127018, Москва, Сущёвский Вал, 18
Телефон: (495) 995 4820
https://CryptoPro.ru
E-mail: info@CryptoPro.ru

Средство Криптографической Защиты Информации

КриптоПро CSP Версия 5.0 R3 KC1 (исполнение 1-Base) Руководство программиста

ЖТЯИ.00101-03 96 01
Листов 42

2024 г

Содержание

Аннотация

Настоящий документ описывает программные интерфейсы, входящие в состав СКЗИ «КриптоПро CSP» версия 5.0 R3 КС1 (исполнение 1-Base), доступные для разработки прикладного ПО с непосредственным вызовом функций СКЗИ, а также определяет требования к некоторым операционным системам при встраивании СКЗИ.

Перечень определений и сокращений

CRL	Список отозванных сертификатов (Certificate Revocation List)
CTL	Список доверенных сертификатов (Certificate Trust List)
HDD	Жесткий магнитный диск (Hard Disk Drive)
SSD	Твердотельный накопитель (Solid-State Drive)
АРМ	Автоматизированное рабочее место
АС	Автоматизированная система
БиоДСЧ	Биологический датчик случайных чисел
ГМД	Гибкий магнитный диск
ДСЧ	Датчик случайных чисел
ИС	Информационная система
МДЗ	Модуль доверенной загрузки
МП	Мобильное приложение
НСД	Несанкционированный доступ
ОС	Операционная система
ПАК	Программно-аппаратный комплекс
ПДСЧ	Программный датчик случайных чисел
ПК	Персональный компьютер
ПО	Программное обеспечение
ПЭВМ	Персональная электронно-вычислительная машина
СВТ	Средства вычислительной техники
СКЗИ	Средство криптографической защиты информации
СФ	Среда функционирования
УЦ	Удостоверяющий Центр
ЦР	Центр Регистрации
ЦС	Центр Сертификации
ЭД	Электронный документ
ЭП	Электронная подпись

1 Описание программных интерфейсов

1.1 CryptoAPI

Использование низкоуровневого интерфейса криптопровайдера CryptoAPI, позволяющего выполнять такие функции, как генерация и работа с ключами, шифрование/расшифрование данных, хэширование и электронная подпись, описывается в файле `CSP_5_0.chm`.

1.2 SSPI

Использование интерфейса SSPI (Security Support Provider Interface), содержащего реализацию протокола TLS, обеспечивающего работу с пакетами безопасности при выборе и инициализации пакета и удостоверениями субъектов безопасности, установление соединений, передачу данных, распределение памяти, описывается в файле `SSPI_5_0.chm`.

1.3 CAPILite

Использование высокоуровневого интерфейса CryptoAPI (CryptoAPI Lite), предоставляющего набор функций для обработки сертификатов, списков отозванных сертификатов, расширенного использования ключа, работы с провайдером, выработки значения функции хэширования и электронной подписи, зашифрования и расшифрования данных, работы с хранилищем сертификатов и поддержки идентификатора объекта, описано в файле `CAPILite_5_0.chm`.

1.4 RDK

Описание инструментария разработчика для создания модуля поддержки считывателей, носителей и датчиков случайных чисел (Reader Development Kit, RDK) содержится в файле `reader_5_0.chm`.

1.5 PKCS#11

Интерфейс PKCS#11, реализующий базовое описание стандарта RSA Labs PKCS #11 v2.30 с учетом использования российских криптографических алгоритмов, описан в файле `PKCS11_5_0.chm`.

1.6 КриптоПро PKI SDK

Включает программные интерфейсы, предназначенные для создания и использования серверных и клиентских служб и приложений, функционирующих в составе Инфраструктуры Открытых Ключей (PKI).

• **КриптоПро TSP SDK** — инструментарий разработчика для создания клиентских приложений, использующих штампы времени в соответствии с рекомендациями RFC 3161 с учетом использования российских криптографических алгоритмов.

Описание использования КриптоПро TSP SDK приведено в файле `cppkisdk.chm`.

• **КриптоПро OCSP SDK** — инструментарий разработчика для создания клиентских приложений службы актуальных статусов сертификатов в соответствии с рекомендациями RFC 2560 с учетом использования российских криптографических алгоритмов.

Описание использования КриптоПро OCSP SDK приведено в файле `cppkisdk.chm`.

• Библиотеки с реализацией **ASN.1** интерфейса включают реализацию поддержку ASN.1-структур, описанных в RFC 5280, RFC 5652, RFC 4210 и RFC 4211, а также простых типов ASN.1.

Описание использования интерфейса ASN.1 приведено в файле `cppkisdk.chm`.

• Библиотека **CPLib** содержит реализацию вспомогательных классов работы с двоичными данными, датой и интервалами времени, а также с ANSI/Unicode-строками.

Описание использования интерфейса CPLib приведено в файле `cppkisdk.chm`.

• Библиотека **PKIValidator** предоставляет различные способы проверки объектов PKI по соответствующим политикам для встраивания их в клиентское ПО.

Описание использования КриптоПро PKIValidator приведено в файле `pkivalidator.chm`.

• **КриптоПро ЭЦП SDK** — инструментарий разработчика для создания клиентского ПО, использующего работу с усовершенствованной ЭП в соответствии с рекомендациями CAdES и XAdES — CMS Advanced Electronic Signatures (RFC 5652, RFC 5126) и XMLDSig (рекомендации W3C) с учетом использования российских криптографических алгоритмов.

Описание использования КриптоПро ЭЦП SDK приведено в файле `cades.chm`.

• **КриптоПро ЭЦП Browser plug-in** предназначен для создания и проверки ЭП в веб-браузерах.

Описание использования КриптоПро ЭЦП Browser plug-in приведено в файле `cades.chm`.

1.7 Java (JCA/JCE)

Описание особенностей реализации и использования модуля КриптоПро JavaCSP, функционирующего под управлением Java-машины и реализующего стандартный интерфейс Java Cryptography Architecture (JCA) с учетом использования российских криптографических алгоритмов, приведено в документе ЖТЯИ.00101-03 96 02. Руководство программиста. JavaCSP.

Использование модуля КриптоПро JavaTLS, функционирующего под управлением Java-машины и реализующего протокол TLS с учетом использования российских криптографических алгоритмов, описано в документе ЖТЯИ.00101-03 96 03. Руководство программиста. JavaTLS.

1.8 cpcurl

Библиотека `cpcurl` является модификацией проекта libcurl – библиотеки с открытым исходным кодом, реализующей клиентскую часть различных клиент-серверных протоколов установки соединения (например, HTTP, POP3, FTP, IMAP, SMTP, а также их TLS-версий). Библиотека cpcurl отличается от libcurl возможностью установки TLS-соединения с использованием алгоритмов шифрования на основе ГОСТ 28147-89/ГОСТ Р 34.12-2015.

Описание поддерживаемых функций и их параметров приведено в Приложении 1. Ограничения, накладываемые на описанные ниже функции `cpcurl` в случае использования их при разработке систем на основе «КриптоПро CSP» версия 5.0 R3 КС1 (исполнение 1-Base) (с учетом требований раздела 4 документа ЖТЯИ.00101-03 95 01. Правила пользования) без дополнительных тематических исследований, описаны в Приложении 2 документа ЖТЯИ.00101-03 95 01. Правила пользования.

Пример установки TLS-соединения с поддержкой ГОСТ с использованием интерфейса cpcurl приведен в Приложении 2 настоящего документа.

Подробнее о настройке и использовании поддерживаемых программных интерфейсов можно узнать на портале техподдержки и форуме КриптоПро.

2 Требования к ОС для встроенного применения

2.1 Linux

2.1.1 Требования к библиотекам ОС

LSB 4.0, раздел III. Base Libraries.

СКЗИ КриптоПро CSP для своего функционирования требует следующие библиотеки базовой операционной системы (список библиотек по пакетам):

```
lsb-cprocsp-rdr
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


lsb-cprocsp-kc1
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


lsb-cprocsp-kc2
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


lsb-cprocsp-capilite
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-apache-modssl
/lib/ld-linux.so.2
libc.so.6
libcrypt.so.1
libcrypto.so.1.0.2
libcrypto.so.1.1
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
librt.so.1
libssl.so.1.0.2
libssl.so.1.1
libstdc++.so.6
linux-vdso.so.1


cprocsp-certprop
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpcsclite.so.1
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-legacy
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-nginx
/lib/ld-linux.so.2
libc.so.6
libcrypt.so.1
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-pki-cades
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-pki-plugin
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-rdr-cryptoki
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-rdr-relay
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
lsb-cprocsp-rcrypt
/lib/ld-linux.so.2
libc.so.6
linux-vdso.so.1
```

```
cprocsp-rdr-gui-gtk
/lib/ld-linux.so.2
libatk-1.0.so.0
libatk-bridge-2.0.so.0
libatspi.so.0
libblkid.so.1
libbsd.so.0
libc.so.6
libcairo.so.2
libcairo-gobject.so.2
libdatrie.so.1
libdbus-1.so.3
libdl.so.2
libepoxy.so.0
libexpat.so.1
libffi.so.7
libfontconfig.so.1
libfreetype.so.6
libfribidi.so.0
libgcc_s.so.1
libgcrypt.so.20
libgdk_pixbuf-2.0.so.0
libgdk-3.so.0
libgio-2.0.so.0
libglib-2.0.so.0
libgmodule-2.0.so.0
libgobject-2.0.so.0
libgpg-error.so.0
libgraphite2.so.3
libgtk-3.so.0
libharfbuzz.so.0
liblz4.so.1
liblzma.so.5
libm.so.6
libmount.so.1
libpango-1.0.so.0
libpangocairo-1.0.so.0
libpangoft2-1.0.so.0
libpcre.so.3
libpcre2-8.so.0
libpixman-1.so.0
libpng16.so.16
libpthread.so.0
libresolv.so.2
librt.so.1
libselinux.so.1
libstdc++.so.6
libsystemd.so.0
libthai.so.0
libuuid.so.1
libwayland-client.so.0
libwayland-cursor.so.0
libwayland-egl.so.1
libX11.so.6
libXau.so.6
libxcb.so.1
libxcb-render.so.0
libxcb-shm.so.0
libXcomposite.so.1
libXcursor.so.1
libXdamage.so.1
libXdmcp.so.6
libXext.so.6
libXfixes.so.3
libXi.so.6
libXinerama.so.1
libxkbcommon.so.0
libXrandr.so.2
libXrender.so.1
libz.so.1
linux-vdso.so.1
```

```
cprocsp-rdr-pcsc
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
lsb-cprocsp-pkcs11
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
cprocsp-curl
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
libz.so.1
linux-vdso.so.1
```

```
cprocsp-cptools-gtk
/lib/ld-linux.so.2
libatk-1.0.so.0
libatk-bridge-2.0.so.0
libatspi.so.0
libblkid.so.1
libbsd.so.0
libc.so.6
libcairo.so.2
libcairo-gobject.so.2
libdatrie.so.1
libdbus-1.so.3
libdl.so.2
libepoxy.so.0
libexpat.so.1
libffi.so.7
libfontconfig.so.1
libfreetype.so.6
libfribidi.so.0
libgcc_s.so.1
libgcrypt.so.20
libgdk_pixbuf-2.0.so.0
libgdk-3.so.0
libgio-2.0.so.0
libglib-2.0.so.0
libgmodule-2.0.so.0
libgobject-2.0.so.0
libgpg-error.so.0
libgraphite2.so.3
libgthread-2.0.so.0
libgtk-3.so.0
libharfbuzz.so.0
libICE.so.6
liblz4.so.1
liblzma.so.5
libm.so.6
libmount.so.1
libpango-1.0.so.0
libpangocairo-1.0.so.0
libpangoft2-1.0.so.0
libpcre.so.3
libpcre2-8.so.0
libpixman-1.so.0
libpng16.so.16
libpthread.so.0
libresolv.so.2
librt.so.1
libselinux.so.1
libSM.so.6
libstdc++.so.6
libsystemd.so.0
libthai.so.0
libuuid.so.1
libwayland-client.so.0
libwayland-cursor.so.0
libwayland-egl.so.1
```

```
libX11.so.6
libXau.so.6
libxcb.so.1
libxcb-render.so.0
libxcb-shm.so.0
libXcomposite.so.1
libXcursor.so.1
libXdamage.so.1
libXdmcp.so.6
libXext.so.6
libXfixes.so.3
libXi.so.6
libXinerama.so.1
libxkbcommon.so.0
libXrandr.so.2
libXrender.so.1
libXxf86vm.so.1
libz.so.1
linux-vdso.so.1
```

```
cprocsp-rdr-cloud
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
cprocsp-stunnel
/lib/ld-linux.so.2
libaudit.so.1
libc.so.6
libcap-ng.so.0
libdl.so.2
libgcc_s.so.1
libm.so.6
libpam.so.0
libpthread.so.0
libstdc++.so.6
libutil.so.1
linux-vdso.so.1
```

```
cprocsp-stunnel-msspi
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
libutil.so.1
linux-vdso.so.1
```

```
cprocsp-ipsec-genpsk
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
cprocsp-ipsec-ike
/lib/ld-linux.so.2
libc.so.6
libgcc_s.so.1
libm.so.6
libstdc++.so.6
linux-vdso.so.1
```

```
lsb-cprocsp-rdr-accord
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
lsb-cprocsp-rdr-ancud
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
lsb-cprocsp-rdr-crypton
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
lsb-cprocsp-rdr-maxim
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
lsb-cprocsp-rdr-sobol
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
lsb-cprocsp-rdr-vityaz
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
cprocsp-rdr-cpfkc
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
cprocsp-rdr-edoc
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
cprocsp-rdr-emv
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
```

```
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-rdr-esmart
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-rdr-infocrypt
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-rdr-inpaspot
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1
```

```
cprocsp-rdr-jacarta
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-rdr-kst
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-rdr-mskey
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-rdr-novacard
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
```

```
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-rdr-rosan
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


cprocsp-rdr-rutoken
/lib/ld-linux.so.2
libc.so.6
libdl.so.2
libgcc_s.so.1
libm.so.6
libpthread.so.0
libstdc++.so.6
linux-vdso.so.1


ifd-rutokens
/lib/ld-linux.so.2
libc.so.6
libpthread.so.0
libusb-0.1.so.4
linux-vdso.so.1
```

Кроме того, пакету `lsb-cprocsp-capilite` для работы с сетью необходим либо пакет `cprocsp-curl`, либо пакет `curl` (последний можно взять из дистрибутива ОС, из поставки CSP или с сайта разработчика). При отсутствии этого пакета базовая функциональность сохранится, но такие функции работы с сетью как автоматическое выкачивание CRL или запрос сертификата на УЦ через утилиту `cryptcp` будут недоступны.

Пакету `lsb-cprocsp-rdr-pcsc` для работы со смарт-картами необходим пакет `libpcsclite` из дистрибутива ОС. В зависимости от того, какой дистрибутив Linux используется, название пакета может варьироваться (`libpcsclite`, `libpcsclite1`).

2.1.2 Требования к системным утилитам

LSB 4.0, раздел VI. Commands and Utilities

Для установки необходимого пакета `lsb-cprocsp-base` требуются утилиты:

- `cat`
- `chmod`
- `cp`
- `crontab`
- `echo`

- fgrep
- grep
- ln
- mkdir
- rm
- sed
- sysctl
- test
- true
- dpkg (только для Debian и Ubuntu)

Для установки всех остальных пакетов (за исключением cprocsp-drv-devel) достаточно подмножества этих утилит. Для установки cprocsp-drv-devel также необходима утилита uname.

2.1.3 Требования к файловой системе

LSB 4.x, раздел VI. Execution Environment 16. File System Hierarchy

Необходимы следующие разделы со следующими возможностями:

Таблица 1. Необходимые разделы

/opt/cprocsp	После установки дистрибутива для функционирования продукта достаточно прав только на чтение.
/etc/opt/cprocsp	После установки дистрибутива для функционирования продукта достаточно прав только на чтение. При изменении настроек и операциях с лицензией также необходимы права на запись.
/var/opt/cprocsp	Во время работы с CSP необходимы права на чтение и на запись. Содержимое директории должно сохраняться между перезагрузками.

2.1.4 Требования к службам

LSB 4.0, раздел VIII. System Initialization 20. System Initialization 20.1. Cron Jobs

Необходимо базовое функционирование cron.

Для использования в качестве отчуждаемого ключевого носителя USB-флэш-накопителя необходимо функционирование службы udev.

2.1.5 Требования к системе управления пакетами

LSB 4.0, раздел X. Package Format and Installation

Необходима поддержка механизма установки rpm.

2.2 Solaris

2.2.1 Требования к библиотекам ОС

СКЗИ КриптоПро CSP для своего функционирования требует следующие библиотеки базовой операционной системы (список библиотек по пакетам):

```
CPROcplx
libc.so.1
libCrun.so.1
libCstd.so.1
libdl.so.1
libm.so.1
libm.so.2
libmd.so.1
libmp.so.2
libnsl.so.1
libpthread.so.1
librt.so.1
libsocket.so.1
libthread.so.1


CPROcurlx
libc.so.1
libCrun.so.1
libCstd.so.1
libdl.so.1
libldap.so.5
libm.so.2
libmd.so.1
libmp.so.2
libnsl.so.1
libnspr4.so
libnss3.so
libnssutil3.so
libplc4.so
libplds4.so
libpthread.so.1
librt.so.1
libsasl.so.1
libsocket.so.1
libssl3.so
libthread.so.1
libz.so.1


CPROkc1x
libc.so.1
libCrun.so.1
libCstd.so.1
libdl.so.1
libm.so.2
libpthread.so.1
librt.so.1
libthread.so.1


CPROkc2x
libc.so.1
libCrun.so.1
libCstd.so.1
libdl.so.1
libm.so.1
libm.so.2
libmd.so.1
libmp.so.2
libnsl.so.1
libpthread.so.1
librt.so.1
libsocket.so.1
libthread.so.1


CPROlegacyx
libc.so.1
libCrun.so.1
libCstd.so.1
libdl.so.1
libm.so.1
libm.so.2
libpthread.so.1
librt.so.1
libthread.so.1


CPROrdnovax
libc.so.1
libdl.so.1
libm.so.2
libpthread.so.1
librt.so.1
libthread.so.1


CPROrdpx
libc.so.1
libCrun.so.1
libCstd.so.1
libdl.so.1
libm.so.2
libmd.so.1
libmp.so.2
libnsl.so.1
libpthread.so.1
librt.so.1
libsocket.so.1
libthread.so.1


CPROrdrx
libadm.so.1
libc.so.1
libCrun.so.1
libCstd.so.1
libdl.so.1
libm.so.1
libm.so.2
libmd.so.1
libmp.so.2
libnsl.so.1
libpthread.so.1
librt.so.1
libsocket.so.1
libthread.so.1
libvolmgt.so.1


CPROstnlx
libc.so.1
libCrun.so.1
libCstd.so.1
libdl.so.1
libm.so.1
libm.so.2
libmd.so.1
libmp.so.2
libnsl.so.1
libpam.so.1
libpthread.so.1
librt.so.1
libsocket.so.1
libthread.so.1
```

Кроме того, пакету `CPROcpl` для работы с сетью необходим либо пакет `CPROcurl` из поставки КриптоПро CSP, либо пакет `curl` (последний можно взять из дистрибутива ОС, из поставки КриптоПро CSP или с сайта разработчика). При

отсутствии этого пакета базовая функциональность сохранится, но такие функции работы с сетью как автоматическое выкачивание CRL или запрос сертификата на УЦ через утилиту cryptcp будут недоступны.

Пакету CPROrdp для работы со смарт-картами необходим пакет pcsclite (например, пакет SUNWpcsclite из дистрибутива ОС).

2.2.2 Требования к системным утилитам

Для установки необходимых пакетов CPRObase, CPROrdr необходимо функционирование утилит:

- cat
- chmod
- cp
- crontab
- echo
- fgrep
- grep
- ln
- mv
- rm
- sed
- sysctl
- test
- true

Для установки всех остальных пакетов за исключением CPROdrv и CPROdrvd достаточно подмножества этих утилит.

Для установки CPROdrv также необходимы утилит:

- add_drv
- isainfo
- rem_drv
- sync

Для установки CPROdrvd:

- add_drv
- isainfo
- rem_drv
- sync
- uname

2.2.3 Требования к файловой системе

Необходимы следующие разделы со следующими возможностями:

Таблица 2. Необходимые разделы файловой системы

/opt/cprocsp	После установки дистрибутива для функционирования продукта достаточно прав только на чтение.
/etc/opt/cprocsp	После установки дистрибутива для функционирования продукта достаточно прав только на чтение. При изменении настроек и операциях с лицензией также необходимы права на запись.
/var/opt/cprocsp	Во время работы с CSP необходимы права на чтение и на запись. Содержимое директории должно сохраняться между перезагрузками.

2.2.4 Требования к службам

Необходимо базовое функционирование cron.

Для работы с отчуждаемыми носителями типа USB flash drive необходимо функционирование службы Volume Management.

2.2.5 Требования к системе управления пакетами

Необходимо штатное функционирование системы управления пакетами.

3 Режим усиленного контроля использования ключей

Примечание. Использование СКЗИ без включения режима усиленного контроля использования ключей разрешается исключительно в тестовых целях.

Режим усиленного контроля использования ключей осуществляет контроль доверенности ключей проверки электронной подписи, срока действия ключей электронной подписи и ключевого обмена, а также корректности использования программных ДСЧ.

Это накладывает дополнительные требования к выполнению ряда операций, функционал которых предоставляется криптопровайдером:

1) Функции, при выполнении которых предполагается выработка случайных данных (CryptGenKey для временных ключей, CryptGenRandom), будут возвращать ошибку, если программный ДСЧ не был инициализирован с физического ДСЧ и провести инициализацию в данный момент невозможно (например, в системе не установлен ни один физический ДСЧ). Для исправления ошибки следует произвести установку хотя бы одного физического ДСЧ (например, БиоДСЧ, внешней гаммы, аппаратного ДСЧ), после чего произвести инициализацию программного ДСЧ одним из следующих способов:

а) воспользоваться утилитой csptest:

```
csptest -keyset -verifycontext -hard_rng
```

б) произвести выработку долговременного ключа с помощью функции CryptGenKey;

в) произвести вызов функции CryptSetProvParam с флагом PP_USE_HARDWARE_RNG.

2) Функция CryptVerifySignature() принимает на вход дескриптор открытого ключа, с помощью которого производится проверка подписи. При включённом режиме усиленного контроля использования ключей допустимо передавать этой функции дескрипторы следующих ключей:

а) Дескриптор ключа, извлечённого из контейнера (AT_SIGNATURE, AT_KEYEXCHANGE).

б) Дескриптор ключа, полученного из сертификата открытого ключа. Для получения этого дескриптора необходимо получить структуру CERT_CONTEXT, содержащую информацию о необходимом сертификате, с помощью функций CertCreateCertificateContext/CertFindCertificateInStore. Затем требуется вызвать функцию CryptImportPublicKeyInfoEx, передав ей указатель на структуру CERT_CONTEXT.pCertInfo.SubjectPublicKeyInfo типа CERT_PUBLIC_KEY_INFO. Функция вернёт необходимый дескриптор.

3) Функции, использующие долговременные ключи, будут возвращать ошибку в случае, если срок действия ключа истёк. Получить информацию о сроке действия ключа можно с помощью:

а) контрольной панели КриптоПро CSP (вкладка **Сервис — Протестировать**);

б) графической утилиты Инструменты КриптоПро (cptools) (вкладка **Контейнеры — Протестировать контейнер**);

в) утилиты csptest, например:

```
csptest -keyset -check -provname "Crypto-Pro GOST R 34.10-2012 Cryptographic Service
Provider" -cont "\\.\REGISTRY\contname" -password XXXX
```

г) вызова функции CryptGetKeyParam с параметром KP_NOTAFTER.

4) Для обеспечения контроля доверенности сертификатов открытых ключей при использовании СКЗИ под управлением ОС Windows при наличии корневого сертификата, установленного в доверенное хранилище КриптоПро локального компьютера (Доверенные корневые сертификаты КриптоПро CSP, CryptoPro Trusted Roots, CryptoProTrustedStore), осуществляется построение цепочки сертификатов. В случае, если такую цепочку построить невозможно, функция CertFindCertificateInStore, на вход которой передаётся структура CERT_CONTEXT, соответствующая недоверенному сертификату, и функция проверки подписи CMS (CryptMsgControl), в случае если

CMS-подпись не содержит указание на доверенный сертификат, будут завершаться с ошибкой. Для того, чтобы избавится от ошибки, необходимо установить необходимый корневой сертификат в доверенное хранилище КриптоПро (см. ЖТЯИ.00101-03 91 02. Руководство администратора безопасности. Windows).

3.1 Включение режима в Apple iOS и Google Android

Для включения режима усиленного контроля использования ключей в СКЗИ под управлением ОС iOS или Android в конфигурационный файл `config.ini` в раздел [Parameters] необходимо добавить строку:

```
StrengthenedKeyUsageControl = 1
```

Конфигурационный файл `config.ini` расположен в ресурсах приложения (`Resources` для iOS, `..\SharedLibrary\res\raw` для Android).

Для обеспечения корректного функционирования провайдера в части выработки электронной подписи, а также работы с временными ключами (в частности, для работы в рамках TLS-соединения без аутентификации клиента) и генерации случайных данных, необходимо произвести выработку долговременных ключей, предварительно проверив, что зарегистрирован хотя бы один ДСЧ.

По умолчанию в СКЗИ под управлением ОС iOS или Android в качестве физического ДСЧ зарегистрирован БиоДСЧ.

В ОС Android для использования внешней гаммы в конфигурационный файл `config.ini` необходимо добавить следующие секции:

```
[Random\Dsrf]
"DLL"="librdrdsrf.so"
[Random\CPSD]
"DLL"="librdrdsrf.so"

[Random\CPSD\Default]
Name = "cpsd rng"
Level = 3
"/db1/kis_1" = "путь//dsrf/db1/kis_1"
"/db2/kis_1" = "путь//dsrf/db2/kis_1"
[Random\DSRF_EX]
"DLL"="librdrdsrf.so"
...
[Random\BIO_AUI\Default]
Name = "\xc1\xe8\xee\xeb\xee\xe3\xe8\xf7\xe5\xf1\xea\xe8\xe9 \xf2\xe5\xea\xf1\xf2\xee\xe2\xfb\xe9"
Level = 5
...
[apppath]
...
"librdrdsrf.so" = "/opt/cprocsp/lib/amd64/librdrdsrf.so"
...
```

Примечание. «Путь» к файлам внешней гаммы должен указываться с учетом наличия прав на чтение/запись в файл kis_1 (например, папка приложения). Внешняя гамма вырабатывается с использованием «АРМ выработки внешней гаммы» (подробнее см. ЖТЯИ.00101-03 94 01. АРМ выработки внешней гаммы).

4 Контроль целостности мобильных приложений со встроенным СКЗИ

4.1 Порядок обеспечения целостности приложения

Для исключения рисков преднамеренной и непреднамеренной модификации МП, загруженного в магазин приложений, разработчику МП необходимо:

1) Зафиксировать исполняемый код разработанного и прошедшего оценку влияния МП перед загрузкой в магазин приложений.

2) После загрузки МП в магазин приложений осуществить контрольное скачивание МП и проверить соответствие значимых модулей скачанного МП зафиксированным на этапе 1).

3) Сформировать эталонные контрольные суммы для проверки целостности МП конечным пользователем. Данные значения должны фиксироваться (например, на сайте разработчика или в эксплуатационной документации) и передаваться пользователю приложения для выполнения процедуры проверки целостности приложения.

4.1.1 Apple iOS

Для контроля целостности МП, предназначенного для использования под управлением ОС iOS, на ПК разработчика под управлением ОС Mac OS необходимо установить следующее программное обеспечение:

- ПО iMazing (https://imazing.com/);
- ПО Frida (https://frida.re/),

Также ПО Frida должно быть установлено на мобильное устройство с jailbreak (получены права доступа суперпользователя);

- инструменты Frida: frida-tools (https://frida.re/docs/installation/) и bagpak (https://github.com/ChiChou/bagbak).

Также необходимо получить доверенным образом средство контроля целостности (утилита `cpverify` из состава СКЗИ или иное сертифицированное средство).

Для контроля целостности МП, загружаемого на сервер App Store, разработчику приложения необходимо выполнить следующие действия:

1) Скомпилировать и собрать МП для публикации в App Store.

Настройка сборки МП должна осуществляться в соответствии с файлом `CPROCSP.xcframework\ReadMe.txt` из состава архива `CPROCSP.xcframework.tgz`, поставляемого в комплекте разработчика в составе СКЗИ.

2) Выполнить публикацию МП в App Store.

3) Установить МП на мобильное устройство с jailbreak.

4) С использованием ПО Frida для получения раскодированного ipa-файла выполнить следующие действия:

- мобильное устройство подключить к ПК (в случае использования на ПК ОС Windows дать согласие мобильному устройству на доверие компьютеру) через USB-соединение;
- выполнить команду (флаг -z создает .ipa файл):

```
bagbak -z APP
```

- результатом выполнения команды будет раскодированный ipa-файл в директории dump.

5) Сравнить секции исполняемого кода МП, загружаемого в App Store в п. 2, и полученного с использованием ПО Frida в п. 4.

6) При искажении или подмене кода выполнить удаление МП из App Store.

7) При неизменности сравниваемого исполняемого кода для получения ipa-файла выполнить следующие действия с использованием ПО iMazing:

- мобильное устройство подключить к ПК через USB-соединение;
- запустить ПО iMazing и выбрать подключенное устройство с выбором пункта «Управление приложениями» (Рисунок 1);

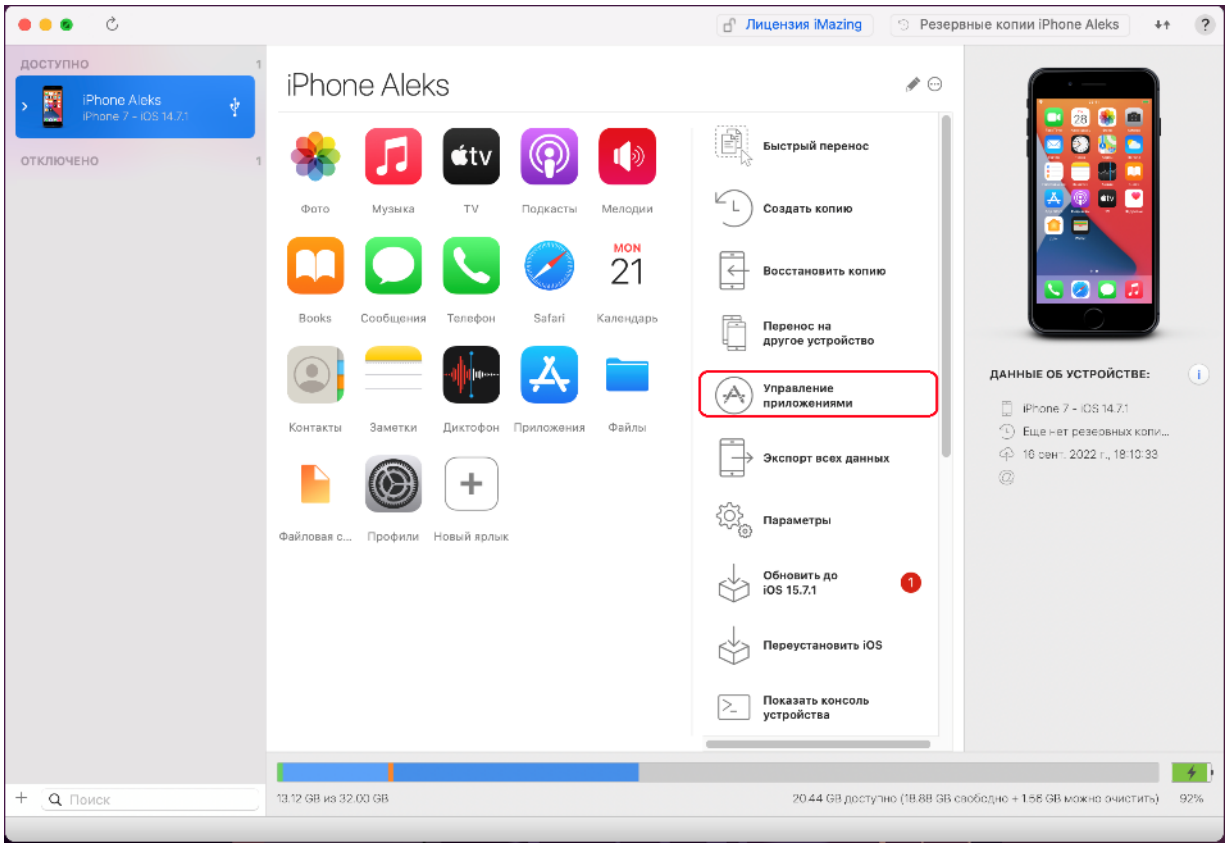

Рисунок 1. Интерфейс ПО iMazing

- для МП через контекстное меню выбрать команду «Загрузить в библиотеку» в разделе «Медиатека», при необходимости ввести Apple ID и пароль в открывшемся окне;
- далее через контекстное меню выбрать ставшую активной команду «Экспорт .IPA» (Рисунок 2), при необходимости ввести Apple ID и пароль в открывшемся окне;

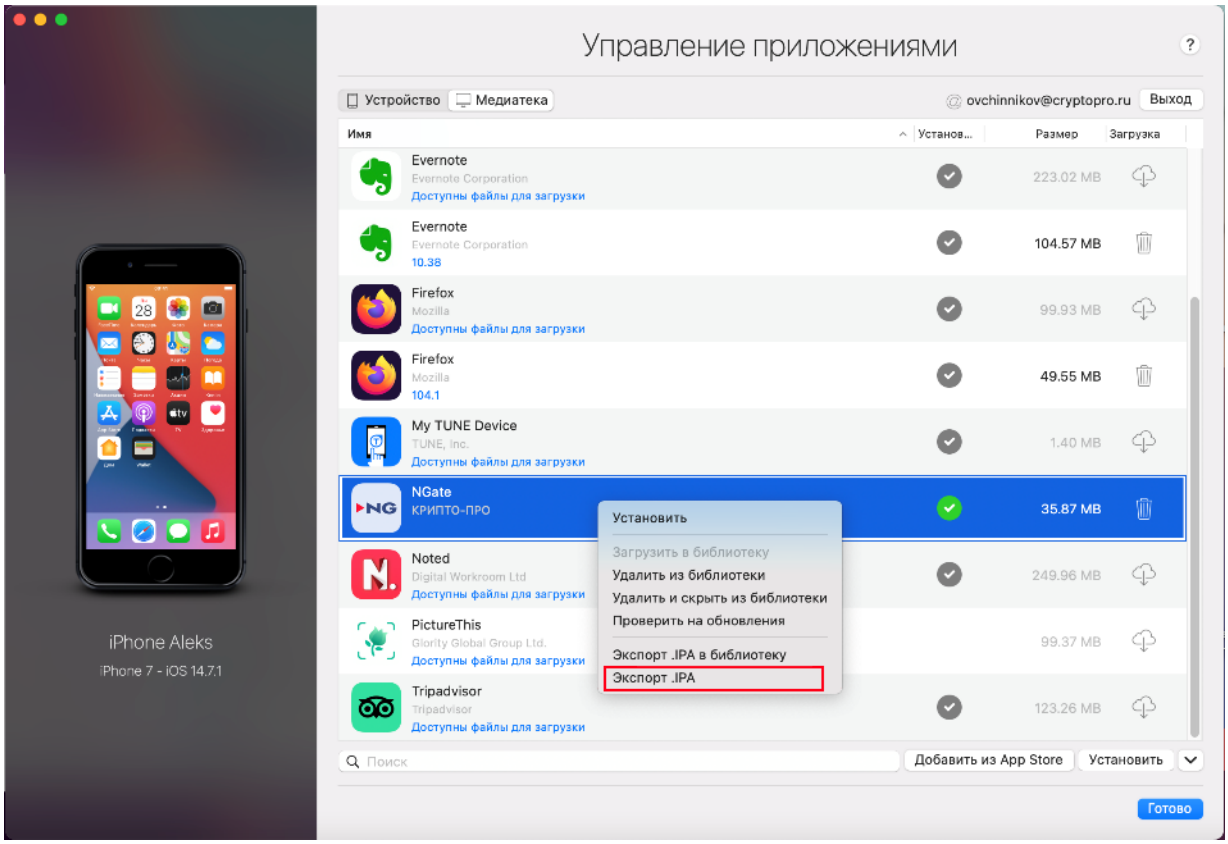

Рисунок 2. Экспорт .IPA

- выбрать необходимый путь и сохранить ipa-файл.

8) Выполнить проверку соответствия ipa-файлов п. 3 и п. 7 следующим образом:
   - привести мобильное устройство с jailbreak в начальное состояние (до установки МП на устройство);
   - с использованием ПО iMazing установить на мобильное устройство ipa-файл, полученный в п. 7, с ПК, выбрав пункт «Установить файл .IPA» в разделе «Устройство» (Рисунок 3);

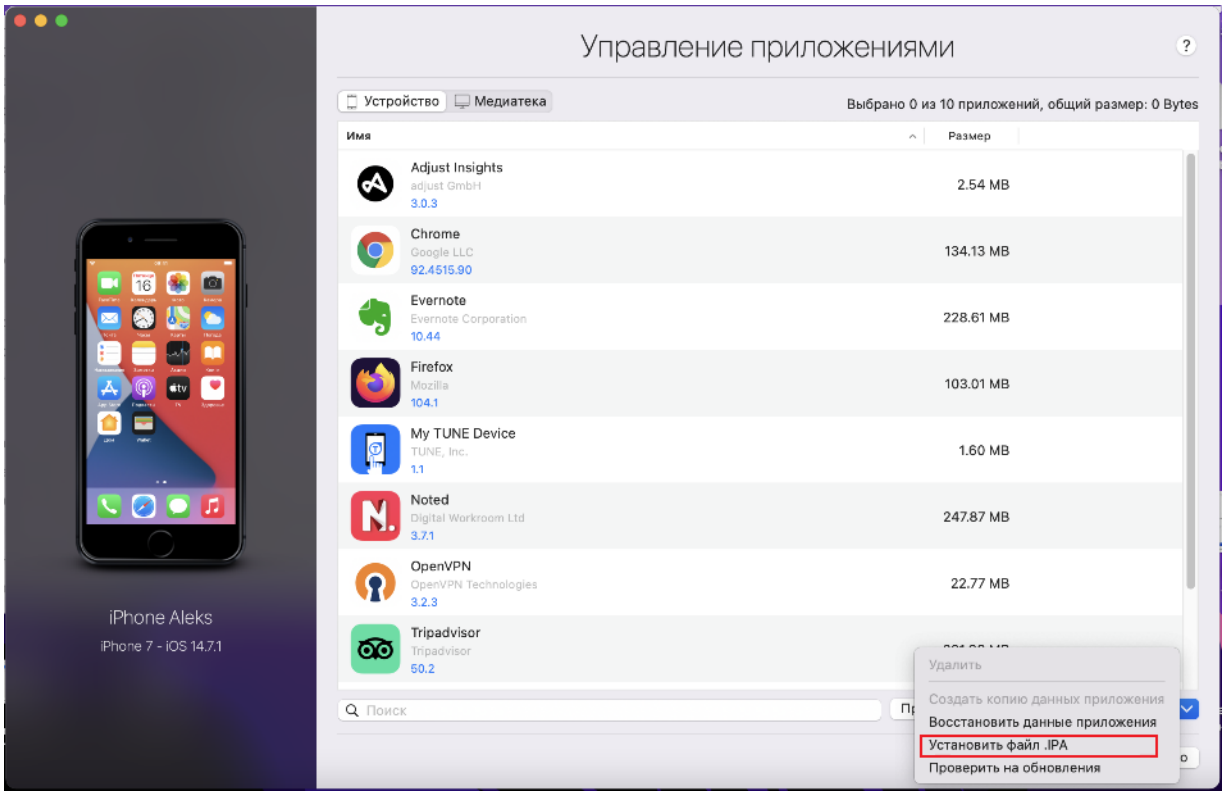

Рисунок 3. Установка .IPA

- повторить шаги п. 4 для установленного МП;
- сравнить раскодированный ipa-файл с ipa-файлом п. 5.

9) При успешном сравнении сформировать с использованием утилиты `cpverify` (или иного сертифицированного средства) контрольные суммы для всех файлов в папке **Payload** .ipa-файла, кроме файлов директорий `_CodeSignature`, `SC_Info` и файла с расширением `.mobileprovision` (данные файлы содержат информацию о подписи приложения, и могут отличаться во время загрузки на разные устройства; туда входит информация об аккаунте пользователя, скачивающего ПО).

10) Опубликовать полученные контрольные суммы для проверки целостности ipa-файла.

4.1.2 Google Android

Для контроля целостности МП, предназначенного для использования под управлением ОС Android, на ПК разработчика необходимо установить утилиту командной строки Android Device Bridge (ADB) и архиватор (WinRAR).

Также необходимо получить доверенным образом средство контроля целостности (утилита `cpverify` из состава СКЗИ или иное сертифицированное средство).

Для контроля целостности МП, загружаемого на сервер Play Маркет, разработчику приложения необходимо выполнить следующие действия:

1) Скомпилировать и собрать МП для публикации в Play Маркет, не подписывая apk-файл ключом магазина приложений.

2) Вычислить значения хэш-функции (в соответствии с ГОСТ Р 34.11-2012, 256 бит) для всех файлов `classes[n].dex` и `*.aap` (модулей, содержащих код приложения) с использованием утилиты `cpverify` (или иного сертифицированного средства):

```
cpverify -mk <file> -alg GR3411_2012_256
```

3) Полученные значения записать в файл и поместить в ресурсы внутри неподписанного apk-файла. Также необходимо зафиксировать полученные эталонные контрольные суммы.

4) Подписать итоговый дополненный apk-файл.

5) Выполнить публикацию МП в Play Маркет.

6) Установить МП на мобильное устройство.

7) Подключить мобильное устройство к ПК через USB-соединение.

8) Для вывода всех установленных на мобильном устройстве apk-файлов ввести в командной строке следующую строку:

```
adb shell pm list packages
```

Для оптимизации поиска используйте утилиту findstr:

```
adb shell pm list packages -f | findstr <Часть наименования apk-файла>
```

9) В списке всех установленных на мобильном устройстве apk-файлов найти необходимый пакет и скопировать полное название пакета (без приставки «package:»).

10) Выполнить поиск расположения apk-файла на мобильном устройстве следующей командой:

```
adb shell pm path <Полное наименование apk-файла>
```

11) Скопировать полный путь расположения apk-файла (без приставки «package:»).

12) Скачать apk-файл на ПК:

```
adb pull <Полный путь расположения apk-файла>
```

13) Распаковать apk-файл с использованием архиватора.

14) Вычислить значения хэш-функции (в соответствии с ГОСТ Р 34.11-2012, 256 бит) для всех файлов `classes[n].dex` и `*.aap` (модули, содержащие код приложения) с использованием утилиты `cpverify` (или иного сертифицированного средства):

```
cpverify -mk <file> -alg GR3411_2012_256
```

15) Сравнить со значениями, записанными в файле с хэш-значениями, находящегося в ресурсах apk-файла, и зафиксированными эталонными значениями (см. п. 3).

16) При искажении или подмене кода выполнить удаление МП из Play Маркета.

17) При успешном сравнении опубликовать зафиксированные эталонные значения для проверки целостности apk-файла.

4.2 Реализация механизма контроля целостности модулей приложения

Помимо модулей МП со встроенным СКЗИ в перечень контролируемых файлов необходимо вносить все модули и подмодули из состава СКЗИ, от которых есть зависимость. Признаком полноты списка должна служить возможность корректной работы МП после любой модификации (в т.ч. удаления) компонентов МП, не вошедших в этот список.

4.2.1 Apple iOS

Автоматический пусковой контроль целостности обеспечивается средствами встраиваемого в МП СКЗИ КриптоПро CSP. Для этого необходимо корректным образом настроить сборку МП (см. файл CPROCSP.xcframework\ ReadMe.txt из состава архива CPROCSP.xcframework.tgz, поставляемого в комплекте разработчика в составе СКЗИ КриптоПро CSP; в частности, необходимо включить скрипт ConfigureApplication в процедуру сборки МП, см. шаг 8 параграфа 2 «Сборка примера в Xcode»).

Пусковой контроль целостности

Контроль целостности реализуется МП встроенными средствами фреймворка СКЗИ КриптоПро CSP путем обращения к методу CryptGetProvParam с параметром PP_SELFTEST. Дополнительно может быть использован флаг CP_ CRYPT_SET_TESTER_STATUS, который указывает СКЗИ КриптоПро CSP сохранить в результатах тестирования подробный статус самотестирования.

При вызове CryptGetProvParam код последней ошибки может быть установлен равным ERROR_MORE_DATA в случае, если буфер для хранения результата проверки был выделен размера меньше необходимого. В случае обнаружения ошибок в работе или нарушения целостности код последней ошибки устанавливается равным NTE_BAD_ SIGNATURE (а вызов CryptGetProvParam вернёт значение False).

Результат тестирования и проверки целостности записывается в возвращаемый параметр в виде структуры PROV_ PP_SELFTEST. Содержащееся в заголовке структуры поле TesterFlags в случае успешного завершения проверок должно быть равно 0.

В случае ошибки при тестировании или обнаружения нарушения целостности выполнение криптографических функций СКЗИ будет заблокировано.

Встроенный периодический контроль целостности

Периодический и динамический контроль целостности реализуются МП автоматически встроенными средствами фреймворка СКЗИ КриптоПро CSP. В случае ошибки при тестировании или обнаружения нарушения целостности будет заблокировано выполнение следующих вызовов:

- CryptAcquireContext
- CryptCreateHash
- CryptDecrypt
- CryptDeriveKey
- CryptEncrypt
- CryptExportKey
- CryptGenKey
- CryptHashData
- CryptHashSessionKey
- CryptImportKey
- CryptSignHash
- CryptVerifySignature

Данные вызовы будут устанавливать код последней ошибки (доступный вызовом GetLastError) равным ERROR_ FILE_CORRUPT, если нарушена целостность, и CRYPT_E_SECURITY_SETTINGS, если тестирование выявило ошибку в работе.

Период проверки задается в конфигурационном файле СКЗИ КриптоПро CSP в параметре TesterPeriod и по умолчанию равен значению 600 секунд.

Регламентный контроль целостности по запросу пользователя

Дополнительно для выполнения предъявляемых к МП требований следует встроить в конечное приложение кнопку, по нажатии на которую будет проверяться целостность приложения, например, одним из следующих способов:

1) в обработчике нажатия на кнопку выполнять обращение к методу CryptGetProvParam с параметром PP_ SELFTEST и флагом CP_CRYPT_SET_TESTER_STATUS, после чего проверять возвращаемый код возврата и сообщать о результате проверки целостности (см. пример выше);

2) в обработчике нажатия на кнопку выполнять вызов панели управления от CSP, в которой будет доступна возможность запустить проверку целостности МП.

Пример обработчика нажатия на такую кнопку, вызывающую панель CSP (с учетом включения ресурсов PaneViewController.xib и PaneViewControllerIPhone.xib из Resources фреймворка в составе приложения):

```
-(void) launchPane
{
  if(UI_USER_INTERFACE_IDIOM() == UIUserInterfaceIdiomPad)
    CPROPane = [[PaneViewController alloc]initWithNibName:@"PaneViewController" bundle:nil];
  else
    CPROPane = [[PaneViewController alloc]initWithNibName:@"PaneViewControllerIPhone" bundle:
    nil];
  UINavigationController * cont = [[UINavigationController alloc]initWithRootViewController:
    CPROPane];
  [cont setModalPresentationStyle:UIModalPresentationFullScreen];
  [self presentViewController:cont animated:YES completion:nil];
}
```

4.2.2 Google Android

Контроль целостности обеспечивается средствами встраиваемого в МП СКЗИ КриптоПро CSP на двух уровнях:

1) для встраиваемого СКЗИ и *.so-библиотек приложения;

2) для всего исполняемого кода МП.

Для обеспечения контроля целостности на уровне СКЗИ и дополнительных *.so-библиотек необходимо корректным образом подготовить контрольные суммы для всех соответствующих файлов следующим образом:

1) нужно посчитать контрольные суммы на дополнительные библиотеки (которых нет в провайдере CSP) по алгоритму GR3411_2012_256 (например, вызовом cpverify -mk <название_библиотеки> -alg GR3411_2012_256);

2) нужно сохранить эти контрольные суммы в отдельный файл, который должен называться специальным образом.

Сохранять нужно в формате:

<название_библиотеки1>=<значение_контрольной_суммы1>
<название_библиотеки2>=<значение_контрольной_суммы2>

Файл должен называться в формате extra_digests*** в зависимости от архитектуры:

- «extra_digests32.properties» — android arm;
- «extra_digests64.properties» — android arm64-v8a;
- «extra_digestsx86» — android x86;
- «extra_digestsx86_64» — android x86_64.

3) Файл extra_digests*** должен находиться в папке raw/ ресурсов перед сборкой приложения. Если файла нет, то автоматическая проверка этого файла выполняться не будет.

Для обеспечения контроля целостности всего исполняемого кода МП необходимо подготовить контрольные суммы для всех *.dex и *.so файлов следующим образом:

1) Скомпилировать МП.

2) До подписи apk/aap-файла ключом магазина приложений вычислить значения хэш-функции (в соответствии с ГОСТ Р 34.11-2012 с длиной хэша 256 битов) для всех файлов classes[n].dex и *.so (модулей, содержащих код приложения) с помощью команды: cpverify -mk <название_библиотеки> -alg GR3411_2012_256

3) Полученные значения записать в файл и поместить в ресурсы внутри неподписанного apk-файла (в файл dexdigests для *.dex файлов и в файл extra_digests*** для *.so файлов).

4) После обновления apk/aab необходимо:

- для apk – выровнять (`zipalign -p -f 4`) и подписать apk-файл (`apksigner`);
- для aab – подписать (`jarsigner`) и выровнять (`zipalign -p -f 4`) aab-файл.

Пусковой контроль целостности

Перед началом проверки при запуске приложения в его главном потоке необходимо инициализировать объект `ru.CryptoPro.JCSP.NCSPConfig` (в случае Android native CSP SDK) или объект `ru.CryptoPro.JCSP.CSPConfig` (в случае Android Java CSP SDK) вызовом статического метода `init`.

Контроль целостности выполняется внутри метода `init`. Если в процессе инициализации проверка целостности завершилась с ошибкой, инициализация объекта также завершится с ошибкой.

Для получения сведений о выполненном в процессе `init` контроле целостности необходимо:

1) получить объект интерфейса `ru.CryptoPro.JCSP.CSPProviderInterface` провайдера (вызовом статического метода `ru.CryptoPro.JCSP.CSPConfig.getCSPProviderInfo`, в случае использования Android native CSP SDK — `ru.CryptoPro.JCSP.NCSPConfig.getCSPProviderInfo`);

2) получить объект интерфейса `IntegrityInterface` вызовом метода `getIntegrity` интерфейса `CSPProviderInterface` для получения результата контроля целостности;

3) обработать результат проверки целостности, возможны три кода ошибки:

- CHECK_INTEGRITY_SUCCESS (0) — целостность не нарушена;
- CHECK_INTEGRITY_INVALID (1) — целостность нарушена;
- CHECK_INTEGRITY_UNKNOWN (-1) — состояние целостности не известно.

4) код метода активации поместить в блок перехвата ошибок `try..catch`, который при возникновении ошибки зафиксирует исключение с соответствующей ошибкой.

В случае успешного выполнения метода `init` для самостоятельной проверки целостности необходимо:

1) запустить проверку целостности (СКЗИ, дополнительных `*.so`-библиотек и кода приложения в `*.dex`-файлах), вызвав метод `check` (с параметром `true`) интерфейса `IntegrityInterface`;

2) обработать результат проверки целостности (соответствующие коды ошибок см. выше).

При получении кода ошибки `CHECK_INTEGRITY_INVALID` или `CHECK_INTEGRITY_UNKNOWN` вызывающее приложение должно заблокировать возможность последующего использования криптографических функций и сообщить пользователю о нарушении целостности.

Подробности об используемых для контроля целостности классах приведено в описании API из пакета `JInitCSP-javadoc.jar`.

Пример исходного кода процедуры проверки целостности:

```
import ru.CryptoPro.ACSPClientApp.util.Logger;
import ru.CryptoPro.JCSP.CSPConfig;
import ru.CryptoPro.JCSP.CSPProviderInterface;
import ru.cprocsp.ACSP.tools.integrity.CSPIntegrityConstants;
import ru.cprocsp.ACSP.tools.integrity.IntegrityInterface;

protected void executeOne() throws Exception {

  CSPProviderInterface providerInfo = CSPConfig.getCSPProviderInfo();
  IntegrityInterface integrity = providerInfo.getIntegrity();
  Logger.log("Checking (v1)...");

  Logger.log("Last check date: " + integrity.getLastDate());
```

```
14  int status = integrity.getLastStatus();
15
16  switch (status) {
17
18    case CSPIntegrityConstants.CHECK_INTEGRITY_SUCCESS: {
19      Logger.log("Status: completed");
20    }
21    break;
22
23    case CSPIntegrityConstants.CHECK_INTEGRITY_INVALID: {
24      Logger.log("Status: failed");
25    }
26    break;
27
28    default: {
29      Logger.log("Status: unknown");
30    }
31
32  } // switch
33
34  int result = integrity.check(true);
35
36  Logger.log("Check completed.");
37  Logger.log("Result: " + result);
38
39  if (result != CSPIntegrityConstants.CHECK_INTEGRITY_SUCCESS) {
40    throw new Exception("Integrity corrupted.");
41  } // if
42
43  Logger.log("Last check date: " + integrity.getLastDate());
44  status = integrity.getLastStatus();
45
46  switch (status) {
47
48    case CSPIntegrityConstants.CHECK_INTEGRITY_SUCCESS: {
49      Logger.log("Status: completed");
50    }
51    break;
52
53    case CSPIntegrityConstants.CHECK_INTEGRITY_INVALID: {
54      Logger.log("Status: failed");
55    }
56    break;
57
58    default: {
59      Logger.log("Status: unknown");
60    }
61
62  } // switch
63
64  // A readable list of checked libraries
65  // and their hashes.
66
67  List<String> items = integrity.getItems(null, null, "");
68
69  for (String item : items) {
```

```
70      Logger.log("\t" + item);
71    } // for
72
73  }
```

Встроенный периодический контроль целостности

В МП необходимо реализовать таймер, согласно которому каждые 10 минут работы МП вызывается процедура контроля целостности средствами СКЗИ КриптоПро CSP для самого СКЗИ, дополнительных `*.so`-библиотек и кода приложения в `*.dex`-файлах (например, тем же образом, как приведено выше).

Регламентный контроль целостности по запросу пользователя

Для выполнения предъявляемых требований в МП следует встроить в конечное приложение кнопку, по нажатии на которую будет вызываться проверка целостности средствами СКЗИ КриптоПро CSP для самого СКЗИ, дополнительных `*.so`-библиотек и кода приложения в `*.dex`-файлах (например, тем же образом, как приведено выше).

Приложение 1
Интерфейс cpcurl

curl_global_init

Функция инициализации работы библиотеки, должна быть вызвана единожды перед началом использования библиотеки.

Потокобезопасная: нет

Входные параметры: flags

CURL_GLOBAL_ALL	инициализация всех доступных возможностей библиотеки, включает в себя все флаги, кроме CURL_GLOBAL_ACK_EINTR
CURL_GLOBAL_SSL	инициализация механизмов для установки TLS-соединений
CURL_GLOBAL_WIN32	инициализация библиотеки работы с сокетами в системах семейства Windows (необходима для работы в данных ОС)
CURL_GLOBAL_NOTHING	отсутствие инициализации всех дополнительных механизмов
CURL_GLOBAL_DEFAULT	инициализация механизмов по умолчанию, в текущей версии равнозначно CURL_GLOBAL_ALL
CURL_GLOBAL_ACK_EINTR	флаг учёта сигнала ошибки EINTR при работе при установке соединения

Возвращаемое значение: 0 в случае успеха или ненулевое значение при ошибке

curl_global_cleanup

Функция деинициализации работы библиотеки, должна быть вызвана единожды после завершения использования библиотеки.

Потокобезопасная: нет

Входные параметры: отсутствуют

curl_easy_init

Функция инициализации сессии работы с библиотекой.

Потокобезопасная: да

Входные параметры: отсутствуют

Возвращаемое значение: дескриптор, используемый в других функциях, или NULL

curl_easy_cleanup

Функция освобождения ресурсов, занятых в рамках указанной сессии работы с библиотекой. Закрывает оставшиеся на момент вызова соединения.

Потокобезопасная: да

Входные параметры: handle (дескриптор, должен быть ранее получен вызовом `curl_easy_init`, если равен NULL — функция не выполняет никаких действий)

curl_easy_setopt

Функция изменения параметров работы TLS-сессии.

Входные параметры:

- handle (должен быть ранее получен вызовом `curl_easy_init`)
- option (допускается устанавливать только 1 опцию для каждого вызова функции)
- parameter (параметр опции)

Следующие опции поддерживаются для **TLS-соединения с поддержкой ГОСТ**:

CURLOPT_STRICT_GOST	задаёт необходимость выполнения проверки соответствия алгоритма ключа проверки ЭП для каждого сертификата из цепочки сертификатов TLS-сервера и соответствия выбранного для TLS-соединения набора шифрования одному из следующих: ◦ TLS_CIPHER_2001 ◦ TLS_CIPHER_2012 ◦ TLS_GOSTR341112_256_WITH_28147_CNT_IMIT ◦ TLS_GOSTR341112_256_WITH_MAGMA_CTR_OMAC ◦ TLS_GOSTR341112_256_WITH_KUZNYECHIK_CTR_OMAC Использование интерфейса cpcurl допускается только в случае установки опции CURLOPT_STRICT_GOST не равной 0.

CURLOPT_SSLVERSION, CURLOPT_PROXY_SSLVERSION	номер версии протокола, применяемого для установки защищённого соединения: ○ CURL_SSLVERSION_TLSv1 ○ CURL_SSLVERSION_SSLv2 ○ CURL_SSLVERSION_SSLv3 ○ CURL_SSLVERSION_TLSv1_0 ○ CURL_SSLVERSION_TLSv1_1 ○ CURL_SSLVERSION_TLSv1_2 ○ CURL_SSLVERSION_TLSv1_3
CURLOPT_SSLCERT, CURLOPT_PROXY_SSLCERT	устанавливают используемый в TLS-соединении сертификат пользователя (задаётся SHA-1 отпечатком сертификата из локального хранилища), определяющий используемые алгоритмы шифрования; по умолчанию выбор сертификата из хранилища "My" выполняется на уровне SSPI
CURLOPT_SSLCERTTYPE, CURLOPT_PROXY_SSLCERTTYPE	устанавливают тип хранилища, в котором будет производиться поиск используемого в TLS-соединении сертификата пользователя, определяющий используемые алгоритмы шифрования; по умолчанию выбор сертификата из хранилища "My" выполняется на уровне SSPI
CURLOPT_CERTINFO	задаёт необходимость сбора сведений и данных о цепочке сертификатов для сертификата сервера TLS-соединения; после эти сведения можно будет получить вызовом curl_easy_getinfo с опцией CURLINFO_CERTINFO
CURLOPT_PINNEDPUBLICKEY, CURLOPT_PROXY_PINNEDPUBLICKEY	задаёт файл «закреплённого» ключа проверки электронной подписи сервера TLS-соединения; в случае заданного файла при установке соединения полученный от сервера ключ в сертификате будет проверен на совпадение с «закреплённым» – если ключи не совпадут, установка соединения будет отменена
CURLOPT_USE_SSL	задаёт «уровень» использования TLS-соединения при обмене с сервером по протоколу FTP, SMTP, POP3 или IMAP (не использовать/попробовать установить TLS-соединение сервером/требовать передачи управляющих команд в рамках TLS-соединения/требовать передачи всех данных в рамках TLS-соединения, по умолчанию TLS не используется); данный параметр определяет, будет ли обеспечиваться защита передаваемых данных по протоколу TLS в случае соединения по протоколам FTP, SMTP, POP3 или IMAP

CURLOPT_SOCKOPTFUNCTION	устанавливает указанную функцию в качестве специального обработчика, вызываемого между открытием сокета соединения и до установки соединения с сервером в целях смены свойств соединения, параметрами функции являются созданный дескриптор сокета, его тип и указатель на пользовательские данные, заданные предварительном вызовом функции с опцией CURLOPT_SOCKOPTDATA
CURLOPT_CLOSESOCKETFUNCTION	устанавливает указанную функцию в качестве специального обработчика, вызываемого взамен функции закрытия сокета соединения с сервером в целях модификации свойств соединения; параметрами функции являются созданный дескриптор сокета и указатель на пользовательские данные, заданные предварительном вызовом функции с опцией CURLOPT_CLOSESOCKETDATA
CURLOPT_RESOLVER_START_FUNCTION	устанавливает указанную функцию в качестве специального обработчика, вызываемого перед работой по каждому Resolve-запросу определения DNS-имени; параметрами функции являются указатель на объект контекста «разрешителя» DNS-имени, указатель на зарезервированный параметр и указатель на пользовательские данные, заданные предварительном вызовом функции с опцией CURLOPT_RESOLVER_START_DATA
CURLOPT_DEFAULT_PROTOCOL	задаёт протокол, который по умолчанию будет использоваться для установки соединения в случае, когда URL-строкой протокол связи не задан
CURLOPT_SSL_SESSIONID_CACHE	определяет возможность хранения ID сессии TLS-соединения и повторного его использования (по умолчанию хранение ID-сессии разрешено); в случае активации данной опции при повторном подключении к серверу (и вызове InitializeSecurityContextA) будут использоваться сведения о его имени, сохранённые с последнего соединения
CURLOPT_REDIR_PROTOCOLS	определяет перечень протоколов, для которых разрешён redirect-переход, по умолчанию переход запрещён только для типов протоколов CURLPROTO_FILE и CURLPROTO_SCP
CURLOPT_NOPROXY	задаёт игнорирование настроек прокси-сервера при связи с указанным сервером (соединение будет устанавливаться только напрямую)
CURLOPT_CAINFO, CURLOPT_PROXY_CAINFO	устанавливают название каталога/набора сертификатов удостоверяющего центра, с использованием которых будет проводиться проверка сертификата сервера, если необходимость такой проверки задана (согласно архитектуре libcurl)

CURLOPT_CAPATH, CURLOPT_PROXY_CAPATH	устанавливают название пути до каталога/набора сертификатов удостоверяющего центра, с использованием которых будет проводиться проверка сертификата сервера, если необходимость такой проверки задана (согласно архитектуре libcurl)
CURLOPT_CRLFILE, CURLOPT_PROXY_CRLFILE	устанавливают название файла списка отозванных сертификатов, с использованием которых будет проводиться проверка сертификата сервера, если необходимость такой проверки задана (согласно архитектуре libcurl)
CURLOPT_ISSUERCERT	устанавливает название сертификата подписанта, с использованием которых будет проводиться проверка сертификата сервера, если необходимость такой проверки задана (согласно архитектуре libcurl; сертификат подписанта должен оказаться в цепочке сертификатов сервера)

Следующие опции, связанные с управлением аутентификационными данными, настройкой прокси-сервера и установкой TLS-соединения, поддерживаются библиотекой cpcurl, однако их установка **не влияет на параметры TLS-соединения с поддержкой ГОСТ**:

CURLOPT_SSL_CIPHER_LIST, CURLOPT_PROXY_SSL_CIPHER_LIST	параметр в установке соединения не участвует, тип провайдера выбирается либо при заданном сертификате пользователя (использованием опций CURLOPT_SSLCERT и CURLOPT_SSLCERTTYPE), либо используется сертификат из хранилища "My", выбранный на уровне SSPI
CURLOPT_TLS13_CIPHERS, CURLOPT_PROXY_TLS13_CIPHERS	параметр в установке соединения не участвует
CURLOPT_RANDOM_FILE	задаёт строку, содержащую путь до файла с исходной последовательностью, инициализирующей программный датчик случайных чисел в OpenSSL
CURLOPT_UNRESTRICTED_AUTH	настройка отправки аутентификационных данных даже при смене наименования сервера
CURLOPT_USERPWD	установка пары значений «имя пользователя:пароль»
CURLOPT_USERNAME	установка имени пользователя
CURLOPT_PASSWORD	установка пароля пользователя
CURLOPT_DEBUGFUNCTION	устанавливает функцию-обработчик отладочной печати, вызываемой в процессе работы библиотеки в случае предварительной настройки verbose-режима

CURLOPT_HEADERFUNCTION	устанавливает указанную функцию в качестве обработчика процедуры записи заголовков передаваемых в рамках соединения пакетов (по умолчанию используется системный вызов fwrite, но допустимо устанавливать свою процедуру разбора); параметрами функции являются указатель на буфер с заголовком открытого пакета, размер этого буфера и указатель на дополнительные пользовательские данные, заданные предварительном вызовом функции с опцией CURLOPT_HEADERDATA
CURLOPT_WRITEFUNCTION	устанавливает указанную функцию в качестве обработчика процедуры записи данных в формируемый в рамках соединения пакет (по умолчанию используется системный вызов fwrite); параметрами функции являются указатель на буфер данных открытого пакета, размер этого буфера и указатель на дополнительные пользовательские данные, заданные предварительном вызовом функции с опцией CURLOPT_WRITEDATA
CURLOPT_READFUNCTION	устанавливает указанную функцию в качестве обработчика процедуры чтения данных из полученного в рамках соединения пакета (по умолчанию используется системный вызов fread); параметрами функции являются указатель на буфер данных открытого пакета, размер этого буфера и указатель на дополнительные пользовательские данные, заданные предварительном вызовом функции с опцией CURLOPT_READDATA
CURLOPT_SEEKFUNCTION	устанавливает указанную функцию в качестве обработчика процедуры смещения по полученным в рамках соединения данным пакета, если необходимо выполнять rewind данных после отправки (по умолчанию смещение не используется); параметрами функции являются указатель на пользовательские данные, заданные предварительным вызовом функции с опцией CURLOPT_SEEKDATA, значение смещения и флаг определения начальной точки отсчёта
CURLOPT_IOCTLFUNCTION	устанавливает указанную функцию в качестве обработчика процедуры ввода/вывода обрабатываемых в рамках соединения данных пакета, если необходимо выполнять rewind после отправки и не была задана функция CURLOPT_SEEKFUNCTION (по умолчанию не используется); параметрами функции являются дескриптор сессии работы библиотеки SessionHandle, код команды и указатель на пользовательские данные, заданные предварительном вызовом функции с опцией CURLOPT_IOCTLDATA
CURLOPT_CONV_TO_NETWORK_FUNCTION	устанавливает указанную функцию в качестве обработчика процедуры кодирования отправляемых по сети данных (согласно архитектуре libcurl)
CURLOPT_CONV_FROM_NETWORK_FUNCTION	устанавливает указанную функцию в качестве обработчика процедуры раскодировки полученных по сети данных (согласно архитектуре libcurl)

CURLOPT_CONV_FROM_UTF8_FUNCTION	устанавливает указанную функцию в качестве обработчика процедуры приведения кодировки сообщения из UTF-8 в кодировку среды функционирования клиента (согласно архитектуре libcurl)
CURLOPT_SSLKEY, CURLOPT_PROXY_SSLKEY	задают название закрытого ключа пользователя, используемого при установке TLS-соединения (согласно архитектуре libcurl)
CURLOPT_SSLKEYTYPE, CURLOPT_PROXY_SSLKEYTYPE	задают формат PEM/DER файла закрытого ключа пользователя, используемого при установке TLS-соединения (согласно архитектуре libcurl)
CURLOPT_SSLENGINE	устанавливает указанный криптодрайвер в качестве допустимого при установке TLS-соединения с использованием реализации OpenSSL (согласно архитектуре libcurl)
CURLOPT_SSLENGINE_DEFAULT	устанавливает указанный по порядковому номеру криптодрайвер реализации OpenSSL в качестве криптодрайвера по умолчанию (согласно архитектуре libcurl)
CURLOPT_KRBLEVEL	задаёт уровень безопасности протокола Kerberos при установке FTP-соединения (согласно архитектуре libcurl)
CURLOPT_SSL_VERIFYSTATUS	задаёт необходимость проверки сертификата сервера на основе расширения TLS «Certificate Status Request» (с помощью OCSP-запроса)
CURLOPT_SSL_CTX_FUNCTION	задаёт указатель на функцию-обработчик, которая будет вызываться перед установкой TLS-соединения после применения остальных настроек (согласно архитектуре libcurl); параметрами функции являются дескриптор сессии работы библиотеки Curl_easy, указатель на контекст TLS-соединения и указатель на пользовательские данные, заданные предварительном вызовом функции с опцией CURLOPT_SSL_CTX_DATA
CURLOPT_SSL_FALSESTART	задаёт применение режима «фальстарта» при установке TLS-соединения (режима отправки прикладных пакетов до проверки последнего Handshake-сообщения сервера)

Следующие опции **запрещено** использовать при установке TLS-соединения с поддержкой ГОСТ в случае использования функций интерфейса при разработке систем на основе СКЗИ без дополнительных тематических исследований:

CURLOPT_VERBOSE	устанавливает флаг, определяющий использование отладочной печати при использовании библиотеки
CURLOPT_FTP_SSL_CCC	настройки возможности отправки FTP-серверу команды CCC (команды перехода из TLS-канала в открытый для отправки служебных сведений)
CURLOPT_OPENSOCKETFUNCTION	устанавливает указанную функцию в качестве специального обработчика, вызываемого взамен функции открытия сокета соединения с сервером в целях модификации свойств соединения; параметрами функции являются тип устанавливаемого соединения, указатель на структуру адреса curl_sockaddr и указатель на пользовательские данные, заданные предварительном вызовом функции с опцией CURLOPT_OPENSOCKETDATA; некорректная перенастройка параметров сокета может привести в том числе к смене выбранного протокола (например, к замене HTTPS на HTTP)
CURLOPT_KEYPASSWD, CURLOPT_PROXY_KEYPASSWD	задают пароль, ассоциированный с сертификатом, который будет использован для установки TLS-соединения
CURLOPT_SSL_VERIFYPEER, CURLOPT_PROXY_SSL_VERIFYPEER	задают необходимость проверки сертификата сервера при установке TLS-соединения и отказа от соединения в случае ошибки проверки (по умолчанию проверка выполняется)
CURLOPT_SSL_VERIFYHOST, CURLOPT_PROXY_SSL_VERIFYHOST	задают необходимость при установке TLS-соединения проверки соответствия имени сервера тому имени, который задан в высланном им сертификате (по умолчанию проверка выполняется)
CURLOPT_SSL_OPTIONS, CURLOPT_PROXY_SSL_OPTIONS	задают отсутствие проверки сертификата сервера по списку отозванных сертификатов при установке соединения (с флагом CURLSSLOPT_NO_REVOKE) и отсутствие защитных мер против атак BEAST (с флагом CURLSSLOPT_ALLOW_BEAST); относится только к реализациям на основе DarwinSSL, NSS и OpenSSL
CURLOPT_PROXY, CURLOPT_PROXYTYPE, CURLOPT_HTTPPROXYTUNNEL, CURLOPT_PRE_PROXY	задают необходимость использования прокси-сервера, указанного типа протокола для соединения с прокси-сервером, туннелирования трафика через прокси-сервер и использование предпрокси-сервера (промежуточного сервера для соединения с прокси-сервером) соответственно
CURLOPT_PROXY_TRANSFER_MODE	задаёт режим передачи данных в случае связи с FTP-сервером через прокси-сервер

Следующие опции **не поддерживаются** в интерфейсе:

- CURLOPT_SSH_AUTH_TYPES
- CURLOPT_SSH_PUBLIC_KEYFILE
- CURLOPT_SSH_PRIVATE_KEYFILE
- CURLOPT_SSH_HOST_PUBLIC_KEY_MD5
- CURLOPT_SSH_KNOWNHOSTS
- CURLOPT_SSH_KEYFUNCTION
- CURLOPT_SSH_KEYDATA
- CURLOPT_SSH_COMPRESSION
- CURLOPT_TLSAUTH_USERNAME
- CURLOPT_PROXY_TLSAUTH_USERNAME
- CURLOPT_TLSAUTH_PASSWORD
- CURLOPT_PROXY_TLSAUTH_PASSWORD
- CURLOPT_TLSAUTH_TYPE
- CURLOPT_PROXY_TLSAUTH_TYPE
- CURLOPT_DNS_SERVERS
- CURLOPT_DNS_INTERFACE
- CURLOPT_DNS_LOCAL_IP4
- CURLOPT_DNS_LOCAL_IP6
- CURLOPT_SSL_ENABLE_NPN
- CURLOPT_SSL_ENABLE_ALPN
- CURLOPT_ALTSVC
- CURLOPT_ALTSVC_CTRL

Другие опции, не указанные в перечнях выше, поддерживаются аналогично libcurl (см. описание функции curl_easy_setopt).

Возвращаемое значение: код ошибки

curl_easy_getinfo

Функция получения сведений о соединении.

Входные параметры:

- curl (дескриптор, должен быть ранее получен вызовом `curl_easy_init`)

- info

В зависимости от указанных параметров можно получить следующие сведения о соединении:
- временные данные (время работы, обработки, доставки и т.п.);
- использовавшиеся в последнем соединении параметры соединения;
- перечень доступных криптодрайверов OpenSSL (по флагу CURLINFO_SSL_ENGINES);
- указатель на private-данные, связанные с указанным дескриптором (по флагу CURLINFO_PRIVATE).

Возвращаемое значение: код ошибки

curl_easy_perform

Функция синхронизируемой передачи данных по заданному дескриптору. Передаваемые данные и прочие настройки задаются в `curl_easy_setopt`. Перед вызовом данной функции необходимо настроить TLS-соединение с помощью `curl_easy_setopt` (подробнее см. Приложение 2 документа ЖТЯИ.00101-03 95 01. Правила пользования).

Входные параметры: easy_handle (должен быть ранее получен вызовом `curl_easy_init`)

Возвращаемое значение: код ошибки

curl_easy_send

Функция отправляет данные типа raw-data по установленному ранее каналу. Для вызова функции необходимо предварительно задать флаг CURLOPT_CONNECT_ONLY вызовом `curl_easy_setopt` и установить соединение вызовом `curl_easy_perform`.

Входные параметры:

- curl (дескриптор, должен быть ранее получен вызовом `curl_easy_init`)

- buffer (указатель на буфер передаваемых данных)

- buflen (размер буфера)

- n (указатель на количество отправленных байтов)

Возвращаемое значение: код ошибки

curl_easy_recv

Функция принимает данные типа raw-data по установленному ранее каналу. Для вызова функции необходимо предварительно задать флаг CURLOPT_CONNECT_ONLY вызовом curl_easy_setopt и установить соединение вызовом curl_easy_perform.

Входные параметры:

- curl (дескриптор, должен быть ранее получен вызовом curl_easy_init)

- buffer (указатель на буфер принимаемых данных)

- buflen (максимальный размер буфера)

- n (указатель на количество принятых байтов)

Возвращаемое значение: код ошибки

Вспомогательные функции обработки строк и данных

curl_easy_escape

Функция перевода заданной строки в URL-кодировку.

Входные параметры:

- curl (дескриптор)

- string (заданная строка символов char)

- length (длина строки, может быть указана нулевой, тогда фактическая длина вычисляется внутри функции)

Возвращаемое значение: указатель на URL-кодированную строку или NULL

curl_easy_unescape
Функция перевода заданной URL-кодированной строки в строку символов char.

Входные параметры:

- curl (дескриптор)

- url (URL-кодированная строка)

- inlength (длина строки, может быть указана нулевой, тогда фактическая длина вычисляется внутри функции)

- outlength (указатель на значение длины возвращаемой строки char-символов)

Возвращаемое значение: указатель на строку символов char или NULL

curl_formadd

Функция добавления новой секции в состав данных типа multipart/formdata, отправляемых по HTTP POST-запросу.

Входные параметры:

- firstitem (указатель на первую секцию данных POST-запроса)
- lastitem (указатель на последнюю секцию данных POST-запроса)
- options (опции, определяют тип добавляемой секции, её название, содержимое и характеристики)

Следующие опции поддерживаются в интерфейсе (в соответствии с документацией libcurl на curl_formadd):

- CURLFORM_COPYNAME (0x1)
- CURLFORM_PTRNAME (0x2)
- CURLFORM_NAMELENGTH (0x3)
- CURLFORM_COPYCONTENTS (0x4)
- CURLFORM_PTRCONTENTS (0x5)
- CURLFORM_CONTENTSLENGTH (0x6)
- CURLFORM_FILECONTENT (0x7)
- CURLFORM_ARRAY (0x8)
- CURLFORM_FILE (0xA)
- CURLFORM_BUFFER (0xB)
- CURLFORM_BUFFERPTR (0xC)
- CURLFORM_BUFFERLENGTH (0xD)
- CURLFORM_CONTENTTYPE (0xE)
- CURLFORM_CONTENTHEADER (0xF)
- CURLFORM_FILENAME (0x10)
- CURLFORM_END (0x11)
- CURLFORM_STREAM (0x13)
- CURLFORM_CONTENTLEN (0x14).

Возвращаемое значение: код ошибки (CURL_FORMADD_* constant)

curl_formfree

Функция освобождения ресурсов, выделенных для данных типа multipart/formdata, составленных ранее набором вызовов `curl_formadd`.

Входные параметры: form (указатель на первую секцию данных POST-запроса, занимаемую память которого необходимо освободить)

curl_slist_append

Функция добавляет строку в список строк структуры `curl_slist`.

Входные параметры:

- list (указатель на структуру curl_slist, в которую необходимо добавлять строку; если равно NULL, функция сформирует новый объект curl_slist и вернёт указатель на этот объект)

- string (строка, добавляемая в список)

Возвращаемое значение: указатель на структуру curl_slist или NULL

curl_slist_free_all

Функция освобождает все ресурсы, занятые структурой curl_slist.

Входные параметры: list (указатель на структуру curl_slist, ресурсы которой необходимо освободить; если равен NULL, функция не выполняет никаких действий)

Приложение 2
Пример установки TLS-соединения с использованием интерфейса cpcurl

```c
#include "curl.h"

struct MemoryStruct {
    char *memory;
    size_t size;
};

static size_t WriteMemoryCallback(void *contents, size_t size, size_t nmemb, void *userp)
{
    size_t realsize = size *nmemb;
    struct MemoryStruct *mem = (struct MemoryStruct *)userp;

    char *ptr = (char *)realloc(mem->memory, mem->size + realsize + 1);
    if(ptr == NULL) {
        /* out of memory! */
        printf("not enough memory (realloc returned NULL)\n");
        return 0;
    }

    mem->memory = ptr;
    memcpy((void*)&(mem->memory[mem->size]), contents, realsize);
    mem->size += realsize;
    mem->memory[mem->size] = 0;

    return realsize;
}

char *getUrl(const char *url) {
    struct MemoryStruct chunk;
    chunk.memory = (char *)malloc(1);  /* will be grown as needed by the realloc above */
    chunk.size = 0;    /* no data at this point */
    CURL *curlHandle = NULL;
    CURLcode errCode = CURLE_OK;
    char *result = NULL;

    curlHandle = curl_easy_init();
    if (!curlHandle){
        free(chunk.memory);
        goto err;
    }

    curl_easy_setopt(curlHandle, CURLOPT_URL, url);

    /* send all data to this function  */
    curl_easy_setopt(curlHandle, CURLOPT_WRITEFUNCTION, WriteMemoryCallback);

    /* we pass our 'chunk' struct to the callback function */
    curl_easy_setopt(curlHandle, CURLOPT_WRITEDATA, (void *)&chunk);

    /* required for GOST white list */
    curl_easy_setopt(curlHandle, CURLOPT_STRICT_GOST, 1L);
    curl_easy_setopt(curlHandle, CURLOPT_NOPROXY, "*");
    curl_easy_setopt(curlHandle, CURLOPT_REDIR_PROTOCOLS, CURLPROTO_HTTPS);
    curl_easy_setopt(curlHandle, CURLOPT_FTPSSLAUTH, CURLFTPAUTH_TLS);

```

```
56      errCode = curl_easy_perform(curlHandle);
57      if (errCode != CURLE_OK) {
58          free(chunk.memory);
59          goto err;
60      }
61
62      result = chunk.memory;
63
64  err:
65      curl_easy_cleanup(curlHandle);
66      return result;
67  }
68
```