Today
- Naming
  - Structured Naming
  - Attribute-based Naming

Structured Naming
- Structural organization of names
- Names are not independent
- Names are related to each other
- E.g.: file names, URLs

Name Space
- Typically hierarchical
  - Can be trees, acyclic graphs, etc.
  - E.g.: file systems, DNS
- Name types:
  - Global name: Name that can be used anywhere in the system
  - Local name: Name that requires context
  - Alias: Another name for an entity
- Closure mechanism: Selecting an initial node in the name space to start name resolution
Name Space Implementation

- Name space for a distributed system is itself distributed
  - Consists of multiple name servers
  - Each is responsible for one part of the name space
- Zone: part of name space maintained by a single name server
- Distribution of names is done hierarchically
  - Different layer for different levels in the hierarchy

Hierarchical Name Space Distribution

- Global layer: Root and its children
  - Organizations and groups of organizations
  - Relatively stable and long-lived
- Administration layer
  - Intra-Organization nodes
  - Departments, users, servers, etc.
- Managerial layer
  - Low-level nodes
  - Local hosts, filenames, usernames, etc.
  - Short-lived and frequently updated

Name Resolution

- Converting names to addresses
- Names are distributed
  - How do we locate appropriate name server?
- Two approaches:
  - Iterative
  - Recursive

Iterative Name Resolution
**Domain Name System (DNS)**

- Used for Internet host names
- Domain:
  - Subtree in the hostname space
  - Domain name: path to domain root
- Each DNS name server contains resource records
  - Name server address
  - Host IP address
  - Mail server address
  - Other information

**DNS Implementation**

- Hierarchical layer-based implementation
- Each zone has a primary name server
  - Updates made at primary server
  - Secondary servers transfer updates from primary
- Uses iterative name resolution
  - Caching at client name resolver

---

**Recursive Name Resolution**

Client Name Resolver

- A (root)
- B (umn.edu)
- C (cs.umn.edu)

**Domain Name System (DNS)**

- Example DNS Entry

<table>
<thead>
<tr>
<th>Name</th>
<th>Record type</th>
<th>Record value</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1.cs.umn.edu</td>
<td>A</td>
<td>192.31.231.44</td>
</tr>
<tr>
<td>b1.cs.umn.edu</td>
<td>A</td>
<td>192.31.231.44</td>
</tr>
<tr>
<td>tpa.umn.edu</td>
<td>CNAME</td>
<td>a1.cs.umn.edu</td>
</tr>
<tr>
<td>csp.cs.umn.edu</td>
<td>A</td>
<td>192.31.231.44</td>
</tr>
<tr>
<td>root.umn.edu</td>
<td>NS</td>
<td>a1.cs.umn.edu</td>
</tr>
<tr>
<td>root.umn.edu</td>
<td>NS</td>
<td>b1.cs.umn.edu</td>
</tr>
<tr>
<td>root.umn.edu</td>
<td>NS</td>
<td>csp.cs.umn.edu</td>
</tr>
<tr>
<td>root.umn.edu</td>
<td>NS</td>
<td>x.y.z.w</td>
</tr>
<tr>
<td>root.umn.edu</td>
<td>NS</td>
<td>x.y.z.w</td>
</tr>
<tr>
<td>root.umn.edu</td>
<td>NS</td>
<td>x.y.z.w</td>
</tr>
<tr>
<td>root.umn.edu</td>
<td>NS</td>
<td>x.y.z.w</td>
</tr>
</tbody>
</table>

**DNS Implementation**

- Hierarchical layer-based implementation
- Each zone has a primary name server
  - Updates made at primary server
  - Secondary servers transfer updates from primary
- Uses iterative name resolution
  - Caching at client name resolver
**Attribute-based Naming**
- Identify an entity by its description
- Want an entity of certain kind
- Name specified by (attribute, value) pairs
- Examples:
  - A laser printer on 4th floor in CS dept
  - A mail server for cs.umn.edu domain
- Also called Directory Service
- Can be implemented using:
  - Structured/hierarchical naming
  - Flat/decentralized naming

**LDAP**
- Light-Weight Directory Access Protocol
  - Simplified version of X.500 Directory Service
  - Uses a hierarchical naming scheme
- Directory entries or records:
  - Contain (attribute, value) pairs
- Several implementations:
  - MS Active Directory, Novell Directory services, openLDAP

**Hierarchical Naming**
- Directory entries are arranged in a tree structure
  - Each node has its own record
  - Each node also has pointers to its children records
- Distinguished name (DN):
  - Each record has a unique global name
  - Sequence of Attribute-value pairs
  - E.g.: /C=US/O=UMN/OU=CS
- Relative Distinguished name (RDN):
  - Name w.r.t. parent’s DN

**LDAP Implementation**
- Directory Service Agents (DSA)
  - Maintain entries for part of the naming tree
  - Similar to DNS name servers
- Directory User Agents (DUA)
  - Client-side name resolvers
- LDAP supports more advanced queries
  - Search operations on attributes
  - Use indexes, filtering, pruning
- LDAP combined with DNS
  - Root directory node of a tree accessible through DNS
Decentralized Implementation
- Use DHTs for attribute-based naming
  - Map (attribute, value) pairs to identifiers
- Two types of queries:
  - Single-value queries: Each attribute has a single desired value
  - Range queries: Each attribute can have a range of values

Single-Value Queries
- Hierarchical naming structure
  - Similar to LDAP
- Attribute-value Tree (AVT)
  - Used to represent each entity
  - Links: attributes
  - Nodes: values
  - Path: a sequence of links and nodes from the root

DHT Conversion
- Where to keep the record of an entity?
- Hash each path of record’s AVT to a key
  - Servers with corresponding hashes keep the record
- Query:
  - Produce an AVT of the query
  - Hash each path in the AVT
  - Contact the corresponding servers

Range Queries
- Each attribute can take a range of values
  - E.g.: Find a machine s.t. CPU in [500-1000 MHz], RAM in [256-512 MB]
- Convert each (attribute, value) pair to key:
  - Partition key by attribute (n bits) and then by value (m bits)
  - k=h1(attr).h2(val)
  - h1, h2 hash functions
- Server corresponding to key k holds the record for the entity
Resolving Range Queries

- Servers partitioned by:
  - Attributes: Those with same h1
- Value ranges partitioned among servers
- Looking for attr a and value range [v1,v2]
  - Among servers hashing to "a":
    - Find subset hashing to [v1,v1'],[v1',v1''], [...,v2]
- What if searching on multiple attribute-value ranges?
- How many servers need to be updated?