IT 6204
Section 4.0

Network Administration

4.1 Network Configuration

Detecting the Ethernet card

- Run the following command and you should see all the network interface cards detected.

  ifconfig –a

- If you can see eth0 then you can start setting up TCP/IP networking.
- If you cannot see eth0 in the output of **ifconfig** then you can check whether the kernel has identified any Ethernet controllers on the PCI bus. You can use the following command for this.

  lspci

Assigning IP address to the interface

- You can use ifconfig to assign an IP address to the interfaces.

  ifconfig eth0 192.168.1.1 netmask 255.255.255.0 up

- This assigns the IP address 192.168.1.1 to the eth0 interface. This interface configuration is not persistent across *reboots*. Later, we will discuss how the network interfaces can be configured using configuration files which keep the these values across reboots.

- Assigning more than ONE IP addresses (IP Aliasing)

  ifconfig eth0:0 192.168.1.1 netmask 255.255.255.0 up
  ifconfig eth0:1 192.168.1.2 netmask 255.255.255.0 up

Setting up the routing table

- The **route** command is used to setup the **routing table**. The IP layer consults the routing table to figure out how an IP packet is sent towards the destination. When you configure the network interface the routing table picks up the **first route entry automatically**.

- You can also add the routing table entry for the own network explicitly as follows.

route add -net 192.168.1.0 netmask 255.255.255.0 eth0

Setting up the routing table cont.…

- To communicate with the rest of the Internet your network must have at least one **router that knows how to** guide your packets towards the destination in other networks. This router must have one network interface connected to your network. We can set this router as our **default gateway** as follows:

  route add default gw 192.168.1.254

Testing your NIC

You can check the routing table by using either one of the following two commands.

route

netstat –nr

Configuring name resolution

You must know the IP address of at least one DNS server, accessible from your network, to resolve names. Applications use a set of library routines (resolver library) to resolve names and these routines consult a file called **/etc/resolv.conf.**

nameserver 192.248.16.91

Testing your NIC cont….

Testing

You can test your network by sending messages to other hosts and bouncing them off those hosts. You can use the ping command for this.

ping 192.168.1.254

Testing your NIC

More Testing

You can test your network by these commands as well.

ping –i 5 127.0.0.1 (Time Interval e.g 0.1 sec)

ping –c 5 –q 127.0.0.1 (Count & Summary)

ping –s 100 127.0.0.1 (Size)

ping –w 5 destination (Timeout)

netstat -a (List all ports)

netstat -at (List all TCP ports; u - UDP)

Testing your NIC cont.….

netstat -l (List all Listening ports; t – TCP; u – UDP)

netstat -s (Statistics of all ports)

netstat -c (Print information of all ports)

netstat -r (List all Kernel Routing information)

Configuring the network

Files and Scripts

As mentioned before the IP address assigned using the ifconfig command and the routing table entries do not persists across reboots. These files are not consistent across boot the machine. There are several different distributions.

Configuring the network cont.…

The primary network configuration files are as follows:

/etc/hosts - The main purpose of this file is to resolve hostname that cannot be resolved any other way. It can also be used to resolve hostnames on small networks with no DNS server.

/etc/resolv.conf - This file specifies the IP address of DNS servers and the search domain.

Configuring the Network cont.…

Files and Scripts

/etc/sysconfig/network - Specifies routing and host information for all network interfaces.

/etc/sysconfig/network-scripts/ifcfg-<interface-name> For each network interface on a Red Hat Linux system, there is a corresponding interface configuration script. Each of these files provide information specific to a particular network interface.

NOTE: The **/etc/sysconfig/networking/** directory is used by the Network Administration Tool (system-config-network) and its contents should not be edited manually.

Configuring the Network cont.…

DEVICE=eth0
BOOTPROTO=none
ONBOOT=yes
NETWORK=192.16.1.0
NETMASK=255.255.255.0
IPADDR=192.16.1.1
USERCTL=no
BROADCAST= <address>, where <address>, is the broadcasts address. This directive is deprecated.
DEVICE=<name>, where<name>, is the name of the physical device

Most of these entries are self explanatory. BOOTPROTO can be set to dhcp to use the Dynamic Host Configuration Protocol to configure the interface. USERCTL indicates whether users other than the superuser is allowed to bring up or shut down this interface.

Dynamic Host Configuration Protocol (DHCP)

- Dynamic Host Configuration Protocol (DHCP) is a network protocol that automatically assigns TCP/IP information to client machines.
- Each DHCP client connects to the centrally located DHCP server, which returns that client's network configuration (including the IP address, gateway, and DNS servers).
- The client retrieves this information from the DHCP server.

Dynamic Host Configuration Protocol (DHCP)

- DHCP is also useful if an administrator wants to change the IP addresses of a large number of systems. Instead of reconfiguring all the systems, he can just edit one DHCP configuration file on the server for the new set of IP addresses.
- If the DNS servers for an organization changes, the changes are made on the DHCP server, not on the DHCP clients.
- When the administrator restarts the network or reboots the clients, the changes will go into effect.

Configuring a DHCP Server

- To configure a DHCP server, you must create the dhcpd.conf configuration file in the /etc/ directory.
- There are two types of statements in the configuration file:
- Parameters — State how to perform a task, whether to perform a task, or what network configuration options to send to the client.
- Declarations — Describe the topology of the network, describe the clients, provide addresses for the clients, or apply a group of parameters to a group of declarations.

Configuring a DHCP Server

- The parameters that start with the keyword option are referred to as options. These options control DHCP options; whereas, parameters configure values that are not optional or control how the DHCP server behaves.
- Parameters (including options) declared before a section enclosed in curly brackets ({ }) are considered global parameters. Global parameters apply to all the sections below it.

Configuring a DHCP Server Example

```
subnet 192.168.0.0 netmask 255.255.255.0 {
        option routers                  192.168.0.1; #Default Gateway
        option subnet-mask              255.255.255.0;
        option domain-name              "home.local";
        option domain-name-servers      192.168.0.2;
        option netbios-name-servers     192.168.0.2; #WINS Server
     range dynamic-bootp 192.168.0.51 192.168.0.100;  #DHCP Range to assign
        default-lease-time 43200;
        max-lease-time 86400;
}
```

- Save your /etc/dhcpd.conf file
- start the dhcpd dameon by “/etc/rc.d/init.d/dhcpd start” command

4.2 Configuration of a Linux Box as a router

Making your Linux Box Into a Router

- Three things are required to make a Linux box into a router.

- From a hardware standpoint you need two NICs. Each NIC must be connected to a different network segment.

- Then you need both masquerading (NAT) and forwarding enabled. All of the network configuration except installing a second NIC can be accomplished without rebooting.

Making your Linux Box Into a Router

IP Forwarding

- This turns on IP Forwarding so that packets can be forwarded from one NIC to another, or from one network or subnet to another. Usually from an internal network to an external network such as the Internet.

- You can turn on IP Forwarding immediately by changing the content of the ip_forward file in the /proc filesystem from 0 to 1 using the command:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Making your Linux Box Into a Router

- To make sure that IP Forwarding is persistent after a reboot, change one line in /etc/sysctl.conf from 0 to 1. Change the line:

  net.ipv4.ip_forward = 0 to

  net.ipv4.ip_forward = 1

Masquerading

- Masquerading modifies the packets coming from the internal network so that the return address is the same as the external NIC. IP Tables maintains an internal table with the ID of the packet transmitted out and the true source address of the packet.

Making your Linux Box Into a Router

- Configure IPTables for masquerading with the following command:

  iptables -t nat -A POSTROUTING -s 192.168.0.0/255.255.255.0 -j MASQUERADE

4.3 Configuring Web Server

Web Server Basics

- What is a web server?
  - Program that responds to requests for documents
    - "http daemon"
  - Uses the Hypertext Transfer Protocol (HTTP) to communicate
  - Physical machine which runs the program
- HTTP is…
  - Designed for document transfer
  - Generic
    - not tied to web browsers exclusively
    - can serve any data type
  - Stateless
    - no persistent client/server connection

Serving a Page

- User of client machine types in a URL
- Server name is translated to an IP address via DNS
- Client connects to server using IP address and port number
- Client determines path and file to request
- Client sends HTTP request to server
- Server determines which file to send
- Server sends response code and the document
- Connection is broken

Apache History

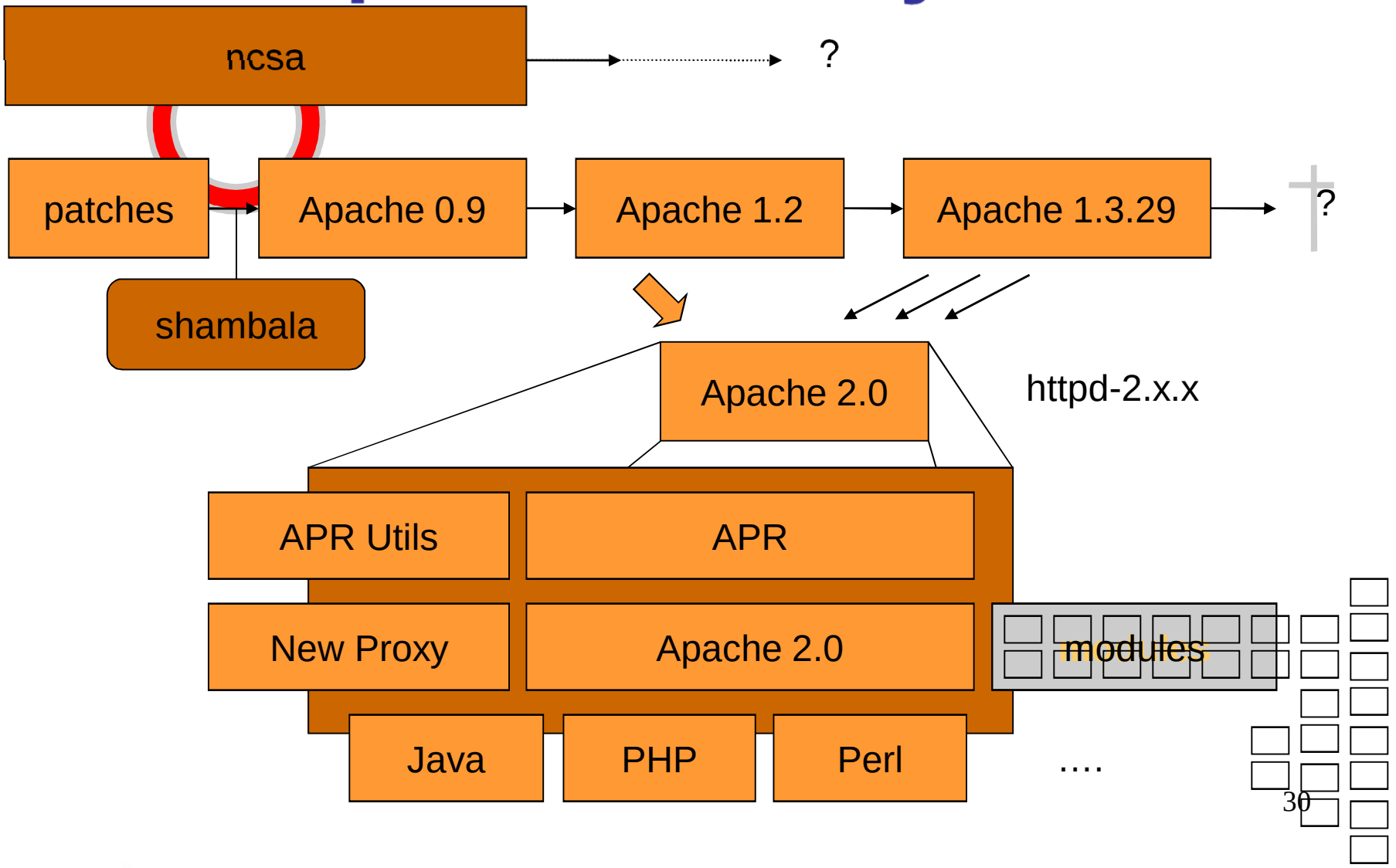
- **NCSA** (National Centre for Supercomputing Applications, Uni of Illinois) webserver was the popular public domain HTTP daemon.
- Developped by **Rob McCool**.
- **Killer** Application of the Linux.
- Rob McCool left NCSA in mid 1994.
- Many webmasters developed their own **extensions** and **bug fixes** that were in need of a common distribution.

- http://httpd.apache.org/ABOUT_APACHE.html
- http://www.ncsa.uiuc.edu/
- http://hoohoo.ncsa.uiuc.edu/

Apache History

- Server continued to grow in popularity but **incompatibilities** between versions began to develop.
- Eventually – a **small** group of administrators began working together to regain control.
- **Brian Behlendorf** and **Cliff Skolnick** put together a **mailing list** and **logins** for core developers. **8 core** contributors formed the **foundation** of the original **Apahe Group**.
- Single path developed → project came to be know as " ***a patchy*** **server** " or **Apache** server.
- Today – Apache posseses a level of complexity that easily surpasses some OS's.

Apache History



Installing Apache

rpm -Uvh httpd-2.x_NN.rpm

Source package （http://httpd.apache.org/）

- Extract the package with (Apache 2.x)
- # gzip -d httpd-2_x_*NN*.tar.gz or
- # tar xvf httpd-2_x_*NN*.tar
- # cd httpd-2.x.NN directory
- ***The next step is to configure the Apache source tree for your particular platform and personal requirements. To configure source tree simply type***
- # ./configure --enable-mods-shared=all
- (Type ./configure –help to see available options with ./configure)
- # make
- # make install

Apache Configuration

- Basic site setup:

```
…/site_home (www)
             /conf
            /logs
            /htdocs
```

- Generally Default site home is **/etc/httpd**
- The configuration file is **httpd.conf**
  - config files reside in the **conf directory**

Apache Configuration

Validating the Configration Files

/usr/local/apache/bin/apachectl configtest

Syntax OK

To start your server, type the command:

/usr/local/apache/bin/apachectl start

To stop your server, type the command:

/usr/local/apache/bin/apachectl stop

Apache Configuration

- Who runs the httpd daemon?
  - Superuser?
    - Security risk
    - Only user who can access port 80
    - Solution is for master process to be started by root, bind to socket, then change to another user
  - Nobody?
    - Not portable across UNIXES
- Creating a user and group to run the web server
  - on Linux, /usr/sbin/useradd and /usr/sbin/groupadd
  - or /etc/passwd and /etc/group
- Put user and group info in httpd.conf
  - **User apache**
  - **Group apache**

Apache Configuration

- Create new user on Linux:

```
/etc/sbin/useradd -d /home/apache -g GID -m -s
/bin/bash -u UID apache
```

  - look up GID and UID in /etc/groups and /etc/passwd, respectively
- Add the user to the httpd.conf file
  - edit /etc/httpd/conf/httpd.conf
  - add the following:
  - **User apache**
  - **Group apache**
- The httpd daemon will now start processes as “apache”
- Setting the host name
  - Add the following to httpd.conf:

    ServerName yourmachinename
  - where yourmachinename is the name used by your server

Apache Configuration

- Setting the default document directory
  - Also called “document root”
  - Example:
    - default document dir: /var/www/html (/etc/httpd/htdocs)
    - request for http://myserver/index.html will look for /etc/httpd/htdocs/index.html
- Add the following to httpd.conf

  DocumentRoot /var/www/html or DocumentRoot html

  DO NOT add a slash at the end of the directory path

- ServerRoot: The top of the directory tree under which the server's configuration, error, and log files are kept.

  ServerRoot "/etc/httpd“

Setting The Host Name

- Set the hostname in httpd.conf
  - If the host is www.foobar.com, use:
  - **ServerName www.foobar.com**
  - Designates the default host name (later - virtual host names)
  - For example, use “localhost”

- Starting the server
  - /etc/init.d/httpd start
  - /usr/local/apache/bin/apachectl start

Error Responses

- Apache can respond to an error by
  - Sending a simple default error page
  - Sending a customized error page
  - Redirecting to a local URL
  - Redirecting to an external URL
- Configured using the ErrorDocument directive in the config file
- Syntax

  ErrorDocument [HTTP code] [URL]
- Examples
  - A nicer “404 Not Found” message

    ErrorDocument 404 nodocument.html
  - Static access denied message

ErrorDocument 403 “Sorry, Dave : note single quote at start of string”

Error Responses

- Examples
  - Redirect server errors to an error logging CGI program:

    ErrorDocument 500 /cgi-bin/log-error
  - Log strange incoming requests, like DELETE

    ErrorDocument 400 /cgi-bin/log-hacks
- Some HTTP Error Codes
  - 400 Bad Request
  - 401 Unauthorized
  - 403 Forbidden
  - 404 Not Found
  - 500 Internal Server Error
  - 503 Service Unavailable
- Notes
  - Any error response page starting with “http://” will cause the server to send a redirect to the client

#ErrorDocument 402 http://www.example.com/subscription_info.html

Log Files

- First line of troubleshooting when setting up a server
- Provided flexible logging
- Logs are written in a Customizable format
- Logs can be written directly to a file or to an external program.
- Conditional logging can be made based on the characteristics of the request.
- **Directives** provided for this,
  - `TransferLog` – To create log file
  - `LogFormat` - To set a custom format
  - `CustomLog` - To define a log file and format
- TransferLog & CustomLog directives can be used multiple times in each server to cause each request to be logged to multiple files.
- Important to remember log file rotation as well.

Log Formats

- Define the log locations in httpd.conf:
  - ErrorLog - logs server errors

    ErrorLog "/var/log/httpd/error_log“ (/etc/httpd/logs/errors)
  - TransferLog - logs user requests and results

    TransferLog “/var/log/httpd/access_log” (/etc/httpd/logs/access)
- TransferLog Format (Common Log Format – CLF)

  192.216.5.178 - - [06/Oct/1999:20:03:47 -0700] "GET / HTTP/1.0" 200 1945
  - IP address of client
  - The clients’s identity & the remote user name if using HTTP authentication (missing in example “- -”)
  - Date, Time and Time Zone of the request

Log Formats

- Content HTTP request
- Server Response code
- Content length in bytes

- The default CLF can be altered to store more information using the **`LogFormat`** directive.

Log Formats

- ErrorLog Format

[Wed Oct 6 20:06:04 1999] [error] [client 192.216.5.178]
File does not exist: /home/httpd/html/foobar

  - Date
  - Error, Information, Security, etc.
  - Client IP address
  - Message

Current state of our config file

- User apache
- Group apache
- ServerName localhost
- DocumentRoot /var/www/html
- TransferLog /var/logs/httpd/access_log
- ErrorLog /var/logs/httpd/errors_log

Log Formats

- LogFormat “%H %m %t %U” simple
- CustomLog logs/access.log simple

This will logs Protocol,Date,Time and URL requested

Exercise:

Try these with your configured apache server.

- LogFormat “%h” ip
- LogFormat %h %l %u %t \”%r\” %>s %b” detailed
- CustomLog logs/access.log detailed
- CustomLog logs/ip.log ip

Basic Server Settings

- Server can set the Content-type header several ways
  - DefaultType directive
    - example: DefaultType text/html (application/octet-stream)
    - sets the type for any document not otherwise recognized
  - AddType directive
    - example: AddType image/jpeg (AddType application/x-tar .tgz)
    - associates MIME type with file extension or override the MIME configuration
    - can be used to set multiple types

Basic Server Settings

- TypesConfig directive
  - TypesConfig /etc/mime.types
  - Describes where the mime.types file (or equivalent) is to be found.
    in the default installation, this is /etc/mime.types
  - by far the easiest way to associate MIME type with file extension – needs an Apache module to work!

Basic Server Settings

- Our MIME-smart httpd.conf

```
LoadModule mime_module
   /etc/httpd/modules/mod_mime.so
User www
Group www
ServerRoot /home/www
ServerName localhost
DocumentRoot /home/www/htdocs
ErrorLog logs/errors
TypesConfig /etc/mime.types
DefaultType text/plain
```

Apache is Modular

- Loadable modules
  - Routines which extend the server
  - Module examples:
    - MIME type determination
    - **`Server-parsed documents`**
    - CGI handler
  - the Apache group distributes 34 modules
  - **LoadModule** directive selectively includes modules in server

Apache is Modular

- Loadable Modules vs. Compiled Modules
  - At install time, you have the option of including any of the loadable modules in the base server
    - increases server executable size
    - easier to access module directives, like TypesConfig
  - Which is better? Personal preference

Apache is Modular

- How do I know which directives are in which modules?
  - The quick reference in textbook lists directives
  - If they are “core,” don’t need LoadModule
  - **`$ httpd -l`** list all the statically compiled modules
  - Try the below online docs which show module names with directive documentation:

    http://www.apache.org/docs/mod/directives.html

The Options Directive

- **Options** directive
  - quick way to set several server features
  - Core directive (not in a module)
  - Applies to entire server, virtual host, directory
  - Uses **+** to turn on an option and **-** to turn it off
- ExecCGI
  - allow execution of CGI scripts
- FollowSymLinks
  - server will follow UNIX symbolic links (ln -s source target) to find files
- SymLinksIfOwnerMatch
  - only follow symlinks if the target file is owned by the link owner

The Options Directive

- Indexes
  - if a URL that maps to a directory is requested and there is no index.html, display a file index
- Includes
  - allow server-side includes (SSI later)
- IncludesNoEXEC
  - SSI allowed, but #exec is disabled
- MultiViews
  - complex file mapping function
  - off by default
- All
  - all options except MultiViews

The Options Directive

- Syntax
  - Options Indexes FollowSymLinks
  - Options +Includes -Indexes
  - Options -ExecCGI
- Using + and - merges options
  - used in <Directory> and <Files> blocks

File and Directory Control

- Config file directives so far can be divided into two groups:
  - ***Server process control***
    - User, Group, ServerRoot, LoadModule, DocumentRoot, ErrorLog, TypesConfig…
  - ***File and Directory control***
    - ErrorDocument, DefaultType, AddType, Options…

- How do we specify different settings for different directories?
  - <Directory> and <DirectoryMatch>
- How do we specify different settings for specific files or file types?
  - <Files> and <FilesMatch>

<Directory>

- What is it?
  - Encloses a group of config file directives which will apply only to a specified directory and its subdirectories
  - Most commands not related to the server process can be used

- Syntax
  - Example: set file indexing and symlink following in the smallco area

```
<Directory /clients/smallco>
        Options Indexes FollowSymLinks
</Directory>
```

<Directory>

- Syntax
  - Example: specify a special “404 not found” document for Cars

```
<Directory /clients/cars>
        ErrorDocument 404 cars404.html
</Directory>
```

- Syntax
  - The directory path is a full path to the directory affected
  - Wild cards * and ? may be used
    - * is match any sequence of characters
    - ? is match any single character
    - may also use [] to enclose character ranges

ie “site_v[1-4]” will match “site_v1”, “site_v2”, “site_v3” and “site_v4”

<Directory>

- Syntax
  - Can also use regular expressions with the tilde character
    - <Directory ~ "^/clients/(cars|vans)">
    - matches /clients/cars and /clients/vans but not /web/clients/cars
    - Don't worry about regular expressions if you haven't learned about them yet!!!

<DirectoryMatch>

- What is it?
  - Exactly the same as
  - <Directory ~ regex>

<Location>

- What is it?
  - Encloses a group of config file directives that apply only to a given URL
  - Similar to <Directory>, but operates on the requested URL
- Why do we need it?
  - Some directives alter the URL to get the file path

- Usage
  - Most useful in conjunction with the **SetHandler** directive
    - SetHandler specifies which program will be used to handle a request

- <LocationMatch>
  - same as <Location ~ regex>

<Files>

- What is it?
  - Allows directives to be applied to only certain files or file types
  - Can be nested inside <Directory> tags
  - Processed in the order they appear in the httpd.conf file
- Example:
  - Make any file with a .foo extension be served as text/plain

```
<Files *.foo>
        ForceType text/plain
</Files>
```

  - Guess what ForceType does…
- Syntax
  - <Files **filename**>
  - Can use * and ? wildcards
  - Can use regular expression with <Files ~ regex> syntax
  - Can be used both inside and outside a <Directory> section

Virtual Hosting

Virtual Hosts

- More than one apparent server on one machine
  - One instance of Apache can serve multiple web sites
  - ISPs do this all the time
  - Can also be used for intranets to separate departmental sites, for example

- Virtual Host types
  - **IP-based virtual hosts**
    - Most common method
    - Requires different IP address for each virtual host
    - Requires configuration of network interface card
    - Supported under HTTP/1.0

Virtual Hosts

- Virtual Host types
  - **Name-based virtual hosts**
    - many host names pointing to same IP address
    - practically unlimited number of servers
    - easy to configure
    - no additional hardware or software
    - BUT client must support HTTP/1.1
      - old browsers may lack support

<VirtualHost> can include:
<Directory>, <Files>, and <Location> inside a <VirtualHost>
These inclusions are processed after the ones outside the <VirtualHost>

Virtual Hosts Example

- Example Scenario
  - We are BigISP.com, an service provider running Apache on Linux
  - We have two clients, Smallco and Bigco, Inc.
  - We want to set up web sites for these companies at:
    - www.smallco.com
    - www.bigco.com

Virtual Hosts Example

- Setup
  - We're going to create the following directories on our server:
    - /home/www server root
    - /home/www/conf server config files
    - /home/www/logs default log file dir
    - /home/www/htdocs default doc dir
    - /clients/smallco Smallco's area
    - /clients/bigco Bigco's area
      - under each client's directory, create logs and htdocs/html

IP-Based Virtual Hosts

- Adding an IP address under Linux
  - Not needed for name-based hosting
  - Assumptions
    - there is already one network card (eth0) up and running with an IP address
    - you have been assigned additional IP addresses for the machine
    - you are logged in as root
- Adding an IP address under Linux
  - Add two more IP addresses to eth0:
  - ifconfig eth0:0 192.168.123.2
  - route add -host 192.168.123.2
  - ifconfig eth0:1 192.168.123.3
  - route add -host 192.168.123.3
  - Check by running ifconfig
  - To make this permanent, add these lines to /etc/rc.d/rc.local

IP-Based Virtual Hosts

- Need to associate virtual host names with new IP addresses
  - If you were assigned IP addresses by network admin, this may already be set up
  - If not, add the following to ***/etc/hosts*** to associate names with IP numbers
    - 192.168.123.2 www.smallco.com
    - 192.168.123.3 www.bigco.com
- The <VirtualHost> directive
  - contains directives we've already seen, like
    - ServerName
    - DocumentRoot
    - ErrorLog
    - TransferLog
    - ServerAdmin
  - applies these settings to requests that come in for the specified host

IP-Based Virtual Hosts

- Example Scenario

```
<VirtualHost www.smallco.com>
    ServerName www.smallco.com
    ServerAdmin webmaster@mail.smallco.com
    DocumentRoot /clients/smallco/htdocs
    ErrorLog /clients/smallco/logs/errors
    TransferLog /clients/smallco/logs/access
</VirtualHost>
```

```
<VirtualHost www.bigco.com>
        ServerName www.bigco.com
        ServerAdmin root@mail.bigco.com
        DocumentRoot /clients/bigco/htdocs
        ErrorLog /clients/bigco/logs/errors
        TransferLog /clients/bigco/logs/access
</VirtualHost>
```

IP-Based Virtual Hosts

- The <VirtualHost> directive
  - The hostname can be either a name or an IP address
  - if a name is used, the IP address is looked up via DNS
    - Pro: easy to administer
    - Con: if DNS is down when Apache is started, so is the virtual server
- The <VirtualHost> directive
  - if an IP address is used and ServerName is not specified, a reverse DNS lookup will be performed to get the name
    - Con: DNS down, name-based requests to the server are down
  - **Solution: use IP address in <VirtualHost> with ServerName directive**

IP-Based Virtual Hosts

Revised example:

```
<VirtualHost 192.168.123.2>
    ServerName www.smallco.com
    ServerAdmin webmaster@mail.smallco.com
    DocumentRoot /clients/smallco/htdocs
    ErrorLog /clients/smallco/logs/errors
    TransferLog /clients/smallco/logs/access
</VirtualHost>
```

– similar revision for www.bigco.com

Name-Based Virtual Hosts

Example Scenario #2

- Smallco and Bigco Inc. have arranged for www.smallco.com and www.bigco.com to point to our IP address, 192.168.123.1
  - BigISP was assigned the primary name server for those names
  - This is a network administrator task
- Our server is server.bigisp.com
- Our /etc/hosts may look like

  192.168.123.1 server.bigisp.com

- Need to add additional hostnames after “server.bigisp.com”
- 192.168.123.1 server.bigisp.com www.smallco.com www.bigco.com

Name-Based Virtual Hosts

Config file

```
NameVirtualHost 192.168.123.1
<VirtualHost 192.168.123.1>
    ServerName www.smallco.com
    ServerAdmin webmaster@mail.smallco.com
    DocumentRoot /clients/smallco/htdocs
    ErrorLog /clients/smallco/logs/errors
    TransferLog /clients/smallco/logs/access
</VirtualHost>
```

```
<VirtualHost 192.168.123.1>
    ServerName www.bigco.com
    ServerAdmin root@mail.bigco.com
    DocumentRoot /clients/bigco/htdocs
    ErrorLog /clients/bigco/logs/errors
    TransferLog /clients/bigco/logs/access
</VirtualHost>
```

Name-Based Virtual Hosts

Notes

- Main server at /home/www no longer available
  - can get around that by adding a new <VirtualHost> entry for server.bigisp.com
- Older browsers will not work
  - first virtual host in config file always used
  - possible workaround with the **ServerPath** directive

Older browser workaround

```
NameVirtualHost 192.168.123.1
<VirtualHost 192.168.123.1>
    ServerName www.smallco.com
    ServerPath /smallco
    DocumentRoot /clients/smallco/htdocs
</VirtualHost>
```

Name-Based Virtual Hosts

ServerPath workaround

- Any request starting with /smallco will be served by this virtual host
- Pages must be accessed as www.smallco.com/smallco
- Requires that any pages in the site use only relative links or /smallco/…
- Newer browsers unaffected

Virtual Hosts

- Two ways of running Apache for virtual hosts
  - Multiple httpd daemons
    - secure – vhost1 cannot read vhost2's data
    - host machine has enough system resources
  - Single httpd daemon
    - some shared configuration is acceptable
    - host machine will service a high volume of requests

- Setting up multiple daemons
  - requires multiple httpd installations

- Setting up a single daemon
  - One site home and config file
  - Config file contains the <VirtualHost> directive

Virtual Hosts

Almost any configuration directive can be put inside <VirtualHost>

- Exceptions are mainly directives that control the httpd daemon, like
  - User, Group
  - ServerRoot
  - BindAddress
  - MinSpareServers, MaxSpareServers, MaxRequestsPerChild

Virtual Hosts

Example Scenario #3

- Bigco Inc merges with Medium Corp
- Web sites are consolidated, so requests to www.mediumco.com should now go to www.bigco.com
- Medium Corp has designated BigISP as their primary nameserver
- Add www.mediumco.com to /etc/hosts
- Use ServerAlias directive

Virtual Hosts

Example Scenario #3

```
<VirtualHost 192.168.123.1>
    ServerName www.gizmos.com
    ServerAlias www.widgets.com
    ServerAdmin root@mail.gizmos.com
    DocumentRoot /clients/gizmos/htdocs
    ErrorLog /clients/gizmos/logs/errors
    TransferLog /clients/gizmos/logs/access
</VirtualHost>
```

Virtual Hosts – Example 1

- **Serving the same content on different IP addresses (such as an internal and external address)**

```
NameVirtualHost 192.168.1.1
NameVirtualHost 172.20.30.40

<VirtualHost 192.168.1.1 172.20.30.40>
DocumentRoot /www/server1
ServerName server.example.com
ServerAlias server
</VirtualHost>
```

Virtual Hosts – Example 2

- **Mixed name-based and IP-based vhosts**

Listen 80

NameVirtualHost 172.20.30.40

<VirtualHost 172.20.30.40>
DocumentRoot /www/example1
ServerName www.example1.com
</VirtualHost>

<VirtualHost 172.20.30.40>
DocumentRoot /www/example2
ServerName www.example2.org
</VirtualHost>

<VirtualHost 172.20.30.40>
DocumentRoot /www/example3
ServerName www.example3.net
</VirtualHost>

IP-based
<VirtualHost 172.20.30.50>
DocumentRoot /www/example4
ServerName www.example4.edu
</VirtualHost>

<VirtualHost 172.20.30.60>
DocumentRoot /www/example5
ServerName www.example5.gov
</VirtualHost>

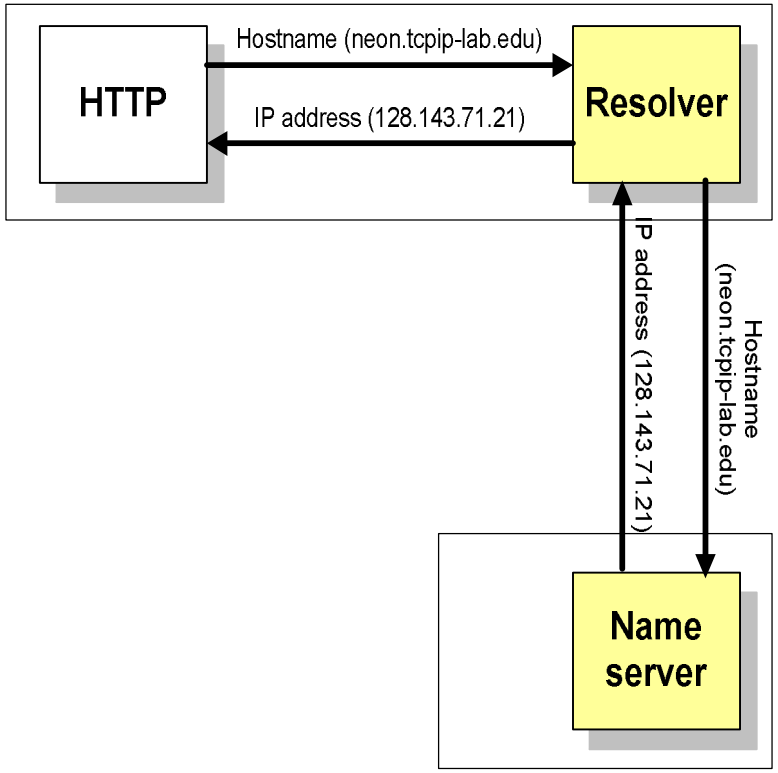
4.4 Configuring DNS Server

What is DNS?

- DNS (Domain Name System)
  - A database that is used by TCP/IP applications to map between hostnames and IP addresses
  - Characteristics of DNS
    - A hierarchical namespace for hosts and IP addresses
    - A host table implemented as a distributed database
    - A Client/Server system
  - Components of DNS
    - Namespace and Resource Record
    - Name Server
    - Resolver (Client)

Domain name resolution

- User program issues a request for the IP address of a hostname
- Local resolver formulates a DNS query to the name server of the host
- Name server checks if it is authorized to answer the query.
  - If yes, it responds.
  - Otherwise, it will query other name servers
- When the name server has the answer it sends it to the resolver.



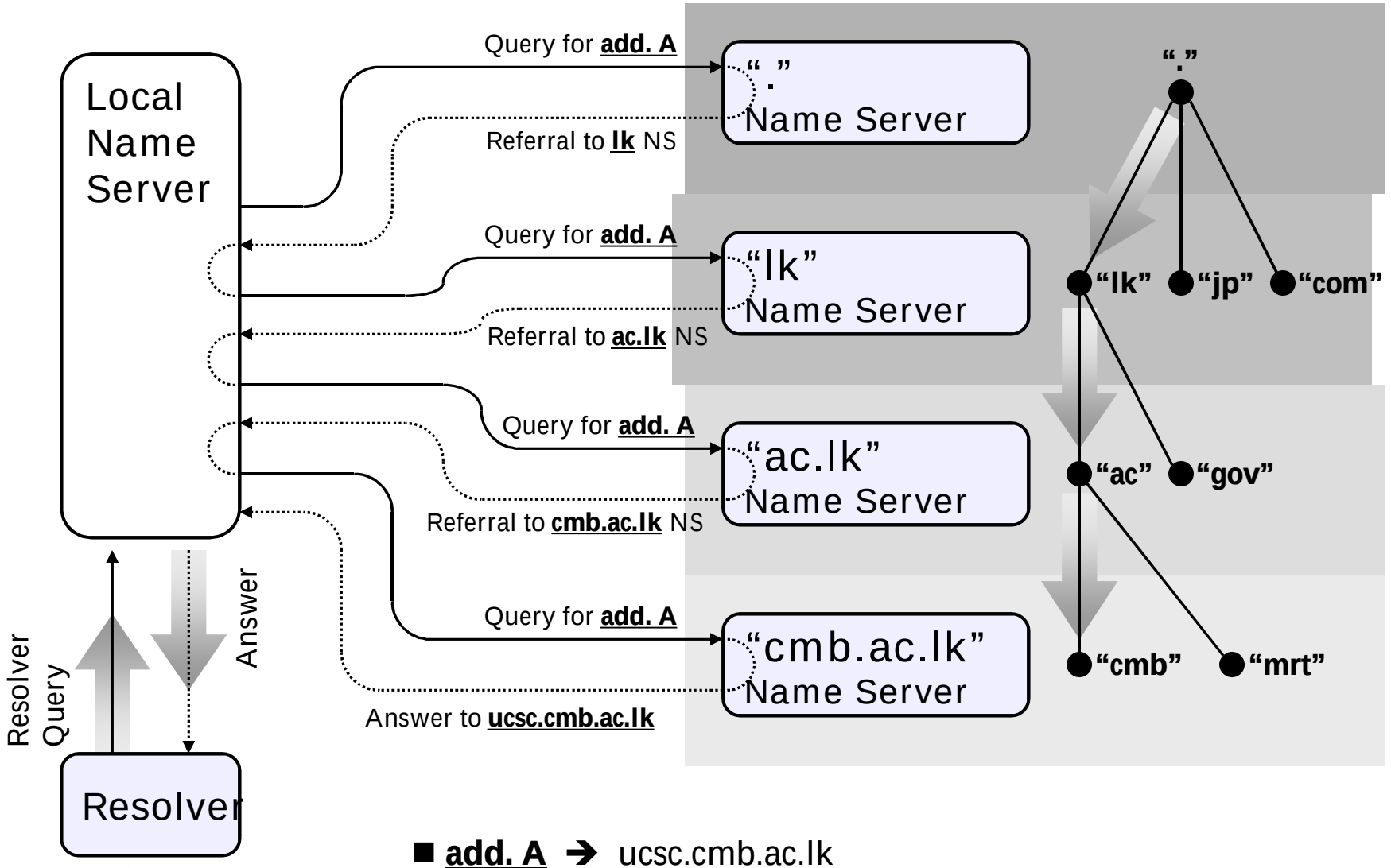
Recursive and Iterative Queries

- There are two types of queries:
  - ***Recursive queries***
  - ***Iterative (non-recursive) queries***

- The type of query is determined by a bit in the DNS query

- Recursive query: When the name server of a host cannot resolve a query, the server issues a query to resolve the query
- Iterative queries: When the name server of a host cannot resolve a query, it sends a referral to another server to the resolver

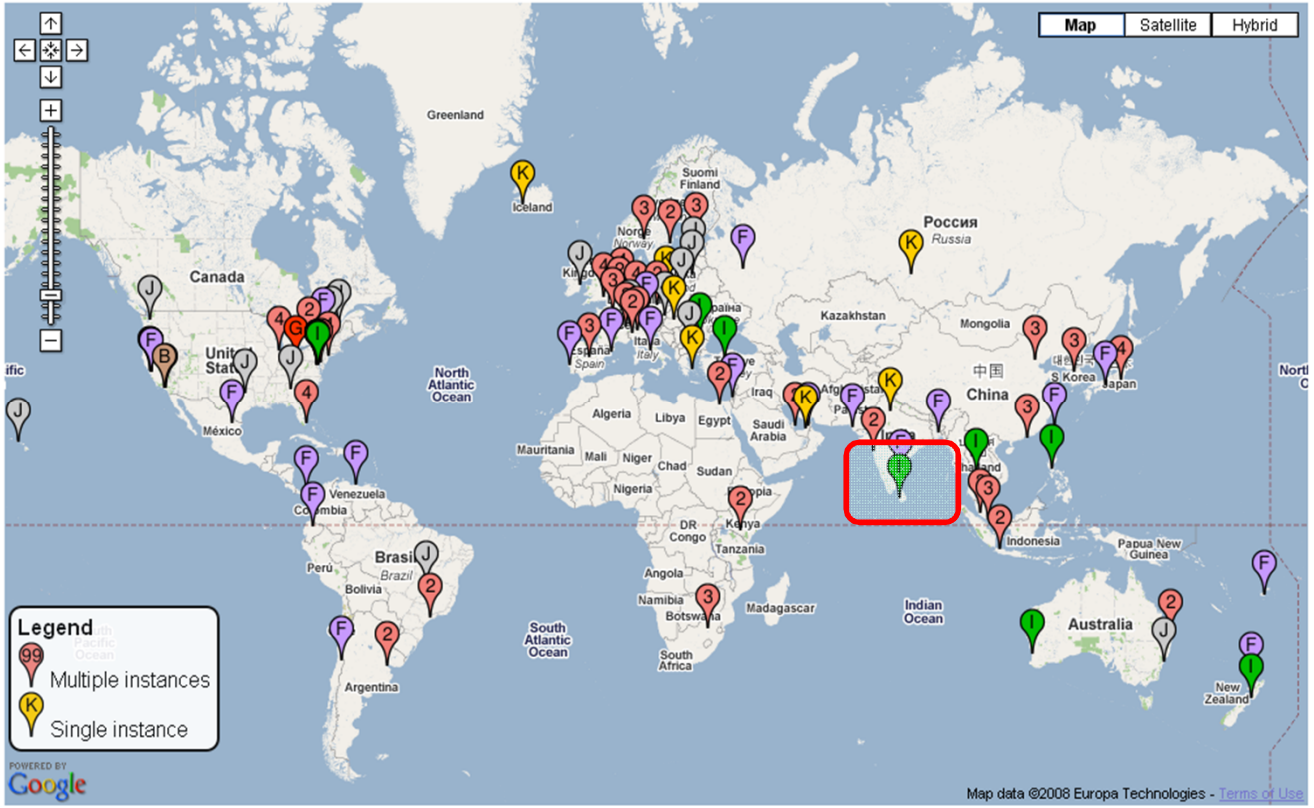
What is DNS? Cont....



- **add. A** ➔ ucsc.cmb.ac.lk

Root Servers

http://www.root-servers.org



I	Autonomica	**Sites: 32** Stockholm, SE; Helsinki, FI; Milan, IT; London, UK; Geneva, CH; Amsterdam, NL; Oslo, NO; Bangkok, TH; Hong Kong, HK; Brussels, BE; Frankfurt, DE; Ankara, TR; Bucharest, RO; Chicago, IL, US; Washington, DC, US; Tokyo, JP; Kuala Lumpur, MY; Palo Alto, CA, US; Jakarta, ID; Wellington, NZ; Johannesburg, ZA; Perth, AU; San Francisco, CA, US; New York, NY, US; Singapore, SG; Miami, FL, US; Ashburn, VA, US; Mumbai, IN; Beijing, CN; Manila, PH; Doha, QA; Colombo, LK	IPv4: 192.36.148.17	29216
J	VeriSign, Inc.	**Sites: 52** Dulles, VA, US (3 sites); Ashburn, VA, US *; Vienna, VA, US; Miami, FL, US; Atlanta, GA, US; Seattle, WA, US; Chicago, IL, US; New York, NY, US; Los Angeles, CA, US; Honolulu, HI, US; Mountain View, CA, US (2 sites); San Francisco, CA, US (2 sites) *; Dallas, TX,	IPv4: 192.58.128.30 IPv6: 2001:503:C27::2:30	26415

UCSC

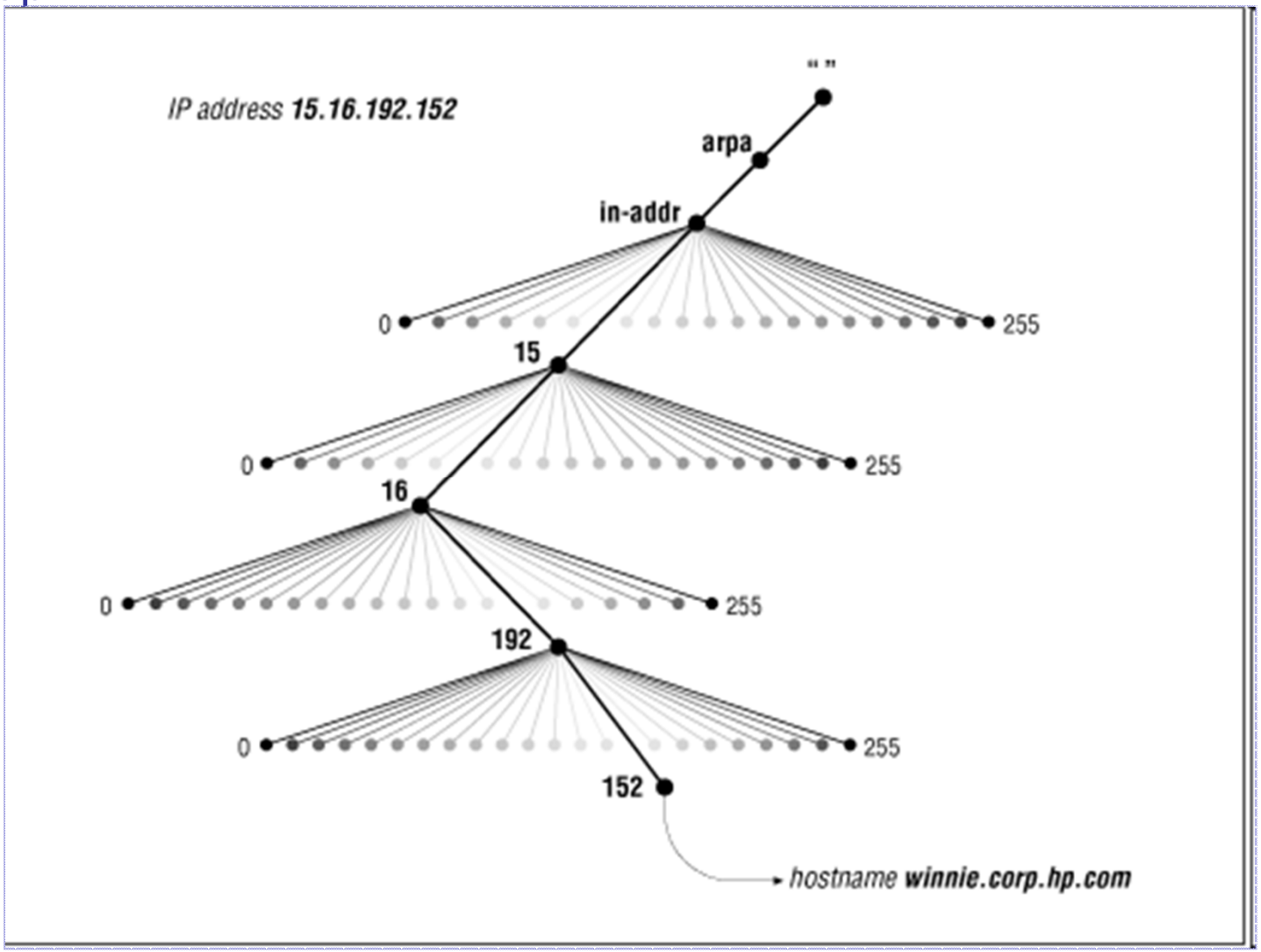
What is DNS? Cont.…

- Top Level Domains

Domain Suffix	Type of Organization
ARPA	Reverse lookup domain (special Internet function)
COM	Commercial
EDU	Educational
GOV	Government
ORG	Non-commercial organization (such as a nonprofit agency)
NET	Network (such as an ISP)
INT	International Treaty Organization
MIL	U.S. military organization
BIZ	Businesses
INFO	Unrestricted use
AERO	Air-transport industry
COOP	Cooperatives
MUSEUM	Museums
NAME	Individuals
PRO	Professionals (such as doctors, lawyers, and engineers)

What is DNS? Cont.…

- Reverse lookup



UCSC

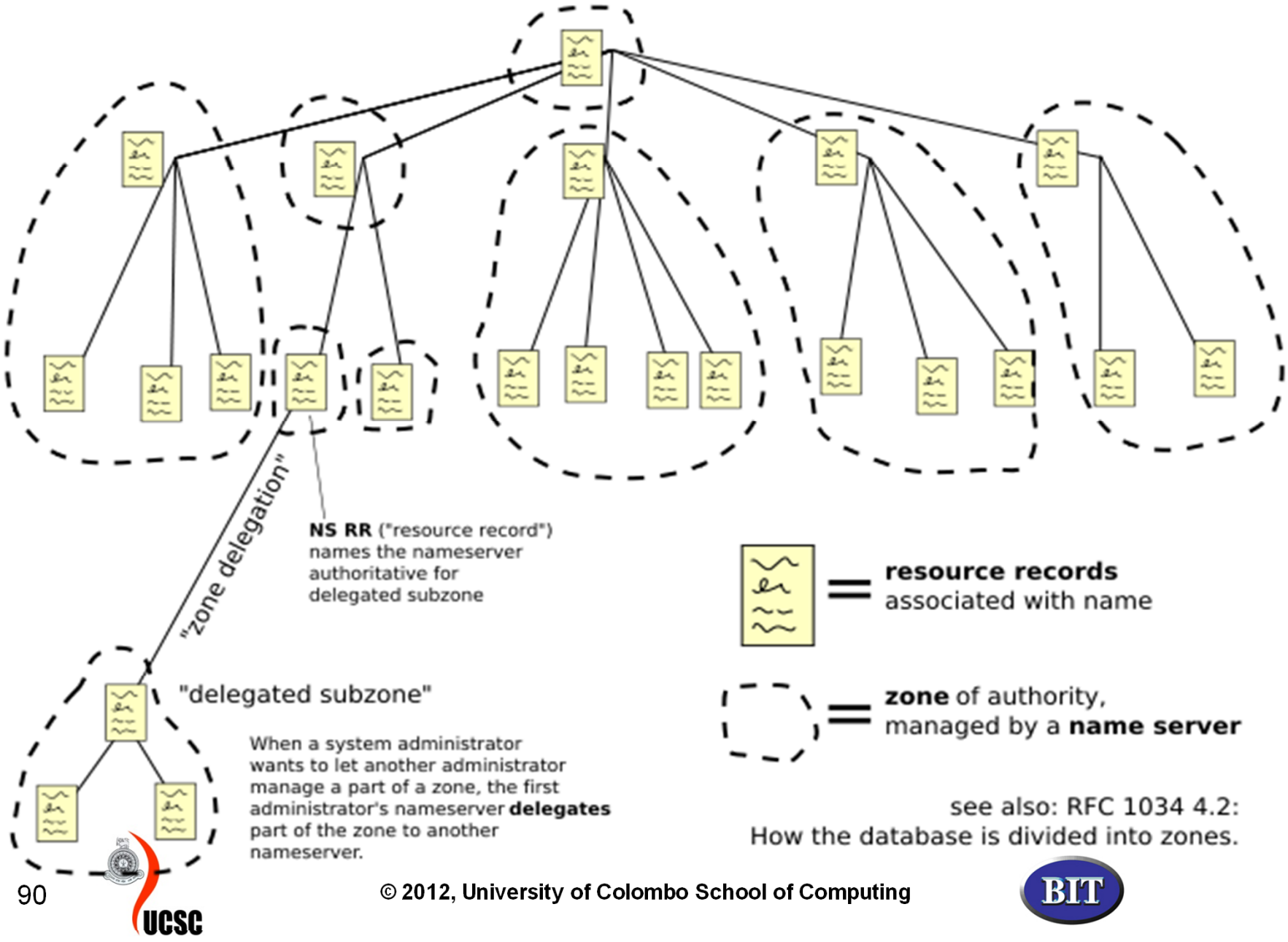
What is DNS? Cont.…

- Namespace
  - DNS namespace is a tree of “domains”
  - Refers to the actual database of IP addresses and their associated names
  - At the highest level of the hierarchy sit the **root servers**
- Zone
  - A DNS zone refers to a certain portion or administrative space within the global Domain Name System. Each DNS zone represents a boundary of authority subject to management by certain entities and is administered as a single separate entity. The total of all DNS zones, which are organized in a hierarchical tree-like order of cascading lower-level domains, form the DNS namespace.

What is DNS? Cont.…

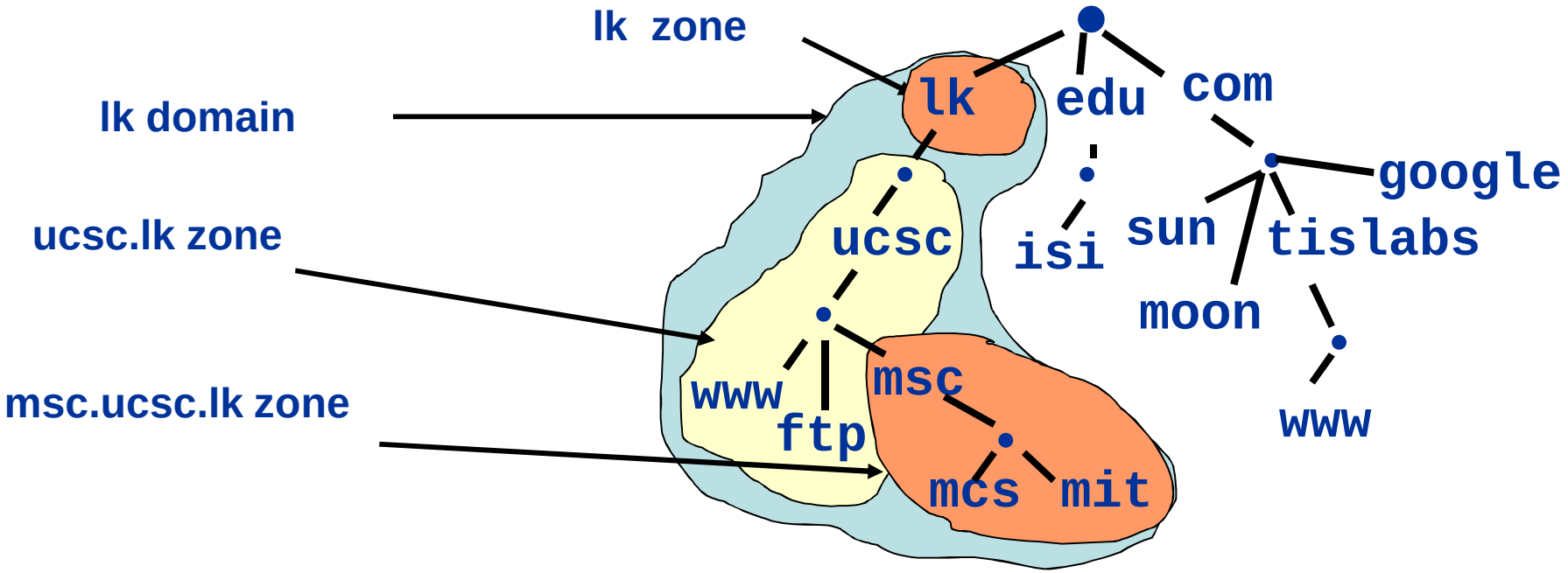
- Resource Records (RR)
  - Resource records are the data elements that define the structure and content of the domain name space. All DNS operations are ultimately formulated in terms of resource records.
- Name Server
  - The server programs that store information about the domain name space
- Resolver (Client)
  - The programs that extract information from name servers in response to client requests

Domain Name Space

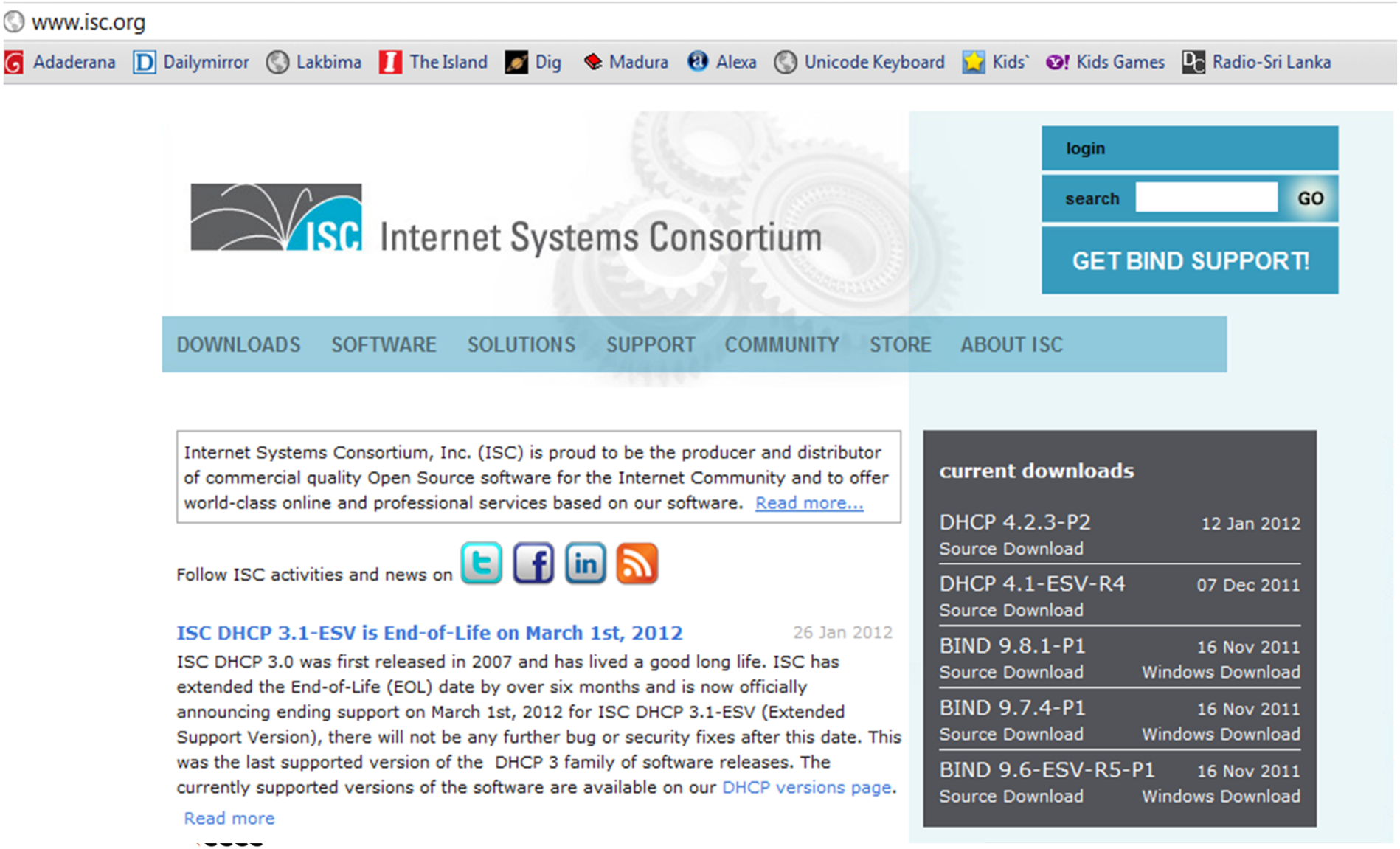


Zones and Delegations

- Zones are “administrative spaces”
- Zone administrators are responsible for portion of a domain’s name space
- Authority is delegated from a parent and to a child



What is BIND?


www.isc.org
Adaderana
Dailymirror
Lakbima
The Island
Dig
Madura
Alexa
Unicode Keyboard
Kids`
Kids Games
Radio-Sri Lanka
login
search
GO
GET BIND SUPPORT!
Internet Systems Consortium
DOWNLOADS
SOFTWARE
SOLUTIONS
SUPPORT
COMMUNITY
STORE
ABOUT ISC
Internet Systems Consortium, Inc. (ISC) is proud to be the producer and distributor of commercial quality Open Source software for the Internet Community and to offer world-class online and professional services based on our software. Read more...
Follow ISC activities and news on
ISC DHCP 3.1-ESV is End-of-Life on March 1st, 2012
26 Jan 2012
ISC DHCP 3.0 was first released in 2007 and has lived a good long life. ISC has extended the End-of-Life (EOL) date by over six months and is now officially announcing ending support on March 1st, 2012 for ISC DHCP 3.1-ESV (Extended Support Version), there will not be any further bug or security fixes after this date. This was the last supported version of the DHCP 3 family of software releases. The currently supported versions of the software are available on our DHCP versions page.
Read more
current downloads
DHCP 4.2.3-P2 12 Jan 2012
Source Download
DHCP 4.1-ESV-R4 07 Dec 2011
Source Download
BIND 9.8.1-P1 16 Nov 2011
Source Download Windows Download
BIND 9.7.4-P1 16 Nov 2011
Source Download Windows Download
BIND 9.6-ESV-R5-P1 16 Nov 2011
Source Download Windows Download

What is BIND?

- BIND (Berkeley Internet Name Domain system)
  - A open source software package that implements the DNS protocol and provides name service on systems (UNIX & NT)
  - Characteristics of BIND
    - Same as DNS, a Client/Server system
    - Client side : resolver & Server side : ***named***
  - Components of BIND
    - DNS Server (***named***)
      Answers queries about hostname and IP addresses
      Asks other servers and caches their responses
      zone transfers
    - DNS Resolver library
      Contains the routines that you need to write your application
      May use the generate query or the name server library routines
    - Tools for verifying the proper operation of the DNS server
      nslookup & ***dig***

Where do I Start? - Client

- Client Configuration
  - **/etc/resolv.conf**
    - Format

search list	**domainname**	// define your resolver's default domain and search
domain	**domainname** …	// define your resolver's default domain
nameserver server	**ipaddr**	// tells your resolver to query a particular name

    - Example

```
% more /etc/resolv.conf
nameserver 203.252.57.2
nameserver 203.252.32.4
domain cmb.ac.lk
```

    - Check : /etc/nsswitch.conf – which to be used first? DNS or /etc/hosts file?

```
# /etc/nsswitch.conf:
hosts: files dns
……..
```


Where do I Start? - Server

- Type of Server
  - Primary
  - Secondary
  - Cache only
  - Stub Server
- Install of BIND
  - Distribution : ISC (Internet Software Consortium)
- Configure Name Server
  - Network configuration
  - BIND boot file configuration
    - BIND-4 boot file : named.boot : script like code
    - BIND-8 boot file : named.conf : C like code
  - Resource Record configuration

/etc/named.conf file

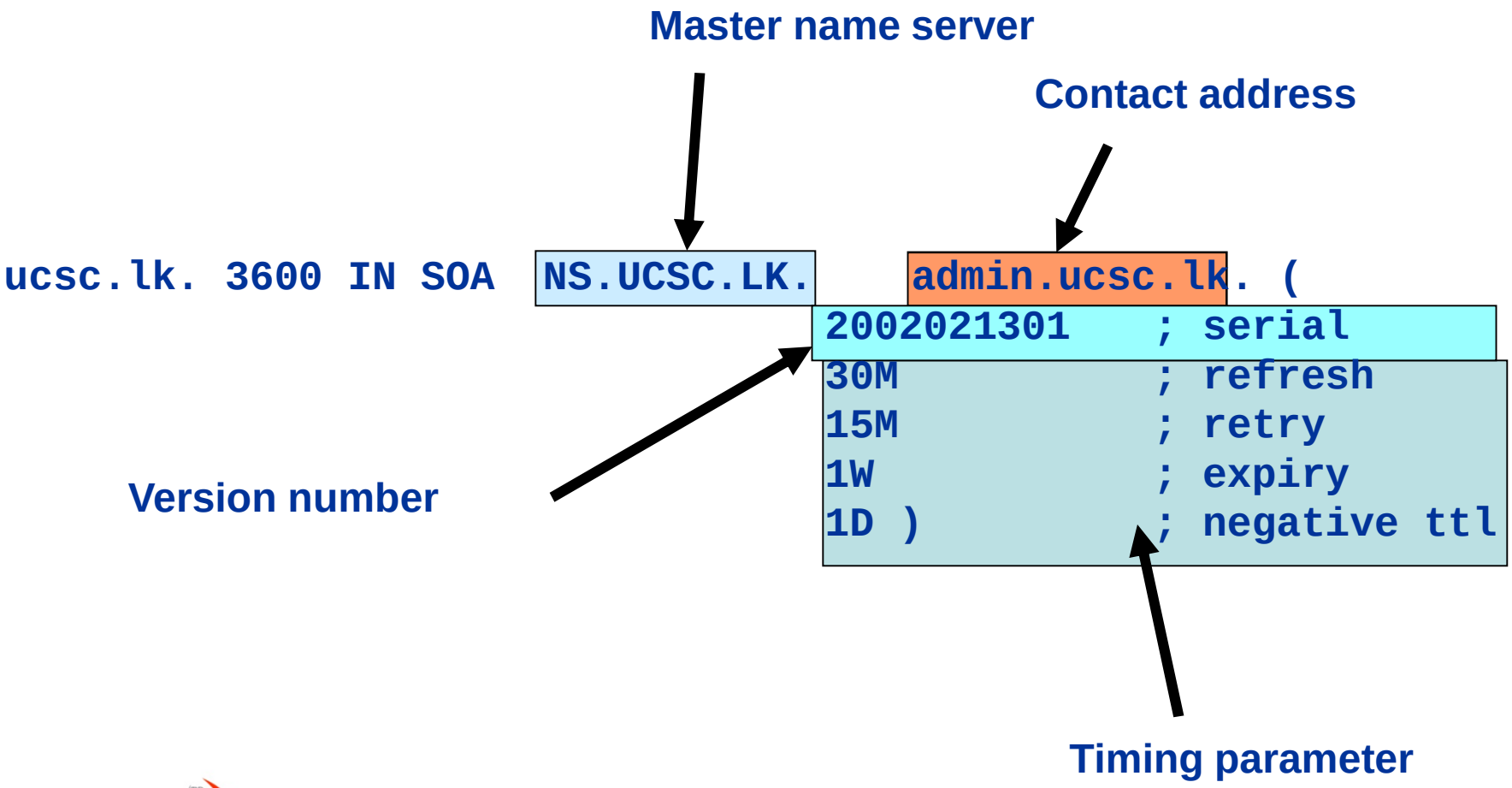
- /etc/named.conf

```
options {
        directory "/var/named";
};

zone "."  IN {                                  // root servers file
        type hint;
        file "root.hints";
};

zone “cmb.ac.lk" IN {                           // for forward zone
        type master;
        file "zone/cmb.ac.lk";
};

zone "0.0.127.in-addr.arpa" IN {                // for localhost
        type master;
        file "zone/127.0.0";
};

zone “20.168.192.in-addr.arpa" IN {             // for reverse zone
        type master;
        file "zone/192.168.20";
};
```

Resource Records (RRs)

Type	Value	meaning
A	**1**	**a host address**
NS	**2**	**an authoritative name server**
MD	3	a mail destination (Obsolete - use MX)
MF	4	a mail forwarder (Obsolete - use MX)
CNAME	**5**	**the canonical name for an alias**
SOA	**6**	**marks the start of a zone of authority**
MB	7	a mailbox domain name (EXPERIMENTAL)
MG	8	a mail group member (EXPERIMENTAL)
MR	9	a mail rename domain name (EXPERIMENTAL)
NULL	10	a null RR (EXPERIMENTAL)
WKS	11	a well known service description
PTR	**12**	**a domain name pointer**
HINFO	**13**	**host information**
MINFO	14	mailbox or mail list information
MX	**15**	**mail exchange**
TXT	**16**	**text strings**

Resource Record: SOA (Start Of Authority)

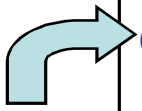


Resource Records (RRs) cont…

- Example
  - **SOA** Record

```
cmb.ac.lk.          IN    SOA   ns.cmb.ac.lk.   admin.cmb.ac.lk. (
                                  2003081001          ; Serial (2003-08-10 #01)
                                  86400               ; Refresh (daily)
                                  1800                ; Retry (30 minute)
                                  1209600             ; Expire (2 weeks)
                                  86400  )            ; Minimum TTL (1 day)
; end of SOA
:
```

NS (Name Server) Record



or replace with @

```
cmb.ac.lk.          IN     NS    ns.cmb.ac.lk.
                           NS    ns2.cmb.ac.lk.
                           NS     ns.ac.lk.
```

Resource Records (RRs) cont…

– **A** (Address) and CNAME (Canonical Name) Record

```
; Host Address
ns.cmb.ac.lk.      IN    A        192.168.20.100
ns2                IN    A        192.168.30.100
localhost                  IN     A        127.0.0.1

; Aliases
www                IN    CNAME    namal
ftp                IN    CNAME    www
```

Resource Records (RRs) cont…

- **MX** (Mail eXchanger) Record

@	IN	MX	10	mail.cmb.ac.lk.
	IN	MX	20	namal.cmb.ac.lk.
mail	IN	A		192.168.20.50
namal	IN	A		192.168.20.55

- **PTR** (Pointer) Record

50.20.168.192.in-addr.arpa.	IN	PTR	mail.cmb.ac.lk.

Glue Record

- A **glue record** is the IP address of a name server held at the domain name registry. Glue records are required when you wish to set the **name servers of a domain name** to a **hostname** under the domain name itself.

- Example:

  set the name servers of **cmb.ac.lk** to **anduna.cmb.ac.lk** and **ns.ac.lk**

  you would need to also provide the glue records (i.e. the IP addresses) for anduna.cmb.ac.lk and ns.ac.lk.

Glue Record

- If you did not provide the glue records for these name servers then your domain name would not work as anyone requiring DNS information for it would get stuck in a loop.

- What is the name server for **cmb.ac.lk**? →
  aduna.cmb.ac.lk
- What is the IP address of **aduna.cmb.ac.lk**? →
  don't know, try looking at name server for cmb.ac.lk
- What is the name server for **cmb.ac.lk**? →
  aduna.cmb.ac.lk

More on DNS

RFC 1537 recommends the following values for top-level domain servers in the SOA:

86400 ; Refresh 24 hours (8hrs – for non-top level domains)

7200 ; Retry 2 hours (2hrs)

2592000 ; Expire 30 days (7 days)

345600 ; Minimum TTL 4 days (1 day)

What values you choose for your SOA record will depend upon the needs of your site. In general, longer times cause less load on your systems and lengthen the propagation of changes; shorter times increase the load on your systems and speed up the propagation of changes.

More on DNS

In the new version of bind, the TTL in SOA is now interpreted as the "**negative caching**" time (See RFC 2308). The default TTL value is defined by $TTL directive in the first line of your zone file. E.G.

```
$TTL 4d

@       IN      SOA  .....
```

Negative Caching

Classical DNS caching stores only the results of successful name resolutions. It is also possible for DNS servers to cache the results of ***unsuccessful*** name resolution attempts; this is called ***negative caching***.

To extend the example above, suppose you mistakenly thought the name of the company's web site was "***www.uccs.lk***" and typed that into your browser. Your local DNS server would be unable to resolve the name, and would mark that name as "**unresolvable**" in its cache; a negative cache entry. Note that "regular" caching is sometimes called "***positive caching***" to contrast it to "negative caching".

Negative Caching

The value to be used for negative caching in a zone is now specified by the ***Minimum*** field in the ***Start Of Authority*** resource record for each zone. As mentioned above, this was formerly used to specify the ***default TTL*** for a zone.

More on DNS

- Root DNS servers (totally 13) – a.root-servers.net, ….., m.root-servers.net
  a.root-servers.net root server is about 12,000 queries/sec

  Why 13 Root Servers?
- Primary Name Server
  - unique
  - has SOA (Source of Authority) of that domain
  - add/change/remove of the domain name records
- Secondary NS
  - possibly many for a domain
  - name records backup from Primary NS periodically
  - add/change/remove are worthless
  - fault tolerance when the Primary NS is down

More on DNS

- Cache Poisoning – an attacker obtains the ability to put data into our nameserver's cache
- Create a separate user for the DNS server, with shell equal to /bin/false.

  (named -u dns_user -g dns_group)
- BIND 8.2.0 and 8.2.1 was vulnerable to a Remote Root exploit! (current release BIND V 9....., check this by yourself)

More on DNS

- configuration syntax
  - /etc/named.boot (v4)
  - /etc/named.conf (v8,9)
- New Name Daemon Control program
  - ***ndc*** (v8), ***rndc*** (v9)
- Hostnames can contain letters, numbers, and hyphens, and may not start with a hyphen. Underscore "_" is not a valid character in a hostname.
- UDP/TCP port 53
  - UDP query/response (< 512 bytes)
    - On normal conditions, DNS UDP traffic occupies **more than 99%** of the total DNS traffic of a specified server!
  - TCP response (>512 bytes) + zone transfer

More on DNS

- Average size of a DNS packet is 150 bytes
- Diagnostic tool on the web: http://www.dnsreport.com
- The BIND name daemon control interface program (***ndc***) can provide version information when used with newer versions of BIND:

  # ndc status

DNS Example

```
options {
        directory  "/etc/namedb";
        pid-file   "/var/run/named/pid";
        version  "Unkonwn";
        listen-on          { a.b.c.d; };
        allow-transfer {"none";};
};
// UCSC Domains and their settings
zone "." in {
        type hint;
        file "named.root";
};
zone "cmb.ac.lk" {
      type master;
      file "master/cmb.ac.lk.db";
        allow-transfer { p.q.r.s; };
        also-notify { p.q.r.s; };
};
```

```
zone "cmb.ac.lk" {
      type slave;
      file "slave/cmb.ac.lk.db";
            masters {a.b.c.d;};
            notify no;
};
```

```
zone "248.192.IN-ADDR.ARPA" {
      type master;
      file "master/cmb.ac.lk-rev.db";
          allow-transfer { p.q.r.s; };
          also-notify { p.q.r.s; };
};
```

DNS Example

```
$TTL 3h
cmb.ac.lk.      IN      SOA     aduna.cmb.ac.lk. root.aduna.cmb.ac.lk. (
                        2011092702      ; serial
                        3h                      ; refresh every 3 hrs
                        1h                      ; rerty every hour
                        2w                      ; expire after 14 days
                        2d )                    ; 2 day
$ORIGIN cmb.ac.lk.
@               IN      TXT     "University of Colombo, Sri Lanka"
                IN      NS      aduna.cmb.ac.lk.
                IN      NS      ns.ac.lk.
                IN      A       10.20.50.112
                IN      MX      10      king.cmb.ac.lk.
                IN      MX      50      queen.cmb.ac.lk.
aduna           IN      A       10.20.50.234
```

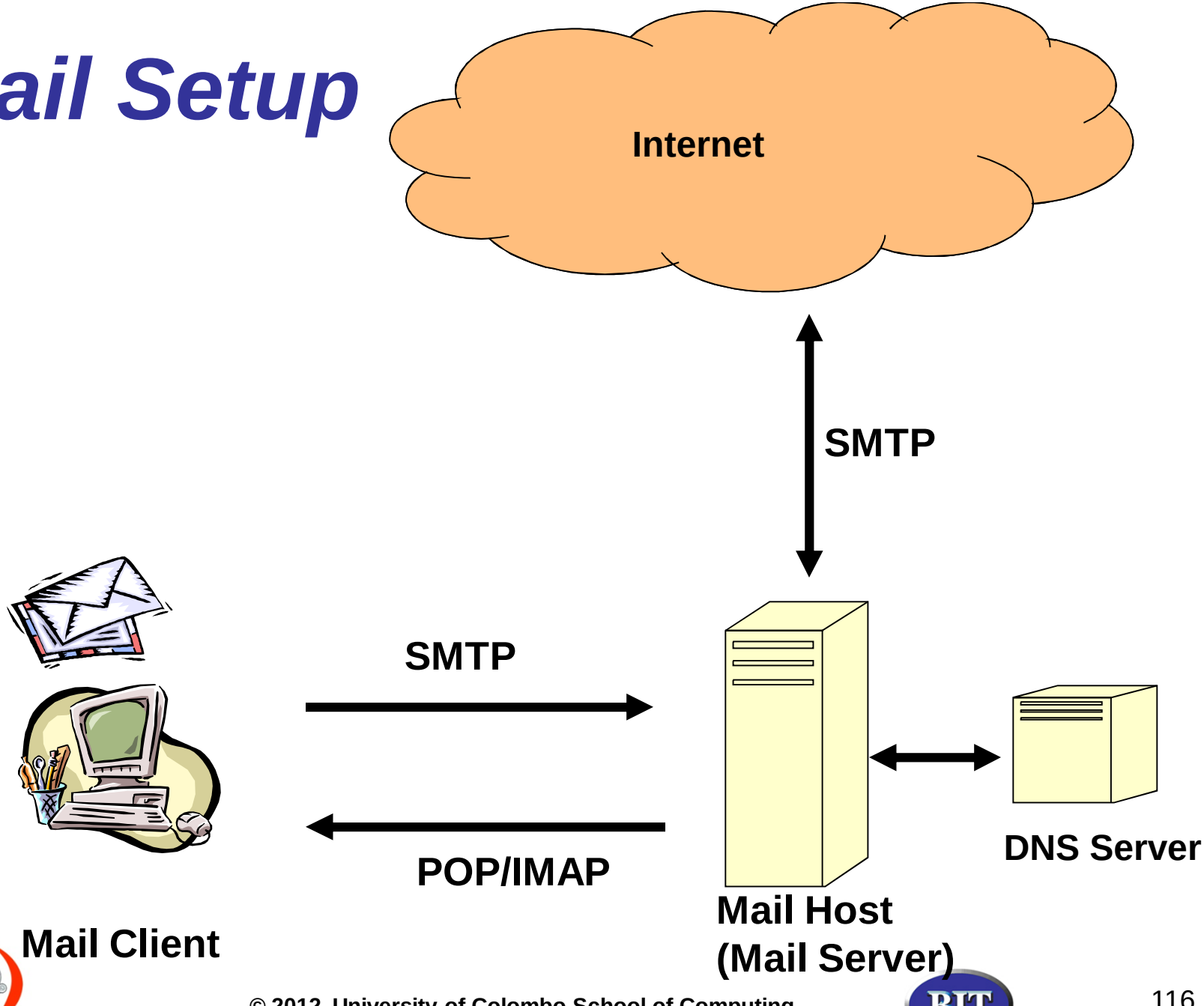
DNS Tools

Domain Information Groper (**dig**) is a network administration command-line tool for querying Domain Name System (DNS) name servers for any desired DNS records. **# dig A www.ucsc.lk**

```
; <<>> DiG 9.6.-ESV-R5 <<>> www.ucsc.lk
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 14345
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 2

;; QUESTION SECTION:
;www.ucsc.lk.                    IN      A

;; ANSWER SECTION:
www.ucsc.lk.            10800   IN      A       192.248.16.84

;; AUTHORITY SECTION:
ucsc.lk.                10800   IN      NS      aduna.cmb.ac.lk.
ucsc.lk.                10800   IN      NS      ns.ac.lk.
ucsc.lk.                10800   IN      NS      lms.bit.lk.

;; ADDITIONAL SECTION:
ns.ac.lk.               31910   IN      A       192.248.1.162
ns.ac.lk.               31910   IN      AAAA    2001:df0:17:1::162

;; Query time: 605 msec
;; SERVER: 71.252.219.43#53(71.252.219.43)
;; WHEN: Sat Oct 29 05:33:15 2011
;; MSG SIZE  rcvd: 155
```

4.5 Configuring Mail Server

A Mail Setup



Email Exchange

The major parts involved in an email exchange

- The user program (send/read mail)
- The server daemon (MTA)
- A daemon for users to read mail from mailhost (MUA)
- DNS

Email Exchange

Mail server daemons: **sendmail**, **qmail, postfix**, **exim**, **mmdf, smail**, **zmailer** etc.

The server daemon usually has 2 function:

- **looks after receiving incoming mail**
- **delivers outgoing mail**

The server daemon does not allow you to read your mail. For this you need an additional daemon (**POP**, **IMAP**, etc).

The DNS and its daemon "**named**" play a large role in the delivery of email.

What is Sendmail?

- Sendmail is widely used **Mail Transport Agent (MTA)** on the Internet
- MTAs send mail from one machine to another.
- Sendmail is **not** a client program, which you use to read your email.
- Sendmail is one of the behind-the-scenes programs which move email over the Internet.
  - Normally it runs as a background daemon

Sendmail Features

- Sendmail uses **DNS** (Domain Naming System)
- DNS provides Mail Exchange (MX) information
- Sendmail default is “mail relay off”
- Realtime Blackhole Lists (**RBL**)

Sendmail Anti-Spam Enhancements

- Mailscanner
  - Minimal anti-spam
  - Anti-virus integration (scan in/outbound)
- Spam Assassin
  - Rule based heuristic
  - Header and text analysis
  - Blacklist (RBL)

Sendmail related config files

- Staring the daemon: /etc/init.d/sendmail [start, restart, stop]
- User mail-boxes: /var/spool/mail/username
  /var/mail/username
- Outgoing email queued: /var/spool/mqueue

Sendmail related config files

/etc/mail/access	➔ specifies which systems can use sendmail for relaying email
/etc/mail/aliases	➔ mailbox aliases
/etc/mail/local-host-names	➔ list of hosts sendmail accepts mail for
/etc/mail/mailertable	➔ specifies instructions that overrides routing for particular domains
/etc/mail/virtusertabl	➔ permits to do a domain-specific form of aliasing, allowing multiple virtual domains to be hosted on one m/c
/etc/mail/domaintable	➔ allows to provide domain name mapping
/etc/mail/sendmail.cf	➔ sendmail master configuration file

/etc/mail/access

What hosts or IP addresses have access to the local mail server and what kind of access (OK, REJECT, RELAY) they have.

OK – allowed to send mail to the host as long as mail's final destination is the local machine

REJECT – reject for all mail connections

RELAY – allowed to send mail for any destination thro. this server

ERROR - Reject message with a specified error message

DISCARD - Silently ignore message from this source

e.g.

a.source.of.spam	REJECT
Okay.cyberspammer.com	OK
128.32	RELAY
evil-ones.net	ERROR:"550 No spam accepted"

The above third entry allows mail from hosts with an IP address that begins with 128.32.*.* would be able to send mail through this mail server.

/etc/mail/aliases

This database contains a list of virtual mailboxes that are expanded to other user(s), files, programs or other aliases.

e.g. root: ajantha
webmaster: kamal, lal
fool: /dev/null
staff: ajantha, saman, lal, staff-archive
staff-archive: /var/atchives/staff
info: |/etc/mail/infobot
customers: :include: /etc/mail/lists/customer-list

The last aliase causes how to keep a list of users for an alises in an external file. This may be a good mechanism to allow unprivileged users to maintain their own mailing lists (but Majordomo/Mailman, etc. – a good solution!!)
Once you have updated this file you need to run the ***newaliases*** program (or run ***make*** in /etc/mail) to update the database

/etc/mail/ other conf files

/etc/mail/local-host-names

List of hostnames sendmail is to accept as the local mail host. For example, if the mail server was to accept mail for the domains cmb.ac.lk and ucsc.cmb.ac.lk the above file contain:

cmb.ac.lk

ucsc.cmb.ac.lk

/etc/mail/sendmail.cf

Controls the overall behavior of sendmail. This configuration file is quite complex. The master sendmail configuration file can be built from ***m4*** macros.

Communicating with the Mail Server

```
telnet smtp-server.domain.com 25
```

If your server is online a connection will be established on port 25 (SMTP).

An Exchange Server answers with the following output:

220 mailserver.domain.com Microsoft ESMTP MAIL Service, Version: 5.0.2195.5329 ready at Sat, 22 May 2006 08:34:14 +0200

When you type the 'help' command the available commands are listed:

214-This server supports the following commands:
214 HELO EHLO STARTTLS RCPT DATA RSET MAIL QUIT HELP AUTH TURN ATRN ETRN BDAT VRFY

Try the following to send an eMail from the command line:

220 mailserver.domain.com Microsoft ESMTP MAIL Service, Version: 5.0.2195.5329 ready at Sat, 22 May 2006 09:01:29 +0200

```
helo myserver.domain.com
```

250 mailserver.domain.com Hello [10.1.11.133]

```
mail from:<myname@mydomain.com>
```

250 2.1.0 myname@mydomain.com....Sender OK

```
rcpt to:<recipientname@mydomain.com>
```

250 2.1.5 recipientname@mydomain.com

```
data
```

354 Start mail input; end with <CRLF>.<CRLF>

```
subject: This is a test mail
to: recipientname@mydomain.com
This is the text of my test mail.
.
```

250 2.6.0 <exchange.domain.com> Queued mail for delivery

```
quit
```

UCSC

Communicating with the Mail Server

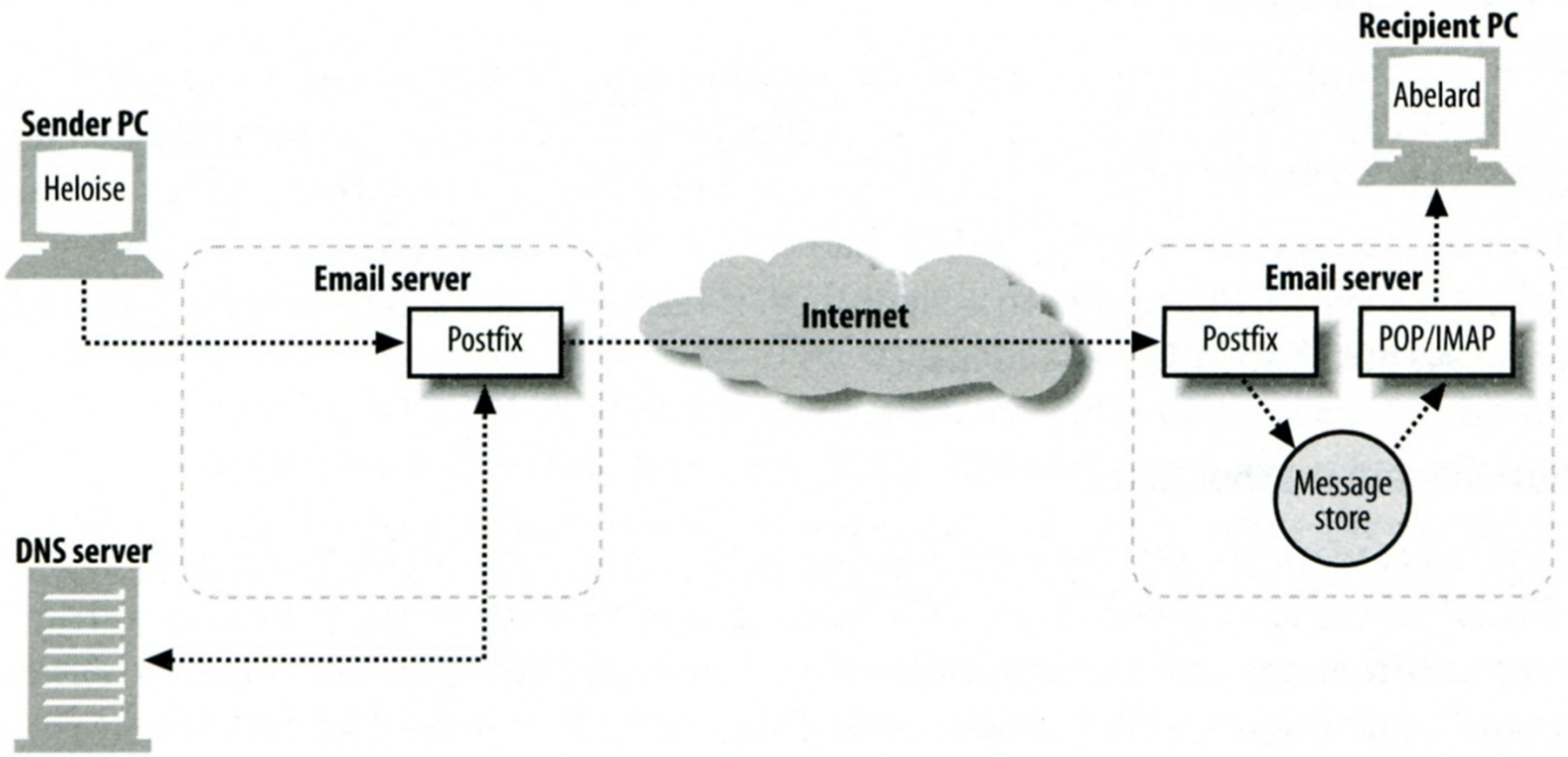
SMTP Commands:	
HELO *sendinghostname*	This command initiates the SMTP conversation. The host connecting to the remote SMTP server identifies itself by it's fully qualified DNS host name.
EHLO *sendinghostname*	An alternative command for starting the conversation. This states that the sending server wants to use the extended SMTP (ESMTP) protocol.
MAIL From:<*source email address*>	This is the start of an email message. The source email address is what will appear in the "From:" field of the message.
RCPT To:<destination email address>	This identifies the receipient of the email message. This command can be repeated multiple times for a given message in order to deliver a single message to multiple receipients.
SIZE=*numberofbytes*	The size command tells the remote sendmail system the size of the attached message in bytes. If ommited, mail readers and delivery agents will try to determine the size of a message based on indicators such as them being terminated by a "." on a line by themselves and headers being sent on a line separated from body text by a blank line. But these methods get confused when you have headers or header like information embedded in messages, attachements, etc.
DATA	This command signifies that a stream of data, ie the email message body, will follow. The stream of data is terminated by a "." on a line by itself.
QUIT	This terminates an SMTP connection. Multiple email messages can be transfered during a single TCP/IP connection. This allows for more efficient transfer of email. To start another email message in the same session, simply issue another "MAIL" command.
VRFY *username*	This command will request that the receiving SMTP server verify that a given email username is valid. The SMTP server will reply with the login name of the user. This feature can be turned off in sendmail because allowing it can be a security hole. VRFY commands can be used to probe for login names on a system. See the security section below for information about turning off this feature.
EXPN *aliasname*	EXPN is similar to VRFY, except that when used with a distribution list, it will list all users on that list. This can be a bigger problem than the "VRFY" command since sites often have an alias such as "all".
Subject: Cc: Reply-To:	Email header lines are not SMTP commands per se. They are sent in the DATA stream for a message. Header lines appear on a line by themselves, and are seperated from the body of a message by a blank line.

Postfix

- Free and open source mail transfer agent (MTA)
  - For the routing and delivery of email
  - Intended as a fast, easy-to-administer, and secure alternative to the widely-used Sendmail
  - Formerly VMailer / IBM Secure Mailer
    - By Wietse Venema at the IBM Thomas J. Watson Research Center
  - IBM Public License
- First released in mid-1999
- http://www.postfix.org
  - http://www.postfix.org/documentation.html

Role of Postfix

- MTA that
  - Receive and deliver email over the network via SMTP
  - Local delivery directly or use other mail delivery agent



Postfix Architecture

- Modular-design MTA
  - Not like sendmail of monolithic system
  - Decompose into several individual program that each one handle specific task
  - The most important daemon: **master daemon**

    Reside in memory

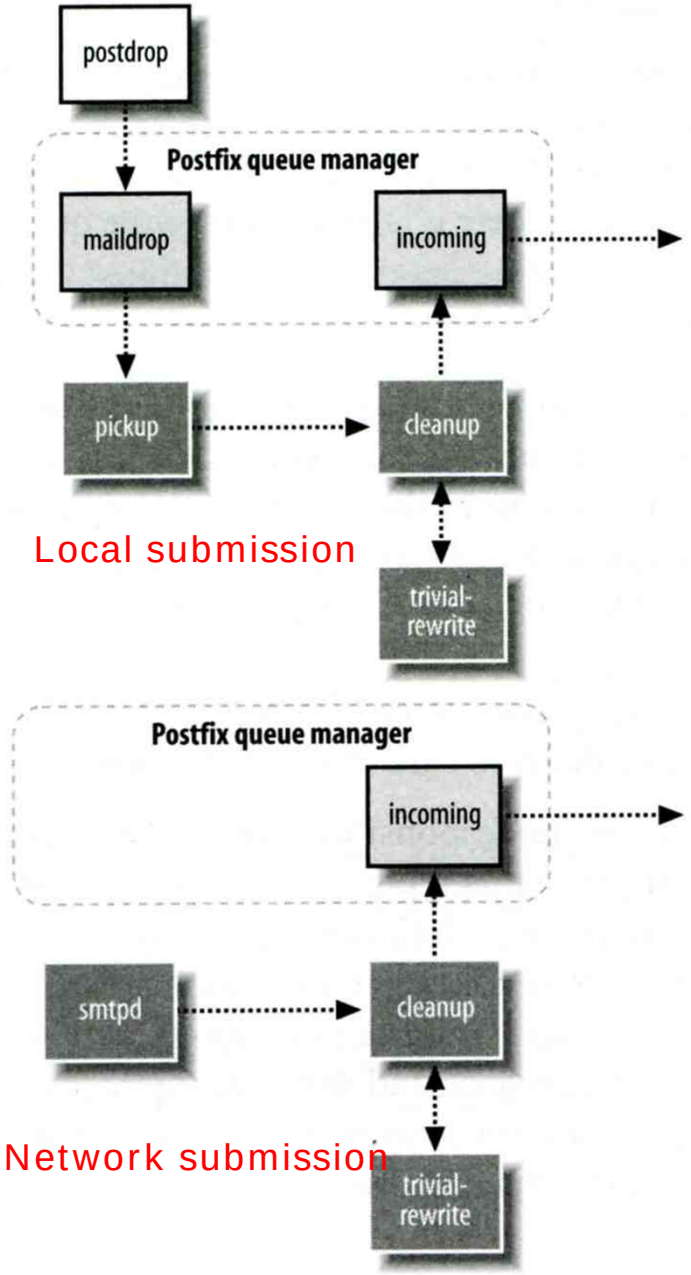
    Get configuration information from master.cf and main.cf

    Invoke other process to do jobs
- Major tasks
  - Receive mail and put in queue
  - Queue management
  - Delivery mail from queue

Postfix Architecture

Postfix Architecture – Message IN

- Four ways
  - **Local submission**
    - postdrop command
    - maildrop directory
    - pickup daemon
    - cleanup daemon
      - Header validation
      - address translation
    - incoming directory
  - **Network submission**
    - smtpd daemon
  - **Local forwarding**
    - Resubmit for such as .forward
  - **Notification**
    - defer daemon
    - bounce daemon



Postfix Architecture – Queue

- Five different queues
  - **incoming**

    The first queue that every incoming email will stay
  - **active**

    Queue manager will move message into active queue whenever there is enough system resources

    Queue manager then invokes suitable DA to delivery it
  - **deferred**

    Messages that cannot be delivered are moved here

    These messages are sent back either with bounce or defer daemons

Postfix Architecture – Queue

- **corrupt**

  Used to store damaged or unreadable message

- **hold**

  Define "smtpd" access(5) policies, or cleanup(8) header/body checks to automatically place messages in the "hold" queue

  Messages placed in the "hold" queue stay there until the administrator intervenes

Postfix Architecture – Message OUT

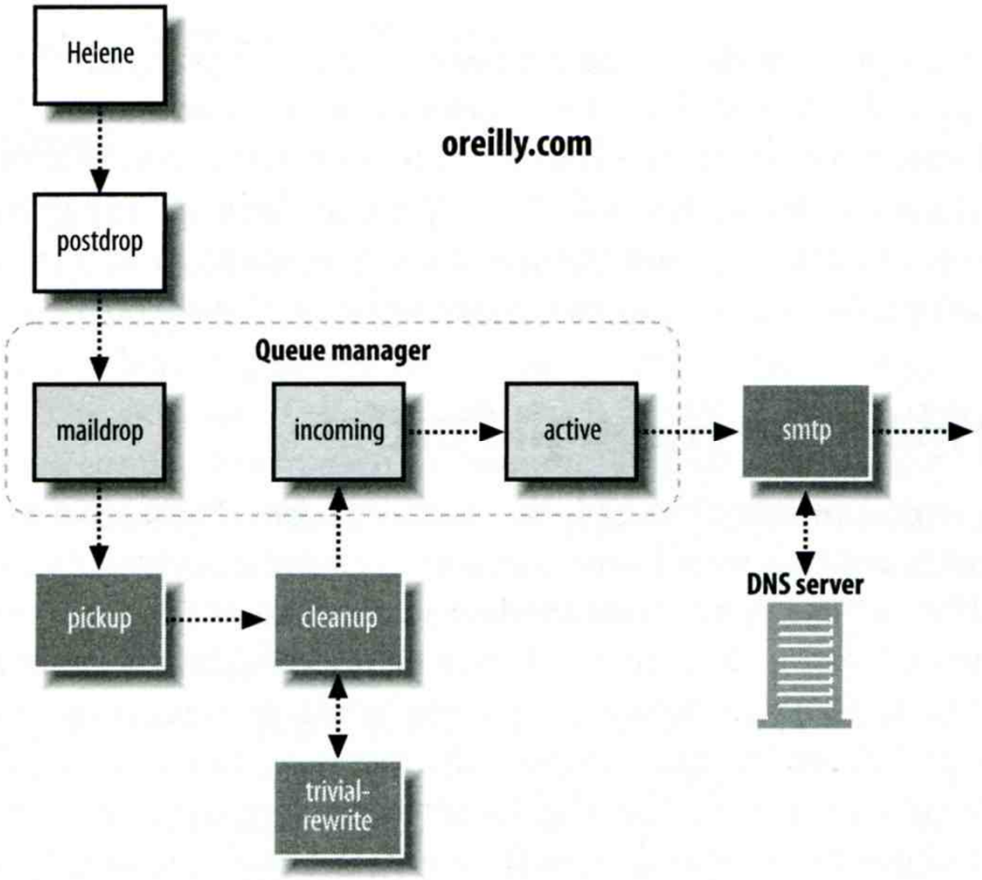
- Address classes
  - Used to determine which destinations to accept for delivery
  - How the delivery take place
- Main address classes
  - Local delivery
    - Domain names in “mydestination” is local delivered
    - Ex: mydestination = ucsc.cmb.ac.lk localhost
    - It will check alias and .forward file to do further delivery
  - Virtual alias

    Ex: virtual-alias.domain

    user1@virtual-alias.domain address1

Message Flow in Postfix (1)

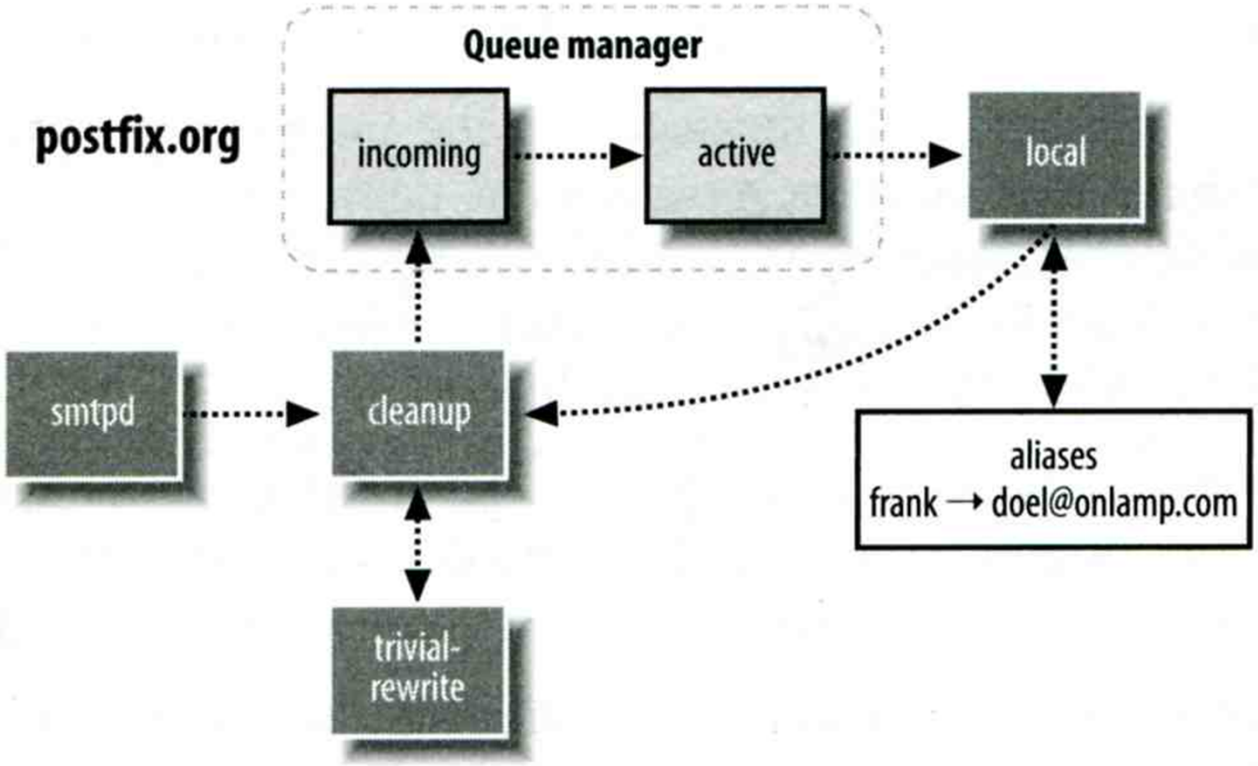
- Example
  - helene@oreilly.com → frank@postfix.org (doel@onlamp.com)
  - Phase1:
    - Helene compose mail using her MUA, and then call postfix's sendmail command to send it



Message Flow in Postfix (2)

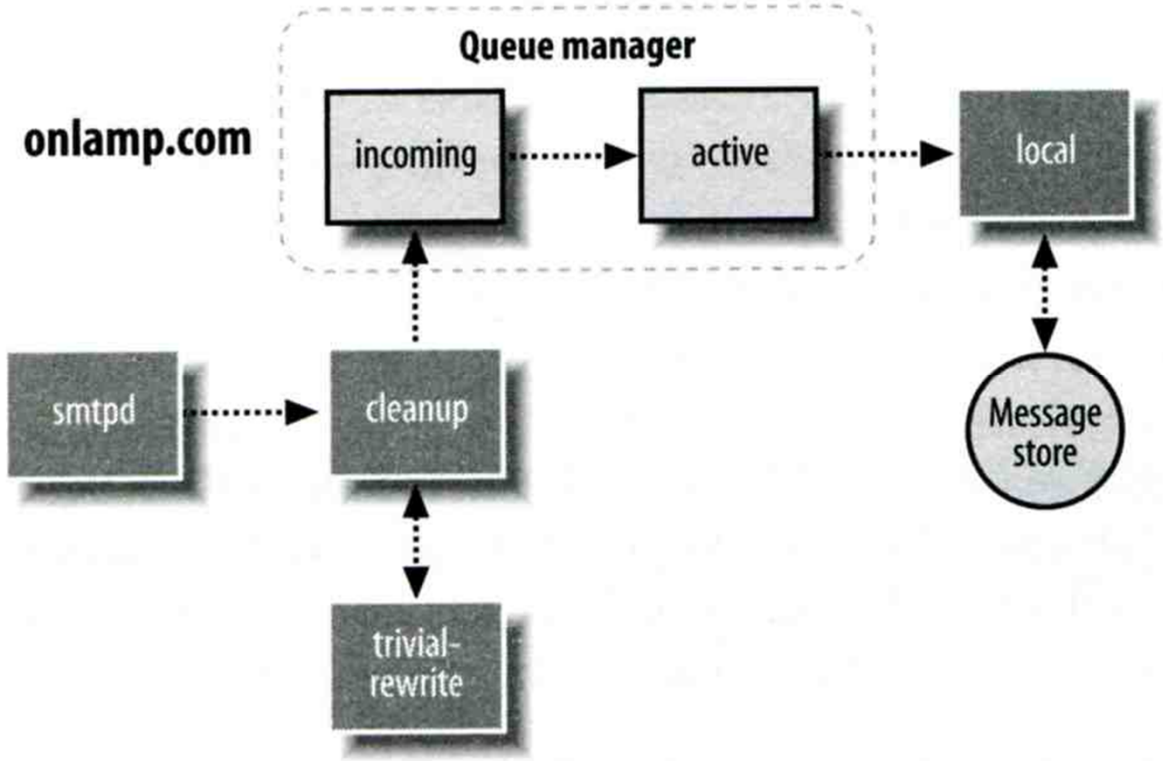
– Phase2:

- The smtpd on postfix.org takes this message and invoke cleanup then put in incoming queue
- The local DA find that frank is an alias, so it resubmits it through cleanup daemon for further delivery



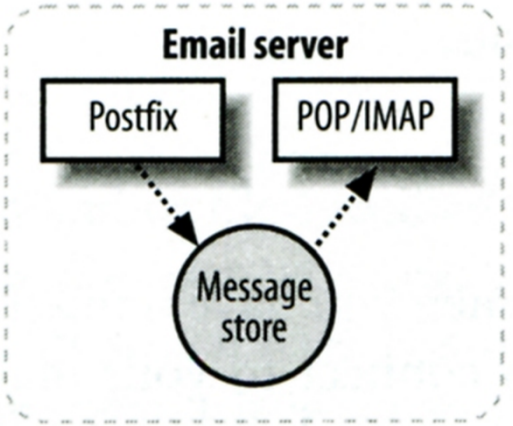
Message Flow in Postfix (3)

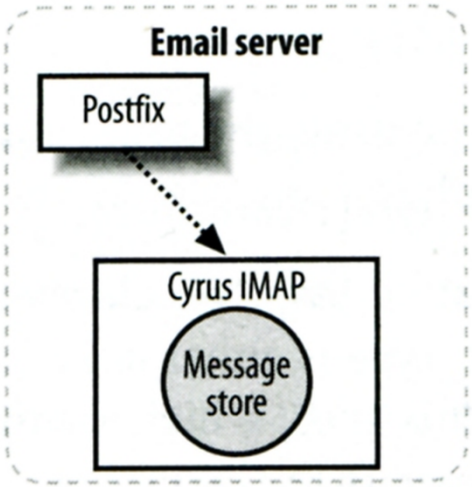
- Phase3
  - The smtpd on onlamp.com takes this message and invoke cleanup then put in incoming queue
  - Local delivery to message store



Postfix and POP/IMAP

- POP vs. IMAP
  - Both are used to retrieve mail from server for remote clients
  - POP has to download entire message while IMAP can download headers only
  - POP can download only single mailbox, while IMAP can let you maintain multiple mailboxes and folders on server
- Cooperation between Postfix and POP/IMAP
  - Postfix and POP/IMAP must agree on the type of mailbox format and style of locking
    - Standard message store
    - Unstandard message store (using LMTP)
      - Such as Cyrus IMAP / Dovecot

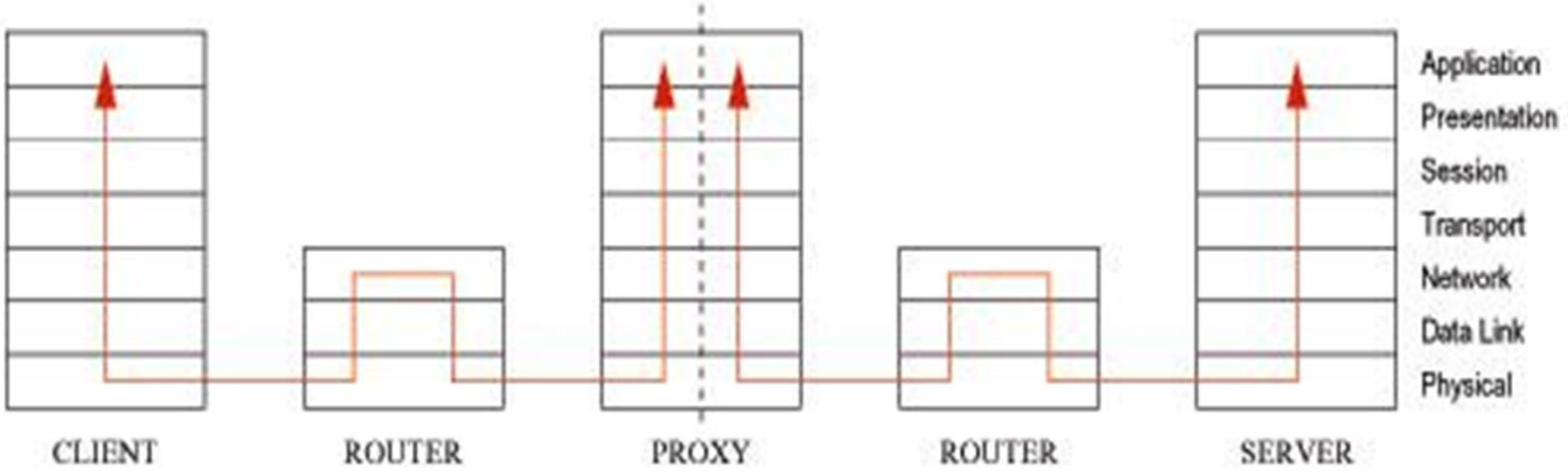




4.6 Configuring Squid Server

Proxy Cache Design: ISO

- A Web proxy is an application-layer gateway.
- Client requests go to the proxy, instead of the origin server.
- Origin serves receive connections from the proxy IP address.



Proxy Cache Design: Cachable Responses

- Some server responses may be stored on disk, and re-used for later requests.

- HTTP defines rules for determining cachability.
  - Request method.
  - Request authentication.
  - Response status code.
  - Response cache-control headers.

Proxy Cache Design: Cachable Responses

- Local policies may also affect cachability.
- Based on origin server hostname.
- Based on server IP address.
- Based on reply size.
- etc.

Proxy Cache Design: Hits and Misses

- A cache miss occurs when the proxy does not have a valid response saved.
- A cache hit occurs when the proxy has a valid (fresh) response saved.
- Hit ratio is the percentage of client requests that are satisfied with a cache hit.
- Typical hit ratios are in the range of 30-50%

Proxy Cache Design: Validation

- Cached responses are either fresh or stale.
- Whether a particular response is fresh or stale depends on the response headers, and the local configuration.
- Fresh responses may be given as hits immediately.
- Stale responses should be validated with the origin server.
- A cache validator is used to identify a particular instance of a resource. Usually this is a timestamp.

HTTP headers are sent by the server before the HTML, and only seen by the browser and any intermediate caches. Typical HTTP 1.1 response headers might look like this:

```
HTTP/1.1 200 OK
Date: Fri, 30 Oct 1998 13:19:41 GMT
Server: Apache/1.3.3 (Unix)
Cache-Control: max-age=3600, must-revalidate
Expires: Fri, 30 Oct 1998 14:19:41 GMT
Last-Modified: Mon, 29 Jun 1998 02:28:12 GMT
ETag: "3e86-410-3596fbbc"
Content-Length: 1040
Content-Type: text/html
```

HTTP 1.1 introduces a new class of headers, the Cache-Control response headers, which allow Web publishers to define how pages should be handled by caches. They include directives to declare what should be cacheable, what may be stored by caches, modifications of the expiration mechanism, and revalidation and reload controls. Interesting Cache-Control response headers include:
max-age, s-maxage, no-cache, must-revalidate , proxy-revalidate

```
Cache-Control: max-age=3600, must-revalidate
```

Installing Squid: Source Code

- Get the most recent, stable release from `http://www.squid-cache.org/` or a mirror site.
- Extract source tree

```
% tar xzvf squid-version.tar.gz
% cd squid-version
```

Installing Squid: Compiling

- Select a Squid installation directory (aka “prefix"). The default is /usr/local/squid .
- Run the configure script:

```
% ./configure --prefix=/usr/local/squid
```

If the configure output looks okay, run make :

```
% make
```

Install the programs:

```
% su
# mkdir /usr/local/squid
# make install
```

Configuration: Userid

- Create a new user and group for the Squid process and files.
- This user must be able to write to Squid's log files and directories (`/usr/local/squid/logs`)
- This user must be able to read the files in the “etc" directory (`/usr/local/squid/etc`)
- This user must be able to execute the programs in the “bin“ directory (`/usr/local/squid/bin`)
- Set the userid in squid.conf

```
cache_effective_user squid
cache_effective_group squid
```

Configuration: Cache Directories

- Create a cache directory on each cache disk

```
# mkdir /disk1/cache
```

- Make sure the Squid user owns the cache directories.

```
# chown squid:squid /disk1/cache
```

Configuration: Cache Directories cont…

- Determine the amount of disk space that Squid should use.

- Performance degrades as the disk becomes full.

```
cache_dir /disk1/cache 10240 16 128
```

This will create a 10 G storage directory in /disk1/cache

Configuration: Port Numbers

- Select which port Squid will use for HTTP requests.
- Squid's default is port 3128.
- Recommend that you do not use port 80.
- Set http port in squid.conf
- `http_port 3128`

Configuration: Access Controls

- Squid's access control system uses ACL elements, and access lists.
- An ACL element consists of a type and a number of values.
- An access list consists of either allow or deny followed by a number of ACL elements.
- OR logic is used within a single ACL element. An ACL element is “matched" if at least one of the values matches.
- AND logic is used within an access list. An access list is matched only when all of its ACL elements match.
- Access lists are evaluated in the order they are written.

Configuration: Access Controls

- Squid's default configuration file denies all requests.
- You must allow client IP addresses (networks) to use your cache.

```
acl mynet src
10.16.32.0/255.255.240.0
http_access allow mynet
```

- Insert these before the final `deny` rule.
- You may also want to change the `Safe_ports acl`.

Configuration: Access Controls

- Configure Squid Not to cache a specific server

```
acl scores dstdomain  .cricinfo.com
no_cache deny scores
```

- Implement an ACL ban list

```
acl Cooking1 url_regex cooking
http_access deny Cooking1
http_access allow all
```

The `url_regex` means to search the entire URL for the regular expression you specify. Note that these regular expressions are case-sensitive. So a url containing “Cooking” not be denied.

Configuration: Access Controls

- Allow some clients to use the cache at specific times

```
acl STUD src 10.16.68.1 10.16.68.40
acl WORKING time MTWHF 08:30-17:30
http_access allow STUD WORKING
http_access deny ALL
```

This will allow access to the Internet during working hours (8:30 - 17:30) for the students having IP address range (10.16.68.1 to 10.16.68.40).

Configuration: Access Controls

- Blocking specific URLs

e.g:

```
acl porn url_regex
                "/usr/local/squid/etc/porno.txt“
http_access deny porn
```

- Custom error message

you want your users to see a special message when they request something that matches your pornography list. First, create a HTML file named ERR_NO_PORNO in the */usr/local/squid/etc/errors* directory. Then add this line before the “http_access” line

```
deny_info ERR_NO_PORNO porn
```

Configuration: Syntax Checking

- Before you run Squid, it is always a good idea to check your configuration file for syntax errors.
- Just run

```
% squid -k parse
```

If there is no output, your configuration file is fine.

Otherwise, Squid reports the line number where an error is found.

Running Squid: First Time

- Whenever you add a new cache directory, you must initialize it.
- This procedure creates the two-level directory tree where cache files are stored.
- Simply run

```
% squid -z
```

Running Squid: Verifying it Works

- Simply type:
  `/usr/local/squid/bin/squid`
- Check syslog messages file:

... squid[204]: Squid Parent: child process 219 started

Check process list

```
BSD: % ps ax | grep squid
Linux: % ps -ef | grep squid
```

Running Squid: Verifying it Works

- Examine `/usr/local/squid/logs/cache.log`

  `1999/10/20 23:23:30| Ready to serve requests.`

- telnet to the HTTP port

```
% telnet localhost 3128
```

Hierarchies: Parents

- A parent cache should be “upstream" along your path to the Internet.
- If your ISP operates a cache, then you might be able to use it as a parent.
- By default, Squid does not send requests for uncachable replies to a parent cache.

Hierarchies: Siblings

- A sibling cache should be “nearby" but not necessarily “up-stream."
- You can request only cache hits from a sibling cache.
- This requires ICP or Cache Digests.
- Your ISP or other customers may have a cache that you can use as a sibling.

Hierarchies: Squid configuration

- Usage:

```
cache_peer hostname type http-port icp-port [options]
```

Examples:

```
cache_peer cache.foo.org parent 3128 3130
cache_peer cache.foo.org sibling 8080 3130
cache_peer cache.foo.org parent 3128 0 no-query default
```

- See squid.conf for additional cache peer options.

Hierarchies: Issues

- Be careful when crossing organizational boundaries.
- Your users will probably get error messages from the neighbor caches.
- ICP adds delays to your cache misses.

Monitoring: Watching Squid

- You should develop scripts and procedures for monitoring Squid's operation.
- If problems arise, performance will suffer, or your users will not get any pages.
- There are many places to look for indications of problems.
  - Squid's log files
  - Syslog
  - The cache manager facilities

Monitoring: cache.log

- This is the primary place to look for warnings and debugging messages.
- Search this file for “WARNING".
- If Squid restarts, search for “FATAL" or “assertion failed".
- The cache.log messages are also sent to syslog with logging facility LOCAL4 , and a priority of either LOG WARNING or LOG NOTICE

Monitoring: access.log

- This log records every client request, after the response has been sent.
- The second field is the elapsed time, in milliseconds.
- The fourth field indicates cache hits and misses.
- The fifth field is the number of bytes written to the client.
- The ninth field indicates where cache misses are forwarded to.

Monitoring: Cache Manager

- The Cache Manager allows you to find out many things about Squid's operation.
- It can be used from the command line, or as a CGI script.
- The cachemgr.cgi program converts the text output into HTML.
- From the command line, you can use the client program:

```
client mgr:info
```

Monitoring: Cache Manager

- The “info" page tells you many things:
  - Hit ratios
  - cache size
  - Median service times
  - Memory usage
  - Filedescriptor usage
  - CPU usage

End of Section 4.0