Bezpieczeństwo sieci komputerowych

dr inż. Andrzej Chmielewski
Wydział Informatyki
Politechniki Białostockiej

2 sprawdziany: na 8 oraz 15 wykładzie

EK1 - Zna i klasyfikuje metody bezpieczeństwa sieci

EK2 - Ma wiedzę z zakresu kryptografii i jej zastosowań

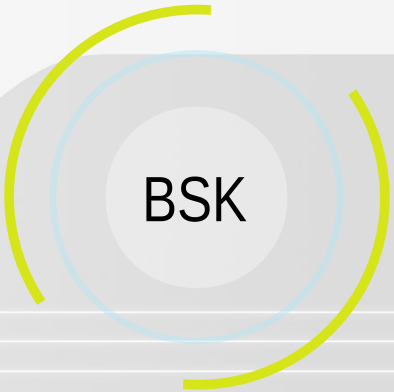
EK3 - na i stosuje zasady bezpieczeństwa sieci

Warunki zaliczenia:

- Min. 30% z każdego EK
- (50 – 60)% punktów – 3.0
- [60 – 70)% punktów – 3.5
- [70 – 80)% punktów – 4.0
- [80 – 90)% punktów – 4.5
- [90 – 100]% punktów – 5.0

Wykład prowadzony wspólnie z dr inż. E. Busłowską

- Konfiguracja serwerów: SSH, HTTPS
- Krytpografia (dr inż. E. Busłowska)
- Analiza pakietów sieciowych
- Testy penetracyjne
- Systemy IDS/IPS, rodzaje ataków sieciowych
- Bezpieczeństwo protokołów routingu

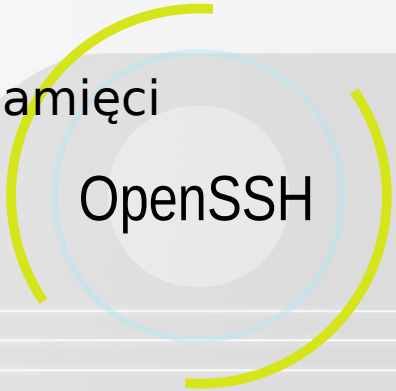


Projekt: http://www.openssh.org
Pakiety (Debian): openssh-server oraz openssh-client

Dostępny na platformy: Linux, BSD, Windows (CygWin)

SSH zapewnia szyfrowane, a zatem bezpieczne, połączenia pomiędzy urządzeniami.

Narzędzia dostarczane z pakietami:
ssh – klient
sshd – demon serwera ssh
ssh-keygen - generowanie kluczy
ssh-keyscan – odczyt hostkey z klucza
ssh-agent – przechowuje odszyfrowany klucz prywatny w pamięci

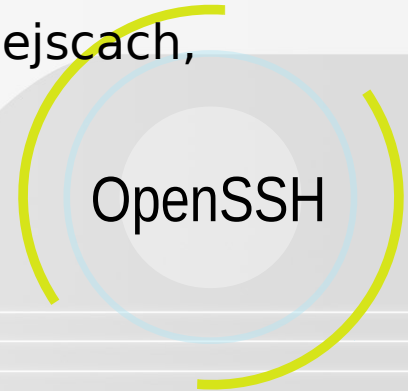


Serwer SSH domyślnie działa na porcie tcp/22.

Parametry konfiguracyjne klienta pobierane są z trzech źródeł:
- parametry przekazane z linii komend
- ~/.ssh/config
- /etc/ssh/ssh_config

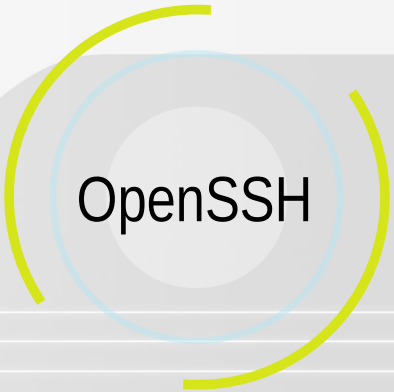
Parametry konfiguracyjne serwera pobierane są z dwóch źródeł:
- parametry przekazane z linii komend
- /etc/ssh/sshd_config

W przypadku występowania danego parametru w wielu miejscach, pobierana jest pierwsza napotkana.



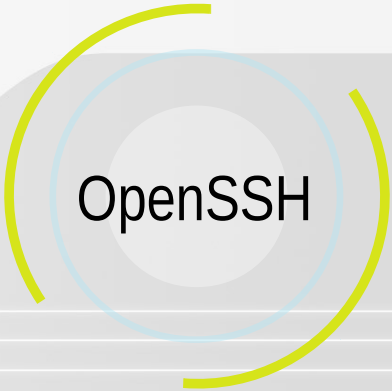
Format pliku konfiguracyjnego: /etc/ssh/ssh_config

```
Host pb 212.33.*
  User student
  Protocol 1

Host 192.* printer wi.pb.edu.pl
  Compression no
  Cipher blowfish
  Protocol 2
  Port 10022

Host *
  ForwardAgent yes
  ForwardX11 yes
  Compression yes
  Protocol 2,1
  Cipher 3des
  EscapeChar ~
```



Format pliku konfiguracyjnego: /etc/ssh/sshd_config

```
# What ports, IPs and protocols we listen for
Port 22
# Use these options to restrict which interfaces/protocols sshd will bind to
#ListenAddress ::
#ListenAddress 0.0.0.0
Protocol 2
# HostKeys for protocol version 2
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
#Privilege Separation is turned on for security
UsePrivilegeSeparation yes
X11Forwarding yes
Subsystem sftp /usr/lib/openssh/sftp-server
UsePAM yes
.......
```



ssh user@hostname


```
ssh hostname
ssh user@hostname    lub    ssh -l username hostname
ssh -o User=username hostname
```

Gdy serwer nasłuchuje na innym porcie:

```
ssh user@hostname -p port
```

OpenSSH

Dodatkowe polecenia dostępne w czasie sesji SSH, po użyciu kombinacji <Enter>~

.	Opuszczenie sesji z jednoczesnym zamknięciem tuneli
&	Opuszczenie sesji bez zamykania tuneli
C	Wejście do powłoki ssh; umożliwia np. zestawienie tunelu
<Ctrl-z>	Zawieszenie ssh i powrót do powłoki, z której ssh zostało uruchomione; polecenie `fg` umożliwia powrót do sesji ssh
#	Lista połączeń tunelujących daną sesję
?	Help (wszystkie dostępne opcje)

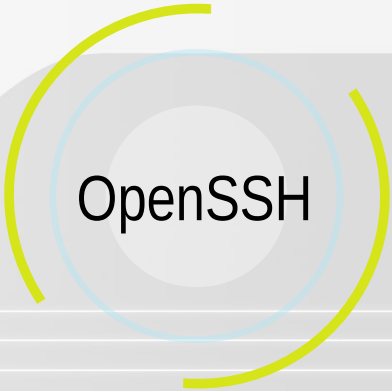
Podstawowe algorytmy: 3des, blowfish

Algorytmy używane do szyfrowania konwersacji z sshd, włączając autentykację oraz tunelowanie:

- 3des – domyślny, bezpieczny, ale znacznie obciążajacy CPU
- blowfish – szybszy

Zmiana algorytmu szyforwania na blowfish:

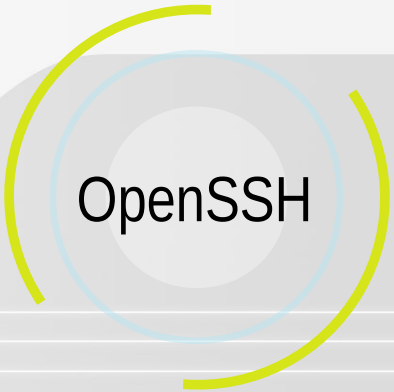
- ssh -c blowfish
- ssh -o Cipher=blowfish
- Dodanie „Cipher blowfish” do ~/.ssh/config or /etc/ssh/ssh_config



Kompresja może zmniejszyć ilość przesyłanych danych do 50% kosztem niewielkiego obciążenia CPU

Włączenie szyforwania:

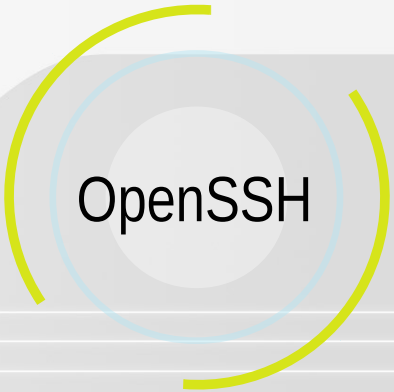
- ssh -C
- ssh -o Compression=yes
- Dodanie „Compression yes” do ~/.ssh/config or /etc/ssh/ssh_config



OpenSSH obsługuje obydwa protokoły v1 oraz v2.

SSHv1 – starszy protokół obsługujący jedynie stałą listę sposobów szyfrowania danych oraz dwie metody rozpoznawania użytkownika (klucz RSA oraz hasło)

SSHv2 – protokół obsługujący dowolne protokoły szyfrowania danych oraz cztery metody uwierzytelniania



Polecenie ssh-keygen

W przypadku SSHv1 (tylko RSA)

```
ssh-keygen -t rsa1
Pliki: ~/.ssh/id_identity[.pub]
```

SSHv2

- RSA

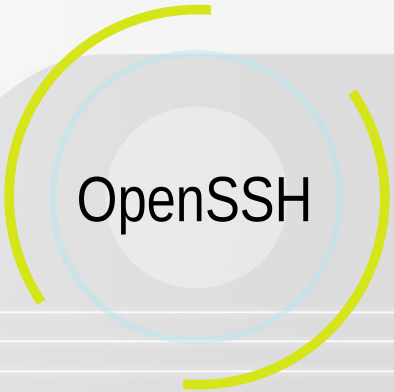
```
ssh-keygen -t rsa
Pliki: ~/.ssh/id_rsa[.pub]
Pocz?tek klucza: ssh-rsa
```

- DSA

```
ssh-keygen -t dsa
Pliki: ~/.ssh/id_dsa[.pub]
Pocz?tek klucza: ssh-dss
```

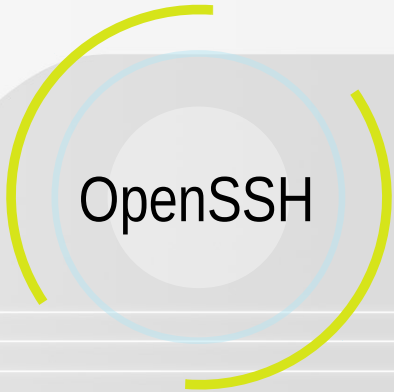
Inne opcje:

```
ssh-keygen -t ecdsa
ssh-keygen -t ed25519
```



Klucze można dodatkowo zabezpieczyć hasłem.

```
user1@user1-HP-EliteBook-8570w:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/user1/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user1/.ssh/id_rsa.
Your public key has been saved in /home/user1/.ssh/id_rsa.pub.
The key fingerprint is:
f9:e4:b8:0c:bf:15:5f:ee:48:b2:7e:9e:ee:6b:21:91 user1@user1-HP-EliteBook-8570w
The key's randomart image is:
+--[ RSA 2048]----+
|                 |
|                 |
|        .        |
|       .E        |
|      S o.  .    |
|       =.o.o     |
|     . . =.o..   |
|      + o +o+    |
|       =oo*B..   |
+-----------------+
```



Plik: known_hosts

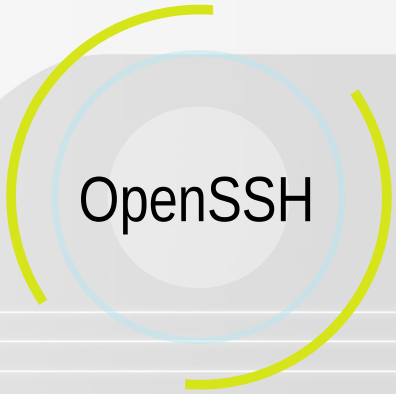
Plik zawiera klucze publiczne serwerów SSH

|1|olEWMamQmVs2Pt6EGT3cb3yz7Cg=|oIj3uJHlujVFvSiXO3/v/481DJc= ecdsa-sha2-nistp256
AAAAE2VjZHNhLXNoYTItbmlzdHAyNTYAAAAIbmlzdHAyNTYAAABBBArR0JgQWNsqQbcdx7CdZUGSTRiYXJPINHO
mKrNdmhKIa63PksGKrdJehHMEWvORMzIwQMACnOycXuGDRpdienw=

Przy pierwszym połączeniu z serwerem, pojawia się komunikat:

```
user1@localhost $ ssh -o StrictHostkeyChecking=ask drukarka
The authenticity of host 'drukarka (192.168.10.11)' can't be established.
RSA key fingerprint is 5c:6e:b2:99:3d:44:03:32:fb:e8:c1:ca:4f:cb:9e:8f.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'drukarka,192.168.10.11' (RSA) to the list of known
hosts.
```

i klucz jest zapisywany w pliku **known_hosts**.



Ochrona przez podszywającym się serwerem

Gdy zostanie wykryta niezgodność klucza, połączenie nie może być nawiązane

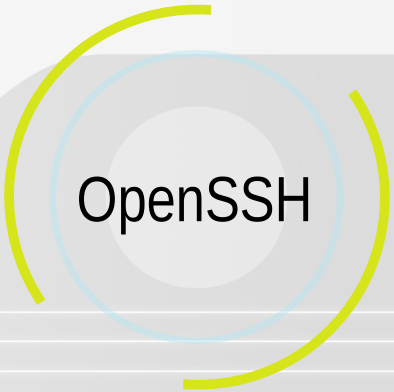
```
user1@localhost $ ssh -o StrictHostkeyChecking=yes drukarka
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@
WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!
@
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
5c:6e:b2:99:3d:44:03:32:fb:e8:c1:ca:4f:cb:9e:8f.
Please contact your system administrator.
Add correct host key in /home/user1/.ssh/known_hosts to get rid of this message.
Offending key in /export/home/user1/.ssh/known_hosts:45
RSA host key for gate has changed and you have requested strict checking.
Host key verification failed.
```

OpenSSH

Autentykacja z kluczem publicznym

Autentykacja z użyciem klucza jest bezpieczniejsza w porównaniu z hasłem.

- klucz dużo trudniej jest złamać, np. za pomocą metody *brute force*
- w przypadku włamania na serwer SSH, hacker będzie miał dostęp jedynie do klucza publicznego użytkownika, a nie do zaszyfrowanego hasła
- korzystając z wielu serwerów SSH można używać tego samego hasła do kluczy
- w przypadku współdzielenia konta, nie trzeba przekazywać wszystkim klientom hasła; każdy użytkownik jest autentykowany za pomocą własnego klucza



Plik: authorized_keys

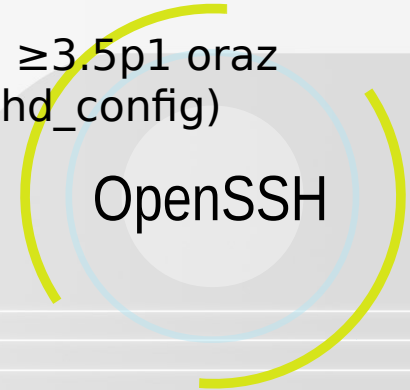
Plik służący do przechowywania kluczy publicznych użytkowników, którzy w ten sposób mogą się łączyć bez hasła

Lokalizacja: ~/.ssh/authorized_keys
UWAGA: „PasswordAuthentication No” w pliku konfiguracyjnym oznacza, że autentykacja jest możliwa tylko za pomocą klucza

Każda wpis składa się z 3 części:
<opcje><klucz><komentarz>

Klucz jest kluczem publicznym klienta (.pub)
Komentarz jest opcjonalny.
Opcje:

- environment="PATH=/home/user1" (opcja dostępna w openssh ≥3.5p1 oraz ustawionym kluczem PermitUserEnvironment=yes w /etc/ssh/sshd_config)
- command="/home/user1/start.sh"
- from="*.pb.edu.pl,aragorn.pb.bialystok.pl"
- no-port-forwarding
- no-X11-forwarding
- permitopen="host:port"
- no-agent-forwarding
- no-pty



Logowanie w sesji SSH za pomocą klucza, bez podawania hasła

Wygenerowanie kluczy (np. RSA2):

```
$ ssh-keygen -t rsa
```

Jeśli nie zostanie ustawione hasło do klucza, autentykacja w sesji SSH będzie zupełnie pozbawiona interakcji (bez hasła).

Przesłanie na zdalny serwer:

```
$ ssh-copy-id <user>@<hostname>
```

lub

```
cat ~/.ssh/id_rsa.pub | ssh <user>@<hostname>
'mkdir ~/.ssh; cat >> ~/.ssh/authorized_keys'
```

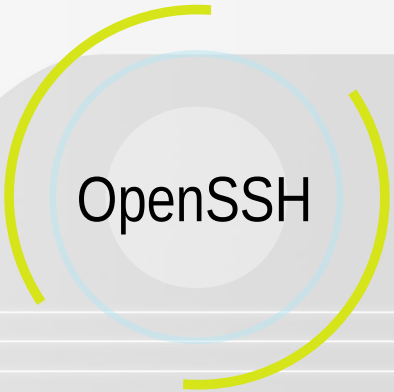
Klucz zostanie umieszczony na serwerze <hostname> w pliku ~<user>/.ssh/authorized_keys

UWAGA: Dostęp po pliku `authorized_keys` powinien być bardzo ograniczony (prawa 0666, aby inni użytkownicy nie mogli zmieniać jego zawartości.

scp – narzędzie do szyfrowanego przesyłania plików pomiędzy urządzeniami

scp <plik_lokalny> <user>@<host>:<sciezka>

scp <user>@<host>:<sciezka_do_pliku> <plik_lub_katalog_lokalny>



Polecenia na zdalnym urządzeniu

Podając jako ostatni parametr polecenie, będzie ono wykonane i automatycznie nastąpi zamknięcie sesji

ssh <user>@<hostname> <polecenie>

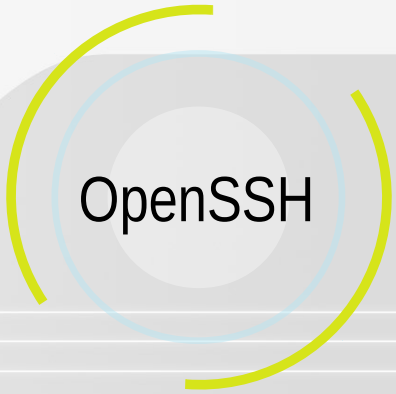
```
$ ssh student@10.0.0.45 ls -al
```

Wykonywanie sekwencji komend:

```
$ ssh<user>@<hostname> $(<lista_komend.txt)
```

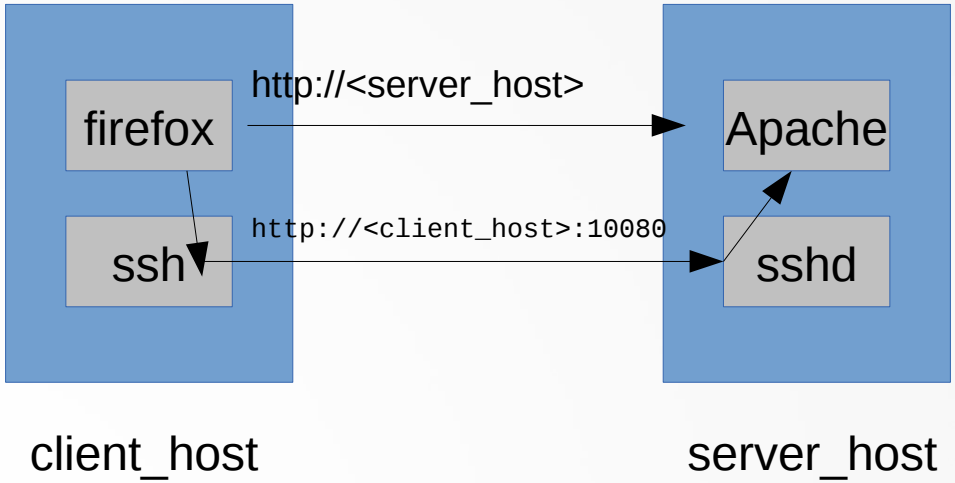
lub

```
ssh <user>@<hostname> "`cat lista_komend.txt`"
```



Przekierowanie lokalne portów

Tunelowanie nieszyfrowanych połączeń w szyfrowanym tunelu ssh



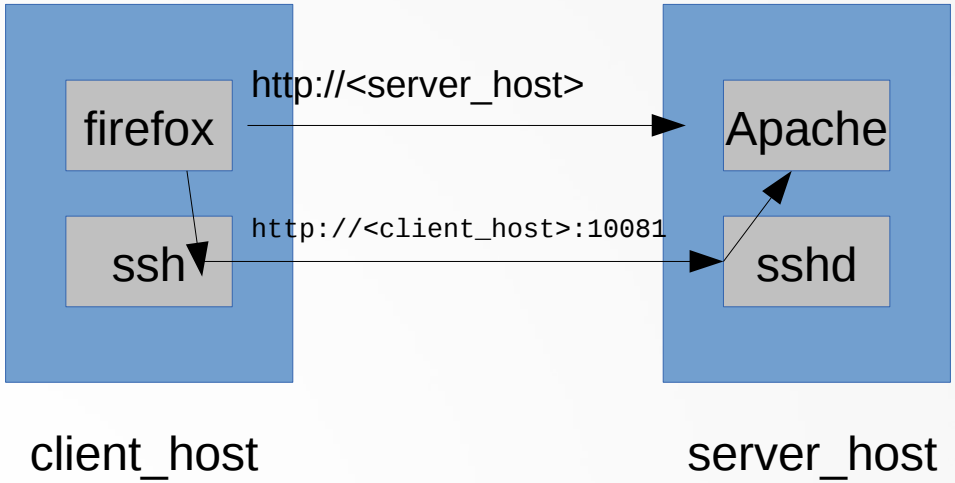
client_host:

```
$ ssh -N -L10080:localhost:80 <server_host>
```

```
http::<localhost>:10080
```

OpenSSH

Tunelowanie nieszyfrowanych połączeń w szyfrowanym tunelu ssh



server_host:

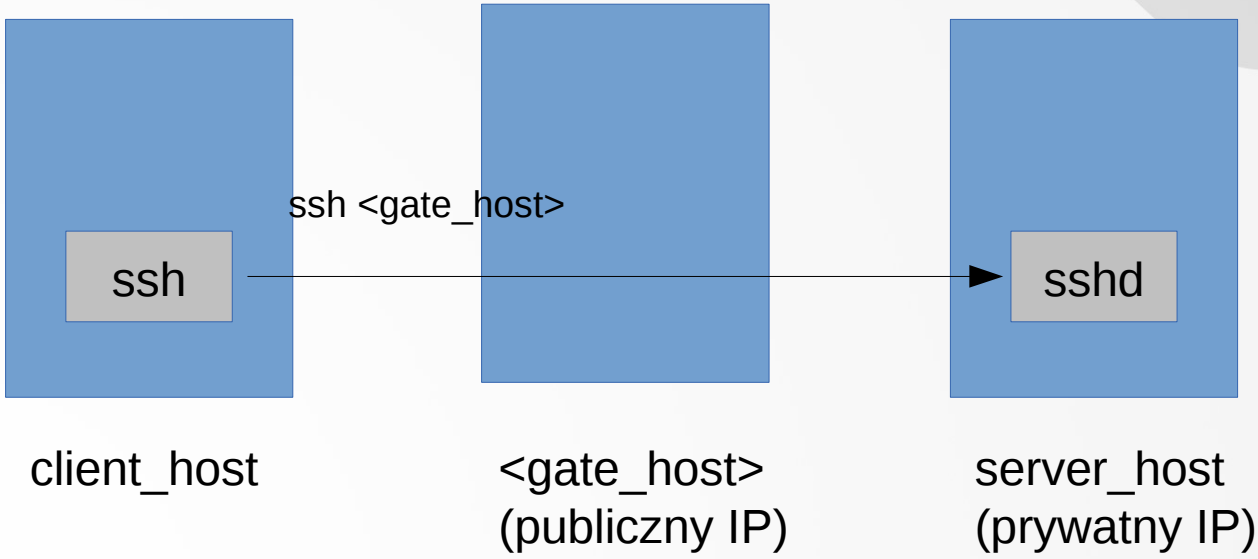
```
$ ssh -R -L10081:<client_host>:80 <server_host>
```

client_host:

```
http::<localhost>:10081
```

OpenSSH

Tunelowanie do serwerów SSH poza maskaradą



client_host:

```
$ ssh -t <gate_host> ssh <server_host>

$ ssh <gate_host>
```

OpenSSH

HTTPS = HTTP + SSL (Secure Socket Layer)

Czasami niezbędne jest odblokowanie modułu

```
# a2enmod ssl
```

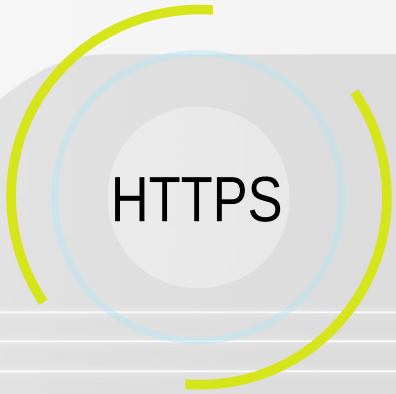
Generowanie certyfikatu

```
# openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout
/etc/apache2/ssl/bsk.key -out /etc/apache2/ssl/bsk.crt

Country Name (2 letter code) [AU]:PL
State or Province Name (full name) [Some-State]:Podlaskie
Locality Name (eg, city) []:Bialystok
Organization Name (eg, company) [Internet Widgits Pty
Ltd]:bsk
Organizational Unit Name (eg, section) []:FIR
Common Name (e.g. server FQDN or YOUR name) []:bsk.pl
Email Address []:admin@bsk.pl
```

HTTPS = HTTP + SSL (Secure Socket Layer)

Dodanie pliku konfiguracyjnego:
/etc/apache2/sites-available/bsk.pl.conf

```
<VirtualHost *:443>
ServerAdmin admin@bsk.pl
ServerName bsk.pl:443
ServerAlias www.bsk.pl
DocumentRoot /var/www/bsk.pl/public_html
ErrorLog ${APACHE_LOG_DIR}/error.log
CustomLog ${APACHE_LOG_DIR}/access.log combined
SSLEngine on
SSLCertificateFile /etc/apache2/ssl/bsk.crt
SSLCertificateKeyFile /etc/apache2/ssl/bsk.key
</VirtualHost>
```



Uruchmomienie serwera SSL

```
# a2ensite default-ssl
```

HTTPS = HTTP + SSL (Secure Socket Layer)

Wymuszenie HTTPS

W pliku konfiguracyjnym:

```
Redirect permanent / https://www.bsk.pl
```

Lub dodać do pliku konfiguracyjnego (np. .htaccess)

```
RewriteEngine On
RewriteCond %{SERVER_PORT} 80
RewriteRule ^(.*)$ https://www.bsk.pl/$1 [R,L]
```