Работа с ГОСТ ключами в openssl

НАЙС.ОС: ГОСТ-криптография в OpenSSL 3.5.X (из коробки)

Настоящий документ устанавливает порядок проверки, генерации ключей, выполнения операций шифрования/подписи и контроля результатов при использовании ГОСТ-алгоритмов в составе OpenSSL в НАЙС.ОС. В НАЙС.ОС ГОСТ-криптография доступна сразу после установки (дополнительные пакеты не требуются) и реализуется через проект gost-engine/engine.

Фиксация версии среды
На момент подготовки документа используется OpenSSL: OpenSSL 3.5.4 30 Sep 2025 (ветка обновляется).
Замечание по архитектуре OpenSSL 3.x
Механизм ENGINE в OpenSSL 3.0+ объявлен устаревшим (deprecated) и стратегически заменён на providers, однако при сборке OpenSSL с поддержкой engines они продолжают работать, но с ограничениями применимости.

1 Термины, сокращения и нормативные ссылки

1.1 Сокращения

- УКМ — user key material (параметр VKO, передаваемый стороне-получателю).
- VKO — механизм выработки общего секрета для ГОСТ Р 34.10 (см. RFC 4357).
- CMS — Cryptographic Message Syntax (PKCS#7).
- PKI — инфраструктура открытых ключей.

1.2 Нормативные ссылки (технические)

- RFC 4357 — описывает VKO и ряд практических профилей применения ГОСТ.
- RFC 4490 / RFC 4491 — профили применения ГОСТ для S/MIME и PKI

(упоминаются в документации движка).

2 Состав поддерживаемых ГОСТ-алгоритмов

Реализация ГОСТ в движке OpenSSL покрывает как минимум следующие классы алгоритмов (перечень приведён по технической документации движка):

- ГОСТ Р 34.10-2001 и ГОСТ Р 34.10-2012 — ЭЦП и обмен ключами (VKO).
- ГОСТ Р 34.11-94 и ГОСТ Р 34.11-2012 — функции хэширования.
- ГОСТ 28147-89 — симметричный шифр (CBC/CFB/CNT) и MAC.
- ГОСТ Р 34.13-2015 — «Кузнечик» (Grasshopper) как симметричный шифр.

Имена алгоритмов в OpenSSL
Для генерации ключей используются имена gost2001, gost2012_256, gost2012_512 и параметры paramset, перечисленные в разделе 4.

3 Контроль готовности ГОСТ-криптографии в системе

3.1 Проверка версии OpenSSL и директории конфигурации

```
openssl version -a
openssl version -d
```

3.2 Проверка доступности ENGINE (диагностическая команда)

Команда openssl engine является устаревшей в OpenSSL 3.x, но применяется для диагностики наличия загруженных engines.

```
# Показать список engines и их возможности
openssl engine -c

# Проверить, что gost доступен и работоспособен (если имя engine = gost)
openssl engine -t -c gost
```

3.3 Инвентаризация алгоритмов через OpenSSL CLI

```
# Хэши
openssl list -digest-algorithms | grep -i gost || true

# Шифры
openssl list -cipher-algorithms | grep -i -E 'gost|kuz|magma' || true
```

Типовой признак некорректной загрузки
Если команды из раздела 4–7 возвращают ошибки вида unknown algorithm / unsupported, необходимо диагностировать конфигурацию OpenSSL (OPENSSLDIR, openssl.cnf) и механизм загрузки движка/провайдера. В OpenSSL 3.x engines считаются «legacy» и могут быть отключены сборкой.

4 Генерация ГОСТ-ключей (34.10-2001 / 34.10-2012)

4.1 Генерация закрытого ключа через genpkey

В качестве параметра paramset допускается использовать буквенные обозначения или числовой OID. Перечни поддерживаемых paramset по алгоритмам приведены в документации движка.

```
# ГОСТ Р 34.10-2001 (пример)
openssl genpkey -algorithm gost2001 -pkeyopt paramset:A -out gost2001_seckey.pem

# ГОСТ Р 34.10-2012 (256 бит) — пример
openssl genpkey -algorithm gost2012_256 -pkeyopt paramset:TCA -out gost2012_256_seckey.pem

# ГОСТ Р 34.10-2012 (512 бит) — пример
openssl genpkey -algorithm gost2012_512 -pkeyopt paramset:A -out gost2012_512_seckey.pem
```

Правило выбора paramset
Paramset, начинающиеся с X, предназначены для ключей обмена (key exchange), без X — для ключей подписи; значение 0 является тестовым и применяется только для тестирования.

4.2 Извлечение открытого ключа

```
openssl pkey -in gost2012_256_seckey.pem -pubout -out gost2012_256_pubkey.pem
```

```
openssl pkey -in gost2012_512_seckey.pem -pubout -out gost2012_512_pubkey.pem
```

4.3 Контроль параметров ключа

```
openssl pkey -in gost2012_256_seckey.pem -text -noout
openssl pkey -pubin -in gost2012_256_pubkey.pem -text -noout
```

Требования к защите закрытых ключей
Закрытые ключи подлежат хранению с правами доступа не шире 0600 и резервированию в защищённом контуре. При эксплуатации в продуктиве рекомендуется применение аппаратных носителей (PKCS#11/токены) согласно политике организации.

5 PKI: CSR и сертификаты X.509 с ГОСТ-ключами

5.1 Формирование CSR (запроса на сертификат)

После получения закрытого ключа дальнейшие операции формирования CSR выполняются штатно, без алгоритм-специфичных приёмов.

```
# Пример: CSR для ГОСТ 2012/256
openssl req -new \
 -key gost2012_256_seckey.pem \
 -out gost2012_256.csr \
 -subj "/C=RU/ST=77/L=Moscow/O=NiceSOFT/OU=PKI/CN=niceos-gost.example"
```

5.2 Самоподписанный сертификат (для стенда/теста)

```
# ВНИМАНИЕ: для продуктивной PKI используйте полноценный CA-процесс и профили расширений.
openssl req -x509 -new \
 -key gost2012_256_seckey.pem \
 -out gost2012_256.crt \
 -days 365 \
 -subj "/C=RU/ST=77/L=Moscow/O=NiceSOFT/OU=PKI/CN=niceos-gost.example"
```

5.3 Просмотр и верификация сертификата

```
openssl x509 -in gost2012_256.crt -noout -text
openssl verify -CAfile gost2012_256.crt gost2012_256.crt
```

Проверка алгоритмов подписи и OID
В выводе openssl x509 -text контролируется поле Signature Algorithm и идентификаторы алгоритмов (OID). Для ГОСТ-сертификатов ожидается семейство OID 1.2.643.* (в зависимости от профиля и параметров УЦ).

6 Хэширование, подпись и проверка подписи

6.1 Хэширование по ГОСТ Р 34.11-94

Пример вычисления хэша md_gost94 приведён в документации движка. Отдельно указано, что стандарт трактует значение как little-endian число, а OpenSSL выводит «сырой» hex-дамп, который при необходимости следует разворачивать по байтам.

```
openssl dgst -md_gost94 data.bin
```

6.2 ГОСТ-хэши 34.11-2012 (Стрибог 256/512)

```
openssl dgst -md_gost12_256 data.bin
openssl dgst -md_gost12_512 data.bin
```

6.3 Создание электронной подписи (GOST 34.10-2012) и проверка

```
# Подпись (пример для 2012/256)
openssl dgst -md_gost12_256 -sign gost2012_256_seckey.pem -out data.sig data.bin

# Проверка подписи
openssl dgst -md_gost12_256 -verify gost2012_256_pubkey.pem -signature data.sig data.bin
```

Требование к контролю входных данных
Подписываемые/проверяемые данные должны рассматриваться как бинарные. Не

допускается незадокументированная перекодировка (CRLF/LF, UTF-8/ANSI), иначе контроль подписи становится недостоверным.

7 MAC и имитовставка (ГОСТ 28147-89 MAC, HMAC на ГОСТ-хэшах)

7.1 HMAC на md_gost94

```
# Ключ — 32 байта (пример: строка для демонстрации; в продуктиве используйте двоичный ключ)
openssl dgst -md_gost94 -hmac "0123456789abcdef0123456789abcdef" data.bin
```

7.2 ГОСТ 28147 MAC (gost-mac)

Пример из документации движка: расчёт MAC через openssl dgst -mac gost-mac с параметром ключа.

```
# Вариант: ключ как строка
openssl dgst -mac gost-mac -macopt key:"0123456789abcdef0123456789abcdef" data.bin

# Вариант: ключ как hex (64 hex-символа = 32 байта)
openssl dgst -mac gost-mac -macopt hexkey:001122...ffeedd data.bin
```

Управление длиной имитовставки
В документации движка указана возможность изменять размер MAC (1..8 байт) через -sigopt size:N (особенности разделения опций init/final для dgst также указаны там же).

8 Симметричное шифрование

8.1 ГОСТ 28147-89 (gost89) через openssl enc

В документации движка приведены примеры шифрования в режимах CFB/CNT/CBC.

```
# CFB
openssl enc -gost89 -e -in plain.bin -out enc.bin -k "passphrase"

# CNT
openssl enc -gost89-cnt -e -in plain.bin -out enc_cnt.bin -k "passphrase"
```

```
# CBC
openssl enc -gost89-cbc -e -in plain.bin -out enc_cbc.bin -k "passphrase"

# Расшифрование (пример CNT)
openssl enc -gost89-cnt -d -in enc_cnt.bin -out plain.dec.bin -k "passphrase"
```

Требование по KDF
Команда openssl enc допускает устаревшие схемы вывода ключа из пароля. В продуктивных сценариях шифрования файлов рекомендуется применять режимы с явным ключом/IV либо контейнерные форматы (CMS), а также обеспечить достаточное число итераций PBKDF2 (если применимо в вашей политике).

8.2 «Кузнечик» и «Магма» (при использовании gost-provider)

В проекте gost-engine описан provider-режим для OpenSSL 3.x и приведены имена шифров (в т.ч. CTR-ACPKM и варианты с OMAC).

```
# Пример: «Кузнечик» CTR-ACPKM (имя алгоритма зависит от включённого провайдера)
openssl enc -kuznyechik-ctr-acpkm -e -in plain.bin -out enc_kuz.bin -K <HEXKEY> -iv <HEXIV>

# Пример: «Магма» CTR-ACPKM
openssl enc -magma-ctr-acpkm -e -in plain.bin -out enc_magma.bin -K <HEXKEY> -iv <HEXIV>
```

Контроль доступности «Кузнечика/Магмы»
При корректной загрузке провайдера имена алгоритмов должны присутствовать в выводе: openssl list -cipher-algorithms.

9 CMS/S/MIME: подпись и шифрование сообщений

9.1 S/MIME-шифрование: явный выбор симметричного алгоритма

В документации движка указано: при шифровании почты GOST-алгоритм симметричного шифрования должен задаваться явно (например, -gost89), так как OpenSSL не выводит алгоритм шифрования из ключей обмена автоматически.

```
# Пример S/MIME (иллюстративно; адаптируйте параметры под вашу PKI)
# 1) Подпись
```

```
openssl smime -sign -binary \
 -in mail.txt -out mail.signed.pem \
 -signer gost2012_256.crt -inkey gost2012_256_seckey.pem \
 -outform PEM

# 2) Шифрование (симметричный алгоритм задаётся явно)
openssl smime -encrypt -binary \
 -in mail.signed.pem -out mail.enc.pem \
 -outform PEM -gost89 \
 recipient.crt
```

Требование к совместимости
CMS/S/MIME совместимость определяется профилем УЦ, набором поддерживаемых OID у получателя и конкретной реализацией криптопровайдера. Перед продуктивным внедрением требуется межсистемное тестирование с целевыми продуктами (почтовые клиенты, шлюзы, СКЗИ).

10 TLS с ГОСТ: контроль поддержки и практический запуск

Документация движка указывает наличие поддержки TLS-наборов шифров ГОСТ и перечисляет 4 ciphersuite, используемых в соответствующем IETF-draft, а также возможность одновременной поддержки ГОСТ и RSA/DSA/EC ключей при наличии нескольких пар ключ/сертификат на сервере.

10.1 Инвентаризация наборов шифров

```
openssl ciphers -v | grep -i gost || true
```

10.2 Тестовый запуск TLS-сервера (стенд)

```
# Сервер
openssl s_server -accept 8443 -cert gost2012_256.crt -key gost2012_256_seckey.pem -www

# Клиент
openssl s_client -connect 127.0.0.1:8443 -servername niceos-gost.example
```

Ограничение применимости
Поддержка ГОСТ-TLS на стороне клиентов/прокси/балансировщиков неоднородна.

Для продуктивной эксплуатации требуется проверка поддерживаемых cipher suites у всех узлов цепочки (клиент прокси/SSL-терминация сервер).

11 PKCS#12: экспорт ключа/сертификата с ГОСТ-шифрованием контейнера

Для совместимости с отдельными CSP в документации движка приведён пример экспорта PKCS#12 с использованием ГОСТ-шифрования и ГОСТ-хэша для derivation MAC/ключей.

```
# Экспорт PKCS#12 (пример; имя команды: pkcs12)
openssl pkcs12 -export \
 -inkey gost2012_256_seckey.pem \
 -in gost2012_256.crt \
 -out gost2012_256.p12 \
 -keypbe gost89 -certpbe gost89 -macalg md_gost94
```

Контроль результатов
После формирования .p12 требуется проверить импорт в целевом ПО (СКЗИ/токен/хранилище), а также контролировать алгоритмы PBE и MAC в структуре контейнера.

12 Производительность: контроль скорости симметричных шифров

Для шифров, предоставляемых через EVP интерфейс, документация движка указывает применение openssl speed -evp.

```
openssl speed -evp gost89
openssl speed -evp gost89-cnt
openssl speed -evp gost89-cbc
```

13 Минимальный самотест (проверка «под ключ»)

```
set -euo pipefail

WORKDIR="$(mktemp -d)"
cd "$WORKDIR"
```

```
# 1) Версии
openssl version -a

# 2) Тестовые данные
printf 'NiceOS GOST self-test\n' > data.bin

# 3) Ключ + публичный ключ (GOST 2012/256)
openssl genpkey -algorithm gost2012_256 -pkeyopt paramset:TCA -out seckey.pem
openssl pkey -in seckey.pem -pubout -out pubkey.pem

# 4) Хэш + подпись + проверка
openssl dgst -md_gost12_256 data.bin
openssl dgst -md_gost12_256 -sign seckey.pem -out data.sig data.bin
openssl dgst -md_gost12_256 -verify pubkey.pem -signature data.sig data.bin

# 5) Симметричное шифрование (GOST 28147 CNT)
openssl enc -gost89-cnt -e -in data.bin -out enc.bin -k "passphrase"
openssl enc -gost89-cnt -d -in enc.bin -out dec.bin -k "passphrase"
cmp -s data.bin dec.bin

echo "OK: GOST self-test passed in $WORKDIR"
```

Критерии успешности самотеста
Требуется одновременно: (1) успешная генерация ключа, (2) успешная проверка подписи, (3) совпадение исходного и расшифрованного файла (cmp без различий).

Приложение А (справочное): paramset и примеры из документации движка

Перечень поддерживаемых paramset для генерации ключей указан в документации движка:

- gost2001: 0,A,B,C,XA,XB
- gost2012_256: 0,A,B,C,XA,XB,TCA,TCB,TCC,TCD
- gost2012_512: A,B,C

Требование к продуктивным профилям
Использование тестовых параметров (paramset=0) в продуктивной эксплуатации не допускается.