

Утилита обслуживания токена
PKCS#11
Руководство пользователя

ООО "ЛИССИ-Софт"

6 мая 2015 г.

Оглавление

1 Введение	3
1.1 Системные требования	3
2 Основные сведения	4
2.1 Состав и структура	4
2.2 Сборка	4
2.3 Установка	4
2.4 Примеры использования	5
2.4.1 Получение информации о библиотеке	5
2.4.2 Получение информации о слотах	6
2.4.3 Получение информации о токене	6
2.4.4 Получение информации о механизмах	6
2.4.5 Получение информации об объектах	10
2.4.6 Удаление всех объектов токена	14
2.4.7 Инициализация токена	14
2.4.8 Назначение PIN администратора безопасности	14
2.4.9 Инициализация пользовательского PIN	15
2.4.10 Изменение пользовательского PIN	15
3 Ссылки	16

1 Введение

Утилита `p11conf` является вспомогательным средством для выполнения информационных и конфигурационных действий с токеном через интерфейс PKCS#11. Она является независимым дополнением к программной реализации библиотеки `ls11sw2012` ООО "ЛИССИ-Софт" [1] стандарта PKCS#11 API [5], дополненного поддержкой российских криптографических алгоритмов в соответствии со спецификациями, выработанными Техническим комитетом по стандартизации (ТК 26) "Криптографическая защита информации" [2, 3]. Начиная с версии 5.0, `ls11sw2012` поддерживает алгоритмы ГОСТ Р34.10-2012, ГОСТ Р34.11-2012, а также сопутствующие алгоритмы и параметры, определенные руководящими документами ТК 26.

В то же время, утилита `p11conf` не зависит от конкретной библиотеки и может работать с любой библиотекой PKCS#11 и, следовательно, с любыми токенами, поддерживающими данный интерфейс.

1.1 Системные требования

Утилита `p11conf` реализована кросс-платформенным образом и работает в режиме командной строки, так что она, в принципе, может быть портирована в любую операционную систему, где поддерживается язык Си. Текущая версия работает в операционных системах Windows, Linux и Mac OS X на 32-х и 64-х разрядных платформах.

2 Основные сведения

2.1 Состав и структура

В состав проекта входят:

- Утилита конфигурации `p11conf`
- Руководство пользователя

Выполняемый файл утилиты `p11conf` в Windows называется `p11conf.exe`, а в Linux и Mac OS X – просто `p11conf`.

2.2 Сборка

В проекте утилиты сборка выполняется под управлением кросс-платформенной сборочной системы CMake версии 2.8 и выше [6].

Для сборки утилиты нужно перейти в папку `build` проекта и выполнить команду:

```
сmake ..
```

В результате, в папке `build` будут созданы проектные файлы, соответствующие сборочной среде.

Далее, в среде Linux или Mac OS X нужно последовательно выполнить команды:

```
make  
сrack
```

Этими командами в папке `build` будет собран выполняемый файл инсталлятора утилиты.

В среде Windows CMake создает проектные файлы для MSVS. Открыв файл `p11conf.sln` в среде MSVS, нужно последовательно выполнить для конфигурации Release построение проектов `p11conf` и `PACKAGE`. В результате, в папке `build` будет собран выполняемый файл инсталлятора утилиты.

2.3 Установка

Утилита `p11conf` устанавливается инсталлятором в целевую папку.

Важное замечание. На целевой платформе нужно обеспечить прикладным программам возможность найти утилиту по имени путем добавления полного пути к содержащей ее папке к значению переменной среды `PATH`.

Документация для `p11conf` скачивается отдельно с сайта "ЛИССИ-Софт"[1].

2.4 Примеры использования

Утилита предоставляет возможности, о которых сообщается при запуске команды `p11conf -h`. К ним относятся инициализация токена, инициализация и изменение PIN администратора безопасности, инициализация и изменение PIN пользователя и др. Следующие примеры показывают опции утилиты `p11conf` и выдачу информации о слотах в системе до инициализации токенов.

Данная программа также позволяет выполнять некоторые простые запросы слотам и токенам, например для получения информации о слотах, токенах и для получения списка поддерживаемых механизмов.

```
> p11conf -h
usage: p11conf [-hitsmIupPred] -A APIpath [-c slotID
-U userPin -S S0Pin -n newPin -L label]
    -h display usage
    -i display PKCS11 info
    -t display token info
    -s display slot info
    -m display mechanism list
    -I initialize token
    -u initialize user PIN
    -p set the user PIN
    -P set the S0 PIN
    -r remove all objects
    -e enumerate objects
    -d dump enumerated object attributes
```

Заметим, что с флагом `-c` задается идентификатор слота. Этот идентификатор необходимо задавать, когда операция выполняется с конкретным токеном. Идентификаторы всех слотов можно получить с флагом `-s`.

Далее приводятся некоторые примеры использования утилиты с библиотекой `ls11sw2012.dll` в среде Windows. В этой среде допускается использование с флагом `-A` имени библиотеки без расширения, если путь к папке, содержащей библиотеку, конфигурирован соответствующим образом в переменной среды `PATH`. В Linux библиотека называется `libls11sw2012.so`, в Mac OS X – `libls11sw2012.dylib`. В этих операционных системах нужно указывать имя библиотеки с префиксом `lib` и с расширением, а путь к папке, содержащей библиотеку, должен быть прописан в переменной среды `LD_LIBRARY_PATH`. Впрочем, с флагом `-A` можно использовать и полный путь к файлу библиотеки.

2.4.1 Получение информации о библиотеке

```
> p11conf -A ls11sw2012 -i
PKCS#11 Info
    Version 2.30
```

```
Manufacturer: LISSI-Soft
Flags: 0x0
Library Description: ls11sw2012 PKCS#11 library
Library Version 5.0
```

OK

2.4.2 Получение информации о слотах

```
> p11conf -A ls11sw2012 -s
Slot ID 0 Info
  Description: LS11SW Slot 0
  Manufacturer: LISSI-Soft
  Flags: 0x7 (TOKEN_PRESENT)
  Hardware Version: 1.0
  Firmware Version: 1.0
```

OK

2.4.3 Получение информации о токене

```
> p11conf -A ls11sw2012 -t -c 0
Token #0 Info:
  Label: Token Label
  Manufacturer: LISSI-Soft
  Model: LS11SW
  Serial Number: 1234567887654321
  Flags: 0x40C (LOGIN_REQUIRED|USER_PIN_INITIALIZED|TOKEN_INITIALIZED)
  Sessions: 0/256
  R/W Sessions: 0/256
  PIN Length: 4-32
  Public Memory: 0xFFFFFFFF/0xFFFFFFFF
  Private Memory: 0xFFFFFFFF/0xFFFFFFFF
  Hardware Version: 1.0
  Firmware Version: 1.0
  Time: 18:58:42
```

OK

Программный токен создается не инициализированным. Прежде чем приложение сможет использовать новый токен, нужно выполнить следующие начальные шаги (каждому шагу соответствует пример кода):

2.4.4 Получение информации о механизмах

Выдается информация о механизмах, поддерживаемых токеном. Программный токен ls11sw2012 поддерживает все стандартные механизмы PKCS#11, определенные ТК 26, и некоторые дополнительные.

```
> p11conf -A ls11sw2012 -m -c 0
Mechanism #0
Mechanism: CKM_GOSTR3410_KEY_PAIR_GEN (0x1200)
Key Size: 256-256
Flags: 0x10000 ( CKF_GENERATE_KEY_PAIR )
Mechanism #1
Mechanism: CKM_GOSTR3410_512_KEY_PAIR_GEN (0xD4321005)
Key Size: 512-512
Flags: 0x10000 ( CKF_GENERATE_KEY_PAIR )
Mechanism #2
Mechanism: CKM_TLS_GOST_KEY_AND_MAC_DERIVE (0xD4321033)
Key Size: 32-32
Flags: 0x80000 ( CKF_DERIVE )
Mechanism #3
Mechanism: CKM_TLS_GOST_PRE_MASTER_KEY_GEN (0xD4321031)
Key Size: 32-32
Flags: 0x8000 ( CKF_GENERATE )
Mechanism #4
Mechanism: CKM_TLS_GOST_MASTER_KEY_DERIVE (0xD4321032)
Key Size: 48-48
Flags: 0x80000 ( CKF_DERIVE )
Mechanism #5
Mechanism: CKM_TLS_GOST_PRF (0xD4321030)
Key Size: 32-32
Flags: 0x80000 ( CKF_DERIVE )
Mechanism #6
Mechanism: CKM_PBA_GOSTR3411_WITH_GOSTR3411_HMAC (0xD4321035)
Key Size: 32-32
Flags: 0x8000 ( CKF_GENERATE )
Mechanism #7
Mechanism: CKM_PKCS5_PBKD2 (0x3B0)
Key Size: 32-32
Flags: 0x8000 ( CKF_GENERATE )
Mechanism #8
Mechanism: CKM_GOST28147_KEY_GEN (0x1220)
Key Size: 32-32
Flags: 0x8000 ( CKF_GENERATE )
Mechanism #9
Mechanism: CKM_GOSTR3410 (0x1201)
Key Size: 256-256
Flags: 0x2800 ( CKF_SIGN|CKF_VERIFY )
Mechanism #10
Mechanism: CKM_GOSTR3410_512 (0xD4321006)
Key Size: 512-512
```

Flags: 0x2800 (CKF_SIGN|CKF_VERIFY)
Mechanism #11
Mechanism: CKM_GOSTR3410_WITH_GOSTR3411 (0x1202)
Key Size: 256-256
Flags: 0x2800 (CKF_SIGN|CKF_VERIFY)
Mechanism #12
Mechanism: CKM_GOSTR3410_WITH_GOSTR3411_12_256 (0xD4321008)
Key Size: 256-256
Flags: 0x2800 (CKF_SIGN|CKF_VERIFY)
Mechanism #13
Mechanism: CKM_GOSTR3410_WITH_GOSTR3411_12_512 (0xD4321009)
Key Size: 512-512
Flags: 0x2800 (CKF_SIGN|CKF_VERIFY)
Mechanism #14
Mechanism: CKM_GOSTR3410_DERIVE (0x1204)
Key Size: 256-256
Flags: 0x80000 (CKF_DERIVE)
Mechanism #15
Mechanism: CKM_GOSTR3410_12_DERIVE (0xD4321007)
Key Size: 512-512
Flags: 0x80000 (CKF_DERIVE)
Mechanism #16
Mechanism: CKM_GOSTR3410_KEY_WRAP (0x1203)
Key Size: 256-256
Flags: 0x60000 (CKF_WRAP|CKF_UNWRAP)
Mechanism #17
Mechanism: CKM_GOSTR3410_PUBLIC_KEY_DERIVE (0xD4321037)
Key Size: 256-512
Flags: 0x80000 (CKF_DERIVE)
Mechanism #18
Mechanism: CKM_GOST28147 (0x1222)
Key Size: 32-32
Flags: 0x60300 (CKF_ENCRYPT|CKF_DECRYPT|CKF_WRAP|CKF_UNWRAP)
Mechanism #19
Mechanism: CKM_GOST28147_KEY_WRAP (0x1224)
Key Size: 32-32
Flags: 0x60000 (CKF_WRAP|CKF_UNWRAP)
Mechanism #20
Mechanism: CKM_GOST28147_KEY_CPDIVERSIFY (0xD4321026)
Key Size: 32-32
Flags: 0x80000 (CKF_DERIVE)
Mechanism #21
Mechanism: CKM_GOST28147_PKCS8_KEY_WRAP (0xD4321036)
Key Size: 32-32

Flags: 0x60000 (CKF_WRAP|CKF_UNWRAP)
Mechanism #22
Mechanism: CKM_GOST28147_ECB (0x1221)
Key Size: 32-32
Flags: 0x60300 (CKF_ENCRYPT|CKF_DECRYPT|CKF_WRAP|CKF_UNWRAP)
Mechanism #23
Mechanism: CKM_GOST28147_CNT (0xD4321025)
Key Size: 32-32
Flags: 0x300 (CKF_ENCRYPT|CKF_DECRYPT)
Mechanism #24
Mechanism: CKM_GOST28147_MAC (0x1223)
Key Size: 32-32
Flags: 0x2800 (CKF_SIGN|CKF_VERIFY)
Mechanism #25
Mechanism: CKM_GOSTR3411 (0x1210)
Key Size: 0-0
Flags: 0x400 (CKF_DIGEST)
Mechanism #26
Mechanism: CKM_GOSTR3411_12_256 (0xD4321012)
Key Size: 0-0
Flags: 0x400 (CKF_DIGEST)
Mechanism #27
Mechanism: CKM_GOSTR3411_12_512 (0xD4321013)
Key Size: 0-0
Flags: 0x400 (CKF_DIGEST)
Mechanism #28
Mechanism: CKM_GOSTR3411_HMAC (0x1211)
Key Size: 32-32
Flags: 0x2800 (CKF_SIGN|CKF_VERIFY)
Mechanism #29
Mechanism: CKM_GOSTR3411_12_256_HMAC (0xD4321014)
Key Size: 32-32
Flags: 0x2800 (CKF_SIGN|CKF_VERIFY)
Mechanism #30
Mechanism: CKM_GOSTR3411_12_512_HMAC (0xD4321015)
Key Size: 64-64
Flags: 0x2800 (CKF_SIGN|CKF_VERIFY)
Mechanism #31
Mechanism: CKM_GOSTR3411_12_256_HMAC_KDF (0xD4321808)
Key Size: 32-32
Flags: 0x80000 (CKF_DERIVE)
Mechanism #32
Mechanism: CKM_SHA_1 (0x220)
Key Size: 0-0

```
Flags: 0x400 ( CKF_DIGEST )
Mechanism #33
Mechanism: CKM_MD5 (0x210)
Key Size: 0-0
Flags: 0x400 ( CKF_DIGEST )
OK
```

2.4.5 Получение информации об объектах

Если нужно получить список типов и меток объектов токена, то это можно сделать следующей командой:

```
> p11conf -A ls11sw2012 -e -c 0
Enter user PIN: *****
Token objects:
1: CKO_PRIVATE_KEY
   label: ''
2: CKO_PUBLIC_KEY
   label: ''
```

С дополнительным флагом `-d` та же команда выдает шестнадцатеричный дамп значений всех атрибутов объектов токена:

```
> p11conf -A ls11sw2012 -e -d -c 0
Enter user PIN: *****
Token objects:
1: CKO_PRIVATE_KEY
   label: ''
=====
Object handle: 0x1
-----
CKA_CLASS
0x03, 0x00, 0x00, 0x00,

CKA_TOKEN
0x01,

CKA_PRIVATE
0x01,

CKA_LABEL: length 0

CKA_VALUE
0x73, 0x97, 0xa3, 0xa6, 0xd0, 0xa6, 0xcd, 0xd9,
0x1a, 0xdf, 0x8d, 0x5a, 0xab, 0xf1, 0xc5, 0xeb,
```

0xff, 0x21, 0x01, 0xcc, 0x17, 0xda, 0x70, 0x5f,
0x10, 0xa0, 0x9e, 0x6b, 0x3a, 0x96, 0xf0, 0x75,

CKA_KEY_TYPE

0x30, 0x00, 0x00, 0x00,

CKA_SUBJECT: length 0

CKA_ID

0xf4, 0x46, 0x4d, 0xc5, 0x0f, 0xec, 0x33, 0x95,
0x1c, 0x15, 0x2b, 0xa3, 0xa6, 0xd2, 0x04, 0x96,
0x70, 0x1b, 0x26, 0x31,

CKA_SENSITIVE

0x00,

CKA_DECRYPT

0x01,

CKA_UNWRAP

0x01,

CKA_SIGN

0x01,

CKA_SIGN_RECOVER

0x01,

CKA_DERIVE

0x01,

CKA_START_DATE: length 0

CKA_END_DATE: length 0

CKA_EXTRACTABLE

0x01,

CKA_LOCAL

0x00,

CKA_NEVER_EXTRACTABLE

0x00,

СКА_ALWAYS_SENSITIVE

0x00,

СКА_MODIFIABLE

0x01,

СКА_COPYABLE

0x01,

СКА_GOSTR3410_PARAMS

0x06, 0x07, 0x2a, 0x85, 0x03, 0x02, 0x02, 0x23, 0x01,

СКА_GOSTR3411_PARAMS

0x06, 0x07, 0x2a, 0x85, 0x03, 0x02, 0x02, 0x1e, 0x01,

СКА_GOST28147_PARAMS

0x06, 0x07, 0x2a, 0x85, 0x03, 0x02, 0x02, 0x1f, 0x01,

2: CKO_PUBLIC_KEY

label: ''

=====
Object handle: 0x2

СКА_CLASS

0x02, 0x00, 0x00, 0x00,

СКА_TOKEN

0x01,

СКА_PRIVATE

0x00,

СКА_LABEL: length 0

СКА_VALUE

0x36, 0x82, 0xac, 0x1a, 0xed, 0x93, 0x26, 0x95,
0x6e, 0x6b, 0xc4, 0xfd, 0x0a, 0x38, 0x1b, 0xd4,
0x43, 0x35, 0x7d, 0xc1, 0x99, 0xd8, 0x94, 0x49,
0xd6, 0xea, 0x61, 0x08, 0xb6, 0x78, 0xe7, 0x97,
0x17, 0x7d, 0x05, 0x2d, 0x48, 0xe6, 0x0c, 0xe4,
0x83, 0xb6, 0xe2, 0x28, 0xf6, 0x13, 0x53, 0xaf,
0x6e, 0x03, 0xd3, 0x19, 0xc6, 0x5c, 0xad, 0xc4,
0x8e, 0x68, 0x7f, 0xd5, 0x49, 0x8f, 0x1f, 0x3a,

CKA_KEY_TYPE

0x30, 0x00, 0x00, 0x00,

CKA_SUBJECT: length 0

CKA_ID

0xf4, 0x46, 0x4d, 0xc5, 0x0f, 0xec, 0x33, 0x95,
0x1c, 0x15, 0x2b, 0xa3, 0xa6, 0xd2, 0x04, 0x96,
0x70, 0x1b, 0x26, 0x31,

CKA_ENCRYPT

0x01,

CKA_WRAP

0x01,

CKA_VERIFY

0x01,

CKA_VERIFY_RECOVER

0x01,

CKA_DERIVE

0x00,

CKA_START_DATE: length 0

CKA_END_DATE: length 0

CKA_LOCAL

0x00,

CKA_MODIFIABLE

0x01,

CKA_COPYABLE

0x01,

CKA_GOSTR3410_PARAMS

0x06, 0x07, 0x2a, 0x85, 0x03, 0x02, 0x02, 0x23, 0x01,

CKA_GOSTR3411_PARAMS

0x06, 0x07, 0x2a, 0x85, 0x03, 0x02, 0x02, 0x1e, 0x01,

```
СКА_GOST28147_PARAMS
0x06, 0x07, 0x2a, 0x85, 0x03, 0x02, 0x02, 0x1f, 0x01,
```

```
-----
OK
```

2.4.6 Удаление всех объектов токена

Иногда требуется очистить токен от всех объектов. Это может быть выполнено командой:

```
> p11conf -A ls11sw2012 -r -c 0
Enter user PIN: *****
OK
```

2.4.7 Инициализация токена

Перед использованием токена он должен быть однажды инициализирован. Ключевой частью данного процесса является назначение токеноу уникальной метки (имени). Для этого понадобится PIN администратора безопасности (SO) для данного токена (начальное значение SO PIN для только что созданного программного токена ls11sw2012 – 87654321).

PIN не обязательно должен быть цифровым, по стандарту допускаются любые символы в кодировке UTF-8, но во избежание проблем с кодировками рекомендуется использовать латинскую половину кодовой таблицы.

Все значения PIN отображаются звездочками при вводе.

```
> p11conf -A ls11sw2012 -I -c 0
Enter the SO PIN: 87654321
Enter a unique token label: LissiSW
```

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

```
> p11conf -A ls11sw2012 -I -c 0 -S 87654321 -L LissiSW
```

2.4.8 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 ls11sw2012 -P -c 0
Enter the SO PIN: 87654321
```

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

Вариант ввода в командной строке:

```
> p11conf -A ls11sw2012 -P -c 0 -S 87654321 -n 76543210
```

2.4.9 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 ls11sw2012 -u -c 0
Enter the SO PIN: 76543210
Enter the new user PIN: 12345678
Re-enter the new user PIN: 12345678
```

Вариант ввода в командной строке:

```
> p11conf -A ls11sw2012 -u -c 0 -S 76543210 -n 12345678
```

2.4.10 Изменение пользовательского PIN

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

```
> p11conf -A ls11sw2012 -p -c 0
Enter user PIN: 12345678
Enter the new user PIN: 01234567
Re-enter the new user PIN: 01234567
```

Вариант ввода в командной строке:

```
> p11conf -A ls11sw2012 -p -c 0 -U 12345678 -n 01234567
```

Если значение SO PIN для токена потеряно, а токен заблокирован из-за нескольких вводов неверного PIN, то штатными средствами ls11sw2012 разблокировать токен нельзя из соображений секретности его данных — его можно только создать заново.

Замечание. Начальное значение USER PIN, установленное SO, запрещается в дальнейшем устанавливать пользователю из соображений безопасности.

3 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. Расширение PKCS#11 для использования российских криптографических алгоритмов. – Технический комитет по стандартизации (ТК 26) "Криптографическая защита информации". – Москва, ТК 26, 2008.
4. 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.
5. PKCS#11 v2.30: Cryptographic Token Interface Standard. – RSA Laboratories, 2009. – <http://www.rsa.com/rsalabs/node.asp?id=2133>.
6. Кроссплатформенная сборочная система CMake. – <http://www.cmake.org/>.