CSci 5105

Introduction to Distributed Systems

Naming
Today

• Naming continued
• Chapter 5 TVS, Active Names paper
Attribute-Based Naming

- Name is attribute-value pairs
- Sometimes called directory services vs. naming services

(color:blue), (size:large), ...
Example: LDAP

• Lightweight directory access protocol

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Abbr.</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Country</td>
<td>C</td>
<td>NL</td>
</tr>
<tr>
<td>Locality</td>
<td>L</td>
<td>Amsterdam</td>
</tr>
<tr>
<td>Organization</td>
<td>O</td>
<td>Vrije Universiteit</td>
</tr>
<tr>
<td>OrganizationalUnit</td>
<td>OU</td>
<td>Comp. Sc.</td>
</tr>
<tr>
<td>CommonName</td>
<td>CN</td>
<td>Main server</td>
</tr>
<tr>
<td>Mail_Servers</td>
<td>—</td>
<td>137.37.20.3, 130.37.24.6, 137.37.20.10</td>
</tr>
<tr>
<td>FTP_Server</td>
<td>—</td>
<td>130.37.20.20</td>
</tr>
<tr>
<td>WWW_Server</td>
<td>—</td>
<td>130.37.20.20</td>
</tr>
</tbody>
</table>

• Name is: /C=NL/O=Vrije Universiteit/O=Comp. Sc.
Hierarchical Implementations: LDAP

- Directory Tree

```
C = NL
  ↓
O = Vrije Universiteit
    ↓
OU = Comp. Sc.
      ↓
CN = Main server
        ↓
N
  Host_Name = star
  Host_Name = zephyr
```
Decentralized: DHT

description {
  type = book
  description {
    author = Tolkien
    title = LOTR
  }
  genre = fantasy
}

H1: Hash (type-book)
H2: Hash (type-book-author)

Challenge: ranges, or + and
Naming: Active Names

Active Names: Flexible Location and Transport of Wide-Area Resources

Vahdat, Dahlin, Anderson, Aggarwal
The Problem

• Accessing remote resources and services is limited by rigid naming schemes
  – need a way to insert flexible (i.e. customizable) services between clients and servers
  – current solutions: done inside the network elements or deferred to application
  – either client-side or strictly server-side
Traditional Internet Naming

Traditional Model

- Static name -> IP address binding
- Naming and transport separate
  - http AND www.cnn.com
- Not flexible or extensible
Motivation

• Consider Scenario: context-sensitive naming
  - User types cnn.com
  - If client is behind a modem, it gets back a b/w image
  - If client is a palm pilot, it gets a distilled image
  - If the client is in Europe, it goes to the European replica

• Combine naming and transport in one framework

• Provide flexibility and extensibility in the way wide area resources are accessed
Current Attempts to Add Flexibility to Name Binding

- HTTP redirect
- DNS round robin
- Cisco Local Director/Distributed Director
- Global object IDs (e.g., Globe, Legion)
- Web caches
- Mobile IP
- … none of them are programmable
**Active Names: Basic Idea**

- Names resolved to mobile, secure programs

**Flexibility**

- Active Names organized into hierarchical *namespaces*. A program is associated with each namespace

**Extensibility**

- Namespace programs can be changed

**Active Names are connection oriented: better end-to-end semantics and performance**
- Location independent programs
  - Programs may run on any AN node
- Stream data model
  - Each program operates on a data stream which is the result of the previous program
- Continuation passing style
  - Control does not have to return to the caller program
Performance Gains

• Application customized transport protocols

• Programs are location independent. Location can be chosen to optimally utilize resources (e.g., distillation)

• Customization can be performed close to client instead of at the server (e.g., to cache dynamic content)

• 3 way RPC
Composing Services

• **Delegation**
  – Active names organized in a hierarchy of namespaces
    
    ![Namespace Diagram]
    
    – Namespace programs can delegate to subordinate namespaces

• **After methods**
  – Continuation passing style programming
  – Namespace programs bundle remaining work into “after methods” before passing control
Security

• Protection between active name programs provided by Java’s type safety mechanism

• Caller passes a certificate to the callee granting it a subset of its rights

• For instance, each caller might grant its callee the right to respond to the client

• Certificates are authenticated via encryption
Active Node Architecture

Bootstrap programs

Distiller

Http  DNS  ...

Resolver  Local Resources

Namespaces

Virtual Machine
Application 1: Replica Selection

- DNS Round-Robin
  - Randomly choose replica
  - Avoid hot-spots
- Distributed Director
  - Route to nearest replica
  - Geographic locality
- Active Naming
  - Previous performance, distance
  - Adaptive
Replica Selection

The graph shows the response time (in seconds) for different offered load levels. The response time increases as the offered load increases, with the Nearest selection method having the highest response time, followed by Random and Active. The y-axis represents the response time, while the x-axis represents the offered load.
Application 2: Mobile Distillation

- Clients name a single object
- Returned object based on client
  - Network connection, screen
- Current approach: proxy maintains client profile
  - Requests object, distills
- Active naming
  - Transmit name + applet
  - Flexible distillation point
  - Tradeoff computation/bandwidth
  - Support mobile clients

Variables:
- Network
- Screen
Application 2: Mobile Distillation

Distillation at

- **Server**: Saves bandwidth
- **Proxy**: Saves server CPU cycles
- **Active**: Cost estimate of both approaches

![Distillation Latency Graph](image)

- **Active**: Light blue bars
- **Server**: Purple bars
- **Proxy**: Yellow bars
Summary

• Active name paradigm
  – Decouples name from location
  – Allows specialized processing “in the network” based on client/server conditions
Next topic: Synchronization, Mutual Exclusion
Read Chapter 6 TVS

Have great weekend!