Application layer

Goals:
- Conceptual aspects of network application protocols
  - Client server paradigm
  - Service models
- Learn about protocols by examining popular application-level protocols
  - HTTP
  - DNS
Application-layer protocols

- One “piece” of an app
- Define messages exchanged by apps and actions taken
WWW: the HTTP protocol

HTTP: hypertext transfer protocol

- WWW’s application layer protocol
- Client/server model
  - **Client**: browser that requests, receives, “displays” WWW objects
  - **Server**: WWW server sends objects in response to requests
- HTTP/1.0: RFC 1945
- HTTP/1.1: RFC 2616
The HTTP protocol: More

HTTP: TCP transport service:
- Client initiates TCP connection (creates socket) to server, port 80
- Server accepts TCP connection from client
- http messages (application-layer protocol messages) exchanged between browser (http client) and WWW server (http server)
- TCP connection closed

HTTP is “stateless”
- Server maintains no information about past client requests

Aside
Protocols that maintain “state” are complex!
- Past history (state) must be maintained
- If server/client crashes, their views of “state” may be inconsistent, must be reconciled
DNS: Domain Name System

People: many identifiers:
   ○ SSN, name, Passport #

Internet hosts, routers:
   ○ IP address (32 bit) – used for addressing datagrams
   ○ “name”, e.g., gaia.cs.umass.edu – used by humans

Q: Map between IP addresses and name?

☒ Secure Domain Name System (DNS) Dynamic Update: RFC 3007
DNS: Domain Name System

Domain Name System:

- Distributed database: implemented in hierarchy of many name servers
- Application-layer protocol: host, routers, name servers communicate to resolve names (address/name translation)
  - Core Internet function implemented as application-layer protocol
  - Complexity at network’s “edge”
DNS name servers

No server has all name-to-IP address mappings

Local name servers:
- Each ISP, company has local (default) name server
- Host DNS query first goes to local name server

Authoritative name server:
- For a host: stores that host’s IP address, name
- Can perform name/address translation for that host’s name
**Simple DNS example**

Host `surf.eurecom.fr` wants IP address of `gaia.cs.umass.edu`

1. Contacts its local DNS server, `dns.eurecom.fr`
2. `dns.eurecom.fr` contacts root name server, if necessary
3. Root name server contacts authoritative name server, `dns.umass.edu`, if necessary

```
+-----------------+    +-----------------+    +-----------------+    +-----------------+
| requesting host |    | local name server|    | authoritative name server |    | root name server |
| surf.eurecom.fr |    | dns.eurecom.fr |    | dns.umass.edu |    |                  |
```
DNS: Root name servers

- Contacted by local name server that can not resolve name
- Root name server:
  - Contacts authoritative name server if name mapping not known
  - Gets mapping
  - Returns mapping to local name server
- ~ dozen root name servers worldwide
Simple DNS example

host `surf.eurecom.fr` wants IP address of `gaia.cs.umass.edu`

1. Contacts its local DNS server, `dns.eurecom.fr`
2. `dns.eurecom.fr` contacts root name server, if necessary
3. Root name server contacts authoritative name server, `dns.umass.edu`, if necessary
DNS example

Root name server:
- May not know authoritative name server
- May know intermediate name server: who to contact to find authoritative name server

Diagram:
- Requesting host: surf.eurecom.fr
- Local name server: dns.eurecom.fr
- Intermediate name server: dns.umass.edu
- Authoritative name server: dns.cs.umass.edu
- Root name server
DNS: Iterative queries

Recursive query:
- Puts burden of name resolution on contacted name server
- Heavy load?

Iterated query:
- Contacted server replies with name of server to contact
- “I don’t know this name, but ask this server”
DNS: Caching and updating records

- Once (any) name server learns mapping, it *caches* mapping
  - Cache entries timeout (disappear) after some time
- Update/notify mechanisms under design by IETF
  - RFC 3007 (Feb. 2004)