# Работа утилиты stunnel в режиме клиента на MAC OS X

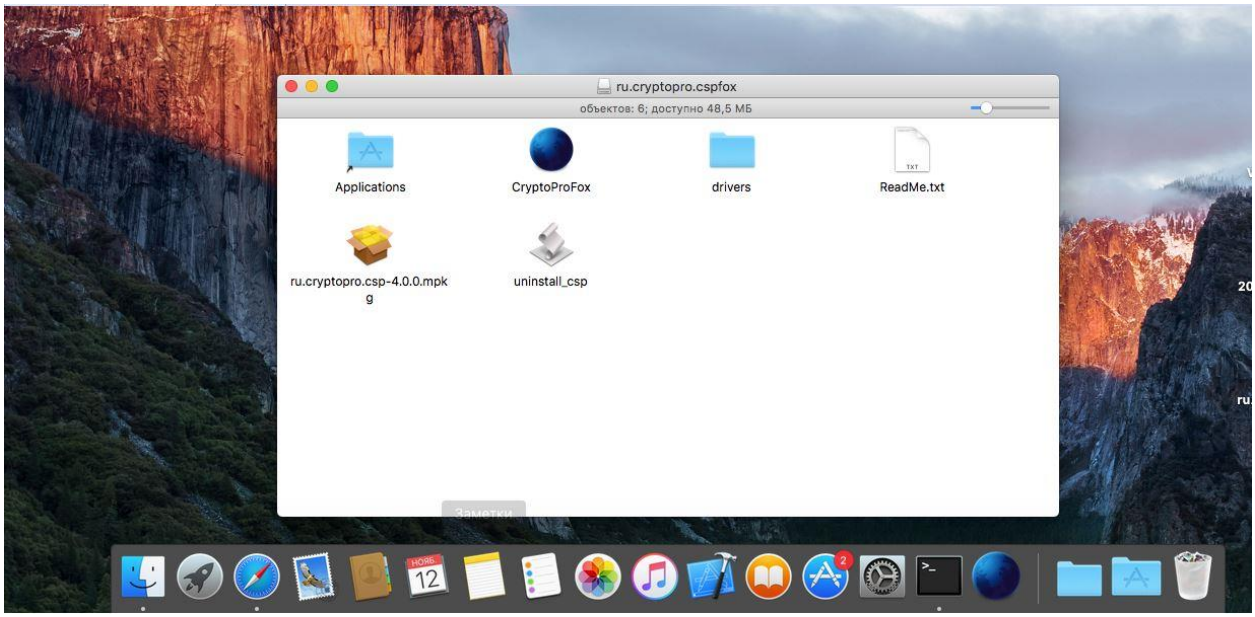
В данном документе будет пошагово описан процесс подготовки к работе и функционирования утилиты stunnel в режиме клиента в среде OS X El Capitan.

Утилита stunnel предназначена для шифрования трафика между приложениями, в которых данный функционал не был реализован. Возможна работа в режиме клиента и сервера. Stunnel, работающий в режиме клиента, принимает незашифрованный трафик от некоего клиентского ПО по указанному ip-адресу и порту, затем зашифровывает (трафик) и передаёт его на сервер. Если сервер поддерживает TLS (IIS, TrustedTLS и тд) – серверная часть stunnel'а не требуется. В режиме сервера, stunnel принимает на заранее определённый в конфигурационном файле ip-адрес и порт, зашифрованный трафик, затем расшифровывает его и передает соответствующему ПО на сервере.

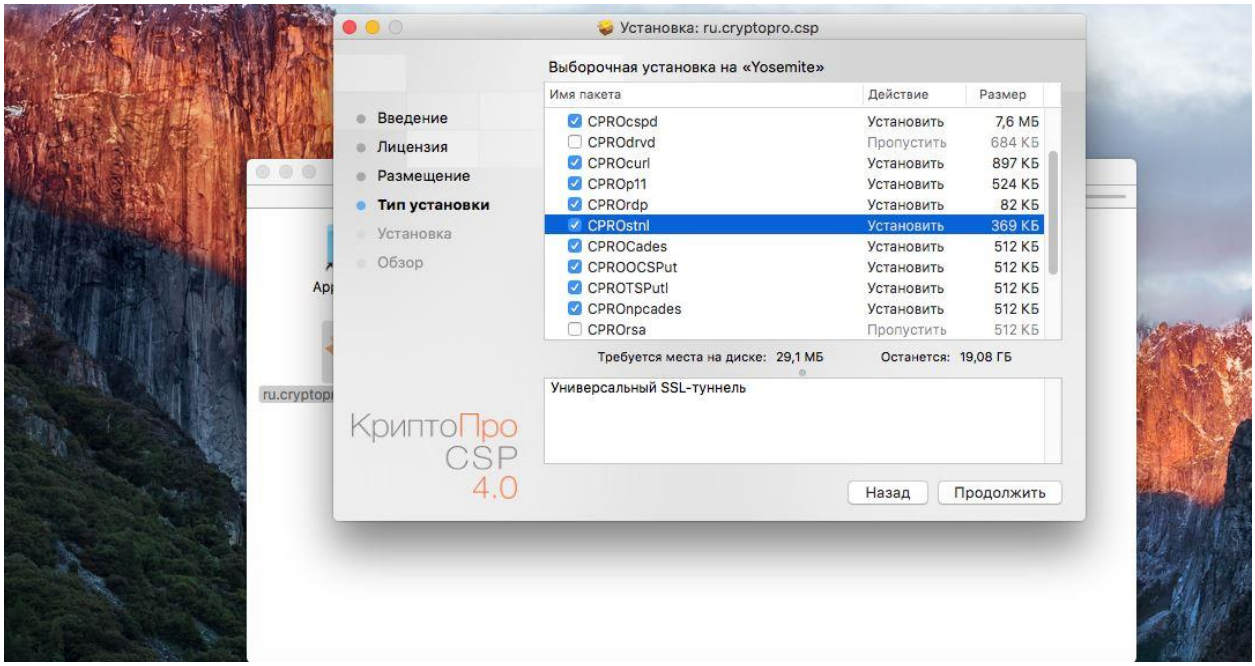
В используемом в Mac OS по умолчанию браузере Safari, отсутствует возможность шифрования трафика по ГОСТ алгоритмам и для того, чтобы обойти данное ограничение будет использоваться утилита stunnel, запущенная в режиме клиента. В качестве же сервера, для примера, будет использоваться сервис проверки работы двухстороннего TLS на сайте Крипто-ПРО https://www.cryptopro.ru:4444/test/tls-cli.asp.

Следует также отметить, что данную задачу можно решить и не прибегая к функционалу stunnel. Можно использовать браузер, в котором реализована поддержка ГОСТ изначально – CryptoProFox http://cryptopro.ru/products/cpfox

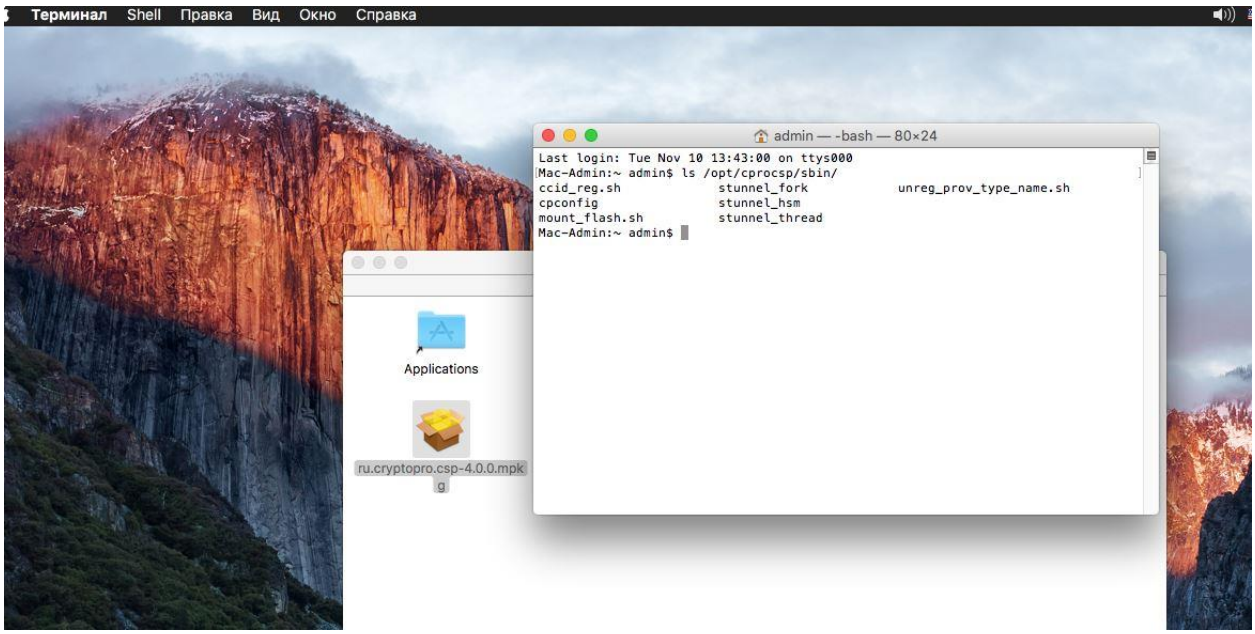
Утилита stunnel входит в дистрибутив Крипто-ПРО для MAC OS X, и должна быть выбрана для установки вместе с остальными пакетами.



Следует удостовериться в наличии “галки” напротив пакета CPROstnl

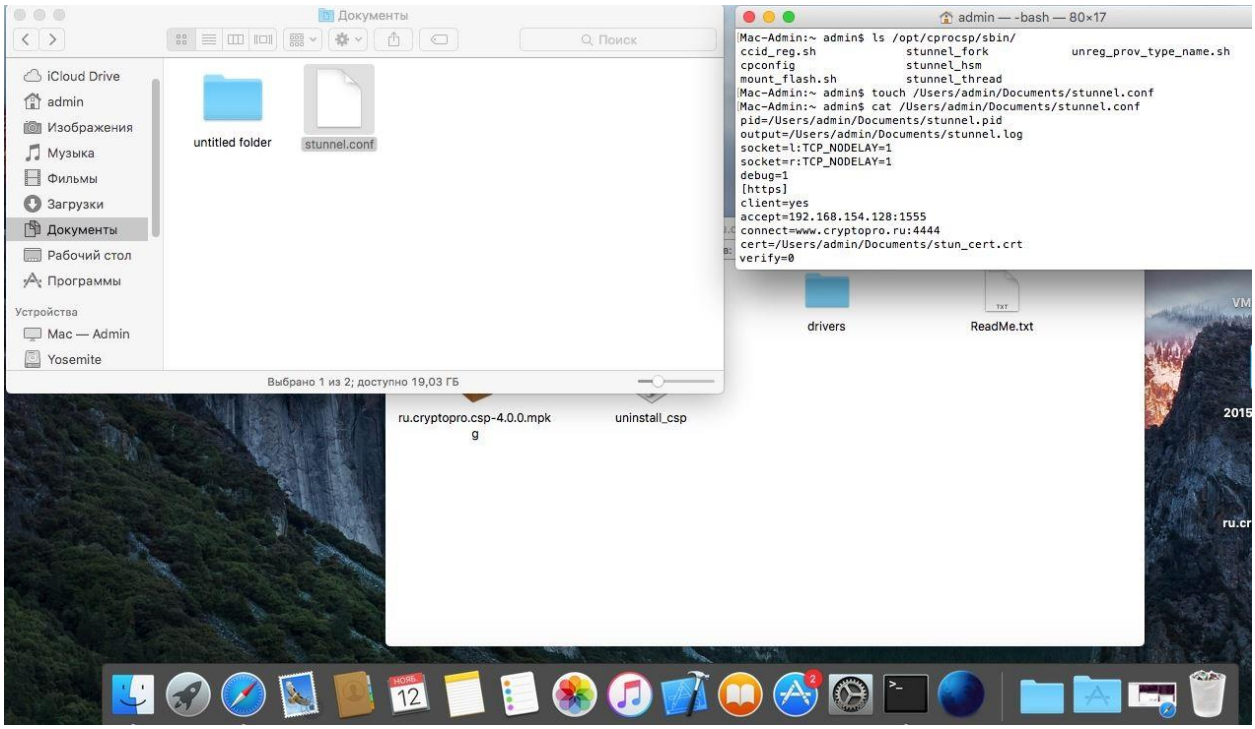



Завершив установку, можно проверить содержимое папки /opt/cprocsp/sbin




Использоваться будет stunnel_thread.

Следующим действием будет создание конфигурационного файла stunnel.conf в произвольном месте и внесения параметров запуска\работы утилиты.

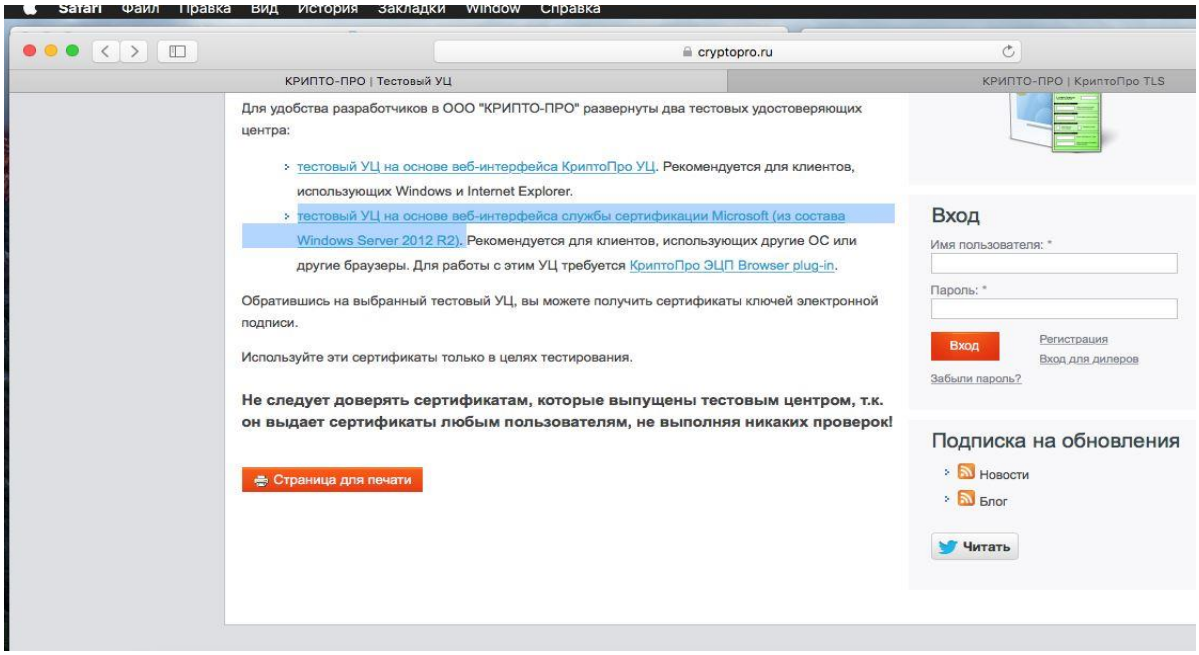



Краткое описание наиболее часто используемых параметров конфигурации.

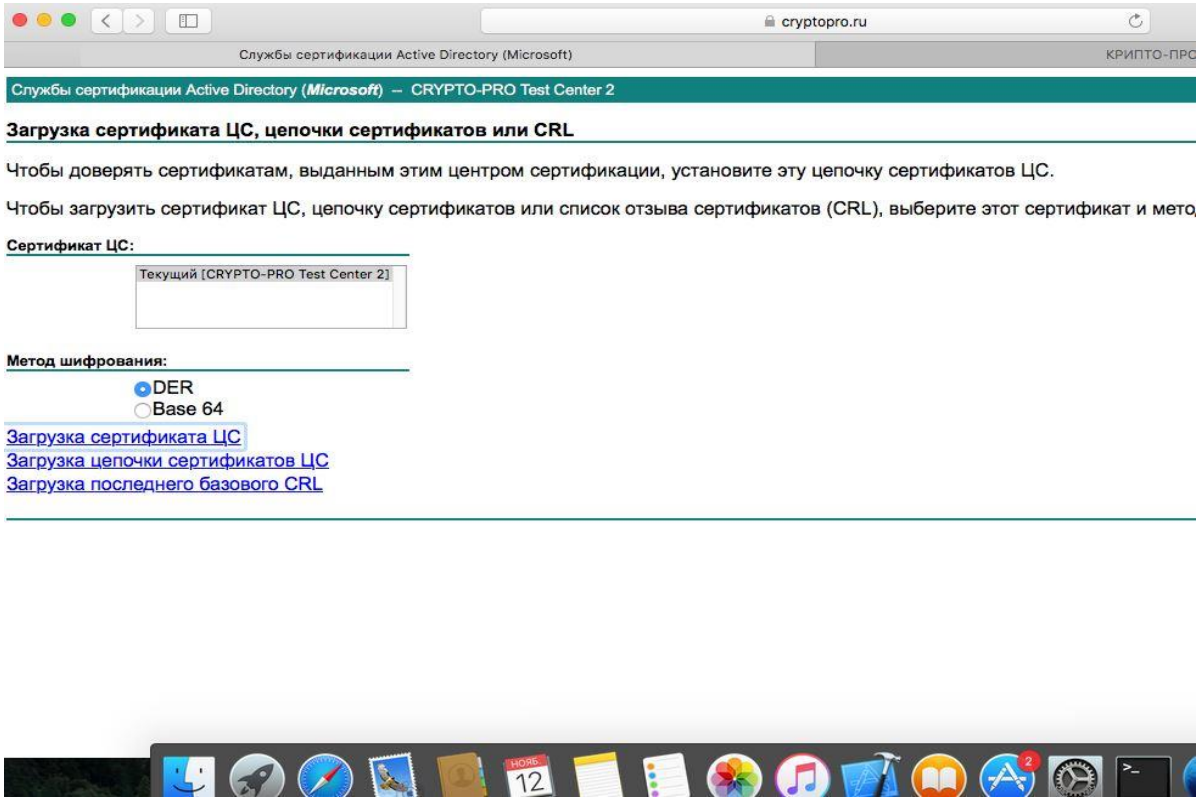
| Опция | Описание |
|---|---|
| pid | Путь к файлу, в котором будет храниться идентификатор процесса |
| output | Путь к лог-файлу |
| socket | Опции для конфигурирования принимающих, локальных, удалённых сокетов. |
| debug | Уровень протоколирования |
| client | Работа в режиме клиента |
| accept | Адрес и порт для приёма незашифрованного трафика |
| connect | Адрес и порт сервера, на который передаётся зашифрованный трафик |
| cert | Путь к файлу сертификата клиента |
| verify | Возможные варианты проверки сертификата удалённого сервера<br>0. Не проверять сертификат сервера<br>1. Проверять сертификат при его наличии<br>2. Проверять сертификат всегда<br>3. Проверять наличие данного сертификата в хранилище TrustedUsers |

Детальную документацию можно найти в /opt/cprocsp/share/man/man8

Ввиду того, что в примере рассматривается двусторонний TSL, для работы понадобится клиентский сертификат, который к примеру можно выпустить средствами тестового удостоверяющего Крипто-ПРО https://www.cryptopro.ru/certsrv/ Нужно также добавить, что этот тестовый удостоверяющий центр, выпустивший личный сертификат является доверенным для сервера, к которому производится подключение.

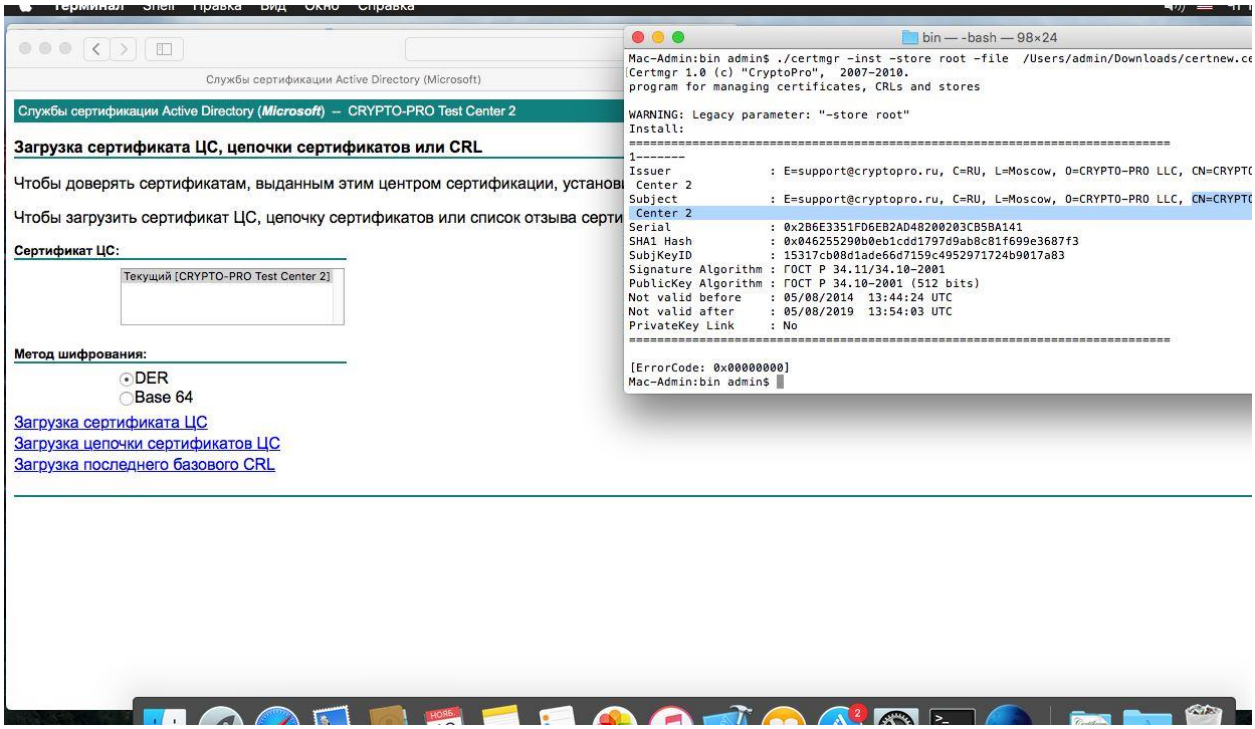



Сперва следует скачать корневой сертификат данного УЦ.

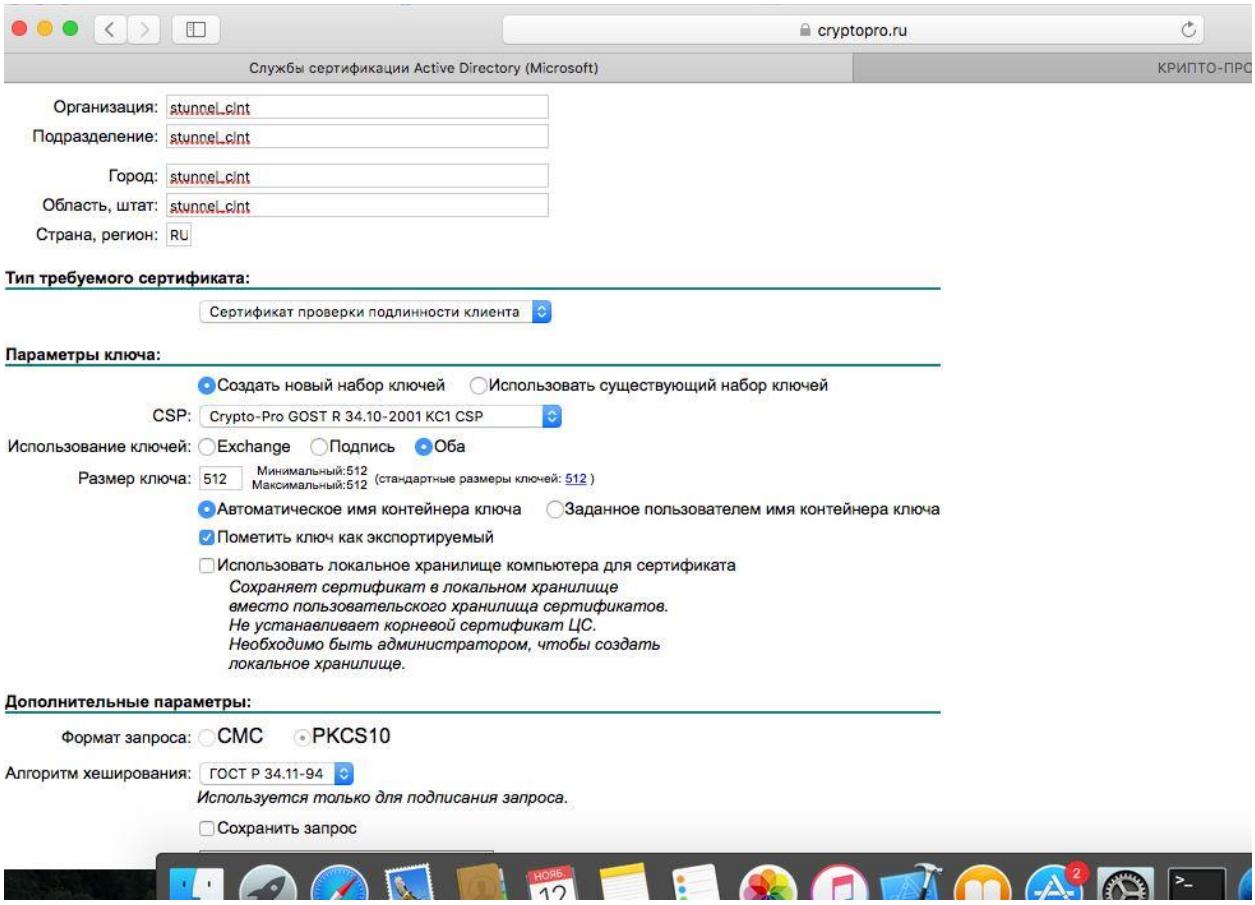


Установка корневого сертификата производится командой

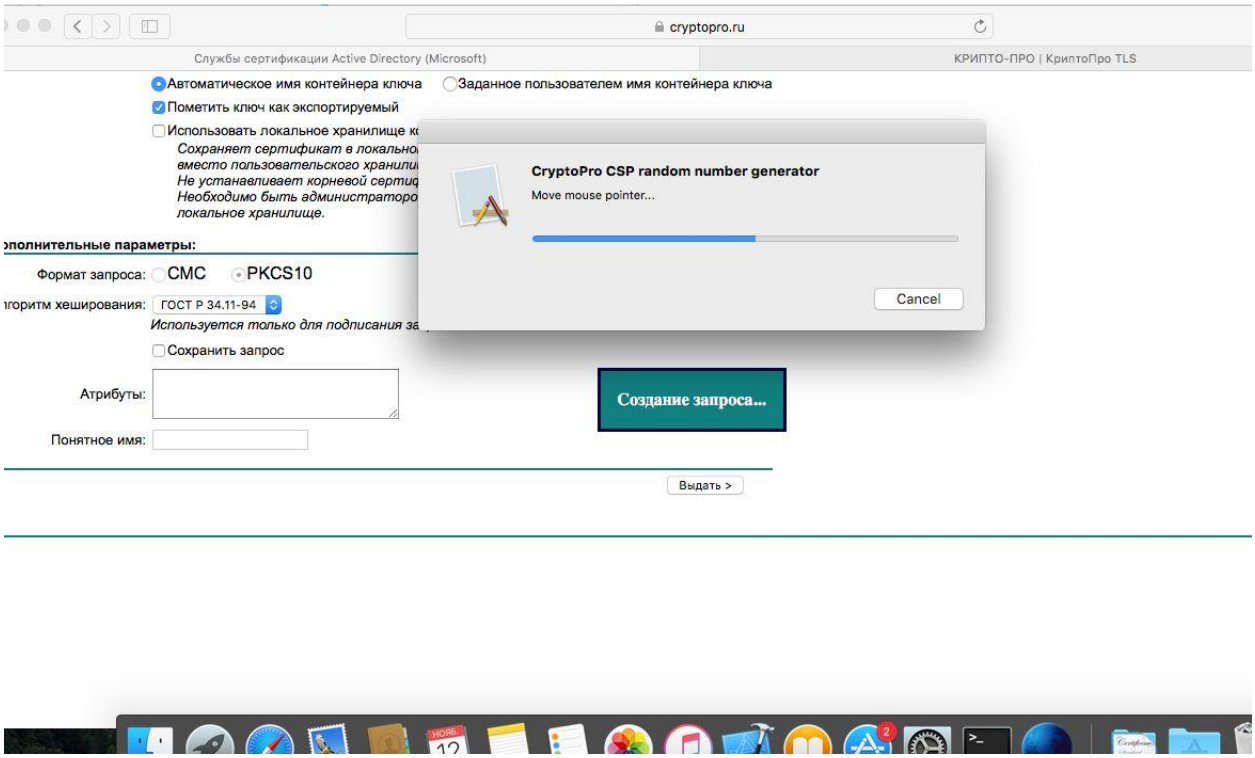
./certmgr –inst –store –root –file /Users/admin/Downloads/certnew.cer




Далее производится генерация контейнера.

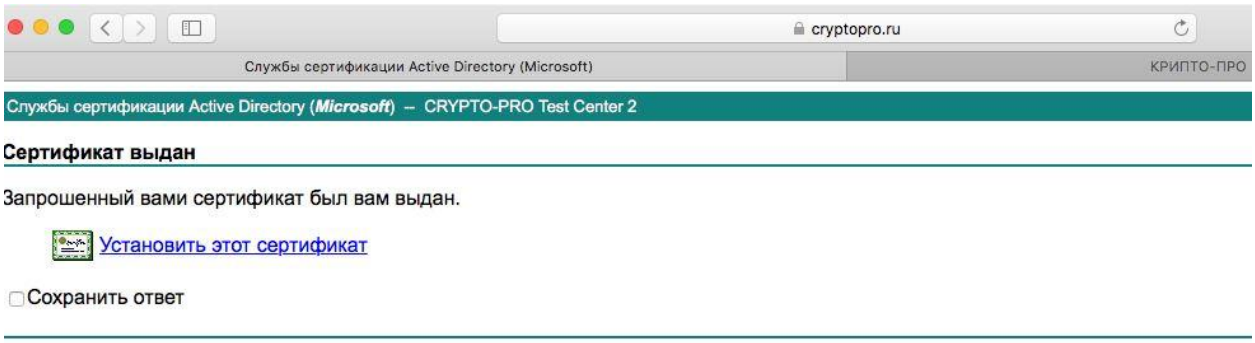


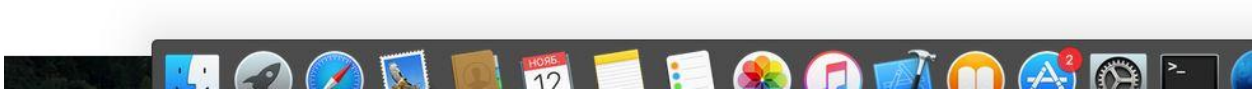
Интерфейс БиоДСЧ.  При необходимости для контейнера задаётся PIN.





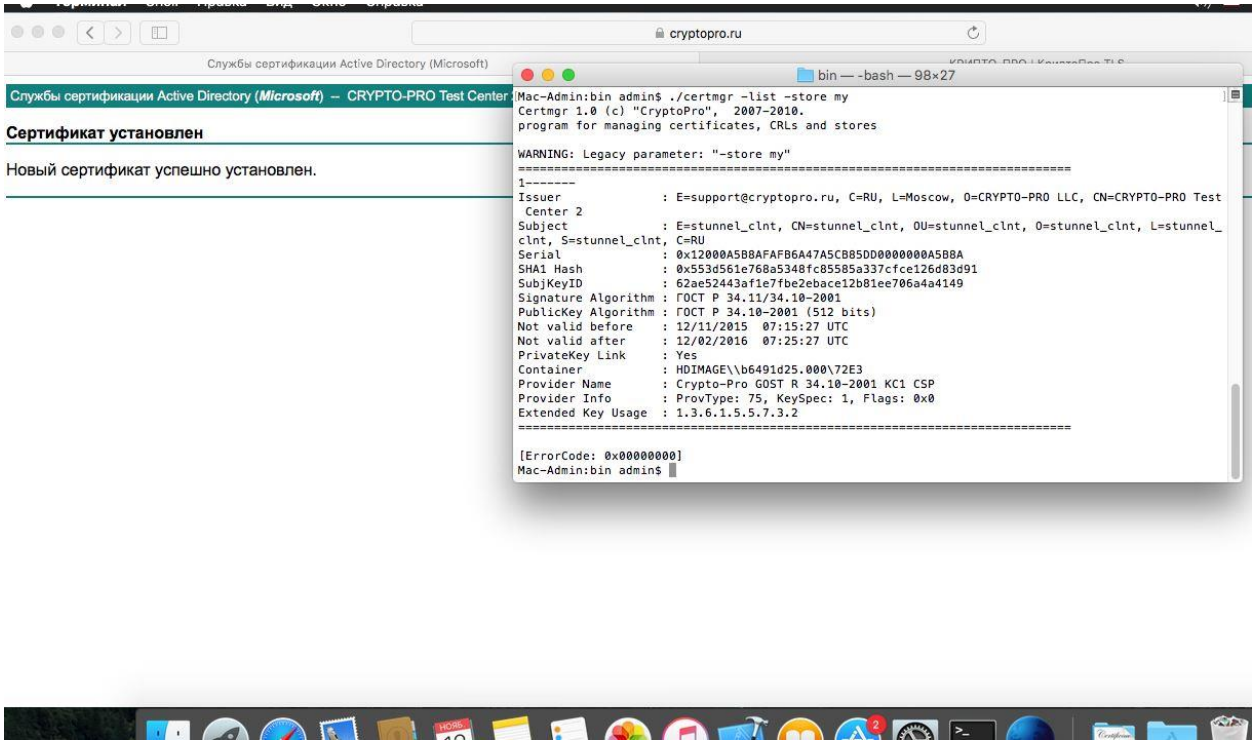

Производится установка сертификата.



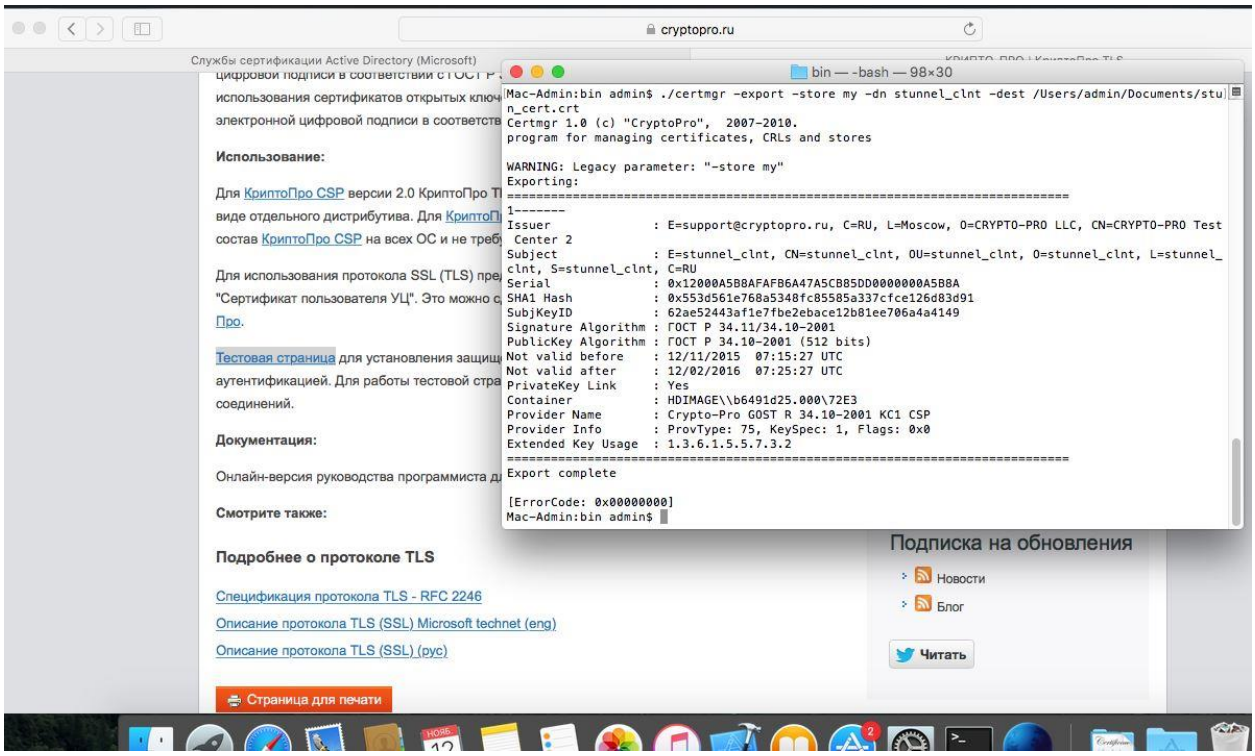
При желании, после установки, можно удостовериться в наличии сертификата в хранилище командой

./certmgr –list –store my

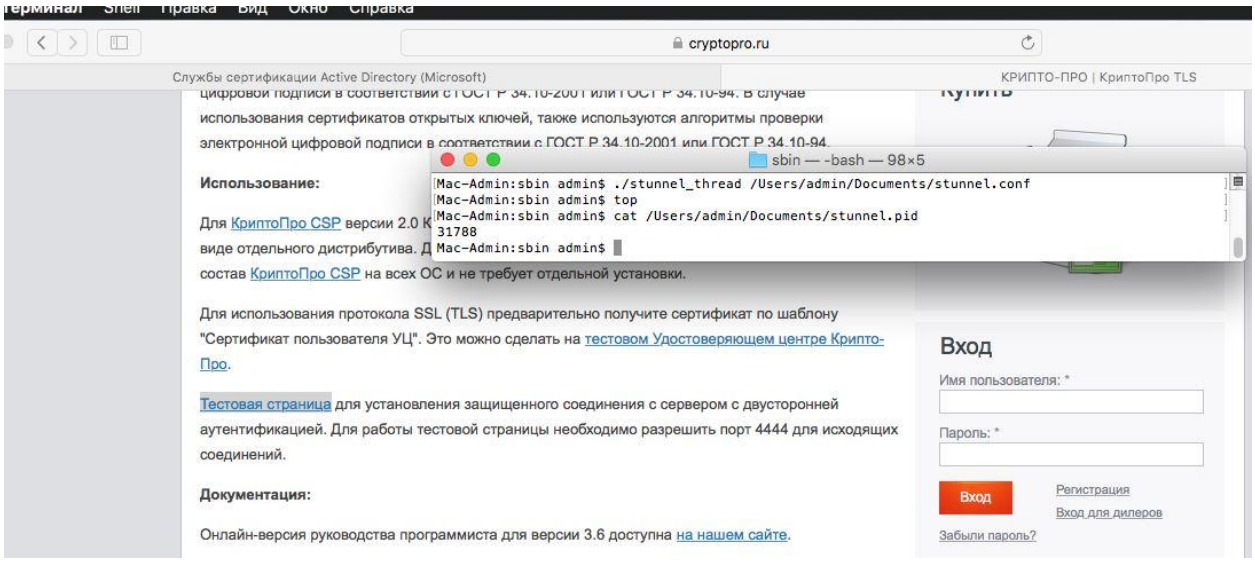



Далее производится экспорт личного сертификата в файл, путь к которому указан в файле конфигурации, используя команду

./certmgr –export –store my –dn stunnel_clnt –dest /Users/admin/Documents/stun_cert.crt

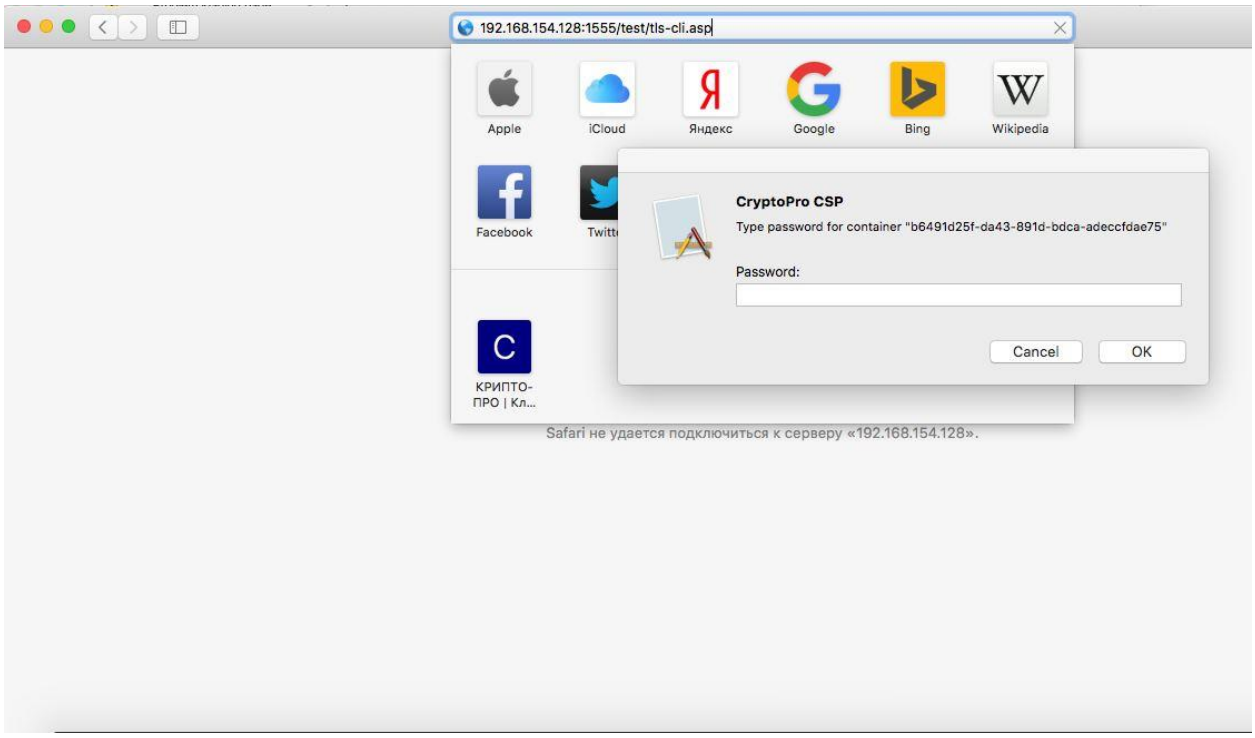


Запуск утилиты с указанием пути к конфигурационному файлу и просмотр идентификатора запущенного процесса производятся командами /stunnel_thread /Users/admin/Documents/stunnel.conf и cat /Users/admin/Documents/stunnel.pid соответственно.

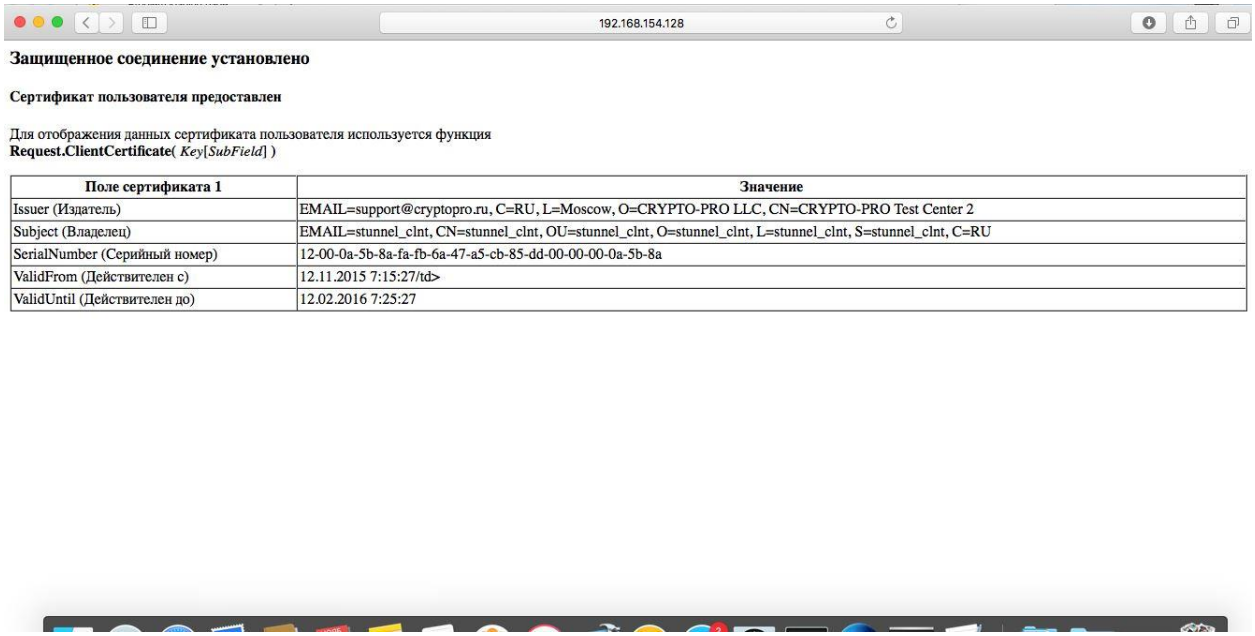



Следует сделать пояснение: рассматриваемый случай не подразумевает проверку сертификата сервера, на который организуется доступ (verify=0). Если указанная проверка всё же требуется, можно указать значение опции verify = 1 или 2 Тогда следует скачать и установить в хранилище root корневой сертификат удостоверяющего центра, выпустившего сертификат веб-сервера https://www.cryptopro.ru/ui/ и

Далее производится попытка подключения. Вводится PIN на контейнер, если при генерации оный был установлен.



При успешном соединении отображается следующее окно.




## Ошибки в работе

В случае возникновения проблем с подключением следует изучить содержимое файла журнала событий, который создаётся автоматически при первом запуске службы по указанному в конфигурационном файле пути. При некорректно сформированном конфигурационном файле утилита не запустится, при этом файлы журнала и идентификатора процесса могут не создаваться.

При отсутствии в хранилище корневого сертификата веб-сервера к которому производится подключение (если в конфигурационном файле имеется требование проверки), в журнале событий отобразится сообщение следующего плана:

```
16:12:39 LOG3[927:123145307148288]: Error 0x20 ((unknown)) returned by CertVerifyCertificateChainPolicy!
16:12:39 LOG3[927:123145307148288]: Error 0x20 when validate certificate

16:12:39 LOG3[927:123145307148288]: Error 0x80092004 returned by VeryfySertChain
16:12:39 LOG3[927:123145307148288]: **** Error 0x80092004 authenticating server credentials!
16:12:39 LOG5[927:123145307148288]: Connection reset: 0 byte(s) sent to SSL, 0 byte(s) sent to socket
16:12:39 LOG7[927:123145307148288]: free Buffers
16:12:39 LOG7[927:123145307148288]: delete c->hContext
16:12:39 LOG7[927:123145307148288]: delete c->hClientCreds
16:12:39 LOG5[927:123145307148288]: incomp_mess = 0, extra_data = 0
16:12:39 LOG7[927:123145307148288]: Remote socket (FD=17) closed
16:12:39 LOG7[927:123145307148288]: Local socket (FD=10) closed
16:12:39 LOG7[927:123145307148288]: Service [https] finished (1 left)
16:12:39 LOG7[927:123145307148288]: str_stats: 6 block(s), 768 data byte(s), 444 control byte(s)
16:12:39 LOG7[927:123145307148288]: str_stats: 128 byte(s) at /dailybuildsbranches/CSP_4_0/CSPbuild/CSP/sa
```

В случае ввода неправильного PIN от контейнера некоторого количества раз – соединение не удастся, а в журнале отобразится следующая ошибка.

```
16:24:40 LOG5[958:123145307148288]: 1405 bytes of handshake(in handshake loop) data received.
16:24:44 LOG5[958:123145307148288]: 935 bytes of handshake data sent
16:24:44 LOG3[958:123145307148288]: **** Error 0x80090304 returned by InitializeSecurityContext (2)
16:24:44 LOG3[958:123145307148288]: Couldn't complete RENEGOTIATE
16:24:44 LOG3[958:123145307148288]: error on SSPI_read
16:24:44 LOG5[958:123145307148288]: Connection reset: 340 byte(s) sent to SSL, 0 byte(s) sent to socket
16:24:44 LOG7[958:123145307148288]: free Buffers
16:24:44 LOG7[958:123145307148288]: delete c->hContext
```