

# Otázky na zkoušku z NTI/UI v1.1

Jakub Koněřza

10. března 2025

## Abstrakt

Remember there's no such thing as a small act of kindness. Every act creates a ripple effect with no logical end.

## 1 SSH

### 1.1 Co to je?

- Secure Shell
- Jedná se o protokol, který slouží k bezpečné komunikaci mezi dvěma zařízeními v síti.
- Nejčastěji se používá pro vzdálený přístup k serverům nebo jiným počítačům.
- Náhrada za telnet.
- Poskytuje šifrování přenášených dat přes nedůvěryhodnou síť, jako je například internet.
- Využívá se pro: zprostředkování přístupu k příkazovému řádku, zabezpečenému tunelování síťového provozu, kopírování souborů, spouštění GUI aplikací pomocí X11 forwarding

### 1.2 Jak se konfiguruje?

- V Unixu (včetně Linuxu) má v současné době dominantní postavení balík OpenSSH.
- Na serveru je potřeba `sshd` (SSH server daemon)
- Konfigurace daemonu se mění v souboru `/etc/ssh/sshd_config`
- Po úpravě a uložení konfigurace se hodí příkaz `sshd -t`, který vypíše případné chyby a vyvaluje se zabít SSH daemona.
- Pokud nejsou syntaktické chyby tak se může restartovat `sshd`.

– `systemctl restart sshd`

### 1.3 Konfigurační soubor + Jaká je běžná konfigurace?

Direktiva	Popis	Výchozí hodnota
Port [číslo]	Určuje číslo portu, na kterém SSH server poslouchá.	22
PermitRootLogin [yes/no]	Určuje, zda je povoleno přihlášení jako root.	yes
PasswordAuthentication [yes/no]	Určuje, zda je povoleno přihlášení pomocí hesla.	yes
PubkeyAuthentication [yes/no]	Určuje, zda je povoleno přihlášení pomocí veřejných klíčů.	yes
X11Forwarding [yes/no]	Povoluje nebo zakazuje přesměrování X11.	no
ListenAddress [adresa]	Určuje IP adresu, na které SSH server poslouchá.	0.0.0.0 (všechny dostupné adresy)
MaxAuthTries [číslo]	Určuje maximální počet pokusů o přihlášení.	6
LoginGraceTime [čas]	Určuje maximální dobu, po kterou může uživatel zůstat nepřihlášený předtím, než je spojení ukončeno.	120 sekund
UsePAM [yes/no]	Povoluje použití PAM (Pluggable Authentication Modules) pro autentizaci.	no
AllowUsers [uživatelé]	Určuje, které uživatele mají povoleno přihlášení.	(žádná výchozí hodnota)
AllowGroups [skupiny]	Určuje, které skupiny mají povoleno přihlášení.	(žádná výchozí hodnota)
ClientAliveInterval [čas]	Určuje interval (v sekundách), po kterém server odešle ping klientovi, aby zjistil, zda je stále aktivní.	0 (vypnuto)
ClientAliveCountMax [číslo]	Určuje maximální počet pingů, které mohou být odeslány bez odpovědi, než je spojení ukončeno.	3

Tabulka 1: Konfigurace v souboru `sshd_config`

### 1.4 Na jakém běží portu?

- Standardně naslouchá na portu TCP/22.
- Může být změněno na jiné číslo.

## 2 SSH Klíče

### 2.1 K čemu slouží?

- SSH klíče slouží k autentizaci uživatele při připojování pomocí SSH (Secure Shell).

### 2.2 Co umožňují?

- Automatické přihlášení bez nutnosti zadávání hesla (tzv. passwordless login).
- Je jednodušší používat delší a složitější "klíč".
- Přihlašování na více serverů jedním veřejným klíčem.

## 2.3 Jak k nim přijdu?

- Generováním klíčového páru (privátní a veřejný klíč) pomocí příkazu `ssh-keygen`:

```
ssh-keygen -t rsa -b 4096 -C "email@example.com"
```

- Po vygenerování jsou klíče obvykle uloženy v adresáři `~/.ssh/`:
  - Privátní klíč: `id_rsa`
  - Veřejný klíč: `id_rsa.pub`
- Veřejný klíč je poté potřeba nahrát na server do souboru `~/.ssh/authorized_keys`.

## 2.4 Proč je chci používat?

- Zabezpečení: Klíče jsou bezpečnější než hesla, protože jsou delší a složitější.
- Pohodlí: Nemusím psát heslo.

## 2.5 Kde je používám?

- Připojení k linuxovým serverům přes SSH.
- Přístup k cloudovým službám (např. AWS, Google Cloud, Azure).
- Správa verzovacích systémů jako Git (např. GitHub, GitLab, Bitbucket).
- Šifrování komunikace při práci s databázemi (např. přístup k MySQL přes SSH tunel).
- Pro zabezpečení přenosu dat v rámci interních sítí.

# 3 Instalace Software na Linuxovém Stroji

## 3.1 Jaké jsou možnosti?

- **Balíčkovací systémy (např. apt, yum, dnf, pacman):** Používají se pro instalaci, aktualizaci a správu softwaru z oficiálních a neoficiálních repozitářů.
- **Manuální instalace z Tarballu:** Zdrojové kódy nebo binární soubory se stahují a rozbalují ručně.
- **Univerzální formáty (AppImage, Snap, Flatpak):** Poskytují předem sestavené balíčky, které fungují na většině distribucí.
- **Kompilace ze zdrojového kódu:** Software je sestavován přímo na cílovém stroji, často pomocí nástrojů jako make.

## 3.2 Jaké jsou úskalí?

- **Tarball:**
  - Neexistuje standardizovaný způsob instalace nebo odinstalace.
  - Tar je jenom způsob "zazipování". Ovšem nemusí to být zkomprimované, samostatný "tar" nepodporuje komprimaci, používá se navíc například gzip nebo bzip2 a poté je koncovka souboru .tar.gz/.tar.bz2.
  - Většinou obsahuje zdrojové soubory a musí se zkompilevat.
  - Jednoduše řečeno: je to jako kdybys stáhl github repo v .zip.
  - Uživatel musí ručně řešit závislosti (!)
- **Balíky a repozitáře:**
  - Závislost na konkrétní distribuci a jejím ekosystému.
  - Může docházet ke konfliktům mezi balíčky.
- **AppImage:**
  - Omezené možnosti integrace se systémem (např. automatické aktualizace).
  - Velikost balíčku může být větší kvůli zahrnutým závislostem.
- **Snapd:**
  - Vyšší nároky na zdroje (běží na pozadí jako služba).
  - Závislost na Canonicalu jako poskytovateli infrastruktury.
- **Flatpak:**
  - Vyžaduje běhové prostředí (runtime), což může zvětšovat velikost instalace.
  - Menší počet aplikací ve srovnání s klasickými balíčkovacími systémy.

## 3.3 Proč je chci používat?

- **Balíčkovací systémy:** Automatizují správu softwaru a závislostí, jsou spolehlivé a snadno použitelné.
- **Tarball:** Flexibilita při instalaci specifických verzí nebo vlastních úpravách.
- **Univerzální formáty:** Umožňují instalaci softwaru na více distribucích bez nutnosti adaptace.
- **Kompilace:** Maximální kontrola nad optimalizací a vlastnostmi softwaru.

## 3.4 Kde je používám?

- **Balíčkovací systémy:** Nejčastěji na serverech a pracovních stanicích pro základní software.
- **Tarball:** Na vývojářských strojích nebo při testování experimentálního softwaru.
- **Univerzální formáty:** Na desktopových stanicích pro aplikace, které nejsou dostupné v repozitářích.
- **Kompilace:** Na specializovaných strojích, kde je vyžadována specifická konfigurace.

## 4 Firewall

### 4.1 K čemu jsou?

- Firewall slouží k zabezpečení síťového provozu.
- Chrání síťové uzly před neoprávněným přístupem zvenčí.
- Filtruje příchozí a odchozí datové pakety podle nastavených pravidel.
- Zabraňuje útokům, jako je skenování portů, DDoS útoky nebo neoprávněné připojení.
- Může být hardwarový (specializované zařízení) nebo softwarový (program běžící na operačním systému).




### 4.2 Co jsou ty *tables*?

- V kontextu Linuxu se jedná o nástroj `iptables` (starší) nebo `nftables` (novější).
- Slouží ke konfiguraci pravidel pro filtrování a manipulaci s datovými pakety.
- `iptables` využívá tabulky (*tables*), kde každá má specifické účely:
  - **filter**: Nejčastěji používaná tabulka pro filtrování paketů.
  - **nat**: Používá se pro překlad síťových adres (NAPT, SNAT, DNAT).
  - **mangle**: Slouží k úpravám paketů (např. změna hlavičky).
  - **raw**: Používá se k deaktivaci některých sledovacích funkcí.
  - **security**: Slouží k označování paketů pro bezpečnostní moduly (např. SELinux).
- Každá tabulka obsahuje *řetězce* (chains), např. `INPUT`, `OUTPUT`, `FORWARD`, do kterých se zapisují konkrétní pravidla.

### 4.3 Jak se dají nakonfigurovat?

- Pomocí příkazového řádku:
  - Pro `iptables`: příkazy jako `iptables -A`, `iptables -D`, `iptables -L`.
  - Pro `nftables`: příkazy pomocí nástroje `nft`, který pracuje s konfigurací v jazyce podobném skriptům.
- Automatizace:
  - Konfigurační soubory: Např. `/etc/iptables/rules.v4` nebo `/etc/nftables.conf`.
  - Systémové služby: `iptables-persistent`, `nftables.service`.
- Grafické nástroje:
  - `GFW`: Jednoduché GUI pro firewall na Linuxu.
  - `Firewalld`: Dynamický firewall pro moderní distribuce.

## 4.4 Ideální použití

- Ochrana serverů:
  - Povolení přístupu pouze na potřebné porty (např. 22/tcp pro SSH).
  - Omezení přístupu na základě IP adres (whitelist/blacklist).
- Ochrana osobních počítačů:
  - Blokování podezřelých příchozích spojení.
  - Omezení přístupu k aplikacím, které nepotřebují síťovou komunikaci.
- Zajištění bezpečnosti v podnikových sítích:
  - Nastavení pravidel pro komunikaci mezi jednotlivými subnety.
  - Implementace demilitarizované zóny (DMZ).   

## 4.5 Porty vs. služby

- **Porty:**
  - Jedná se o čísla, která označují specifické služby na síťových vrstvách (např. TCP/UDP).
  - Například: 80/tcp (HTTP), 443/tcp (HTTPS), 22/tcp (SSH).
- **Služby:**
  - Konkrétní aplikace nebo procesy běžící na serveru (např. Apache, Nginx, SSHD).
  - Každá služba může být přidružena k jednomu nebo více portům.
- **Rozdíl:**
  - Port je technický prostředek pro identifikaci komunikace na síti.
  - Služba je aplikace, která tento port využívá.

## 5 Nastavení sítě

### 5.1 IPv4 a IPv6

- **IPv4** (Internet Protocol version 4) je čtvrtá verze IP protokolu, která používá 32bitové adresy, což umožňuje až přibližně 4,3 miliardy unikátních adres.
  - Formát adresy: čtyři desítková čísla oddělená tečkami (např. 192.168.0.1).
  - Omezený adresní prostor vedl k zavedení NAT (Network Address Translation) a vývoji IPv6.
- **IPv6** (Internet Protocol version 6) je modernější protokol, který používá 128bitové adresy, což poskytuje mnohem větší adresní prostor.
  - Formát adresy: osm hexadecimálních bloků oddělených dvojtečkami
  - (např. 2001:0db8:85a3:0000:0000:8a2e:0370:7334).
  - Nabízí lepší podporu pro multicast, bezpečnostní funkce (IPsec) a jednodušší automatickou konfiguraci.

## 5.2 Jak nastavím IPv4 nebo IPv6 na serveru

- Pro konfiguraci IP adres se používají různé nástroje v závislosti na distribuci Linuxu (např. `ifconfig`, `ip` nebo konfigurace přes soubor).

- **IPv4 konfigurace pomocí příkazu `ip`:**

```
ip addr add 192.168.1.100/24 dev eth0
ip link set dev eth0 up
```

- **IPv6 konfigurace pomocí příkazu `ip`:**

```
ip -6 addr add 2001:db8::1/64 dev eth0
ip link set dev eth0 up
```

- Konfigurace může být také trvalá. V Debianu/Ubuntu upravte soubor `/etc/network/interfaces`:

```
auto eth0
iface eth0 inet static
    address 192.168.1.100
    netmask 255.255.255.0
    gateway 192.168.1.1

iface eth0 inet6 static
    address 2001:db8::1
    netmask 64
    gateway 2001:db8::ff
```

## 5.3 Jak řešit firewall pro IPv4 a IPv6

- Firewall spravuje příchozí a odchozí provoz podle definovaných pravidel.

- **Pro IPv4:**

- Používá se `iptables`.
- Příklad: Blokování konkrétní IP adresy:

```
iptables -A INPUT -s 192.168.1.200 -j DROP
```

- **Pro IPv6:**

- Používá se `ip6tables`.
- Příklad: Povolení příchozího ICMPv6 (ping):

```
ip6tables -A INPUT -p icmpv6 --icmpv6-type echo-request -j ACCEPT
```

- Moderní distribuce používají nástroj `nftables`, který podporuje IPv4 i IPv6:

```
nft add rule ip filter input ip saddr 192.168.1.0/24 accept
nft add rule ip6 filter input ip6 saddr 2001:db8::/32 accept
```

## 5.4 Lze nějakou verzi vynechat?

- **Záleží na potřebách a infrastruktuře sítě.**
- IPv4 může být vynechán v moderních sítích, které plně podporují IPv6. Příkladem jsou některé cloudové platformy.
- Naopak IPv6 lze vynechat ve starších sítích nebo sítích bez požadavku na větší adresní prostor.
- Nejčastější je hybridní přístup, kdy jsou obě verze protokolu využívány současně (dual-stack).

## 6 Logovací Soubory

### 6.1 Kde je najdu a co v nich najdu?

- **Umístění:**
  - Logovací soubory se obvykle nacházejí v adresáři `/var/log`.
  - Méně běžné logy mohou být v uživatelských adresářích nebo v aplikačních složkách (např. `7.local/share/`).
- **Obsah:**
  - Obsahují informace o chybách, událostech a provozu systému či aplikací.
  - Příklady: systémové chyby, přihlašovací pokusy, zprávy od démonů, informace o aktualizacích.

### 6.2 Příklad aspoň 4 souborů a proč si všímám zrovna jich?

- `/var/log/syslog` nebo `/var/log/messages`
  - Obsahuje obecné systémové zprávy.
  - Důležité pro monitorování běhu systému a diagnostiku problémů.
- `/var/log/auth.log`
  - Zaznamenává události související s autentizací (např. SSH přihlášení).
  - Klíčové pro sledování bezpečnostních incidentů.
- `/var/log/dpkg.log`
  - Obsahuje informace o instalaci, aktualizaci a odinstalaci balíčků (Debian/Ubuntu).
  - Užitečné při hledání chyb souvisejících s balíčkovým systémem.
- `/var/log/kern.log`
  - Záznamy od jádra systému.
  - Pomáhá při diagnostice problémů hardwaru a nízkourovňových chyb.

### 6.3 Jak probíhá rotace logů?

- **Rotace logů:**
  - Proces pravidelného archivování a mazání starých logovacích souborů, aby nezabíraly místo na disku.
  - Typicky zajišťováno službou `logrotate`.
- **Mechanismus:**



- Původní logovací soubor je přejmenován (např. `auth.log` na `auth.log.1`).
- Vytvoří se nový prázdný soubor, do kterého systém zapisuje.
- Staré logy mohou být komprimovány (např. pomocí `gzip`).

- **Konfigurace:**

- Nachází se typicky v `/etc/logrotate.conf` nebo v souborech v `/etc/logrotate.d/`.
- Může zahrnovat parametry jako maximální počet archivů, frekvence rotace nebo komprese.

## 6.4 Co je vzdálené logování?

- **Definice:**

- Vzdálené logování je proces, kdy logovací zprávy ze systému jsou odesílány na jiný server.
- Využívá se pro centralizaci logů, zvýšení bezpečnosti a lepší auditování.

- **Mechanismus:**

- Typicky realizováno pomocí protokolu `syslog` (UDP port 514) nebo jeho vylepšené verze `rsyslog`/`syslog-ng`.
- Konfigurace se provádí v `/etc/rsyslog.conf` nebo `/etc/syslog.conf`.

- **Výhody:**

- Logy nejsou uloženy lokálně, což brání jejich smazání při útoku.
- Snazší monitorování více systémů z jednoho místa.

## 7 FTP

### 7.1 Na co je?

- **FTP (File Transfer Protocol)** slouží k přenosu souborů mezi klientem a serverem po síti.
- Umožňuje efektivní správu souborů na vzdálených serverech, včetně:
  - Nahrávání souborů (upload).
  - Stahování souborů (download).
  - Mazání souborů.
  - Změnu práv souborů a adresářů.
- Používá se v případech, kdy je třeba přístup k velkému množství dat nebo jejich správa na serveru.

### 7.2 Komu slouží?

- **Administrátorům serverů:** Pro správu a údržbu souborového systému na serverech.
- **Vývojářům:** K nahrávání aplikací, webových stránek a dalších dat na servery.
- **Koncovým uživatelům:** Například pro stahování nebo nahrávání dat v rámci veřejných FTP serverů.

## 7.3 Jak se instaluje?

- **Na Linuxu:**

- Instalace balíčku FTP serveru (např. `vsftpd`, `proftpd` nebo `pure-ftpd`):  
`sudo apt install vsftpd`

- **Na Windows:**

- Instalace pomocí IIS (Internet Information Services) nebo alternativních aplikací.

- **Na macOS:**

- Použití třetích stran jako `pure-ftpd` nebo `proftpd`.

## 7.4 Jak se konfiguruje?

- Konfigurace závisí na používaném serveru (např. `vsftpd`):

- **Úprava konfiguračního souboru:** Obvykle `/etc/vsftpd.conf`.
- **Klíčové parametry:**
  - \* `anonymous_enable=YES/NO`: Povolení anonymního přístupu.
  - \* `local_enable=YES`: Povolení přístupu pro lokální uživatele.
  - \* `write_enable=YES`: Povolení zápisu na server.
  - \* `chroot_local_user=YES`: Uzamčení uživatelů do jejich domovských adresářů.
- Restart služby po změně konfigurace:

```
sudo systemctl restart vsftpd
```

## 7.5 Jaké je běžné nastavení?

- Povolení přístupu pro lokální uživatele (`local_enable=YES`).
- Zakázání anonymního přístupu (`anonymous_enable=NO`).
- Uzamčení uživatelů do jejich domovských adresářů (`chroot_local_user=YES`).
- Povolení zápisu (`write_enable=YES`) jen pro konkrétní uživatele nebo skupiny.
- Nastavení šifrování pomocí SSL/TLS (`ssl_enable=YES`).
- Omezení přístupu z vybraných IP adres (pomocí `/etc/hosts.allow` nebo `/etc/hosts.deny`).

# 8 Samba

## 8.1 Na co je?

Samba je softwarový balík, který umožňuje sdílení souborů a tiskáren mezi různými operačními systémy, především mezi:

- **Linux/Unix** servery a **Windows** klienty,

**Hlavní účely Samby:**

- Sdílení složek a souborů v síti.
- Sdílení tiskáren.
- Autentizace a řízení přístupu pomocí Windows domén.
- Integrace Linux/Unix systémů do prostředí Active Directory.

## 8.2 Komu slouží?

Samba je určena pro:

- Administrátory, kteří chtějí propojit Linux/Unix servery s Windows klienty.
- Organizace, které potřebují centralizovat správu uživatelů a sdílených prostředků.
- Domácí uživatele, kteří chtějí sdílet soubory mezi Linuxem a Windows.
- Školy, firmy a instituce, které používají smíšené prostředí operačních systémů.

## 8.3 Jak se instaluje?

Instalace Samby se liší podle distribuce, ale obecně se postupuje takto:

- Na **Debian/Ubuntu**:

```
sudo apt update
sudo apt install samba
```

- Na **Red Hat/Fedora/CentOS**:

```
sudo dnf install samba samba-client
```

- Ověření instalace:

```
smbd --version
```

## 8.4 Jak se konfiguruje?

Konfigurace Samby probíhá úpravou hlavního konfiguračního souboru `/etc/samba/smb.conf`:

- Otevřete soubor:

```
sudo nano /etc/samba/smb.conf
```

- Přidejte sdílený adresář. Například:

```
[shared]
path = /srv/samba/shared
read only = no
browsable = yes
```

- Restartujte Samba služby:

```
sudo systemctl restart smbd nmbd
```

- Přidejte uživatele Samby:

```
sudo smbpasswd -a uživatel
```

## 8.5 Jaké je běžné nastavení?

Typická nastavení Samby zahrnují:

- **Sdílené složky:**

```
[shared]
path = /srv/samba/shared
read only = no
browsable = yes
valid users = uživatel
```

- **Práce v síti:**

- Nastavení pracovního názvu skupiny:

```
workgroup = WORKGROUP
```

- Povolení autentizace:

```
security = user
```

- **Práva a uživatelé:**

- Vytvoření Samby uživatelů:

```
sudo smbpasswd -a uživatel
```

- Nastavení přístupových práv:

```
chmod 770 /srv/samba/shared
```

## 9 NFS Share

### 9.1 Na co je?

- NFS (Network File System) je protokol určený pro sdílení souborových systémů přes síť.
- Umožňuje přístup ke vzdáleným souborům tak, jako by byly uloženy na lokálním disku.
- Používá se pro centralizaci dat, například v podnicích nebo laboratořích, kde je potřeba sdílet soubory mezi více uživateli či servery.

### 9.2 Komu slouží?

- Administrátorům serverů k usnadnění správy a centralizace dat.
- Uživatelským stanicím v sítích, kde je nutný přístup ke společným datům.
- Systémům, které vyžadují síťově sdílené souborové systémy, například v clusterech nebo výpočetních farmách.

### 9.3 Jak se instaluje?

- Na straně serveru:
  - V Linuxu se instaluje balíček `nfs-kernel-server`.
  - Příklad instalace na Debianu/Ubuntu: `sudo apt-get install nfs-kernel-server`.
- Na straně klienta:
  - Je nutné nainstalovat balíček `nfs-common`.
  - Příklad instalace na Debianu/Ubuntu: `sudo apt-get install nfs-common`.

### 9.4 Jak se konfiguruje?

- Na serveru:
  - Sdílené adresáře se definují v souboru `/etc/exports`.
  - Syntaxe pro sdílení adresáře: `/cesta/k/adresari hostname_nebo_IP(opce)`.
  - Příklad: `/srv/data 192.168.1.0/24(rw,sync,no_subtree_check)`.
  - Po úpravě konfiguračního souboru je nutné restartovat NFS server: `sudo systemctl restart nfs-kernel-server`.
- Na klientovi:
  - Připojení se provádí příkazem `mount`.
  - Syntaxe: `sudo mount -t nfs server:/cesta/k/adresari /místní/mountpoint`.
  - Připojení lze automatizovat přidáním záznamu do souboru `/etc/fstab`.

### 9.5 Jaké je běžné nastavení?

- Na serveru:
  - Povolené čtení i zápis: `rw`.
  - Synchronní operace: `sync` (zajišťuje konzistenci dat).
  - Zákaz kontrolování podstromů: `no_subtree_check`.
  - Sdílení jen pro určité sítě nebo IP adresy.
- Na klientovi:
  - Použití parametrů jako `soft`, `hard`, `timeo=n`, aby se optimalizovala stabilita a výkon.
  - Přidání sdíleného adresáře do `/etc/fstab` pro automatické připojení.

## 10 Web Server Apache

### 10.1 Co to je?

- **Apache HTTP Server** (zkracován jako Apache) je otevřený, bezplatný a velmi populární webový server.
- Byl vytvořen v roce 1995 a je spravován nadací Apache Software Foundation.
- Umožňuje hostovat a poskytovat webové stránky a aplikace na internetu nebo intranetu.
- Podporuje modulární architekturu, což znamená, že může být rozšířen o další funkce pomocí modulů.
- Je dostupný pro různé operační systémy včetně Linuxu, Windows a macOS.

## 10.2 Proč je to dobré?

- **Otevřený kód:** Uživatelé mohou prohlížet, upravovat a přizpůsobovat kód podle potřeby.
- **Modulární architektura:** Podporuje mnoho modulů pro rozšíření funkčnosti, například pro autentizaci, zabezpečení nebo cachování.
- **Spolehlivost a stabilita:** Dlouholetá historie a široká uživatelská základna znamená, že server je důkazem kvality a odolnosti.
- **Bezpečnost:** Podporuje šifrování SSL/TLS, což umožňuje provoz zabezpečených webových stránek.
- **Kompatibilita:** Dobře spolupracuje s různými databázemi a skriptovacími jazyky, jako je PHP, Python nebo Perl.

## 10.3 Jak ho použít?

- **Instalace:** Na Linuxu lze Apache obvykle nainstalovat pomocí balíčkovacích systémů, jako jsou `apt` nebo `yum`:
  - Příklad pro Debian/Ubuntu: `sudo apt-get install apache2`
  - Příklad pro CentOS/RedHat: `sudo yum install httpd`
- **Konfigurace:** Konfigurační soubory jsou obvykle umístěny v `/etc/apache2/` (Debian/Ubuntu) nebo `/etc/httpd/` (CentOS/RedHat). Hlavní konfigurační soubor se nazývá `httpd.conf` nebo `apache2.conf`.
- **Spuštění a zastavení:** Použijte příkazy jako `sudo systemctl start apache2` nebo `sudo systemctl stop apache2` pro správu služby.
- **Nasazení webových stránek:** Umístěte soubory webové aplikace do kořenového adresáře serveru, obvykle `/var/www/html`.

## 10.4 Co od něj čekat?

- **Vysoký výkon:** Apache je optimalizovaný pro provoz v prostředích s vysokou zátěží, a to díky výkonným modulům jako `mpm_event`.
- **Bezpečnostní funkce :** Možnost nastavovat pravidla pro zabezpečení pomocí modulů jako `mod_security` nebo `mod_ssl`.
- **Flexibilita :** Podpora mnoha protokolů a rozšíření zajišťuje možnost snadné integrace s jinými systémy.
- **Komunita a podpora:** Rozsáhlá uživatelská komunita a dostupná dokumentace pomáhají řešit problémy a implementovat pokročilé funkce.

# 11 Web Server Nginx

## 11.1 Co to je?

- **Nginx ("Engine-X")** je vysoce výkonný, open-source webový server a reverzní proxy server.
- Původně byl vyvinut v roce 2004 Igorem Sysoevem pro řešení problému "C10k", což je výzva obsluhovat současně 10 000 a více klientských připojení.
- Kromě webového serveru podporuje také funkce load balancingu, cacheování a streamování.
- Nginx je navržen s ohledem na vysokou propustnost a nízké využití zdrojů, což jej činí vhodným pro moderní webové aplikace.

## 11.2 Proč je to dobré?

- **Vysoká výkonnost:** Nginx je asynchronní a událostmi řízený, což umožňuje efektivně obsluhovat velké množství současných připojení.
- **Nízké systémové nároky:** Díky své architektuře spotřebovává méně paměti a CPU ve srovnání s tradičními servery, jako je Apache.
- **Flexibilita:** Nabízí široké možnosti konfigurace a podporuje moderní webové technologie, jako je HTTP/2 a TLS.
- **Široké použití:** Může fungovat jako reverzní proxy server, vyrovnávač zátěže, HTTP cache a server pro statické soubory.

## 11.3 Jak ho použít?

- **Instalace:** Nginx lze nainstalovat pomocí balíčkovacích systémů (např. `apt`, `yum`) nebo zkompilovat ze zdrojového kódu.
- **Základní konfigurace:**
  - Konfigurační soubor se obvykle nachází v `/etc/nginx/nginx.conf`.
  - Konfigurace se dělí na `http`, `server` a `location` bloky pro specifické části aplikace.
- **Spuštění a správa:**
  - Použití příkazů `sudo systemctl start nginx`, `stop`, `restart`, atd.
  - Kontrola konfigurace pomocí `nginx -t`.
- **Rozšíření:** Nginx podporuje moduly pro rozšíření funkcionalit, například `ngx_http_rewrite_module` pro přepisování URL.

## 11.4 Co od něj čekat?

- **Rychlá odezva:** Nginx je ideální pro aplikace s vysokou návštěvností díky své efektivitě.
- **Stabilita:** I při velké zátěži zůstává stabilní a spolehlivý.
- **Škálovatelnost:** Může být snadno použit ve škálovaných architekturách jako load balancer.
- **Podpora moderních protokolů:** Nginx podporuje protokoly jako HTTP/2, WebSocket, a HTTPS s TLS.
- **Komunita a zdroje:** Široká uživatelská základna poskytuje množství dokumentace, příkladů a podpory.

## 12 Co je to LAMP Server?

**LAMP Server** je zkratka označující sadu softwarových technologií, které se tradičně využívají pro provoz dynamických webových stránek a aplikací. Skládá se z následujících komponent:

- **L** – Linux: Operační systém poskytující základní platformu pro provoz serveru.
- **A** – Apache: Webový server pro zpracování a poskytování požadavků na webové stránky.
- **M** – MySQL: Systém správy relačních databází, který ukládá data aplikací a stránek.
- **P** – PHP (nebo Perl/Python): Programovací jazyk používaný pro psaní serverové logiky a dynamického generování webového obsahu.

## 12.1 Proč ty písmenka LAMP už nejsou aktuální?

Písmena zkratky LAMP nejsou již zcela aktuální z následujících důvodů:

- **Linux:** Ačkoliv Linux je stále populární, moderní aplikace mohou běžet na cloudových kontejnerech, kde se operační systém často abstrahuje.
- **Apache:** Apache byl částečně nahrazen lehčími a výkonnějšími servery, jako je Nginx nebo serverless architektura.
- **MySQL:** MySQL byl nahrazen nebo doplněn alternativami, jako je MariaDB (fork MySQL), PostgreSQL (pokročilejší relační databáze) nebo NoSQL databáze, jako MongoDB či DynamoDB.
- **PHP:** PHP je mrtvý a je nahrazován moderními jazyky, jako jsou Python, Ruby, Node.js, Go nebo frameworky jako Django, Flask či React.

Současné aplikace se proto často označují obecněji jako **webové stacky** (např. MEAN stack – MongoDB, Express.js, Angular, Node.js) nebo cloudové platformy (např. AWS, Azure, GCP).

## 13 Skriptovací Jazyky pro Webserver?

### 13.1 Jaké znáte?

- **PHP** - Jeden z nejpoužívanějších skriptovacích jazyků pro dynamické generování webových stránek. \$\$\$
- **Python** - Používaný ve webových frameworkech jako Django nebo Flask.
- **Ruby** - Základ frameworku Ruby on Rails.
- **JavaScript (Node.js)** - Serverová implementace JavaScriptu.
- **Perl** - Tradiční jazyk pro CGI skripty.
- **Bash** - Používá se hlavně pro jednoduché automatizace na webserverech.
- **Lua** - Známé z videoher jako třeba Garry's Mod nebo World of Warcraft, FiveM. Musím zmínit i existenci LuaTeX.

### 13.2 Co dělají?

- **Generování obsahu** - Dynamické vytváření webových stránek (např. podle uživatelských vstupů).
- **Zpracování dat** - Validace a manipulace s daty odeslanými z formulářů.
- **Interakce s databází** - Posílání SQL dotazů, čtení a ukládání dat.
- **Správa uživatelských relací** - Práce s cookies a sessions pro autentizaci a personalizaci.
- **Integrace API** - Komunikace se vzdálenými servery a službami (např. REST API).
- **Automatizace** - Automatické úkoly, jako je odesílání emailů nebo generování reportů.

### 13.3 Proč je chci?

- **Dynamický obsah** - Umožňují přizpůsobit obsah webové stránky konkrétnímu uživateli.
- **Integrace s databází** - Pro práci s daty uloženými na serveru.
- **Škálovatelnost** - Výkonné skripty podporují velký počet uživatelů.
- **Automatizace úloh** - Snižují nároky na manuální údržbu serveru.
- **Bezpečnostní kontrola** - Validace vstupů uživatelů, ochrana před útoky jako SQL Injection.



## 13.4 Jak je připojím k webserveru?

- **PHP** - Použití modulu `mod_php` na Apache serveru nebo PHP-FPM pro Nginx.
- **Python** - Nasazení pomocí WSGI (např. Gunicorn, uWSGI) nebo FastCGI.
- **Ruby** - Využití Rack-based aplikací ve spojení s webservery jako Puma nebo Unicorn.
- **Node.js** - Přímé spuštění serveru pomocí JavaScriptu nebo použití reverzního proxy serveru (např. Nginx).
- **Perl** - CGI skripty nebo moduly jako `mod_perl`.
- **Bash** - Spouštění jako CGI skripty nebo cron úlohy pro automatizované úlohy.

## 14 HTTPS

### 14.1 Proč a jak jej zařídíte?

- **Proč používat HTTPS:**
  - Zabezpečuje komunikaci mezi klientem (prohlížečem) a serverem šifrováním.
  - Zabraňuje odposlechu dat třetími stranami.
  - Ověřuje identitu serveru pomocí certifikátu.
  - Zvyšuje důvěryhodnost webu a je požadován pro moderní webové aplikace.
- **Jak zařídit HTTPS:**
  - Získání certifikátu od důvěryhodné certifikační autority (CA).
  - Instalace certifikátu na webový server.
  - Nastavení serveru pro použití protokolu HTTPS (např. konfigurace Apache nebo Nginx).
  - Testování správné funkčnosti HTTPS pomocí nástrojů, jako je **SSL Labs**.

### 14.2 Co je CA?

- **CA (Certificate Authority):**
  - Důvěryhodná třetí strana, která vydává digitální certifikáty.
  - Certifikát potvrzuje totožnost držitele (např. domény) a obsahuje veřejný klíč.
  - Vystavuje certifikáty po ověření žadatele (např. validace DNS, organizace nebo právních dokumentů).
  - Příklady důvěryhodných CA: Let's Encrypt, DigiCert, GlobalSign.

### 14.3 Jak vypadá výměna klíčů?

- **Proces výměny klíčů:**
  - Používá se při navázání šifrovaného spojení (např. během *TLS handshake*).
  - Klient (prohlížeč) odešle seznam podporovaných šifrovacích algoritmů serveru.
  - Server vybere algoritmus a odešle svůj certifikát s veřejným klíčem.
  - Klient ověří certifikát serveru (pomocí CA) a vytvoří symetrický klíč.
  - Symetrický klíč je zašifrován veřejným klíčem serveru a odeslán zpět.
  - Server dešifruje symetrický klíč svým soukromým klíčem a spojení je navázáno.

## 14.4 Jak probíhá instalace certifikátu?

- **Instalace certifikátu na server:**
  - Získání certifikátu od CA nebo vygenerování pomocí nástroje jako **Certbot**.
  - Uložení certifikátu a soukromého klíče na server.
  - Konfigurace webového serveru (např. v `/etc/apache2/sites-available/` nebo `/etc/nginx/`):
    - \* Specifikace cesty k certifikátu (`SSLCertificateFile`).
    - \* Specifikace cesty k soukromému klíči (`SSLCertificateKeyFile`).
  - Restartování serveru a ověření funkčnosti HTTPS.

## 15 Mailserver

### 15.1 Co dělá mailserver?

Mailserver je klíčovou součástí systému elektronické pošty, která zajišťuje přijímání, odesílání a správu e-mailů. Jeho hlavní úkoly zahrnují:

- **Přijímání e-mailů:** Přijímá zprávy od jiných mailserverů nebo uživatelů a ukládá je pro další zpracování.
- **Odesílání e-mailů:** Zajišťuje předání zpráv dalším serverům podle cílové adresy.
- **Správa schránek:** Ukládá zprávy do konkrétních uživatelských schránek.
- **Filtrování a zabezpečení:** Provádí spamové kontroly, antivirové skenování a ověřování identity odesílatelů.

### 15.2 Jak ho nainstaluju?

Instalace mailserveru zahrnuje následující kroky:

- **Výběr software:**
  - Pro **odesílání e-mailů** jsem použil Postfix, který se běžně používá jako MTA (Mail Transfer Agent).
  - Pro **přijímání e-mailů** jsem použil mailutils. Existují i pokročilejší řešení, například Dovecot, pokud je v plánu používat IMAP/POP3.
- **Instalace balíčků:**
  - Instalace balíčku

```
sudo apt update && sudo apt install postfix mailutils
```
- **Konfigurace DNS záznamů:**
  - Nastavte záznam **MX**, aby byly příchozí e-maily směrovány na tvůj server.
  - Pro zvýšení důvěryhodnosti a bezpečnosti e-mailů se nastavují záznamy **SPF**, **DKIM** a **DMARC**.
    - \* **SPF (Sender Policy Framework):** Umožňuje příjemcům ověřit, které servery mohou odesílat e-maily jménem vaší domény, čímž brání zneužití (spoofing).
    - \* **DKIM (DomainKeys Identified Mail):** Přidává k e-mailům digitální podpis, který zajišťuje, že zpráva pochází od vás a nebyla během přenosu změněna.
    - \* **DMARC (Domain-based Message Authentication, Reporting, and Conformance):** Kombinuje SPF a DKIM a definuje, jak příjemci mají zpracovat e-maily, které těmito ověřeními neprojdou. Zvyšuje bezpečnost a umožňuje detekci zneužití.

- **Konfigurace mailserveru:**
  - V souboru `/etc/postfix/main.cf` nastavte doménu, relay a další základní parametry pro odesílání e-mailů.
  - Pokud plánujete používat protokoly IMAP/POP3 pro příjem e-mailů, nainstalujte a nakonfigurujte software jako Dovecot.
- **Spuštění a testování:**
  - Spustíte a aktivujete Postfix:

```
sudo systemctl enable postfix
sudo systemctl start postfix
```
  - Ověřte funkčnost odesílání e-mailů příkazem:

```
echo "Testovací zpráva" | mail -s "Předmět" uzivatel@domena.cz
```
  - Pokud přijímáte e-maily přes IMAP/POP3, otestujte připojení ke schránce například pomocí e-mailového klienta (např. Thunderbird).

## 15.3 Co pomocí něj můžu provádět?

Mailserver umožňuje:

- **Příjem e-mailů:** Ukládání zpráv do uživatelských schránek.
- **Odesílání e-mailů:** Přeposílání zpráv do jiných domén nebo serverů.
- **Správa uživatelských účtů:** Přidávání a odstraňování uživatelů a jejich poštovních schránek.
- **Filtrování obsahu:** Identifikace spamu, podezřelých příloh nebo malwaru.
- **Šifrování komunikace:** Zajištění zabezpečeného přenosu dat pomocí protokolu TLS/SSL.
- **Logování a monitoring:** Sledování provozu a ladění případných problémů.

## 15.4 Jaké jsou protokoly pro přijímání a posílání mailů?

Protokoly používané mailservery lze rozdělit podle jejich účelu:

- **Pro odesílání e-mailů:**
  - **SMTP (Simple Mail Transfer Protocol):**
    - \* Používá se pro přenos e-mailů mezi servery a odesílání zpráv klientem na server.
    - \* Standardní porty: 25 (nezabezpečený), 465 (SSL), 587 (TLS).
- **Pro přijímání e-mailů:**
  - **POP3 (Post Office Protocol v3):**
    - \* Umožňuje stažení e-mailů ze serveru do klienta a jejich případné odstranění ze serveru.
    - \* Standardní porty: 110 (nezabezpečený), 995 (SSL/TLS).
  - **IMAP (Internet Message Access Protocol):**
    - \* Umožňuje přístup ke zprávám uloženým na serveru a jejich správu přímo na serveru.
    - \* Standardní porty: 143 (nezabezpečený), 993 (SSL/TLS).

## 16 Crond

### 16.1 Co to je za daemona?

- **crond** je daemon určený k plánování a automatizaci spouštění úloh na systémech typu Unix.
- Periodicky kontroluje tabulku plánovače (tzv. **cron tabulku**), kde jsou definovány úlohy, které má spouštět.
- Je součástí většiny Unixových systémů a běží na pozadí jako služba.

### 16.2 Jak se plní?

- Plnění se provádí pomocí editačního nástroje **crontab**.
- Příkaz **crontab -e** otevře editor pro úpravu aktuálního **cron** souboru uživatele.
- Každý řádek v souboru představuje jednu úlohu a obsahuje specifikaci času, kdy se má úloha spustit, a příkaz, který má být vykonán.
- Syntaxe je následující:

```
minut hodina den měsíc den_v_týdnu příkaz
```

- Například: `0 5 * * 1 /home/user/script.sh` spustí skript každý pondělík v 5:00.

### 16.3 K čemu slouží?

- Automatizace pravidelných úloh, jako jsou zálohování, čištění logů, aktualizace databází apod.
- Minimalizace nutnosti manuálních zásahů při opakujících se úlohách.
- Zajištění konzistence a přesnosti při spouštění úloh podle definovaného plánu.
- V systémech správy serverů slouží **crond** jako klíčový nástroj pro správu a údržbu.

### 16.4 Ideální použití?

- Pravidelné zálohování dat: Například denní zálohy na serveru ve specifikované době.
- Automatizované údržbové úlohy: Vyčištění starých dočasných souborů nebo rotace logovacích souborů.
- Monitorování systémových procesů: Periodické kontroly, zda určité služby běží.
- Notifikace: Nastavení upozornění na email, pokud dojde k určité události (např. výskyt chyby).
- Optimalizace výkonu: Naplánování náročných úloh mimo špičku (např. noční hodiny).

## 17 Docker

### 17.1 Proč je tak oblíbený?

- **Snadná kontejnerizace aplikací:** Docker umožňuje zabalit aplikace a jejich závislosti do jednoho kontejneru, což zjednodušuje nasazení a běh aplikací v různých prostředích.
- **Přenositelnost:** Kontejnery vytvořené v Dockeru lze spouštět na jakémkoliv systému, který podporuje Docker, což zaručuje konzistenci mezi vývojovým a produkčním prostředím.
- **Efektivní využití zdrojů:** Na rozdíl od virtuálních strojů (VM) sdílejí kontejnery jádro operačního systému, což zajišťuje jejich rychlé startování a nižší náročnost na systémové prostředky.
- **Automatizace a integrace:** Docker se snadno integruje s moderními DevOps nástroji a CI/CD procesy.

## 17.2 Co přináší?

- **Standardizované prostředí:** Docker zajišťuje, že aplikace běží vždy ve stejném prostředí, čímž eliminuje problémy typu „u mě to funguje“.
- **Rychlé nasazení:** Díky lehkosti kontejnerů je možné rychle vytvářet, nasazovat a odstraňovat aplikace.
- **Modularita:** Aplikace lze rozdělit na menší části (mikroslužby), které mohou běžet v samostatných kontejnerech, což zvyšuje flexibilitu a škálovatelnost.
- **Lepší správa závislostí:** Každý kontejner obsahuje všechny závislosti, což zjednodušuje správu a snižuje riziko konfliktů mezi různými verzemi knihoven.

## 17.3 Pro koho je to dobré?

- **Vývojáři:** Docker umožňuje vývojářům testovat aplikace v prostředí shodném s produkčním, což minimalizuje riziko chyb při nasazení.
- **DevOps:** Uspodňuje správu infrastruktury, automatizaci nasazení a monitorování aplikací.
- **Firmy provozující mikroslužby:** Díky podpoře modularity a škálovatelnosti je Docker ideální pro firmy, které využívají architekturu mikroslužeb.
- **Administrátoři a správci IT infrastruktury:** Docker pomáhá efektivně využívat zdroje serverů a snadno spravovat různé aplikace.
- **Vzdělávací a výzkumné instituce:** Umožňuje snadno vytvářet a sdílet reprodukovatelné prostředí pro výuku nebo výzkumné experimenty.

# 18 Bezpečnostní Audit Systému

Bezpečnostní audit systému je proces, při kterém se analyzuje stav zabezpečení operačního systému a jeho služeb. Nástroj **lynis** poskytuje automatizovaný způsob, jak identifikovat zranitelnosti, slabiny konfigurace a doporučení pro zlepšení bezpečnosti.

## 18.1 Na co se zaměřit?

Při provádění bezpečnostního auditu je vhodné zaměřit se na následující klíčové oblasti:

- **Aktualizace systému:** Ověřit, zda jsou nainstalovány nejnovější bezpečnostní aktualizace operačního systému a software.
- **Uživatelské účty:** Kontrola neaktivních účtů, silných hesel a správné konfigurace oprávnění.
- **Síťová bezpečnost:** Provéřit otevřené porty, firewall a konfiguraci služeb běžících na síti.
- **Konfigurace služeb:** Zajistit, že služby jako SSH, webové servery nebo databáze jsou správně nastaveny.
- **Logování:** Ověřit, zda je logování správně nastavené a logy se ukládají na bezpečné místo.
- **Zranitelnosti:** Identifikace potenciálních slabin, jako jsou neaktuální knihovny nebo zranitelné balíčky.

## 18.2 Jaké nástroje použít?

Pro provedení auditu a analýzu bezpečnosti lze využít následující nástroje:

- **Lynis:** Automatizovaný nástroj pro auditování Linux/Unix systémů, který poskytuje detailní analýzu stavu zabezpečení.
- **nmap:** Nástroj pro skenování sítí a identifikaci otevřených portů.
- **chkrootkit** a **rkhunter:** Nástroje pro detekci rootkitů a malwaru.
- **fail2ban:** Monitorování logů a ochrana před útoky typu brute-force.
- **auditd:** Podrobný monitorovací nástroj pro sledování aktivit v systému.
- **psad:** Nástroj pro detekci a analýzu pokusů o skenování portů a dalších podezřelých aktivit v síti.

## 18.3 Co si z toho odnést?

Bezpečnostní audit systému je klíčovým prvkem v udržení vysoké úrovně zabezpečení. Po provedení auditu by měl administrátor získat:

- **Přehled o stavu systému:** Identifikace aktuálních zranitelností a slabin.
- **Seznam doporučení:** Konkrétní kroky, jak zlepšit bezpečnost systému, například změnou konfigurací nebo instalací aktualizací.
- **Zlepšení povědomí:** Pochopení toho, jak různé komponenty systému ovlivňují celkovou bezpečnost.
- **Proaktivní přístup:** Možnost implementovat bezpečnostní opatření dříve, než dojde k útoku nebo zneužití.

Pravidelné provádění auditů a implementace doporučení zvyšují celkovou odolnost systému vůči kybernetickým hrozbám a zároveň pomáhají udržovat vysokou úroveň zabezpečení v dynamicky se měnícím prostředí.