GnuTLS

Petr Ročkai

GnuTLS User Tools

- `certtool` – X.509 certificate manipulation
  - `-p` to generate RSA keypairs
  - `-s` to generate a self-signed certificate
- `tpmtool` – talking to TPM chips
- `p11tool` – talking to PKCS#11 smart cards

Exercise 1

- log in to `aisa` using `ssh` (holds for all exercises today)
- use `certtool` to generate a self-signed cert
- generate an RSA keypair first (`-p --outfile foo.key`)
- generate the cert itself `-s --load-privkey foo.key`
  - make a certificate for an HTTPS server
  - let the certificate expire in 30 days
  - set the DNS name to `example.com`

GnuTLS API

- `#include <gnutls/gnutls.h>`
- build with `gcc src.c -lgnutls`
- there are additional headers you may need
  - `#include <gnutls/crypto.h>`
  - `#include <gnutls/abstract.h>`
  - and so on, depending on application

GnutLS API Modules

- `crypto.h` – symmetric cryptography
  - `gnutls_cipher_*` functions
  - `GNUTLS_CIPHER_*` macros
  - AES, ChaCha20, Salsa20, Camellia, …
  - CBC, GCM, CCM
- `abstract.h` – abstract key operations
  - `gnutls_privkey_*` and `gnutls_pubkey_*`
  - asymmetric crypto – private and public keys
  - RSA, DSA, ECDSA

GnutLS API Modules (cont'd)

- `x509.h` – X.509 certificates
  - `gnutls_x509_*`
  - includes another set of privkey/pubkey functions
- `pkcs11.h` – smart cards
  - `gnutls_pkcs11_*`
- and a number of other modules
  - `tpm.h` – trusted platform module
  - `pkcs7.h`
  - `pkcs12.h`
  - `dtls.h`
  - `openpgp.h`

GnuTLS Documentation

- available from `https://gnutls.org`
- also `doc/examples` in the source tarball

Exercise 2

- download and configure `libnettle` 3.4
  - `wget .../nettle-3.4.tar.gz`
  - `tar xzf nettle-3.4.tar.gz`
  - `cd nettle-3.4`
  - `./configure --prefix=$HOME/nettle`
- build and install
  - `make && make install`
  - `NETTLE=$HOME/nettle`
  - `export PKG_CONFIG_PATH=$NETTLE/lib64/pkgconfig`
  - `export LD_LIBRARY_PATH=$NETTLE/lib64`

Exercise 2 (cont'd)

- download `gnutls` 3.5.19 & configure it
  - `tar xaf gnutls-3.5.19.tar.xz`
  - run `./configure` (see below for args)
  - pass `--with-included-libtasn1`
  - and `--with-included-unistring`
  - and `--without-p11-kit`
  - and `--prefix=$HOME/gnutls`
- run `make && make install`
  - add `$HOME/gnutls/lib` to your `LD_LIBRARY_PATH`

Exercise 3

- write a small app that uses gnutls
- you will need to pass some flags to `gcc`
  - `-I$HOME/gnutls/include`
  - `-L$HOME/gnutls/lib`
  - don't forget `-lgnutls`
- compute HMAC of a file using a fixed key
  - you will need `gnutls_hmac_*` from `crypto.h`
  - use SHA-256 as the algorithm

Assignment

- same as lab 6+7 but with `gnutls`
- part 1: 128b AES-CBC [5pt]
  - use the same key / IV as for lab 6
  - cross-check with your `openssl` implementation
  - use `gnutls_cipher_*`
- part 2: RSA keypair generation [5pt]
  - no restriction on the public exponent
  - use `gnutls_privkey_generate`
  - print all key data to the screen

Assignment Hints

- look at `src/certtool-common.c`
  - in the gnutls source code
  - you can copy `print_rsa_pkey`

Deadline

- November 14th 2018, midnight
- the exercise should be easy enough