Università della Svizzera italiana

Faculty of Informatics

HTTP

HyperText Transfer Protocol

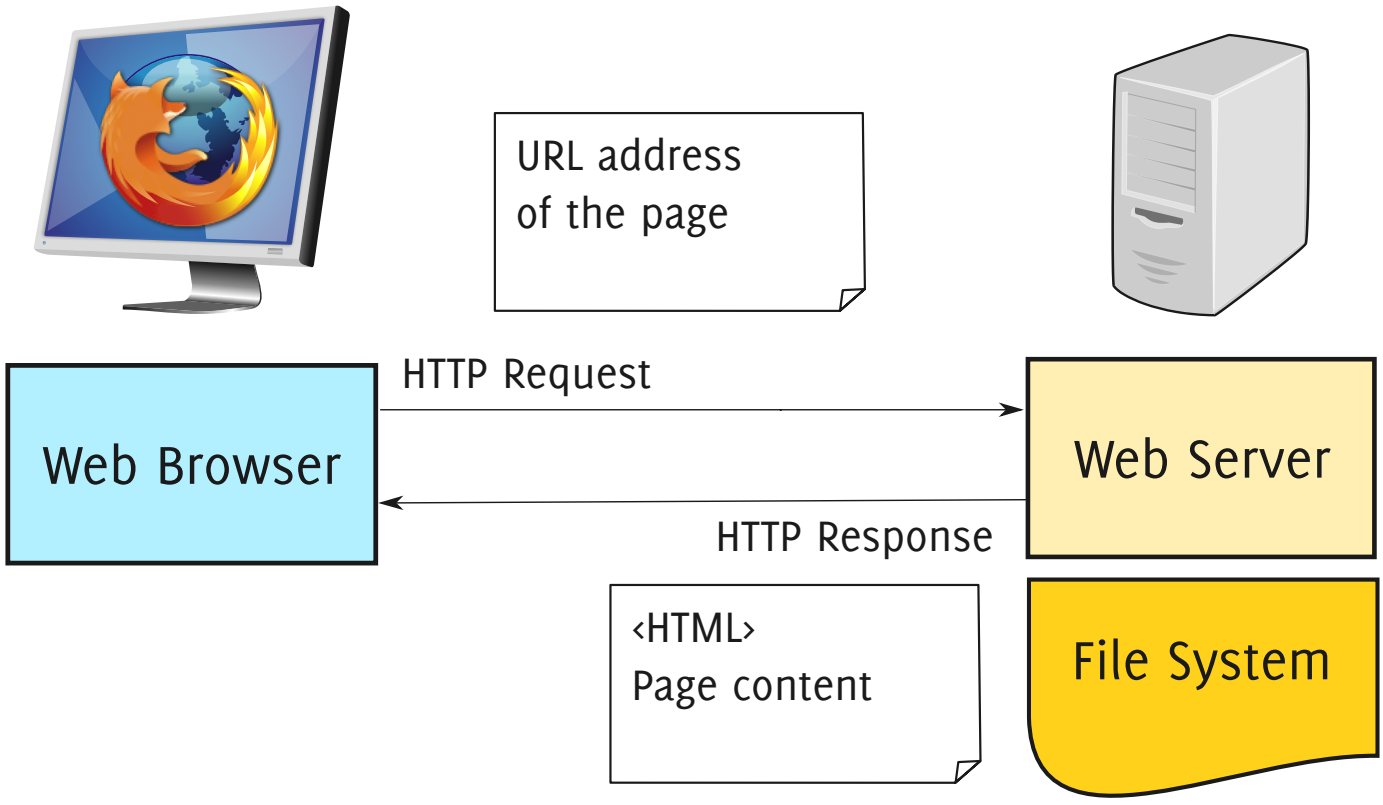
Prof. Cesare Pautasso

http://www.pautasso.info

cesare.pautasso@usi.ch

@pautasso

HTTP Protocol



HTTP Request

```
GET /faculty/pautasso/ HTTP/1.1
Host: www.inf.usi.ch
User-Agent: User-Agent: Mozilla/5.0 (Mac OS X 10.6) Firefox/15.0
Accept: text/xml,application/xml,application/xhtml+xml,*/*
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Pragma: no-cache
Cache-Control: no-cache
```


HTTP Response

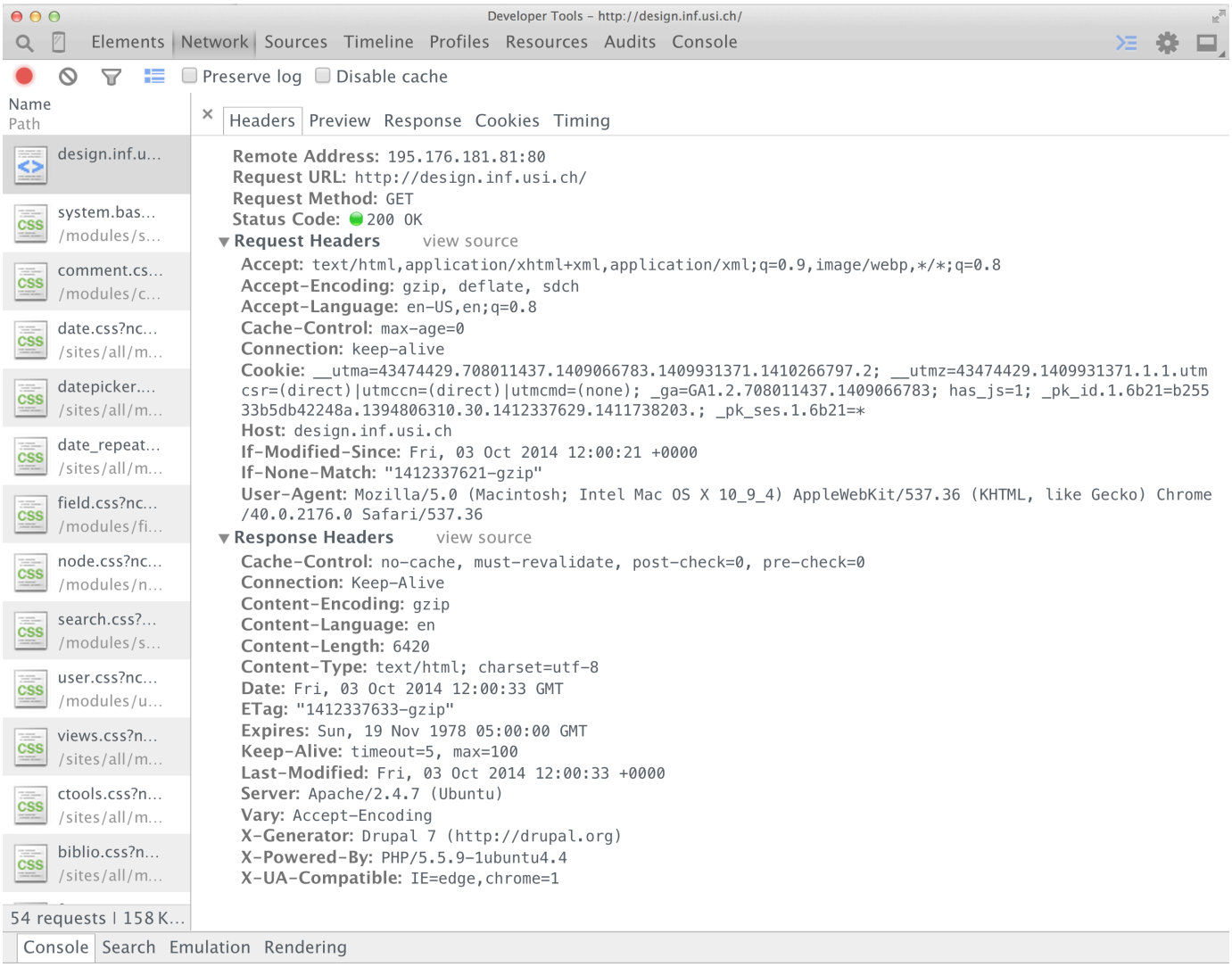
```
HTTP/1.x 200 OK
Transfer-Encoding: chunked
Date: Tue, 18 Sep 2007 14:46:38 GMT
Server: Apache/2.0.52 (Red Hat)
Last-Modified: Sun, 16 Sep 2007 21:50:53 GMT
Etag: "e7198-7030-ae960940"
Accept-Ranges: bytes
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html

<html> <head> <title>Prof. Cesare Pautasso</title>
<style type="text/css"> …</style>
<script type="text/javascript"> <!-- … --> </script>
</head> <body> … </body> </html>
```

HTTP Status Codes

- 1xx Informational
  - 100 Continue
- 2xx Successful
  - 200 OK
  - 201 Created
  - 202 Accepted
- 3xx Redirection
  - 301 Moved Permanently
  - 303 See Other
  - 307 Temporary Redirect
- 4xx Client Error
  - 400 Bad Request
  - 401 Unauthorized
  - 403 Forbidden
  - 404 Not Found
  - 405 Method Not Allowed
  - 414 Request URI Too Long
- 5xx Server Error
  - 500 Internal Server Error
  - 501 Not Implemented
  - 502 Bad Gateway
  - 503 Service Unavailable

HTTP Headers

- HTTP Headers carry meta-data describing the content of the request or the response
- Observe actual HTTP Headers in the developer tools (Network tab)



Content Type Negotiation

- Originally, HTTP was designed to carry Web pages written in the HTML markup language
- In general, the body of an HTTP message can carry any content as specified by the **Content-Type** response header
- The client can request the content to be represented in a specific format using the **Accept** request header

```
GET /time
Accept: text/html
```

```
HTTP/1.1 200 OK
Content-Type: text/html
```


MIME Types

```
application/json
application/pdf
application/soap+xml
application/xhtml+xml
application/zip
audio/mpeg
image/jpeg
image/png
multipart/form-data
text/css
text/html
text/plain
text/xml
video/quicktime
```

http://www.iana.org/assignments/media-types/

Caching

```
GET /faculty/pautasso/ HTTP/1.1
Host: www.inf.unisi.ch
If-Modified-Since: Sun, 16 Sep 2007 21:50:53 GMT
If-None-Match: "e7198-7030-ae960940"
Cache-Control: max-age=0
```

```
HTTP/1.x 304 Not Modified
Etag: "e7198-7030-ae960940"
Date: Tue, 18 Sep 2007 14:50:12 GMT
```

http://www.mnot.net/cache_docs/

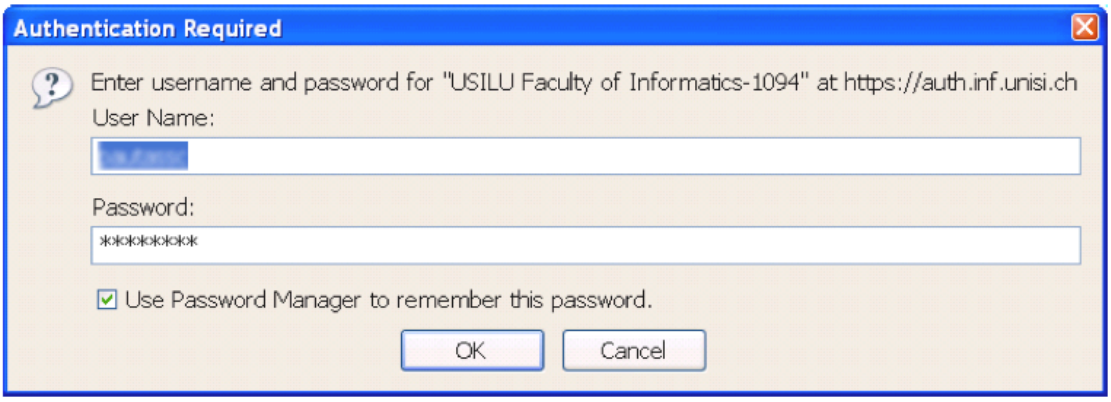
HTTP Authentication

- The Web server can be configured to authenticate clients before they can access some resource
- **Basic Authentication**: the user/password is sent to the server in clear text (Base-64)
- **Digest Authentication** uses a challenge-response scheme that avoids to send the user/password. Instead a response hash value is computed from:
  - username/password
  - nonce, cnonce
  - nc (number of retries)
  - realm, uri
  - qop (quality of protection)

http://www.ietf.org/rfc/rfc2617.txt

HTTP Basic Authentication

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Basic realm="USILU..."
```




```
Authorization: Basic QWxhZGRpbjpvcGVuIH...
```

Base-64 encoding of `username:password`

HTTP Digest Authentication

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Digest
                  realm=“USI",
                  qop="auth",
                  nonce="dcd98...0bf093“
```

```
Authorization: Digest
               username="student",
               nonce="dcd98...0bf093",
               uri="/index.html",
               qop="auth",
               nc=00000001,
               cnonce="0a4f113b",
               response="6629fae49393a05..."
```

HTTP Properties

Standard

- Communication Protocol between any Web browser and any Web server
- Many (competing) server stacks and client libraries exists for all programming languages and operating system/hardware platforms
- Firewalls are typically configures with HTTP Port 80 open by default
- Latest (experimental) version: HTTP2 (binary, performance optimizations)

Synchronous

- The Web server must be available to answer the request of the Web browser
- The interaction completes only after the response has been received by the Web browser

Secure

- Basic Authentication is not secure!
- Use HTTP over SSL (HTTPS) for a secure communication channel

Stateless

- In HTTP 1.0 every Request/Response pair requires to establish a new TPC/IP connection
- Performance Optimization: existing connections can be **kept-alive** with HTTP 1.1
- Every request is independent from all of the others and must contain all information needed by the Web server to answer it
- Cookies can be used for establishing stateful sessions

References

- Robert W. Sebesta, Programming the World Wide Web, Addison-Wesley, 2013
- IETF, HTTP/1.1 Standard, RFC2616, June 1999 http://www.ietf.org/rfc/rfc2616.txt (http://www.ietf.org/rfc/rfc2616.txt)
- Roy Fielding, Architectural Styles and the Design of Network-based Software Architectures (http://roy.gbiv.com/pubs/dissertation/fielding_dissertation.pdf) , University of California, Irvine, 2000, Chapter 5
- Jacob Nielsen, URI are UI, http://www.useit.com/alertbox/990321.html (http://www.useit.com/alertbox/990321.html)