Plan

- Application layer
  - Principles of application layer protocols
  - The Web and HTTP
  - File transfer: FTP
  - Electronic mail in the Internet
  - **DNS — The Internet's directory service**

Internet Directory Service (DNS)

The internet protocols use **IP addresses** to identify hosts. They are a series of numbers, between 0 and 255, separated by periods like `202.30.38.143`.
But these numbers have are difficult to remember, contrary to **host names**, which are made of a series of words separated by periods, like `mail.konkuk.ac.kr`.
The internet provides an application-layer service that allows applications to find the IP address of a host from its name: the **Domain Name Service** (**dns**).
Because of its functional similarity with the telephone directories, which map subscribers' names to their phone number, this service is also called **Internet Directory Service**.

Internet Directory Service (DNS) (cont)

The DNS is

- a client/server protocol;
- a distributed database implemented in a hierarchy of **name servers**;
- an application-layer protocol that allows hosts and name servers to communicate in order to provide the translation service.

Name servers are often machines running the UNIX operating system and the Berkeley Internet Name Domain (BIND) software.
The DNS protocol runs over UDP and uses port 53.
DNS is commonly used by other application-layer protocols, like HTTP, SMTP and FTP.

Internet Directory Service (DNS) (cont)

When a user wants to retrieve the web page whose URL is `http://konkuk.ac.kr/index.html`,

1. its browser extracts the hostname `konkuk.ac.kr` from the URL,
2. passes it to the client side of the DNS application,
3. this application sends to a DNS server the hostname,
4. after some delay, it receives back the IP address of the referred host,
5. this IP address is returned to the browser,
6. the browser opens a TCP connection to the host and to the HTTP server process located there.

Internet Directory Service (DNS) (cont)

DNS provides a few other important services in addition to translating hostnames to IP addresses:

- **Host aliasing.** A host with a complicated name can have one or more (simpler) names. For example `relay1.west-coast.enterprise.com` may have two aliases, e.g., `enterprise.com` and `www.enterprise.com`. In this case, `relay1.west-coast.enterprise.com` is said to be the **canonical hostname**.
- **Mail server aliasing.** Mail server names can also be aliased in order to get a simple e-mail address, as `bob@hotmail.com` instead of `bob@relay1.west-coast.hotmail.com`.

Internet Directory Service (DNS) (cont)

- **Load distribution.** Busy sites are replicated over multiple servers, hence an *ordered list* of IP addresses is associated with one canonical hostname. Each time a client requires an IP address in the list, the first is returned and the list is rotated. This way, the load is balanced between all the servers.

Internet Directory Service (DNS) (cont)

What if there were only one name server in the internet?

- If the name server crashes, so does the internet.
- A single name server would have to handle all the DNS queries (from HTTP requests and e-mails at hundreds of millions of hosts).
- A single name server cannot be geographically close to all the hosts, imposing unfair delays to the hosts.
- A single name server would need a huge database to store all the hostname on the internet.

Therefore *a centralised database does not scale*.

Internet Directory Service (DNS) (cont)

That is why

- there is a large number of name servers around the world
- which are organised in a hierarchy;
- no single name server has all the mappings for all the hosts;
- the mappings are distributed across the name servers;
- there are three kind of name servers:
  1. **local name servers**,
  2. **root name servers**,
  3. **authoritative name servers**.

DNS/Local name servers

Each ISP has a **local name server** (also called a **default name server**), so it is relatively close to the users of this ISP:

- in an institution, it can be on the same LAN as the user;
- in case of a residential ISP, it is separated from the users by no more than a few routers.

The IP address of the local name server has to be set by the user in the network configuration of his host.
If the user requests a translation for a host which is part of the same ISP domain, the request will be *immediately* served by the local name server. As host `surf.eurecom.fr` requesting the IP address of `baie.eurecom.fr`.

DNS/Root name servers

When a local name server can not serve a request, it then acts as a DNS client and queries a **root name server**.
If the root name server has the required mapping, it returns the IP address to the local name server which, in turn, delivers it to the querying (initial) host.
If the root name server has not the required record, it knows the IP address of an *authoritative name server* which has it.

DNS/Authoritative name servers

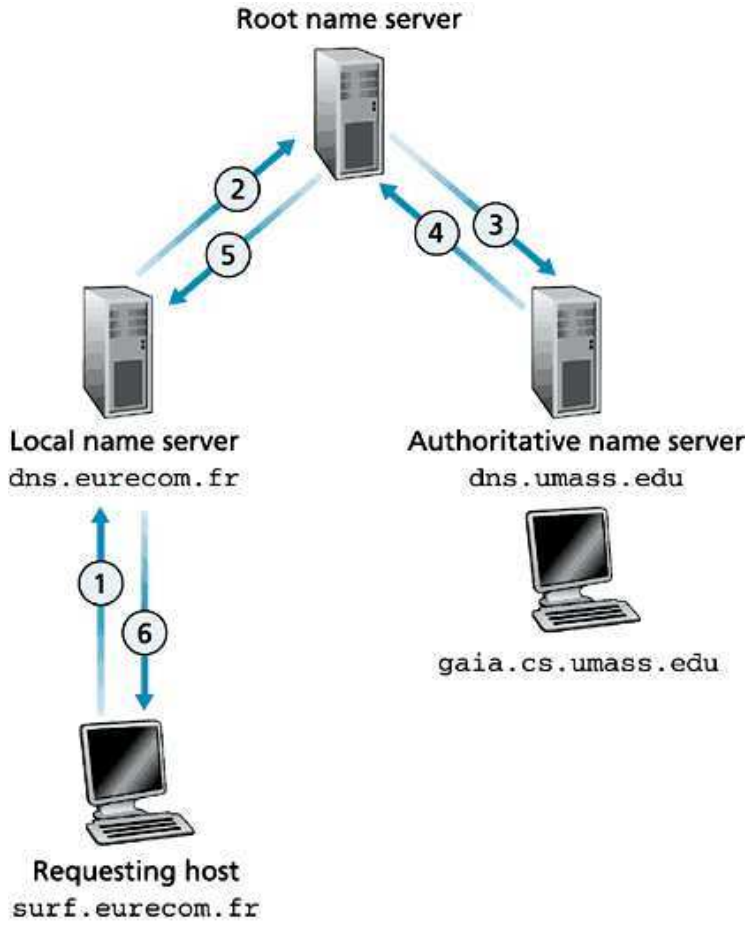
Every host is registered with an authoritative name server.
This authoritative name server is a name server in the (*registered*) host's local ISP.
In principle, each host should be registered in two authoritative name servers, in case of failures.
By definition, a name server is authoritative for a given host if it always translates the host name into its IP address.
Many name servers act both as local and authoritative name servers.

DNS/Example

Let us consider an example. Assume that host `surf.eurecom.fr` queries the IP address of host `gaia.cs.umass.edu`.

1. `surf.eurecom.fr` sends its query to its local name server `dns.eurecom.fr`,
2. which, in turn, forwards the query to a root name server,
3. which, in turn, forwards the query to an authoritative name server for the host `gaia.cs.umass.edu`, named `dns.umass.edu`.

DNS/Example (cont)



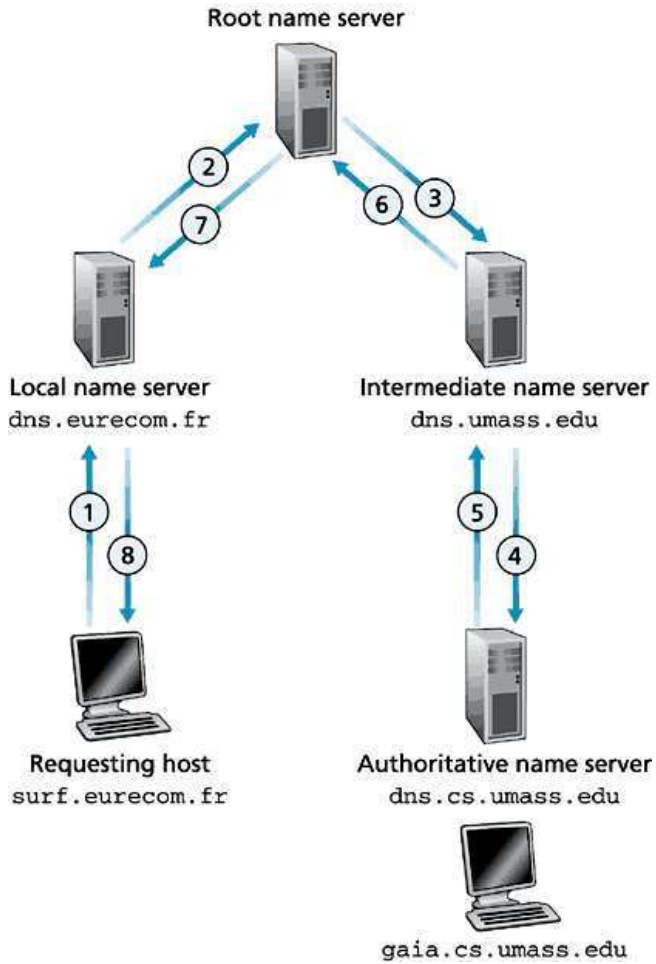
DNS/Example (cont)

Finally, the IP address of host `gaia.cs.umass.edu` is forwarded back to host `surf.eurocom.fr`, through all the DNS clients (steps 4, 5 and 6).

Until now we assumed that a root name server always knows an authoritative name server for the requested host. In fact, this is not always true: it usually knows an **intermediary name server**, which may know an authoritative name server or another intermediary name server.

Let us reconsider the previous example and assume that the root name server does not know an authoritative name server for `gaia.cs.umass.edu` but, instead, a name server for the whole domain (the university) `umass.edu`, called `dns.umass.edu`. This intermediary name server knows all the name servers for all the departments, in particular `cs.umass.edu`.

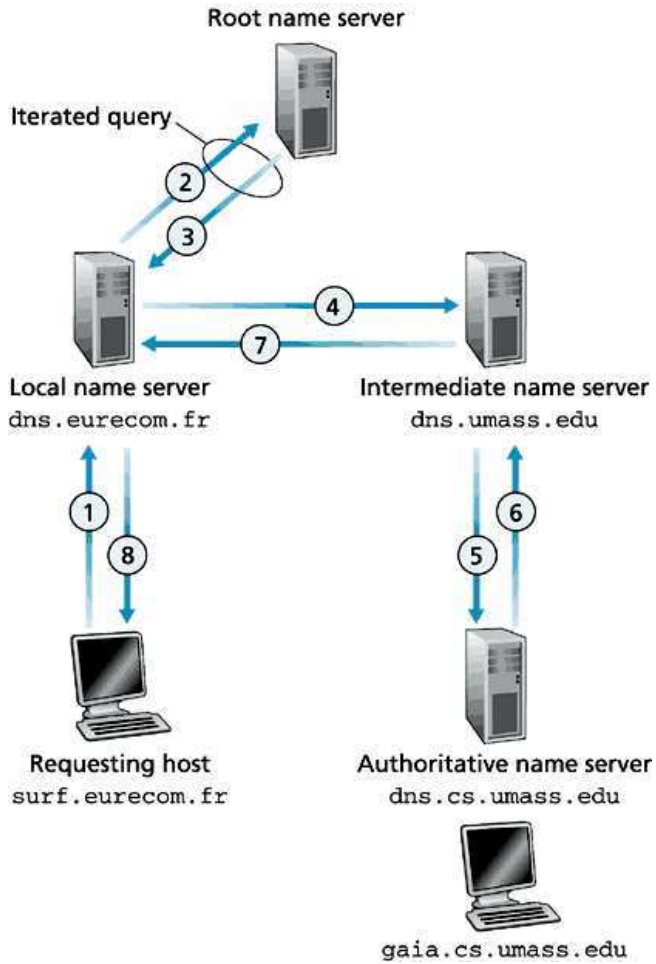
DNS/Example (cont)



DNS/Recursive and iterative queries

As in our example, when a name server A becomes a DNS client of name server B which, in turn, is client of C and that C sends the result to B, this is called a **recursive query** because B acts in behalf of A.
It is also possible that B responds to A with the IP address of name server C, leaving to A the task to *directly* query C: this is an **iterative query** — A makes all the queries.
It is possible to mix recursive and iterative queries for serving a single user's request.
Iterative queries save bandwidth and processing resources in some name servers (here, B), that is why it is a good idea to use them to ease the burden of root name servers (here, B).

DNS/Recursive and iterative queries (cont)



DNS/Caching

In order to improve the performances, name servers make use of **caches**.
When a name server receives a translation query and, later, the corresponding IP address, it copies the pair (hostname, address) in its memory or saves it on its local disk — as well as serving the result to its client.
Therefore, the next time the same query is received, the name server can answer just but by searching its cache.
In order to cope with ephemeral hosts, mapping pairs are kept in the caches only for a pre-configured period of time (often set to two days).

DNS/Records

A (hostname, address) is embedded in a data structure called a **resource record** (**RR**). In turn, resource records are stored in databases in each name server.
Each DNS message (either query or response) carries one or more resource records.
More precisely, a resource record has the structure (*name*, *value*, *type*, *TTL*) where *TTL* is the **Time To Live** record; it determines the time at which a resource should be removed from the cache.
We will ignore the *TTL* field in the following description.

DNS/Records (cont)

The meaning of *name* and *value* depend on the field *type*:

- if *type* is *A*, then *name* is a hostname and *value* is the corresponding IP address. This is the standard hostname-to-address mapping. For example (`konkuk.ac.kr`, 202.30.38.143, *A*).
- if *type* is *NS*, then *name* is a domain (such as `umass.edu`) and *value* is the hostname of an authoritative name server for all the hosts in *name*. This kind of record is used to route DNS queries along in the query chain. For instance, (`umass.edu`, `dns.umass.edu`, *NS*).

DNS/Records (cont)

- if *type* is *CNAME*, then *value* is a canonical hostname for the alias hostname *name*. This record provide querying hosts the canonical name of a host. As an example, (`google.com`, `www.google.com`, *CNAME*).
- if *type* is *MX*, then *value* is the canonical name of the mail server *name*, like the record (`google.com`, `mail.google.com`, *MX*). A company can both have an *MX* record and a *CNAME* record, allowing the same alias both for its mail server and its web server: depending on the querying application, the corresponding record will be requested.

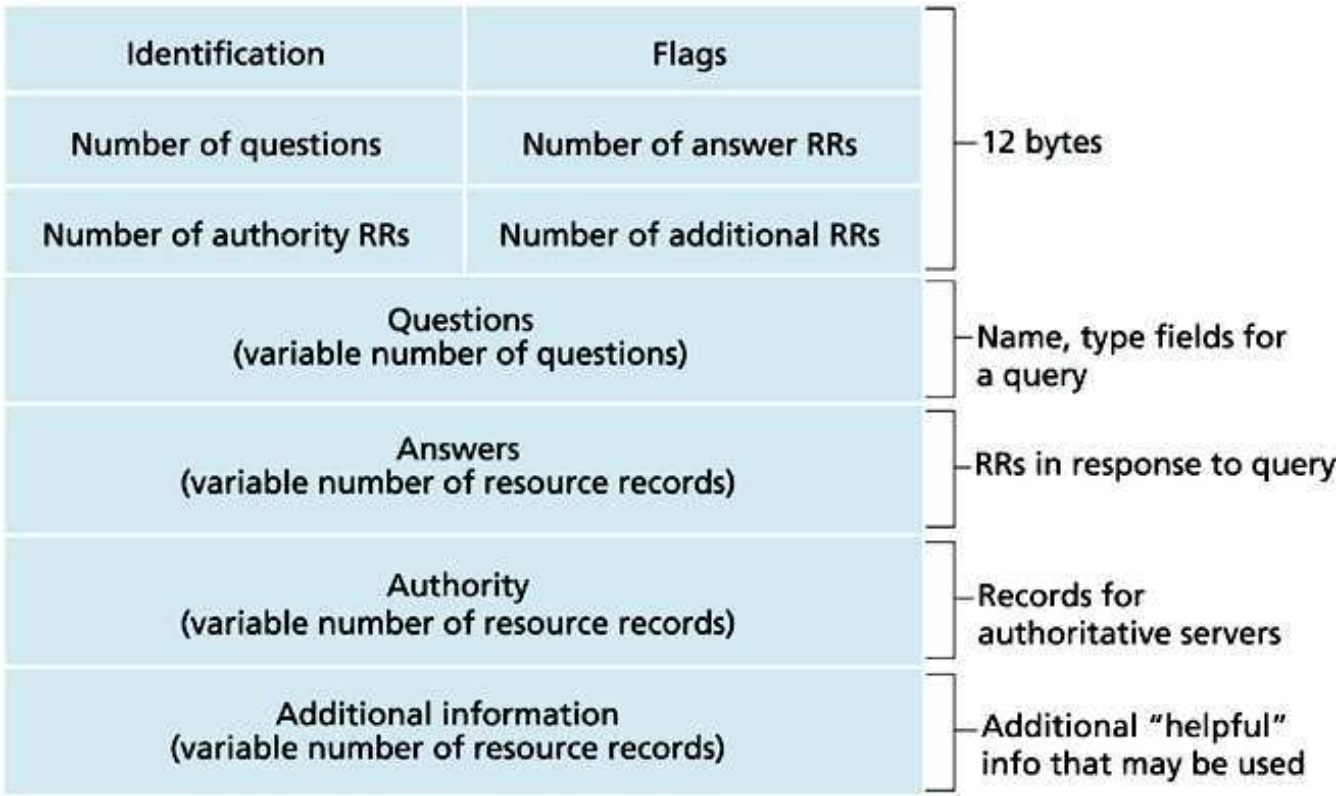
DNS/Records (cont)

Now we can better understand what is being an authoritative name server or not.

- If a name server is authoritative for a given hostname, then it must contain a *type A* record for this hostname.
- If a name server is not authoritative for a given hostname, it may contain a *type A* record in its cache. Otherwise it must contain a *type NS* record for the domain including the hostname, as well as a *type A* record giving the IP address of the domain name server given in the *type NS* record.
  For example, assume that a root name server is not authoritative for the hostname `gaia.cs.umass.edu` and no *type A* record is in cache for it. Therefore this name server must contain a record like (`umass.edu`, `dns.umass.edu`, *NS*) and (`dns.umass.edu`, 128.119.40.111, *A*).

DNS/Messages

There are only two kind of DNS messages and both queries and replies have the same format:



DNS/Messages (cont)

The first 12 bytes are the **header section**, which is made of successive fields:

1. the **identifier** is a 16 bits number that identifies a query and its answer;
2. the **flags** are one bit flags:
   - **query** is 0 and **reply** is 1;
   - **authoritative** name server or not;
   - **recursive query** required/not required (by client) or supported/not supported (by server);
3. the **numbers** are four numbers that count the number of occurrences of four kind of data in the next sections.

DNS/Messages (cont)

There are now four **data sections**:

1. the **question section** contains information about the current query:
   1.1 the **hostname** being queried;
   1.2 the type of query, e.g., *type A* or *type MX*;
2. the **answer section** contains information about the current reply: possibly multiple records (because a host can be duplicated);
3. the **authority section** contains records of other authoritative servers;
4. the **additional section** complements other informations, e.g., if a reply is an *MX* record, this section provides the IP address of the canonical mail server.