LECTURE 12

Structured Naming

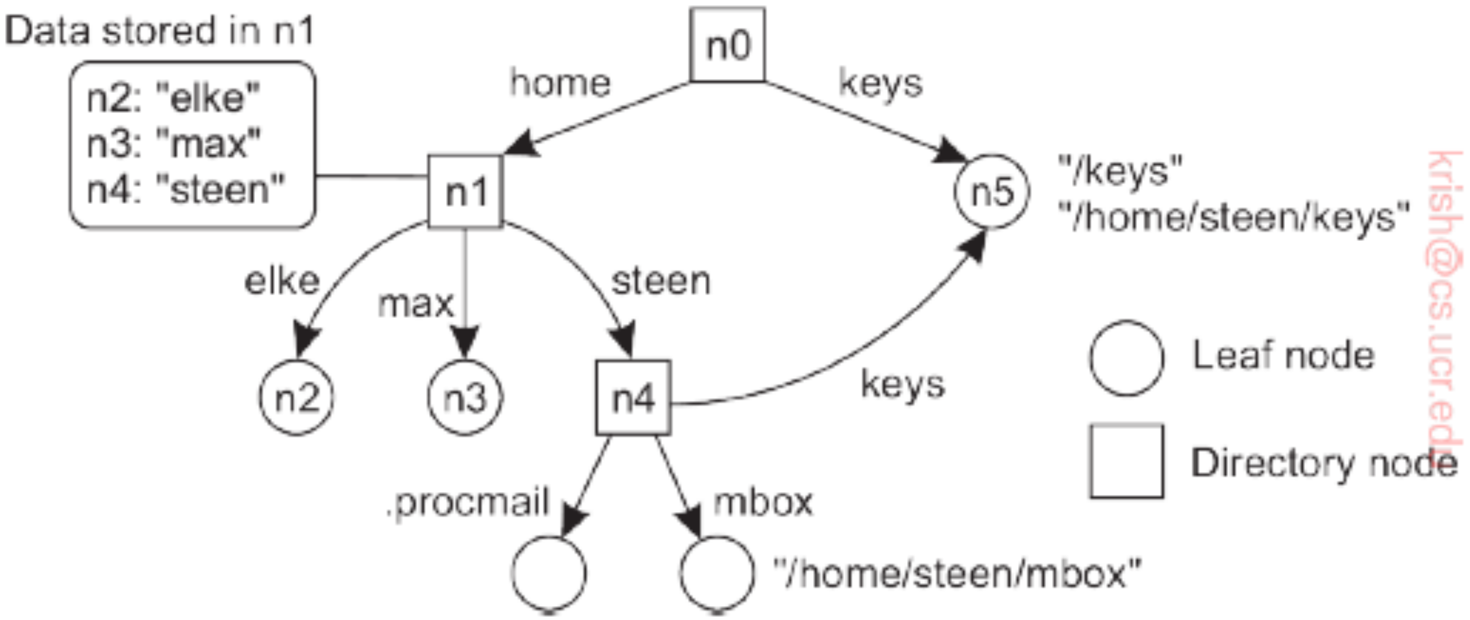
Naming

- The use of a DHT could be used for name resolution.
  - Finding the path to a desired object
- While this works in certain contexts (e.g., peer to peer file sharing), it is not used en masse.
- Distributed systems use structured names
  - Easy for humans to read

Name spaces

- Names organized into name spaces
- Structured as a labeled, directed graph
  - Leaves have no outgoing edges and generally store information (e.g., the address to a file)
  - Directory nodes have a number of outgoing edges
    - stores a table which is represented as a node identifier, edge label (so as to enable the graph traversal)

Example



- The node n0 has only outgoing edges – the root
- The naming graph is a directed acyclic graph -- no cycles

Path names

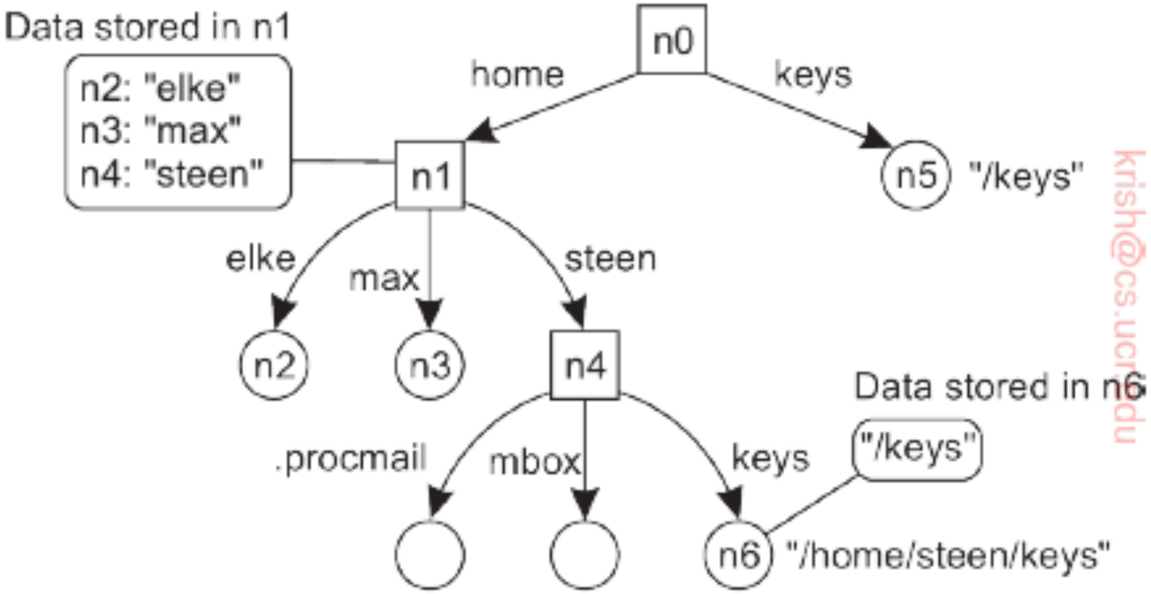
- Each path in a naming graph – referred to by a sequence of labels
  - no:[n1,n4, mbox] or n0/n1/n4/mbox
- If the first node is the root, it is an absolute path name; else a relative path name.

Name resolution

- How do we look up a name ?
  - This is referred to as name resolution.
- Consider a path name N:[label$_1$, label$_2$, … label$_n$]
- The resolution here starts at node N
  - That node looks up label$_1$ in its directory table, and returns the identifier of the node associated with that label.
  - Resolution then continues at that node.
- Typically at least need the root.

Aliasing and symbolic links

- Alias : an alternate name for the same entity
- Below both /keys and /home/steen/keys refer to the same entity.
- When resolving an absolute path name leading to a node (here n6), the resolution returns the path stored at n6.
  - It then continues with resolving that new path name.



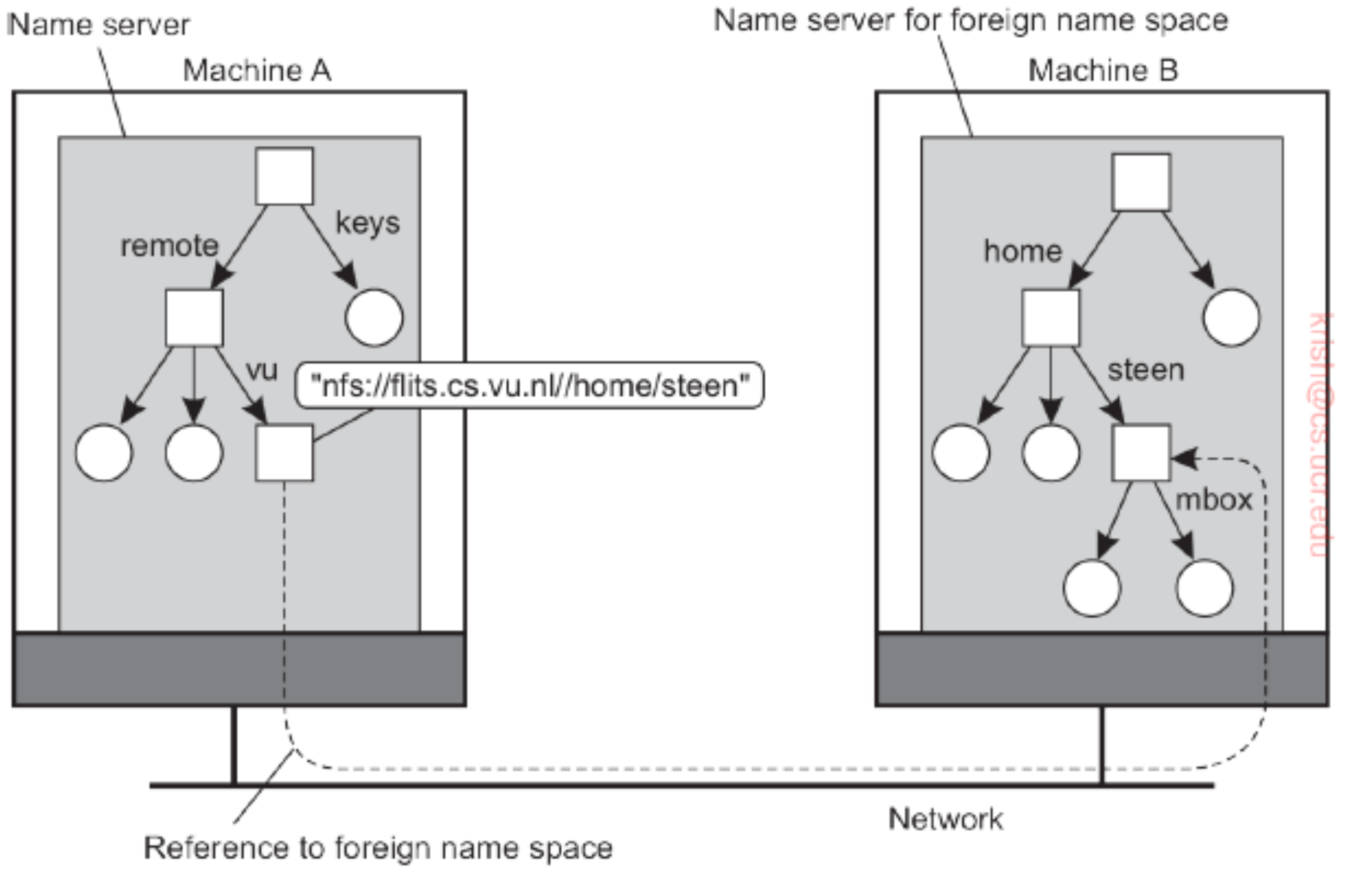
Naming in a distributed system

- The concept of a foreign name space
  - Each part of the system has its own name space
    - Foreign to the other naming services.
- Name resolution can be used to merge the spaces in a transparent way.
  - One can mount the foreign name space
  - The directory node (e.g., root) is called a mounting point and is attached to the name space at the ``home" name server.

What is necessary ?

- To mount a foreign name space, a distributed system requires at least the following:
  - The name of an access protocol (e.g., nfs or Network File System)
    - URL nfs://flits.cs.vu.nl/home/steen
  - The name of the server
  - The name of the mounting point in the foreign name space.
    - Latter two specified above.

Mounting remote name spaces


Name server
Machine A
remote
keys
vu
"nfs://flits.cs.vu.nl//home/steen"
Name server for foreign name space
Machine B
home
steen
mbox
Network
Reference to foreign name space

- Root directory has a subdirectory remote
- Includes mount points for various foreign name spaces including a URL nfs://flits.cs.vu.nl/home/steen

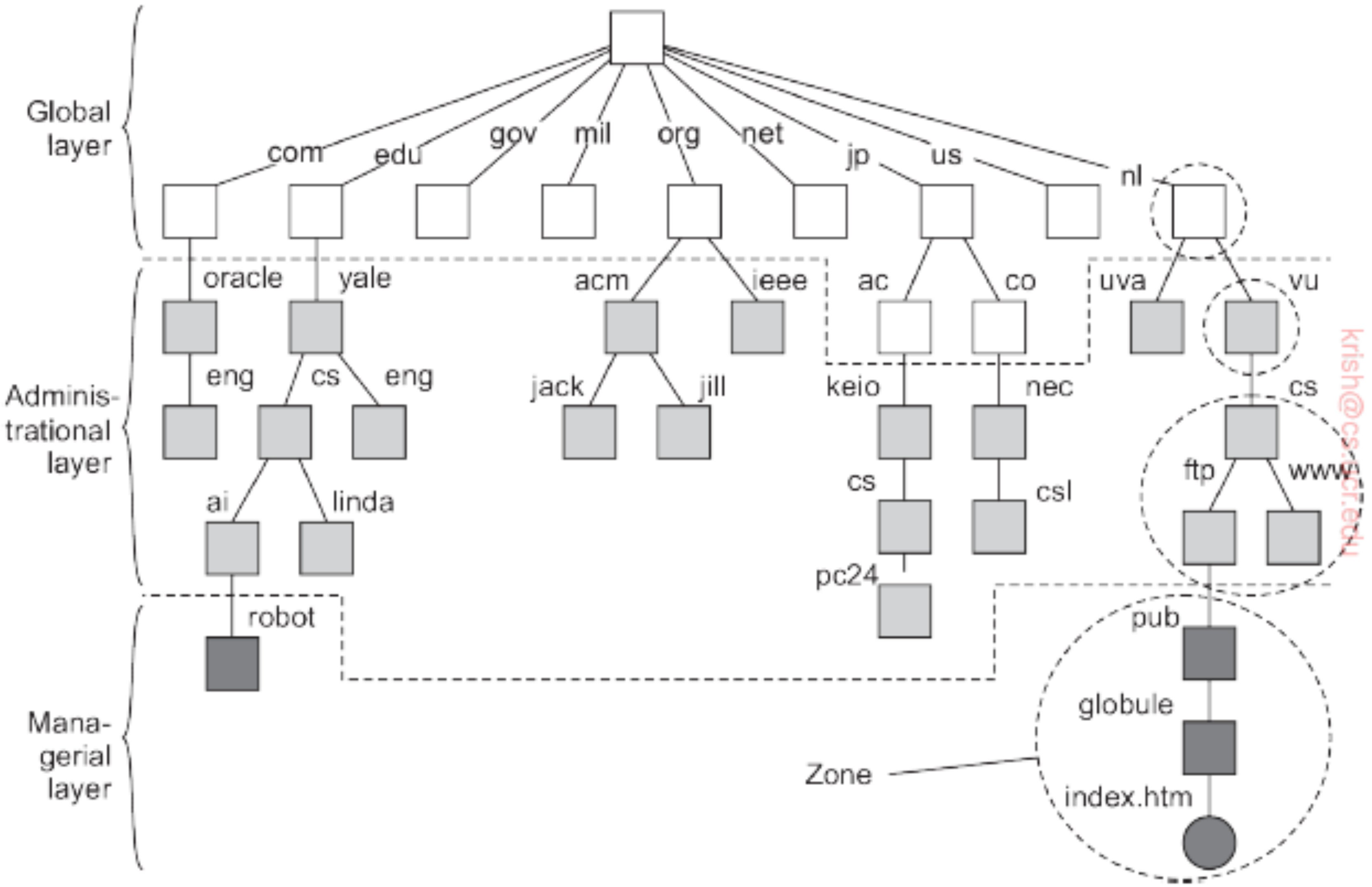
How does this work?

- Consider a remote file system that is mounted using the approach described.
- The mounted remote file system allows a client machine to execute commands like they are done locally.
  - For example, cd /remote/vu and then running ls.

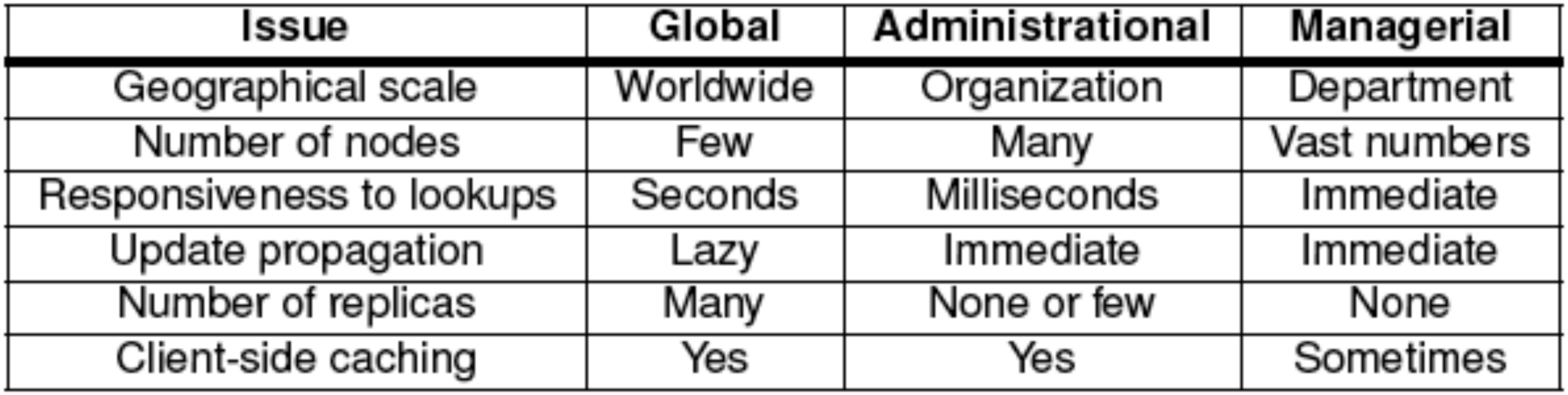
Implementing a name space

- A hierarchical structure
  - At the top a global layer
    - Very stable – organizations or groups of organizations
  - Next layer – administrational layer
    - Also stable but less than global – groups of entities belonging to same organization or administrative unit.
  - Managerial layer
    - Individuals manage these typically and they change regularly.

The DNS name space example



Comparison between name servers

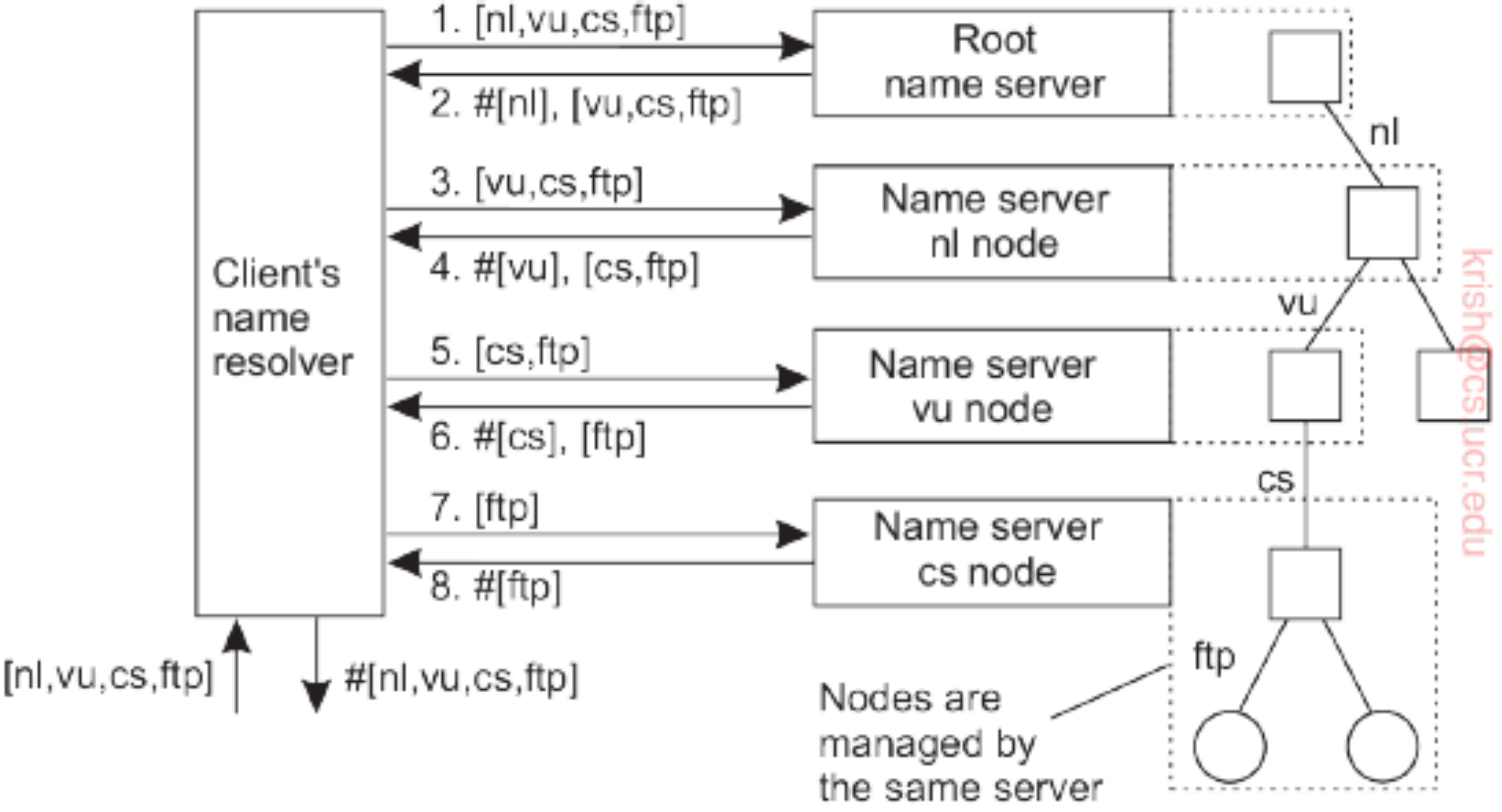


Issue	Global	Administrational	Managerial
Geographical scale	Worldwide	Organization	Department
Number of nodes	Few	Many	Vast numbers
Responsiveness to lookups	Seconds	Milliseconds	Immediate
Update propagation	Lazy	Immediate	Immediate
Number of replicas	Many	None or few	None
Client-side caching	Yes	Yes	Sometimes

Name resolution in a distributed sytem

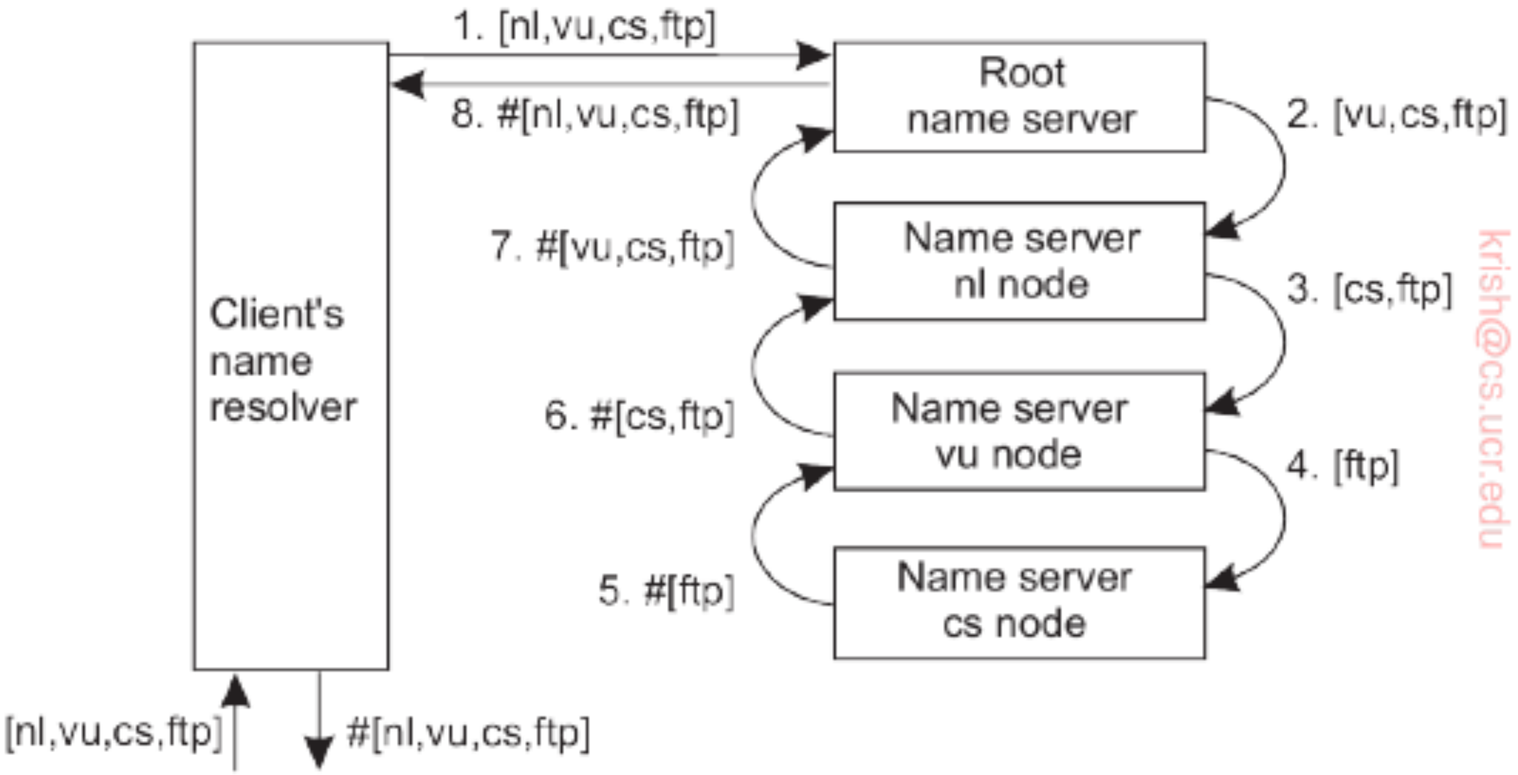
- Typically when a client wants to resolve a name, it has access to a local name resolver.
- This is responsible for ensuring that the name resolution process is carried out.
- Let us take an example:
  - ftp://ftp.cs.vu.nl/pub/globe/index.html
- Two ways in which this can be resolved
  - Iterative name resolution
  - Recursive name resolution

Iterative name resolution



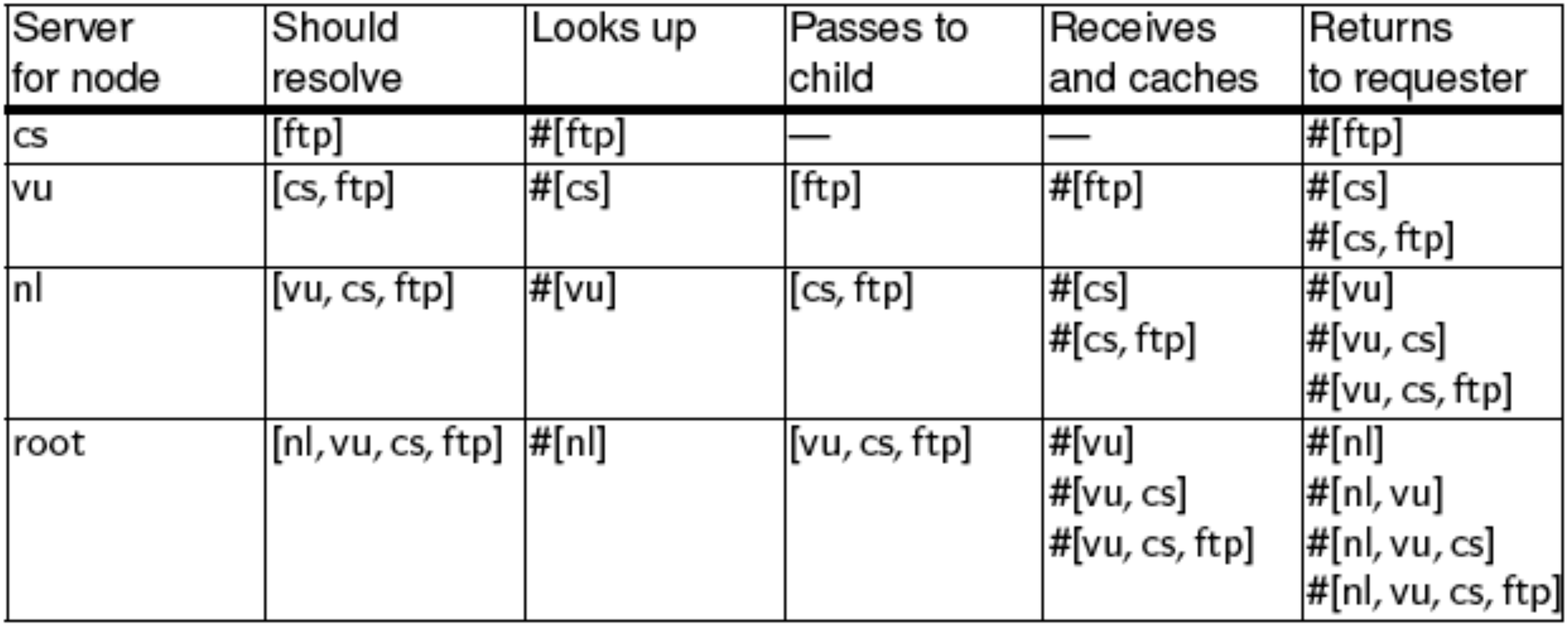
- Iteratively go through the hierarchical structure of name servers until the service is obtained.
- The last step of contacting the ftp server and retrieving the file is carried out by the client process.

Recursive name resolution



- Each server tries to contact the resolved name server in the chain by itself.
  - However this can add significant load on some of the higher tier name servers (e.g., global).

What is returned ?



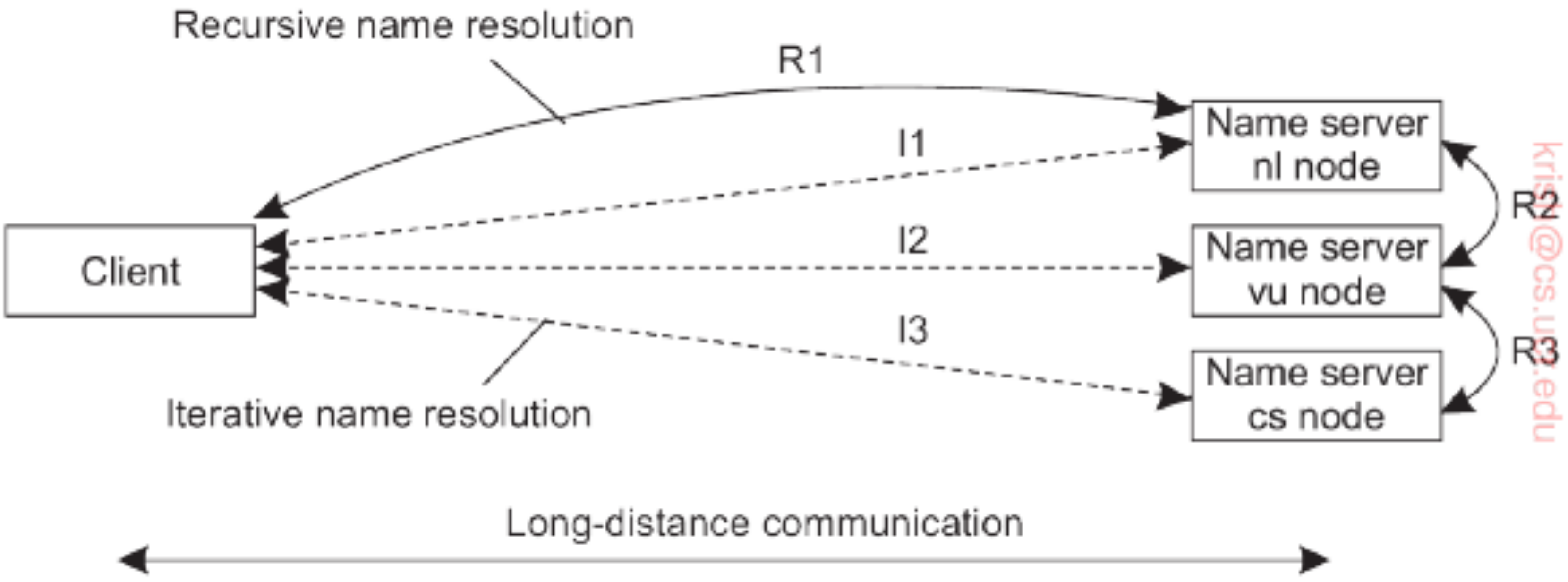
Server for node	Should resolve	Looks up	Passes to child	Receives and caches	Returns to requester
cs	[ftp]	#[ftp]	—	—	#[ftp]
vu	[cs, ftp]	#[cs]	[ftp]	#[ftp]	#[cs] #[cs, ftp]
nl	[vu, cs, ftp]	#[vu]	[cs, ftp]	#[cs] #[cs, ftp]	#[vu] #[vu, cs] #[vu, cs, ftp]
root	[nl, vu, cs, ftp]	#[nl]	[vu, cs, ftp]	#[vu]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

Caching

- Note that the higher layer entities can cache the results so that one does not need to go and resolve the names each time.
- In fact, resolved information about higher layers (layers that do not change often) can be cached at the clients
  - This information does not change much.

Reduction in communication costs with recursion

- If the client (say in LA) is far from the server (say in NL), then iterative name resolution consumes more communication cost.


Recursive name resolution
R1
I1
I2
I3
Client
Name server nl node
Name server vu node
Name server cs node
R2
R3
Iterative name resolution
Long-distance communication