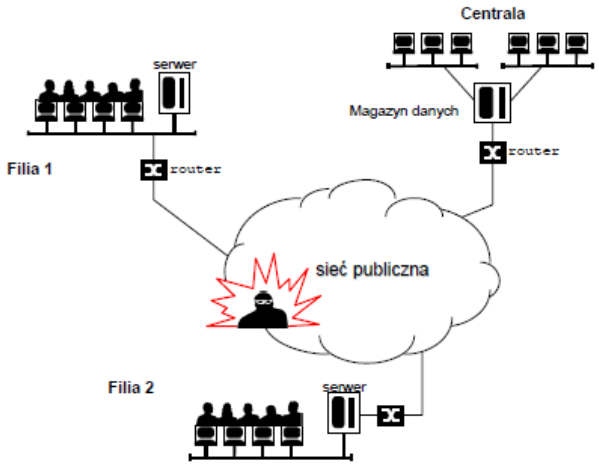
Tunelowanie, VPN i elementy kryptografii

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 siec VPN tworzymy logiczne tunele między wyznaczonymi lokalizacjami. W ten sposób technologia VPN tworzy iluzje w której odległe od siebie lokalizacje są fizycznie bezpośrednio połączone. Ta cecha sieci VPN wpływa na uproszenie 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 zagrożeń

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 osiągania 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
     - klucz prywatny – jest przechowywany w bezpiecznym miejscu i służy do rozszyfrowywania danych zaszyfrowanych kluczem publicznym oraz do tworzenie 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 większych mocy obliczeniowych 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, ze nikt nie podszył się pod żądną 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 (problematyczne jest zarzadzanie 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 trzecia 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 miedzy zdalnymi lokalizacjami zamiast dzierżawy fizycznych łącz z uwagi na oszczędności dla klienta oraz większe elastyczności w zarzadzaniu 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. Inna możliwość to użycie odpowiedniego oprogramowania klienckiego na komputerze pracownika pracującego poza siedziba 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 tun linuksa pozwalający tworzyć 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 klucz sesyjnego) ja 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 klucz 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 student@10.0.2.186 -L 8080:localhost:80
```

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 loclahost (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 stosunek możliwości do niewielkich rozmiarów.
`wget <opcje> URL`
niektóre opcje:

-O (duże „o”) skierowanie pobieranego pliku do pliku o innej nazwie „-” standardowe wyjście
-o (małe „o”) - skierowanie komunikatów programu do pliku

Przykład użycia:

```
wget http://Adres_IP/plik
```

Pobranie pliku o nazwie „plik” z komputera o adresie Adres_IP.

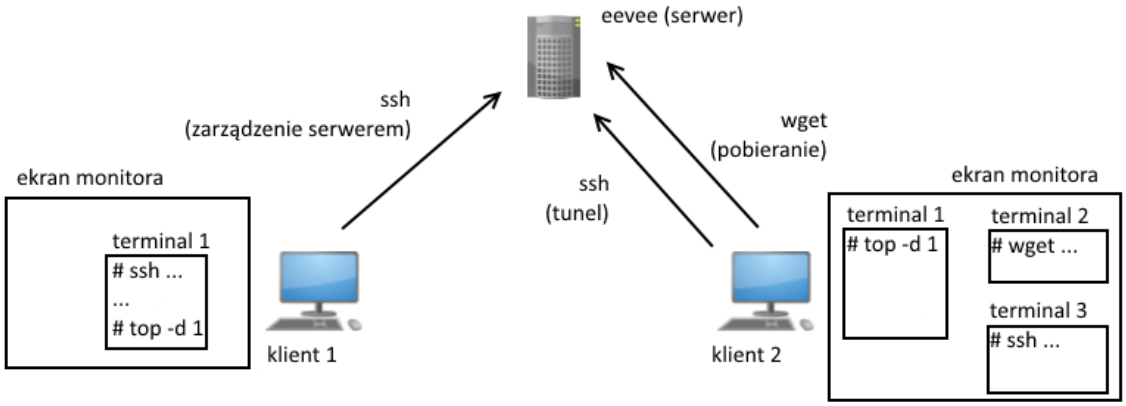
Literatura:

[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

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:**



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

3. **Wariant scenariusza**
   a. Plik do testów: ……
   b. Metody szyfrowania : ……
   c. 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 warość od 0-9
c)	3DES	nie	-c 3des -o "Compression no"
d)	3DES	tak	-c 3des -C -o "CompressionLevel X" - X warość od 0-9
e)	Arcfour	nie	-c arcfour -o "Compression no"
f)	Arcfour	tak	-c arcfour -C -o "CompressionLevel X" - X warość od 0-9
g	Cast	nie	-c cast128-cbc -o "Compression no"
h	Cast	tak	-c cast128-cbc -C -o "CompressionLevel X" - X warość 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 4 powtarzamy dla każdej ustalonej kombinacji metody szyfrowania i poziomu kompresji)

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**)

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):

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”)
b) 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 tabele i opracować statystycznie (porównanie poszczególnych serii testów)

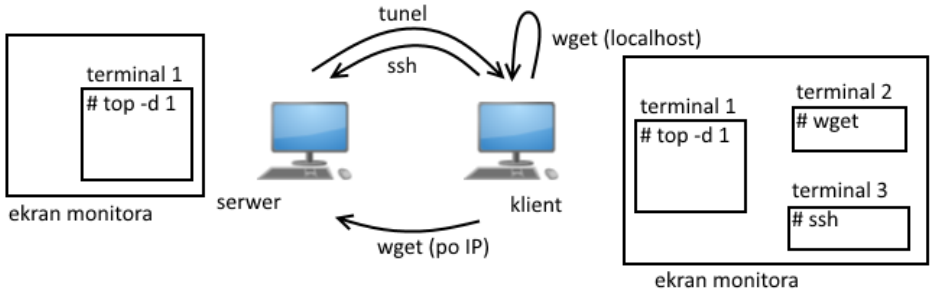
- ocenić wpływ ustawionych parametrów na obciążenie procesora i sieci na obydwu komputerach

Scenariusz nr 2: SSH z serwerem lokalnym

1. **Sprzęt i oprogramowanie:**

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

2. **Przygotowanie do wykonania ćwiczenia:**



   a. Serwer – 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`)
   b. Klient – Po zalogowaniu do systemu Windows, uruchomić wirtualną maszynę z systemem CentOS 7 i zalogować się tam na konto root (hasło root).
   c. Na serwerze
      i. 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)
      ii. uruchomić serwer http Apache komendą: `service httpd start`
      iii. wyłączyć kontrolę zabezpieczeń i uprawnień dla plików – `setenforce Permissive`
   d. Zarówno na kliencie oraz na serwerze:
      i. uruchomić usługę sshd: `service sshd start`
      ii. wyłączyć firewall: `systemctl stop firewalld` oraz `systemctl disable firewalld`
      iii. uruchomić skrypt monitorowania procesów: `top -d 1`
      (terminal 1)

3. **Wariant scenariusza**
   a. Plik do testów:
      - plik binarny 250MB / 500MB
      - plik video 200MB / 400MB
      - plik tekstowy 300MB / 600MB
   b. Wersje tunelu: ……
   c. 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 warość od 0-9
c)	3DES	nie	-c 3des -o "Compression no"
d)	3DES	tak	-c 3des -C -o "CompressionLevel X" - X warość od 0-9
e)	Arcfour	nie	-c arcfour -o "Compression no"
f)	Arcfour	tak	-c arcfour -C -o "CompressionLevel X" - X warość od 0-9
g	Cast	nie	-c cast128-cbc -o "Compression no"
h	Cast	tak	-c cast128-cbc -C -o "CompressionLevel X" - X warość od 0-9

4. 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://192.168.102.___/plik -O - >/dev/null
```

(punkty od 2 do 4 powtarzamy dla każdej ustalonej kombinacji metody szyfrowania i poziomu kompresji)

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**)

4. Uwagi do dokumentacji testów:

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

c) Zrzut ekranu w trakcie pobierania danych, na którym w terminalu nr 1 widać obciążenie procesora dla procesu sshd (serwer) i ssh (klient) (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 powinna być także widoczna wersja tunelu (polecenie ssh wraz z parametrami, zestawiane w terminalu nr 3).

5. 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 tabele i opracować statystycznie (porównanie poszczególnych serii testów)

- ocenić wpływ ustawionych parametrów na obciążenie procesora i sieci na obydwu komputerach