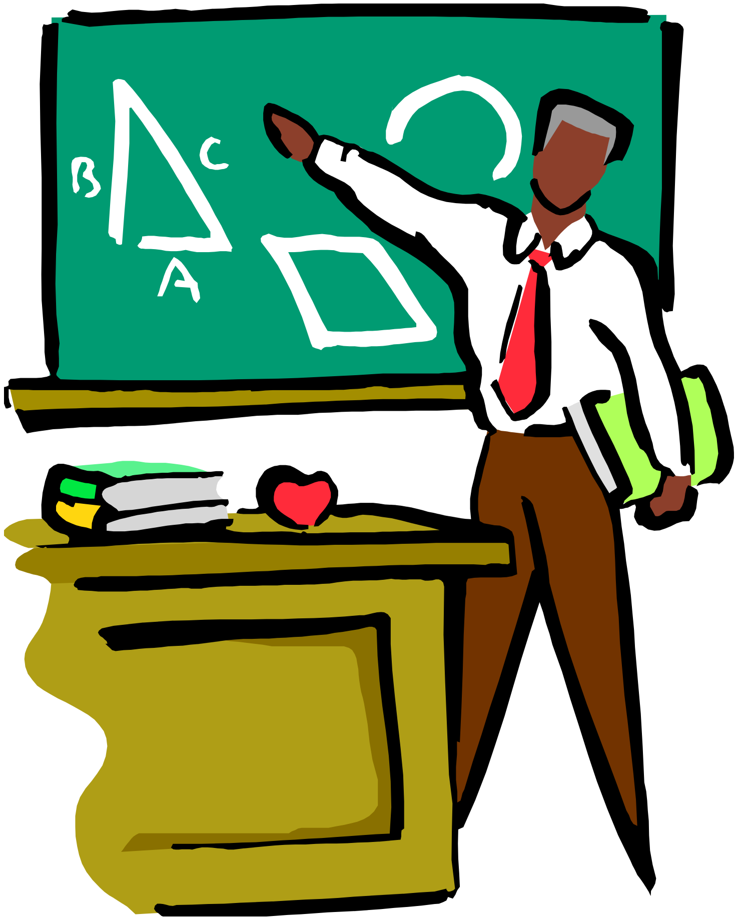
Chapter VII
Support Infrastructure for Application Layer: Distributed Name System DNS)

Support Infrastructure

Support infrastructure for application layer

- Why?
  - Re-usability across application layer protocols
  - Modularity (i.e. separation between application layer protocol specification / design and infrastructure specification / design)
- Examples discussed in this course
  - Distributed Name System (DNS)
    - Mapping between application layer symbolic addresses and IP addresses
  - Peer to peer overlays
    - Connectivity, routing and messaging between peers for applications such as file sharing, IP telephony

Domain Name System



- 1 – Conceptual Framework
- 2 - Implementation architecture

Conceptual Framework

Genesis

- Early 80s
- Replacement of the HOSTS.TXT file used in the early days of the Internet
  - HOSTS.TXT file
    - Contains list of all Internet host and mapping between symbolic names and IP addresses
    - Centrally managed and periodically distributed to all Internet hosts via file transfer
- Not sustainable with Internet growth
  - Scalability issues
  - Administrative issues (e.g. who will store and maintain this critical file?)

Conceptual Framework

Design goals / requirements

- Provide as a minimum all information provided by HOSTS.TXT
- Distributed storage and maintenance
- Flexible syntax for names and sizes of data associated with names
- Inter-operability across Internet
- Tolerable performance
- Extensibility
- Independence of network topology and operating systems

Conceptual Framework

Design goals / requirements

- Examples of implied design decisions
  - **Hierarchical name space**
    - Distributed storage and maintenance
    - Flexible syntax for names and sizes of data associated with names
  - **Data caching whenever possible**
    - Inter-operability across Internet
    - Tolerable performance
  - **Lean solution vs. comprehensive solution**
    - Extensibility

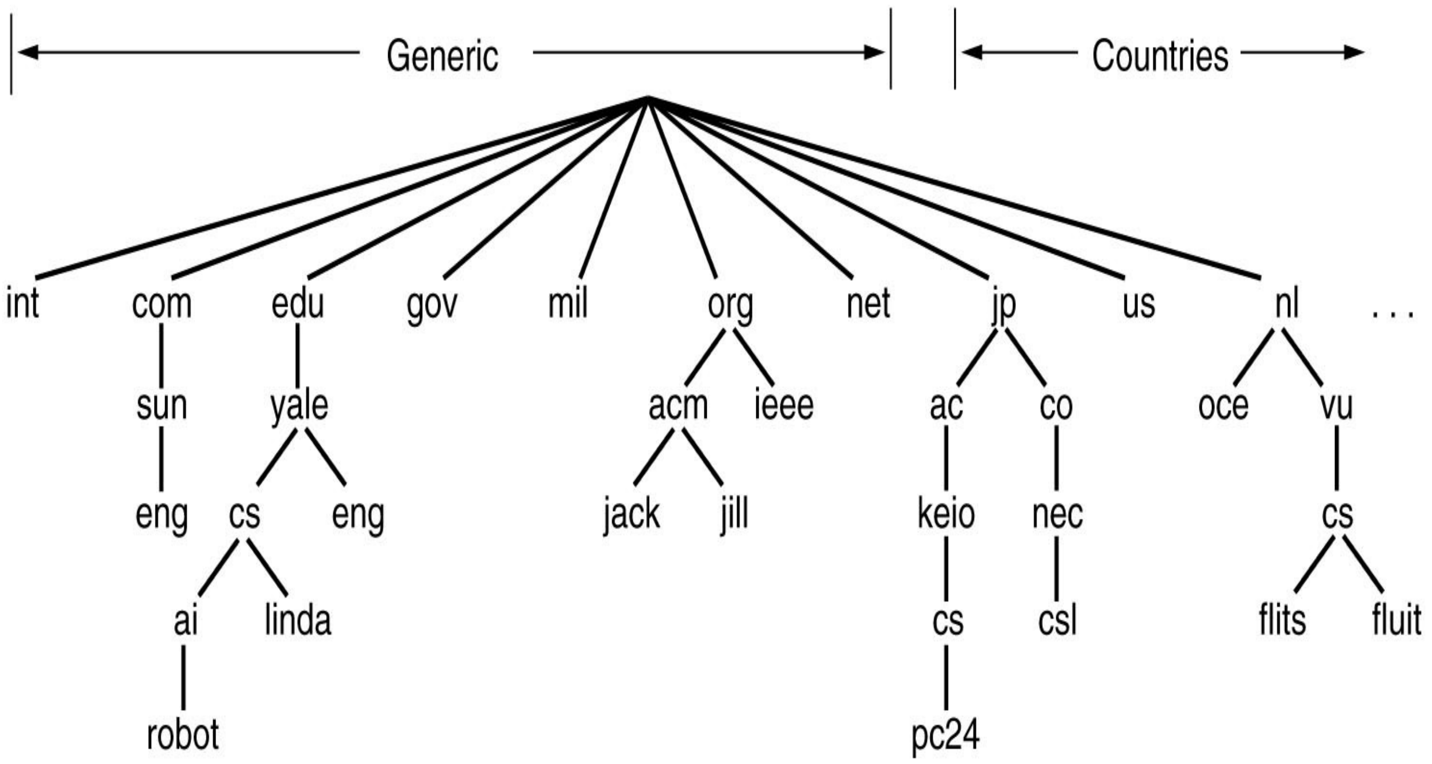
Conceptual Framework

Key concepts

- Domain name space
  - Variable depth tree with labelled nodes
    - Labels
      - » Variable length strings of octets
      - » Case insensitive
- Domain name of a node
  - Concatenation of all labels from the node to the root
- Administrative decision
  - Top levels correspond to:
    - Country codes
    - Broad organization types

Conceptual Framework

Domain name space



Conceptual Framework

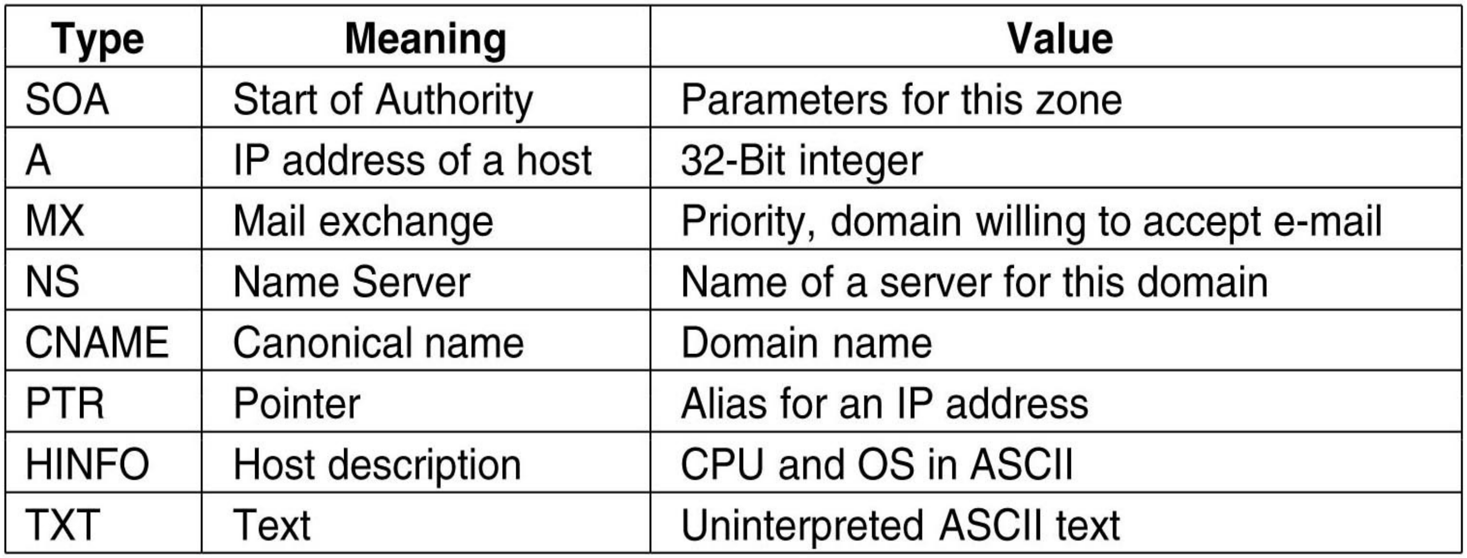
Key concepts

- Resource records (RRs)
  - Data attached to each name
    - Type
      - Abstract resource (e.g. host addresses)
    - Class field
      - Protocol family (e.g. DARPA Internet)
    - Application data
  - Authoritative record vs. cached record
    - Cached record may be out of date

Conceptual Framework

Key concepts

- Resource records (RRs)



Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept e-mail
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

Conceptual Framework

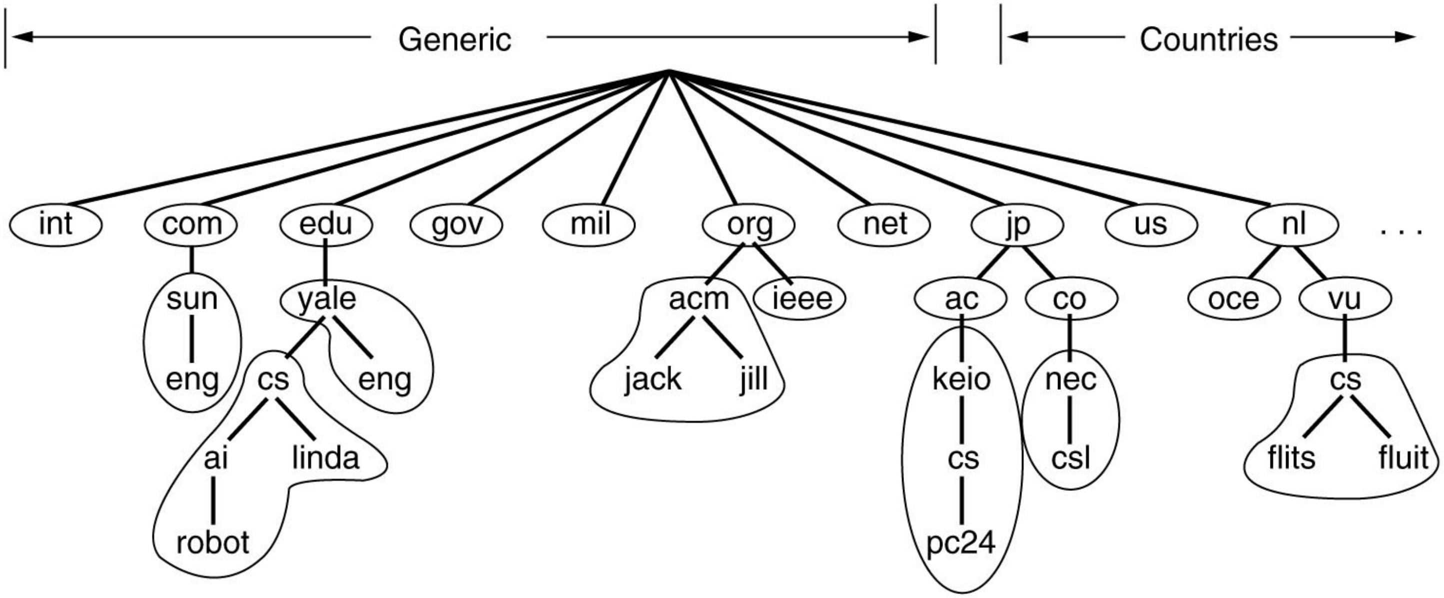
Key concepts

- Zones
  - Non overlapping sub-trees under the control of a single organization
    - Single nodes and whole tree are excluded
  - Procedure
    - Initial authorisation obtained from a parent organization for a single node
    - Growth to an arbitrary size without involving the parent organisation
  - Organization responsibilities
    - Maintenance of zone data
      - Ensure reliability through redundancy

Conceptual Framework

Key concepts

- Zones



Conceptual Framework

Key operations

- Zone transfers
  - Organizations make their zones available throughout the Internet
- Caching
  - Organizations cache whenever possible the zones they received from other organizations

Implementation Architecture

Client – server implementation

Functional entities

- Name server
  - Information repository
  - Does the actual mapping (i.e name resolution)
  - Can support any number of zones
    - Flexibility
      - » A name server for a given zone does not need to be in the zone
      - » Optimal distribution when distribution follows name space hierarchy

Implementation Architecture

Functional entities

- Resolver
  - Interface to client programs
    - Implement algorithms for finding the name server that has the required mapping information

Implementation Architecture

Implementation considerations

- Name server and resolver may be in separate boxes (nodes) or in a same node
  - Resolver is usually centralized in dedicated servers at organization levels
    - Re-use of cached information
    - No need for less powerful machines to implement their own resolving function

Implementation Architecture

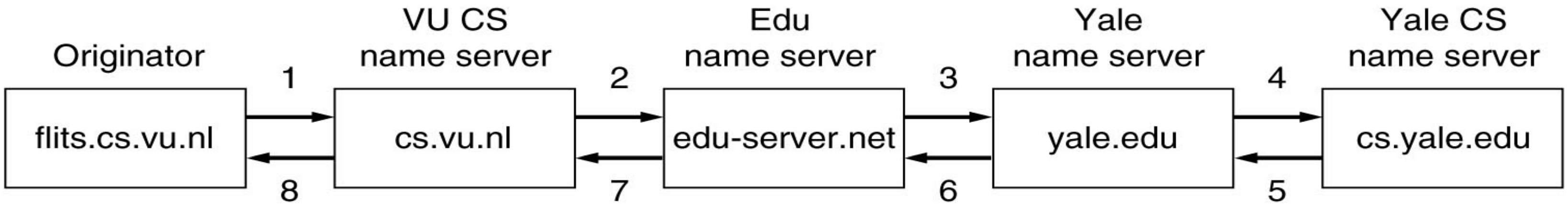
Implementation considerations

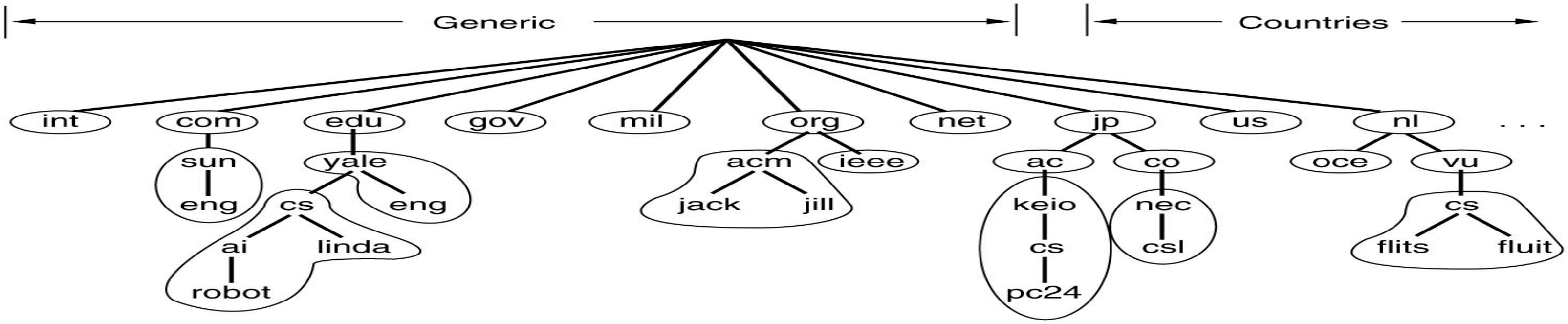
- Queries
  - Two approaches
    - Recursive (optional)
      - » When a name server does not have the requested information, it tries directly a name server that may have it
      - » Process is iterated till information found
    - Iterative
      - » When a name server does not have information, it returns to the resolver the name of the next name server on the path

Implementation Architecture

Implementation considerations

- Example of recursive look up (linda@cs.yale.edu)





Implementation Architecture

Implementation considerations

- Messages
  - Single format (Valid for both request and reply)

```
+---------------------+
|       Header        |
+---------------------+
|      Question       | the question for the name server
+---------------------+
|       Answer        | RRs answering the question
+---------------------+
|      Authority      | RRs pointing toward an authority
+---------------------+
|      Additional     | RRs holding additional information
```

Implementation Architecture

Implementation considerations

- Transport
  - UDP
    - Queries
      - Eventual retransmissions left up to applications (i.e. resolver and name server)
        - Guidelines provided
  - TCP
    - Zone related information (i.e. refresh)

Implementation Architecture

Problems with current implementation

- As for early 00s
- Backed by experience and experimentation
  - Vulnerability to network failures and Dos
    - Key reasons:
      - small number of name servers and limited redundancy
      - Known vulnerabilities in commonly deployed name servers
  - Performance issues
    - Name resolution latency
      - Reasons:
        - Low cache hit
        - Human errors (i.e. misconfigurations)

Implementation Architecture

Research on alternatives

- Rely mostly on P2P design instead of client / server design

References

1, P.V. Mockapetris and K. Dunlap, Development of the Domain Name System, ACM SIGCOMM Computer Review, 1995

2. P.V. Mockapetris, RFC 1034 and RFC 1035, November 1987
3. A. Tanembaum, Computer Networks, Chapter 7
4. V. Ramasubramanian and E. Sirer, The Design and Implementation of a Next Generation Name Service for the Internet, SIGCOMM'04, August 30 – Sept. 3,2004