Webový server

Webový server

- označuje druh serveru, který v počítačové síti s architekturou klient–server poskytuje klientům, nejčastěji webovým prohlížečům, požadovaný webový obsah specifikovaný webovou adresou (URL)
- server může označovat počítač nebo počítačový program (démon), který s klientem komunikuje prostřednictvím protokolu HTTP nebo zabezpečeného HTTPS
- server může poskytovat služby ve veřejné síti jako je World Wide Web (internet) nebo v uzavřené, privátní síti (intranet)

Historie

- **1989**: první webový server na světě, později známý jako CERN httpd běžící na systému NeXTSTEP
- mezi **1991** a **1994** jednoduchost a efektivita prvních technologií pro surfování a výměnu dat přes World Wide Web

Základní vlastnosti

- webový server je připojen k počítačové síti a přijímá požadavky ve tvaru HTTP
- požadavky vyřizuje a počítači, který požadavek vznesl, vrací odpověď (HTML dokument, text, obrázek, ...)

- odpověď serveru je ve tvaru HTTP, je uvozena hlavičkou obsahující stavový kód, za níž následuje samotný obsah
- stavový kód je ve tvaru
  - 2xx – úspěšné vyřízení požadavku
  - 3xx – problémy spojené s přesměrováním
  - 4xx – chyby související s vyřízením požadavku (stránka není dostupná, apod.)
  - 5xx – interní chyby serveru
- server má téměř vždy nějaké možnosti konfigurace
  – stanovení kořenového adresáře
  - konfigurace pro každý jeho podadresář
    - například jaký soubor zpracovat implicitně, obsahuje-li URL pouze daný adresář, nebo v jaké časové zóně se nachází či jaké podporuje jazyky a přípony souborů

Průběh zpracování dotazů

- zadaný odkaz, např. `http://www.example.com/path/file.html`, je převeden klientem na požadavek

```
GET path/file.html HTTP/1.1
Host: www.example.com
```

- webový server na adrese www.example.com přidá tuto cestu k cestě kořenového adresáře příslušného webového serveru
  `/var/www/html/path/file.html`
- následně server soubor přečte (v případě statického obsahu) nebo zpracuje

Dostupné webové servery

- na trhu je několik programů pro webové servery, mezi nejvíce používané patří Apache a nginx

Date	nginx (Nginx, Inc.)	Apache (ASF)	OpenResty (OpenResty Software Foundation)	Cloudflare Server (Cloudflare, Inc.)	IIS (Microsoft)	GWS (Google)	Others
October 2021[55]	34.95%	24.63%	6.45%	4.87%	4.00% (*)	4.00% (*)	Less than 22%
February 2021[56]	34.54%	26.32%	6.36%	5.0%	6.5%	3.90%	Less than 18%
February 2020[57]	36.48%	24.5%	4.00%	3.0%	14.21%	3.18%	Less than 15%
February 2019[58]	25.34%	26.16%	N/A	N/A	28.42%	1.66%	Less than 19%
February 2018[59]	24.32%	27.45%	N/A	N/A	34.50%	1.20%	Less than 13%
February 2017[60]	19.42%	20.89%	N/A	N/A	43.16%	1.03%	Less than 15%
February 2016[61]	16.61%	32.80%	N/A	N/A	29.83%	2.21%	Less than 19%

NGINX

- software vyvinutý ruským programátorem Igorem Sysoevem a publikovaný v roce 2004
- jedná se o bezplatný open-source software
- může být nakonfigurován tak, aby obsluhoval statický webový obsah nebo fungoval jako proxy server
- může být nasazen také k obsluze dynamického obsahu v síti

- Nginx byl napsán s cílem překonat webový server Apache
- zvýšení výkonu bylo za cenu snížení flexibility, například možnosti přepsat nastavení přístupu pro celý systém na základě jednotlivých souborů (Apache toho dosahuje pomocí souboru .htaccess, zatímco Nginx takovou funkci zabudovanou nemá)
- dále oproti Apache některé přidávané modulu potřebují statické linkování (nutná překompilace)

Konfigurace nginx

- základní konfigurační soubor `nginx.conf` nalezneme v adresářích jako `/usr/local/nginx/conf`, `/etc/nginx`, nebo `/usr/local/etc/nginx`
- pro účely výuky si ukážeme ukázkové zprovoznění na systému s Arch Linux https://wiki.archlinux.org/title/Nginx

Instalace a nastavení Nginx Serveru na Arch Linux

- v terminálu Arch Linux stáhneme a nainstalujeme balík `nginx` příkazem

  `sudo pacman -Suy nginx`
- dále povolíme `nginx.service` pro automatické spučtění

  `sudo systemctl enable nginx.service`
- spustíme `nginx.service` příkazem

  `sudo systemctl start nginx.service`
- zkontrolujeme status dané služby pomocí

  `sudo systemctl status nginx.service`
  - pokud se objeví chybová hláška obsahující slovo `type_hash`, přidáme do bloku `html` v `/etc/nginx/nginx.conf` řádek `types_hash_max_size 4096;` a restartujeme službu

    `sudo systemctl restart nginx.service`

- základní kofigurace je umístěná v souboru `/etc/nginx/nginx.conf`
- nginx má jeden hlavní proces a několik dílčích pracovních procesů
  - hlavním účelem hlavního procesu je načítat a vyhodnocovat konfiguraci a udržovat pracovní procesy
  - pracovní procesy provádějí vlastní zpracování požadavků
    - počet pracovních procesů je definován v konfiguračním souboru a může být pro danou konfiguraci pevně stanoven nebo automaticky přizpůsoben počtu dostupných jader CPU

Struktura konfiguračního souboru

- služba nginx se skládá z modulů, které jsou řízeny direktivami zadanými v konfiguračním souboru
  - direktivy se dělí na jednoduché a blokové
    - jednoduchá direktiva se skládá z názvu a parametrů oddělených mezerami a končí středníkem `(;)`
    - Bloková direktiva má stejnou strukturu jako jednoduchá direktiva, ale místo středníku je zakončena sadou dalších instrukcí obklopených závorkami `({` a `})`

- bloková direktiva může mít uvnitř závorek další direktivy, nazývá se kontextová (context)
  - direktivy umístěné v konfiguračním souboru mimo jakýkoli kontext jsou brány jako v hlavním kontextu
  - direktivy `events` a `http` se nacházejí v hlavním kontextu
  - direktiva `server` v `http` kontextu
  - direktiva `location` v `server` kontextu

Poskytování statického obsahu

- jedna ze základních služeb je poskytování statického obsahu, tj. souborů s daným umístěním na serveru
- jako příklad uvažujme dvě umístění
  - `/data/www` se statickou webovou stránkou
  - `/data/images` s obrázky
- defaultní nastavení v `nginx.conf` obashuje základní nastavení s direktivou `html` a `server`
  - zakomentujme vše v základní direktivě `server` a vytvořme pro účely cvičení novou

    - obecně může konfigurační soubor obsahovat několik bloků `server` rozlišených podle portů, na kterých naslouchají, a podle názvů serverů

    - jakmile server nginx rozhodne, který `server` zpracuje požadavek, otestuje URI zadané v hlavičce požadavku na parametry směrnic umístění definovaných uvnitř bloku `server`

- vytvoříme základní adresáře pro `images` a `www` pomocí příkazu

  `sudo mkdir /usr/share/nginx/html/www`

  `sudo mkdir /usr/share/nginx/html/images`

- přidejme základní umístění

```
location / {
    root /usr/share/nginx/html/www;
}

location /images/ {
        root /usr/share/nginx/html;
    }
```

- toto je základní konfigurace serveru, který naslouchá na standardním portu 80 a je přístupný na localhostu `http://localhost/`
- pozor, pokud jsou adresáře prázdné, dostaneme chybu 404

Chybové hlášení

- při vývoji webového serveru je dobré zachytávat chybové hlášení, které udávají typ chyby a často i navrhují nápravu
- v případě nginx nastavíme cestu k error a log souborům pomocí direktiv

```
access_log /var/log/nginx/access.log;
error_log /var/log/nginx/error.log warn;
```

Spuštění pod jiným uživatelem

- ve výchozím nastavení nginx spouští hlavní proces jako root a pracovní procesy jako uživatel http

Bloky serverů

- pomocí bloků serverů je možné obsluhovat více domén
  - jsou srovnatelné s „VirtualHosts“ v serveru Apache HTTP
- lze umístit různé bloky serveru do různých souborů
  - pro použití přístupu `sites-enabled` a `sites-available` vytvořte následující adresáře:

```
sudo mkdir /etc/nginx/sites-available
sudo mkdir /etc/nginx/sites-enabled
```

- vytvořte soubor v adresáři `sites-available`, který obsahuje jeden nebo více bloků serveru:

```
/etc/nginx/sites-available/example.conf

server {
    listen 80;
    listen [::]:80;
    ...
}
```

- na konec bloku http přidejte `include sites-enabled/*;`

```
/etc/nginx/nginx.conf

http {
    ...
    include sites-enabled/*;
}
```

- chcete-li web povolit, jednoduše vytvořte symlink

```
ln -s /etc/nginx/sites-available/example.conf /etc/nginx/sites-enabled/example.conf
```

- chcete-li web zakázat, zrušte aktivní symlink

```
unlink /etc/nginx/sites-enabled/example.conf
```

Cvičení: vytvořte 2 servery dle následujícího příkladu

-vytvořte jednoduchý html soubory, přidejte do `/etc/hosts` řádky `127.0.0.1 domainname1.com` a `127.0.0.1 domainname2.com`

```
...
server {
    listen 80;
    listen [::]:80;
    server_name domainname1.com;
    root /usr/share/nginx/domainname1.com/html;
    location / {
        index index.php index.html index.htm;
    }
}

server {
    listen 80;
    listen [::]:80;
    server_name domainname2.com;
    root /usr/share/nginx/domainname2.com/html;
    ...
}
```

Zabezpečené spojení, aneb HTTPS

- pro účely zabezpečeného spojení je třeba vytvořit patřičný certifikát
- vytvoření soukromý klíč a certifikát s vlastním podpisem

```
mkdir /etc/nginx/ssl
cd /etc/nginx/ssl
openssl req -new -x509 -nodes -newkey rsa:4096 -keyout server.key -out server.crt -days 1095
chmod 400 server.key
chmod 444 server.crt
```

- pro nakonfigurování server HTTPS, je třeba v bloku serveru povolit parametr `ssl` pro naslouchající sokety a zadat umístění souborů certifikátu a soukromého klíče serveru

```
server {
    listen              443 ssl;
    server_name         www.example.com;
    ssl_certificate     www.example.com.crt;
    ssl_certificate_key www.example.com.key;
    ssl_protocols       TLSv1 TLSv1.1 TLSv1.2 TLSv1.3;
    ssl_ciphers         HIGH:!aNULL:!MD5;
    ...
}
```

- pomocí direktiv ssl_protocols a ssl_ciphers lze omezit připojení pouze na silné verze a šifry protokolu SSL/TLS

- základní nastavení SSL/TSL lze také získat pomocí **moz://a SSL Configuration Generator**, kde jsou dostupné nastavení pro různé typy serverů (Nginx, Apache, ...)
  https://ssl-config.mozilla.org/#server=nginx&version=1.17.7&config=intermediate&openssl=1.1.1k&guideline=5.7

- softwarový webový server s otevřeným kódem pro GNU/Linux, BSD, Solaris, macOS, Microsoft Windows a další platformy
- vývoj již od roku 1993
- podporuje velké množství funkcí, mnoho z nich je implementováno jako kompilované moduly rozšiřující jádro
  - příkladem podporovaných programovacích jazyků je Perl, Python, Tcl nebo PHP
- obsahuje:
  - externí modul pro kompresi dat webových stránek posílaných protokolem HTTP (mod_gzip)
  - open source modul pro ochranu a prevenci webových aplikací před napadením (mod_security)

- popis zprovoznění na serveru lze nalézt např. zde https://httpd.apache.org/docs/current/getting-started.html
- pro účely výuky si ukážeme ukázkové zprovoznění na systému s Arch Linux
    - popis instalace lze nalézt zde https://wiki.archlinux.org/title/Apache_HTTP_Server

Instalace a nastavení Apache HTTP Serveru na Arch Linux

- v terminálu zadáme příkaz pro instalaci z repozitáře

  `sudo pacman -Suy apache`

- Server Apache HTTP se konfiguruje pomocí jednoduchých textových souborů
  - v případě Arch Linux se nachází `/etc/httpd/conf` se základním konfiguračním souborem `/etc/httpd/conf/httpd.conf`
  - ve výchozím nastavení se každému, kdo navštíví vaše webové stránky, zobrazí adresář `/srv/http`
  - pro zprovoznění apache je třeba spustit (povolit) daný service

    `sudo systemctl enable httpd.service`

    `sudo systemctl start httpd.service`

    `sudo systemctl status httpd.service`

- konfigurace je často rozdělena do několika menších souborů
  - tyto soubory se načítají pomocí direktivy `Include`
- server se konfiguruje umístěním konfiguračních direktiv do konfiguračních souborů
  - direktiva je klíčové slovo následované jedním nebo více argumenty, které nastavují jeho hodnotu
  - pokud se jedná o globální nastavení, mělo by se objevit v konfiguračním souboru mimo sekce `<Directory>`, `<Location>`, `<VirtualHost>` nebo jinéc

Soubory logů a řešení potíží

- nejcennějším majetkem správce serveru Apache HTTP jsou soubory logů, zejména chybový log
- umístění protokolu chyb je definováno direktivou `ErrorLog`, kterou lze nastavit globálně nebo pro každého virtuálního hosta
- záznamy v logu chyb informují o tom, co a kdy se pokazilo
  - často také uvádějí, jak ji opravit
- log chyb můžete také nakonfigurovat tak, aby obsahoval ID logu, které pak můžete přiřadit k záznamu logu přístupu, abyste mohli určit, jaký požadavek chybový stav způsobil

Některé pokročilé nastavení

- `User http`
  - jakmile je Apache spuštěn uživatelem root
  - výchozím uživatelem je http, který je vytvořen automaticky při instalaci
- `Listen 80`
  - port, na kterém bude Apache naslouchat
  - pro přístup k internetu pomocí routeru musíte port přesměrovat
  - pokud chceme nastavit Apache pro lokální vývoj, uděláme aby byl přístupný pouze z vašeho počítače
    - změňte tento řádek na Listen 127.0.0.1:80

- `ServerAdmin you@example.com`
  - e-mailová adresu správce, kterou lze nalézt např. na chybových stránkách
- `DocumentRoot "/srv/http"`
  - do tohoto adresáře byste měli umístit své webové stránky
  - pokud chcete, změňte jej, ale:
    - nezapomeňte také změnit `<Directory „/srv/http“>` na to, na co jste změnili `DocumentRoot`
    - změňte řádek `Require all denied` na `Require all granted`
    - adresář `DocumentRoot` a jeho nadřazené složky musí povolit oprávnění ke spuštění ostatním

- `AllowOverride None`
  - tato direktiva v sekcích `<Directory>` způsobuje, že Apache zcela ignoruje soubory `.htaccess`
- **User directories**
  - uživatelské adresáře jsou ve výchozím nastavení dostupné prostřednictvím adresy `http://localhost/~yourusername/` a zobrazují obsah adresáře `~/public_html`
  - **Cvičení:** Vyzkoušejte nahrát jednoduchou stránku do public_html a otestovat přístupnost v prohlížeči
    - pokud daný soubor neexistuje, vytvoříme ho a změníme příslušná přístupová práva a restartujeme `httpd.service`
      - `chmod o+x ~`

        `chmod o+x ~/public_html`

        `chmod -R o+r ~/public_html`

- **Virtual hosts**
  - chcete-li mít více než jednoho hosta, odkomentujte řádek `Include conf/extra/httpd-vhosts.conf` v souboru `/etc/httpd/conf/httpd.conf`
  - v souboru `/etc/httpd/conf/extra/httpd-vhosts.conf` nastavte virtuální hostitele
  - velké množství virtuálních hostitelů lze snadno zakázat a povolit
    - dobré je vytvořit pro každého virtuálního hostitele jeden konfigurační soubor a uložit je všechny do jedné složky, např: `/etc/httpd/conf/vhosts`
    - po jejich vytvoření vložíme do `/etc/httpd/conf/httpd.conf` řádky jejichž od/zakomentováním ovlviníme přístupnost

      ```
      Include conf/vhosts/domainname1.dom
      Include conf/vhosts/domainname2.dom
      ```

- nakonec je třeba do `etc/hosts` přidat:

  ```
  127.0.0.1 domainname1.dom
  ```

  ```
  127.0.0.1 domainname2.dom
  ```

- **Cvičení:** otestujte vytvoření dvou virtuálních serverů

Zabezpečené spojení, aneb HTTPS

- pro zabezpečené spojení potřebujeme vytvořit certifikát, který nám zabezpečení obstará
  - existuje celá řada možností jak daný certifikát vytvořit (placené i neplacené služby)
  - pro otestování použijeme `openssl`
- začneme vytvořením složky pro ssl klíč s patřičným oprávněním

```
sudo mkdir /etc/httpd/conf/ssl
```

```
sudo chmod o+x /etc/httpd/conf/ssl
```

- pomocí `openssl` vytvoříme klíč ve vytvořeném adresáři

  `sudo openssl genrsa -des3 -out server.key 2048`

  - parametry určují algoritmus (RSA) pro vytvoření klíče (`-des3` specifikuje jak je klíč šifrován) a parametr `2048` určuje délku (délka menší může způsobit problémy)
- dále vytvoříme klíčový soubor pomocí

  `sudo openssl req -new -key server.key -out server.crt`
- odstraníme heslo z vytvořeného souboru s klířem

  `sudo cp server.key server.key.org`

  `sudo openssl rsa -in server.key.org -out server.key`

  - jako servername nastavime `127.0.0.1`
- nastavíme platnost klíče na jeden rok

  `sudo openssl x509 -req -days 365 -in server.crt -signkey server.key -out server.crt`

- dále pro "zapnutí https" upravíme soubor `/etc/httpd/conf/httpd.conf`
  - odkomentujeme řádek s

    `LoadModule ssl_module modules/mod_ssl.so`

    `LoadModule socache_shmcb_module modules/mod_socache_shmcb.so`

    `Include conf/extra/httpd-ssl.conf`
  - nastavíme `ServerName 127.0.0.1`
  - zkontrolujeme v souboru `conf/extra/httpd-ssl.conf` cesty k vytvořeným certifikátům
- restartujeme server

  `sudo systemctl restart httpd.service`

Přesměrování HTTP na HTTPS

- obecně je bezpečnější provozovat web pouze přes protokol HTTPs
- pro automatické přesměrování stačí odkomentovat v souboru `/etc/httpd/conf/httpd.conf` řádku s

  `LoadModule rewrite_module modules/mod_rewrite.so`
- dále do téhož souboru přidat následující řádky

  `RewriteEngine on`

  `RewriteCond %{HTTPS} off`

  `RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI}`
- nakonec opět restartujeme server pomocí

  `sudo systemctl restart httpd.service`