L7: Web Servers

Web Engineering
188.951 2VU SS20

Jürgen Cito
L7: Webservers

- Overview of web servers (hardware and server software)
- Web servers as part of internet architecture
- Serving static resources over the filesystem
- Dynamic resources through server-side scripting and HTTP
Learning Goals

- Understand the difference between web servers as hardware and software
- Place web servers on the map of broader scale of internet architecture
- Describe static and dynamic resources with respect to web servers and HTTP
- Ability to write a basic web server with JavaScript/Node.js
Recap: High Level Web Overview

Domain Name System (DNS): Translating hostname to IP address

www.google.at —> 172.217.23.227

- Servers wait for requests
- They serve web resources

Informatics

Icons by the Noun Project: Cattaleeya Thongsriphong, Flatart, Graphic Tigers, I Putu Kharismayadi
"Web Server" is an ambiguous term:

1. **Hardware**: A computer ("server") connected to the internet (or any network)

2. **Software**: A program running on a computer/server that accepts HTTP requests over a specific port and answers with HTTP responses
Web Server - Hardware

**Hardware:** A computer ("server") connected to the internet (or any network)

Properties of contemporary web servers

- Part of large data centres
- Latency is geographically dependent, so web servers are often geographically distributed (works through, e.g., DNS)
- Virtual servers: Physical servers can host many virtualized (web) servers

Can also be your own computer (localhost)
Web Server - Geo DNS

Geo-location based Serving through DNS:
Serving resources from geographically closer data centres

DNS: \{Hostname, “Location”\} \rightarrow IP

LDNS = “Local” DNS provided by the ISP

If not present in LDNS, contacts global DNS. They determine IP address to be returned based on policies regarding “location” features (e.g., IP address)
Web Server - Virtual Servers and Containers

One physical server can host multiple **virtual servers** and/or **containers**

**Virtual Machines**
- Enables multiple virtual instances of different operating systems to run in isolation through technology called “hypervisor”
- Hypervisors divide physical resources so that virtual servers can use them and “translates” kernel operations

**Containers**
- Containers “feel” like virtual machines, but are not virtualized
- They provide lightweight process isolation (through cgroups) but share the Host OS kernel
- Beware that containers do not offer the same security boundaries

[Image: https://www.redhat.com/en/topics/virtualization]
Web Server - Cloud

The “cloud” enables provisioning of computational resources over an API

Infrastructure as a service (IaaS)
- API-driven infrastructure (web servers) at scale
- Provides the ability to write a script that automates retrieving new (virtual) server capacity
- Examples: AWS EC2, Google Compute Engine, …

Platform as a Service (PaaS)
- Managed application runtimes (e.g., web servers) that are built on top of IaaS for scalability
- Underlying infrastructure (server) is abstracted away, configuration can provide directives
- Example: Heroku, CloudFoundry, App Engine
  - Deploy web applications by providing directives on process to start or providing container
Web Server Software: A program running on a computer/server that accepts HTTP requests over a specific port and answers with HTTP responses

Web Server/HTTP Server

- Makes resources accessible over a URL and HTTP/S (standard ports 80 and 433)
- Starting a web server on local computer makes it accessible over
  - http://localhost
  - http://127.0.0.1
- Maps **path component** of URL to
  - static asset on the file server
  - dynamically rendered resources
- Often incorporates some functionality for caching and session handling

URL Example: https://localhost:3000/members/rackets?year=2020

Path component + query parameters
Web Server - Static Assets

Serving static assets from the file system

- Web server automatically wraps static files with HTTP Response Headers
- Static assets directly map URL path to relative part of the file system
  - They cannot react to other part of the request (e.g., query parameter)
- MIME-Type is inferred through heuristics (e.g., file endings)
- Example of common static files in web servers
  - HTML, CSS
  - JavaScript (for use in browser)
  - Media (Images, Video, Audio, etc.)

Example:

- Static assets made available at path `/var/www/public_html` on the server
- If we determine [this is configurable] `http://localhost/static/js/search.js` to be a request for static assets we could return `/var/www/public_html/js/search.js`
Web Server - Dynamic Resources

Dynamic Resources
- Executing programs in a server side programming language on the server
- Dynamic resources can react to complete HTTP request (including header information)
  - Path and Query Parameters
  - HTTP Method (GET, POST, PUT, ...)
  - Content Negotiation (Accept: application/json)
  - ...
- System output is treated as the complete HTTP response (including headers)
- However, many programming languages offer library support for basic HTTP related functions and provide abstractions (e.g., for dealing with response headers)
Web Server - Examples

Apache/httpd with CGI (Common Gateway Interface)

- One of the earliest methods of providing dynamic scripting
- Live Example: https://github.com/web-engineering-tuwien/docker-cgi-python

nginx

- Reverse proxy and web server

Node.js Web Server

```javascript
const http = require('http);

const requestListener = function (req, res) {
  res.writeHead(200);
  res.end('Hello, World!');
}

const server = http.createServer(requestListener);
server.listen(8080);
```