

Unix System Administration

Chris Schenk

Lecture 12 – Thursday Feb 20

CSCI 4113, Spring 2008

Extra Credit

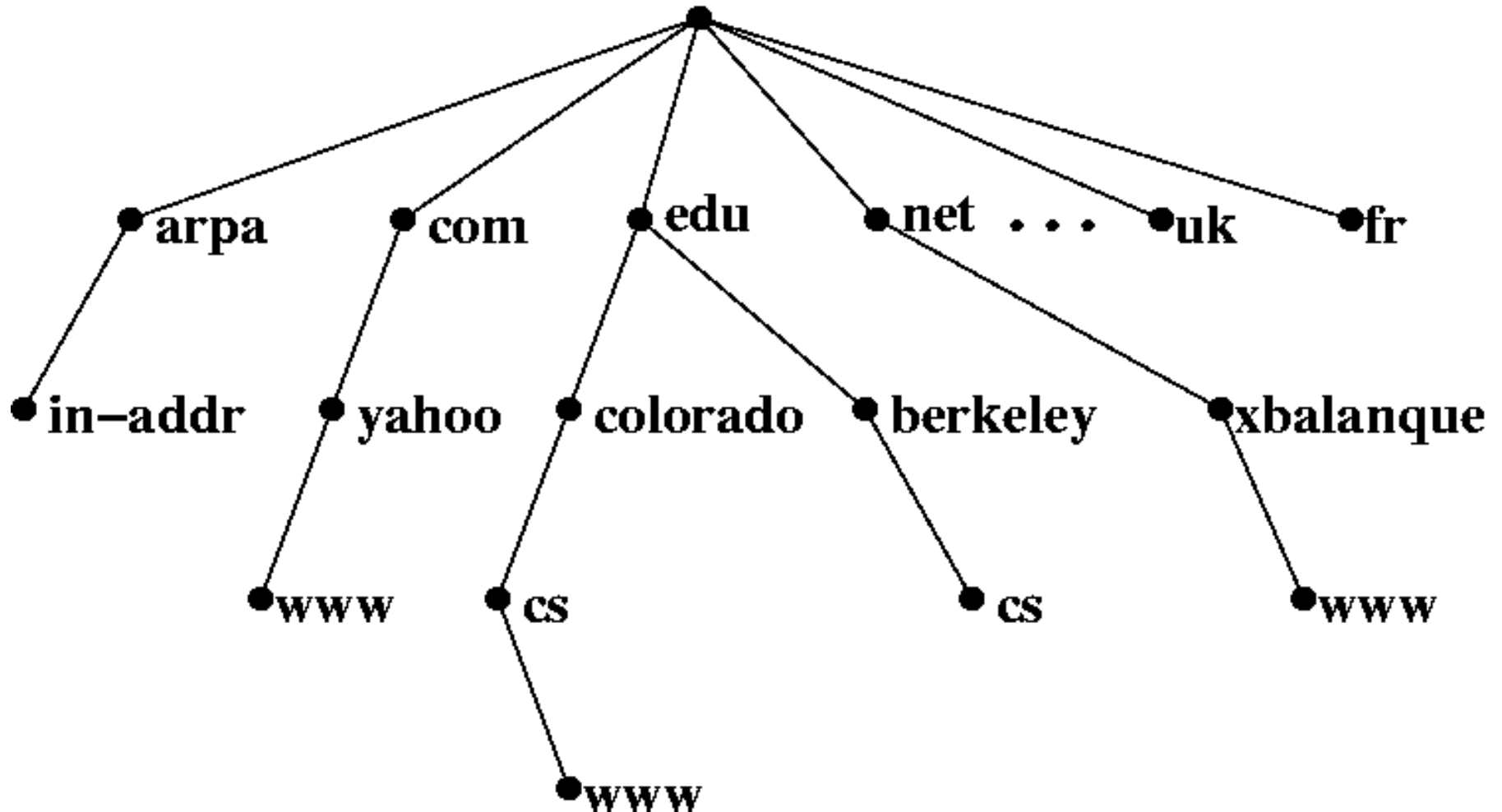
- Project proposal of your choice
 - Things that you have always wanted to do
 - Or dig in deeper to things we've covered
- Things you can't do projects on
 - Anything we've already done or things we will cover
 - I will let you know if an idea is valid or not
- A good project will be worth one lab, an 'ok' one will be worth half a lab
- EC won't count if you haven't completed other labs. Get them working!

IPTables – Limit Match

- Completely non-intuitive
 - Intuitive would be: Drop X packets that match rule R per Y time unit
 - IPTables is not this.
- `-m limit --limit 5/s --limit-burst 10`
 - Limit-burst token bucket set to 10
 - Each packet matched depletes one token from bucket
 - The `5/s` means restore 1 token to the bucket every $1/5$ second
 - Once token bucket is empty, rules no longer match!

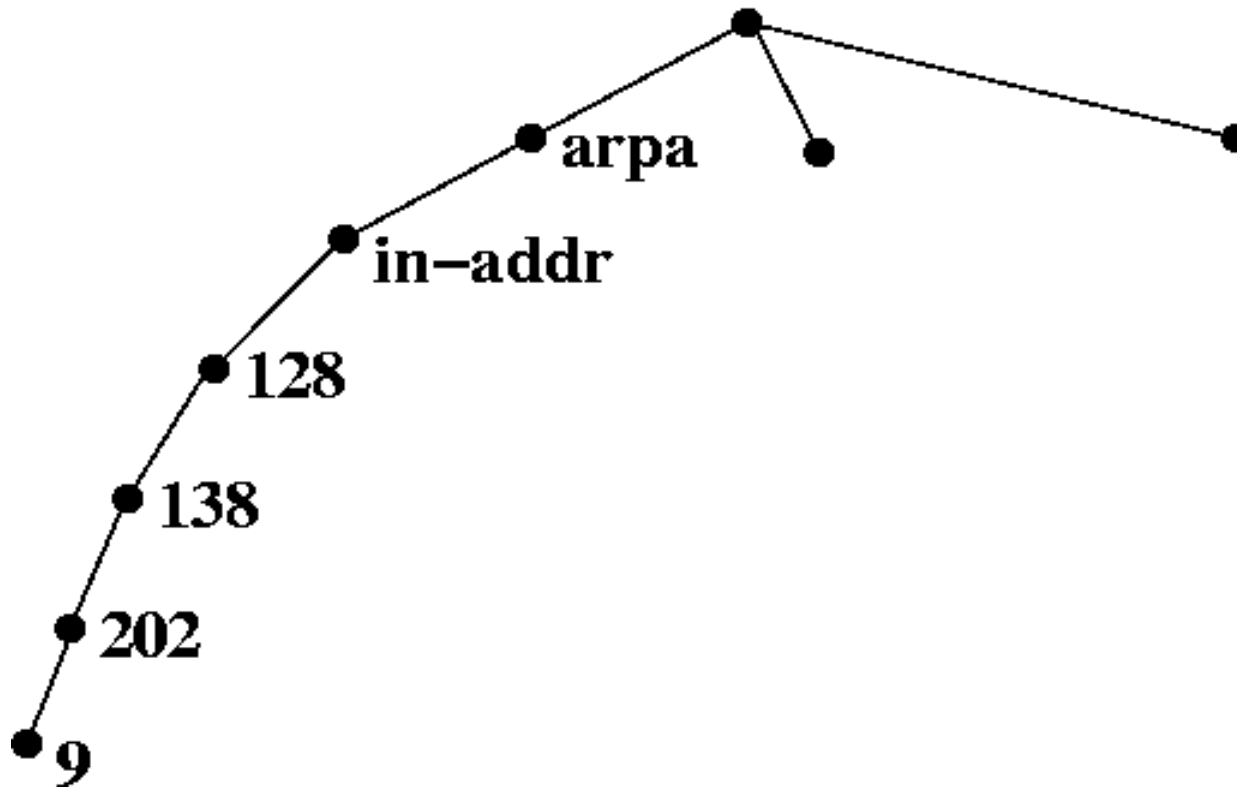
DNS – Domain Name Space

- Distributed database framework



DNS – in-addr.arpa

- This subtree is used for reverse lookups



DNS – Zones and Delegation

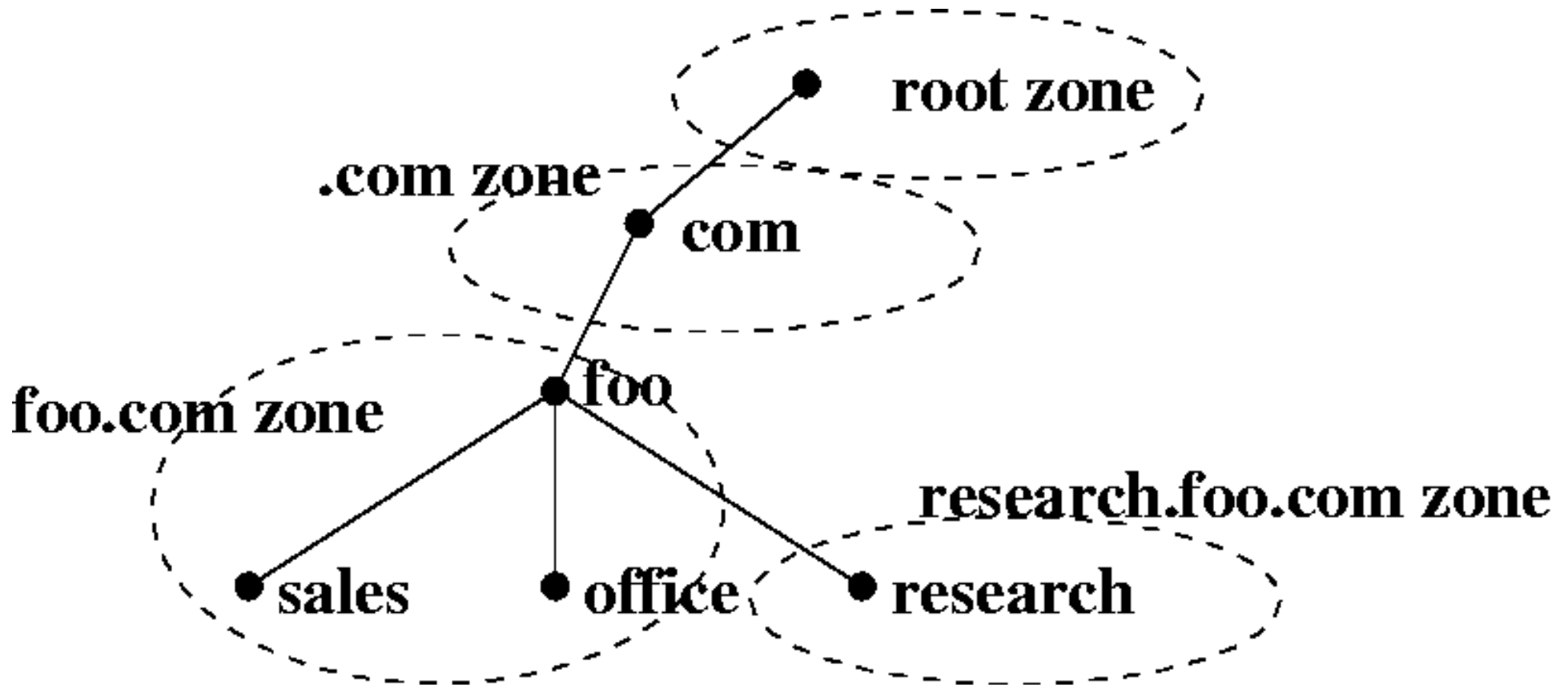
- Authority for a zone is delegated by a parent zone
 - The **root** zone delegates **edu** authority to people who manage the **edu** zone
 - The **edu** zone delegates **colorado.edu** authority to the University of Colorado
 - CU then delegates the **cs.colorado.edu** zone to Computer Science
- Delegation of authority means that some parent nameserver has data records that associate a given child zone with some **specific** nameservers

DNS – Zones and Subdomains

- A zone can include its subdomains
 - child zones don't have to be delegated out
- Subdomains are in the parent zone if no **glue** records exist to delegate authority
- A zone is a point of delegation in the DNS tree
 - Don't get this confused!
 - A zone consists of these contiguous parts for which a name server has complete information authority
- Delegation points are marked by one or more NS records

DNS – Zone Delegation Example

- foo.com has multiple domains available
- research.foo.com likes to do their own thing



DNS – Zone Delegation (cont)

- Zone authority usually includes at least two zones
 - Forward and Reverse zones together
- Usually each separate subnet has its own reverse zone
 - This is done even though there is a single forward zone
 - cs.colorado.edu has one forward zone but many reverse zones
- The term 'zone' often refers to both forward and reverse
 - Hardly ever is a forward zone delegated without the reverse or vice versa, but it can happen!

DNS – Queries Revisited

- DNS clients contact nameservers for host resolution
- Local DNS servers may be authoritative for local names and addresses
 - cs.colorado.edu domain is managed locally
- All other queries must be re-transmitted to nameservers that are authoritative for the information requested
 - Exactly how this process works is at the heart of DNS as a robust, distributed database

DNS – Query Resolution Example

- Here's how most standard queries are resolved:
 - I point my browser to www.cs.berkeley.edu
 - A recursive query is sent to my local nameserver (1st in resolv.conf)
 - What is IP address for www.cs.berkeley.edu?
 - If the server doesn't immediately know in the cache, query is sent to the ROOT nameserver
 - Referral received to talk to an EDU server
 - Local server sends query to EDU server
 - Receives referral to berkeley.edu server

DNS – Query Resolution Example (cont)

- Local server queries berkeley.edu server
 - Receives referral to cs.berkeley.edu server
- Finally the query is sent to cs.berkeley.edu
 - And the IP address of www.cs.berkeley.edu is returned
- The local server then sends the response back to the client
 - Client uses IP address to talk to 169.229.60.205 directly

DNS – Resource Records

- There are dozens of resource records
 - Only a few are used to any great extent
- Two records are categorized as 'zone' records
 - SOA “Start of Authority” record – only ONE per zone
 - Indicates the beginning of a zone and should occur first in file
 - Defines values for serial number and expiration times
 - NS “Nameserver” record
 - Defines an authoritative nameserver for a zone
 - Delegates authority to a nameserver for a child zone
 - NS records are the GLUE that binds the distributed database together

DNS – Host Records

- Records that deal mainly with host information
 - A Address record
 - Name->IP address **forward** record
 - PTR Pointer record
 - IP->Name **reverse** record
 - CNAME Canonical Name record
 - Aliases for A records
 - MX Mail eXchange record
 - Specifies a host that will accept mail on behalf of another host

DNS – MX Records

- A single host may have multiple MX records
 - Records for a host make up a priority list
 - The LOWEST number is HIGHEST priority
 - If delivery cannot be made to highest priority, the next highest is attempted
- The idea is highest priority is final destination for email
 - MX backup hosts hold mail to attempt delivery to final host
- MX records used to be optional
 - Many mail configurations won't deliver mail to a host w/out MX

DNS – Security Records

- Records designed to guarantee authenticity of records
 - KEY Public key record
 - Associates a public key with a DNS name
 - Same public/private keypair idea
 - SIG Signature record
 - Cryptographically sign the resource record
 - NXT Next Record
 - Bit misleading
 - Used to authoritatively deny the existence of a record
 - i.e. “host xyz doesn't exist in zone foo”

DNS – Nameservers

- Daemon process that answers DNS queries
 - Come in three basic types: Primary, Secondary, and Caching
- Usually there is one primary server for a zone
 - aka Primary or Master server
 - Primary server loads zone information from files on disk
 - The term 'zone' refers to both forward and reverse zones
- Most sites have one or more secondary servers
 - aka Secondary or Slave server

DNS – Nameservers (cont)

- Secondary servers load zone information from the master
 - All data is transferred at one time via a “zone transfer”
 - Newer implementations of DNS support incremental transfers
- Secondary servers get new data by comparing serial #s
 - Serial numbers found in the SOA record
 - If serial number on slave is less than serial number on master
 - then transfer all new data to slave from master

DNS – Nameservers (cont)

- Third type is the Caching nameserver
 - Caching servers don't serve data for any specific zone
 - Server merely remembers answers for queries it has made
 - Increases network efficiency because repeated queries don't need to be re-sent
 - Cached answer can be used instead
- One nameserver CAN perform all three roles
 - Server can be primary for some zones, secondary for others
 - Can also cache answers for all other queries outside of above zones

DNS – Nameservers (cont)

- Final category of name server is the Stealth server
 - By definition, stealth servers don't have NS records pointing to them
 - NOT compliant with DNS RFC specifications
 - You have to know the server's IP address to use it
 - Stealth servers can be authoritative for a zone
 - Very beneficial for running on a network to increase performance
 - Stealth servers are very common in NAT'ed environments
 - CSEL internal network uses a stealth server

DNS – Zones

- Every zone hosted by an authoritative server
 - Both Primary and Secondary servers authoritative
- Having more than one Primary server is bad
 - Have to manually synchronize data in the zone between servers
- Should have at least one Secondary server for the zone
 - Required by the RFCs to run an official domain
- Frequently the Secondary server will be offsite
 - Often two sites will host Secondaries for each other
 - I'll scratch your back, you scratch mine

DNS – Parent Zones

- Parent zones must delegate to all authoritative servers
 - Primary and Secondary nameservers for a zone should all be referenced with NS records in the Parent zone
 - Many sites only have a select subset of servers referenced by the parent
 - Useful for staying secure by only allowing exposure of certain hosts
 - When in reality there could be many DNS servers for a zone
 - Usually only 2-3 servers are referenced by the parent
 - Again, this is the GLUE that binds the DNS database together

DNS – Forwarding

- Name servers can forward all queries to another server
 - Stealth servers that may not be able to make outgoing queries due to firewalling
 - These servers cache the answer given
 - But they don't learn about any intervening nameservers
- Sites will designate some beefy servers as NON-forwarding servers
 - All other DNS servers forward requests to these ones
 - This builds a large cache of DNS data on the beefy servers

DNS – Network Information

- Servers listen on port 53 both TCP and UDP
 - Port 53 is listed under /etc/services
 - `% grep domain /etc/services`
domain 53/tcp nameserver
domain 53/udp nameserver
- Most queries use UDP
 - Single packet goes to server with a single packet response
- TCP is used for 'zone transfers'
 - Large amount of data to transfer, TCP makes more sense

DNS – Zone Files

- All Primary zone data is loaded from zone files
 - One zone file per Primary zone
 - File format is standard and is independent of implementation
- Zone files usually maintained by hand
 - Sometimes files are generated by scripts for manageability
 - Straight ASCII files with zone data
- Most entries in zone files are Resource Records (RRs)
 - Few other control directives and options
 - \$TTL, comments

DNS – Zone Files (cont)

- First entry must be the \$TTL directive
 - Sets the default Time-To-Live on all data in the zone
 - The amount of time, in seconds, that a remote server will cache the data
 - Syntax is simple
 - \$TTL 7200
 - Number of seconds is specified as an integer
 - \$TTL is not specified in the SOA record
 - Used to be

DNS – Resource Record Format

- `[name] [ttl] [class] <type> <data>`
 - The name is a domain and is sometimes optional
 - The '@' sign is shorthand for the domain-name of the zone
 - Can be changed with the \$ORIGIN directive
 - When consecutive records refer to the same name
 - Only first record must specify the name
 - Subsequent records MUST start with a blank space
 - When names ARE specified they MUST start at column one
 - Unqualified names (i.e. names without a trailing dot) have the domain appended to the name

DNS – Resource Record Format (cont)

- Fully qualified names **MUST** have a trailing dot
 - You know when you're missing a trailing dot with names like
 - `csel.cs.colorado.edu.cs.colorado.edu.`
- The TTL record can be set explicitly on a record
 - Although never really seen, always uses the default \$TTL at the top
- The 'class' field defaults to the IN class
 - The 'internet' class, the most common
 - Two other classes are CH from ChaosNet and HS from Hesiod
- Type field specifies the Resource Record type
 - e.g. A, NS, TXT, etc

BIND Zone Files - \$TTL Directive

- First entry must be the \$TTL directive
 - Sets the default Time-To-Live on all data in the zone
 - The amount of time, in seconds, that a remote server will cache the resource record data
 - Syntax is simple
 - \$TTL 7200
 - Number of seconds is specified as an integer
 - \$TTL is not specified in the SOA record
 - Minimum TTL is specified in SOA, but is different!!!

BIND Zone Files – SOA Record

- Start of Authority record comes after \$TTL
- Specifies a few things:
 - Root name for the zone
 - Fully qualified Master nameserver name
 - Email address for person responsible for the server
 - Serial no. to identify current zone data
 - refresh, retry, expiry, and minimum TTL fields
- All time values expressed in seconds or symbolic form
 - 7200 or 2H

BIND Zone Files – SOA Record (cont)

- A typical SOA RR looks like this:
 - ```
@ IN SOA cs.colorado.edu. admin.cs.colorado.edu. (
 20010421 ; Serial Number
 86400 ; Refresh - every 24 hours
 1800 ; Retry - 30 minutes
 1209600 ; Expire - 2 weeks
 7200) ; Negative answers - 2 hours
```
- Name
  - Root name of the zone
  - '@' sign is shorthand reference to the zone configured in `/etc/named.conf` for this file

# BIND Zone Files – SOA Record (cont)

- `@ IN SOA cs.colorado.edu. admin.cs.colorado.edu. (`  
    `20010421 ; Serial Number`  
    `86400 ; Refresh - every 24 hours`  
    `1800 ; Retry - 30 minutes`  
    `1209600 ; Expire - 2 weeks`  
    `7200 ) ; Negative answers - 2 hours`
- Class is Internet 'IN'
- Type of record is SOA
- Name of Master nameserver is `cs.colorado.edu`
  - MUST have a trailing dot
  - This is NOT the root of the zone!

# BIND Zone Files – SOA Record (cont)

- @ IN SOA cs.colorado.edu. **admin.cs.colorado.edu.** (  
    **2001042100** ; Serial Number  
    86400 ; Refresh - every 24 hours  
    1800 ; Retry - 30 minutes  
    1209600 ; Expire - 2 weeks  
    7200 ) ; Negative answers - 2 hours
- Contact email address for responsible person
  - Also must have a trailing dot
  - The '@' is replaced with a period '.' because @ has special meaning in zone files
- Serial No.
  - Unique identifier for current zone information

# BIND Zone Files – SOA Record (cont)

- @ IN SOA cs.colorado.edu. admin.cs.colorado.edu. (  
2001042100 ; Serial Number  
**86400** ; Refresh - every 24 hours  
**1800** ; Retry - 30 minutes  
1209600 ; Expire - 2 weeks  
7200 ) ; Negative answers - 2 hours

- Refresh field

- Tells a secondary nameserver how long to wait between checking for updates to the zone

- Retry field

- If the master is down, how long to wait before the next attempt to grab the zone

# BIND Zone Files – SOA Record (cont)

- @ IN SOA cs.colorado.edu. admin.cs.colorado.edu. (  
20010421 ; Serial Number  
86400 ; Refresh - every 24 hours  
1800 ; Retry - 30 minutes  
**1209600** ; Expire - 2 weeks  
7200 ) ; Negative answers - 2 hours

- Expire field

- Tells a secondary how long the data they hold is *authoritative*

- Usually is a very large value for potentially long outages

- Minimum TTL field – two possible meanings

- How long negative answers (NXDOMAIN) are to be cached
- Also could be the TTL for all resource records

# BIND Zone Files – SOA Record Values

- All values can be tweaked from short to long
  - Tradeoffs for efficiency and flexibility
  - Short timeouts mean accurate data (quick changing zones)
  - Long timeouts lend to longer caching (efficiency)
- Refresh value isn't as important anymore
  - Most servers notify secondaries for a zone change
- Expire value should be very long
  - Somewhere between 2-4 weeks

# BIND – Daemon Information

- BIND can run in a 'chrooted jail'
  - 'chroot' means 'change root' – `man chroot`
    - Changes the location of the '/' directory to a subdirectory for your shell
  - Means that the process only sees a subset of the filesystem
  - Used to prevent an exploited process from doing bad things
  - Many other daemons can also run in this way
    - SSHD, Apache, and others
- Jail location is usually `/var/named`
  - All config files, zone files, everything for BIND exist here

# Process Signaling Tangent

- Unix allows the sending of signals to a process
  - Used for inter-process communication
- Many different types of signals to send
  - Hangup, kill, interrupt, stop, and others
- Some signals can be caught by process to perform a task
  - Hangup is most common used for a 'reload'
  - Usually reloads configuration from files on disk
- The `kill` command is used to signal another process

# Process Signaling – kill

- Usage: `kill [-signal] <pid>`
- Using `kill` without a signal sends a TERM signal by default
- Send a HANGUP signal to a process
  - `kill -HUP 4758`
- Send a suspend signal to a process (similar to CTRL-Z)
  - `kill -STOP 7472`
- Send continue signal to process (similar to fg)
  - `kill -CONT 7472`

# Controlling BIND

- All versions of BIND respond to various signals
  - BIND reloads configuration with a HUP signal
- BIND 9 has a control command called `rndc`
  - Speaks to BIND over TCP
  - `rndc` requires its own configuration under `/etc/rndc.conf`
  - Use of `rndc` requires a secret key exchanged for authentication
    - Uses a Message Authentication Code (MAC) function to verify key
    - Key MUST be shared between both hosts
  - Send commands to see the status or reload configuration

# BIND – named Configuration

- Config file `/etc/named.conf`
  - Comments with either `#`, `//`, or `/* ... */`
    - Semicolons are NOT comments unlike zone files!
  - All config statements end with a semicolon ;
- Statements in the file include the following
  - `acl`, `include`, `options`, `key`, `trusted-keys`, `server`, `controls`, `zone`, `logging`, `view`
- Most statements are used more than once
  - Except for 'options' and 'logging'

# BIND – named Configuration (cont)

- Address Match Lists specify a set of IP addresses
  - ACLs are labeled versions of these lists
  - List may consist of one or more of the following types
    - IP address
    - CIDR mask
    - Previously defined ACL
    - A negated version of any of the above with a preceding ! sign
  - Address may also utilize a keyword
    - any, none, localhost (127.0.0.1), localnets (networks attached to network interfaces)

# BIND – acl Statement

- ```
acl <acl-id> {  
    ! 1.2.3.4; 1.2.3/24; localhost;  
};
```
- First-match wins for ACLs
- ACLs must be top-level in the config file
 - Can't be defined within other blocks
- Must also be defined before it is used
 - Otherwise named will barf on startup

BIND – include Statement

- include `"/path/to/file";`
- Absolute paths start at the chrooted jail
 - Relative paths start at the working directory of named
- Does a simple text-insert of the include file
 - The include must create a syntactically correct config file
- Usually rndc keys are included in this manner
 - So the config file is world-readable but the key file is not

BIND – options Statement

- There can be only one!
 - Error is returned otherwise
- Defines global options
 - Can be individually overridden in sub-statements
- Over 50 options available, all viewable in the manpage
 - We'll go over important ones
 - Many setups only require a few options

BIND – options Configuration

- ```
options {
 version "we're not telling!";
 directory "/";
};
```
- **version** - specifies the text to return on a version query
  - If omitted, shows real version by default
- **directory** - sets named's runtime directory
  - "/" for chrooted servers
  - /var/named for non-chrooted servers

# BIND – options Configuration (cont)

- ```
options {  
    . . .  
    notify yes;  
    also-notify { IP1 port 1053; IP2; };  
    listen-on [port NN] {acl_id3}  
};
```
- **notify** – turns on secondary server notification
- **also-notify** – notifies secondaries of new zone data
- **listen-on** – optionally specify a port and a list of allowed addresses to query the server

BIND – key Statement

- ```
key key_id {
 algorithm hmac-md5; // the only alg
 secret "XYZZY";
};
```
- Defines a secret key and MACing algorithm to use
- 'secret' is an unencrypted base64 encoded string
- 'key' statement is top-level
  - Must be unique between different servers if you are managing from a single machine
  - Must match in the server and client rndc config

# BIND – controls Statement

- ```
controls {  
    inet IP_ADDR [port N] allow { acl_id8 }  
    keys { key_id1; key_id2; };  
};
```
- Configures access for rndc
- Multiple 'inet' clauses may be given
- IP_ADDR is the inet interface to listen on
 - use '*' for all interfaces
- Port number defaults to 953
- Keys must be used in BIND 9

BIND - /etc/rndc.conf

- Client rndc configuration, similar to named.conf
- Only four valid statements - key, options, server, include

- key statement looks exactly like it does in named.conf

- ```
key key_id {
 algorithm hmac-md5; // the only alg
 secret "XYZZY";
};
```

- Typical options:

- ```
options {  
    default-server localhost;  
    default-key key_id;  
};
```

BIND – zone Statement

- ```
zone "domain_name" {
 type master; // or slave
 file "path/to/zone/file";
};
```
- Other possible types are `stub`, `forward`, and `hint`
  - Indicate whether your server is primary or secondary
  - Other options allowed here (e.g. zone transfers)
- Special zone type `hint`
  - All nameservers need to include a `hint` zone
  - Includes a seed file for the root nameservers

# BIND – zone Statement (cont)

- Most nameservers have a reverse zone for localhost
  - The 'localhost' name is dealt with in forward zones
  - The in-addr.arpa. address needs to be in a separate zone
  - ```
zone "0.0.127.in-addr.arpa" {  
    type master;  
    file "localhost";  
    notify no;  
};
```
 - Every nameserver is always the 'master' for the zone
 - Zone doesn't need to receive notifications

BIND – logging Statement

- Here is the general syntax for a logging statement
- ```
logging {
 <channel-definition-1>;
 <channel-definition-2>;
 ...
 category <category-a> {
 <channel-name-1>;
 ...
 };
 category <category-b> {
 <channel-name-2>;
 ...
 };
};
```

# BIND – logging Statement (cont)

- Here is the syntax for a channel definition
- ```
channel <channel-name> {  
    file /file/path [versions N | unlimited] [size M];  
    syslog <facility>;  
    severity <severity>;  
    print-category yes; // or no  
    print-severity yes; // or no  
    print-time      yes; // or no  
};
```
- Use either the `file` clause OR the `syslog` + `severity` clauses

BIND – logging Statement (cont)

- There are some default channels
 - default_syslog - logs with facility 'daemon' and severity 'info' and worse
 - default_debug - logs to a file called 'named.run'
 - There are some others
- The default logging statement for Bind 9 is
- ```
logging {
 category default { default_syslog; default_debug; };
};
```

# BIND – view Statement

- Views facilitate a concept known as 'Split DNS'
  - Provides different data for the same zone
    - Information returned depends on the query source
    - Traditionally used to hide some information about hosts from outside
    - Also useful for NATed networks
- If a `view` is used at all, views must be used completely
  - All `zone` statements must occur within a `view`
- Relies on ACLs to match IP addresses to serve data
  - uses the `match-clients` option

# BIND – view Statement Example

- ```
view "internal" {
    match-clients { "my_net"; };
    recursion yes;
    zone "xbalanque.net" {
        type master;
        file "xbalanque.forward";
    };
    zone "0.0.10.in-addr.arpa" {
        type master;
        file "xbalanque.reverse";
    };
    zone "." {
        type hint;
        file "cache.db";
    };
    zone "0.0.127.in-addr.arpa" {
        type master;
        file "localhost";
        notify no;
    };
};
```

Bind Configuration – Views

- Views contain zones and a few options
 - But general options only are declared once
 - Include `/etc/bind/named.conf.options`
- Have to match-clients using the ACL
 - `match-clients { "CSELnet"; };`
 - Watch your semicolons!
- Also-notify and allow-transfer for your slave
- Always have all zones in each view
 - Simply copy/paste other zones into the view
 - They can reference the same files!

Bind Configuration - Chroot

- Create directories and symlinks (all as root)
 - `mkdir -p /etc/bind/etc/bind`
 - `mkdir -p /etc/bind/var/cache/bind`
 - `mkdir -p /etc/bind/var/run/bind/run`
 - `mkdir /etc/bind/dev`
 - `cd /etc/bind/etc/bind`
 - `ln -s ../../named.conf named.conf`
 - `ln -s ../../rndc.key rndc.key`
- Create special nodes (as root)
 - `mknod /etc/bind/dev/null c 1 3`
 - `mknod /etc/bind/dev/random c 1 8`
 - `Chmod 666 /etc/bind/dev/random /etc/bind/dev/null`

Bind Configuration – Chroot (cont)

- Change ownership (can only ever do as root)
 - `chown -R bind:bind /etc/bind/var/*`
- Edit `syslogd` defaults file
 - `vim /etc/default/syslogd`
 - `SYSLOGD="-a /etc/bind/dev/log"`
 - `Sudo /etc/init.d/syslogd restart`
- Edit `bind9` defaults file
 - `vim /etc/default/bind9`
 - `OPTIONS="-u bind -t /etc/bind -c /named.conf"`
- `Sudo /etc/init.d/bind9 restart`