

## Lecture #9

# Internet Applications

# Contents

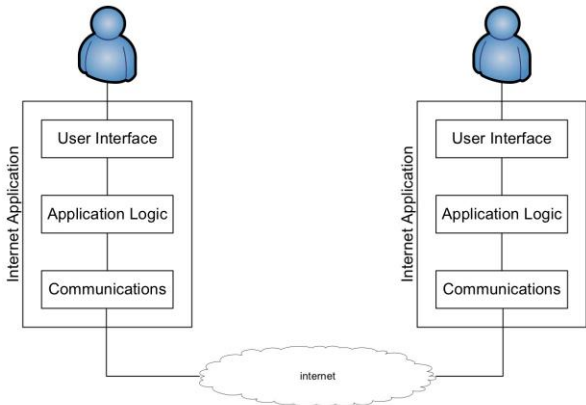
## Internet Applications

### Naming and DNS

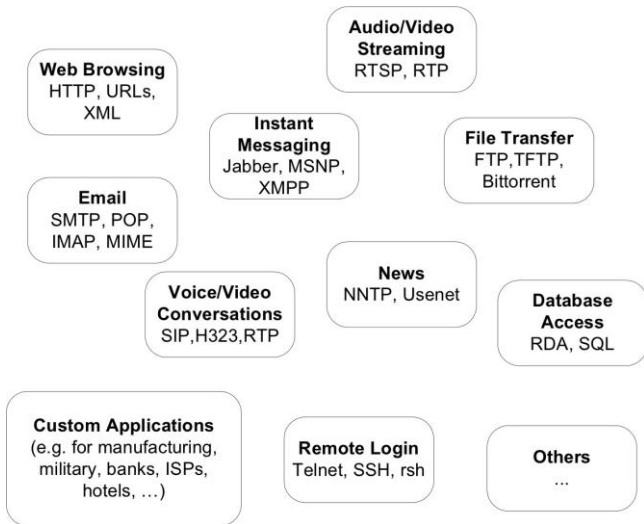
### Web Browsing and HTTP

# Internet Applications

- ▶ Software applications that involve communication with other applications over the Internet
- ▶ Internet applications have three basic functions: user interface; application logic; communications
- ▶ Communications follow an **application layer protocol**



# Types of Internet Applications



# Internet Applications

- ▶ Most Internet applications are **client/server** based
- ▶ Internet applications are implemented as user-level software processes
- ▶ Application layer protocols make use of Transport layer for communications
- ▶ **Sockets** interface is commonly used to allow user-level applications to use transport protocol in OS

# Contents

Internet Applications

Naming and DNS

Web Browsing and HTTP

# Identifying Computers and Files on the Internet

- ▶ IP addresses are used to identify computer (interfaces)
- ▶ **Domain names** are user-friendly way to identify computers
- ▶ Domain names follow a hierarchical structure: an organisation manages a domain, and can allocate sub-domains to other organisations
- ▶ Top Level Domains (TLDs) are managed by Domain Name Registrars .com .biz .org .net .name .info ...
- ▶ Country Code TLDs managed by national registrars (.eg, .kw, .au, .de)
- ▶ Typically ccTLDs are divided into sub-domains and sub-managed by registrars

## Identifying Files on the Internet

- ▶ **Uniform Resource Locators** (URLs) are used to identify files (resources) on the Internet
- ▶ URLs are a specific form of **Uniform Resource Identifier**
- ▶ scheme : user @ host : port path ? query
  - ) scheme: identifies the application protocol used to access the resource, e.g. http, ftp, https, dns, ipp, news, sip, ...; often followed by //
  - ) user: identifies the user that is accessing the resources; password is optional
  - ) host: identifies the host that stores the resources; typically a domain name or IP address
  - ) port: identifies the port number of the application on the host; if not given, the default value for the scheme will be used, e.g. http (80); https (443)
  - ) path: pathname of the file where the resource is located
  - ) query: additional identification information for the resource location; typically in attribute-value pairs, e.g. key=value
- ▶ Most parts are optional and there are exceptions!



# Example URLs

## Web Browsing

<http://www.example.com/dir/file.html>

<http://73.16.0.4:40240/dir/file.html>

<https://www.example.com/dir/file.html>

<http://example.com/dir/file?id=6&name=steve>

## Email

<mailto:steve@example.com>

<mailto:steve@example.com?subject=test>

## Remote Login

<telnet://steve@example.com>

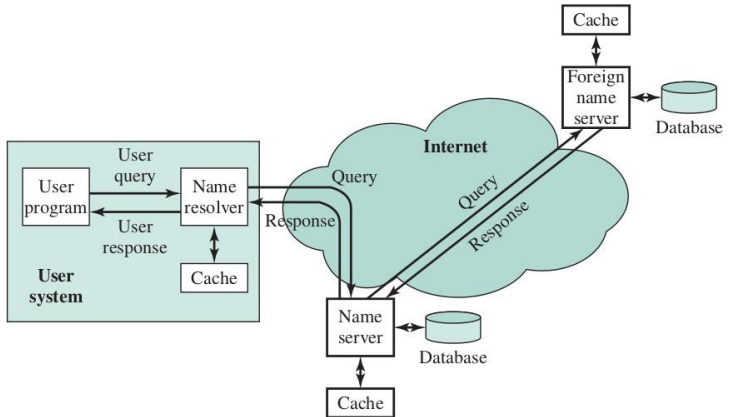
<telnet://steve:mypassword@example.com:46>

<ssh://steve@example.com>

# Domain Name System

- ▶ **Domain Name System**(DNS) specifies: format and structure of domain names; and how to map domain names to IP addresses
- ▶ Domain names and their corresponding IP address are registered at DNS servers
- ▶ Application uses DNS protocol to retrieve the corresponding IP address from the DNS server; request from application goes to name **resolver**
- ▶ DNS servers are structured in hierarchical manner to provide fast and accurate responses
  - ) DNS servers may be **authoritative server** for domains within its portion of domain name space; domain and corresponding IP address are stored in a database
  - ) DNS servers may cache other domain names (and their IP)
  - ) DNS servers may know IP address of other DNS servers for **recursive** or **iterative** requests
  - ) **Root DNS servers** know the IP address of authoritative DNS servers for TLDs and ccTLDs

# DNS Name Resolution



# Contents

Internet Applications

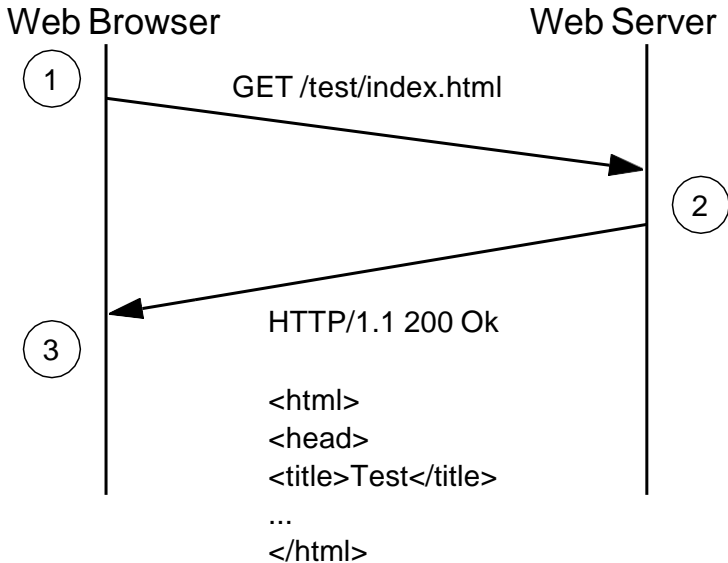
Naming and DNS

Web Browsing and HTTP

# Web Access with Hypertext Transfer Protocol

- ▶ HTTP is a request/response protocol for web browsing
- ▶ HTTP is stateless; no dependence between a request and previous request
- ▶ User Agent (client) sends HTTP Request message
- ▶ Server responds with HTTP Response message
- ▶ Default server port number: 80
- ▶ Generic HTTP message format:
  - Start line
  - Optional header lines
  - <empty line>
  - Optional message body
- ▶ Start line differs for request and response
- ▶ Header format: field-name: value

# HTTP Example



# HTTP Request Messages

- ▶ Start line: Method URLVersion
- ▶ Methods:
  - ) GET: retrieve the resource at the specific URL
  - ) HEAD: same as GET, except do not return message body (only header)
  - ) OPTIONS: retrieve options available for resource or server
  - ) POST: asks server to accept and process the attached data at the resource
  - ) ...
- ▶ Version: version of HTTP, e.g. HTTP/1.0, HTTP/1.1

# HTTP Response Messages

- ▶ Start line: Version StatusCodeStatusReason
- ▶ Status Codes and Reasons:
  - ) 100: Continue (the client should continue with its request)
  - ) 200: OK (the request succeeded)
  - ) 301: Moved Permanently (the requested resource has a new URL)
  - ) 304: Not Modified (resource hasn't changed since last request, client should use cached copy)
  - ) 401: Unauthorized (request must include user authentication)
  - ) 403: Forbidden (request was understood, but server refuses to process it)
  - ) 404: Not Found (server cannot find resource at requested URL)
  - ) 503: Service Unavailable (server currently unable to handle request, e.g. server is too busy)



## HTTP Headers

- ▶ Date: data and time of message generation
- ▶ Host: domain name of host of resource (means relative URLs can be used)
- ▶ Accept-Charset, Accept-Encoding, Accept-Language: indicate the character sets, encodings and languages that client can accept
- ▶ Authorization: include user credentials (e.g. username, password) if authorization is required
- ▶ User-Agent: indicates information about the client (user agent), e.g. web browser
- ▶ Referrer: URL from which this request came from
- ▶ Content-Encoding: encoding or compression, e.g. gzip
- ▶ Content-Length: length of message body on bytes
- ▶ Content-Type: the type of content in messagebody
- ▶ Last-Modified: indicates data/time when content was last modified on server

# HTTP Example

