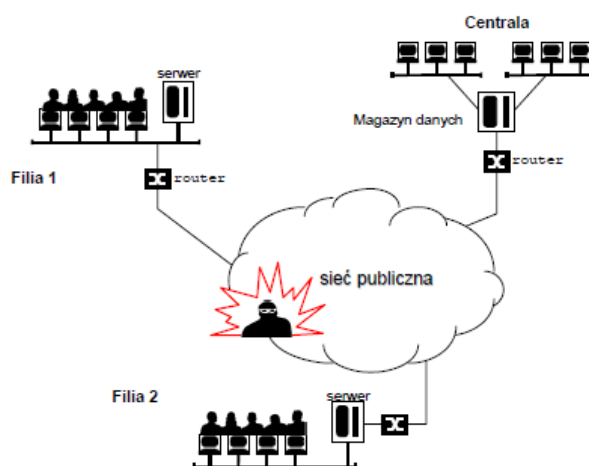


Tunelowanie, VPN i elementy kryptografii

1. Wprowadzenie

W wielu sytuacjach w których mamy do czynienia z rozbudowaną architekturą sieciową zależy nam aby całe środowisko rozproszone (np. oddziały firmy w różnych lokalizacjach) było jak najbardziej spójne. Dotyczy to również adresacji IP. Technologia wirtualnych sieci prywatnych (*ang. Virtual Private Network*) pozwala na rozwiązanie takich problemów. Dzięki utworzeniu sieci VPN można zbudować logiczną sieć komputerową, która będzie łączyć całe rozproszone środowisko ukrywając sieci łączące odległe od siebie lokalizacje i tym samym uprości wymianę danych między nimi. Budując sieć VPN tworzymy logiczne tunele między wyznaczonymi lokalizacjami. W ten sposób technologia VPN tworzy iluzję w której odległe od siebie lokalizacje są fizycznie bezpośrednio połączone. Ta cecha sieci VPN wpływa na uproszczenie sposobu wymiany ruchu między tymi lokalizacjami.

Tunel wirtualny (Virtual Private Network, VPN) jest to kanał komunikacyjny chroniony przed niepowołanym dostępem (odczytem i modyfikacją) poprzez zastosowanie kryptografii. Tunel wirtualny VPN umożliwia chronioną transmisję w obszarze publicznej sieci rozległej, np.: w celu realizacji bezpiecznego połączenia pomiędzy różnymi jednostkami, najczęściej geograficznie odległymi. W sieci publicznej należy się liczyć z potencjalnymi naruszeniami poufności, integralności i autentyczności transmitowanych danych. Mechanizmy kryptograficzne umożliwiają skuteczną ochronę wszystkich tych własności informacji.



Rysunek 1. Schemat sieci publicznej analizowany jako scenariusz zagrożenia

Najprostsze rozwiązania mogą w ogóle nie wspierać ochrony poufności przesyłanych danych, bardziej zaawansowane mogą korzystać z mechanizmu współdzielonego klucza i algorytmów kryptografii symetrycznej do ochrony poufności, najbardziej zaawansowane rozwiązania korzystają z mechanizmów kryptografii klucza publicznego, certyfikatów cyfrowych.

Tunel jest to zestawienie połączenia między dwoma odległymi komputerami tak, by stworzyć wrażenie, że są połączone bezpośrednio. Przesłanki do tworzenia tuneli:

- wygoda
- umożliwienie połączenia między komputerami ukrytymi w sieciach prywatnych (za firewallem, lub z adresami prywatnymi - warunek jeden z hostów biorących udział w tworzeniu tunelu musi mieć adres publiczny)
- bezpieczeństwo - szyfrowanie połączenia: SSL/TLS, SSH
- możliwość przyspieszenia transmisji – kompresja sprawdza się szczególnie na wolnych łączach

Pojęcia tunelowanie zazwyczaj używa się w odniesieniu do przesyłania poprzez tunel tylko jednego protokołu. Zwykle jest to jeden z typowych protokołów np.: POP3, SMTP, HTTP, które przesyłają dane w sposób jawny (w tym nazwy użytkowników i hasła) za pomocą bezpiecznych protokołów TLS/SSL lub SSH. Wirtualna sieć prywatna (VPN) – umożliwia transmisję w sposób bezpieczny wielu protokołów poprzez publiczną sieć. Hosty będące po obu stronach VPN-u „widzą się” tak jakby były w jednej sieci. Takie sieci przeważnie działają w warstwie 3 modelu TCP/IP i tunelują warstwę 3 i wyższe, niektóre z nich umożliwiają także transmisję 2 warstwy (np tworzą wirtualną kartę sieciową). Kompresja powinna spełniać dwa podstawowe kryteria: niskie zapotrzebowanie na moc procesora i szybkość działania. Kompresji może zostać poddana całość transmisji lub tylko ładunek użyteczny pakietów IP.

Mechanizmy kryptografii są powszechnie wykorzystywane w bezpieczeństwie systemów komputerowych. Stanowią uniwersalne narzędzie do osiągnięcia poufności, integralności czy autentyczności.

Szyfrowanie umożliwia zabezpieczenie transmisji. Znane są dwie podstawowe metody szyfrowania:

- a) symetryczna przy pomocy współdzielonego klucza
 - ten sam klucz służy do zaszyfrowania i rozszyfrowania danych
 - jest to szybka i skuteczna metoda szyfrowania
 - podstawowym problemem jest bezpieczne dostarczenie do obu hostów klucza służącego do szyfrowania i deszyfrowania
 - stosowane klucze są stosunkowo krótkie 128, 256, 512b (dawniej także 40 i 56b)
- b) asymetryczna
 - polega na użyciu dwóch kluczy
 - klucza publicznego – klucz jest dostępny dla każdego użytkownika i służy do zaszyfrowania transmisji oraz do weryfikacji podpisu
 - klucza prywatnego – jest przechowywany w bezpiecznym miejscu i służy do rozszyfrowywania danych zaszyfrowanych kluczem publicznym oraz do tworzenia podpisu elektronicznego
 - konieczne jest stosowanie znacznie dłuższych ciągów bitów (zwykle generowane losowo ciągi liczb pierwszych) od długościach równych lub przekraczających 1kb (1024, 2048, 4096b)
 - wymaga większej mocy obliczeniowej niż klucz symetryczny
 - jest znacznie wygodniejsza w użyciu

Aby wykorzystać zalety i zniwelować wady obu metod kryptograficznych większość powszechnie używanych rozwiązań stosuje kombinację obu technik:

- bezpieczne połączenie tworzone jest z wykorzystaniem kluczy publicznego i prywatnego (przy tym sprawdzana jest wiarygodność obu stron)
- generowany jest klucz sesyjny – używany jednorazowo dla konkretnego połączenia i co jakiś czas może być wymieniany dla większego bezpieczeństwa
- klucz sesyjny przesyłany jest przy pomocy bezpiecznego już połączenia
- dalsza transmisja odbywa się z użyciem klucza symetrycznego

2. Rodzaje sieci VPN

Jest wiele rodzajów sieci VPN różniących się sposobem realizacji transmisji, stosowanymi mechanizmami zapewniającymi bezpieczeństwo i cechami funkcjonalnymi. Wśród nich wyróżniamy:

- a. oparte na protokole IPSec
 - sieci typu site-to-site łączące ze sobą w sposób bezpieczny dwie lub więcej sieci; „tunele” pomiędzy tymi sieciami najczęściej są zakończone na dedykowanych urządzeniach takich jak routery z funkcją VPN, firewalle lub koncentratory VPN; nie wymagają instalacji żadnego oprogramowania na komputerach;
 - sieci typu remote-access lub client-to-site łączące w sposób bezpieczny pojedyncze komputery z sieciami; wymagają instalacji na komputerach specjalnego oprogramowania typu VPN Client
- b. oparte na protokole SSL – najczęściej typu remote-access, nie wymagają instalacji specjalnego oprogramowania na komputerze, za to mają mniejszą funkcjonalność niż sieci VPN oparte na protokole IPSec
- c. oparte na innych protokołach / technologiach, np. L2TP

3. Zalety stosowania sieci IPsec VPN

- zapewnienie poufności poprzez szyfrowanie danych silnymi algorytmami kryptograficznymi,
- zapewnienie integralności poprzez uniemożliwienie modyfikacji danych w trakcie transmisji,
- uwierzytelnianie stron poprzez zapewnienie, że nikt nie podszył się pod żadną ze stron,
- zapewnienie niezaprzeczalności, które oznacza, że strony nie mogą zaprzeczyć, że nie wysłały danej informacji, o ile informacja ta była podpisana kluczem prywatnym i podpis został poprawnie zweryfikowany.

4. Metody uwierzytelniania

Zanim zostanie zestawiony wirtualny „tunel” VPN, obie strony muszą się wzajemnie uwierzytelnić, aby mieć pewność, że urządzenie po drugiej stronie tunelu jest tym, za kogo się podaje. Istnieją trzy metody uwierzytelniania:

- a. hasło statyczne, klucze współdzielone (pre-shared key): W trakcie przygotowywania do pracy urządzenia klucz wpisuje się bezpośrednio do pliku konfiguracyjnego. Metody tej nie poleca się z uwagi na łatwość popełnienia pomyłki w trakcie konfiguracji, możliwość podszycia się trzeciej strony w przypadku kompromitacji klucza a także z przyczyn administracyjnych (problematiczne jest zarządzanie połączeniami w obrębie kilku czy kilkunastu urządzeń)
- b. klucze publiczne RSA: Na każdym z urządzeń biorących udział w połączeniu generowana jest para kluczy: prywatny-publiczny. Klucze publiczne należy następnie wymienić ze wszystkimi uczestnikami połączenia. W procesie tym bierze udział człowiek, który musi „ręcznie” dokonać wymiany kluczy. Rozwiązanie to jest praktycznie nieskalowalne, przy większej liczbie urządzeń konieczne jest dokonanie $N*(N-1)$ wymiany kluczy, co jest czasochłonne. Dodatkowo w przypadku kompromitacji jednego z urządzeń należy wykasować stare i wgrać nowe klucze na pozostałych urządzeniach.
- c. certyfikaty cyfrowe: (ze względu na swoją strukturę stanowią najbardziej zaufany mechanizm uwierzytelniania, możliwe jest zautomatyzowanie procesu ich wymiany w przypadku kompromitacji jednej ze stron. Ta metoda uwierzytelniania cechuje się również skalowalnością. Przy „N” stronach biorących udział w połączeniu konieczne jest „N” uwierzytelnień i „N” certyfikatów)

5. Certyfikaty cyfrowe

Przez „certyfikat” rozumiemy dane podpisane cyfrowo przez tzw. „zaufaną trzecią stronę”. Dane, o których mowa zawierają zazwyczaj następujące informacje:

- Klucz publiczny właściciela certyfikatu.
- Nazwę zwyczajową (np. imię i nazwisko, pseudonim, etc.)
- Nazwę organizacji.
- Jednostkę organizacyjną.
- Zakres stosowania (podpisywanie, szyfrowanie, autoryzacji dostępu itp.)
- Czas, w jakim certyfikat jest ważny.
- Informacje o wystawcy certyfikatów.
- Sposób weryfikacji certyfikatu (np. adres, pod którym można znaleźć listy CRL).
- Adres, pod którym znajduje się polityka certyfikacji

Struktura certyfikatu nie jest sztywna i w zależności od potrzeb można umieszczać w niej dodatkowe pola, wykraczające poza definicję standardu. W rozwiązaniach dla sieci VPN certyfikat stanowi element uwierzytelniający każdą ze stron biorących udział w połączeniu. Dzięki temu rozwiązaniu podszycie się pod jedną ze stron biorących udział w połączeniu jest wysoce nieprawdopodobne.

Zalety stosowania certyfikatów:

- uwierzytelniają strony biorące udział w połączeniu
- zapewniają poufność danych
- zapewniają integralność danych
- zapewniają niezaprzeczalność danych

6. Zastosowanie technologii VPN

Głównym zastosowaniem technologii VPN jest tworzenie logicznych kanałów między zdalnymi lokalizacjami. Wiele dużych firm telekomunikacyjnych oferuje swoim klientom możliwość zestawiania połączeń VPN między zdalnymi lokalizacjami zamiast dzierżawy fizycznych łączy z uwagi na oszczędności dla klienta oraz większe elastyczności w zarządzaniu takimi połączeniami dla operatora telekomunikacyjnego. W rozwiązaniach typu SOHO (*ang. Small Office Home Office*) bardziej rozbudowane rozwiązania również wspierają tworzenie sieci VPN. Przydaje się to w sytuacjach, gdy pracownik pracuje w domu i potrzebuje połączenia do sieci firmowej. Inną możliwością jest użycie odpowiedniego oprogramowania klienckiego na komputerze pracownika pracującego poza siedzibą firmy. Takie oprogramowanie zestawia połączenie VPN między komputerem pracownika a urządzeniem dostępowym w siedzibie firmy. Oprócz tradycyjnych rozwiązań VPN wspierających protokół IPsec istnieje druga, również popularna grupa rozwiązań opartych o wykorzystanie protokołu SSL jako mechanizmu do budowy bezpiecznych kanałów wymiany danych. Sieci tworzone z użyciem mechanizmu SSL VPN określane są również jako „sieci bez klienta” (*ang. clientless*) z uwagi, że inaczej niż ma to miejsce w klasycznych sieciach VPN opartych o IPsec, nie jest potrzebne dedykowane oprogramowanie klienckie dla klienta takiej sieci, wystarczy przeglądarka stron www wspierająca protokół SSL. Klient posiadający przeglądarkę wspierającą protokół SSL łączy się z serwerem dostępowym i po ustanowieniu połączenia SSL, po przez przeglądarkę uzyskuje dostęp do wewnętrznych zasobów sieci VPN. Istnieją również inne rozwiązania SSL, które posiadają dedykowane oprogramowanie do zestawiania połączeń SSL VPN. Dzięki takiemu podejściu możliwości połączeń SSL VPN zbliżają się do klasycznych sieci VPN opartych o protokół IPsec i pozwalają przesyłać ruch sieciowy dowolnej aplikacji opakowując go protokołem SSL. Takim rozwiązaniem jest program OpenVPN.

7. Standardy i Protokoły

SSL/TLS

SSL jest połączeniowym protokołem oferującym dwupunktowy tunel kryptograficzny z wykorzystaniem certyfikatów, zaprojektowany z myślą o ochronie sesji takich protokołów jak HTTP, SMTP, POP/IMAP. TLS jest rozwinięciem standardu SSL opracowanego przez firmę Netscape. TLS zapewnia poufność transmisji i uwierzytelnienie stron lub przynajmniej jednej ze stron.

Do działania wymagany jest

- w najprostszej wersji - przynajmniej 1 certyfikat (samo podpisany certyfikat serwera) w tym przypadku klient zawsze będzie informował użytkownika o ułomności tego rozwiązania.
- typowo – Certyfikat(y) CA oraz certyfikat serwera przez niego podpisany, jeżeli klient ma w swojej bazie certyfikat CA połączenie zostanie nawiązane bez ostrzeżeń (potwierdzona zostaje autentyczność tylko jednej ze stron – serwera)
- w wersji najbezpieczniejszej - obie strony transmisji posiadają podpisane certyfikaty, podpisane przez CA znajdujące się w bazie obu stron – potwierdzona jest tożsamość obu stron Bazę certyfikatów CA klienta i serwera uzupełnia lista certyfikatów, które straciły wiarygodność tzw CRL (Certification Revocation List)

SSH1/2

Podstawowym celem powstania oprogramowania SSH była bezpieczna alternatywa dla przestarzałych i niezabezpieczonych protokołów rsh i telnet umożliwiających zdalne wykonywanie poleceń powłoki systemów uniksowych oraz zdalną pracę. Opierały się na domyślnym zaufaniu hostów (rsh) lub na autoryzacji za pomocą nazwy użytkownika i hasła ale przesyłanych w sposób nieszyfrowany i łatwy do przechwycenia szczególnie w sieciach publicznych. Stopniowo funkcjonalność SSH rozszerzono o tunelowanie protokołów rodziny X oraz bezpieczną transmisję plików scp (lub sftp). Obecnie tunelować przy pomocy SSH można dowolny protokół, który używa pojedynczych portów lub wykorzystać wirtualny interfejs tunelingu pozwalający stworzyć tunel w warstwie 3. Początkowo SSH było oprogramowaniem komercyjnym. Bardzo szybko powstała wersja OpenSSH (początkowo obsługująca tylko wersję SSH 1), która od końca lat 90-tych wyparła całkowicie wersję komercyjną. Serwer SSH wykorzystuje standardowo port 22. SSH wykorzystuje zarówno kryptografię asymetryczną (do nawiązania połączenia i przesłania klucza sesyjnego) jak i symetryczną (klucz sesyjny). Przy pierwszym połączeniu klient ssh wymaga akceptacji skrótu kryptograficznego serwera i zapisuje go do bazy a następnie pyta o hasło. SSH w wersji 1 używa asymetrycznego klucza RSA a do transmisji symetrycznych kluczy 3DES i Blowfish.

Wersja 2 SSH aby uniknąć problemów patentowych z RSA (już nie aktualne) używa asymetrycznych algorytmów DSA i DH oraz wiele różnych algorytmów dla kluczy symetrycznych (IDEA, 3DES, Blowfish, AES, Arcfour pochodna RC4).

8. Instrukcje do użytego oprogramowania

a. OpenSSH

Serwer SSH (OpenSSH) zwykle uruchamiany jest jako demon/usługa systemu operacyjnego. Pliki konfiguracyjne oraz klucze przechowywane są w katalogu `/etc/ssh`.

W katalogu użytkownika znajduje się plik `known_hosts` zawierający listę hostów z ich adresami i kluczami publicznym zapisywane po zatwierdzeniu pierwszego połączenia i sprawdzane przy kolejnych.

Przykłady użycia:

```
ssh -L 8080:localhost:80 student@10.0.2.186
```

tworzy tunel pomiędzy lokalnym komputerem nasłuchując na porcie 8080 i przekazuje pakiety do komputera 10.0.2.186 (gdzie jesteśmy zalogowani jako student) ten (zdalny) przekazuje już bez szyfrowania połączenie do adresu localhost (127.0.0.1) na port 80 (WWW)

Połączenie można sprawdzić wywołując URL <http://localhost:8080/> z lokalnego komputera

b. wget

Wget - program służący do pobierania plików z Internetu za pośrednictwem protokołów HTTP, HTTPS i FTP. Jego cechą wyróżniającą jest dobry stosunek możliwości do niewielkich rozmiarów.

```
wget <opcje> URL
```

niektóre opcje:

`-O nazwa_pliku` (duże „O”) zapisuje pobrany dokument w pliku o podanej nazwie

Przykład użycia (do wykorzystania w laboratorium):

```
wget -O /dev/null http://Adres_IP/plik.txt
```

Pobranie pliku o nazwie „plik.txt” z komputera o adresie Adres_IP i przekierowanie strumienia (przez opcję `-O`) do `/dev/null` (czyli wirtualne urządzenie, które usuwa wszystkie dane przekierowane do niego, nie zaśmieca dysku).

c. VirtualBox

Klikamy na zakładkę *Maszyna->Dodaj* wybieramy partycję `C:/` katalog `Vm's/CentOS 7.0 Graph` tam wskazujemy na obraz CentOS.

Literatura:

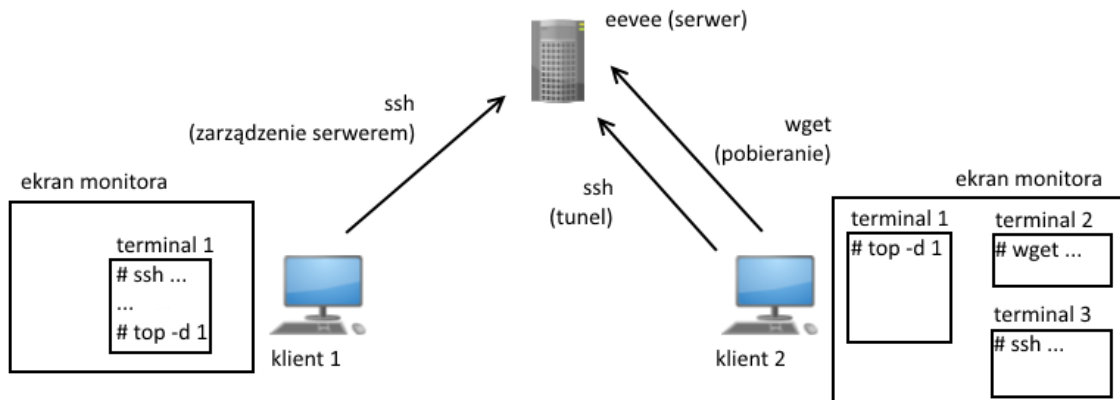
- [1] Stawowski Mariusz, Projektowanie i praktyczne implementacje sieci VPN, Warszawa, ArsKom, 2004
- [2] Karbowski Marcin, Podstawy kryptografii, Gliwice, Helion, 2006

Scenariusz nr 1: SSH z serwerem zewnętrznym (stacjonarne)

1. Sprzęt i oprogramowanie:

- OpenSSH
- Linux CentOS (na wirtualnej maszynie, root/root)
- Maszyna eevee 192.168.102.253 (root/haslo_admin)

2. Przygotowanie do wykonania ćwiczenia:



- Klient 1 – Po zalogowaniu do systemu Windows, uruchomić wirtualną maszynę z systemem CentOS 7 i zalogować się tam na konto root (hasło root). Uruchomić konsolę (terminal 1). Przy użyciu ssh połączyć się z maszyną eevee (serwer) za pomocą polecenia: `ssh 192.168.102.253`
- Klient 2 – Po zalogowaniu do systemu Windows, uruchomić wirtualną maszynę z systemem CentOS 7 i zalogować się tam na konto root (hasło root).
- Z klienta 1, w konsoli zalogowanej na serwerze eevee (terminal 1)
 - uruchomić serwer http Apache komendą: `service httpd start`
 - wyłączyć kontrolę zabezpieczeń i uprawnień dla plików: `setenforce Permissive`
 - upewnić się że plik wybrany do testów znajduje się w katalogu `/var/www/html`
- Na kliencie 1, w konsoli zalogowanej na serwer (terminal 1) ORAZ na kliencie 2 (terminal 1)
 - uruchomić usługę ssh terminal 1: `service sshd start`
 - wyłączyć firewall: `systemctl stop firewalld` oraz `systemctl disable firewalld`
 - uruchomić monitor procesów: `top -d 1`

3. Wariant scenariusza

- Plik do testów:
- Metody szyfrowania :
- Poziomy kompresji:

wersja	metoda szyfrowania	kompresja	uwagi
a)	Blowfish	nie	-c blowfish-cbc -o "Compression no"
b)	Blowfish	tak	-c blowfish -C -o "CompressionLevel X" - X wartość od 0-9
c)	3DES	nie	-c 3des -o "Compression no"
d)	3DES	tak	-c 3des -C -o "CompressionLevel X" - X wartość od 0-9
e)	Arcfour	nie	-c arcfour -o "Compression no"
f)	Arcfour	tak	-c arcfour -C -o "CompressionLevel X" - X wartość od 0-9
g)	Cast	nie	-c cast128-cbc -o "Compression no"
h)	Cast	tak	-c cast128-cbc -C -o "CompressionLevel X" - X wartość od 0-9

4. Wykonanie ćwiczenia:

Wszystko dokumentujemy w postaci zrzutów ekranu !!

1. Na kliencie 2, w nowej konsoli (terminal 2 ze schematu).

Pobrać 4 razy (pierwszy pomiar czasu się nie liczy) za pomocą programu wget plik z serwera (eevee).

```
wget http://192.168.102.253/plik -O - >/dev/null
```

(punkty od 2 do 7 powtarzamy dla każdej metody szyfrowania)

2. Na kliencie 2, w nowej konsoli (terminal 3) zestawić tunel do serwera bez kompresji,

```
ssh 192.168.102.253 -L 8888:localhost:80 <odpowiednie opcje>
```

(!!Uwaga!! – przelogowuje na eevee. Kolejna komenda (w punkcie nr 3) w terminalu nr 2)

3. Na kliencie 2, w konsoli (terminal 2) w oparciu o zestawiony w poprzednim kroku tunel pobrać wyznaczony plik 3 razy za pomocą komendy:

```
wget http://localhost:8888/plik -O - >/dev/null
```

4. Na kliencie 2, w terminalu nr 3, rozłączyć poprzedni tunel (**Ctrl+D**)

(punkty od 5 do 7 powtarzamy dla każdego poziomu kompresji)

5. Na kliencie 2, w terminalu nr 3, zestawić nowy tunel do serwera z kompresją na poziomie X =

```
ssh 192.168.102.253 -L 8888:localhost:80 <odpowiednie opcje>
```

6. Na kliencie 2, w konsoli (terminal 2) w oparciu o zestawiony w poprzednim kroku tunel z kompresją pobrać wyznaczony plik 3 razy za pomocą komendy:

```
wget http://localhost:8888/plik -O - >/dev/null
```

7. Na kliencie 2, w terminalu nr 3, rozłączyć poprzedni tunel z kompresją (**Ctrl+D**)

(przy zmianie poziomu kompresji w kolejnym kroku powrócić do punktu nr 5, przy zmianie metody szyfrowania w kolejnym kroku powrócić do punktu nr 2)

5. Uwagi do dokumentacji testów:

Dla każdej konfiguracji połączenia wykonać dwa zrzuty ekranu na kliencie 2 (a oraz b) oraz jeden zrzut ekranu na kliencie 1 (a):

- Zrzut ekranu w trakcie pobierania danych, na którym w terminalu nr 1 widać obciążenie procesora dla procesu sshd (klient 1) i ssh (klient 2) (za pomocą narzędzia „top”)
- Zrzut ekranu po zakończeniu trzeciego pobrania pliku, na którym w terminalu nr 2 widać czas/przepustowość przesyłania danych we wszystkich trzech próbach.

W trakcie wykonywania obydwu zrzutów (a oraz b) na kliencie 2 powinna być także widoczna wersja tunelu (polecenie ssh wraz z parametrami, zestawiane w terminalu nr 3).

6. Wyniki pomiarów:

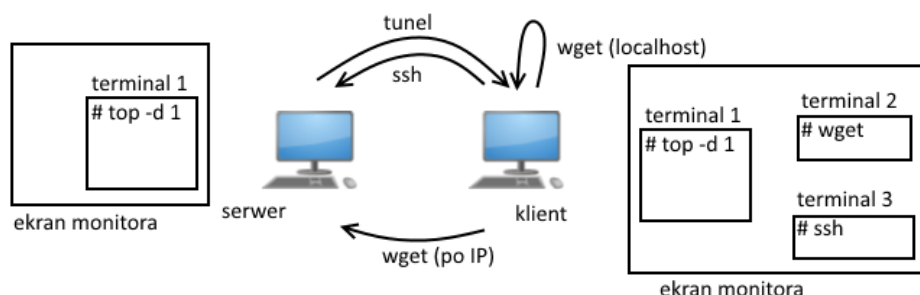
- umieścić zrzuty ekranu dokumentujące realizację głównych faz ćwiczenia. (W celu poprawy przejrzystości i ograniczenia rozmiaru grafiki, ze zrzutów ekranu umieszczanych w sprawozdaniu można wyciąć jedynie fragmenty zawierające odpowiednie informacje. Dodatkowe zrzuty – różniące się metodą szyfrowania, poziomem kompresji można umieścić w sieci/chmurze)
- pogrupować w tabelę i opracować statystycznie (porównanie poszczególnych serii testów)
- ocenić wpływ ustawionych parametrów na obciążenie procesora i sieci na obydwu komputerach

Scenariusz nr 2: SSH z serwerem lokalnym (stacjonarne)

1. Sprzęt i oprogramowanie:

- OpenSSH
- Linux CentOS (na wirtualnej maszynie, root/root)

2. Przygotowanie do wykonania ćwiczenia:



- Server – Po zalogowaniu do systemu Windows, uruchomić wirtualną maszynę z systemem CentOS 7 i zalogować się tam na konto root (hasło root). Zanotować swój adres IP z podsieci 192.168.102.____ (polecenie ifconfig)
- Klient – Po zalogowaniu do systemu Windows, uruchomić wirtualną maszynę z systemem CentOS 7 i zalogować się tam na konto root (hasło root).
- Na serwerze
 - w katalogu `/var/www/html` umieścić plik podany przez prowadzącego i pobrany ze strony `heavy.metal.agh.edu.pl`. Zmienić mu nazwę „plik” (jeżeli „plik” już tam jest, należy go usunąć i umieścić własny)
 - uruchomić serwer http Apache komendą: `service httpd start`
 - wyłączyć kontrolę zabezpieczeń i uprawnień dla plików – `setenforce Permissive`
- Zarówno na kliencie oraz na serwerze:
 - uruchomić usługę sshd: `service sshd start`
 - wyłączyć firewall: `systemctl stop firewalld` oraz `systemctl disable firewalld`
 - uruchomić skrypt monitorowania procesów: `top -d 1` (terminal 1)

3. Wariant scenariusza

- Plik do testów:
 - plik binarny 250MB / 500MB
 - plik video 200MB / 400MB
 - plik tekstowy 300MB / 600MB
- Wersje tunelu:
- Poziomy kompresji:

wersja	metoda szyfrowania	kompresja	uwagi
a)	Blowfish	nie	-c blowfish-cbc -o "Compression no"
b)	Blowfish	tak	-c blowfish -C -o "CompressionLevel X" - X wartość od 0-9
c)	3DES	nie	-c 3des -o "Compression no"
d)	3DES	tak	-c 3des -C -o "CompressionLevel X" - X wartość od 0-9
e)	Arcfour	nie	-c arcfour -o "Compression no"
f)	Arcfour	tak	-c arcfour -C -o "CompressionLevel X" - X wartość od 0-9
g)	Cast	nie	-c cast128-cbc -o "Compression no"
h)	Cast	tak	-c cast128-cbc -C -o "CompressionLevel X" - X wartość od 0-9

4. Wykonanie ćwiczenia:

1. Na kliencie, w nowej konsoli (terminal 2 ze schematu).
Pobrać 4 razy (pierwszy pomiar czasu się nie liczy) za pomocą programu wget plik z serwera.

```
wget http://192.168.102.___/plik -O - >/dev/null
```

(punkty od 2 do 7 powtarzamy dla każdej metody szyfrowania)

2. Na kliencie, w nowej konsoli (terminal 3) zestawić tunel do serwera bez kompresji,

```
ssh 192.168.102.___ -L 8888:localhost:80 <odpowiednie opcje>
```

(!!Uwaga!! – przelogowuje na serwer. Kolejna komenda (w punkcie nr 3) w terminalu nr 2)

3. Na kliencie, w konsoli (terminal 2) w oparciu o zestawiony w poprzednim kroku tunel pobrać wyznaczony plik 3 razy za pomocą komendy:

```
wget http://localhost:8888/plik -O - >/dev/null
```

4. Na kliencie, w terminalu nr 3, rozłączyć poprzedni tunel (**Ctrl+D**)

(punkty od 5 do 7 powtarzamy dla każdego poziomu kompresji)

5. Na kliencie, w terminalu nr 3, zestawić nowy tunel do serwera z kompresją na poziomie X =

```
ssh 192.168.102.___ -L 8888:localhost:80 <odpowiednie opcje>
```

6. Na kliencie, w konsoli (terminal 2) w oparciu o zestawiony w poprzednim kroku tunel z kompresją pobrać wyznaczony plik 3 razy za pomocą komendy:

```
wget http://localhost:8888/plik -O - >/dev/null
```

7. Na kliencie, w terminalu nr 3, rozłączyć poprzedni tunel z kompresją (**Ctrl+D**)

(przy zmianie poziomu kompresji w kolejnym kroku powrócić do punktu nr 5, przy zmianie metody szyfrowania w kolejnym kroku powrócić do punktu nr 2)

7. Uwagi do dokumentacji testów:

Dla każdej konfiguracji połączenia wykonać dwa zrzuty ekranu na kliencie 2 (a oraz b) oraz jeden zrzut ekranu na kliencie 1 (a):

- c) Zrzut ekranu w trakcie pobierania danych, na którym w terminalu nr 1 widać obciążenie procesora dla procesu sshd (klient 1) i ssh (klient 2) (za pomocą narzędzia „top”)
- d) Zrzut ekranu po zakończeniu trzeciego pobrania pliku, na którym w terminalu nr 2 widać czas/przepustowość przesyłania danych we wszystkich trzech próbach.

W trakcie wykonywania obydwu zrzutów (a oraz b) na kliencie 2 powinna być także widoczna wersja tunelu (polecenie ssh wraz z parametrami, zestawiane w terminalu nr 3).

8. Wyniki pomiarów:

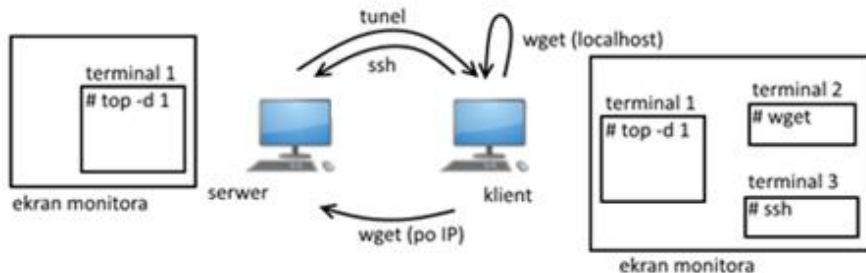
- umieścić zrzuty ekranu dokumentujące realizację głównych faz ćwiczenia. (W celu poprawy przejrzystości i ograniczenia rozmiaru grafiki, ze zrzutów ekranu umieszczanych w sprawozdaniu można wyciąć jedynie fragmenty zawierające odpowiednie informacje. Dodatkowe zrzuty – różniące się metodą szyfrowania, poziomem kompresji można umieścić w sieci/chmurze)
- pogrupować w tabelę i opracować statystycznie (porównanie poszczególnych serii testów)
- ocenić wpływ ustawionych parametrów na obciążenie procesora i sieci na obydwu komputerach

Scenariusz nr 3: SSH (zdalne)

Sprzęt i oprogramowanie:

- PC wyposażony w system Windows
- VirtualBox (<https://www.virtualbox.org>)
- Obraz .iso Linux CentOS 7 (http://isoredirect.centos.org/centos/7/isos/x86_64/)

Przygotowanie do wykonania ćwiczenia:

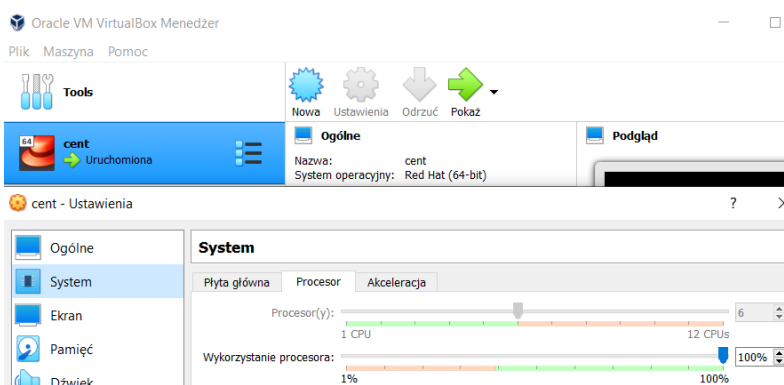


Każdy student na swoim komputerze ściąga program VirtualBox i go instaluje. Dodatkowo ściąga obraz z linuxem. Instrukcja do VirtualBox, w materiale przedstawiono sposób instalacji wirtualnej maszyny z systemem Linux (podczas instalacji najlepiej przydzielić odpowiednią ilość miejsca na dysku twardym dla maszyny wirtualnej, ~10GB):

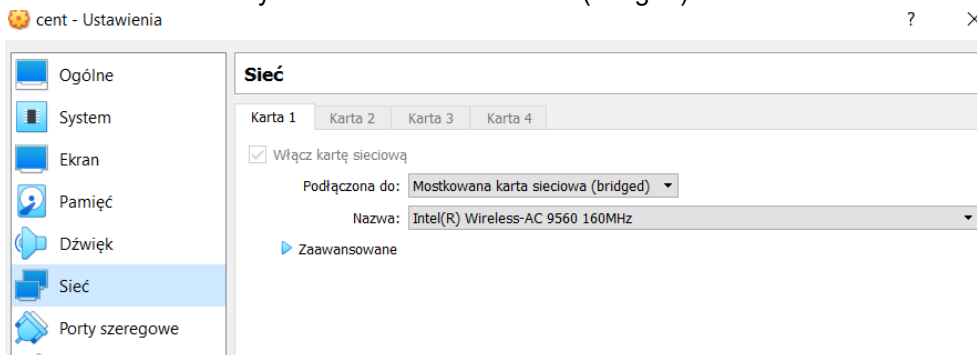
<https://www.download.net.pl/jak-zainstalowac-linuxa-na-virtualbox-dowolna-dystrybucja/n/15416/>

Tak utworzoną maszynę wirtualną wstępnie konfigurujemy:

- W ustawieniach maszyny (PPM na nazwie maszyny i wybieramy Ustawienia), w zakładce System->Procesor ustawiamy na połowę dostępnych procesorów (jego wątków).



- W zakładce Sieć ustawiamy Mostkowa karta sieciowa (bridged)



Tak wstępnie skonfigurowaną maszynę musimy sklonować **Ustawienia->Sklonuj**, automatycznie druga maszyna poza inną nazwą będzie miała tak samo ustawioną konfigurację, sprawdzamy i ewentualnie ustawiamy tak jak dla oryginalnej maszyny.

Uruchamiamy obydwie maszyny i logujemy się na utworzone konta z uprawnieniami Administratora. Do poprawnego zainstalowania w systemie CentOS czasami należy podnieść uprawnienia użytkownika poprzez komendę (w terminalu) **su** i podanie hasła do root'a. Sprawdzamy dostęp do Internetu, musi być. W terminalu sprawdzamy i ewentualnie przy braku instalujemy wymagane oprogramowanie (**realizujemy na obydwu maszynach, jedną nazywać będziemy serwer, drugą klient !**):

- Serwer Apache: w terminalu wpisujemy `service httpd status`, jeżeli usługi brak to ją instalujemy `yum install httpd`
- Demon ssh: w terminalu wpisujemy `service sshd status`, jeżeli usługi brak to ją instalujemy `yum install sshd`
- Sprawdzamy adresy IP i je notujemy: `ifconfig`.

Uruchamiamy/wyłączamy usługi:

Na klient i serwer (w terminalu):

- Serwer Apache: `service httpd start`
- Demon ssh: `service sshd start`
- Firewall: `systemctl stop firewalld` oraz `systemctl disable firewalld`
- Monitor wydajności: `top`

Tylko na serwerze:

- Kontrola zabezpieczeń i uprawnień dla plików: `setenforce Permissive`

Na serwerze (jedna z maszyn) tworzymy plik tekstowy o wybranym rozmiarze (rozmiar dobieramy tak aby czas jego transferu pomiędzy serwer-klient wynosił ok. 10 sekund z wykorzystaniem samego polecenia `wget` bez tworzenia tunelu, proszę zacząć od utworzenia pliku `txt` o rozmiarze 1000MB, w razie potrzeby później można go usunąć i utworzyć inny). Plik tworzymy poleceniem w terminalu:

```
dd if=/dev/zero of=nazwa_pliku.txt bs=rozmiar(np.: 1M) count=1
czyli:
```

```
dd if=/dev/zero of=test.txt bs=1000M count=1
```

Plik ten tworzony jest i zapisywany do katalogu: *Miejsca->Katalog domowy*, należy go przenieść do katalogu: **`/var/www/html`**

Ewentualnie jeżeli prowadzący zaleci to ze strony *heavy.metal.agh.edu.pl* należy ściągnąć podany plik przeznaczony do tunelowania. Ten/te dodatkowe pliki także przenosimy do katalogu: `/var/www/`

Pamiętamy pełne nazwy i rozszerzenia plików – później przy transferach podajemy ich pełne nazwy + to rozszerzenie.

Wariant scenariusza

- Pliki do testów:
- Metody szyfrowania :
- Poziomy kompresji:

wersja	metoda szyfrowania	kompresja	uwagi
a)	Blowfish	nie	<code>-c blowfish-cbc -o "Compression no"</code>
b)	Blowfish	tak	<code>-c blowfish -C -o "CompressionLevel X" - X wartość od 0-9</code>
c)	3DES	nie	<code>-c 3des -o "Compression no"</code>
d)	3DES	tak	<code>-c 3des -C -o "CompressionLevel X" - X wartość od 0-9</code>
e)	Arcfour	nie	<code>-c arcfour -o "Compression no"</code>
f)	Arcfour	tak	<code>-c arcfour -C -o "CompressionLevel X" - X wartość od 0-9</code>
g)	Cast	nie	<code>-c cast128-cbc -o "Compression no"</code>
h)	Cast	tak	<code>-c cast128-cbc -C -o "CompressionLevel X" - X wartość od 0-9</code>

Wykonanie ćwiczenia: Wszystko dokumentujemy w postaci zrzutów ekranu !!

1. Na kliencie, w nowej konsoli (terminal 2 ze schematu).

Pobrać 4 razy (pierwszy pomiar czasu się nie liczy) za pomocą programu wget plik z serwera

```
wget http://adres_IP/pełna_nazwa_pliku -O ->/dev/null
```

adres_IP – adres serwera, na którym są udostępnione pliki

pełna_nazwa_pliku – nazwa + rozszerzenie

na zrzucie poniżej (terminal 2, klient) są podstawowe informacje: prędkość przesyłu i jego czas

```
[root@linux-1 mp]# wget http://192.168.1.40/test2.txt -O ->/dev/null
--2020-12-18 12:09:47-- http://192.168.1.40/test2.txt
  Łączenie się z 192.168.1.40:80... połączono.
  Żądanie HTTP wysłano, oczekiwanie na odpowiedź... 200 OK
  Długość: 723648512 (690M) [text/plain]
  Zapis do: `STDOUT'

100%[=====] 723.648.512 121MB/s w 7,3s

2020-12-18 12:09:55 (94,1 MB/s) - zapisano na standardowe wyjście [723648512/723648512]
```

(punkty od 2 do 7 powtarzamy dla każdej metody szyfrowania, podczas każdej próby, na obydwu maszynach, w terminalu 1 – top, monitorujemy zużycie procesora dla procesów odpowiedzialnych za przesył: ssh, sshd)

2. Na kliencie, w nowej konsoli (terminal 3) zestawić tunel do serwera bez kompresji,

```
ssh -L 8080:localhost:80 adres_IP odpowiednie opcje (uwagi z tabeli powyżej)
```

(!!Uwaga!! – przelogowuje na serwer. Kolejna komenda (w punkcie nr 3) w terminalu nr 2)

Terminal 3 (klient) – przykład użycia komendy do stworzenia tunelu z wykorzystaniem metody szyfrowania Blowfish oraz bez poziomów kompresji. Podczas tworzenia tunelu czasami pojawi się zapytanie na które odpowiadamy pełne yes, podajemy też hasło do root'a maszyny, z którą się łączymy, czyli hasło serwera. Nastąpi przelogowanie na serwer, co obrazuje zmiana root@linux-1 na root@linux-2. Ten terminal pozostaje tylko i wyłącznie do tworzenia i zamykania tunelu !

```
[root@linux-1 mp]# ssh -L 8080:localhost:80 192.168.1.40 -c blowfish-cbc -o "Compression no"
root@192.168.1.40's password:
Last login: Fri Dec 18 12:06:30 2020
[root@linux-2 ~]#
```

3. Na kliencie, w konsoli (terminal 2) w oparciu o zestawiony w poprzednim kroku tunel pobrać wyznaczony plik 3 razy za pomocą komendy:

```
wget http://localhost:8080/pełna_nazwa_pliku -O ->/dev/null
```

Terminal 2 (klient) – pobieranie pliku przez tunel, z metodą szyfrowania bez poziomu kompresji, czas jest dłuższy, prędkość mniejsza. Notujemy te wartości, lub robimy zrzuty ekranu.

```
[root@linux-1 mp]# wget http://localhost:8080/test2.txt -O ->/dev/null
--2020-12-18 12:38:28-- http://localhost:8080/test2.txt
  Translacja localhost (localhost)... ::1, 127.0.0.1
  Łączenie się z localhost (localhost)[:1]:8080... połączono.
  Żądanie HTTP wysłano, oczekiwanie na odpowiedź... 200 OK
  Długość: 723648512 (690M) [text/plain]
  Zapis do: `STDOUT'

100%[=====] 723.648.512 73,1MB/s w 9,9s

2020-12-18 12:38:38 (70,0 MB/s) - zapisano na standardowe wyjście [723648512/723648512]
```

Podczas pobierania w terminalu 1, z uruchomioną usługą top (na serwerze i kliencie) monitorujemy zużycie procesora, które albo notujemy albo realizujemy zrzut ekranu.

Terminal 1 (klient) – przedstawia zużycie procesora dla usługi ssh podczas pobierania.

```
top - 12:45:14 up 1:09, 4 users, load average: 0,18, 0,06, 0,06
Tasks: 247 total, 2 running, 245 sleeping, 0 stopped, 0 zombie
%Cpu(s): 4,2 us, 4,7 sy, 0,0 ni, 82,0 id, 0,0 wa, 0,0 hi, 9,1 si, 0,0 st
KiB Mem : 1881316 total, 390252 free, 792196 used, 698868 buff/cache
KiB Swap: 839676 total, 839676 free, 0 used. 902616 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
4504	root	20	0	183336	4892	3808	R	99,7	0,3	0:22.73	ssh
4703	root	20	0	151052	2532	1928	S	6,3	0,1	0:00.21	wget
2302	mp	20	0	3925480	191512	69404	S	5,3	10,2	0:51.09	gnome-shell
1681	root	20	0	272800	52552	26076	S	0,7	2,0	0:00.50	x

Terminal 1 (serwer) - przedstawia zużycie procesora dla usługi sshd podczas pobierania.

```
top - 12:47:06 up 1:11, 4 users, load average: 0,16, 0,08, 0,06
Tasks: 250 total, 2 running, 248 sleeping, 0 stopped, 0 zombie
%Cpu(s): 5,9 us, 0,9 sy, 0,0 ni, 83,3 id, 0,0 wa, 0,0 hi, 9,9 si, 0,0 st
KiB Mem : 1881316 total, 83660 free, 812032 used, 985624 buff/cache
KiB Swap: 839676 total, 754932 free, 84744 used. 902956 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
5613	root	20	0	160812	7684	4452	R	88,0	0,4	0:30.50	sshd
5942	setroub+	20	0	379996	60296	10124	S	20,9	3,2	0:00.63	setroubleshootd
2382	mp	20	0	4002100	159896	42584	S	1,0	8,5	1:10.30	gnome-shell
24	root	20	0	0	0	0	S	0,7	0,0	0:00.45	ksoftirqd/3
4838	apache	20	0	230528	3868	1936	S	0,7	0,2	0:00.04	httpd

4. Na kliencie, w terminalu nr 3, rozłączyć poprzedni tunel (**Ctrl+D**)

Terminal 3 (klient) – po wciśnięciu kombinacji klawiszy Ctrl+D, nastąpi zamknięcie tunelu.

```
[root@linux-1 mp]# ssh -L 8080:localhost:80 192.168.1.40 -c blowfish-cbc -o "Compression no"
root@192.168.1.40's password:
Last login: Fri Dec 18 12:27:54 2020 from linux-1.home
[root@linux-2 ~]# logout
Connection to 192.168.1.40 closed.
[root@linux-1 mp]# █
```

Punkty od 5 do 7 powtarzamy dla każdego poziomu kompresji, podczas każdej próby z punktów 5-7 monitorujemy: na serwerze zużycie procesora, na kliencie: zużycie procesora, czas i prędkość przesyłu, zgodnie z przedstawionymi powyżej w formie zrzutów ekranu przykładami.

5. Na kliencie, w terminalu numer 3, zestawić nowy tunel do serwera z kompresją na poziomie X (podanym przez prowadzącego).

ssh -L 8080:localhost:80 adres_IP odpowiednie opcje

```
[root@linux-1 mp]# ssh -L 8080:localhost:80 192.168.1.40 -c blowfish -C -o "Compression Level 3"
root@192.168.1.40's password:
Last login: Fri Dec 18 12:38:08 2020 from linux-1.home
[root@linux-2 ~]#
```

6. Na kliencie, w terminalu numer 2 w oparciu o zestawiony w poprzednim kroku (5) tunel z kompresją pobrać wyznaczony plik 3 razy za pomocą komendy:

```
wget http://localhost:8080/pełna_nazwa_pliku -O ->/dev/null
```

```
[root@linux-1 mp]# wget http://localhost:8080/test2.txt -O ->/dev/null
--2020-12-18 12:55:28-- http://localhost:8080/test2.txt
Translacja localhost (localhost)... ::1, 127.0.0.1
Łączenie się z localhost (localhost)::1:8080... połączono.
Żądanie HTTP wysłano, oczekiwanie na odpowiedź... 200 OK
Długość: 723648512 (690M) [text/plain]
Zapis do: `STDOUT'

100%[=====>] 723.648.512 79,9MB/s w 8,6s
2020-12-18 12:55:36 (80,6 MB/s) - zapisano na standardowe wyjście [723648512/723648512]

[root@linux-1 mp]# █
```

7. Na kliencie, w terminalu nr 3, rozłączyć poprzedni tunel z kompresją (**Ctrl+D**)

```
[root@linux-1 mp]# ssh -L 8080:localhost:80 192.168.1.40 -c blowfish -C -o "Compression
Level 3"
root@192.168.1.40's password:
Last login: Fri Dec 18 12:38:08 2020 from linux-1.home
[root@linux-2 ~]# logout
Connection to 192.168.1.40 closed.
[root@linux-1 mp]#
```

**(przy zmianie poziomu kompresji w kolejnym kroku powrócić do punktu nr 5,
przy zmianie metody szyfrowania w kolejnym kroku powrócić do punktu nr 2)**

Uwagi do dokumentacji testów:

Dla każdej konfiguracji połączenia wykonać zrzuty ekranu:

- Zrzut ekranu w trakcie pobierania danych, na którym w terminalu nr 1 widać obciążenie procesora dla procesu sshd/ssh (klient i serwer, za pomocą narzędzia „top”)
- Zrzut ekranu po zakończeniu trzeciego pobrania pliku, na którym w terminalu nr 2 widać czas/przepustowość przesyłania danych we wszystkich trzech próbach.
- W trakcie wykonywania zrzutów powinna być także widoczna wersja tunelu (polecenie ssh wraz z parametrami, zestawiane w terminalu nr 3).

Wyniki pomiarów:

- umieścić zrzuty ekranu dokumentujące realizację głównych faz ćwiczenia. (W celu poprawy przejrzystości i ograniczenia rozmiaru grafiki, ze zrzutów ekranu umieszczanych w sprawozdaniu można wyciąć jedynie fragmenty zawierające odpowiednie informacje.
- pogrupować w tabelę i opracować statystycznie (porównanie poszczególnych serii testów)
- ocenić wpływ ustawionych parametrów na obciążenie procesora i sieci na obydwu komputerach

Scenariusz nr 4: Tunelowanie putty (zdalne)

Sprzęt i oprogramowanie:

- Komputer wyposażony w system Windows
- Zainstalowany pakiet XAMPP (lub dowolny inny serwer http)
- Klient ssh putty <https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html>

Przykładowe zastosowanie:

W ramach tego scenariusza laboratorium testowane jest zestawianie zdalnego tunelu w wersji remote z hosta ukrytego za NATem do hosta o publicznym adresie IP. W tym laboratorium tunelowany jest serwer HTTP, który z powodu mechanizmu NAT jest normalnie niedostępny dla użytkowników z zewnątrz. Po zastosowaniu tunelu, zostaje on udostępniony na zdefiniowanym w opcjach tunelu porcie serwera zewnętrznego. Inne przypadki wykorzystania takich tuneli mogą obejmować tunelowanie innych typów usług ukrytych za natem (strumieniu video, udostępnianie plików, czy też dowolnej innej komunikacji sieciowej nasłuchującej na otwartym porcie serwera w ramach NATu).

Inną wersją takiego tunelu jest wersja tunelowania local, która umożliwia przetunelowanie w ramach portu SSH innych usług działających na serwerze zdalnym do sieci lokalnej. Tunelowanie z wykorzystaniem SSH można też wykorzystać do uruchamiania zdalnych aplikacji z serwerów linuxowych (tzw. tunelowanie Xów).

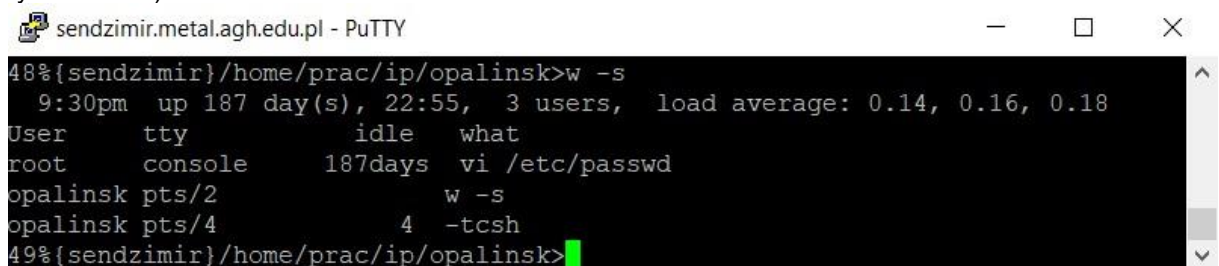
Przygotowanie do wykonania ćwiczenia:

1. Instalacja pakietu XAMPP (lub innego serwera http)
2. Pobranie z serwera <http://heavy.metal.agh.edu.pl/> z części Lab5-tunelowanie, Scenariusz nr 4 (z wykorzystaniem Putty) (prawy przycisk myszy i „zapisz link jako”) i umieszczenie w katalogu serwera http (C:\katalog_instalacji\xampp\htdocs) plików:
 - a. binarnego
 - b. video
 - c. tekstowego

w rozmiarze zależnym od przepustowości wykorzystywanego łącza.

3. Uruchomienie XAMPP'a z uprawnieniami administratora
4. Uruchomienie serwera http (Apache) w panelu XAMPP.
Poprawną konfigurację i uruchomienie serwera Apache można sprawdzić poprzez wpisanie w pasku przeglądarki:
 - localhost – powinna wyświetlić się strona startowa XAMPP
 - localhost/250MB.bin – powinno rozpocząć się pobieranie pliku
5. Zestawienie pierwszego połączenia poprzez putty (diagnostycznego) z serwerem `sendzimir.metal.agh.edu.pl`

Po uzyskaniu połączenia uruchamiamy w terminalu polecenie: „w -s” i wykonujemy zrzut ekranu (powinniśmy uzyskać informacje o aktualnie zalogowanych użytkownikach, jak na rysunku nr 1)



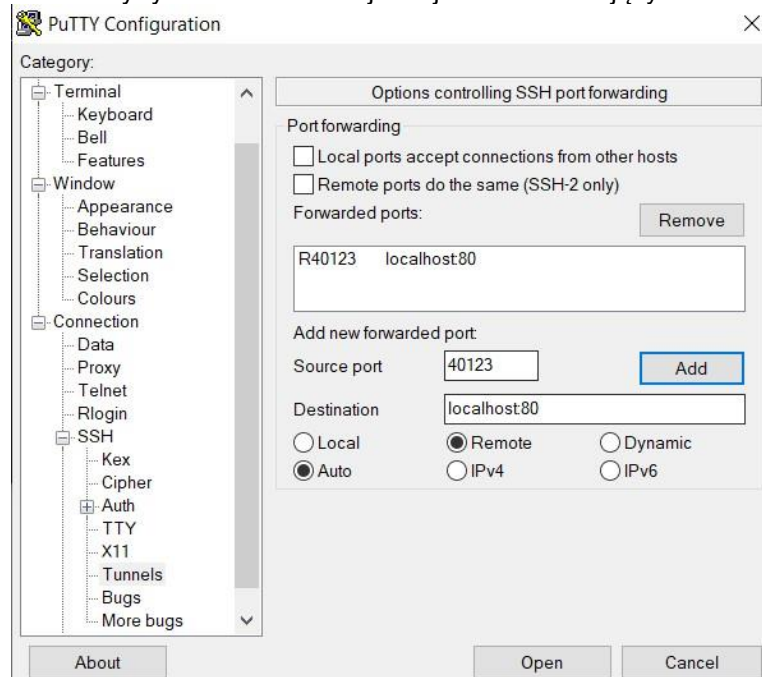
```
sendzimir.metal.agh.edu.pl - PuTTY
48%{sendzimir}/home/prac/ip/opalinsk>w -s
 9:30pm up 187 day(s), 22:55,  3 users,  load average: 0.14, 0.16, 0.18
User      tty          idle   what
root      console     187days vi /etc/passwd
opalinsk  pts/2              w -s
opalinsk  pts/4              4   -tcsh
49%{sendzimir}/home/prac/ip/opalinsk>
```

Rys nr 1: Sprawdzenie użytkowników zalogowanych na serwerze

6. W konsoli w której poprzednio sprawdzaliśmy kto jest zalogowany na serwerze, wykonujemy polecenie „top -s 1”
Ta konsola będzie wykorzystana w celach diagnostycznych.

Wykonanie ćwiczenia:

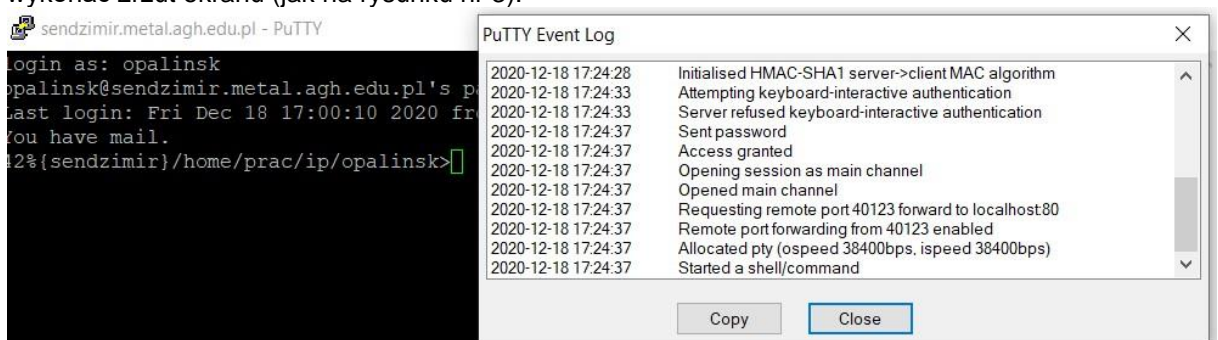
1. Za pomocą Putty należy zestawić drugie połączenie ssh z tunelowaniem (jak na rysunku nr 2). W zakładce Connection->SSH->Tunnels w polu „Source port” należy wpisać **40xyz**, gdzie xyz to 3 ostatnie cyfry numeru indeksu jednej z osób realizujących laboratorium.



Rys nr 2: Konfiguracja połączenia z tunelowaniem

W polu „Destination” należy wpisać „localhost:80” oraz zaznaczyć opcję „Remote”. Po uzupełnieniu wszystkich pól należy kliknąć „Add”, aby dodać parametry tunelu. (po kliknięciu „Add” zestawione przekierowanie portów pojawi się w polu tekstowym). Następnie należy kliknąć „Open” aby nawiązać połączenie.

2. Aby sprawdzić, czy udało się nawiązać połączenie z odpowiednim przekierowaniem portu należy uruchomić konsolę logów putty, przez kliknięcie prawym klawiszem myszy w nagłówek okna i wybranie opcji „Event log”. Należy przewinąć okno do miejsca w którym widać informacje o przekierowanym porcie i wykonać zrzut ekranu (jak na rysunku nr 3).



Rys. nr 3. Konsola Event log z informacją o porcie przekierowania.

3. W konsoli uruchomionej w poprzednim punkcie (tunelującej porty), za pomocą polecenia wget należy kolejno pobrać wszystkie trzy pliki (video, binarny, tekstowy).

Przykładowe polecenie pobrania 20 megabajtowego pliku tekstowego, przy przekierowanym porcie 40123 zaprezentowano na rysunku nr 4. Przy pobieraniu pliku nie można zapomnieć o opcji „-O /dev/null” (średnik duże O (nie zero) spacja i /dev/null przez dwa „/”), która spowoduje niezapisanie pobieranego pliku. W wypadku pominięcia tej opcji bardzo prawdopodobne jest przekroczenie dopuszczalnej quote systemu plików, co uniemożliwi dalsze wykonywanie laboratorium (w takim wypadku należy usunąć pobrane przypadkowo

pliki)

```
62%{sendzimir}/home/prac/ip/opalinsk/tunnel>wget -O /dev/null http://localhost:40123/20MB.txt
```

Rys nr 4. Przykładowa komenda pobrania pliku tekstowego w terminalu tunelującym porty.

W trakcie pobierania pliku (w okolicy 50% pobierania) w terminalu diagnostycznym należy uruchomić polecenie: „ps -xv” * i dopasować rozmiar konsoli w ten sposób, aby widoczne były na niej wszystkie elementy oznaczone na rysunku nr 5, jako:

A) **pełne polecenie wget** (z numerem portu oraz nazwą pliku) – proszę uważać aby nazwa pliku nie została „obcięta”

B) % zużycia CPU dla procesu demona SSH (sshd)

C) % zużycia CPU dla procesu pobierania wget

Po zakończeniu procesu pobierania pliku **należy wykonać zrzut ekranu**, na którym widoczne będą wszystkie powyższe elementy, oraz dodatkowo oznaczony na rysunku nr 5 element D), czyli średnia przepustowość w trakcie pobierania pliku, która znajduje się w konsoli tunelującej.



Rys nr 5: Zrzut ekranu z informacjami podsumowującymi pobieranie pliku.

Cały ten punkt powtarzamy trzykrotnie - dla wszystkich 3 plików (tekstowego, video i binarnego).

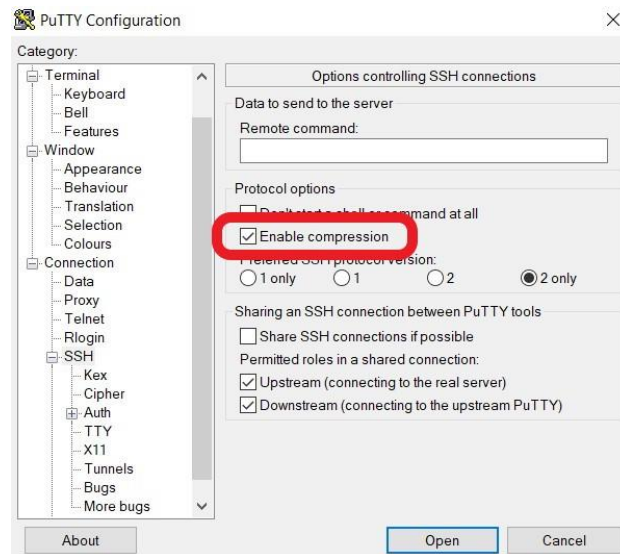
* Jeśli komenda „ps -xv” nie zadziała, należy podać ścieżkę bezwzględną do polecenia, lub uruchomić ją (względnie lub bezwzgl.) z innymi opcjami formatującymi (jak na rysunku 5b)

```
46%{sendzimir}/home/prac/ip/opalinsk>/usr/ucb/ps -xv
  PID TT      S   TIME SIZE  RSS %CPU %MEM COMMAND
 4389 pts/1    S   0:00 2776 2096 0.1 0.5 -tcsh
 1766 pts/2    S   0:00 2696 1992 0.1 0.4 -tcsh
 1764 ?        S   0:01 5960 1856 0.1 0.4 /usr/local/sbin/sshd -R
 4387 ?        S   0:00 5952 1816 0.1 0.4 /usr/local/sbin/sshd -R
 4518 pts/1    O   0:00 1672 1184 0.1 0.3 /usr/ucb/ps -xv
 4504 pts/2    S   0:00 1968 1144 0.1 0.3 ping -s agh.edu.pl
47%{sendzimir}/home/prac/ip/opalinsk>/usr/bin/ps -ao user,pid,stime,pcpu,args
  USER  PID  STIME %CPU COMMAND
opalinsk 4504 12:24:17 0.0 ping -s agh.edu.pl
  root  4550 12:24:54 0.1 /usr/bin/ps -ao user,pid,stime,pcpu,args
48%{sendzimir}/home/prac/ip/opalinsk>
```

Rys 5b. Alternatywne wersje komendy ps zwracającej obciążenie procesora dla procesów

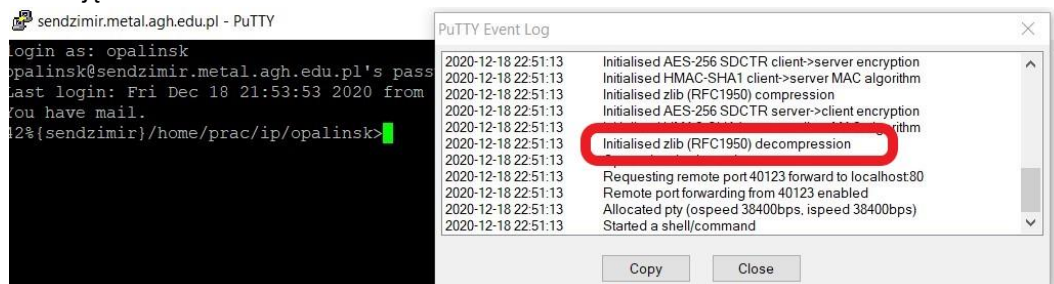
4. Po pobraniu wszystkich trzech plików rozłączamy konsolę tunelującą. (wpisując „exit” w terminalu)

5. Zestawiamy ponownie połączenie w oparciu o Putty z tunelowaniem portów, z tym że tym razem w trakcie konfiguracji połączenia w zakładce Connection->SSH zaznaczamy dodatkowo checkbox przy pozycji „Enable compression” (jak na rysunku nr 6). Pozostałe parametry tunelu (numery portów), są identyczne jak w przypadku konfiguracji tunelu z punktu 1 (rysunek nr 2).



Rys nr 6. Konfiguracja tunelu z kompresją

Poprawne zestawienie tunelu z kompresją można potwierdzić poprzez uruchomienie konsoli Event Log (jak w punkcie 2), gdzie powinna widnieć informacja o kompresji, jak na rysunku nr 7. Po zestawieniu takiego połączenia należy wykonać zrzut ekranu na którym widać taką informację.



Rys nr 7. Informacja o kompresji w konsoli Event Log

6. Po zestawieniu tunelu z kompresją ponownie pobieramy wszystkie 3 pliki (tekstowy, video oraz binarny) wykonując odpowiednie pomiary zużycia procesora („ps -xv” w konsoli diagnostycznej) oraz przepustowości (na koniec procesu pobierania wget w konsoli tunelującej) udokumentowane zrzutami ekranu – identycznie jak opisano w punkcie 3 i zaprezentowano na rysunku nr 5.

Wyniki pomiarów:

Wynikami przeprowadzonych testów powinno być:

- 6 zrzutów ekranów (jak na rys. nr 5) na których widać wartości opisane w punktach A,B,C i D.
- 1 zrzut ekranu z konsoli diagnostycznej z początku laboratorium z informacją o użytkownikach zalogowanych na serwerze w momencie testów
- 2 zrzuty konsoli Event Log dokumentujące zestawienie połączenia z przekierowaniem portu (w punkcie 3 i 6) oraz dodatkowo z kompresją (w punkcie 6)

Uzyskane dane należy opracować (tabele i wykresy) pod kątem zużycia procesora i obciążenia sieci w zależności od rodzaju transmitowanych danych i zastosowania (lub nie) kompresji tunelu. Należy wyciągnąć wnioski z uzyskanych rezultatów i spróbować wytłumaczyć przyczyny uzyskanych wyników (zaistniałych różnic i podobieństw poszczególnych serii testów).