evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

PSK3-13

Název školy:	Vyšší odborná škola a Střední průmyslová škola, Božetěchova 3
Autor:	Ing. Marek Nožka
Anotace:	Pokročilé vlastnosti SSH
Vzdělávací oblast:	Informační a komunikační technologie
Předmět:	Počítačové sítě a komunikační technika (PSK)
Tematická oblast:	Operační systém Linux/Unix
Výsledky vzdělávání:	Žák se přihlásí ke vzdálenému uživatelskému účtu, přenese na vzdálený počítač soubory pomocí SSH, vytvoří SSH tulen
Klíčová slova:	Linux, Unix, SSH tunel, SSH klíč
Druh učebního materiálu:	Online vzdělávací materiál
Typ vzdělávání:	Střední vzdělávání, 4. ročník, technické lyceum
Ověřeno:	VOŠ a SPŠE Olomouc; Třída: 4L
Zdroj:	Vlastní poznámky, Vilém Vychodil: Linux Příručka českého uživatele

Secure Shell II

Pohodlíčko

Aby nebylo nutné stále znovu a znovu zadávat heslo, je možné ověřovat identitu pomocí klíče (souboru, uloženého na disku). Celý proces autentizace je založen na asymetrické kryptografii. Nejprve je nutné vygenerovat si **keyPair**. To udělá příkaz `ssh-keygen`.

```
user@pc:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/m/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/m/.ssh/id_rsa.
Your public key has been saved in /home/m/.ssh/id_rsa.pub.
The key fingerprint is:
61:07:e5:02:3f:2c:68:61:2e:fd:ba:e7:b4:45:6a:25 user@pc
The key's randomart image is:
+--[ RSA 2048]----+
|     o . ...     |
|    + o + o      |
|   . = . B o     |
|    o . o =      |
|       E S       |
|      . =        |
|     . + .       |
|      +.o        |
|     .oo         |
+-----------------+
```

Program se nejprve ptá na umístění klíče a poté na passphrase. Vzniknou dva soubory -- privátním klíčem a veřejným klíčem (veřejný klíč má příponu .pub). Veřejný klíč ne nutné přenést na server a uložit ho do souboru `~/.ssh/authorized_keys` vzdáleného uživatele. To je možné provést *složitě jednoduchými příkazy*:

```
user@pc:~$ scp ~/.ssh/id_rsa.pub uzivatel@pocitac.nekde.daleko.tld:/tmp
user@pc:~$ ssh uzivatel@nekde.daleko.tld


uzivatel@daleko:~$ mkdir ~/.ssh
uzivatel@daleko:~$ cat /tmp/id_rsa.pub >>~/.ssh/authorized_keys
```

... nebo *jednoduše složitým příkazem* (viz SSH jako roura):

```
user@pc:~$ ssh <~/.ssh/id_rsa.pub uzivatel@nekde.daleko.tld "mkdir ~/.ssh; cat >>~/.ssh/authorized_keys"
```

V souboru `~/.ssh/authorized_keys` může být klíčů i víc. Na každém řádku jeden.

Pokud jste při tvorbě klíče zadali prázdné passphrase je možné hlásit se bez hesla. Důrazně ale doporučuji passphrase zadat. Lze totiž použít program `ssh-agent` a passphrase zadávat pouze jednou.

Úkol

- Vysvětlete význam výše uvedených příkazů a popište, co který z nich dělá.

Provádění vzdálených příkazů

Programu `ssh` můžeme jako parametr předat příkaz, který se má na vzdáleném počítači spustit;

```
ssh uzivatel@nekde.daleko.tld prikaz
```

Vytvoří se SSH spojení, na vzdáleném počítači se spustí `příkaz` a jeho výstup se zobrazí na lokálním terminálu. Jakmile je `příkaz` ukončen SSH spojení se ukončí.

Například pokud bychom chtěli zjistit čas na vzdáleném počítači:

```
user@pc:~$ ssh uzivatel@nekde.daleko.tld date
Ne led  5 19:47:09 CET 2014
user@pc:~$
```

Pokud je potřebné spustit více příkazů je nutné je oddělit středníkem. Protože ale středník má nabýt svého významu až na vzdáleném počítači je nutné aby se choval jako běžný znak a nikoli metaznak. Proto dáme příkaz do uvozovek.

```
user@pc:~$ ssh uzivatel@nekde.daleko.tld "hostname; echo --------; date;"
nekde.daleko.tld
--------
Ne led  5 19:51:07 CET 2014
user@pc:~$
```

SSH jako roura

SSH lze použít jak rouru pro datový proud mezi dvěma systémy.

- To co program `ssh` na svůj standardní vstup předá na

standardní vstup programu spuštěného na vzdáleném počítači.

- To co program na vzdáleném počítači pošle na svůj standardní výstup, to se objeví i na standardním výstupu programu `ssh`.

Například:

```
user@pc: ~$ ssh <lokalni_soubor uzivatel@nekde.dalelo.tld "mkdir ~/test; cat >>~/test/vzdaleny_soubor"
```

1. Na vstup programu `ssh` přijde soubor `lokalni_soubor`,
2. `ssh` ho přenese na vstup vzdáleného programu `cat`,
3. vzdálený program `cat` uloží obsah lokálního souboru na konec vzdáleného souboru,
4. program `mkdir` přenos neovlivní, protože ten nečte standardní vstup.

SSH tunel

Pomocí SSH je možné vytvářet šifrované kanály:

```
$ ssh -N -L 8008:localhost:80 borec@hroch.spseol.cz
```

Tento příkaz provede spojení uživatele `borec` ne server `hroch.spseol.cz` a propojí vzdálený port 80 s lokálním portem 8008. Nenechte se zmást hostitelským jménem `localhost` to se bere z pohledu vzdáleného počítače a uvedený příkaz má stejný efekt jako

```
$ ssh -N -L 8008:hroch.spseol.cz:80 borec@hroch.spseol.cz
```

Parametr `-N` říká, že se nemá spouště žádný příkaz, tedy ani shell a jen se vytvoří tunel.

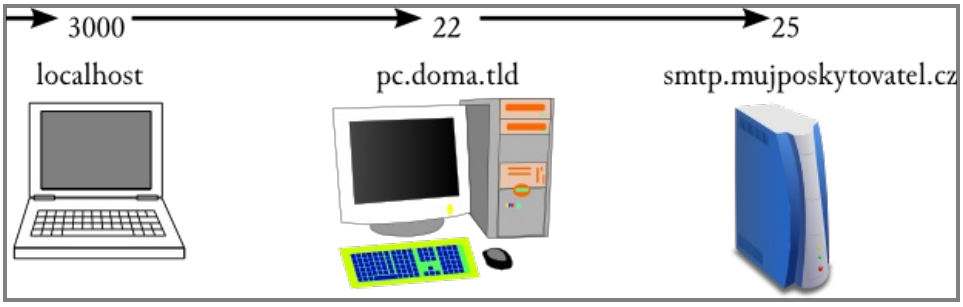
V tuto chvíli můžete zadat do prohlížeče adresu http://localhost:8008/~nozka/hroch a můžete se přesvědčit, že požadavky, které přichází na lokální port 8008 jsou přeposílány na vzdálený port 80. Vtip je v tom, že toto spojení je šifrované.

K čemu je to dobré?

Představme si následující modelovou situaci: Jste připojení k Internetu z některé veřejné sítě například ve škole nebo v kavárně. Z bezpečnostních důvodů je zde ale zakázán port 25 a vy potřebujete nutně odeslat e-mail. (Důrazně žádám všechny šťouraly, kteří chtějí namítat, že mám použít web-mail, aby nic neříkaly.)

```
student@skola: $ ssh -N -L 3000:smtp.mujposkytovatel.cz:25 borec@pc.doma.tld
```

Vytvoří se spojení mezi lokálním počítačem a `pc.doma.tld`. Lokální port 3000 bude přesměrován na port 25 počítače `smtp.mujposkytovatel.cz` skrze `pc.doma.tld`. Nyní je možné odeslat e-mail na lokální port 3000.



K čemu je to dobré? II

Pro větší názornost ještě jeden příklad: Máte malou domácí síť ve které je několik počítačů a vy na ně potřebujete vzdáleně přistupovat (například ze školy), ale máte jen jednu veřejnou IP adresu a proto je z Internetu dostupný je jeden počítač (`pc.doma.tld`).

Pokud se chci připojit pomocí SSH (port `22`) na počítač, který má ve vnitřní síti adresu `192.168.68.1` spustím příkaz:

```
student@skola:$ ssh -N -L 4000:192.168.68.1:22 borec@pc.doma.tld
```

Poté je možné připojit se na tento počítač na lokálním portu `4000` příkazem

```
student@skola:$ ssh -p 4000 uzivatel@localhost
```

Pokud bych požadoval, připojit se na vzdálenou plochu Windows stanice (port `3389`) na počítači s vnitřní IP adresou `192.168.68.2` spustím příkaz:

```
student@skola:$ ssh -N -L 5000:192.168.68.2:3389 borec@pc.doma.tld
```

Na vzdálenou plochu se nyní můžu připojit na lokálním portu `5000` například takto:

```
student@skola:$ rdesktop -u Administrator localhost:5000
```

Pokud by bylo potřebné provést opačnou akci, tedy že vzdálený port přesměrujeme na lokální, použijeme parametr `-R`. Dalším zajímavým parametrem v souvislosti s tunelováním je `-f`, který před spuštěním příkazu umístí SSH na pozadí.