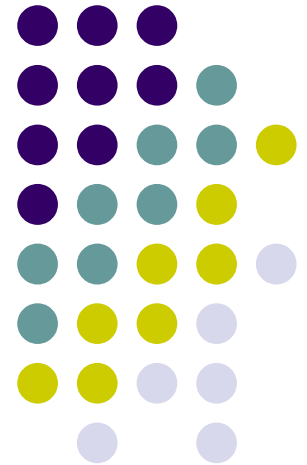


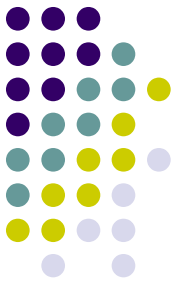
Crypto libraries

OpenSSL II (cont.)

Milan Brož
xbroz@fi.muni.cz

PV181, FI MUNI, Brno

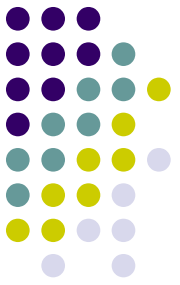




Today's exercise

- Continue with OpenSSL on Linux
- OpenSSL I/O abstraction (BIO)
- Trivial TLS client

Example 7: OpenSSL BIO (I/O abstraction)

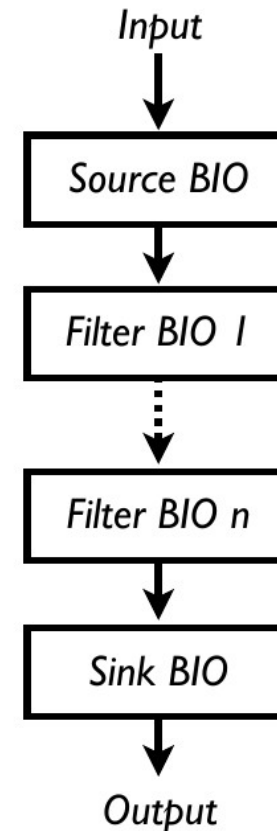


Source/sink BIOs:

BIO_s_mem() - memory I/O
BIO_s_file() - file I/O
BIO_s_fd() - file descriptor IO
BIO_s_socket() - sockets
BIO_s_accept()
BIO_s_connect()
BIO_s_null() - discard (like /dev/null)

Filters

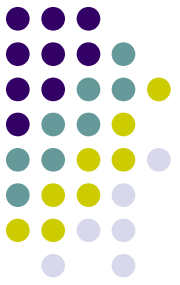
BIO_f_base64() - Base64 encoding
BIO_f_buffer() - buffering I/O
BIO_f_cipher() - encryption/decryption
BIO_f_md() - message digest
BIO_f_ssl() - SSL support for BIO



Example 7: the same encryption as in Example 4 using BIO interface.
See `7_bio_openssl` directory.

Example 8:

TLS connection & certificates



BIO TLS connection

- `SSL_CTX_set_verify`, `SSL_get_peer_certificate`,
`SSL_get_verify_result`
- `BIO_new_ssl_connect`, `BIO_get_ssl`, `BIO_do_connect`,
`BIO_do_handshake`

X509

- `X509_STORE_CTX_get_current_cert`, `X509_print_ex_fp`,
`X509_NAME_get_entry`, ...

Connect to `https://www.google.com`.

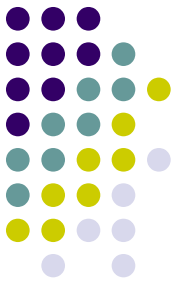
Read and validate certificates.

Sent HTTP GET and receive `/robots.txt` through a secured connection.

See `8_tls_client_openssl` directory.

(optional example)

Signing and certificates



PKCS12

- PKCS12_verify_mac, PKCS12_parse

PKCS7

- PKCS7_sign, PKCS7_verify

X509

- X509_STORE_add_lookup

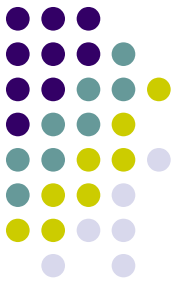
BIO

- BIO_new, BIO_new_mem_buf, BIO_new_file
- BIO_push, BIO_f_cipher, BIO_set_cipher
- BIO_flush, BIO_free_all
- d2i_PKCS12_bio, d2i_PKCS7_bio

See *opt_cert_sign_openssl* directory.

prepare CA signed cert.

script: `opt_cert_sign_openssl/create_CA`



```
#!/bin/bash

CA=ca
CA_SUBJ='/C=CZ/ST=Utopia/L=Brno/O=Test s.r.o./OU=Test CA'
SIGN=sub-ca
SIGN_SUBJ='/C=CZ/ST=Utopia/L=Brno/O=Test s.r.o./OU=Test sign'

PASSWORD="mypassword"

# Generate RSA key for root CA
openssl genrsa -out $CA.key 4096
# self-sign CA
openssl req -new -x509 -days 365 -key $CA.key -out $CA.crt -subj "$CA_SUBJ"

# Generate sub-ordinate CA signed by CA
openssl genrsa -out $SIGN.key 4096
openssl req -new -key $SIGN.key -out $SIGN.csr -subj "$SIGN_SUBJ"
# sign by root CA
openssl x509 -req -days 365 -in $SIGN.csr -CA $CA.crt -CAkey $CA.key -set_serial 01 -out $SIGN.crt

# Package it as PKCS12 file
openssl pkcs12 -export -out $SIGN.p12 -inkey $SIGN.key -in $SIGN.crt -chain -CAfile $CA.crt -password "pass:$PASSWORD"

for i in $(ls *.crt)
do
    h=$(openssl x509 -hash -noout -in $i)
    echo "$i => $h.o"
    ln -s $i $h.o
done
```