

# Distributed Systems

## 4. Naming and Binding

Paul Krzyzanowski  
pxk@cs.rutgers.edu

- 
- My 15" MacBook Pro
  - The rightmost computer on my desk
  - Paul's aluminum laptop, but not the big or the small one
  - hedwig
  - hedwig.pk.org
  - 192.168.60.148
  - 00:14:51:ec:f2:5b

# Naming things

---

- User names
  - Login, email
- Machine names
  - rlogin, email, web
- Files
- Devices
- Variables in programs
- Network services

# Naming Service

---

Allows you to look up **names**

- Often returns an **address** as a response

Might be implemented as

- Search through file
- Client-server program
- Database query
- ...

# What's a name?

---

**Name:** identifies what you want

**Address:** identifies where it is

**Route:** identifies how to get there

**Binding:** associates a name with an address

- “choose a lower-level-implementation for a higher-level semantic construct”

*RFC 1498: Inter-network Naming, addresses, routing*

# Names

---

We may need names for:

- **Services**: e.g., time of day
- **Nodes**: computer that can run services
- **Paths**: route
- **Objects within service**: e.g. files on a file server

Naming convention can take any format

- Ideally one that will suit application and user
- E.g., human readable names for humans, binary identifiers for machines

# Uniqueness of names

---

- Easy on a small scale
- Problematic on a large scale
  - Globally unique names are hard to keep unique
- Hierarchy allows uniqueness to be maintained
  - Globally unique components
  - E.g.:
    - Ethernet MAC address: 3 bytes: organization, 3 bytes: controller
    - IP address: network & host
  - **compound name**: set of atomic names connected with a name separator
    - Domain name (www.rutgers.edu), URLs

# Terms: Naming convention

---

Naming system determines syntax for a name

- Unix file names:

  - Parse components from left to right separated by /  
`/home/paul/src/gps/gui.c`

- Internet domain names:

  - Ordered right to left and delimited by .  
`www.cs.rutgers.edu`

- LDAP names

  - Attribute/value pairs ordered right to left, delimited by ,  
`cn=Paul Krzyzanowski, o=Rutgers, c=US`

# Terms: Context

---

- A particular set of **name** → **object** bindings
- Each context has an associated naming convention
- A name is *always* interpreted relative to some context
  - E.g., directory **/usr** in a **UNIX** file system

# Terms: Naming System

---

Connected set of contexts of the same type (same naming convention) along with a common set of operations

For example:

- System that implements DNS
- System that implements LDAP

# Terms: Name space

---

Set of names in the naming system

For example,

- Names of all files and directories in a UNIX file system
- All domain names on the Internet

# Terms: Resolution

---

## Name lookup

- Return the underlying representation of the name

## For example,

- `www.rutgers.edu` → `128.6.4.5`

# Directory Service

---

## Extension of naming service:

- Associates names with objects
- Allows objects to have attributes
- Can search based on attributes

## For example,

- LDAP (Lightweight Directory Access Protocol)
- Directory can be an object store:
  - Look up printer object and send data stream to it

# Name resolution

---

To send data to a service:

1. Find a node on which the service resides (service name resolution)
2. Find an address (or network attachment point) for that node (node name location)
3. Find a path from this location to the service (routing service)

# Name resolution

---

E.g., access “paul’s service”:

**File lookup:** (service name to service address)

“paul’s service” → cs.rutgers.edu:1234

**DNS lookup:** (domain name to IP address)

cs.rutgers.edu → 128.6.4.2

**ARP resolution:** (IP name to MAC address)

128.6.4.2 → 08:00:20:90:9c:23

**IP routing:** (*dynamic for IP*)

route: remus → lcsr-gw → aramis

# Binding

---

The association of a resolution

## **Static binding**

- Hard-coded

## **Early binding**

- Look up binding before use
- Cache previously used binding

## **Late binding**

- Look up just before use

# IP Domain Names

---

Human readable names

e.g. `aramis.rutgers.edu`

Hierarchical naming scheme

- No relation to IP address or network class

# Example: DNS

---

## Internet Domain Name Service

- Maps machine names (`www.rutgers.edu`) to IP addresses (`128.6.4.5`)

## In the past:

- Search `/etc/hosts` for machine name
- File periodically downloaded from Network Information Center (NIC) at the Stanford Research Institute (SRI)

# Internet Domain Name Space

---

## Tree structure

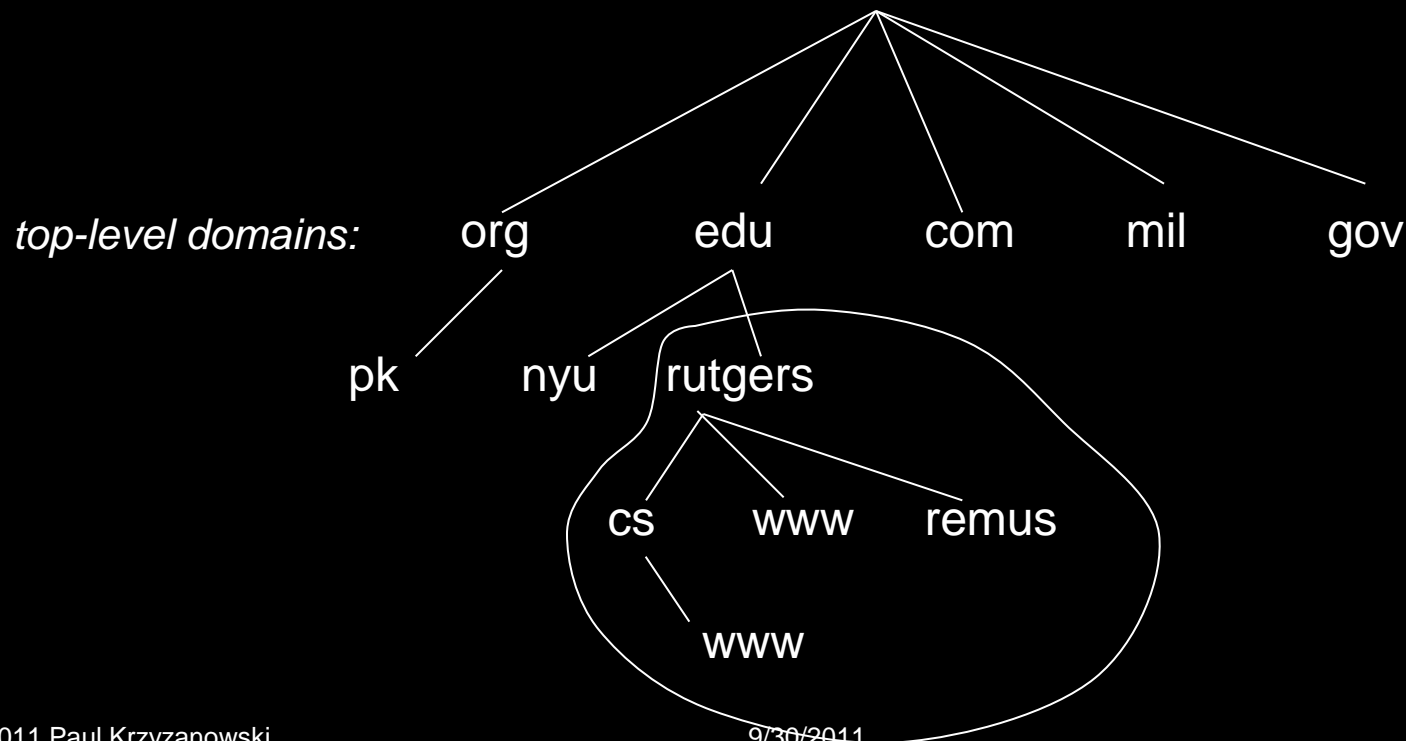
- Each node has resource information associated with it
- **owner**: domain name whose resource record is found
- **type of resource**:
  - Host address (A)
  - Alias name (C)
  - Name server for domain (NS)
  - Mail server (MX)
- **TTL** (time to live) → *for caching*
- **Relevant data** (e.g., address)

# Domain Name Server

---

## Essential task

Answer queries about data in its **zone**  
(group of machines under a root – e.g. rutgers)



# Sample Query

---

- Rutgers registers **rutgers.edu** with a domain registry
  - educause.net for .edu domain
  - See internic.net for ICANN-accredited list of registrars for top-level domains
- Top-level domain names and their associated name server info loaded to **root name servers**
  - 13 computers: replicated information
  - Contain addresses for all registries of top-level domains (.com, .edu, .org, ...)

# Sample Query

---

Submit query to a local *DNS resolver*:

1. query(cs.rutgers.edu) → root name server  
root name servers identify authoritative servers for top-level domains  
send query to A.ROOT\_SERVERS.NET: 198.41.0.4
2. *referral* to edu name server  
returns list of DNS servers for .edu:  
L3.NSTLD.COM: 192.41.162.32
3. query(cs.rutgers.edu) → edu name server  
send query to 192.41.162.32
4. *referral* to rutgers.edu name servers:
  - DNS1.rutgers.edu 165.230.144.131
  - DNS2.rutgers.edu 128.6.21.9
  - DNS3.rutgers.edu 198.151.130.254
5. query(cs.rutgers.edu) → rutgers name server  
send query to 165.230.144.131
6. rutgers name server returns  
A: 128.6.4.2 address  
MX: dragon.rutgers.edu domain name for email

# Iterative vs. Recursive name resolution

---

- Referrals force iterative name resolution
  - Send query to root name server
  - Send query to edu name server
  - Send query to rutgers name server
  - *Advantage: stateless*
- Recursive:
  - Send query to root name server
  - Root name server sends query to edu name server
  - Edu name server sends query to rutgers name server
  - *Advantage: increased caching opportunities, reduced communication*

# DNS

---

## BIND

- Implementation of DNS provided by the Internet Software Consortium ([www.isc.org](http://www.isc.org))

## Programs to perform queries:

- dnsquery, nslookup, dig, host

# Naming: files

---

File system maps file pathname

/home/paul/src/map.c



*namei* in kernel

major=3, minor=6, inode=6160

The End