COSC 301
Network Management and Security

Lecture 15: WWW

Today's Focus

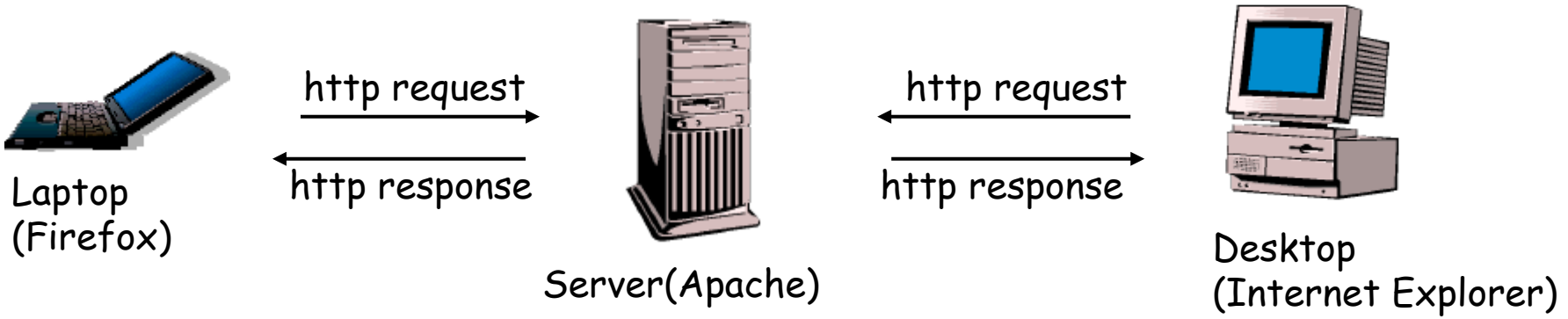
How does World Wide Web (WWW) work?

-- HTTP protocol

-- web server

-- web security/privacy

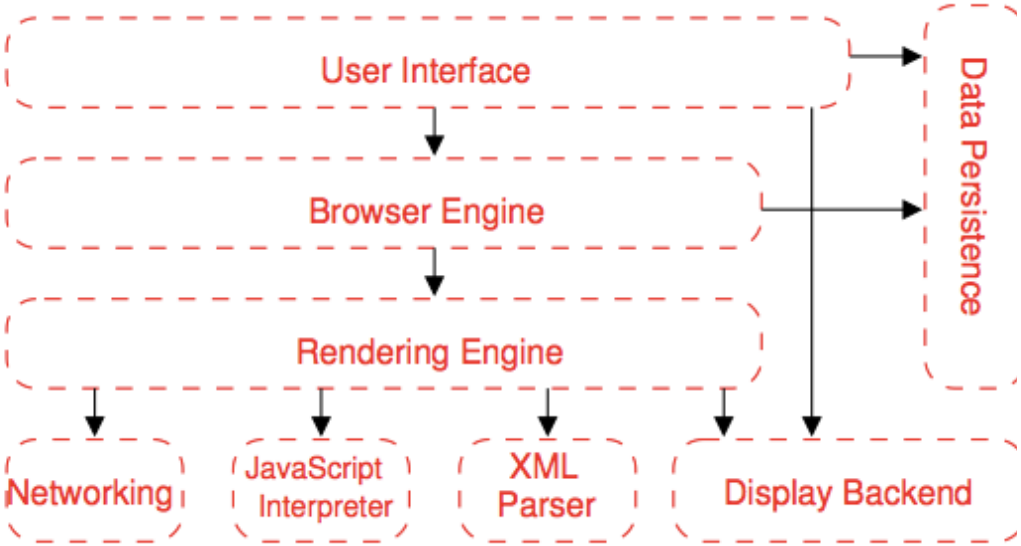
Overview



- A repository of interlinked documents accessed via Internet.
- A distributed client-server service
  - Web Client
  - Web Sever
  - Website
- HyperText Transfer Protocol (HTTP)

Web Browser

- Basic functions
  - Interact with the user
  - Communicate with server
  - Render HTML documents
  - Interpret web resources, e.g. images, videos, etc.
  - Run JavaScript programs
  - Apply CSS rules



- Other functions
  - **Caching**: keep local copies of documents
  - **Authentication**: validate the credentials of the users
  - **State maintenance**: keep “cookies”

Web Server

- Primary function
  - To store, process and deliver web pages to web clients.
- Features
  - **Virtual hosting**: serve many websites using one IP address
    - Name-based, IP-based, Port-based
  - **Server-side scripting**: generate dynamic web pages
  - **Bandwidth throttling**: minimize bandwidth congestion to serve more clients.
- Top web servers

Product	Vendor	April 2014	Percent	May 2014	Percent
Apache	Apache	361,853,003	37.74%	366,262,346	37.56%
IIS	Microsoft	316,843,695	33.04%	325,854,054	33.41%
nginx	NGINX, Inc.	146,204,067	15.25%	142,426,538	14.60%
GWS	Google	20,983,310	2.19%	20,685,165	2.12%

Uniform Resource Locator (URL)

- Need a unique identifier for each webpage. Four identifiers are required to define a webpage
  - Protocol: HTTP, HTTPS, FTP, …
  - Host: IP address or IP name
  - Port: explicitly given if not use a well-known port
  - Path: the location and name of the file

URLs can be quite comprehensive.
http://user:password@host:port/path#anchor?p1=x&p2=y

http://titanium.otago.ac.nz:8080/devel/<username>/projects/connect.php

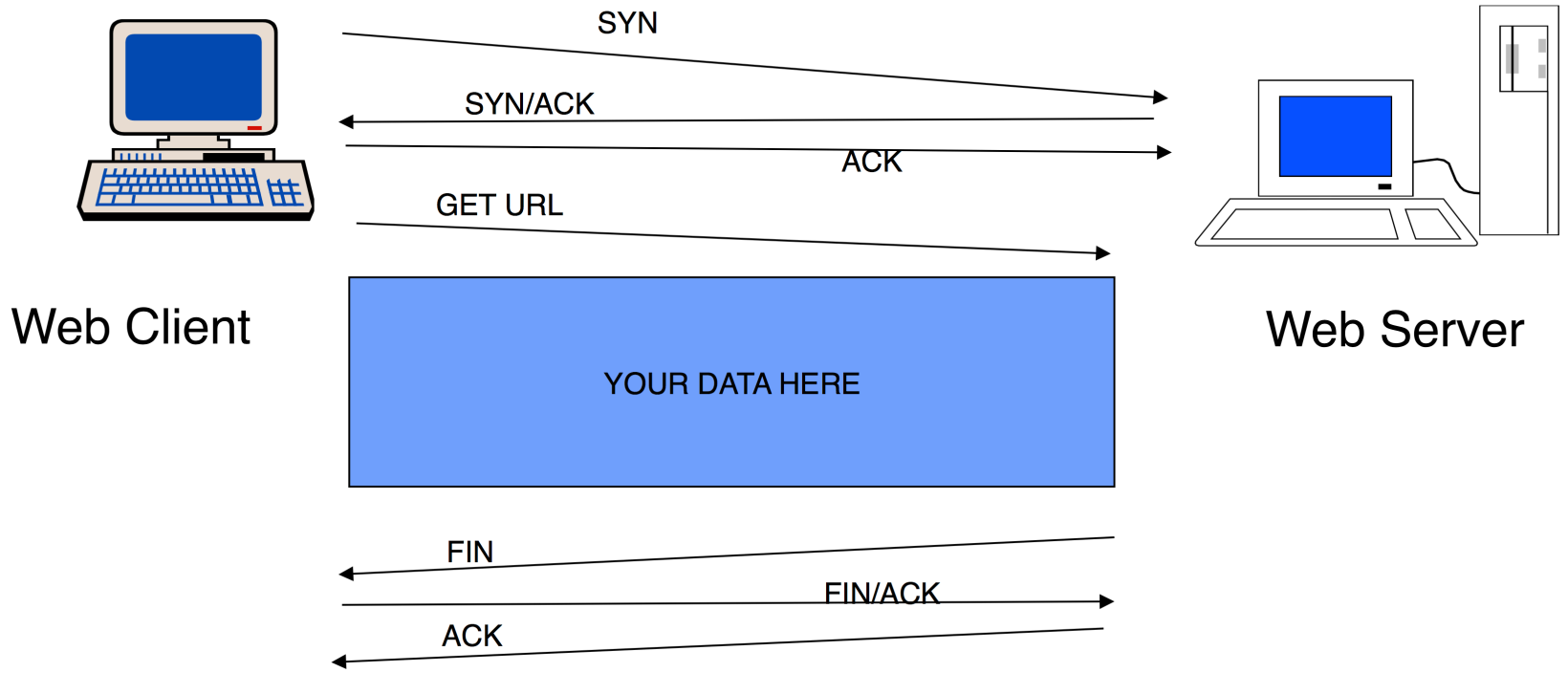
- URL alias
  - Create a user friendly alias for the website path

Web Documents

- Static documents
  - Fixed-content documents, and cannot be changed at clients
  - Prepared using HTML, XML, XSL, XHTML, etc.
- Dynamic documents
  - Created dynamically by a web server upon receiving a request
  - Scripting languages: JSP, ASP, PHP, etc.
- Active documents
  - A program or script stored in web server, but has to be downloaded and run at the clients.
  - Java Applets, JavaScripts

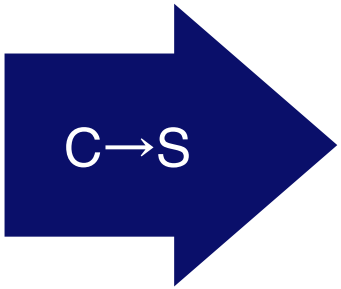
HTTP (1)

- HyperText Transfer Protocol
  - Communication between HTTP clients and server
  - Server uses port 80; Client uses a temporary port number
  - Use the service of TCP (connected-orient & reliable)

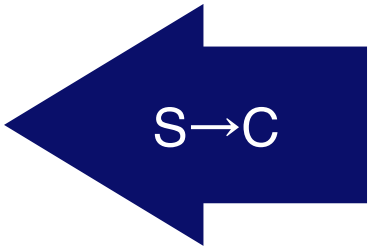


Example HTTP Transaction

GET /cosc301/ HTTP/1.1\r\n
Host: www.cs.otago.ac.nz\r\n
Connection: keep-alive\r\n
User-Agent: Mozilla/5.0...\r\n
\r\n



HTTP/1.1 200 OK\r\n
Date: Sun, 14 Apr 2013 03:31:16 GMT\r\n
Server: Apache\r\n
Last-Modified: Thu, 14 Mar 2013 05:11:48 GMT\r\n
Content-Length: 11162\r\n
Content-Type: text/html; charset=UTF-8\r\n
Connection: close\r\n
\r\n
HTML document appears here



HTTP (2)

- Request methods
  - GET: retrieve a file (95% of requests)
  - HEAD: just get meta-data (e.g., mod time)
  - POST: submitting a form to a server
  - PUT: store enclosed document as URI
  - DELETE: removed named resource
  - TRACE: http “echo” for debugging (added in 1.1)
  - CONNECT: used by proxies for tunneling (1.1)
  - OPTIONS: request for server/proxy options (1.1)

Example Web Page

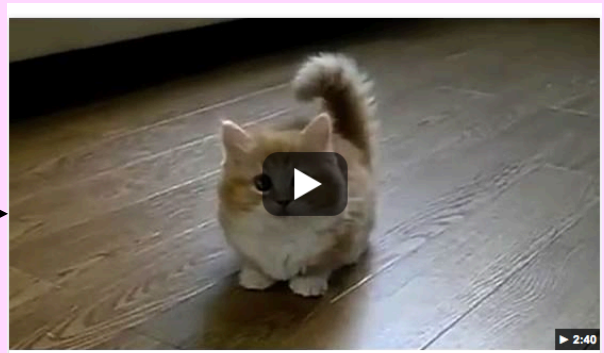
animal.html

cat.mp4

memo.jpg

Animals

Many of us could easily explain why **animals** hold such a special place in our hearts, but what does that say about our relationships with people?

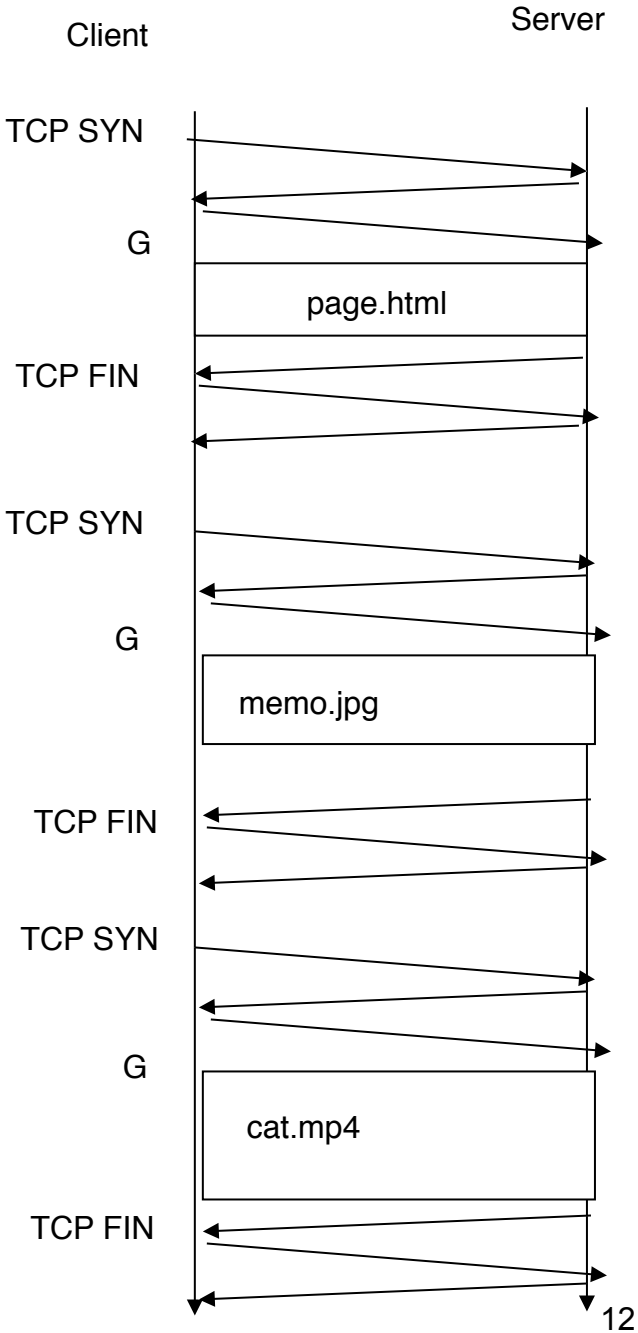


Nonpersistent Connection

- 1 HTTP request/TCP connection
- A file containing links to N different objects in different files (in the same sever) needs N+1 TCP connections.
- Used in HTTP prior to version 1.1

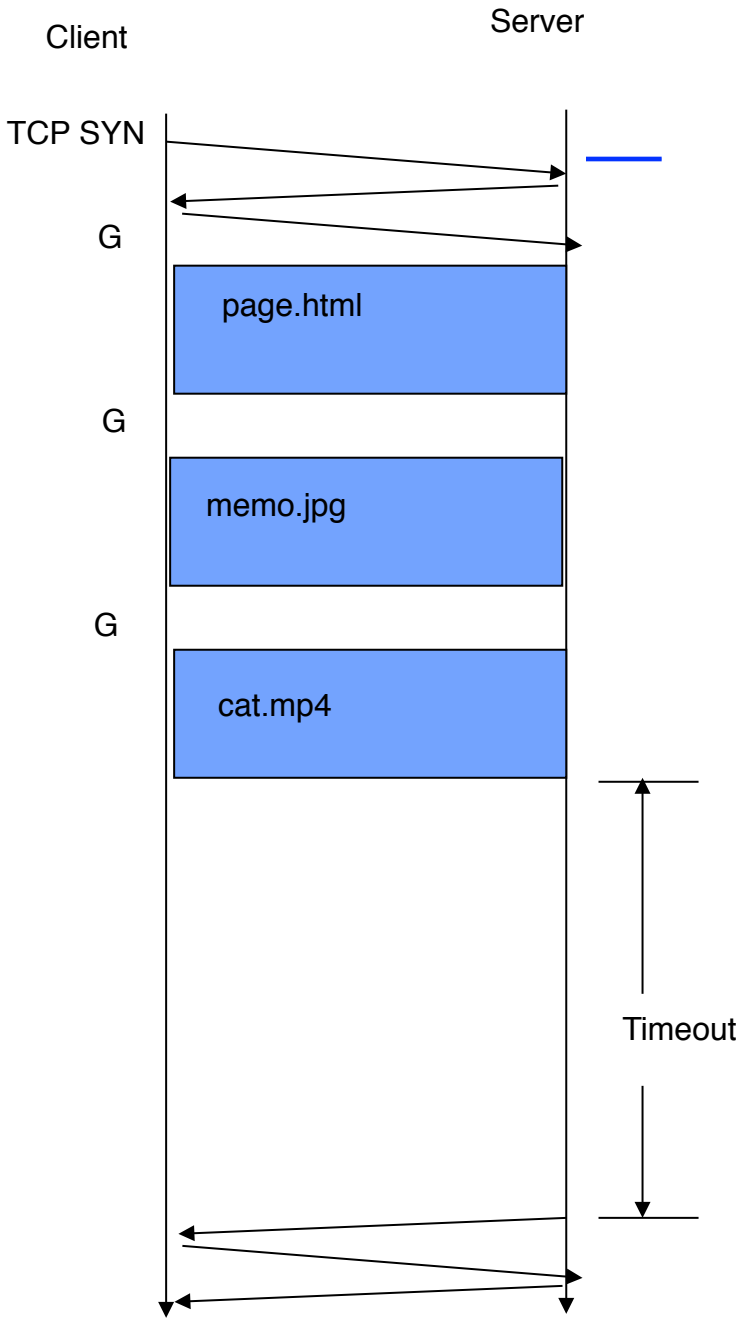
Disadvantages:

Impose high overhead on the server



Persistent Connection

- Multiple HTTP requests/TCP connection
- Default in HTTP version 1.1 and later



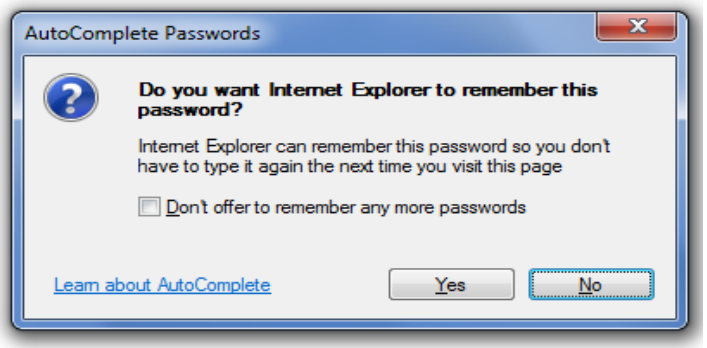
Cookies (1)

- HTTP is a *stateless* protocol
  - Client requests a page, and server sends it
  - Client later requests a 2nd page; it is sent

- HTTP doesn't give a way for the server to know it's from the same user
  - Being stateless is simpler for HTTP
  - But limiting to applications

Cookies (2)

The Web NEEDs state information for clients



- Authentication
  - User-id, password stored on client
  - Sent on next visit. No login required!
- Personalization
  - Remember user preference for fonts, colors, skin, site-options, etc.
- Shopping carts
  - Tracking clients
- Tracking
  - How is our site used?
  - Multi-site tracking by companies looking for usage profiles, etc.

Cookies (3)

- What is HTTP Cookie?

> A small piece of text **made** by the server and **eaten** by the server.

Upon receiving a Cookie, the browser:

(1) Stores the cookie
(2) Sends the cookie back to the server every time it requests a new web page.

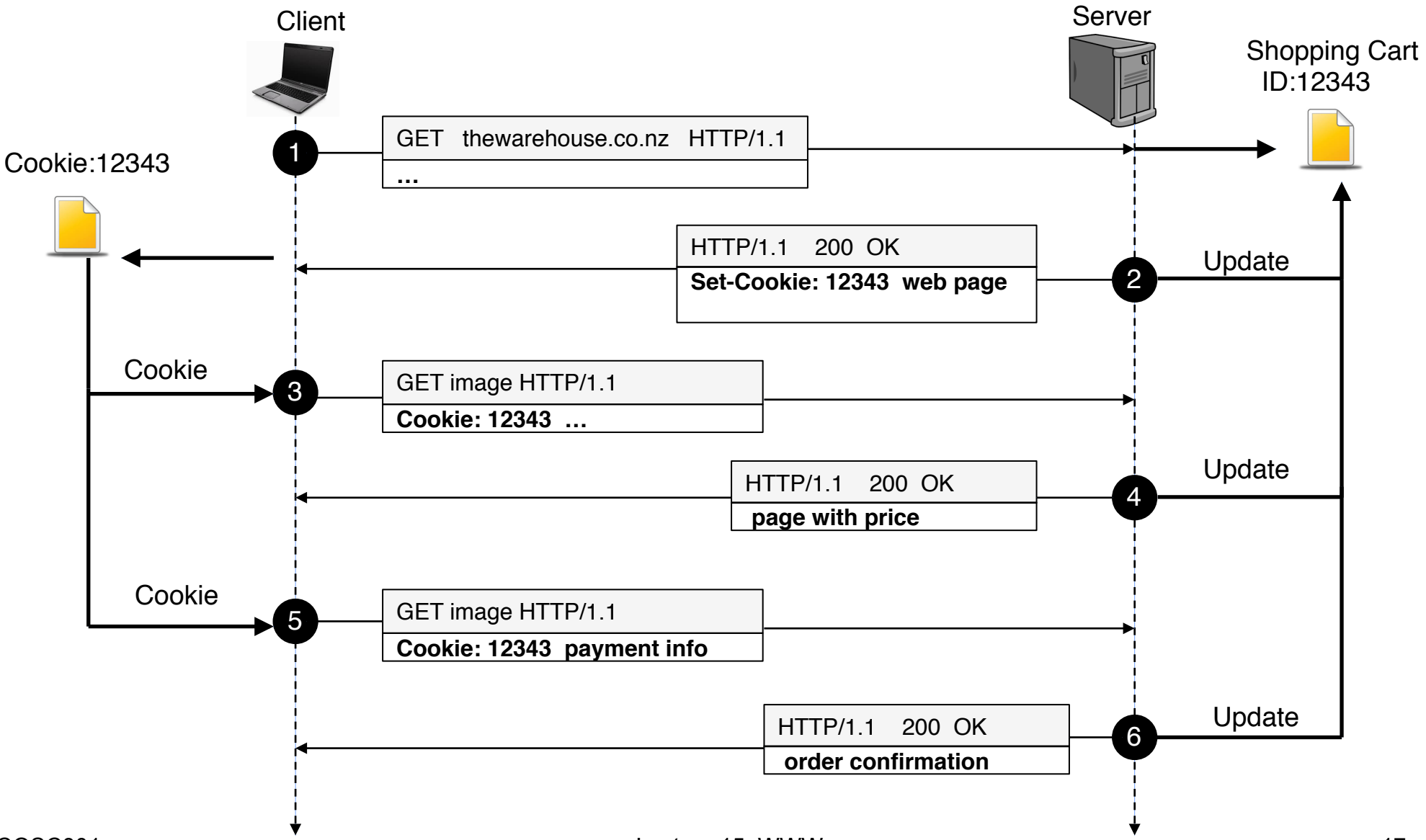
- How does a Cookie look like?

A cookie is a name-value pair:

cookie name = cookie value

Example: languagePreference = EN.

A scenario of an online shopping

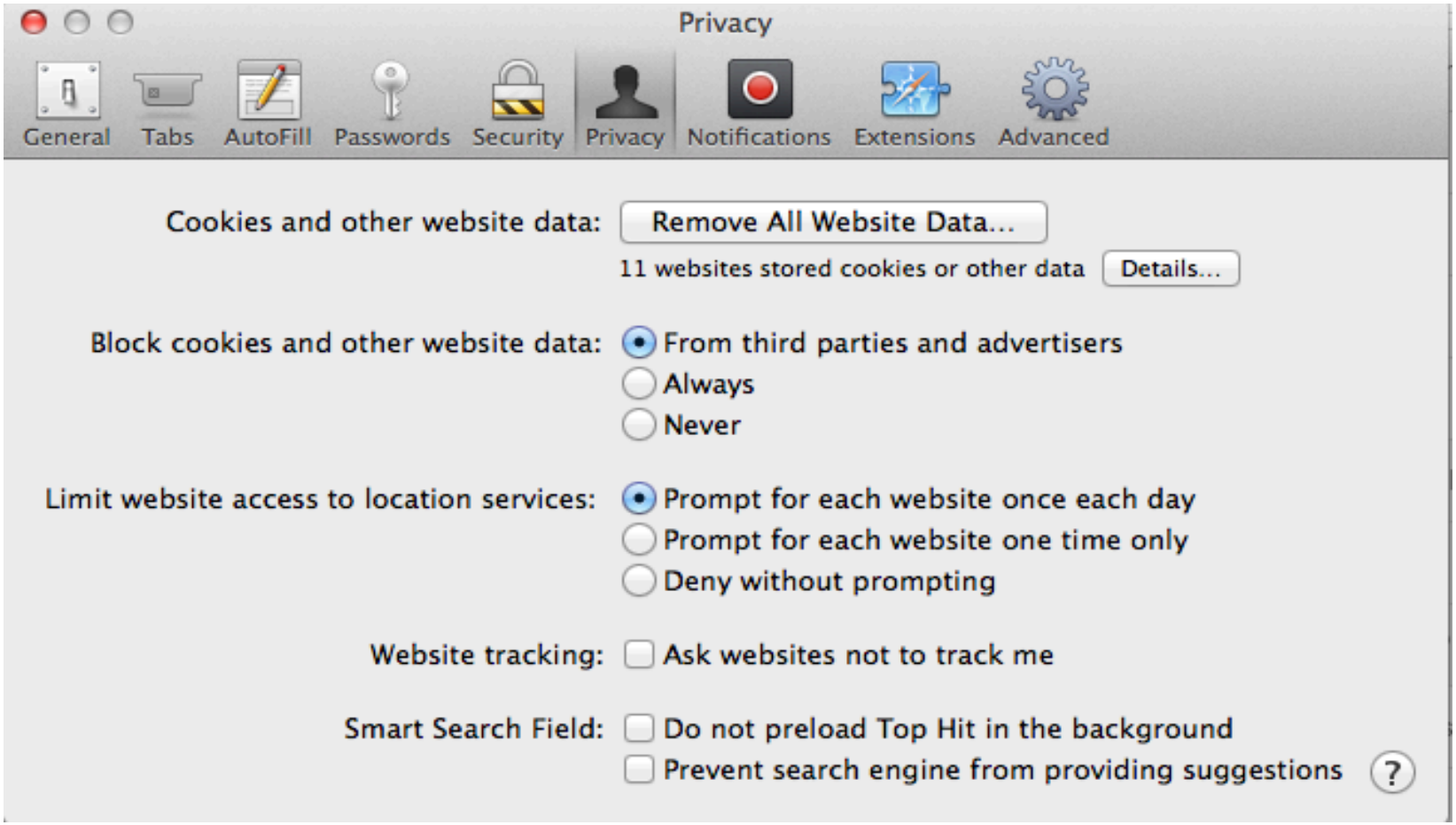


Cookies (4)

- Security
  - Users can change cookies before continuing to browse.
  - Users could swap / steal cookies.
  - Session Hijacking
- Privacy
  - Servers can remember your previous actions
  - If you give out personal information, servers can link that information to your previous actions
  - Servers can share cookie information through use of a cooperating third party
  - Poorly designed sites store sensitive information like credit card numbers directly in cookie

Cookie Management in Safari

- Delete Cookies
- Block Cookies



HTTP Weakness

- HTTP Authentication Security Risks
  - Username and password are encoded, not encrypted.
    - Base 64 encoding and decoding tools are freely available.
  - Authentication information does not change between different requests.
    - Sniffer can replay!
  - Digest authentication is better, but
    - Requesting unnecessary authentication leads to password sharing.
    - only authenticates the browser (user), not the server, so impersonating websites could harvest passwords

HTTP over TLS (HTTPS)

Validating requests

- Beware embedded mark-up!
  - Cross-Site Scripting (XSS)
  - PHP form code at server: <form method="post" action="<?php echo $_SERVER["PHP_SELF"];?>">
  - Suppose the URL is http://www.example.com/test_form.php
  - Server returns <form method="post" action="test_form.php">
  - But if the URL is set by the attacker like http://www.example.com/test_form.php/%22%3E%3Cscript%3Ealert('hacked')%3C/script%3E
  - The server returns <form method="post" action="test_form.php/"><script>alert('hacked')</script> and executed at the client side.
  - Worse, the embedded script could be <script src=https://evil-user.net/badscript.js>
  - Solution: the server should use htmlspecialchars($_SERVER["PHP_SELF"]) to filter the special characters

Validating requests

- Database query placeholders
  SQL Injection attacks
  - statement = SELECT * FROM users WHERE name = '"' + userName + '";"
  - User name from attacker: ' OR '1'='1
  - The SQL becomes SELECT * FROM users WHERE name = '' OR '1'='1';
  - $sth = prepare('SELECT * FROM table WHERE name = ?'); $sth->execute($name); *YES*
  - $sth = prepare("SELECT * FROM table WHERE name = $name"); $sth->execute(); *NO*

Summary

- HTTP non-persistent and persistent connections
- What are cookies and their security/privacy issues
- Security issues related to HTTP and webpages
  - SQL injection
  - Cross-site scripting
  - Lesson: sanity check of user input

References

- HTTP Authentication – RFC2617
  www.rfc-editor.org
- Open Web Appl. Security Project
  www.owasp.com
- Hacking Linux Exposed (2nd Ed.)
  Brian Hatch & James Lee, ISBN
  0-07-222564-5
  www.hackinglinuxexposed.com