Referência Rápida de OpenSSL

Certificados, chaves, criptografia e diagnóstico

Certificados

Visualizar Detalhes do Certificado

```
openssl x509 -in cert.pem -text -noout
openssl x509 -in cert.pem -subject -noout
openssl x509 -in cert.pem -dates -noout
openssl x509 -in cert.pem -issuer -noout
```

Converter Formatos

```
# PEM to DER
openssl x509 -in cert.pem -outform DER \
    -out cert.der
# DER to PEM
openssl x509 -in cert.der -inform DER \
    -out cert.pem
```

Formatos Comuns

PEM	Codificado em Base64, **-----BEGIN CERTIFICATE-----**
DER	Formato binário, compacto
PFX / P12	Bundle PKCS#12 (cert + chave + cadeia)
CRT / CER	Arquivo de certificado (geralmente PEM ou DER)

Geração de Chaves

Chaves RSA

```
openssl genrsa -out key.pem 4096
openssl rsa -in key.pem -pubout \
    -out pubkey.pem
openssl rsa -in key.pem -text -noout
```

Chaves EC

```
openssl ecparam -genkey -name prime256v1 \
    -out ec_key.pem
openssl ec -in ec_key.pem -pubout \
    -out ec_pub.pem
```

Chaves Ed25519

```
openssl genpkey -algorithm Ed25519 \
    -out ed25519_key.pem
openssl pkey -in ed25519_key.pem -pubout \
    -out ed25519_pub.pem
```

Comparação de Algoritmos de Chave

RSA 2048/4096	Amplamente suportado, chaves maiores
ECDSA (P-256)	Chaves menores, mais rápido, TLS moderno
Ed25519	Mais rápido, menor tamanho, não suportado em todos os sistemas

CSR

Gerar CSR

```
openssl req -new -key key.pem \
    -out request.csr
# Non-interactive
openssl req -new -key key.pem -out req.csr \
    -subj "/CN=example.com/O=MyOrg/C=US"
```

Gerar Chave + CSR Juntos

```
openssl req -new -newkey rsa:4096 \
    -nodes -keyout key.pem -out req.csr \
    -subj "/CN=example.com"
```

Inspecionar CSR

```
openssl req -in request.csr -text -noout
openssl req -in request.csr -verify -noout
```

Campos Comuns do CSR

CN	Nome comum (domínio ou hostname)
O	Nome da organização
OU	Unidade organizacional
C	País (código de 2 letras)
ST	Estado ou província
L	Localidade / cidade

Autoassinado

Certificado Autoassinado Rápido

```
openssl req -x509 -newkey rsa:4096 -nodes \
    -keyout key.pem -out cert.pem -days 365 \
    -subj "/CN=localhost"
```

Com SAN (Subject Alternative Name)

```
openssl req -x509 -newkey rsa:4096 -nodes \
    -keyout key.pem -out cert.pem -days 365 \
    -subj "/CN=myapp.local" \
    -addext "subjectAltName=\
DNS:myapp.local,DNS:*.myapp.local,IP:127.0.0.1"
```

A Partir de Chave Existente

```
openssl req -x509 -key key.pem \
    -out cert.pem -days 365 \
    -subj "/CN=example.com"
```

Verificação

Verificar Certificado

```
openssl verify -CAfile ca.pem cert.pem
openssl verify -CAfile ca.pem \
    -untrusted intermediate.pem cert.pem
```

Verificar Correspondência Chave/Cert

```
# Modulus must match for key and cert
openssl x509 -in cert.pem -modulus -noout
openssl rsa -in key.pem -modulus -noout
openssl req -in req.csr -modulus -noout
```

Verificar Expiração

```
openssl x509 -in cert.pem -checkend 86400
# Returns 0 if valid for 86400s (24h)
openssl x509 -in cert.pem -enddate -noout
```

Certificado de Servidor Remoto

```
openssl s_client -connect example.com:443 \
    < /dev/null 2>/dev/null \
    | openssl x509 -text -noout
```

Criptografia

Criptografia Simétrica

```
openssl enc -aes-256-cbc -salt -pbkdf2 \
    -in plain.txt -out encrypted.bin
openssl enc -aes-256-cbc -d -pbkdf2 \
    -in encrypted.bin -out plain.txt
```

Criptografia Assimétrica

```
# Encrypt with public key
openssl pkeyutl -encrypt \
    -pubin -inkey pub.pem \
    -in secret.txt -out secret.enc
# Decrypt with private key
openssl pkeyutl -decrypt \
    -inkey key.pem \
    -in secret.enc -out secret.txt
```

Cifras Comuns

aes-256-cbc	AES 256-bit, modo CBC (padrão comum)
aes-256-gcm	AES 256-bit, modo GCM (autenticado)
chacha20-poly1305	Cifra de fluxo moderna (rápida em ARM)

Listar todas: `openssl enc -list`

Hashing

Hashes de Arquivo

```
openssl dgst -sha256 file.txt
openssl dgst -sha512 file.txt
openssl dgst -md5 file.txt  # legacy only
```

HMAC

```
openssl dgst -sha256 -hmac "secret" file.txt
echo -n "message" | openssl dgst \
    -sha256 -hmac "mykey"
```

Algoritmos de Hash

SHA-256	Escolha padrão para verificação de integridade
SHA-384 / SHA-512	Variantes SHA-2 mais fortes
SHA3-256	Padrão mais recente (baseado em Keccak)
MD5	Quebrado, apenas legado — não usar para segurança
BLAKE2	Alternativa rápida e segura (se suportada)

S/MIME

Assinar E-mail

```
openssl smime -sign -in msg.txt \
    -signer cert.pem -inkey key.pem \
    -out signed.msg
```

Verificar E-mail Assinado

```
openssl smime -verify -in signed.msg \
    -CAfile ca.pem -out original.txt
```

Criptografar / Descriptografar E-mail

```
# Encrypt for recipient
openssl smime -encrypt -aes256 \
    -in msg.txt -out encrypted.msg \
    recipient_cert.pem
# Decrypt
openssl smime -decrypt -in encrypted.msg \
    -recip cert.pem -inkey key.pem
```

Diagnóstico

Testar Conexão TLS

```
openssl s_client -connect host:443
openssl s_client -connect host:443 \
    -servername example.com  # SNI
openssl s_client -connect host:443 \
    -tls1_3  # force TLS 1.3
```

Referência Rápida de OpenSSL

Exibir Cadeia de Certificados

```
openssl s_client -connect host:443 \
    -showcerts < /dev/null
```

Verificar Cifras TLS

```
openssl ciphers -v 'HIGH:!aNULL'
openssl s_client -connect host:443 \
    -cipher 'ECDHE-RSA-AES256-GCM-SHA384'
```

Operações PKCS#12

```
# Create PFX bundle
openssl pkcs12 -export -out bundle.pfx \
    -inkey key.pem -in cert.pem -certfile ca.pem
# Extract from PFX
openssl pkcs12 -in bundle.pfx -nodes \
    -out all.pem
```

Padrões Comuns

Gerar Aleatório Seguro

```
openssl rand -hex 32    # 32 random bytes, hex
openssl rand -base64 24 # 24 random bytes, b64
```

Codificar / Decodificar Base64

```
openssl base64 -in file.bin -out file.b64
openssl base64 -d -in file.b64 -out file.bin
```

Hash de Senha

```
openssl passwd -6 -salt xyz "password"
# -6 = SHA-512, -5 = SHA-256, -1 = MD5
```

Rápido: Chave + Cert + Verificar

```
openssl req -x509 -newkey rsa:4096 -nodes \
    -keyout k.pem -out c.pem -days 365 \
    -subj "/CN=test"
openssl x509 -in c.pem -text -noout
```