Bezpečnost v Informačních telchnologiích

Stelnet – tunel pro zabezpečení telnetu

(Semestrální práce)

28.5.2017

Milan Hajžman

A13B0303P

qwerty@pilsfree.net

Zadání

Telnet je velmi starý nezabezpečený protokol pro vzdálenou správu. V praxi již vymizel a nahradilo ho velmi složité SSH, ale to je někdy složité příliš. Například pokud bychom chtěli provozovat ssh na nějakém mikrokontroleru nemusel by se ním programový kód vejít do paměti mikrokontroleru. Proto jsem se rozhodl vytvořit sadu programů, které budou používat již existující telnet a ten budou zašifrovávat.

Analýza problému

Jako hlavní šifru jsem si zvolil AES. Ta má ale nevýhodu v tom že je bloková, kde délka bloku je délkou klíče, v mém případě 128 bitů. Telnet oproti tomu potřebuje posílat často pouze jen jeden znak. To se pokusím odstranit s pomocí paddingu kdy zbytek bloku vyplním náhodnými znaky přičemž poslední znak je číslo kolik znaků v posledním bloku je platných, přičemž pokud budu posílat násobky 16ti tak budu muset přidat celý další neplatný blok. Další problém může být shluky ve streamech a proto před odesláním bloků pošlu jeden bajt s počtem bloků které budou následovat. Nevýhodou AESU je že bloky reflektují data pokud pošlu dvakrát stejná data pak i zašifrovaná data jsou pořád stejná. To lze odstranit pomocí CBC (Cipher block chaining).

Jelikož mám v plánu použít ze seminární práce klientskou část a serverovou reimplemetovat na mikrokontroleru musím volit knihovny které nezaberou příliš mnoho paměti v rom.

Návrh řešení

Pro zahajovací handshake použiji RSA a pro následující komunikaci AES.

AES

Malou implementaci jsem našel na https://github.com/kokke/tiny-AES128-C. Tato implementace umí CBC, která ale nepočítá s více streamy a ukládá si poslední blok do proměnné. Proto jsem obalil třídou AesCbc která počítá s dvěma proudy.

RSA

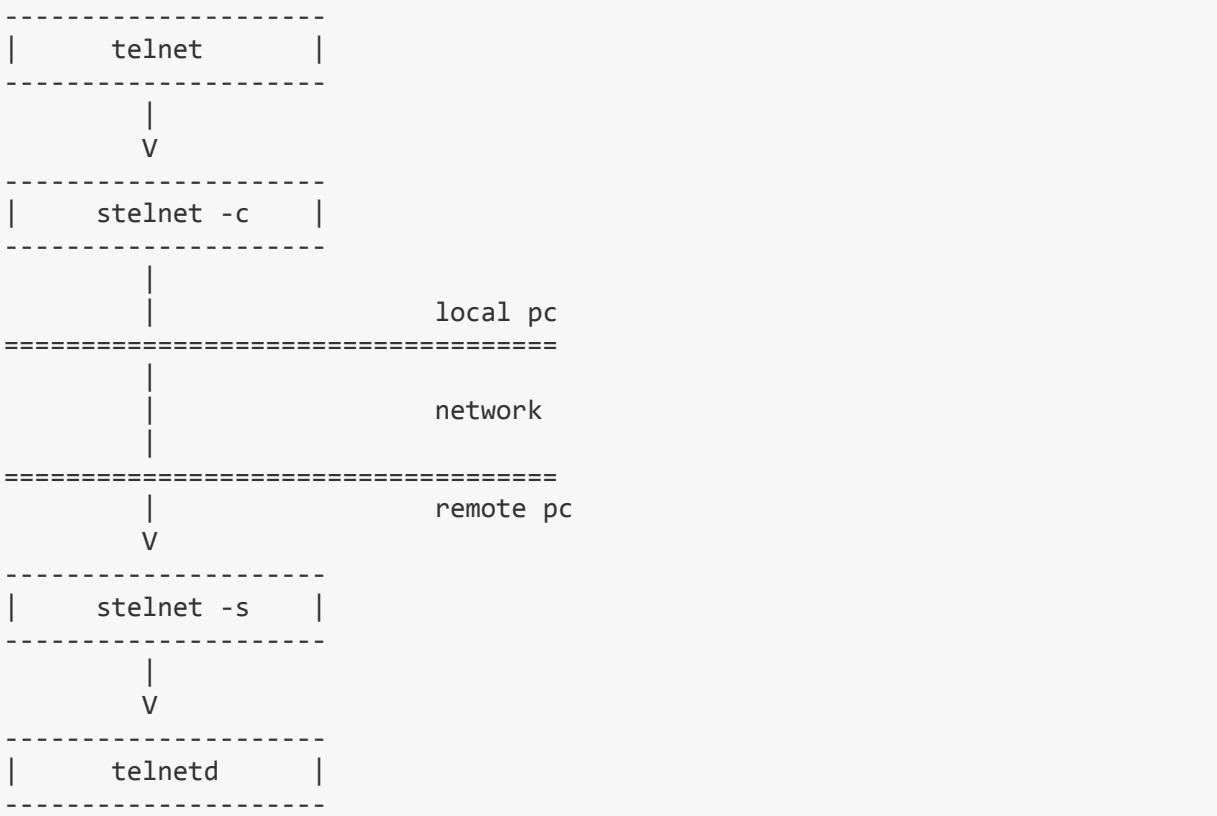
Pro RSA jsem použil knihovnu libgmp, která bohužel není zrovna malá, ale snad půjde obrat o nepotřebné části později. Tato knihovna používá strukturu mpz_t pro reprezentaci velkých celých čísel a obsahuje velké množství matematických funkcí

Popis řešení

Handshake

Proces navazování spojenái probíhá následovně. Serverová a klienstká část čekají na příchozí spojení. Ve chvíli kdy klientská část přijme spojení klienta telnetu, připojí se server. Server odešle binární blob obsahující heslo, klíč a iniciační vektor to vše zašifrované veřejným RSA klíčem. Klient rozšifruje data svým soukromým klíčem a v plaintextu odešle heslo na server. Server zkontroluje heslo a pokud nesedí ukončí spojení. Na ukončené spojení klient reaguje také ukončením spojení s telnetem. Server naváže spojení s telnetd serverem a následují cí komunikace už probíhá pomocí předaného klíče a inicializačního vektoru.

Telnet klient obvykle nezahájí žádnou komunikaci a čeká až mu pošle server nastavovací znaky. Telnetd hned po spojení posílá telnetu speciální znaky. To je pro mě výhodné , protože v tu chvíli už je most kompletní a telnet a telnetd si mohou udělat vlastní handshake.



Důležité zdrojové soubory

tcp_base_object.cpp

Základní objekt od kterého server a klient dědí.

server.cpp

Třída serveru

client.cpp

Třída klienta

key_file.cpp

Objekt pro načtení souboru ve formátu openssl

aes_cbc.cpp

Třída pro šifrování AES a padding cbc

rsa.cpp

Třída pro šifrování RSA

my_random.cpp

funkce pro čtení náhodných čísel z /dev/urandom

base64.cpp

dekódovnání z base64 čísel

Uživatelská dokumentace

Nainstalujeme libgmp

```
apt install libgmp-dev
```

stáhneme zdrojový kód

```
git clone https://github.com/qwerty2586/stelnet
```

zkompilujeme

```
cmake .

make
```

připavíme soubor s klíčem

```
openssl genrsa -out keyfile 2048
```

Spustíme server na počítači s telnetd

```
./stelnet --server --listen-port 3000 --telnetd-port 23 --keyfile keyfile

# or shorter

./stelnet -s -l 3000 -d 23 -k keyfile
```

na klientském počítači pak

```
./stelnet --client --address server_address --port 3000 --telnet 5000 --
keyfile keyfile
# or shorter
./stelnet -c -a server_address -p 3000 -t 3000 -k keyfile
```

a konečně se připojíme z lokalního počítače

```
telnet localhost 5000
```

Závěr

Pokusil jsem se zabezpečit protokol telnet a naimplementoval jsem řešení mezi dvěma počítači. Nejsem spokojený s knihovnou použitou na implementaci RSA šifry. libGMP je velká a musí se přidat k programu dynamicky. Navíc z ní používám jen jeden datový typ a 3 funkce. Při kódění mikrokontroleru se tedy pokusím tyto části z knihovny extrahovat, aby se nemusela přibalovat celá.