

# Služby, webový server a ssh

Eliška Jégrová

17. 11. 2025

# Obsah

Úvod do problematiky

Firewall

Webový server

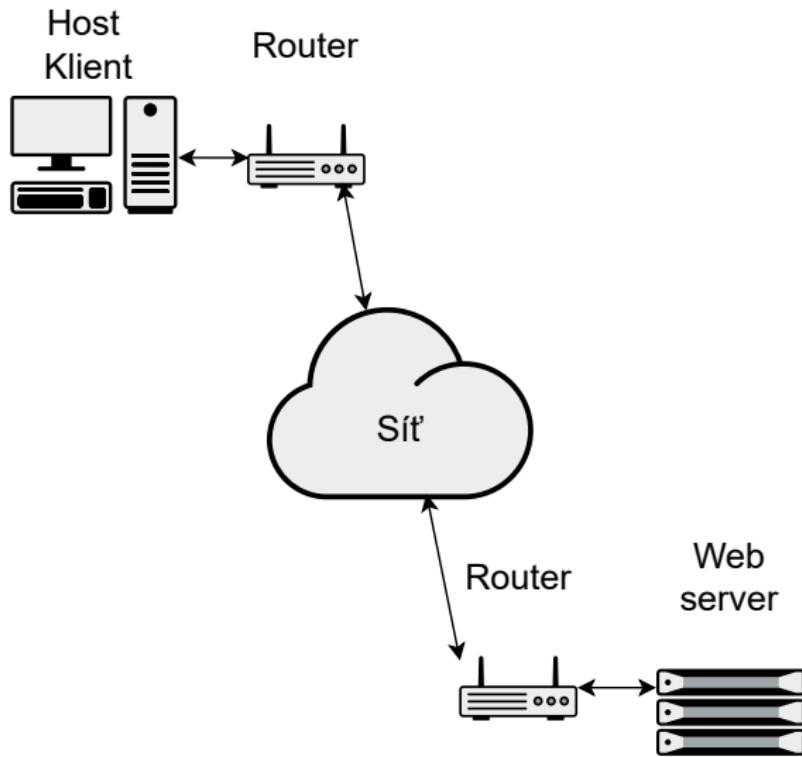
SSH – vzdálená administrace

Cron

# Co je počítačová síť

- ▶ Pro komunikaci mezi počítači je potřeba **sítě**.
- ▶ Umožňuje sdílení dat, provoz síťových služeb.
- ▶ Základní pojmy:
  - ▶ **host** – zařízení v síti,
  - ▶ **server** – poskytuje službu,
  - ▶ **klient** – službu využívá.

## Sít' – host, router, server



# IP adresa

- ▶ IP adresa = identifikátor zařízení v síti.
- ▶ Dva hlavní typy:
  - ▶ **IPv4** (32 bitů): např. 192.168.1.10
  - ▶ **IPv6** (128 bitů): např. 2001:db8::1
- ▶ IP adresa umožní směrovat data na správný počítač.
- ▶ Často se rozlišuje:
  - ▶ **privátní adresa** – v lokální síti,
  - ▶ **veřejná adresa** – viditelná na internetu.
- ▶ Příkaz:
  - ▶ `$ ip address`
  - ▶ `$ hostname -I` – vypíše jen IP adresy hosta

# DNS – Domain Name System

- ▶ Lidé si snadněji pamatují názvy než čísla.
- ▶ **DNS** převádí názvy (např. example.com) na IP adresy.
- ▶ Funguje podobně jako telefonní seznam.
- ▶ Typické příkazy:
  - ▶ \$ ping seznam.cz
  - ▶ \$ dig seznam.cz
- ▶ Bez DNS by web nefungoval – museli bychom psát IP adresy ručně.

# Porty a služby

- ▶ Jeden server může poskytovat mnoho služeb zároveň.
- ▶ Aby se služby nepletly, používají **porty**.
- ▶ Port = „číslo služby“.
- ▶ Příklady běžných portů:
  - ▶ 22 – SSH
  - ▶ 80 – HTTP (web)
  - ▶ 443 – HTTPS (web zabezpečený)
- ▶ Kombinace: **IP adresa + port** = konkrétní služba.
- ▶ Např.:

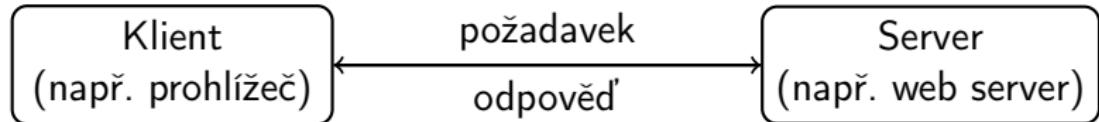
192.168.1.20:80 → webový server

192.168.1.20:22 → SSH

192.168.1.20:25 → emailový server

## Jak spolu vše souvisí

- ▶ Uživatel zadá do prohlížeče název: `seznam.cz`.
- ▶ DNS jej přeloží na IP adresu serveru.
- ▶ Prohlížeč se spojí na port 80/443.
- ▶ Server vrátí odpověď – webovou stránku.
- ▶ SSH funguje podobně, jen na portu 22 a bez webových dat.



# Firewall

- ▶ Firewall je síťové zařízení nebo software, který **povoluje nebo blokuje připojení** podle nastavených pravidel.
- ▶ Slouží k ochraně serveru nebo počítače před nežádoucí komunikací.
- ▶ V nových instalacích je firewall obvykle **zapnutý a defaultně blokuje většinu portů**.
- ▶ Na Fedora se používá **firewalld** (dynamický firewall s podporou zón).
- ▶ Příkaz pro okamžité povolení:  

```
# firewall-cmd --add-service=http
```
- ▶ Příkaz pro trvalé povolení po restartu:  

```
# firewall-cmd --permanent --add-service=http
```

# Služba a démon

- ▶ **Služba (*service*):**
  - ▶ něco, co lze zapnout a vypnout
  - ▶ má stav (běží/neběží), konfiguraci, logy a závislosti,
  - ▶ příklady: webový server, firewall, tiskový server.
- ▶ **Démon (*daemon*):**
  - ▶ dlouho běžící proces „na pozadí“, typicky spuštěný službou,
  - ▶ nespouští se z shellu
  - ▶ často má v názvu d na konci: httpd, firewalld, sshd.
- ▶ Na Fedora služby spravuje **systemd** (PID 1); ovládáme je příkazem **systemctl**.

## Webový server httpd

- ▶ Použijeme **Apache HTTP Server** (balíček `httpd`).
- ▶ Běží jako **démon** – dlouho běžící proces na pozadí.
- ▶ Naslouchá typicky na portu **80** (HTTP), případně **443** (HTTPS).
- ▶ Webový server běží jako služba `httpd.service`.
- ▶ Existují alternativy (např. `nginx`), ale princip správy služby je podobný.

# Instalace webového serveru

- ▶ Instalace na Fedoře:

```
# dnf install httpd
```

- ▶ Balíček obsahuje binárku serveru, konfiguraci i dokumentaci.
- ▶ Důležité konfigurační soubory:

- ▶ /etc/httpd/conf/httpd.conf – hlavní konfigurace,
- ▶ /etc/httpd/conf.d/\*.conf – další části nastavení,
- ▶ /etc/httpd/conf.modules.d/\*.conf – seznam modulů.

## Start, stop, stav služby

- ▶ Systémové služby se spravují jako **root** (např. \$ sudo -i).
- ▶ Základní příkazy:

```
# systemctl start httpd      - spuštění služby (a démona)  
# systemctl stop httpd       - zastavení služby  
# systemctl status httpd    - informace o běhu
```

- ▶ status ukáže:
  - ▶ zda služba běží,
  - ▶ PID a počet procesů httpd,
  - ▶ poslední hlášky z logu.

## Test webového serveru

- ▶ Po systemctl start httpd by server měl naslouchat na portu 80.

- ▶ Test z příkazové řádky:

```
$ curl http://192.168.1.10
```

- ▶ Textový prohlížeč (ve virtuálce):

```
# dnf install links
```

```
$ links http://192.168.1.10
```

- ▶ V prohlížeči zadejte do adresního řádku:

```
http://192.168.1.10
```

- ▶ Při správně nastavené síti a povolené službě http ve firewallu je testovací stránka dostupná i z jiného počítače.

# Automatické spuštění po startu

- ▶ Na Fedora se nově nainstalované služby **nespouští automaticky**.
- ▶ Nastavení spuštění při startu systému:

```
# systemctl enable httpd    # zapnout při startu  
# systemctl disable httpd   # nezapínat při startu
```

- ▶ Ověření:
  - ▶ restart systému,
  - ▶ znovu systemctl status httpd.

# Změna konfigurace a reload

- ▶ Konfigurační soubory (např.

/etc/httpd/conf.d/welcome.conf) se načítají při startu služby.

- ▶ Po změně konfigurace je potřeba server informovat:

```
# systemctl reload httpd    # načtení nové konfigurace  
# systemctl restart httpd   # úplný restart služby
```

- ▶ **reload:**

- ▶ běžící spojení zůstanou zachována,
- ▶ nová spojení už používají nové nastavení.

- ▶ **restart:**

- ▶ služba se ukončí a znova spustí,
- ▶ aktuální spojení se zavřou.

# DocumentRoot a obsah webu

- ▶ **DocumentRoot** = adresář, odkud server servíruje soubory.
- ▶ Nastavení je v /etc/httpd/conf/httpd.conf
- ▶ Výchozí DocumentRoot na Fedoře (Apache):  
/var/www/html/.
- ▶ Vytvoř testovací soubor:  

```
# echo "Hello, World" > /var/www/html/hello.txt
```
- ▶ Otevři v prohlížeči:  

```
http://192.168.1.10/hello.txt
```
- ▶ Změny v **obsahu** DocumentRootu se projeví hned – není potřeba reload.

# Logy webového serveru

- ▶ Pro hledání chyb a kontrolu běhu slouží logy.
- ▶ Základní přehled: `systemctl status httpd`.
- ▶ Detailnější log přes journalctl:  
`$ journalctl -u httpd`
- ▶ V logu najdeš např.:
  - ▶ start/stop služby,
  - ▶ chybová hlášení,
  - ▶ informaci, na jakém portu server naslouchá.

# Signály a démoni

- ▶ Každá služba může reagovat na signály trochu jinak.
- ▶ **systemd unit** popisuje, jak se služba spouští, zastavuje a reloaduje.
- ▶ Jak to zjistit:
  - ▶ zobrazit definici služby:

```
$ systemctl cat httpd
$ systemctl cat sshd
```
- ▶ Z toho pak vyčteš:
  - ▶ jestli reload jen načte konfiguraci,
  - ▶ jak „elegantně“ se služba ukončuje,
  - ▶ jaké signály daný démon pro tyto akce používá.

# Samostatná práce 1 – služby a webový server

1. Ověř, zda je na virtuálce nainstalovaný balíček httpd. Pokud ne, nainstaluj ho.
2. Zjisti, jestli se služba httpd spouští automaticky po startu systému. Pokud ne, nastav, aby se spouštěla.
3. Vytvoř jednoduchou stránku status.html v DocumentRootu (/var/www/html/), která bude obsahovat:
  - ▶ jméno serveru (např. Virtuálka),
  - ▶ tvoje jméno,
  - ▶ k čemu tento server slouží.
4. V prohlížeči na hostitelském systému otevři:  
`http://IP_TVE_VIRTUALKY/status.html` a ověř, že se stránka načte.

## Samostatná práce 2 – úpravy httpd.conf

1. V `/var/www/html` vytvoř soubor:
  - ▶ `uvod.html` – krátký text o serveru,
2. V `httpd.conf` nastav direktivu:  
`DirectoryIndex uvod.html index.html.`  
Ověř v prohlížeči, že se při otevření `http://IP_SERVERU/` načte právě `uvod.html`.
3. Bonus: Přidej vlastní chybovou stránku v `httpd.conf`:  
`ErrorDocument 404 /chyba-404.html.`  
Vytvoř `/var/www/html/chyba-404.html` a ověř, že se zobrazí při zadání neexistující URL (např. `testuji.html`).
4. Bonus: Zkontroluj, že v logu není chyba konfigurace.

# SSH – vzdálená administrace

- ▶ **SSH** = *Secure Shell* – bezpečné vzdálené přihlášení na server.
- ▶ Správa serveru „jako přes terminál“, ale přes síť / internet.
- ▶ Veškerá komunikace (včetně hesla) je **šifrovaná**.
- ▶ Strany spojení:
  - ▶ **server** – služba sshd běžící na vzdáleném stroji,
  - ▶ **klient** – program ssh nebo grafický klient (např. PuTTY).
- ▶ Typické použití:
  - ▶ přihlášení na server,
  - ▶ spouštění příkazů, správa služeb, editace konfigurace.

# Instalace SSH serveru (OpenSSH)

- ▶ Na Fedora se používá implementace **OpenSSH**.
- ▶ Instalace serveru:

```
$ sudo dnf install openssh-server
```

- ▶ Spuštění služby sshd:

```
$ sudo systemctl start sshd
```

- ▶ Otevření ve firewallu (port 22):

```
$ sudo firewall-cmd --add-service=ssh
```

- ▶ Po nastavení můžeš z jiného stroje používat SSH přihlášení.

## První připojení – localhost

- ▶ Nejdřív si SSH vyzkoušíme „sama na sebe“:  

```
$ ssh localhost
```
- ▶ localhost = „tento počítač“ (bez ohledu na IP).
- ▶ Při prvním připojení:
  - ▶ SSH vypíše **otisk (fingerprint)** klíče serveru,
  - ▶ zeptá se, zda důvěřuješ tomuto serveru (yes/no),
  - ▶ po potvrzení a zadání hesla jsi přihlášená.
- ▶ Otisk se dá na serveru zjistit (správce posílá přes jiný kanál):  

```
$ sudo ssh-keygen -l -f /etc/ssh/ssh_host_ecdsa_key.pub
```
- ▶ Po přihlášení vidíš shell na vzdáleném stroji, prompt bude např. petr@localhost:~\$.

## Soubor known\_hosts a změna klíče

- ▶ Po prvním připojení si klient SSH uloží klíč serveru do:  
~/.ssh/known\_hosts
- ▶ Při dalším připojení kontroluje, zda se klíč nezměnil.
- ▶ Když se klíč změní, SSH varuje (může to znamenat útok typu „man-in-the-middle“).
- ▶ Postup:
  - ▶ ověř u administrátora, že klíč byl skutečně změněn,
  - ▶ pokud je vše v pořádku, smaž starý řádek z known\_hosts,
  - ▶ připoj se znovu a nový otisk znovu potvrd'.

# Připojení na vzdálený server

- ▶ Typický tvar příkazu:

```
$ ssh uzivatel@192.168.122.133
```

- ▶ Uživatelské jméno bude jiné než na „tvém“ počítači.

## Jméno serveru

- ▶ Pro orientaci je dobré mít na serveru **smysluplné jméno**.
- ▶ Zobrazení jména:

```
$ hostname  
$ hostnamectl
```

- ▶ Dočasná změna (do restartu):

```
$ sudo hostname virtualka
```

- ▶ Trvalé nastavení je v souboru /etc/hostname

- ▶ Permanentní změna:

```
$ sudo hostnamectl set-hostname virtualka
```

# SSH klíče – motivace

- ▶ Přihlašování **heslem**:
  - ▶ jednoduché na začátek,
  - ▶ méně pohodlné a méně bezpečné (hesla se dají hádat).
- ▶ Přihlašování pomocí **SSH klíčů**:
  - ▶ používá se dvojice **veřejný + soukromý** klíč,
  - ▶ veřejný klíč je na serveru, soukromý máš jen ty,
  - ▶ po nastavení se už k serveru přihlašuješ bez zadávání hesla k účtu (jen případné heslo ke klíči).
- ▶ Výhoda:
  - ▶ bezpečnější než hesla,
  - ▶ pohodlnější při častém připojování (skripty, Git, automatizace).

# SSH klíče – vytvoření a použití

- ▶ Vytvoření klíčů (na svém počítači):

```
$ ssh-keygen
```

- ▶ Výchozí umístění:

- ▶ soukromý klíč: ~/.ssh/id\_rsa
  - ▶ veřejný klíč: ~/.ssh/id\_rsa.pub

- ▶ Veřejný klíč se kopíruje na server do  
~/.ssh/authorized\_keys.

- ▶ Pohodlný způsob: ssh-copy-id:

```
$ ssh-copy-id uzivatel@server
```

- ▶ Po úspěchu se můžeš přihlásit:

```
$ ssh uzivatel@server
```

- ▶ Pokud má klíč nastavené heslo, zadáš jen to.

# Nastavení SSHD

- ▶ Konfigurace serveru sshd:
  - ▶ hlavní soubor: /etc/ssh/sshd\_config,
  - ▶ další části: /etc/ssh/sshd\_config.d/.
- ▶ Lze měnit např.:
  - ▶ port (např. z 22 na 2222),
  - ▶ povolené metody přihlášení (hesla/klíče),
  - ▶ verze protokolu, povolené šifrovací algoritmy atd.
- ▶ Při změně portu nezapomeň na:
  - ▶ firewall,
  - ▶ parametr -p na straně klienta.

## Samostatná práce 3 – SSH a hostname

1. Zkontroluj, zda je nainstalovaný a spuštěný **SSH server**:
  - ▶ pokud neběží, spusť ho a nastav automatické spuštění po startu.
2. Nastav na virtuálce hostname, např. `web-vm1` (`hostnamectl`)  
Pomocí odhlášení/přihlášení ověř, že se jméno objeví v promptu.
3. Z hostitelského systému se přihlas na server a spusť tam příkaz:
  - ▶ `hostname`; `whoami`; `uptime`
4. Najdi ve svém domovském adresáři soubor `.ssh/known_hosts` a podívej se, jak je v něm tvůj server uložený (IP, jméno, typ klíče).

## Samostatná práce 4 (BONUS) – kopírování přes SSH (scp)

1. Na virtuálce v domovském adresáři vytvoř soubor poznamky.txt s libovolným obsahem.
2. Z hostitelského systému zkopíruj soubor z virtuálky k sobě:
  - ▶ `scp uzivatel@IP_VIRTUALKY:~/poznamky.txt .`Ověř, že se soubor objevil v aktuálním adresáři na hostiteli.
3. Vytvoř na hostitelském systému soubor readme.txt a zkopíruj ho na virtuálku do domovského adresáře:
  - ▶ `scp readme.txt uzivatel@IP_VIRTUALKY:~/`Na závěr se na virtuálku přihlas přes SSH a ověř, že tam oba soubory opravdu jsou.

# Co je cron

- ▶ **cron** je služba, která spouští úlohy v definovaných časech.
- ▶ Úlohy se zapisují do souborů zvaných *crontab*.
- ▶ Umožňuje plánovat opakující se příkazy:
  - ▶ Zálohování databáze
  - ▶ Rotace logů
  - ▶ Kontrola stavu služby
  - ▶ Čištění dočasných souborů
  - ▶ Spouštění skriptů pro sběr dat v určitých intervalech

# Syntaxe crontab

- ▶ Každý řádek má 5 polí pro čas + příkaz:

```
* * * * *   příkaz
| | | | |
| | | | |-- den v týdnu (0-6)
| | | |-- měsíc (1-12)
| | |-- den v měsíci (1-31)
| |-- hodina (0-23)
|-- minuta (0-59)
```

```
$ crontab -e
```

```
$ crontab -l
```

## Příklad cron úlohy

- ▶ Spustit skript každý den ve 2:30 ráno

```
30 2 * * * /home/uzivatel/backup.sh
```

– spustí se 'backup.sh' každý den v 2:30

- ▶ Spustit skript každou minutu

```
* * * * * mplayer
```

```
/usr/share/sounds/gnome/default/alerts/hum.ogg
```

- ▶ Pro přesměrování výstupu a chyb:

```
30 2 * * * /path/to/script.sh >
```

```
/var/log/muj.log 2>&1
```

## Samostatná práce 5 – cron

- ▶ Vytvoř skript `~/check_mem.sh`, který vypíše aktuální datum a obsazení paměti:

```
#!/bin/bash  
date  
free -m
```

- ▶ Nastav mu spustitelný příznak:

```
$ chmod +x ~/check_disk.sh
```

- ▶ Pomocí crontab -e naplánuj, aby se skript spouštěl každých 5 minut a výstup zapisoval do souboru `~/cron_mem.log`.

Můžeš využít: <https://crontab.guru/>

- ▶ Počkej, až cron úlohu několikrát spustí, a zkontroluj obsah `cron_mem.log`. Ověř, že jsou v něm různé časy a aktuální využití disku.