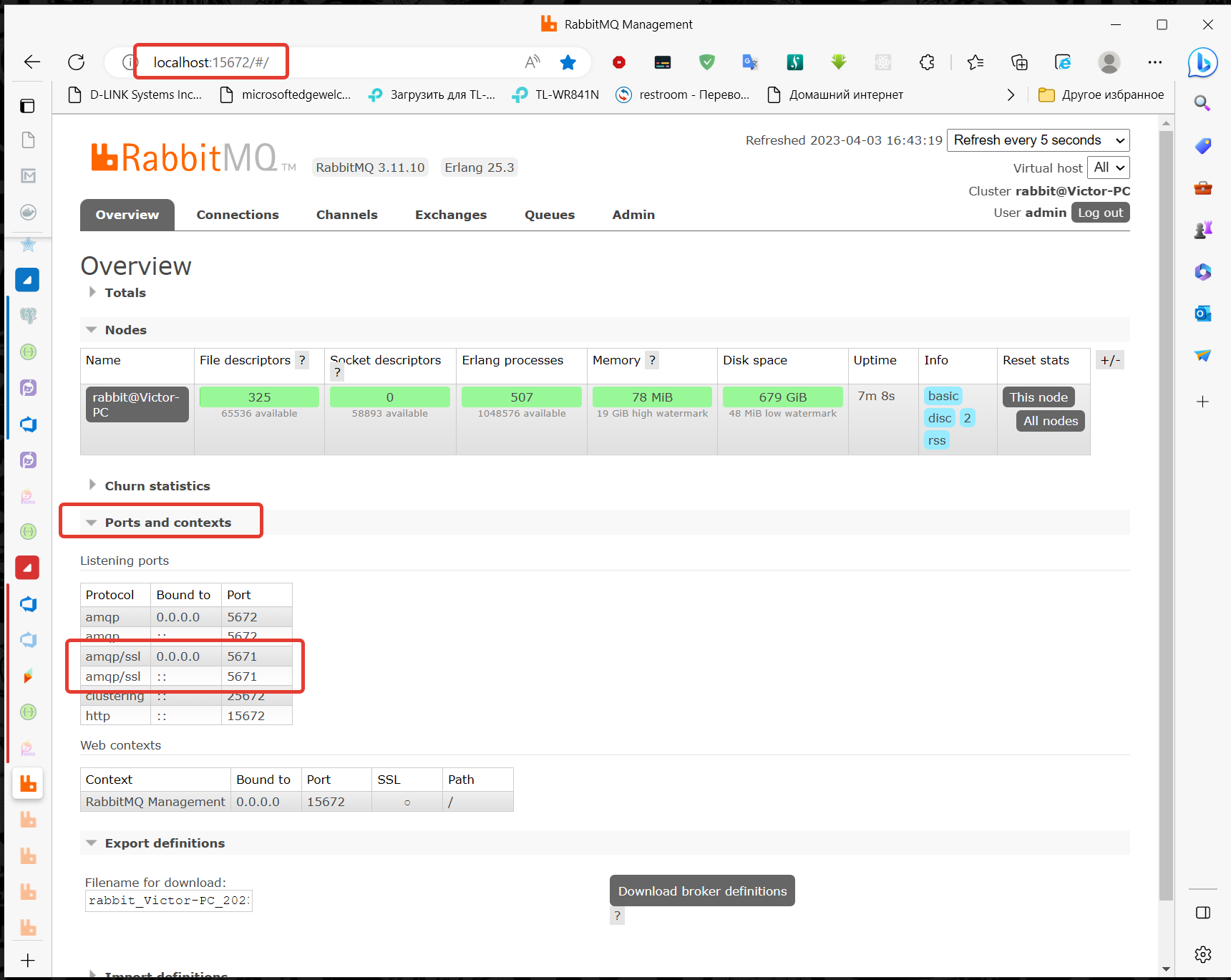
* 1. Следующий шаг: необходимо активировать встроенный в **RabbitMQ** плагин для аутентификации через **SSL**. Сделать это можно выполнив в командной такую команду:

|  |
| --- |
| rabbitmq-plugins enable rabbitmq\_auth\_mechanism\_ssl |

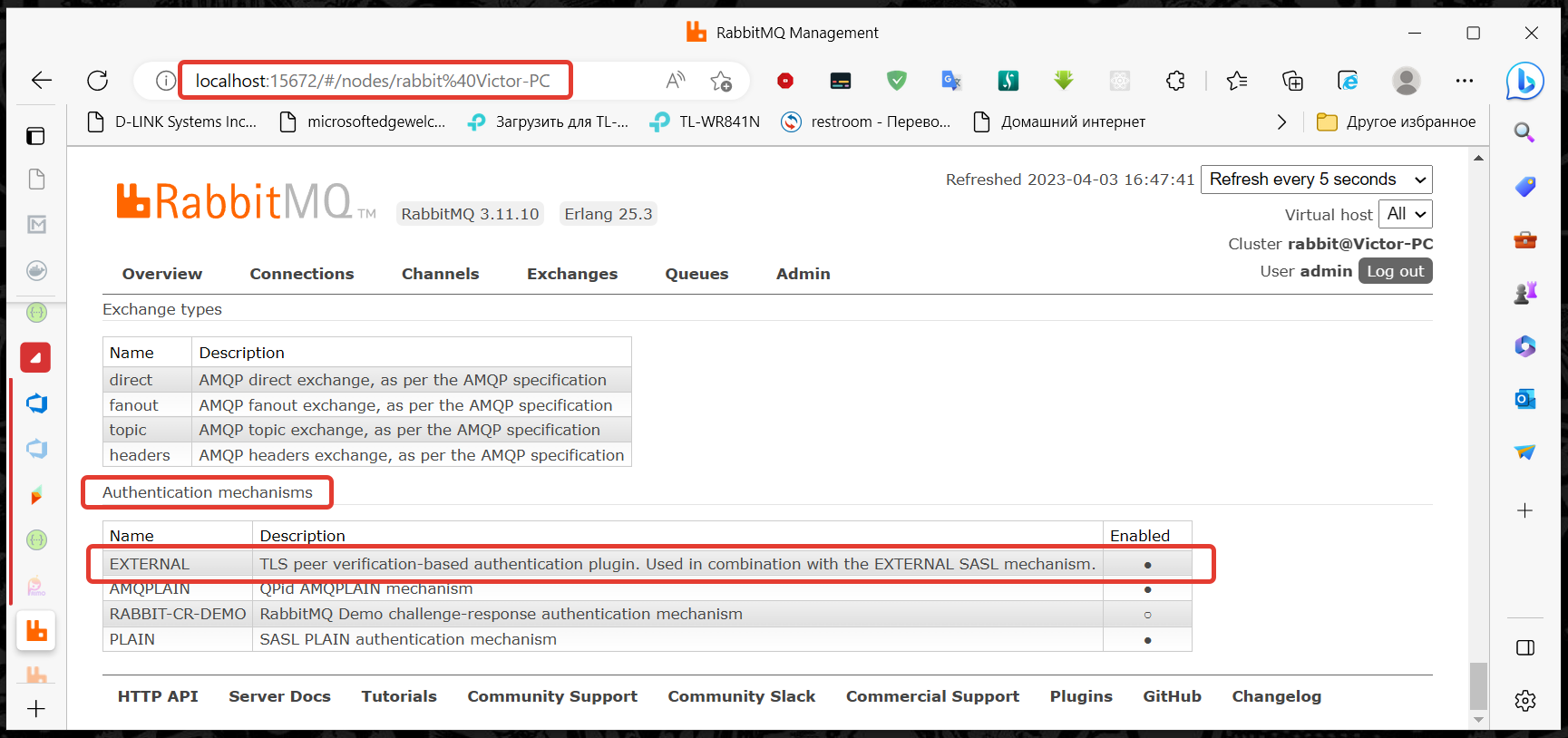
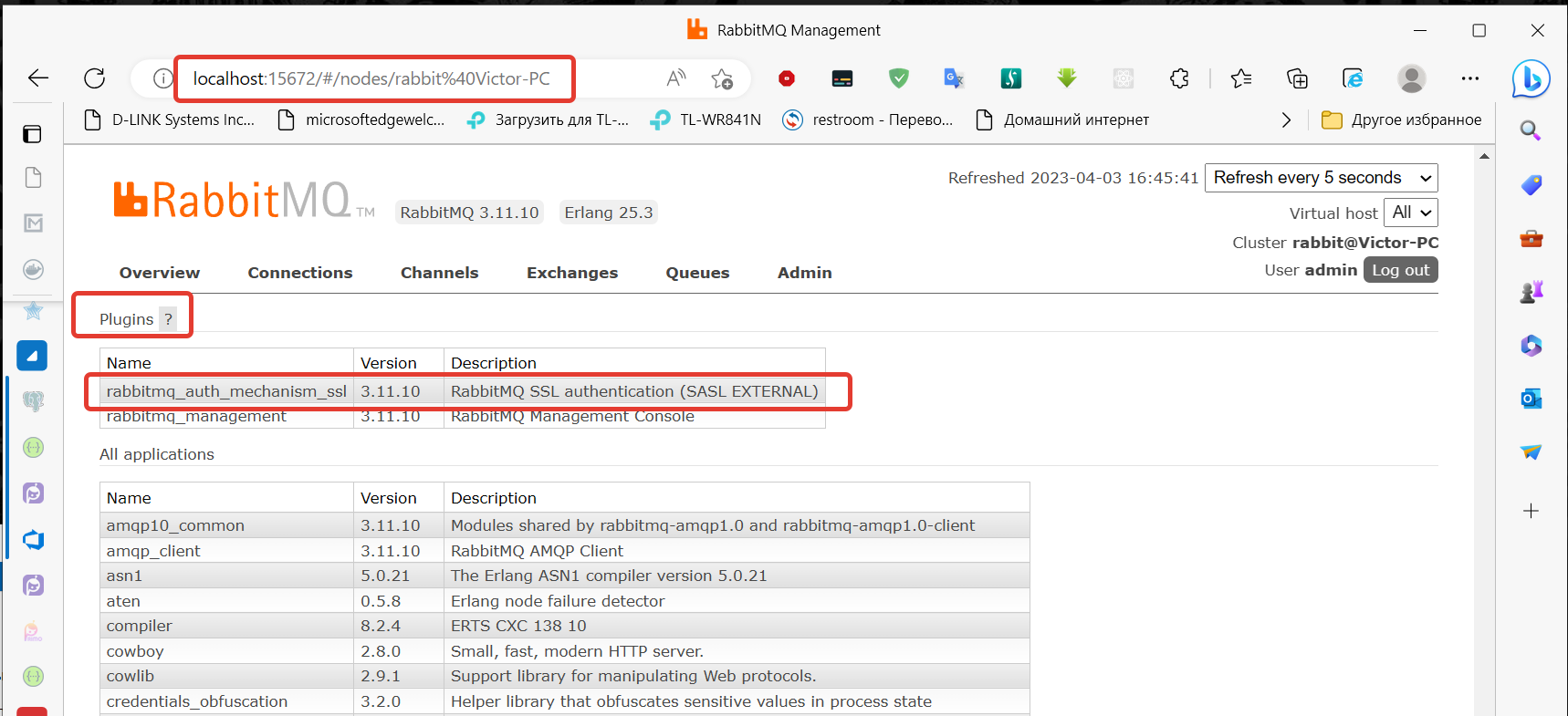
* 1. После этого необходимо перезагрузить сервис **RabbitMQ**. Сделать это можно выполнив в командной строке с правами **администратора** такие 2 команды:

|  |
| --- |
| rabbitmq-service stop  rabbitmq-service start |

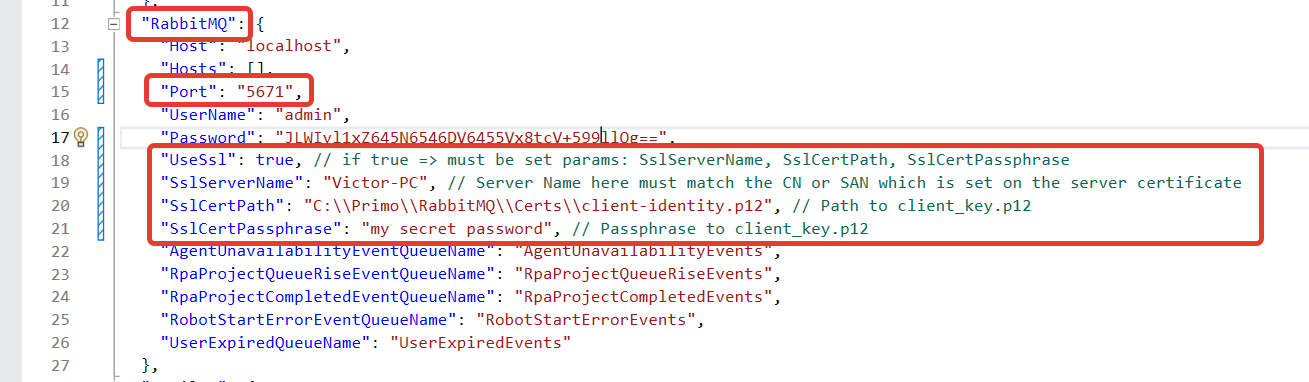
* 1. Теперь проверяем, что мы все сделали правильно и сервис **RabbitMQ** может работать через **SSL/TLS**. Запускаем **RabbitMQ management UI** и проверяем, что запущен SSL-порт **5671**:



Затем кликнув по кнопке с названием текущего узла, проверяем, что запущен плагин для работы через **SSL** и включен механизм аутентификации **EXTERNAL** :



1. **Настройка сервисов оркестратора для работы с RabbitMQ через SSL**
   1. Копируем специальный тип сертификата **PKCS12** (созданный ранее в **п.1.** настоящего руководства) в **удобный для Вас каталог** на машине, где развернут требуемый сервис оркестратора.
   2. Далее меняем конфигурационный файл сервиса (файл **appsettings.ProdWin.json** для машин семейства **Windows**, или **appsettings.ProdLinux.json** для машин семейства **Linux**). Для этого изменяем секцию RabbitMQ конфига таким образом:
      1. Для параметра Port **у**казываем значение 5671.
      2. Для параметра UseSsl **у**казываем значение true.
      3. Для параметра SslServerNameуказываем имя сервера, которое было указано в сертификате сервера для **RabbitMQ** (см. **п.1.** данного руководства).
      4. Для параметра SslCertPath указываем полный путь и имя клиентского сертификата **PKCS12**.
      5. Для параметра SslCertPassphraseуказываем **парольную фразу**, которую мы задали для сертификата **PKCS12**, когда его создавали (см. **п.1.** данного руководства).
   3. Например, эти секции конфига для машин с ОС **Windows** могут выглядеть так:



* 1. Эти же секции конфига для машин с ОС **Linux** могут выглядеть так:

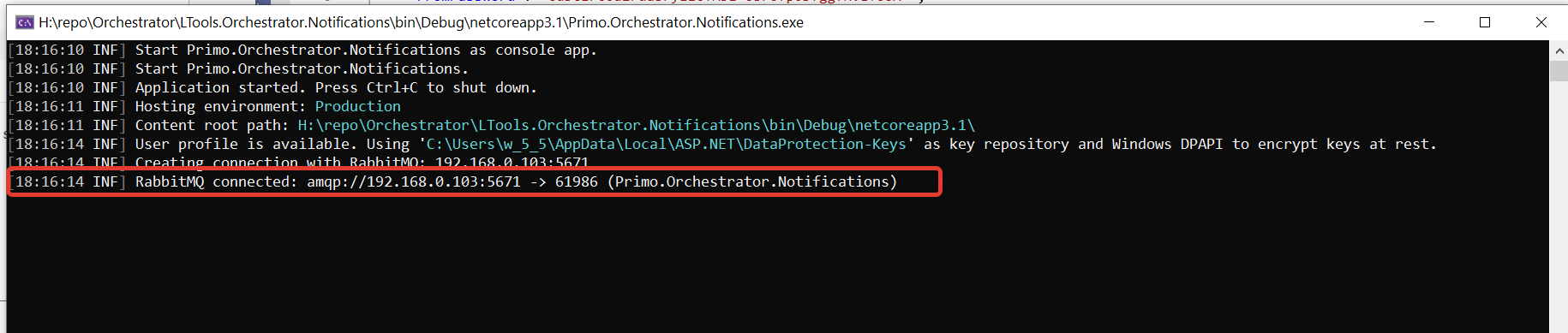


* 1. Перед тем как запустить сервис оркестратора, рекомендуется выставить в конфиге этого сервиса уровень логирования в **Information**, это поможет быстрее разобраться с возможными проблемами при его некорректном запуске.

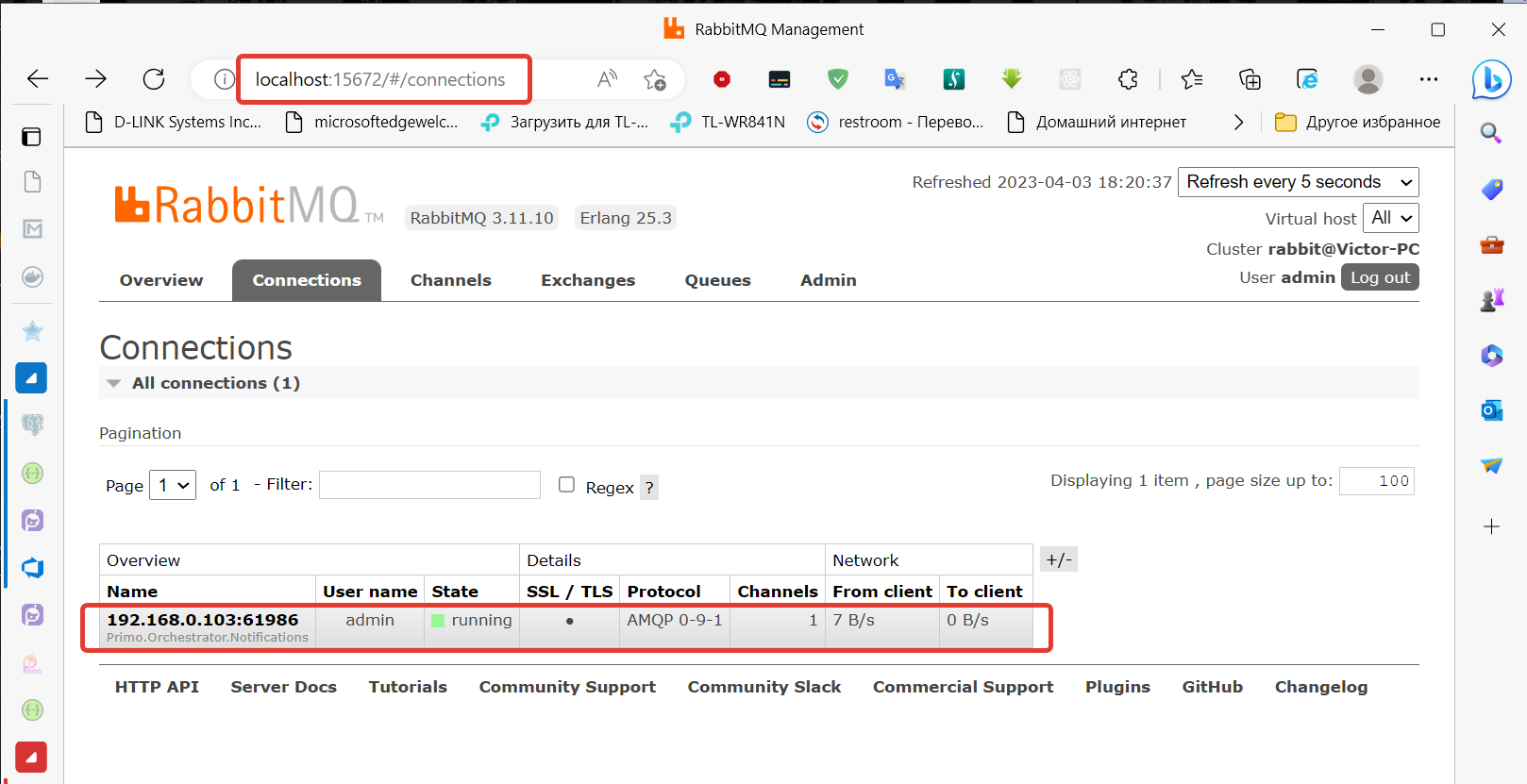
Например, как показано в данном случае:



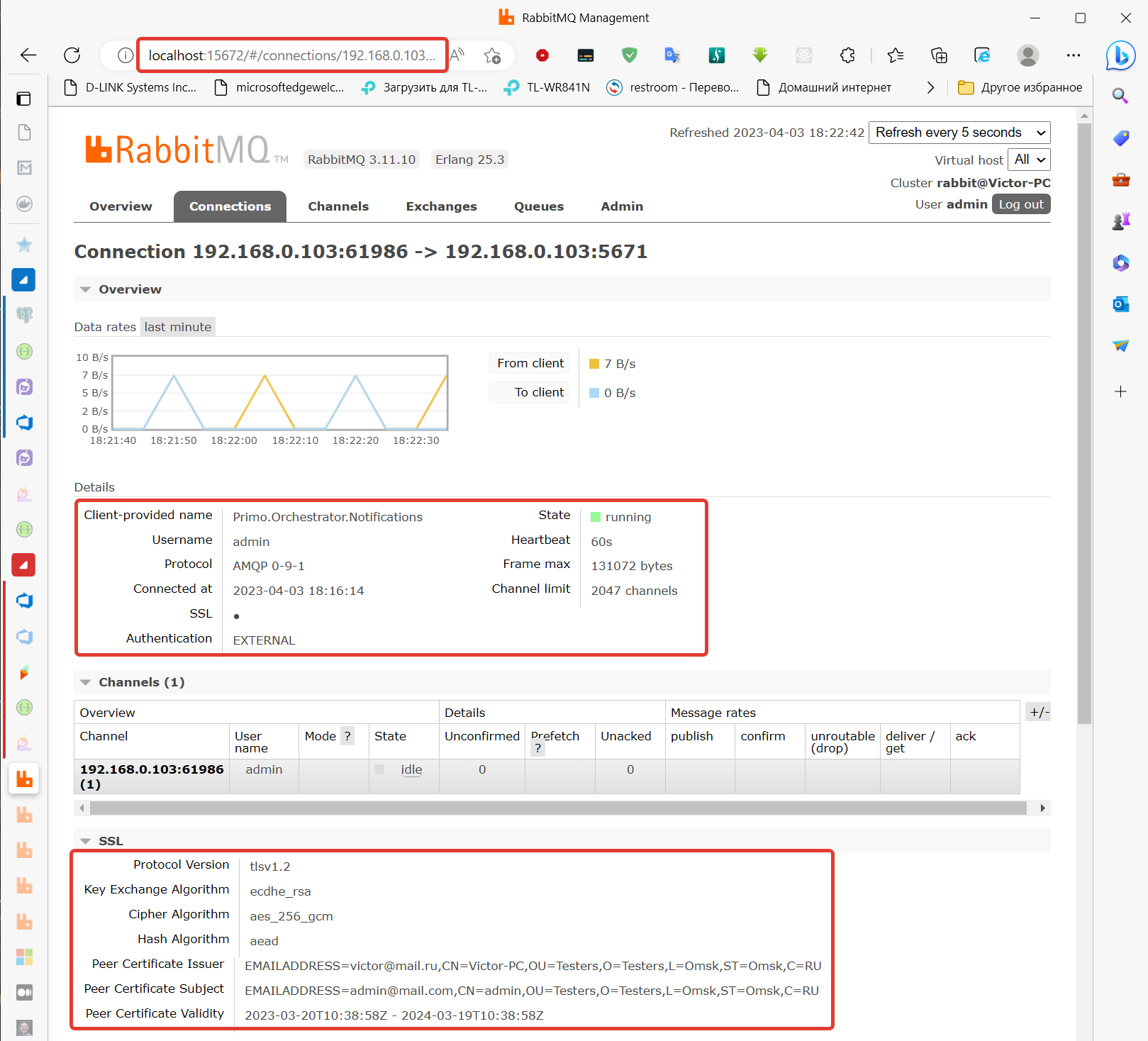
* 1. Если все сконфигурировано корректно, то после запуска сервиса в его логах отобразится запись, что коннект с **RabbitMQ** установлен:



При этом в **RabbitMQ management UI** на вкладке **Connections** можно будет увидеть вновь созданный коннект:

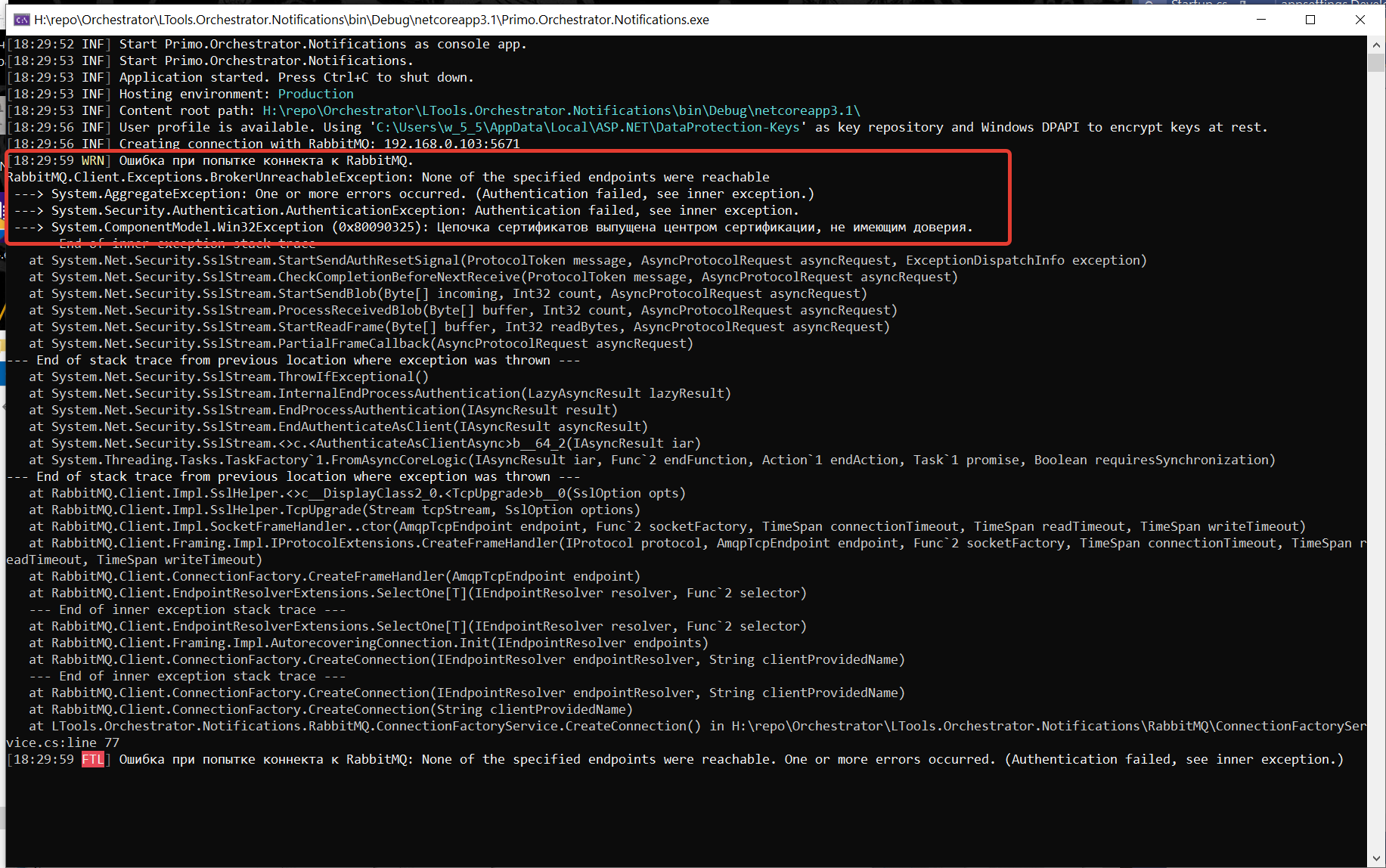


И если кликнуть по названию этого коннекта, то можно получить его детальные характеристики:

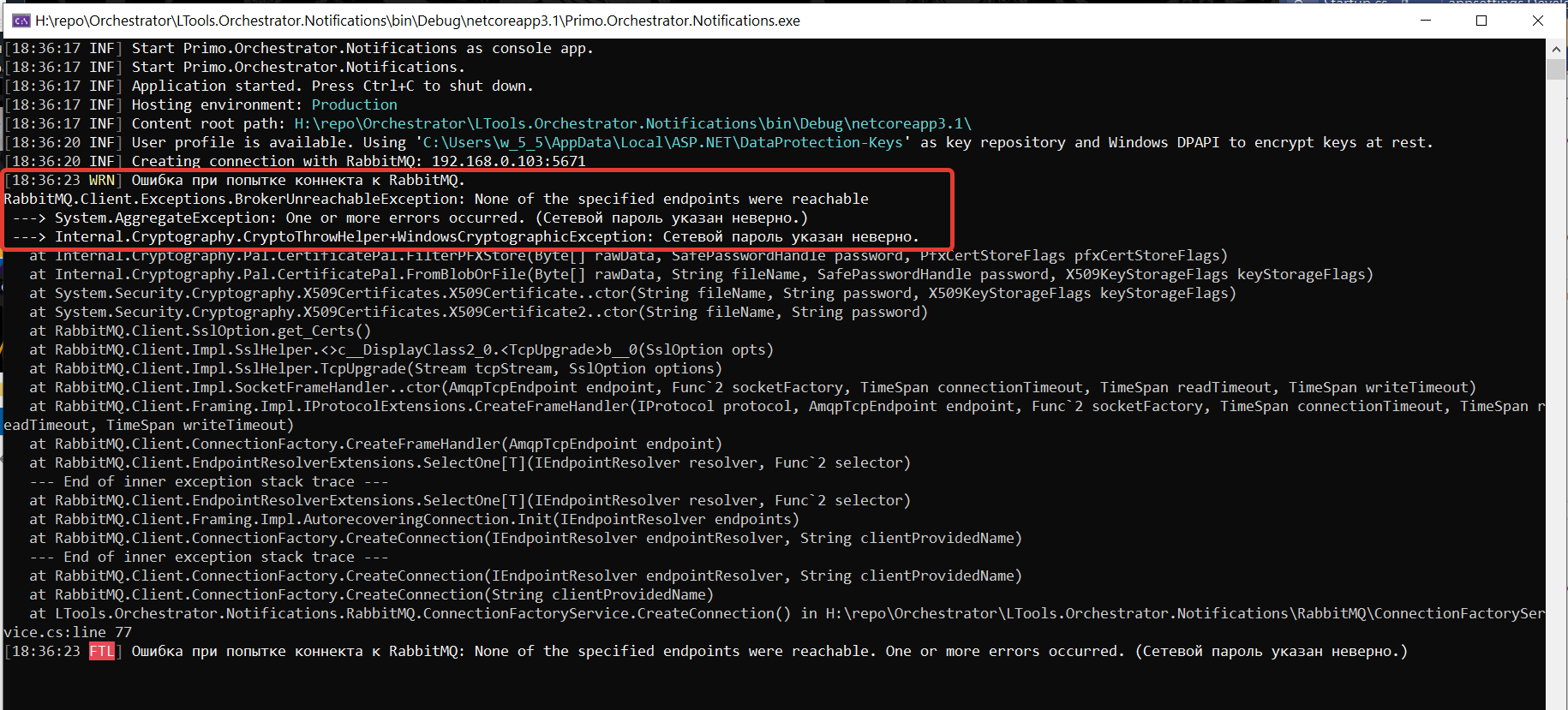


* 1. Если же в конфиге сервиса что-то не устроит **RabbitMQ**, то в логах будет отображена детальная информация о том, что конкретно не так, как требуется.

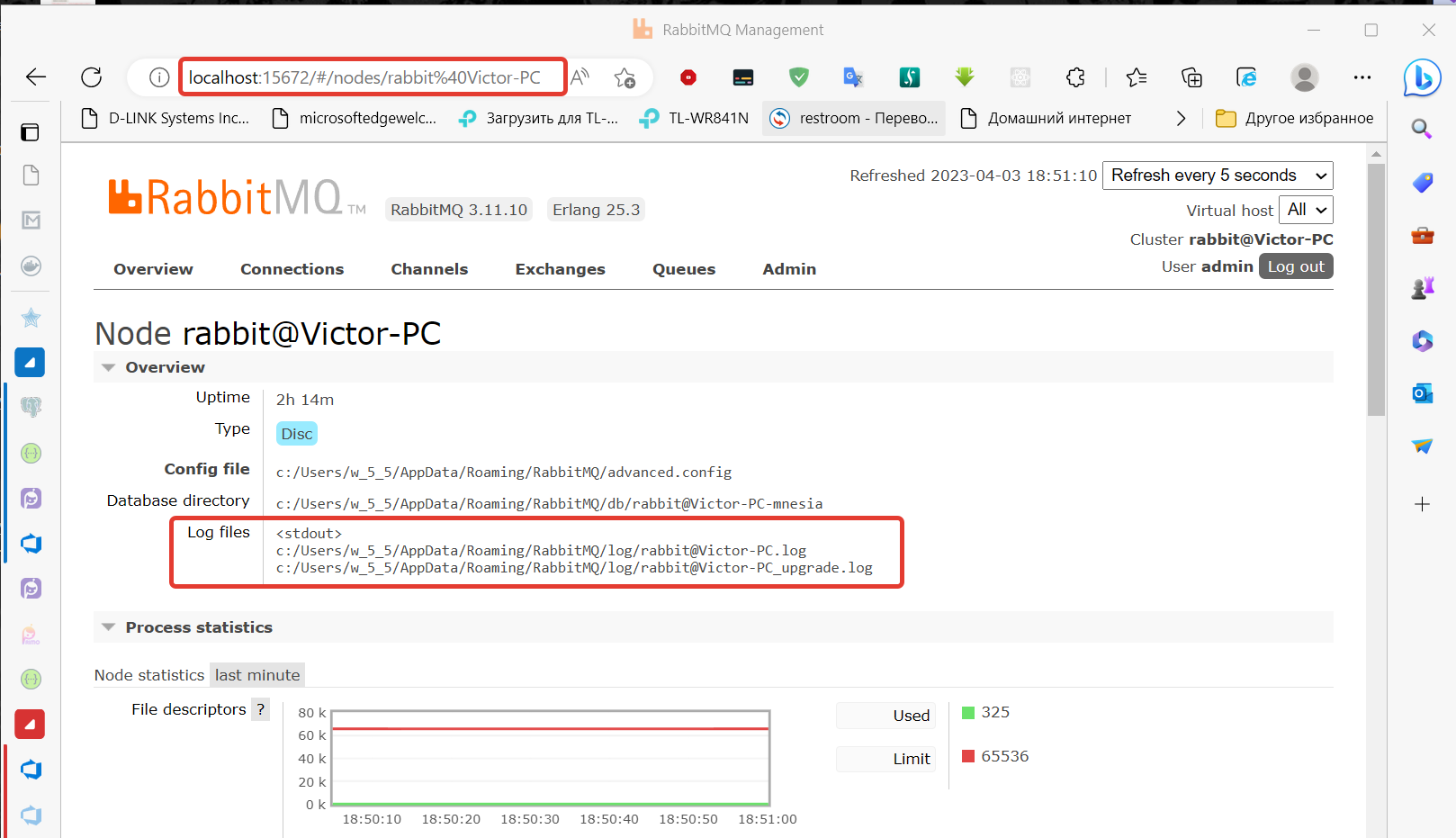
Например, вот что будет в логах, если в конфиге указан сертификат, подписанный каким-либо другим центром сертификации:



Или вот что будет в логах, если в конфиге указана некорректная парольная фраза для сертификата:



* 1. Если возникли совсем непонятные проблемы с **SSL**-сертификатом, то также будет полезно посмотреть логи самого **RabbitMQ**. Расположение этих логов можно узнать из **RabbitMQ management UI** кликнув по названию текущего узла:



1. **Полезные ссылки:**
   1. Сайт OpenSSL: <https://www.openssl.org/>
   2. Документация по сетевым настройкам RabbitMQ (включая SSL/TLS, на английском языке): <https://www.rabbitmq.com/networking.html>
   3. Документация по настройкам TLS в RabbitMQ (на английском языке): <https://www.rabbitmq.com/ssl.html>
   4. Описание кодов ошибок при работе с SSL/TLS (на английском языке): <https://techcommunity.microsoft.com/t5/iis-support-blog/ssl-tls-alert-protocol-and-the-alert-codes/ba-p/377132>
   5. Описание параметров rabbitmq.conf (на английском языке): <https://github.com/rabbitmq/rabbitmq-server/blob/main/deps/rabbit/docs/rabbitmq.conf.example>
   6. Описание механизма аутентификации с использованием x509 (TLS/SSL) сертификатов для RabbitMQ (на английском языке): <https://github.com/rabbitmq/rabbitmq-auth-mechanism-ssl>
   7. Реализация аутентификации RabbitMQ с использованием SSL/TLS по стандарту .NET Standard 2.1 (на английском языке): <https://www.cshelton.co.uk/blog/2019/12/18/rabbitmq-client-certificate-authentication/>