LS11CLOUD – облачная реализация PKCS#11 v2.40 с поддержкой ГОСТ Р 34.10-2012, ГОСТ Р 34.11-2012, ГОСТ Р 34.12-2015 и ГОСТ Р 34.13-2015

Руководство Пользователя

21 декабря 2018 г.

Оглавление

1 Общее описание

LS11CLOUD является облачной реализацией ООО "ЛИССИ-Софт" [1] стандарта PKCS#11 [15], дополненного поддержкой российских криптографических алгоритмов в соответствии со спецификациями, выработанными Техническим комитетом по стандартизации (ТК 26) "Криптографическая защита информации" [2, 10]. LS11CLOUD поддерживает алгоритмы ГОСТ Р 34.10-2012 [5], ГОСТ Р 34.11-2012 [9], ГОСТ Р 34.12-2015 [6] и ГОСТ Р 34.13-2015 [7], а также сопутствующие алгоритмы и параметры, определенные руководящими документами ТК 26.

Обеспечение безопасного удаленного взаимодействия с защищенным личным контейнером криптографических объектов (токеном) по шифрованному сетевому каналу осуществляется с применением протокола аутентификации SESPAKE (Security Evaluated Standardized Password-Authenticated Key Exchange) [13], рекомендованному ТК 26.

На стороне пользователя основная функциональность обеспечивается динамической библиотекой ls11cloud со стандартным программным интерфейсом PKCS#11. Предварительная регистрация пользователя на сервере и обслуживание учетной записи производятся утилитой `ls11cloud_config`. Удаленная инициализация и конфигурирование личного токена на сервере выполняются утилитой p11conf, работающей через интерфейсы библиотеки ls11cloud.

1.1 Особенности реализации

Аутентификация пользователя и сервера по протоколу SESPAKE производится при каждой инициализации библиотеки ls11cloud. Кроме того, сетевой доступ к облачному токену также требует определенных затрат времени на передачу данных по сети. Эти затраты следует учитывать при использовании облачного токена в прикладных программах.

Пароль учетной записи пользователя и значения PIN его токена никак не связаны между собой. Пароль учетной записи используется для двусторонней аутентификации пользователя и сервера по протоколу SESPAKE. А значения PIN предназначены для доступа к токену. В то же время, использование пароля учетной записи и PIN токена усиливает безопасность облачного токена.

Введенные пароли учетной записи в открытом виде никогда не сохраняются и по сети не передаются.

Все сообщения, передаваемые между клиентом и сервером, шифруются уникальными ключами, вырабатываемыми заново в каждой сессии протокола аутентификации SESPAKE. Шифрование производится с использованием алгоритма "Кузнечик" в режиме CTR [7].

Если по каким-то причинам сервер не доступен, функции библиотеки ls11cloud будут завершаться с ошибкой `CKR_TOKEN_NOT_PRESENT`.

Приватные объекты токена шифруются на сервере с использованием алгоритма "Кузнечик" в режиме CFB [7] . Случайное значение мастер-ключа шифрования, в свою очередь, шифруется ключом, сгенерированным на USER PIN и защищается имитовставкой.

Значения закрытых ключей на токене с атрибутами неизвлекаемости никогда не покидают сервер. Все операции с такими объектами выполняются на сервере, а пользователю передаются только результаты выполнения операций. Большинство других операций выполняется локально на стороне пользователя библиотекой ls11cloud для обеспечения высокой производительности.

Данная документация соответствует LS11CLOUD версии 3.0. Некоторые механизмы и другие конструкции PKCS#11 для новых российских алгоритмов пока еще не утверждены в ТК 26 окончательно, поэтому в следующие версии проекта могут быть внесены соответствующие изменения.

2 Установка

Библиотека ls11cloud и утилиты системы устанавливаются на компьютере пользователя инсталлятором. Кроме того, инсталлятор создает служебную папку ls11cloud в домашней папке пользователя. В процессе установки формируется начальное значение криптографического генератора случайных чисел, при котором пользователю предлагается нажимать соответствующие клавиши по запросу инсталлятора. Это начальное значение сохраняется в файле `prng_start.bin` в папке ls11cloud.

Тестовый проект `ls11cloud_tests` инсталлятором не устанавливается и поставляется отдельно в виде архива.

3 Регистрация и учетная запись

Учетная запись создается на сервере. Кроме того, в локальной папке ls11cloud создаются файлы, необходимые для подключения к серверу. Операции с учетной записью выполняются утилитой `ls11cloud_config`.

3.1 Утилита обслуживания учетной записи

```
LS11CLOUD User Utility
Usage:
ls11cloud_config <command> [-p <password>] [-n <new password>]
NB: Use -n <new password> with change_pswd command only!
Commands:
  register <host> <port> <id> - register new user on the server
  duplicate <host> <port> <id> - duplicate user account on other computer
  change_pswd - change account password
  status      - display current configuration data
  log         - display server log file
  recreate    - re-create token to initial empty state
  unregister  - remove all user files from the server
  save_pswd_hash - save account password hash to local file
NB: Don't use non-latin letters to avoid encoding problems!
Copyright(C) LISSI-Soft, Ltd (http://soft.lissi.ru) 2017-2019
```

3.1.1 Регистрация пользователя

Пользователь должен зарегистрироваться на сервере с уникальным идентификатором длиной не менее 6-ти символов. Желательно не использовать русские буквы в идентификаторе и пароле, во избежание расхождений в кодировке при подключении к серверу из различных систем. Идентификатор пользователя ls11cloud не обязан совпадать с именем пользователя, с которым он работает на данном компьютере. Регистрация производится командой register утилиты `ls11cloud_config`:

```
>ls11cloud_config register <host> <port> <id>,
где
<host> - сетевое имя или IP-адрес сервера ls11cloud
<port> - номер порта на сервере
<id> - идентификатор пользователя
```

Пример запуска регистрации пользователя vblazhnov на тестовом сервере pkcs11.ru с номером порта 4444:

```
>ls11cloud_config register pkcs11.ru 4444 vblazhnov
```

В процессе работы утилита дважды запросит у пользователя начальное значение пароля (не менее 6-ти символов), с которым он в дальнейшем будет подключаться к серверу по протоколу SESPAKE. Утилита также запоминает параметры подключения в локальной папке ls11cloud.

Поскольку первоначально на сервере нет пароля пользователя, то защищенное соединение по протоколу SESPAKE производится с паролем, равным идентификатору пользователя. Как только такое соединение установлено, по нему на сервер передается зашифрованное значение пароля пользователя, и соединение по протоколу SESPAKE переустанавливается заново уже с паролем, заданным пользователем. Таким образом, значение пароля пользователя никогда не передается серверу в открытом виде.

Заметим, что значение пароля не сохраняется, поэтому библиотека ls11cloud будет каждый раз запрашивать это значение в диалоге для подключения к серверу при инициализации сеанса PKCS#11. Если пользователь работает в защищенной среде, он может с помощью утилиты `ls11cloud_config` сохранить значение хэша от пароля в файле `ls11cloud/sdata.bin`, чтобы не вводить его при каждом запуске. В этом случае и библиотека, и утилита `ls11cloud_config` не будут запрашивать пароль в диалоге, а будут использовать хэш из файла.

```
>ls11cloud_config save_pswd_hash -p 01234567
LS11CLOUD User Utility
command: save_pswd_hash
OK
```

В дальнейших примерах мы для простоты будем предполагать, что хэш пароля уже сохранен в локальном файле.

В случае успешного подключения к серверу, на нем будет создана учетная запись и пустой личный токен для хранения объектов с начальным значением SO PIN 87654321. Такой токен требует конфигурирования, которое описано в разделе 4 .

3.1.2 Дублирование учетной записи

Доступ к существующей учетной записи с другого компьютера можно организовать с помощью команды duplicate с теми же параметрами, которые использовались при регистрации, например:

```
>ls11cloud_config duplicate pkcs11.ru 4444 vblazhnov
```

Утилита запросит пароль SESPAKE, подключится к серверу, произведет двустороннюю аутентификацию и создаст соответствующие локальные файлы конфигурации на данном компьютере.

В доверенной среде значение пароля SESPAKE можно также задавать в командной строке утилиты после флага -p.

3.1.3 Проверка учетной записи

Проверить состояние учетной записи можно командой status утилиты `ls11cloud_config`:

```
>ls11cloud_config status
LS11CLOUD User Utility
command: status
Using SESPAKE password from config.txt file
Try to connect to server at pkcs11.ru:4444
Connected to server
----------------------------------------
A: Successful SESPAKE authentication of B
----------------------------------------
==== Server config.txt ====
id = "vblazhnov"
c1 = 5 - sequential invalid password attemts remain
c2 = 10 - overall invalid password attempts remain
c3 = 99964 - overall connections with current password remain
last_date_time = 08/06/2023 10:40:48 - account expiration date and time
----------------------------------
Token Info:
----------------------------------
Label: vblazhnov
Manufacturer: LISSI-Soft, Ltd
Model: LS11CLOUD
Serial Number: 2370F1197131E3B0
Flags: 0x40D
 ( CKF_RNG|CKF_LOGIN_REQUIRED|CKF_USER_PIN_INITIALIZED|CKF_TOKEN_INITIALIZED )
Sessions: 0/256
R/W Sessions: 0/256
PIN Length: 4-32
Public Memory: CK_UNAVAILABLE_INFORMATION/CK_UNAVAILABLE_INFORMATION
Private Memory: CK_UNAVAILABLE_INFORMATION/CK_UNAVAILABLE_INFORMATION
Hardware Version: 3.0
Firmware Version: 3.0
Time: 13:09:04
----------------------------------

OK
```

По запросу status дополнительно выдается информация о токене пользователя, по которой можно судить о его состоянии.

Согласно протоколу SESPAKE [13], значения c1, c2 и c3 содержат соответственно оставшееся до блокировки учетной записи количество неудачных попыток ввода

пароля подряд, оставшееся общее количество неудачных попыток и оставшееся количество сеансов с текущим значением пароля. При изменении пароля SESPAKE, значения этих констант устанавливаются в начальные значения c1 = 5, c2 = 20, c3 = 100000.

При каждой неудачной попытке ввода пароля подряд значение c1 уменьшается на 1. Если оказывается, что c1 = 0, то доступ пользователя к серверу блокируется на 1 час. При правильно введенном пароле значение c1 снова устанавливается в 5.

При каждой неудачной попытке ввода пароля (подряд или не подряд) значение c2 уменьшается на 1. Если оказывается, что c2 = 0, то доступ пользователя к серверу предоставляется только для изменения пароля.

При каждой попытке ввода пароля (удачной или не удачной) значение c3 уменьшается на 1. Если оказывается, что c3 = 0, то доступ пользователя к серверу предоставляется только для изменения пароля.

После изменения пароля константы c1, c2 и c3 снова устанавливаются в начальные значения c1 = 5, c2 = 20, c3 = 100000.

Следует также обратить внимание на значение поля `last_date_time`. В текущей тестовой версии учетная запись предоставляется на 30 дней с момента регистрации. После указанной в этом поле даты токен становится недоступным, а утилита `ls11cloud_config` будет выполнять только команды status и unregister.

3.1.4 Просмотр лог-файла

После аутентификации сервер записывает диагностические сообщения сессии в персональный лог-файл пользователя, содержимое которого можно увидеть с помощью команды log утилиты `ls11cloud_config`:

```
>ls11cloud_config log
LS11CLOUD User Utility
command: log
Try to connect to server at pkcs11.ru:4444
Connected to server
----------------------------------------
A: Successful SESPAKE authentication of B
----------------------------------------

==== BEGIN log.txt ====

...
========== vblazhnov session started 21/04/2017 at 11:32:49 ===========
21/04/2017 at 11:32:49 FTOKEN session started for vblazhnov
21/04/2017 at 11:32:49 FTOKEN operation:
 ftoken = FTOKEN_Initialize, ret = CKR_OK, ftoken_buf_len = 0
FTOKEN_Initialize rc = 0x0(CKR_OK)
21/04/2017 at 11:32:49 FTOKEN operation finished
```

```
21/04/2017 at 11:32:49 FTOKEN operation:
ftoken = FTOKEN_LoadTokenData, ret = CKR_OK, ftoken_buf_len = 0
FTOKEN_LoadTokenData rc = 0x0(CKR_OK)
21/04/2017 at 11:32:49 FTOKEN operation finished
21/04/2017 at 11:32:49 FTOKEN operation:
ftoken = FTOKEN_Login, ret = CKR_OK, ftoken_buf_len = 16
FTOKEN_Login rc = 0x0(CKR_OK)
21/04/2017 at 11:32:49 FTOKEN operation finished
21/04/2017 at 11:32:49 FTOKEN operation:
ftoken = FTOKEN_LoadTokenData, ret = CKR_OK, ftoken_buf_len = 0
FTOKEN_LoadTokenData rc = 0x0(CKR_OK)
21/04/2017 at 11:32:49 FTOKEN operation finished
21/04/2017 at 11:32:49 FTOKEN operation:
ftoken = FTOKEN_DeleteObject, ret = CKR_OK, ftoken_buf_len = 9
FTOKEN_DeleteObject rc = 0x0(CKR_OK)
21/04/2017 at 11:32:49 FTOKEN operation finished
token_func(): Terminating FTOKEN session...
21/04/2017 at 11:32:49 FTOKEN operation finished
========== vblazhnov session finished 21/04/2017 at 11:32:49 ===========
========== vblazhnov session started 21/04/2017 at 11:37:13 ===========
LCC_SESPAKE_A_GET_LOG session started
LCC_SESPAKE_A_GET_LOG session finished
========== vblazhnov session finished 21/04/2017 at 11:37:13 ===========

==== END log.txt ====

OK
```

Просмотр содержимого лог-файла позволяет разобраться, что и когда выполнялось пользователем на сервере. Одна информация в лог-файле предназначена для администратора сервера, другая позволяет пользователю убедится в успешности или ошибочности выполнения своих операций.

Размер лог-файла ограничен 64Kb, поэтому по мере его увеличения старая информация может быть вытеснена новой и удалена из файла. Информация о сессиях упорядочена по дате и времени. Информация о последней сессии пользователя располагается в конце файла.

3.1.5 Изменение пароля

```
>ls11cloud_config change_pswd
LS11CLOUD User Utility
command: change_pswd
host = "pkcs11.ru"
port = "4444"
```

```
id = "vblazhnov"
Try to connect to server at pkcs11.ru:4444
Connected to server
----------------------------------------
A: Successful SESPAKE authentication of B
----------------------------------------
Enter new SESPAKE password for vblazhnov, please:
******
Enter new SESPAKE password for vblazhnov once more, please:
******
OK
```

В доверенной среде новое значение пароля можно также задавать в командной строке после флага -n.

4 Удаленное конфигурирование токена

Когда пользователь зарегистрирован, он может использовать библиотеку ls11cloud для прикладной работы. Однако ему еще нужно конфигурировать свой токен, используя утилиту p11conf и библиотеку ls11cloud. Утилита p11conf может работать через стандартный программный интерфейс с любой библиотекой PKCS#11, а не только с ls11cloud.

4.1 Утилита p11conf

Для удаленного конфигурирования токена используется утилита командной строки p11conf со следующим интерфейсом:

```
> p11conf -h
p11conf [-hitsmIupPredf] -A <PKCS#11 library path>
        [-c <slot ID> -U <user PIN> -S <SO PIN> -n <new PIN> -L <label>]
Flags:
        -h display usage
        -i display PKCS#11 library info
        -s display slot(s) info (-c <slot ID> is optional)
        -t display token(s) info (-c <slot ID> is optional)
Others must use -c <slot ID> too
        -m display mechanism list
        -I initialize token
        -u initialize user PIN
        -p set the user PIN
        -P set the SO PIN
        -e enumerate objects
        -d dump all object attributes (additional to -e and to -f)
        -r remove all objects
        -e -r remove enumerated objects with prompt
        -f enumerate certificates and write them to DER-files with prompt
Version 5.7
Copyright(C) LISSI-Soft Ltd (http://soft.lissi.ru) 2011-2018
```

В качестве `<PKCS#11 library path>` должен быть задан путь к динамической библиотеке ls11cloud. Если путь к папке с библиотекой задан в переменной среды PATH, то достаточно указать имя файла библиотеки. В Windows можно просто

указать ls11cloud без расширения dll, а в Linux нужно указывать имя полностью - libls11cloud.so.

Заметим, что с флагом -c задается идентификатор слота. Для ls11cloud этот идентификатор всегда равен 0.

После успешной регистрации у пользователя на сервере имеется пустой личный токен. В соответствии со стандартом PKCS#11, начальные операции с токеном выполняются администратором безопасности (Security Officer -SO). Начальное значение SO PIN для токена ls11cloud - 87654321. Для подготовки токена к прикладной работе ему нужно назначить уникальную метку, изменить умалчиваемое значение SO PIN, назначить начальное значение USER PIN от имени SO и изменить это значение от имени пользователя. Эти четыре операции производятся утилитой p11conf.

Далее приводятся примеры использования утилиты p11conf для подготовки токена к тестированию. Все значения PIN отображаются звездочками при вводе, но мы покажем здесь тестовые значения PIN.

4.1.1 Инициализация токена

```
> p11conf -A ls11cloud-I -c 0
Try to connect to server at pkcs11.ru:4444
Connected to server
----------------------------------------
A: Successful SESPAKE authentication of B
----------------------------------------
Enter the SO PIN: 87654321
Enter a unique token label: vblazhnov
OK
```

То же самое можно выполнить, задавая значения SO PIN и метки прямо в командной строке:

```
> p11conf -A ls11cloud -I -c 0 -S 87654321 -L vblazhnov
```

4.1.2 Назначение PIN администратора безопасности

Правильной организационной практикой является изменение администратором безопасности своего PIN сразу после инициализации токена. Данная процедура предотвращает возможность инициализировать токен посторонним лицам и удалить тем самым все созданные объекты (например, ключи и сертификаты).

```
> p11conf -A ls11cloud -P -c 0
Try to connect to server at pkcs11.ru:4444
Connected to server
----------------------------------------
A: Successful SESPAKE authentication of B
----------------------------------------
```

```
Enter the SO PIN: 87654321
Enter the new SO PIN: 76543210
Re-enter the new SO PIN: 76543210
OK
```

Вариант ввода в командной строке:

```
> p11conf -A ls11cloud -P -c 0 -S 87654321 -n 76543210
```

4.1.3 Инициализация пользовательского PIN

Данная операция выполняется администратором безопасности перед передачей токена пользователю. Программный токен изначально создается в файловом пространстве пользователя, однако формальные требования стандарта должны быть выполнены и для него.

```
> p11conf -A ls11cloud -u -c 0
Try to connect to server at pkcs11.ru:4444

Connected to server
----------------------------------------
A: Successful SESPAKE authentication of B
----------------------------------------
Enter the SO PIN: 76543210
Enter the new user PIN: 12345678
Re-enter the new user PIN: 12345678
OK
```

Вариант ввода в командной строке:

```
> p11conf -A ls11cloud -u -c 0 -S 76543210 -n 12345678
```

4.1.4 Изменение пользовательского PIN

Первое, что должен сделать пользователь после получения токена от администратора безопасности, - это изменение PIN.

```
> p11conf -A ls11cloud -p -c 0
Try to connect to server at pkcs11.ru:4444
Connected to server
----------------------------------------
A: Successful SESPAKE authentication of B
----------------------------------------
Enter user PIN: 12345678
Enter the new user PIN: 01234567
Re-enter the new user PIN: 01234567
OK
```

Вариант ввода в командной строке:

```
> p11conf -A ls11cloud -p -c 0 -U 12345678 -n 01234567
```

4.1.5 Получение информации о библиотеке

```
> p11conf -A ls11cloud -i
Try to connect to server at pkcs11.ru:4444
Connected to server
----------------------------------------
A: Successful SESPAKE authentication of B
----------------------------------------
PKCS#11 Info
        Version 2.40
        Manufacturer: LISSI-Soft
        Flags: 0x0
        Library Description: ls11cloud PKCS#11 library
        Library Version 2.0
OK
```

4.1.6 Получение информации о токене

```
> p11conf -A ls11cloud -t -c 0
Try to connect to server at pkcs11.ru:4444
Connected to server
----------------------------------------
A: Successful SESPAKE authentication of B
----------------------------------------
Slot 0 Token Info:

Label: vblazhnov
Manufacturer: LISSI-Soft, Ltd
Model: LS11CLOUD
Serial Number: 2370F1197131E3B0
Flags: 0x40D
 ( CKF_RNG|CKF_LOGIN_REQUIRED|CKF_USER_PIN_INITIALIZED|CKF_TOKEN_INITIALIZED )
Sessions: 0/256
R/W Sessions: 0/256
PIN Length: 4-32
Public Memory: CK_UNAVAILABLE_INFORMATION/CK_UNAVAILABLE_INFORMATION
Private Memory: CK_UNAVAILABLE_INFORMATION/CK_UNAVAILABLE_INFORMATION
Hardware Version: 3.0
Firmware Version: 3.0
Time: 13:09:04
```

```
OK
```

5 Тестирование

Пользователям предоставляется тестовый CMake-проект`ls11cloud_tests`, содержащий программы, проверяющие функционирование различных механизмов PKCS#11. Эти программы могут также служить примерами для разработки собственных прикладных программ. Для генерации проектных файлов в операционной системе должна быть установлена сборочная система CMake[16].

5.1 Тестовый проект

Для генерации проектных файлов нужно из папки build вызвать команду:

```
>cmake ..
```

В результате, в системе будут созданы проектные файлы для имеющейся среды программирования. Далее сборка тестов производится либо средствами MS Visual Studio (в Windows), либо командой make (в Linux).

5.2 Запуск тестов

В программах тестового проекта по умолчанию предполагается, что у пользовательского токена SO PIN равен 76543210, а USER PIN равен 01234567. При необходимости, эти значения можно изменить в сборочном файле проекта CMakeLists.txt.

При запуске команды ctest из папки build будут поочередно запущены все тесты. Названия большинства тестов указывают либо на название тестируемого механизма, либо на тестируемый класс объекта.

```
Test project G:/sespake/build
      Start  1: info
 1/87 Test  #1: info ................................   Passed    1.07 sec
      Start  2: cko_data
 2/87 Test  #2: cko_data ............................   Passed    1.61 sec
      Start  3: cko_certificate
 3/87 Test  #3: cko_certificate .....................   Passed    1.82 sec
      Start  4: ckm_gostr3411
 4/87 Test  #4: ckm_gostr3411 .......................   Passed    1.26 sec
      Start  5: ckm_gostr3411_12_256
 5/87 Test  #5: ckm_gostr3411_12_256 ................   Passed    0.99 sec
      Start  6: ckm_gostr3411_12_512
```

```
 6/87 Test  #6: ckm_gostr3411_12_512 ...................   Passed    1.05 sec
      Start  7: ckm_gostr3411_hmac
 7/87 Test  #7: ckm_gostr3411_hmac .....................   Passed    1.02 sec
      Start  8: ckm_gostr3411_12_256_hmac
 8/87 Test  #8: ckm_gostr3411_12_256_hmac ..............   Passed    1.16 sec
      Start  9: ckm_gostr3411_12_512_hmac
 9/87 Test  #9: ckm_gostr3411_12_512_hmac ..............   Passed    1.08 sec
      Start 10: ckm_kdf_gostr3411_2012_256
10/87 Test #10: ckm_kdf_gostr3411_2012_256 .............   Passed    1.05 sec
      Start 11: ckm_kdf_tree_gostr3411_2012_256
11/87 Test #11: ckm_kdf_tree_gostr3411_2012_256 ........   Passed    1.13 sec
      Start 12: ckm_gost_generic_secret_key_gen
12/87 Test #12: ckm_gost_generic_secret_key_gen ........   Passed    1.09 sec
      Start 13: ckm_extract_key_from_key
13/87 Test #13: ckm_extract_key_from_key ...............   Passed    1.16 sec
      Start 14: ckm_gost_cipher_key_gen
14/87 Test #14: ckm_gost_cipher_key_gen ................   Passed    1.11 sec
      Start 15: ckm_gost_cipher_ecb
15/87 Test #15: ckm_gost_cipher_ecb ....................   Passed    1.30 sec
      Start 16: ckm_gost_cipher_cbc
16/87 Test #16: ckm_gost_cipher_cbc ....................   Passed    1.12 sec
      Start 17: ckm_gost_cipher_ctr
17/87 Test #17: ckm_gost_cipher_ctr ....................   Passed    1.07 sec
      Start 18: ckm_gost_cipher_ofb
18/87 Test #18: ckm_gost_cipher_ofb ....................   Passed    1.17 sec
      Start 19: ckm_gost_cipher_cfb
19/87 Test #19: ckm_gost_cipher_cfb ....................   Passed    1.12 sec
      Start 20: ckm_gost_cipher_acpkm_ctr
20/87 Test #20: ckm_gost_cipher_acpkm_ctr ..............   Passed    0.93 sec
      Start 21: ckm_gost_cipher_omac
21/87 Test #21: ckm_gost_cipher_omac ...................   Passed    1.01 sec
      Start 22: ckm_gost_cipher_acpkm_omac
22/87 Test #22: ckm_gost_cipher_acpkm_omac .............   Passed    1.02 sec
      Start 23: ckm_gost_cipher_key_wrap
23/87 Test #23: ckm_gost_cipher_key_wrap ...............   Passed    0.98 sec
      Start 24: ckm_gost_cipher_pkcs8_key_wrap
24/87 Test #24: ckm_gost_cipher_pkcs8_key_wrap .........   Passed    3.19 sec
      Start 25: ckm_gost28147_key_gen
25/87 Test #25: ckm_gost28147_key_gen ..................   Passed    0.99 sec
      Start 26: ckm_gost28147
26/87 Test #26: ckm_gost28147 ..........................   Passed    1.06 sec
      Start 27: ckm_gost28147_ecb
27/87 Test #27: ckm_gost28147_ecb ......................   Passed    1.11 sec
      Start 28: ckm_gost28147_ecb_mac_wrap
```

```
28/87 Test #28: ckm_gost28147_ecb_mac_wrap .............   Passed    1.02 sec
      Start 29: ckm_gost28147_cnt
29/87 Test #29: ckm_gost28147_cnt ......................   Passed    1.13 sec
      Start 30: ckm_gost28147_cbc
30/87 Test #30: ckm_gost28147_cbc ......................   Passed    1.07 sec
      Start 31: ckm_gost28147_mac
31/87 Test #31: ckm_gost28147_mac ......................   Passed    1.28 sec
      Start 32: ckm_gost28147_cfb_random
32/87 Test #32: ckm_gost28147_cfb_random ...............   Passed    1.10 sec
      Start 33: ckm_gost28147_key_wrap
33/87 Test #33: ckm_gost28147_key_wrap .................   Passed    0.99 sec
      Start 34: ckm_gost28147_pkcs8_key_wrap
34/87 Test #34: ckm_gost28147_pkcs8_key_wrap ...........   Passed    4.68 sec
      Start 35: ckm_kdf_4357
35/87 Test #35: ckm_kdf_4357 ...........................   Passed    1.01 sec
      Start 36: ckm_magma_key_gen
36/87 Test #36: ckm_magma_key_gen ......................   Passed    1.28 sec
      Start 37: ckm_magma_ecb
37/87 Test #37: ckm_magma_ecb ..........................   Passed    1.00 sec
      Start 38: ckm_magma_cbc
38/87 Test #38: ckm_magma_cbc ..........................   Passed    0.94 sec
      Start 39: ckm_magma_ctr
39/87 Test #39: ckm_magma_ctr ..........................   Passed    0.97 sec
      Start 40: ckm_magma_acpkm_ctr
40/87 Test #40: ckm_magma_acpkm_ctr ....................   Passed    1.20 sec
      Start 41: ckm_magma_ofb
41/87 Test #41: ckm_magma_ofb ..........................   Passed    1.09 sec
      Start 42: ckm_magma_cfb
42/87 Test #42: ckm_magma_cfb ..........................   Passed    1.02 sec
      Start 43: ckm_magma_omac
43/87 Test #43: ckm_magma_omac .........................   Passed    1.11 sec
      Start 44: ckm_magma_acpkm_omac
44/87 Test #44: ckm_magma_acpkm_omac ...................   Passed    1.08 sec
      Start 45: ckm_magma_key_wrap
45/87 Test #45: ckm_magma_key_wrap .....................   Passed    1.23 sec
      Start 46: ckm_magma_cfb_errors
46/87 Test #46: ckm_magma_cfb_errors ...................   Passed    1.10 sec
      Start 47: ckm_magma_cfb_random
47/87 Test #47: ckm_magma_cfb_random ...................   Passed    1.03 sec
      Start 48: ckm_kuznyechik_key_gen
48/87 Test #48: ckm_kuznyechik_key_gen .................   Passed    1.26 sec
      Start 49: ckm_kuznyechik_ecb
49/87 Test #49: ckm_kuznyechik_ecb .....................   Passed    1.05 sec
      Start 50: ckm_kuznyechik_cbc
```

```
50/87 Test #50: ckm_kuznyechik_cbc .....................   Passed    1.05 sec
      Start 51: ckm_kuznyechik_ctr
51/87 Test #51: ckm_kuznyechik_ctr .....................   Passed    0.98 sec
      Start 52: ckm_kuznyechik_acpkm_ctr
52/87 Test #52: ckm_kuznyechik_acpkm_ctr ...............   Passed    1.01 sec
      Start 53: ckm_kuznyechik_ofb
53/87 Test #53: ckm_kuznyechik_ofb .....................   Passed    0.95 sec
      Start 54: ckm_kuznyechik_cfb
54/87 Test #54: ckm_kuznyechik_cfb .....................   Passed    0.97 sec
      Start 55: ckm_kuznyechik_omac
55/87 Test #55: ckm_kuznyechik_omac ....................   Passed    1.03 sec
      Start 56: ckm_kuznyechik_acpkm_omac
56/87 Test #56: ckm_kuznyechik_acpkm_omac ..............   Passed    0.96 sec
      Start 57: ckm_kuznyechik_key_wrap
57/87 Test #57: ckm_kuznyechik_key_wrap ................   Passed    0.97 sec
      Start 58: ckm_kuznyechik_cfb_random
58/87 Test #58: ckm_kuznyechik_cfb_random ..............   Passed    1.10 sec
      Start 59: ckm_gostr3410_key_pair_gen
59/87 Test #59: ckm_gostr3410_key_pair_gen .............   Passed    1.82 sec
      Start 60: ckm_gostr3410_public_key_derive
60/87 Test #60: ckm_gostr3410_public_key_derive .........   Passed    2.64 sec
      Start 61: ckm_gostr3410
61/87 Test #61: ckm_gostr3410 ..........................   Passed    1.79 sec
      Start 62: ckm_gostr3410_512
62/87 Test #62: ckm_gostr3410_512 ......................   Passed    2.17 sec
      Start 63: ckm_gostr3410_key_derive
63/87 Test #63: ckm_gostr3410_key_derive ...............   Passed    2.17 sec
      Start 64: ckm_gostr3410_12_256_key_derive
64/87 Test #64: ckm_gostr3410_12_256_key_derive .........   Passed    2.55 sec
      Start 65: ckm_gostr3410_12_512_key_derive
65/87 Test #65: ckm_gostr3410_12_512_key_derive .........   Passed    2.41 sec
      Start 66: ckm_gostr3410_2012_256_vko_256
66/87 Test #66: ckm_gostr3410_2012_256_vko_256 ..........   Passed    2.22 sec
      Start 67: ckm_gostr3410_2012_512_vko_256
67/87 Test #67: ckm_gostr3410_2012_512_vko_256 ..........   Passed    1.35 sec
      Start 68: ckm_gostr3410_2012_512_vko_512
68/87 Test #68: ckm_gostr3410_2012_512_vko_512 ..........   Passed    2.68 sec
      Start 69: ckm_gostr3410_key_wrap
69/87 Test #69: ckm_gostr3410_key_wrap ..................   Passed    3.56 sec
      Start 70: ckm_gostr3410_with_gostr3411
70/87 Test #70: ckm_gostr3410_with_gostr3411 ............   Passed    1.78 sec
      Start 71: ckm_gostr3410_with_gostr3411_12_256
71/87 Test #71: ckm_gostr3410_with_gostr3411_12_256 .....   Passed    1.73 sec
      Start 72: ckm_gostr3410_with_gostr3411_12_512
```

```
72/87 Test #72: ckm_gostr3410_with_gostr3411_12_512 .....   Passed    2.02 sec
      Start 73: cka_always_authenticate
73/87 Test #73: cka_always_authenticate .................   Passed    2.69 sec
      Start 74: keypair_import
74/87 Test #74: keypair_import ..........................   Passed    1.74 sec
      Start 75: ckm_tls_gost_prf
75/87 Test #75: ckm_tls_gost_prf ........................   Passed    1.06 sec
      Start 76: ckm_tls_gost_prf_2012
76/87 Test #76: ckm_tls_gost_prf_2012 ...................   Passed    1.06 sec
      Start 77: ckm_tls_gost_pre_master_key_gen
77/87 Test #77: ckm_tls_gost_pre_master_key_gen .........   Passed    1.12 sec
      Start 78: ckm_tls_gost_master_key_derive
78/87 Test #78: ckm_tls_gost_master_key_derive ..........   Passed    1.24 sec
      Start 79: ckm_tls_gost_key_and_mac_derive
79/87 Test #79: ckm_tls_gost_key_and_mac_derive .........   Passed    1.04 sec
      Start 80: ckm_tls12_master_key_derive
80/87 Test #80: ckm_tls12_master_key_derive .............   Passed    1.03 sec
      Start 81: ckm_tls12_key_and_mac_derive
81/87 Test #81: ckm_tls12_key_and_mac_derive ............   Passed    1.10 sec
      Start 82: ckm_tls_mac
82/87 Test #82: ckm_tls_mac .............................   Passed    1.29 sec
      Start 83: ckm_tls_kdf
83/87 Test #83: ckm_tls_kdf .............................   Passed    1.11 sec
      Start 84: ckm_tls_tree_gostr3411_2012_256
84/87 Test #84: ckm_tls_tree_gostr3411_2012_256 .........   Passed    1.11 sec
      Start 85: ckm_pkcs5_pbkd2
85/87 Test #85: ckm_pkcs5_pbkd2 .........................   Passed    1.32 sec
      Start 86: ckm_pba_gostr3411_with_gostr3411_hmac
86/87 Test #86: ckm_pba_gostr3411_with_gostr3411_hmac ...   Passed    1.17 sec
      Start 87: create_obj
87/87 Test #87: create_obj ..............................   Passed    9.14 sec

100% tests passed, 0 tests failed out of 87

Total Test time (real) = 127.99 sec
```

6 Ссылки

1. Официальный сайт ООО "ЛИССИ-Софт". - http://http://soft.lissi.ru//.
2. Официальный сайт Технического комитета по стандартизации (ТК 26) "Криптографическая защита информации". - https://www.tc26.ru.
3. ГОСТ 28147-89. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. – http://protect.gost.ru/document.aspx?control=7&id=139177.
4. ГОСТ Р 34.10-2001. Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи. – http://protect.gost.ru/document.aspx?control=7&id=131131.
5. ГОСТ Р 34.10-2012. Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи. – Москва, Стандартинформ, 2012.
6. ГОСТ Р 34.12-2015. Блочные шифры. – Москва, Стандартинформ, 2015.
7. ГОСТ Р 34.13-2015. Режимы блочных шифров. – Москва, Стандартинформ, 2015.
8. ГОСТ Р 34.11-94. Информационная технология. Криптографическая защита информации. Функция хеширования. – http://protect.gost.ru/document.aspx?control=7&id=134550.
9. ГОСТ Р 34.11-2012. Информационная технология. Криптографическая защита информации. Функция хеширования. – Москва, Стандартинформ, 2012.
10. Расширение PKCS#11 для использования российских криптографических алгоритмов. – Технический комитет по стандартизации (ТК 26) "Криптографическая защита информации". – Москва, ТК 26, 2008.
11. Расширение PKCS#11 для использования российских стандартов ГОСТ Р 34.10-2012 и ГОСТ Р 34.11-2012 (готовится к публикации). – Технический комитет по стандартизации (ТК 26) "Криптографическая защита информации". – Москва, ТК 26, 2013.
12. Расширение PKCS#11 для использования российских стандартов ГОСТ Р 34.12-2015 и ГОСТ Р 34.13-2015 (готовится к публикации). – Технический комитет по стандартизации (ТК 26) "Криптографическая защита информации". – Москва, ТК 26, 2016.

13. Протокол выработки общего ключа с аутентификацией на основе пароля (готовится к публикации). – Технический комитет по стандартизации (ТК 26) "Криптографическая защита информации". – Москва, ТК 26, 2015.

14. Криптографические алгоритмы, сопутствующие применению алгоритмов блочного шифрования (готовится к публикации). – Технический комитет по стандартизации (ТК 26) "Криптографическая защита информации". – Москва, ТК 26, 2017.

15. PKCS#11 v2.40: Cryptographic Token Interface Standard. - OASIS PKCS#11 TC. - https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=pkcs11.

16. Кроссплатформенная сборочная система CMake. - http://www.cmake.org/.