
Naming: Hierarchical, Attribute, Mutual Exclusion

15 December 2024
Lecture 7

Slide Credits: Maarten van Steen

Topics for Today

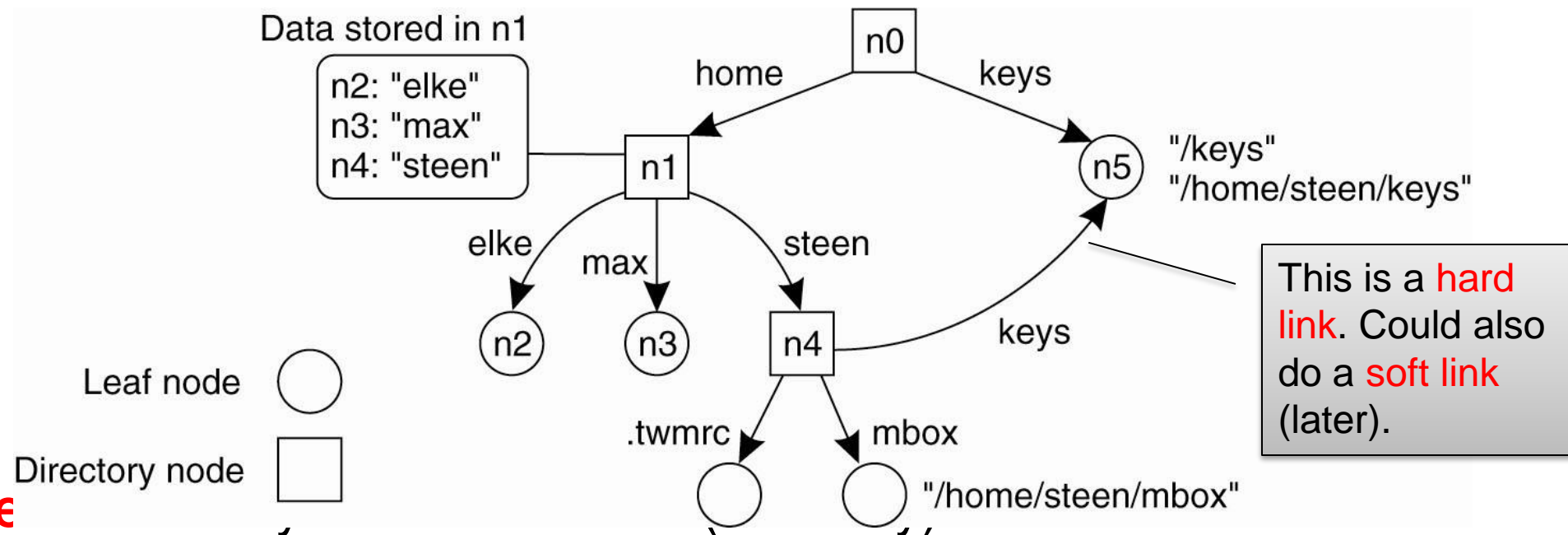
- Structured Naming
 - Name Spaces
 - Name Resolution
 - Domain Name System (DNS)
- Attribute Based Naming
- Mutual Exclusion

Sources:

- TvS 6.1-6.5, 7.1-7.3

File System Name Space

- **Essence**: a graph in which a **leaf node** represents a (named) entity. A **directory node** is an entity that refers to other nodes.



- **Note**
(*edge label, node identifier*) pairs.

File System Name Space

Observation: We can easily store all kinds of **attributes** in a node, describing aspects of the entity the node represents:

- Type of the entity
- An identifier for that entity
- Address of the entity's location
- Nicknames
- ...

Observation: Directory nodes can also have attributes, besides just storing a directory table with *(edge label, node identifier)* pairs.

Name Resolution

Problem: To resolve a name we need a directory node. How do we actually find that (initial) node?

Closure mechanism: The mechanism to select the implicit context from which to start name resolution:

- `www.kinneret.ac.il`: start at a DNS name server
- `/home/users/mjmay`: start at the local NFS file server (possible recursive search)
- `0097246653793`: dial a phone number
- `212.150.112.29`: route to Kinneret's Web server

Question: Why are closure mechanisms always **implicit**?

Observation: A closure mechanism may also determine how name resolution should proceed

Name Linking

Hard link: What we have described so far as a **path name**: a name that is resolved by following a specific path in a naming graph from one node to another.

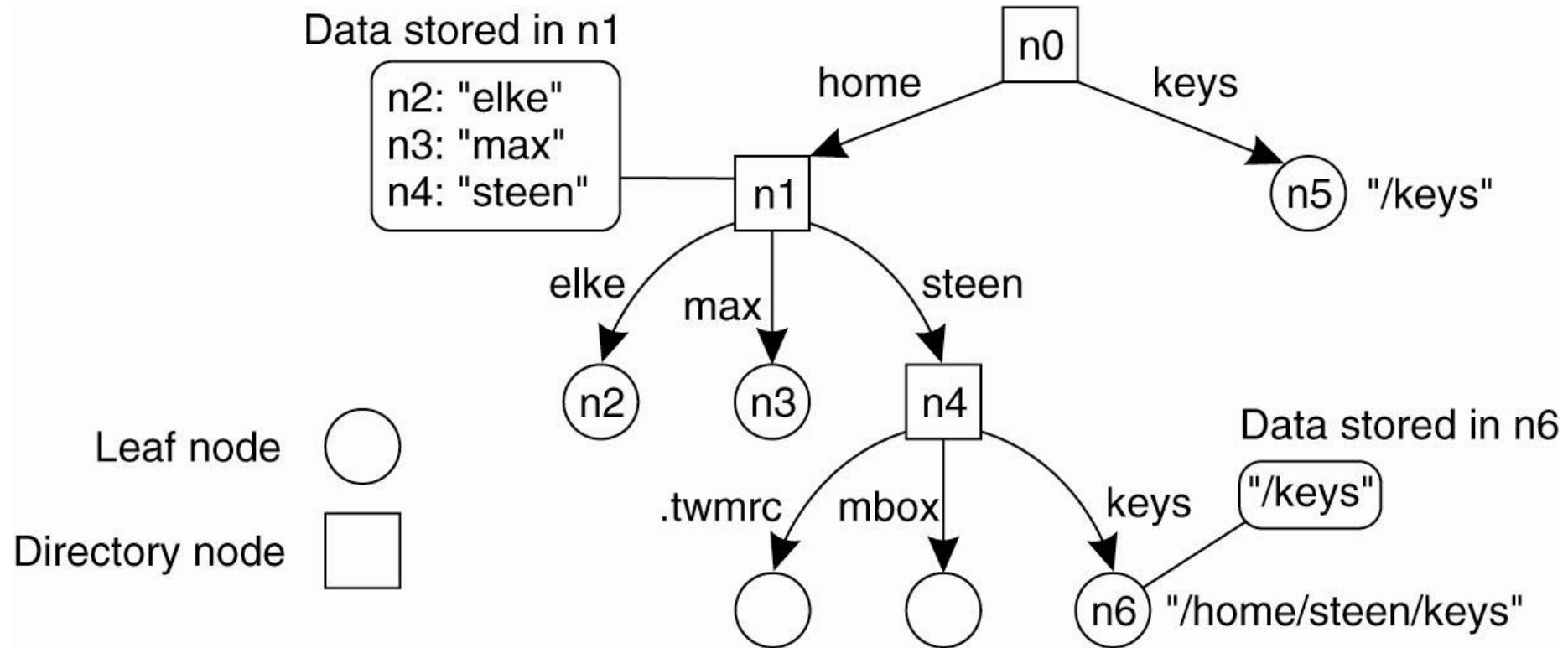
Soft link: Allow a node O to contain a **name** of another node:

- First resolve O 's name (leading to O)
- Read the content of O , yielding *name*
- Name resolution continues with *name*

Observations:

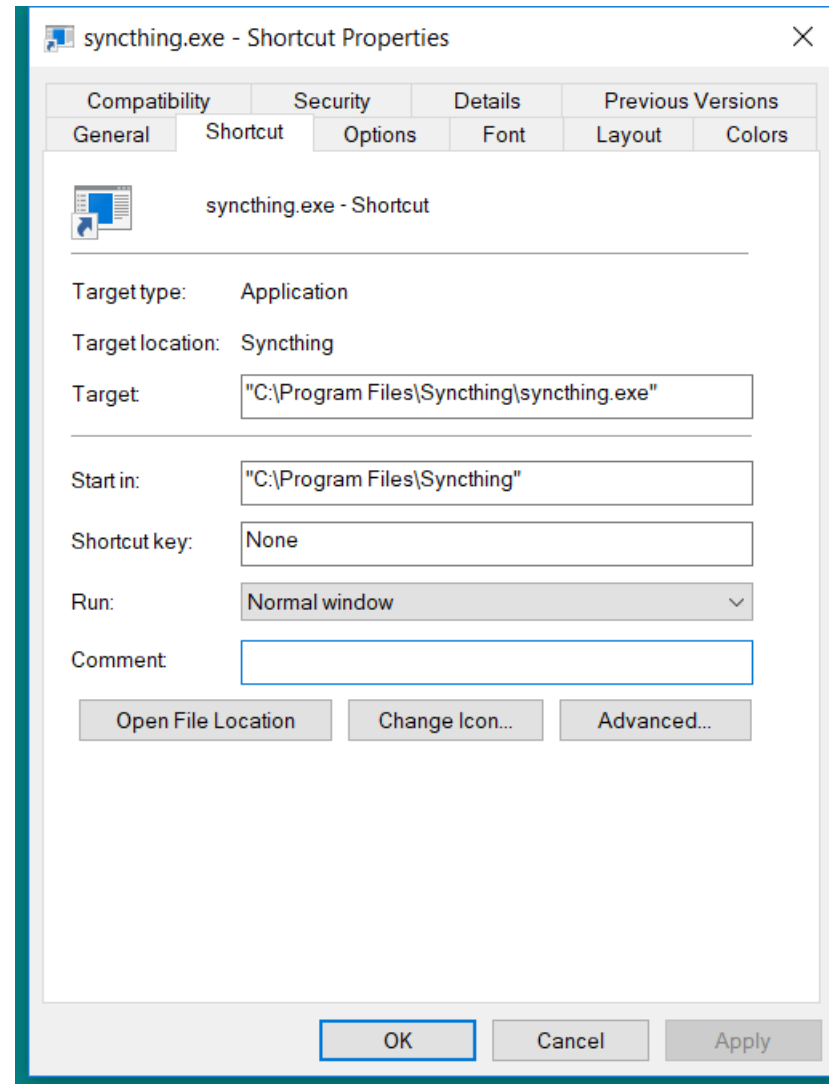
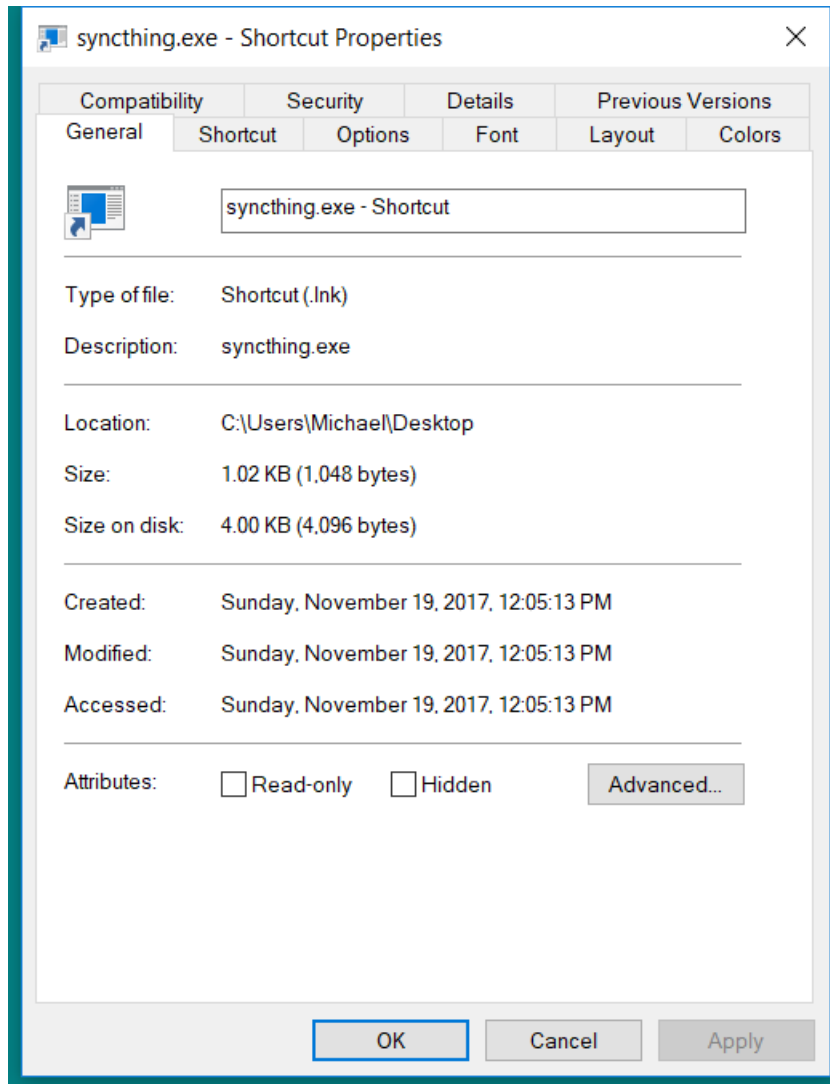
- The name resolution process determines that we read the **content** of a node, in particular, the name in the other node that we need to go to.
- One way or the other, we know where and how to start name resolution given *name*

Soft Linking



























Observation: Node n5 has only one name

Soft Linking (Windows)



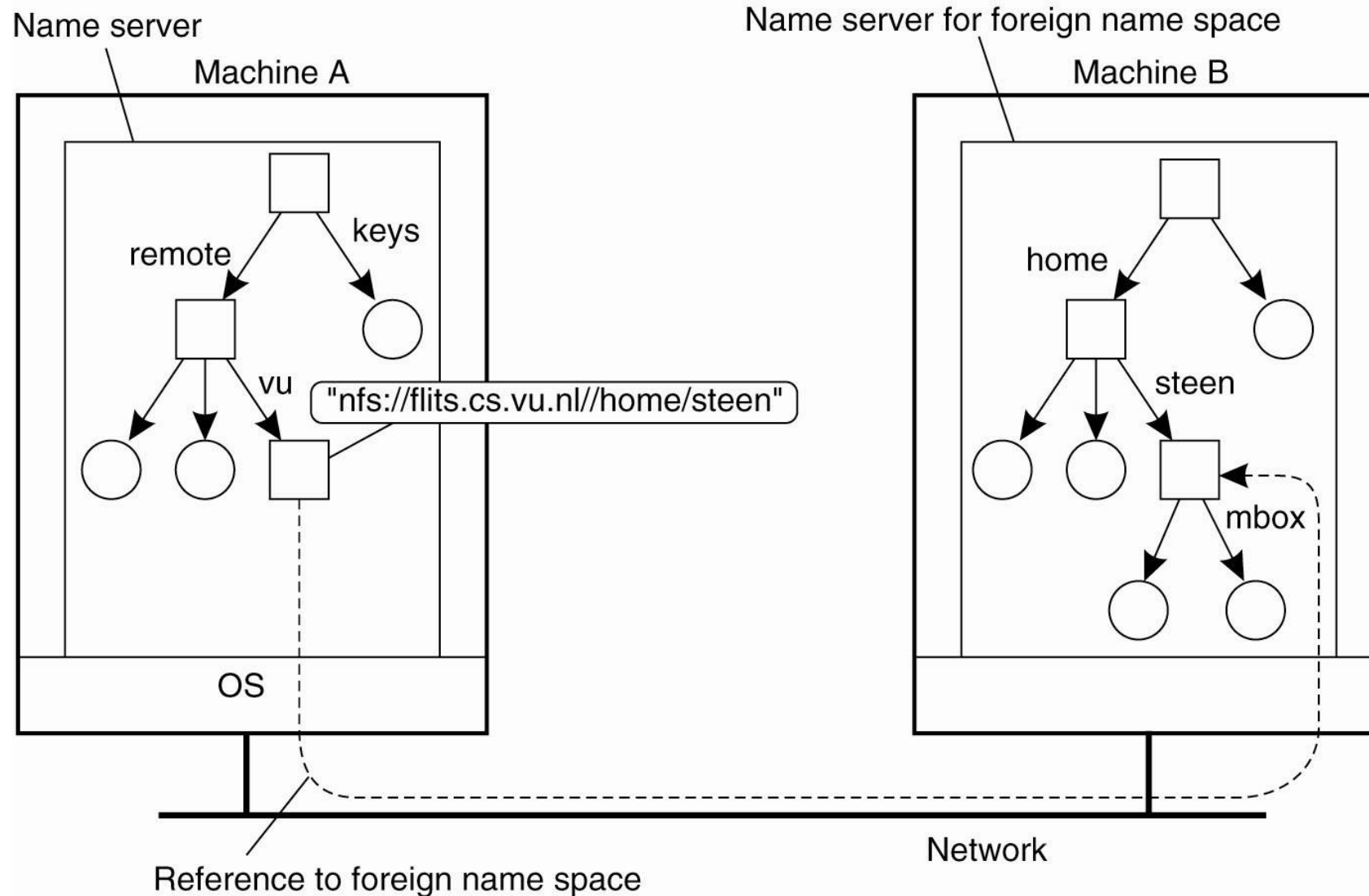
Soft Linking (Android)

Name	Permissions	Date	Size
>  acct	dr-xr-xr-x	2021-12-15 14:32	0 B
>  apex	drwxr-xr-x	2021-12-15 14:32	920 B
>  bin	lrw-r--r--	2009-01-01 00:00	11 B
>  bugreports	lrw-r--r--	2009-01-01 00:00	50 B
>  config	drwxr-xr-x	2021-12-15 14:32	0 B
>  data	drwxrwx--x	2021-12-15 14:33	4 KB
>  debug_ramc	drwxr-xr-x	2009-01-01 00:00	4 KB
>  dev	drwxr-xr-x	2021-12-15 14:32	1.3 KB
>  etc	lrw-r--r--	2009-01-01 00:00	11 B
>  lost+found	drwx-----	2009-01-01 00:00	16 KB
>  mnt	drwxr-xr-x	2021-12-15 14:32	320 B
>  odm	drwxr-xr-x	2009-01-01 00:00	4 KB
>  oem	drwxr-xr-x	2009-01-01 00:00	4 KB
>  proc	dr-xr-xr-x	2021-12-15 14:32	0 B
>  product	drwxr-xr-x	2009-01-01 00:00	4 KB
>  res	drwxr-xr-x	2009-01-01 00:00	4 KB
>  sdcard	lrw-r--r--	2009-01-01 00:00	21 B
>  storage	drwx--x---	2021-12-15 14:32	80 B
>  sys	dr-xr-xr-x	2021-12-15 14:32	0 B
>  system	drwxr-xr-x	2009-01-01 00:00	4 KB
>  system_ext	drwxr-xr-x	2009-01-01 00:00	4 KB
>  vendor	drwxr-xr-x	2009-01-01 00:00	4 KB
 d	lrw-r--r--	2009-01-01 00:00	17 B
 default.prop	lrw-----	2009-01-01 00:00	23 B

Soft Linking (Android)

```
d????????? ? ? ? ? ? metadata
d????????? ? ? ? ? ? linkerconfig
-????????? ? ? ? ? ? init.environ.rc
l????????? ? ? ? ? ? init -> ?
d????????? ? ? ? ? ? data_mirror
l????????? ? ? ? ? ? cache -> ?
drwxr-xr-x 8 root shell 4096 2009-01-01 00:00 vendor
drwxr-xr-x 9 root root 4096 2009-01-01 00:00 system_ext
drwxr-xr-x 11 root root 4096 2009-01-01 00:00 system
lrw-r--r-- 1 root root 21 2009-01-01 00:00 sdcard -> /storage/self/primary
drwxr-xr-x 3 root root 4096 2009-01-01 00:00 res
drwxr-xr-x 10 root root 4096 2009-01-01 00:00 product
drwxr-xr-x 2 root root 4096 2009-01-01 00:00 oem
drwxr-xr-x 2 root root 4096 2009-01-01 00:00 odm
drwx----- 2 root root 16384 2009-01-01 00:00 lost+found
lrw-r--r-- 1 root root 11 2009-01-01 00:00 etc -> /system/etc
lrw----- 1 root root 23 2009-01-01 00:00 default.prop -> system/etc/prop.default
drwxr-xr-x 2 root root 4096 2009-01-01 00:00 debug_ramdisk
lrw-r--r-- 1 root root 17 2009-01-01 00:00 d -> /sys/kernel/debug
lrw-r--r-- 1 root root 50 2009-01-01 00:00 bugreports -> /data/user_de/0/com.android.shell/files/bugreports
lrw-r--r-- 1 root root 11 2009-01-01 00:00 bin -> /system/bin
drwxr-xr-x 23 root root 4096 2009-01-01 00:00 ..
drwxr-xr-x 23 root root 4096 2009-01-01 00:00 .
dr-xr-xr-x 281 root root 0 2021-12-15 14:32 proc
drwxr-xr-x 3 root root 0 2021-12-15 14:32 config
dr-xr-xr-x 13 root root 0 2021-12-15 14:32 sys
dr-xr-xr-x 76 root root 0 2021-12-15 14:32 acct
drwx--x--- 4 shell everybody 80 2021-12-15 14:32 storage
drwxr-xr-x 15 root system 320 2021-12-15 14:32 mnt
drwxr-xr-x 46 root root 920 2021-12-15 14:32 apex
drwxr-xr-x 21 root root 1320 2021-12-15 14:32 dev
drwxrwx--x 47 system system 4096 2021-12-15 14:33 data
```

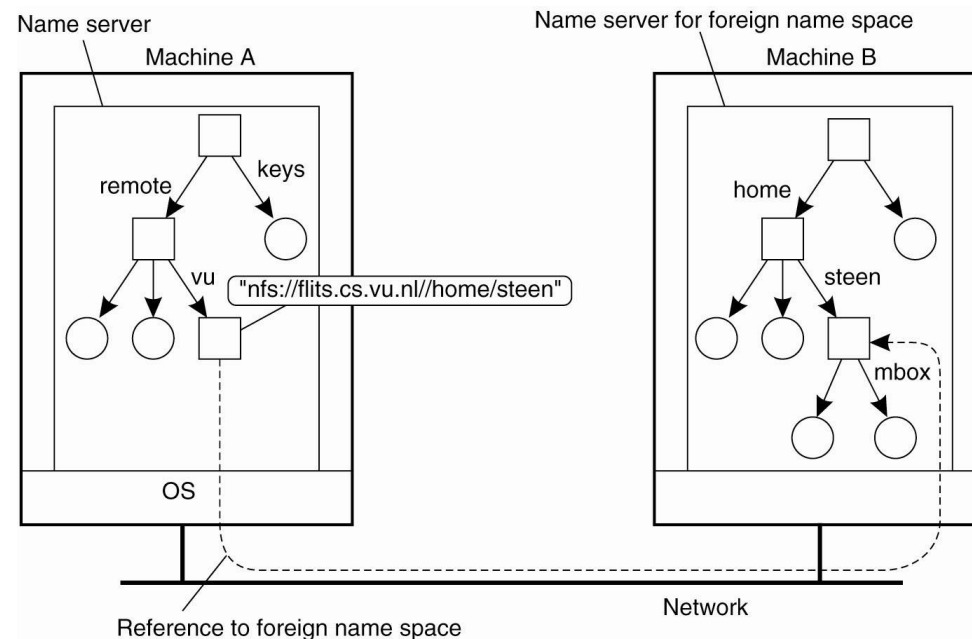
Mounting (NFS)



Mounting (NFS)

Information required to mount a foreign name space in a distributed system

- The name of an access protocol.
- The name of the server.
- The name of the mounting point in the foreign name space.



Name Space Implementation (1/2)

Basic issue: Distribute the name resolution process as well as name space management across multiple machines, by distributing nodes of the naming graph.

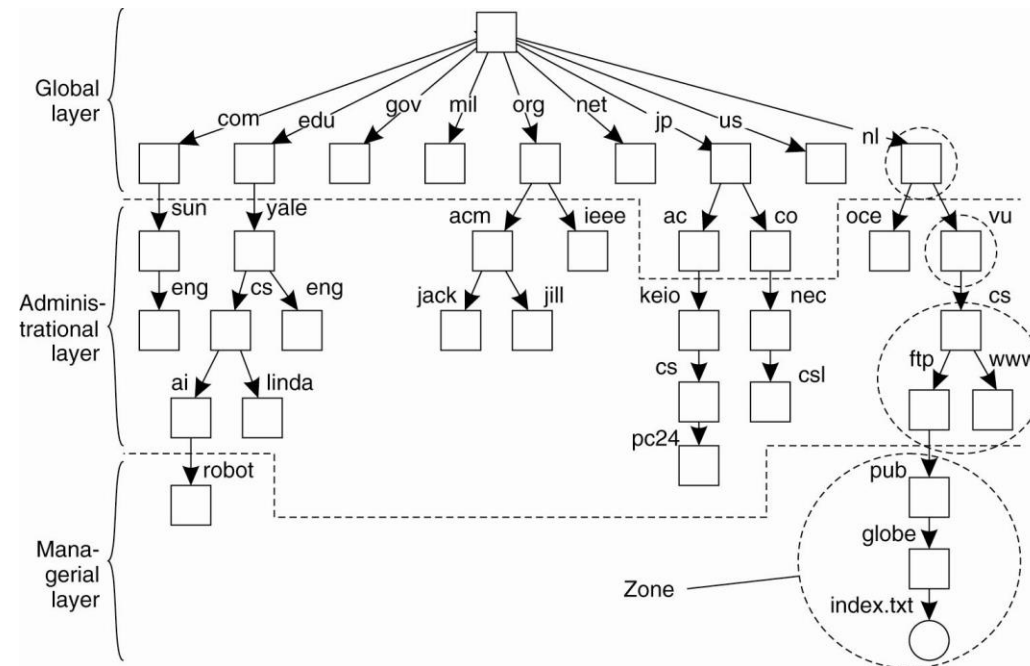
Consider a hierarchical naming graph and distinguish three levels:

Global level: Consists of the high-level directory nodes. Main aspect is that these directory nodes must be jointly managed by different administrations

Administrational level: Contains mid-level directory nodes that can be grouped in such a way that each group can be assigned to a separate administration.

Managerial level: Consists of low-level directory nodes within a single administration. Main issue is effectively mapping directory nodes to local name servers.

Name Space Implementation (2/2)



Item	Global	Administrational	Managerial
Geographical scale of network	Worldwide	Organization	Department
Total number of nodes	Few	Many	Vast numbers
Responsiveness to lookups	Seconds	Milliseconds	Immediate
Update propagation	Lazy	Immediate	Immediate
Number of replicas	Many	None or few	None
Is client-side caching applied?	Yes	Yes	Sometimes

DNS Levels

Global
Administrational
Managerial

איגוד
האינטרנט
הישראלי
ISOC-IL



NISSAN MOTOR CORPORATION



<https://mashitasugi.wixsite.com/mashitasworld/about-me>

Bus צמח

This site was designed with the **WIX**.com website builder. Create your website today. [Start Now](#)

cis.upenn.edu/index.php

Bus צמח



LOYOLA
UNIVERSITY CHICAGO

Technion - Israel Institute of Technology

Computer Science Department

DEPARTMENT OF COMPUTER SCIENCE

So Far

- Structured Naming
 - Name Spaces
 - Name Resolution
 - Domain Name System (DNS)
- Attribute Based Naming
- Mutual Exclusion

Iterative Name Resolution

- $\text{resolve}(\text{dir}, [\text{name1}, \dots, \text{nameK}])$ is sent to Server0 responsible for dir
- Server0 resolves $\text{resolve}(\text{dir}, \text{name1}) \rightarrow \text{dir1}$, returning the identification (address) of Server1, which stores dir1.
- Client sends $\text{resolve}(\text{dir1}, [\text{name2}, \dots, \text{nameK}])$ to Server1, etc.

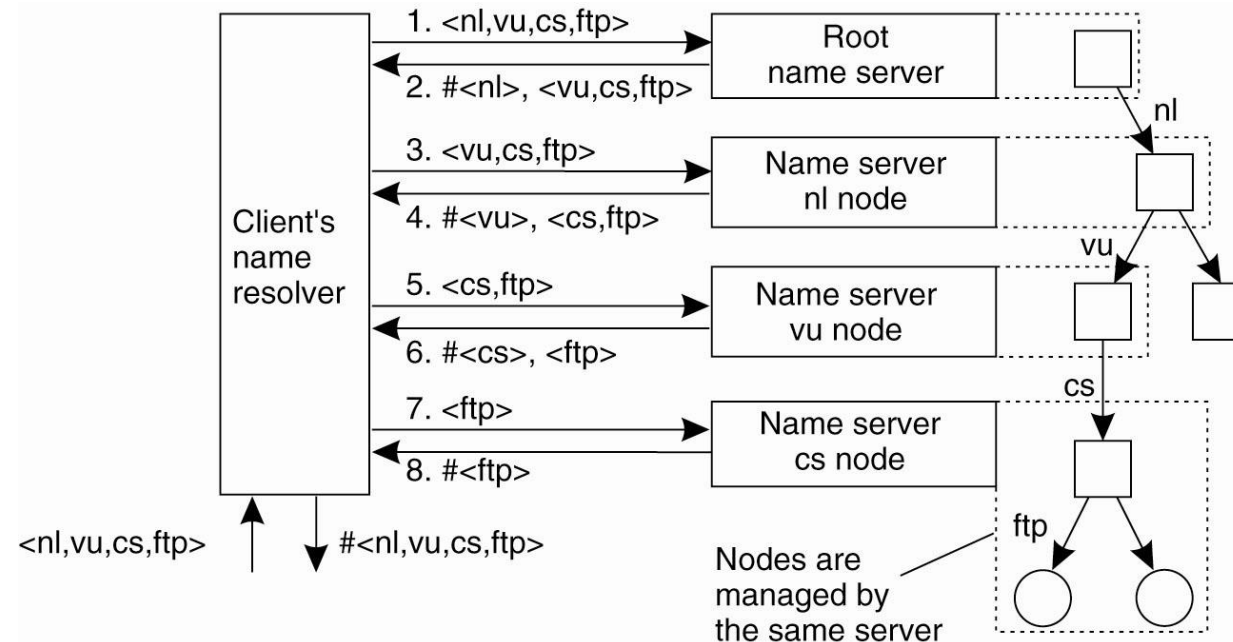
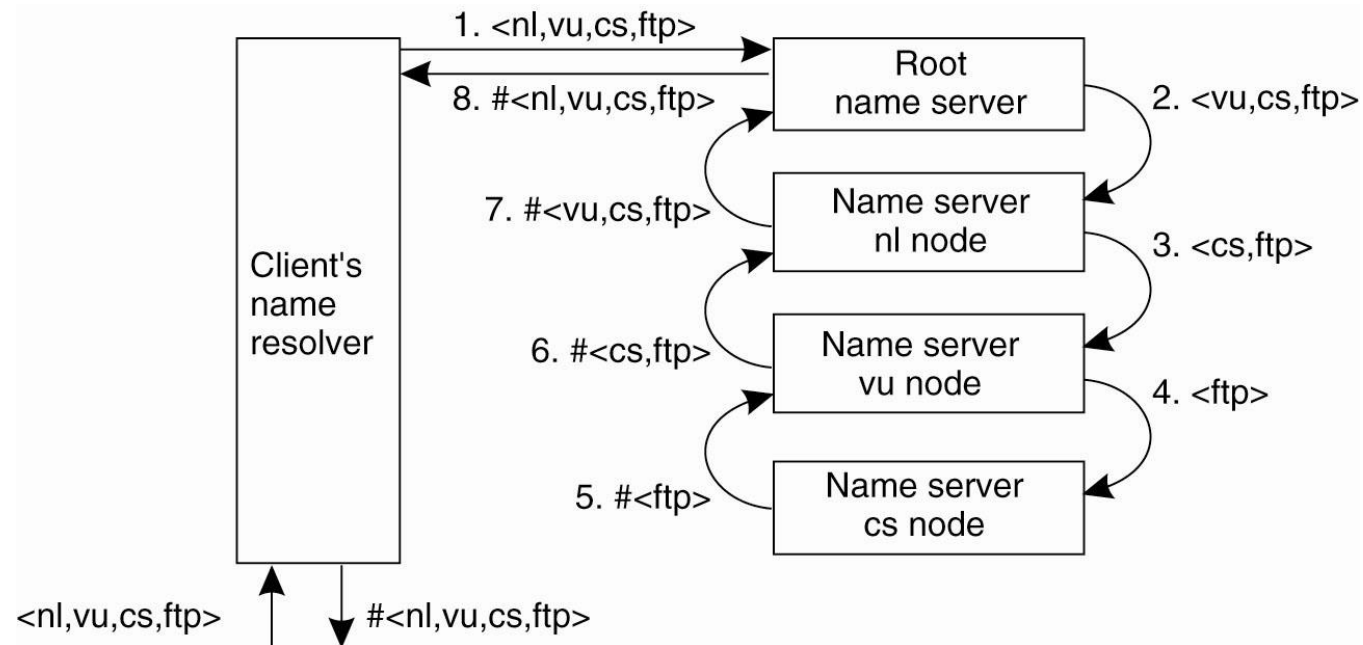




Image source: Thomas Hawk, <https://www.flickr.com/photos/thomashawk/6660194875/in/photostream/>

Recursive Name Resolution

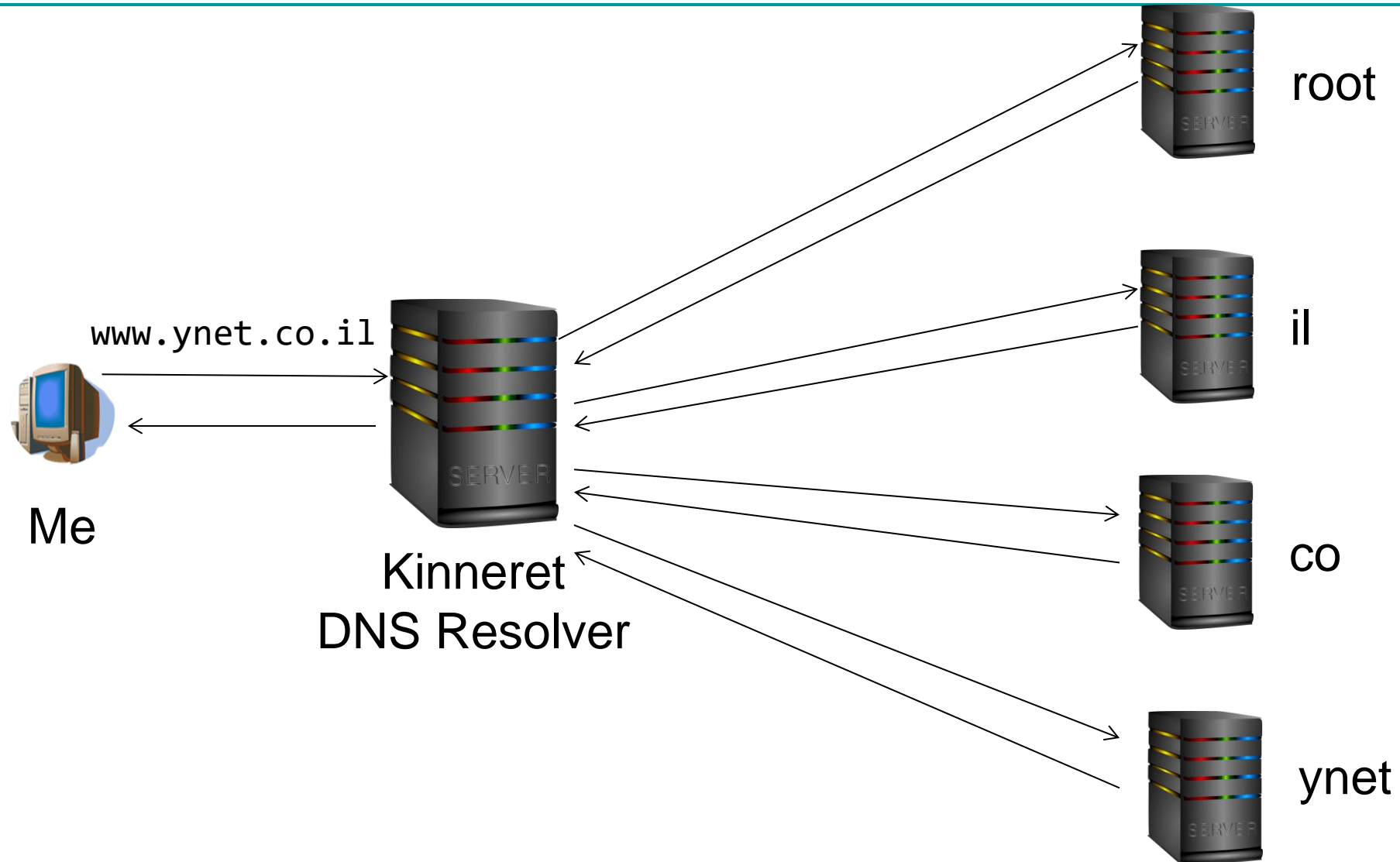
- $\text{resolve}(\text{dir}, [\text{name}_1, \dots, \text{name}_K])$ is sent to Server0 responsible for dir
- Server0 resolves $\text{resolve}(\text{dir}, \text{name}_1) \rightarrow \text{dir}_1$, and sends $\text{resolve}(\text{dir}_1, [\text{name}_2, \dots, \text{name}_K])$ to Server1, which stores dir1.
- Server0 waits for the result from Server1, and returns it to the client.



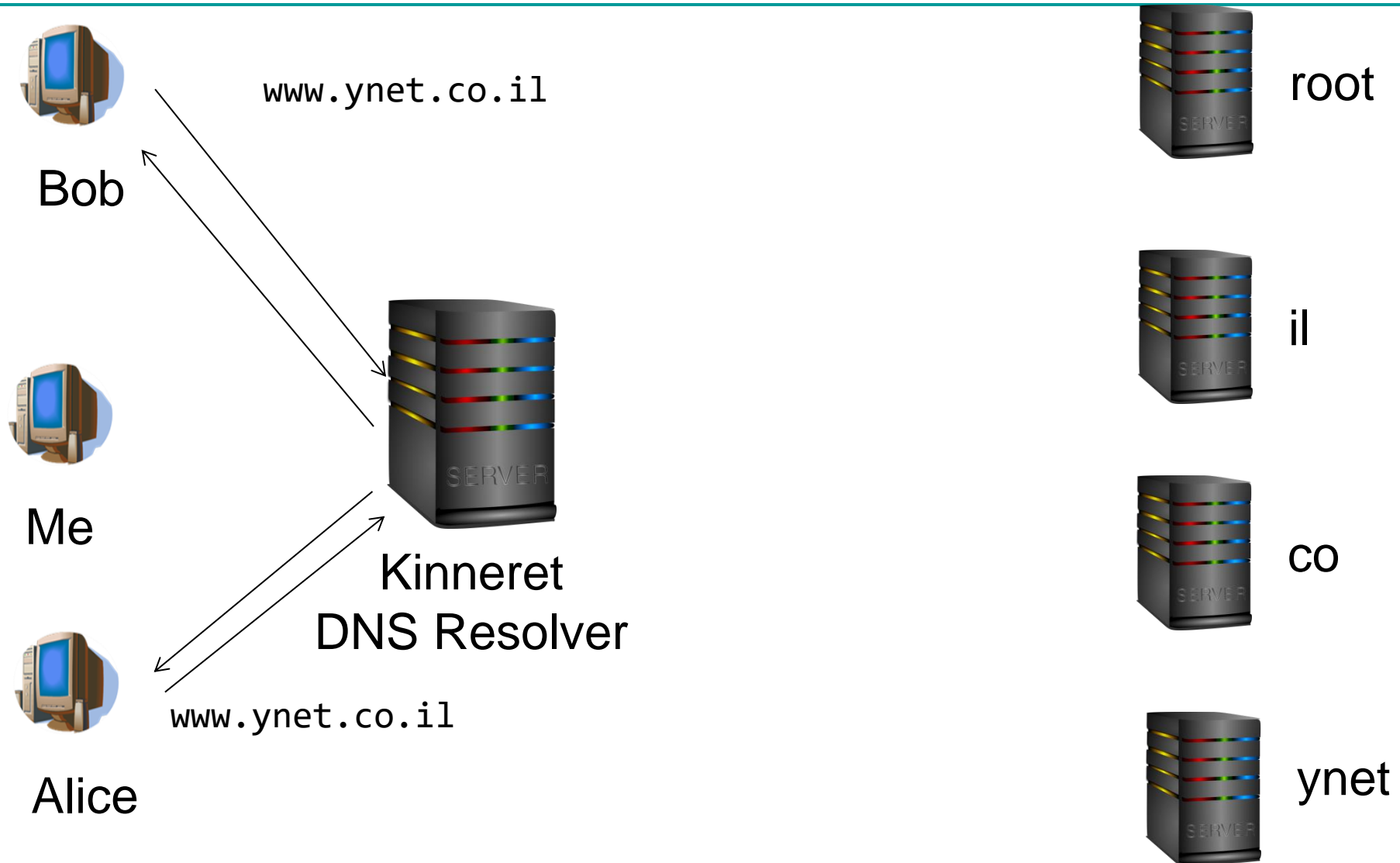
Caching in Recursive Name Resolution

Server for node	Should resolve	Looks up	Passes to child	Receives and caches	Returns to requester
cs	<ftp>	#<ftp>	—	—	#<ftp>
vu	<cs,ftp>	#<cs>	<ftp>	#<ftp>	#<cs> #<cs, ftp>
nl	<vu,cs,ftp>	#<vu>	<cs,ftp>	#<cs> #<cs,ftp>	#<vu> #<vu,cs> #<vu,cs,ftp>
root	<nl,vu,cs,ftp>	#<nl>	<vu,cs,ftp>	#<vu> #<vu,cs> #<vu,cs,ftp>	#<nl> #<nl,vu> #<nl,vu,cs> #<nl,vu,cs,ftp>

Combined Name Resolution



Combined Name Resolution



Scalability Issues

Size scalability

We must ensure servers can handle a large number of requests per time unit
→ high-level servers are in big trouble.

Solution

- Assume node content rarely changes.
- Then, use extensive replication by mapping nodes to multiple servers
- Start name resolution at the nearest server.

Observation

- Important attribute of many nodes is the **address** where the represented entity can be contacted.
- Replicating nodes makes large-scale traditional name servers unsuitable for locating mobile entities.

So Far

- Structured Naming
 - Name Spaces
 - Name Resolution
 - Domain Name System (DNS)
- Attribute Based Naming
- Mutual Exclusion

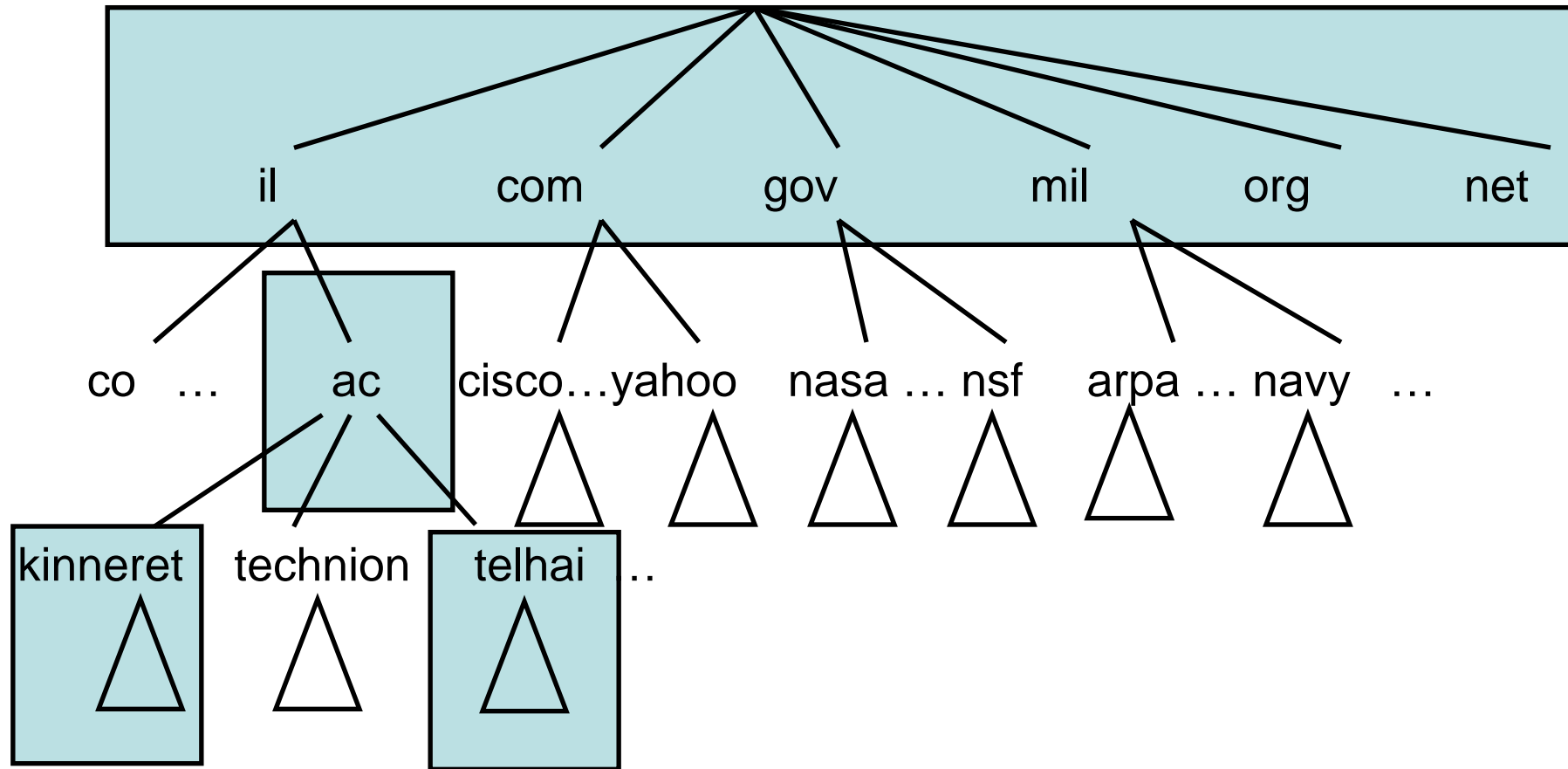
Domain Name System

- System for mapping mnemonic names for computers into IP addresses.

softwares.kinneret.ac.il \longrightarrow 104.22.3.77

- Domain Hierarchy
- Name Servers
 - 13 Root servers map top-level domains such as ".com" or ".net"
- Name Resolution
 - Protocol for looking up hierarchical domain names to determine the IP address
 - Protocol runs on UDP port 53

Domain Name Hierarchy



DNS Records

- The most important types of resource records forming the contents of nodes in the DNS name space.

Type of record	Associated entity	Description
SOA	Zone	Holds information on the represented zone
A	Host	Contains an IP address of the host this node represents
MX	Domain	Refers to a mail server to handle mail addressed to this node
SRV	Domain	Refers to a server handling a specific service
NS	Zone	Refers to a name server that implements the represented zone
CNAME	Node	Symbolic link with the primary name of the represented node
PTR	Host	Contains the canonical name of a host
HINFO	Host	Holds information on the host this node represents
TXT	Any kind	Contains any entity-specific information considered useful

DNS Records (1/2)

- An excerpt from the DNS database for the zone *cs.vu.nl*.

Name	Record type	Record value
cs.vu.nl.	SOA	star.cs.vu.nl. hostmaster.cs.vu.nl. 2005092900 7200 3600 2419200 3600
cs.vu.nl.	TXT	"Vrije Universiteit - Math. & Comp. Sc."
cs.vu.nl.	MX	1 mail.few.vu.nl.
cs.vu.nl.	NS	ns.vu.nl.
cs.vu.nl.	NS	top.cs.vu.nl.
cs.vu.nl.	NS	solo.cs.vu.nl.
cs.vu.nl.	NS	star.cs.vu.nl.
star.cs.vu.nl.	A	130.37.24.6
star.cs.vu.nl.	A	192.31.231.42
star.cs.vu.nl.	MX	1 star.cs.vu.nl.
star.cs.vu.nl.	MX	666 zephyr.cs.vu.nl.
star.cs.vu.nl.	HINFO	"Sun" "Unix"
zephyr.cs.vu.nl.	A	130.37.20.10
zephyr.cs.vu.nl.	MX	1 zephyr.cs.vu.nl.
zephyr.cs.vu.nl.	MX	2 tornado.cs.vu.nl.
zephyr.cs.vu.nl.	HINFO	"Sun" "Unix"

DNS Records (2/2)

- An excerpt from the DNS database for the zone *cs.vu.nl*.

ftp.cs.vu.nl.	CNAME	soling.cs.vu.nl.
www.cs.vu.nl.	CNAME	soling.cs.vu.nl.
soling.cs.vu.nl.	A	130.37.20.20
soling.cs.vu.nl.	MX	1 soling.cs.vu.nl.
soling.cs.vu.nl.	MX	666 zephyr.cs.vu.nl.
soling.cs.vu.nl.	HINFO	"Sun" "Unix"
vucs-das1.cs.vu.nl.	PTR	0.198.37.130.in-addr.arpa.
vucs-das1.cs.vu.nl.	A	130.37.198.0
inkt.cs.vu.nl.	HINFO	"OCE" "Proprietary"
inkt.cs.vu.nl.	A	192.168.4.3
pen.cs.vu.nl.	HINFO	"OCE" "Proprietary"
pen.cs.vu.nl.	A	192.168.4.2
localhost.cs.vu.nl.	A	127.0.0.1

Kinneret DNS Records (1/3)

- An excerpt from the DNS database for zone kinneret.ac.il

kinneret.ac.il	NS	tiffany.ns.cloudflare.com.
tiffany.ns.cloudflare.com.	A	108.162.194.60
tiffany.ns.cloudflare.com.	A	162.159.38.60
tiffany.ns.cloudflare.com.	A	172.64.34.60
tiffany.ns.cloudflare.com.	A	2606:4700:50::a29f:263c
tiffany.ns.cloudflare.com.	A	2803:f800:50::6ca2:c23c
tiffany.ns.cloudflare.com.	A	2a06:98c1:50::ac40:223c

Kinneret DNS Records (2/3)

- An excerpt from the DNS database for zone kinneret.ac.il

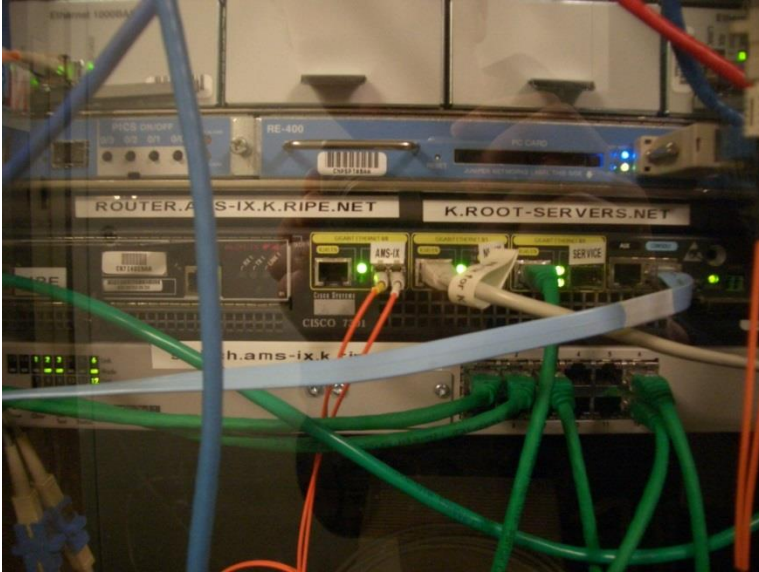
kinneret.ac.il	NS	uriah.ns.cloudflare.com.
uriah.ns.cloudflare.com.	A	108.162.195.194
uriah.ns.cloudflare.com.	A	172.64.35.194
uriah.ns.cloudflare.com.	A	162.159.44.194
uriah.ns.cloudflare.com.	A	2606:4700:58::a29f:2cc2
uriah.ns.cloudflare.com.	A	2a06:98c1:50::ac40:23c2
uriah.ns.cloudflare.com.	A	2803:f800:50::6ca2:c3c2

Kinneret DNS Records (2/2)

- An excerpt from the DNS database for zone kinneret.ac.il

kinneret.ac.il	MX	preference = 10, mail exchanger = mail-secure.kinneret.ac.il
kinneret.ac.il	A	88.218.117.88
kinneret.ac.il	SOA	origin = tiffany.ns.cloudflare.com mail addr = dns.cloudflare.com serial = 2357024139 refresh = 10000 retry = 2400 expire = 604800 minimum = 1800

DNS Roots



Root server K in Amsterdam, Holland
(Wikipedia)



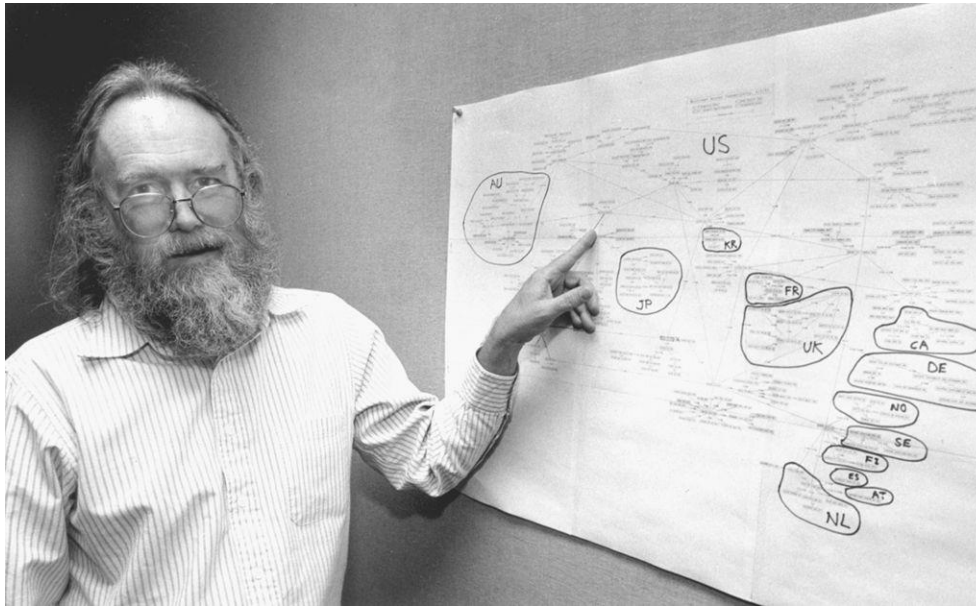
ICANN is responsible for managing roots and top level domains

- 13 DNS root servers heavily replicated around the world
 - 12 independent orgs run the roots

Distributed Control (DNS)

Jan 1998: **Jon Postel** of IANA told 8 of the 12 roots at the time to contact **IANA's root copy** instead of the **US government's root copy** (Network Solutions, Inc. in Herndon, VA)

- Postel said it was a test and changed it back when asked (?)
- Sept 1998 - ICANN is formed and takes over IANA's job

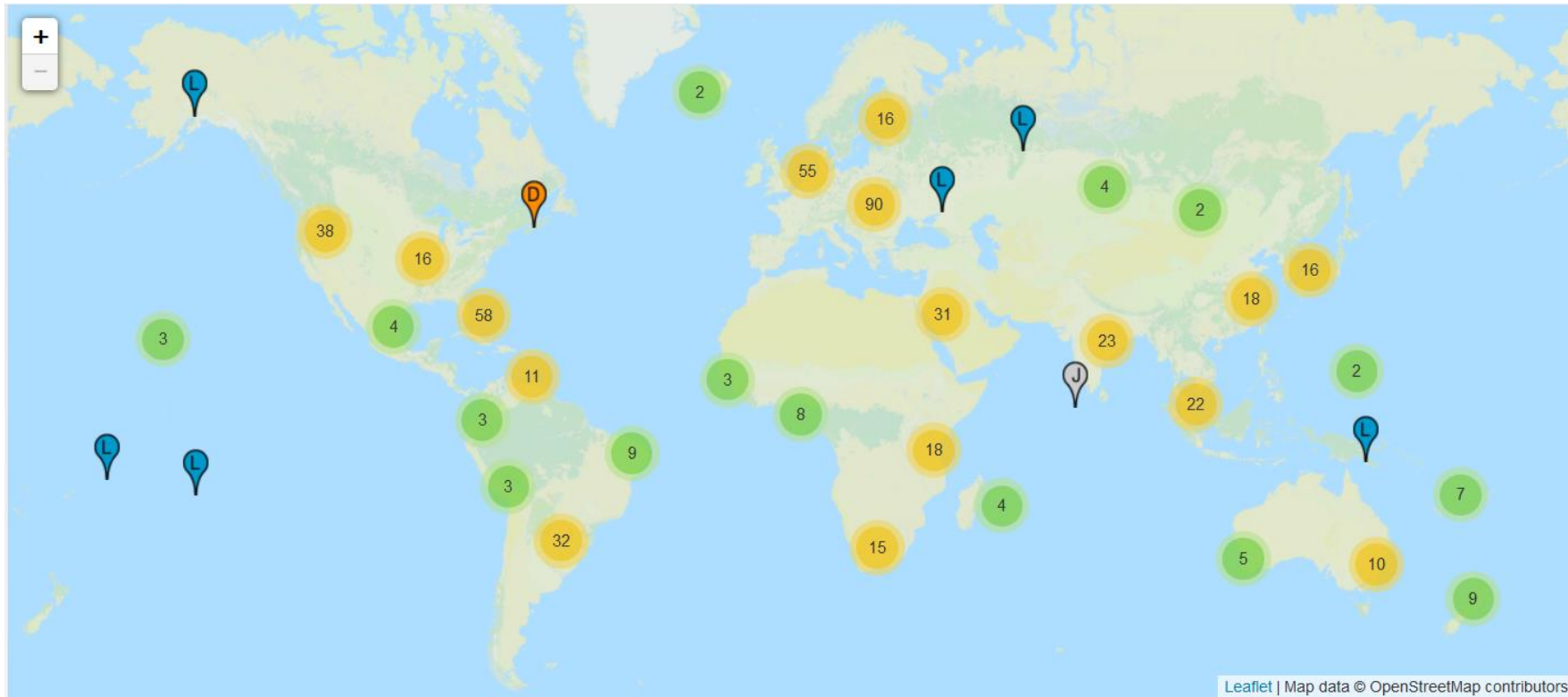


From <http://www.postel.org/pr.htm>: Photo by Irene Fertik, USC News Service. © 1994, USC. Permission granted for free use and distribution, conditioned upon inclusion of the above attribution and copyright notice.

DNS Roots Worldwide (2015)



DNS Roots Worldwide (2016)



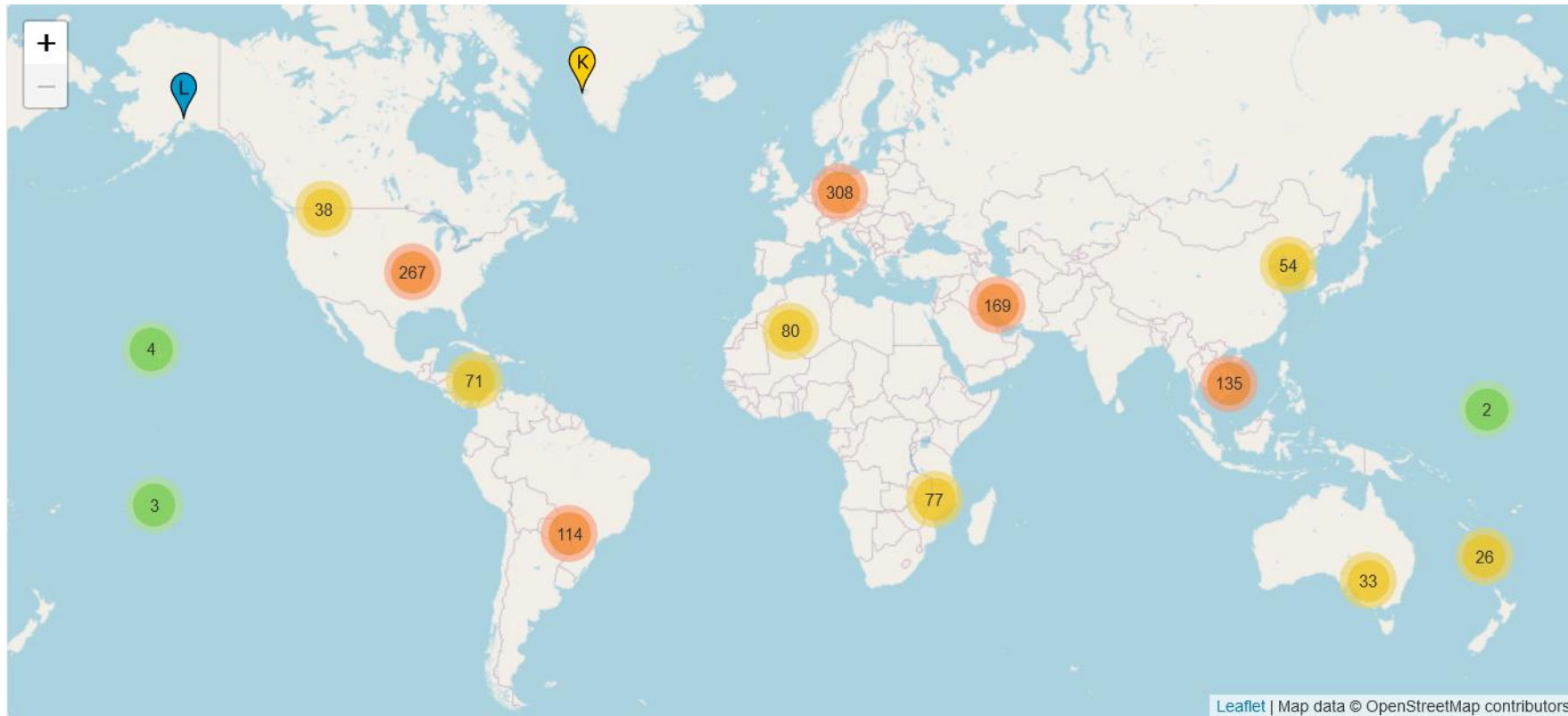
DNS Roots Worldwide (2018)



DNS Roots Worldwide (2021)



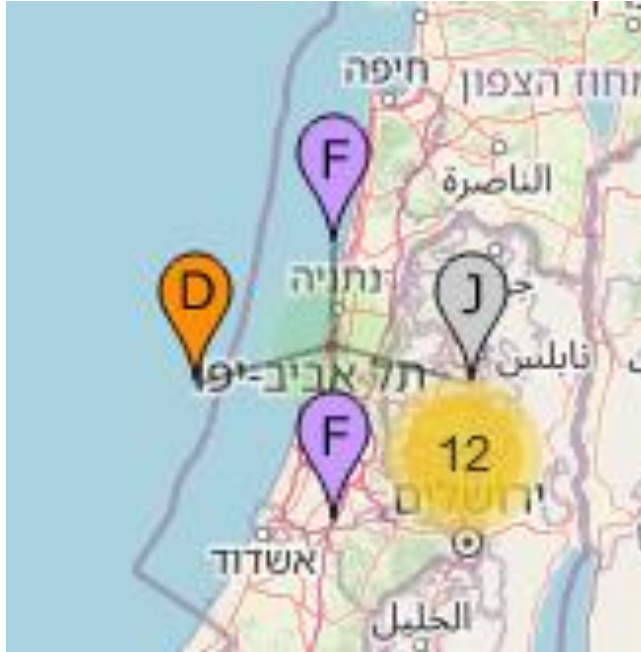
DNS Roots Worldwide (2022)



DNS Roots Worldwide (2024)



DNS Roots in Israel



Map includes some in Jordan, Ramallah and Gaza.

Total of 7 in Petah Tikvah and Tel Aviv.

DNS TLDs

1,591 TLDs (Top Level Domains) are maintained by private networking companies and organizations (Feb 2024)

- Private registrars sign up customers

TLDs are

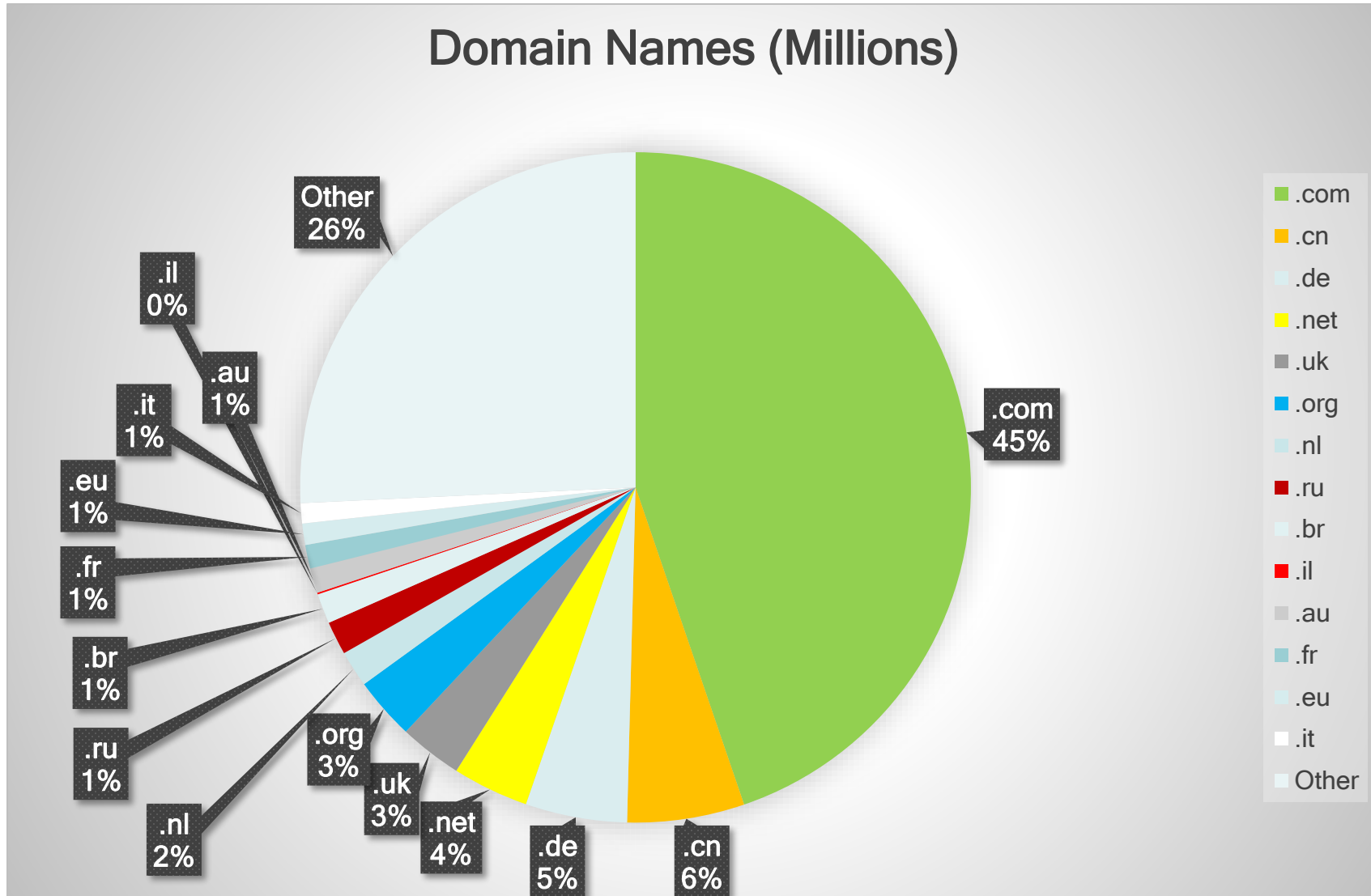
- By business sector (ex. .bike, .clothing, .plumbing)
- By country (ex. .us, .il, .ca, .uk)
- By organization type (ex. .org, .ac.il, .edu, .co.uk)
- By language (ex. XN--1QQW23A (Chinese), XN--3E0B707E (Korean), XN--45BRJ9C (Hindi), XN--4GBRIM (Arabic - Saudi Arabia))
- Generic (ex. .info, .xyz, .center, .cards)

Notable TLDs:

- .com used to be run by US DoD, now by Verisign - 160.9 million domains (Dec 2022)
- .edu run by Educause (contracted to Verisign)
- .il is run by ISOC Israel - 288K domains (2024)
.ישראל is also run by ISOC 22K domains (2024)

Domain Name Distribution

Data source: Domain Name Industry Brief Q3 2023
(<https://dnib.com/articles/the-domain-name-industry-brief-q3-2023>)



So Far

- Structured Naming
 - Name Spaces
 - Name Resolution
 - Domain Name System (DNS)
- Attribute Based Naming
- Mutual Exclusion

Attribute-Based Naming

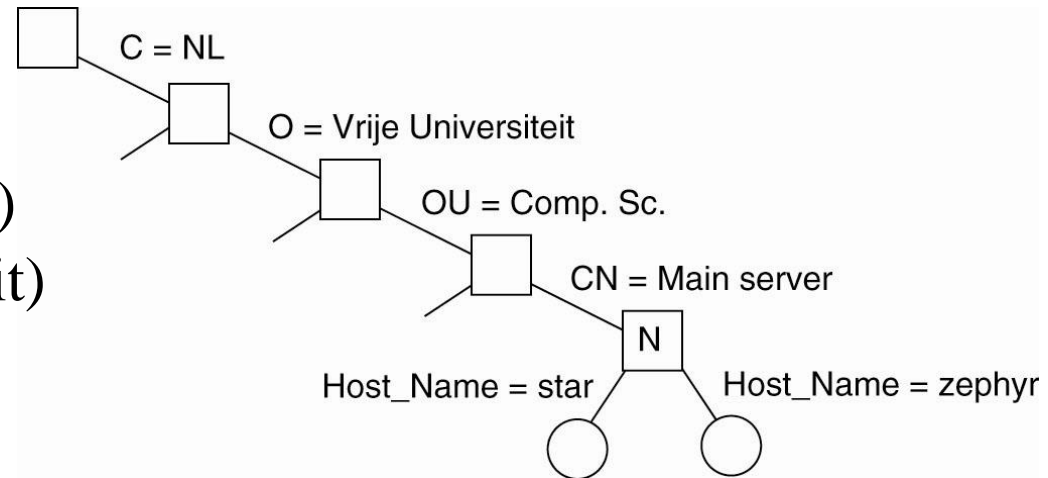
Observation: In many cases, it is much more convenient to name, and look up entities by means of their **attributes** → traditional **directory services** (aka **yellow pages**).

Problem: Lookup operations can be extremely expensive, as they require to match **requested attribute values**, against **actual attribute values** → inspect **all entities** (in principle).

Solution: Implement basic directory service as database, and combine with traditional structured naming system.

Example: LDAP

answer = **search** ("&(C = NL)
(O = Vrije Universiteit)
(OU = *)
(CN = Main server)")



Attribute	Value
Country	NL
Locality	Amsterdam
Organization	Vrije Universiteit
OrganizationalUnit	Comp. Sc.
CommonName	Main server
Host_Name	star
Host_Address	192.31.231.42

Attribute	Value
Country	NL
Locality	Amsterdam
Organization	Vrije Universiteit
OrganizationalUnit	Comp. Sc.
CommonName	Main server
Host_Name	zephyr
Host_Address	137.37.20.10

(b)

Example: LDAP

`(&(objectClass=person)(ou=Software Engineering)(o=Kinneret)(givenName=Natan))`

So Far

- Structured Naming
 - Name Spaces
 - Name Resolution
 - Domain Name System (DNS)
- Attribute Based Naming
- Mutual Exclusion

Mutual Exclusion

Problem: A number of processes in a distributed system want exclusive access to some resource.

Basic solutions:

Via a **centralized server**.

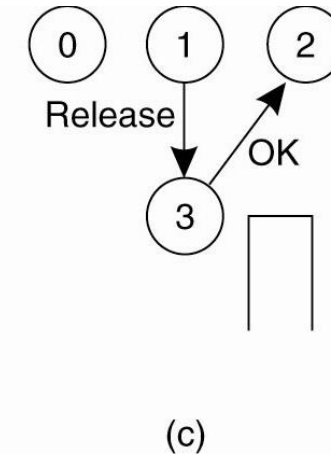
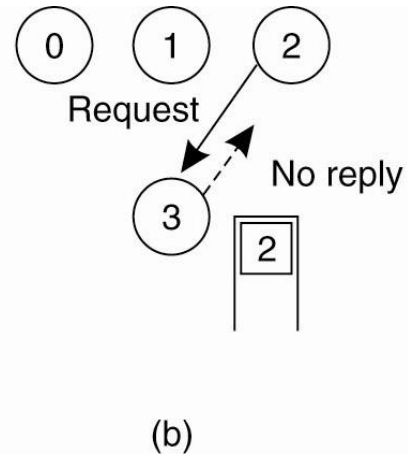
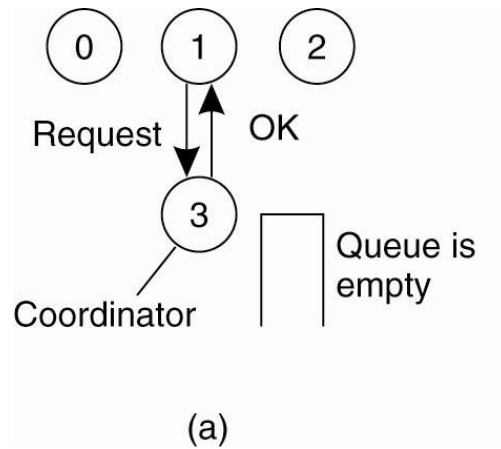
Completely decentralized, using a peer-to-peer system.

Completely distributed, with no topology imposed.

Completely distributed along a **(logical) ring**.

Mutual Exclusion: Centralized

Version 1: Queue requests



Version 2: Respond "NO" if the resource is busy

Decentralized Mutual Exclusion

Principle: Assume every resource is replicated n times, with each replica having its own coordinator → access requires a **majority vote** from $m > \frac{n}{2}$ coordinators. A coordinator always responds immediately to a request.

Assumption: When a coordinator crashes, it will recover quickly, but will have forgotten about permissions it had granted.

Issue: How robust is this system? Let $p = \frac{\Delta t}{T}$ denote the probability that a coordinator crashes and recovers in a period Δ_t while having an average lifetime T → probability that k out of m coordinators **reset**:

$$p_v = \sum_{k=2m-n}^m \binom{m}{k} p^k (1-p)^{m-k}$$

With $p = 0.001$, $n = 32$, $m = 0.75n$, $p_v < 10^{-40}$

Decentralized Mutual Exclusion

Violation probabilities for various parameter values

N	M	P	Violation
8	5	3 sec/hour	$< 10^{-5}$
8	6	3 sec/hour	$< 10^{-11}$
16	9	3 sec/hour	$< 10^{-4}$
16	12	3 sec/hour	$< 10^{-21}$
32	17	3 sec/hour	$< 10^{-4}$
32	24	3 sec/hour	$< 10^{-43}$
8	5	30 sec/hour	$< 10^{-3}$
8	6	30 sec/hour	$< 10^{-7}$
16	9	30 sec/hour	$< 10^{-2}$
16	12	30 sec/hour	$< 10^{-13}$
32	17	30 sec/hour	$< 10^{-2}$
32	24	30 sec/hour	$< 10^{-27}$

Mutual Exclusion: Ricart & Agrawala

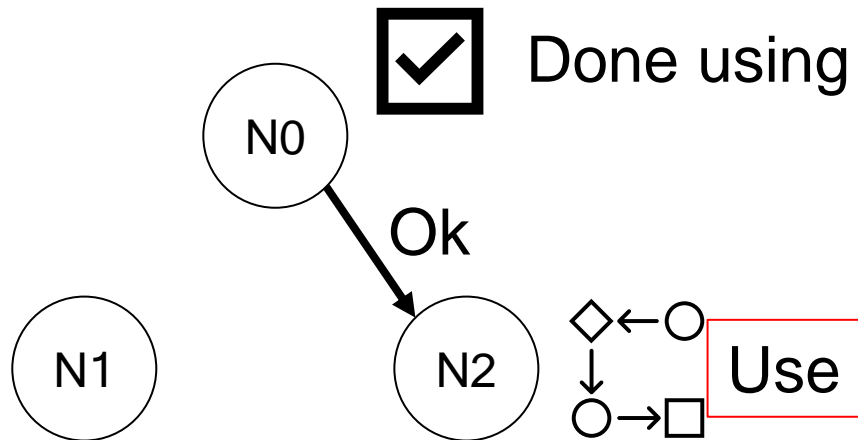
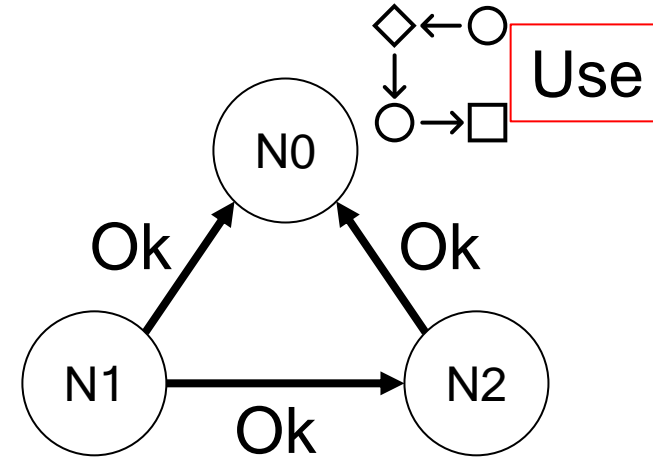
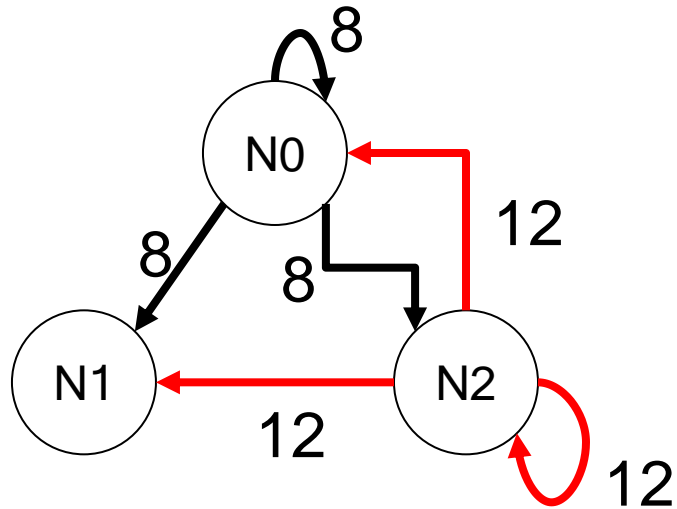
Must receive permission (grant) from all other participants to use the shared resource

Principle: Same as Lamport except that acknowledgments aren't sent. Instead, replies (i.e., grants) are sent only when:

- The receiving process has no interest in the shared resource; or
- The receiving process is waiting for the resource, but has lower priority (known through comparison of timestamps).

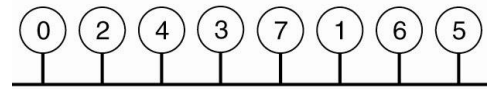
In all other cases, reply is **deferred**, implying some more local administration.

Mutual Exclusion: Ricart & Agrawala

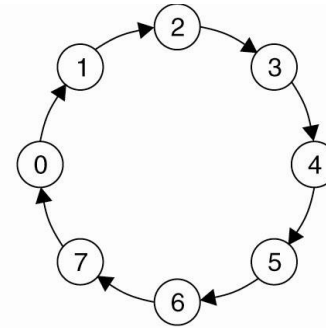


Mutual Exclusion: Token Ring Algorithm

Essence: Organize processes in a *logical* ring, and let a token be passed between them. The one that holds the token is allowed to enter the critical region (if it wants to)



(a)



(b)

Issues:

- Token creation and loss
- Maximum token holding time

Mutual Exclusion Summary

Algorithm	Messages per entry/exit	Delay before entry (in message times)	Problems and Issues
Centralized	3	2	Coordinator crash
Decentralized	$3mk$ For k rounds	$2m$	Starvation (rounds with no winner), low efficiency
Distributed (Ricart and Agrawal)	$2(n - 1)$	$2(n - 1)$	Crash of any process
Token Ring	1 to ∞	0 to $n - 1$	Lost token, creation, process crash

Conclusion

- Structured Naming
 - Name Spaces
 - Name Resolution
 - Domain Name System (DNS)
- Attribute Based Naming
- Mutual Exclusion