Warsztaty z Sieci komputerowych
Lista 7

Na dzisiejszej pracowni karty `enp4s0` są podpięte są do koncentratora, zaś karty `enp3s0` standardowo do przełącznika. W pierwszym zadaniu będziemy korzystać z karty `enp4s0`, zaś we wszystkich następnych z karty `enp3s0`.

Zadanie 1. Przypisz karcie `enp4s0` adres IP równy `10.0.0.`*`nr_komputera`*`/8` poleceniami

```
#> ip link set up dev enp4s0
#> ip addr add 10.0.0.nr_komputera/8 dev enp4s0
```

Pingając adres rozgłoszeniowy sieci `10.0.0.0/8` sprawdź, czy masz połączenie z pozostałymi komputerami w pracowni.

Dobierzcie się trójkami (będziemy posługiwać się określeniami „osoba A", „osoba B" i „osoba C"). Osoba A powinna zmienić hasło użytkownika `student` poleceniem

```
$> passwd
```

na trudne do odgadnięcia, a następnie przekazać je w sekrecie osobie B. Obecne hasło tego użytkownika to `student.8`. Osoba B za pomocą polecenia

```
$> telnet adres_IP_komputera_A_w_sieci_10.0.0.0/8
```

powinna zalogować się na konto `student` na komputerze osoby A. W tym czasie osoba C powinna spróbować podsłuchać Wiresharkiem ruch pomiędzy komputerami osób A i B i odkryć zmienione hasło. Przydatna będzie opcja `Follow | TCP stream`, wybierana z menu kontekstowego Wiresharka po kliknięciu jednego z pakietów należących do połączenia `telnet`.

Przeprowadźcie eksperyment ponownie zamiast `telnet` używając polecenia

```
$> ssh adres_IP_komputera_A
```

Czy podsłuchanie hasła jest nadal możliwe?

Zdekonfiguruj interfejs `enp4s0` poleceniami

```
#> ip addr flush dev enp4s0
#> ip link set down dev enp4s0
```

i poleceniem `passwd` przywróć standardowe hasło użytkownika `student`.

Zadanie 2. W tym zadaniu skonfigurujemy `ssh` tak aby możliwe było łączenie się bez hasła. Najpierw poleceniem

```
$> ssh-keygen
```

wygeneruj klucz publiczny i prywatny. Zapisz je w domyślnych plikach (`.ssh/id_rsa.pub` oraz `.ssh/id_rsa`). Hasło zabezpieczające klucz pozostaw puste.[1] Obejrzyj właśnie wygenerowane pliki z kluczami.

Teraz wystarczy dopisać klucz publiczny do pliku `.ssh/authorized_keys` na serwerze SSH, z którym będziemy się łączyć. Tym serwerem będzie komputer sąsiada; niech *A* oznacza jego adres IP. Najpierw skopiuj klucz na komputer *A* poleceniem

```
$> scp .ssh/id_rsa.pub A:plik_docelowy
```

Następnie używając SSH zaloguj się na ten komputer

```
$> ssh A
```

jako hasło podając `student.8`. Na komputerze *A* dopisz skopiowany właśnie klucz publiczny do pliku `.ssh/authorized_keys`, a następnie wyloguj się:

```
$> cat plik_docelowy >> .ssh/authorized_keys
$> rm plik_docelowy
$> exit
```

Sprawdź, czy działania odniosły skutek, tj. czy możesz zalogować się teraz na komputer *A* bez podawania hasła. Polecenie

```
$> ssh -v A
```

wyświetli kolejne etapy nawiązywania połączenia.

Zadanie 3. W tym zadaniu będziemy korzystać z protokołu SMTP do wysyłania poczty. Najpierw — jak na poprzednich zajęciach — skonfigurujemy program pocztowy bez szyfrowania, a następnie pokażemy, jak szyfrowanie włączyć.

Skonfiguruj program pocztowy KMail do korzystania z adresu `ccnai@example.com`, gdzie *i* jest numerem Twojego komputera. W tym celu w Kmailu wybierz z menu opcję *Settings | Configure KMail*. W oknie konfiguracji z menu po lewej stronie wybierz ikonę *Identities*, a następnie zmodyfikuj domyślną tożsamość *Lab 109 Student* wpisując w polu *Email address* napis `ccnai@example.com`.

W tym samym oknie skonfiguruj serwer poczty przychodzącej. W tym celu z menu wybierz ikonę *Accounts* i w karcie *Receiving* kliknij przycisk *Add*. Wybierz *POP3 E-Mail Server*.[2] W polu *Incoming mail server* wpisz `eagle-server.example.com`, w polu *Username* wpisz `ccnai`, zaś w polu *Password* — `cisco`. W tym samym oknie, w karcie *Advanced* wybierz brak szyfrowania i port 110.

[1] W praktyce pozostawianie klucza prywatnego niezabezpieczonego hasłem to zazwyczaj zły pomysł.

[2] Być może w tym momencie konieczne będzie uzyskanie dostępu do *KDE Wallet Service*; w takim przypadku wybierz opcję *Classic, blowfish encrypted file* i ustal własne hasło zabezpieczające *KDE Wallet*.

W tym samym oknie skonfiguruj również serwer poczty wychodzącej. Wybierz kartę *Sending* i kliknij przycisk *Add*. Wybierz swoją własną nazwę dla tego transportu maili i zaznacz, żeby było on wykorzystywany domyślnie. Następnie w polu *Outgoing mail server* wpisz `eagle-server.example.com` i pozostaw pole uwierzytelnienia puste. W karcie *Advanced* wybierz brak szyfrowania i port 25.

Napisz email testowy do sąsiada lub do samego siebie i zaobserwuj w Wiresharku dane przesyłane za pomocą protokołu SMTP (znowu przydatna będzie wybierana z menu kontekstowego opcja *Follow* | *TCP Stream*). Zapisz komunikaty wymieniane między klientem i serwerem SMTP do pliku.

Następnie włącz szyfrowanie SSL protokołu SMTP w Kmailu. W tym celu wybierz z menu opcję *Settings* | *Configure KMail*. W oknie konfiguracji z menu po lewej stronie wybierz ikonę *Accounts* i w karcie *Sending* kliknij przycisk *Modify*. W karcie *Advanced* wybierz szyfrowanie SSL i port 465.

Wyślij ponownie email testowy do sąsiada lub samego siebie. Być może dostaniesz ostrzeżenie o niepodpisanym certyfikacie; w tym (i tylko w takim!) przypadku testowego serwera SMTP można taki komunikat zignorować. Zaobserwuj przesyłane dane w Wiresharku. Wybierając z menu kontekstowego opcję *Follow* | *TCP Stream* sprawisz, że w oknie z pakietami znajdą się tylko te należące do jednego połączenia TCP. Dane SMTP powinny być zaszyfrowane i nie powinno się ich dać odczytać. Obejrzyj przesyłane przez protokół SSL komunikaty; w szczególności znajdź pakiet, w którym serwer wysyła certyfikat SSL i obejrzyj go.

Na poprzednich zajęciach wysyłaliśmy email posługując się poleceniem `telnet eagle-server.example.com 25`. Dziś zrobimy to ponownie, lecz wykorzystując połączenie szyfrowane. Wykonaj polecenie:

```
$> openssl s_client -quiet -connect eagle-server.example.com:465
```

i wyślij maila posługując się zapisanymi do pliku poleceniami protokołu SMTP (`EHLO`, `MAIL FROM`, `RCPT TO` i `DATA`). Obejrzyj przesyłane dane w Wiresharku i sprawdź w Kmailu czy email został dostarczony.

Zadanie 4. Innym sposobem na uzyskanie szyfrowanego połączenia z serwerem SMTP jest stworzenie tunelu SSH do serwera `eagle-server.example.com`. Sprawdź najpierw, czy potrafisz się zalogować na ten serwer za pomocą SSH poleceniem

```
$> ssh ccnai@eagle-server.example.com
```

gdzie *i* jest numerem Twojego komputera. Hasłem jest `cisco`. Następnie wyloguj się z serwera `eagle-server.example.com` i utwórz tunel SSH łączący port 2525 lokalnego komputera z portem 25 serwera `eagle-server.example.com` poleceniem

```
$> ssh -N -L 2525:localhost:25 ccnai@eagle-server.example.com
```

Sprawdź, jaka usługa odpowiada po drugiej stronie, jeśli (w innym terminalu) wpiszesz polecenie

```
$> telnet localhost 2525
```

Upewnij się, że Wireshark nasłuchuje na wszystkich interfejsach. Zmień konfigurację programu pocztowego, wyłączając w Kmailu szyfrowanie protokołu SMTP i zmieniając adres serwera SMTP na `127.0.0.1` i port `2525`. Sprawdź ustawienia wysyłając testowy email i podglądając pakiety w Wiresharku: pakiety powinny być widoczne w postaci niezaszyfrowanej z `127.0.0.1:`*`lokalny_port`* do `127.0.0.1:2525`, a następnie w postaci zaszyfrowanych danych protokołu SSH wysyłanych do `eagle-server.example.com:22`.

Na końcu przywróć oryginalną konfigurację serwera SMTP w Kmailu, wpisując jako nazwę serwera `eagle-server.example.com`, włączając szyfrowanie i wybierając port 465.

Zadanie 5. W tym zadaniu zapoznamy się z programem `gpg` będącym wolną implementacją standardu OpenPGP. Poleceniem

```
$> gpg --gen-key
```

utwórz parę kluczy PGP: publiczny i prywatny. Wybierz wartości domyślne poza: (i) swoimi danymi, (ii) adresem email (wpisz `ccna`*`i`*`@example.com`) oraz (iii) sensownym hasłem zabezpieczającym klucz prywatny.

Posiadane klucze (odpowiednio prywatne i publiczne) można wyświetlić poleceniami

```
$> gpg --list-secret-keys
$> gpg --list-keys
```

Na razie będą tam widoczne tylko Twoje klucze. Zapisz swój klucz publiczny w czytelnej postaci do pliku `klucz-pgp` poleceniem

```
$> gpg -a --export identyfikator_klucza > klucz-pgp
```

Wyślij powyższy plik sąsiadowi mailem. Otrzymany od sąsiada klucz zapisz w pliku `klucz-pgp-sasiada`. Zaimportuj go do programu `gpg` poleceniem

```
$> gpg --import < klucz-pgp-sasiada
```

Ponownie wyświetl listę posiadanych kluczy:

```
$> gpg --list-secret-keys
$> gpg --list-keys
```

W pliku `wiadomosc` umieść tajną treść. Podpisanie wiadomości swoim kluczem prywatnym i zaszyfrowanie kluczem publicznym sąsiada następi po wydaniu polecenia

```
$> gpg -a -r odbiorca -se wiadomosc
```

gdzie *odbiorca* jest ciągiem umożliwiającym zidentyfikowanie klucza odbiorcy (np. imię odbiorcy lub identyfikator jego klucza). Program ostrzeże nas, że nie mamy zaufania do klucza, który właśnie wykorzystujemy. (Dostaliśmy go pocztą, ale czy jesteśmy pewni, że nadawcą był sąsiad, a nie ktoś podszywający się pod sąsiada?) Aby to naprawić, przerwijmy szyfrowanie wciskając `Ctrl+C` i wydajmy polecenie

```
$> gpg --edit-key identyfikator_klucza_sasiada
```

Po znaku zachęty wpisz polecenie

```
> fpr
```

co wyświetli skrót dla posiadanego klucza publicznego sąsiada. Niech Twój sąsiad sprawdzi, czy skrót dla jego własnego klucza jest taki sam (sąsiad powinien wejść do trybu edycji swojego klucza i również wywołać polecenie `fpr`). Jeśli tak jest, to można bezpiecznie założyć, że posiadany klucz faktycznie należy do sąsiada i podpisać go poleceniem

```
> sign
```

a następnie opuścić tryb edycji poleceniem

```
> quit
```

Zauważ, że jeśli teraz wyświetlisz dostępne klucze publiczne, to przy kluczu sąsiada będzie informacja o pełnym (`full`) zaufaniu do tego klucza. Następnie ponownie wydaj polecenie

```
$> gpg -a -r odbiorca -se wiadomosc
```

Szyfrogram zostanie zapisany do pliku `wiadomosc.asc`, który należy wysłać e-mailem sąsiadowi. Otrzymaną od sąsiada wiadomość zapisz do pliku `otrzymane`. Następnie odszyfruj ją swoim kluczem prywatnym i zweryfikuj prawdziwość podpisu sąsiada poleceniem

```
$> gpg -d otrzymane
```

Każdorazowe wpisywanie poleceń `gpg` jest mało wygodne. Warto zatem skonfigurować obsługę kluczy PGP w programie pocztowym. W tym celu wybierz z menu Kmaila opcję *Settings* | *Configure KMail*. W oknie konfiguracji z menu po lewej stronie wybierz ikonę *Identities*, a następnie zmodyfikuj domyślną tożsamość *Lab 109 Student* wybierając kartę *Cryptography*. W tej karcie wybierz swój klucz prywatny (*OpenPGP signing key*) i publiczny (*OpenPGP encryption key*). Następnie napisz email do sąsiada i przed jego wysłaniem kliknij przyciski *Sign* i *Encrypt*. (Być może konieczne będzie wtedy wybranie z menu klucza publicznego odbiorcy). Sprawdźcie, czy sąsiad może poprawnie odszyfrować wiadomość i zweryfikować poprawność podpisu nadawcy.

Lista i materiały znajdują się pod adresem http://www.ii.uni.wroc.pl/~mbi/dyd/.

Marcin Bieńkowski