

Unix System Administration

Chris Schenk

Lecture 14 – Tuesday Mar 07

CSCI 4113, Spring 2006

Outline

- Package Management Tools
 - yum, apt-get, emerge, FreeBSD ports
- SSH
 - Asymmetric key introduction, Block ciphers and Modes of Operation
 - Server configuration, Client key generation
- DNS and BIND
 - Some history of DNS
 - Berkeley Internet Name Daemon

Package Management – yum

- Yellowdog Updater Modified
 - Included with Fedora (not RedHat!)
- Usage: `yum <command> <package(s)>`
 - Run `yum` without any parameters to see all commands
- Allows searching/installing/removing of packages
 - Uses RPMs as the packages
- Checks for any required dependencies and installs them
 - By default will prompt you to verify you want to install extras

Package Management - apt-get

- Used in Debian/Ubuntu
 - Usage: `apt-get [opts] <command> <package(s)>`
- Very similar to yum in its commands
 - Have to `man apt-get` to see them, however
- Will also check for dependencies and will prompt to install
- Uses tar.gz files for packages
- Distribution sources managed under
 - `/etc/apt/sources.list`

Package Management – emerge

- The gentoo utility to manage portage tree
 - Very similar to FreeBSD ports tree
- Usage: `emerge [opts] [action] <package>`
 - Default simply installs a package
- All packages are compiled on the machine
 - Optionally can install a pre-compiled package
- `emerge` is ridiculously powerful for managing packages
 - Can fine-tune your machine to run very efficiently

Package Management – ports tree

- The ports tree tracks all available packages on the system
 - exists under `/usr/ports`
- Ports tree performs same task using complex makefiles
 - Allows for searching of ports tree on the system
 - Usage: `[/usr/ports]% make search name=nmap`
- Everything is also compiled with portage
 - And the option of downloading a pre-built package
 - Usage: `[/usr/ports/security/nmap]% make`

SSH – Secure Shell

- Built on asymmetric and symmetric key encryption
- Provides secure communication between client and server
- Designed to replace many insecure tools under unix
 - ssh, scp, sftp
- Other utilities included
 - ssh-agent, ssh-keygen, ssh-keysign, and others
- Tunneling for other protocols, X windows
 - Provides encryption for other programs

SSH – RSA Keys

- Asymmetric key relationship
 - Built on premise that factoring large numbers is difficult
- RSA provides an initial communication
 - Which is VERY slow compared to symmetric keys
 - Allows a secure transfer of symmetric keys
- Server always provides the public key for communication
 - “RSA key fingerprint is XX:XX...
Are you sure you want to continue connecting (yes/no)?”

SSH – RSA Keys (cont)

- Asymmetric keypairs split into 'public' and 'private' halves
 - Each half is used for communication in a specific direction
- A message encrypted with the 'public' key can only be decrypted with the 'private' key
 - and vice-versa
- SSH server uses the 'private' key to send data to client
 - Client decodes with the public key originally presented

SSH – Symmetric Keys

- Asymmetric keys are butt slow
- Symmetric keys takeover after initial RSA exchange
 - Encryption algorithms (block ciphers) use a single shared key
- Key size depends on algorithm used
 - AES supports 128, 192, 256 bit key lengths
- Block size is the max length of input to the block cipher
 - Messages must be split into parts equal to the block size
 - This is NOT the same as key size!

SSH – Large Numbers

- Remember each bit doubles the keyspace
- Years till next ice age 2^{14}
- Age of the universe 2^{34}
- Number of atoms in the planet 2^{170}
- Number of atoms in the sun 2^{190}
- Number of atoms in the galaxy 2^{223}
- Number of atoms in the universe 2^{265}
- Number of bits in an RSA key 2^{1024}

SSH – Block Ciphers

- A Block Cipher is an encryption algorithm
 - Takes a fixed sized input
 - Produces an output of EQUAL size to the input
- A “good” block cipher creates output that is seemingly indistinguishable from random
 - Output is based on the **input and the key**
 - What does “good” mean?
- Many different block ciphers available
 - 3DES, **AES**, Blowfish, and others

SSH – Modes of Operation

- A block cipher must be reversible
 - Have to get the message back somehow!
 - encryption/decryption uses the shared key
 - The input message *ALWAYS* produces the same ciphertext
- A mode of operation is key to secure communication
 - Provides a way of further randomizing the ciphertext
 - Prevents frequency pattern attacks
- Common modes – CBC (cipher block chaining) and CTR (counter)

SSH – sshd Configuration

- Usually found under `/etc/ssh/sshd_config`
 - Sometimes `/etc/openssh/sshd_config`
 - Server keys also found in this location
- Many different options for how the server behaves
 - Disallow root access
 - Disallow empty passwords
 - Allow only client ssh keypairs (no password authentication)
 - Set a login banner
 - And more

SSH – ssh-keygen

- Create an SSH keypair
- Usage: `ssh-keygen -t <type> [-f filename] [-b bits]`
 - `ssh-keygen -t rsa`
 - If options are omitted you may be prompted
 - Default keysize is 1024 bits (2^{1024})
- Optionally you can enter a passphrase
 - Stronger protection of your logins, highly recommended
- View a key's fingerprint (very useful)
 - `ssh-keygen -l ~/.ssh/id_rsa.pub`

SSH – ssh-agent

- Keeps track of all authentication information in memory
 - You open your ssh keys once at the beginning with passphrase
 - All passphrases tracked thereafter for the duration of ssh-agent
 - Useful if you ssh to one machine more than once
 - And want to keep access very secure!
- ssh-agent runs in the background
 - So any new shells you spawn are tracked by ssh-agent
 - May not be true for older versions

DNS – History

- The Domain Name System proposed in early 80s
 - To address an important need and serious problem
- Prior to DNS, all internet machines used a common HOSTS.TXT file
 - Provided all hostnames and addresses in use
 - As more machines came online, HOSTS file became huge
- The need was host->address translation
- Problem was

DNS – History (cont)

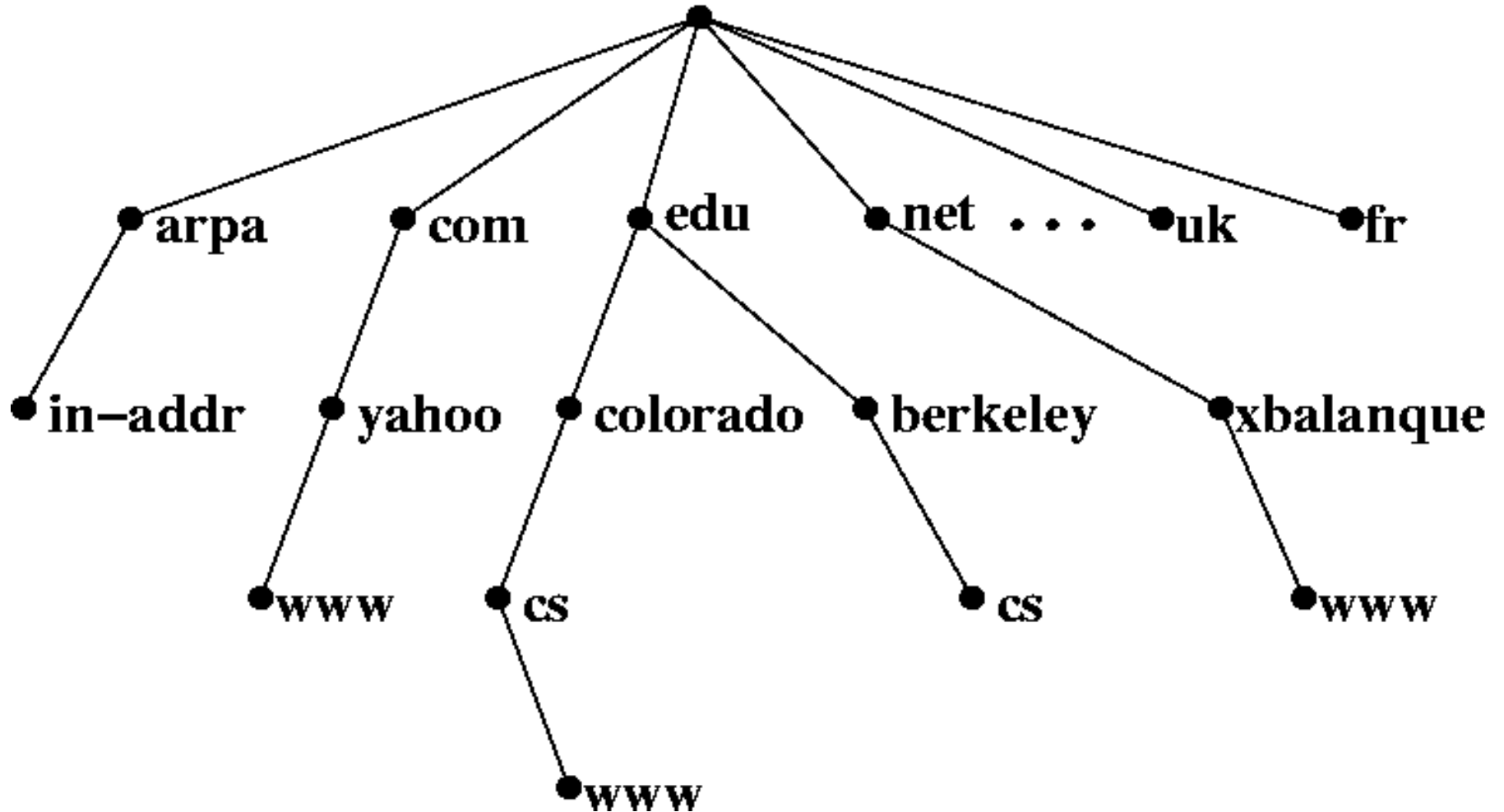
- The need was host address translation
 - Forward lookups resolve names->IP addresses
 - Reverse lookups for IP addresses->name resolution required
 - Some programs required hostnames and IPs to point to each other exclusively
- The problem was one of scale
 - Master HOSTS.TXT file similar in format to modern /etc/hosts
 - Hosts file eventually too large to download via ftp
- Solution was DNS as a distributed database
 - No reliance on a central server or file

DNS – Distributed Database

- Key to keep the internet functioning
 - No machine could store all the data
 - cs.colorado.edu domain has over 1700 hosts alone
 - Solves the 'large hosts file' and 'single machine bottleneck' problems
 - Individual organizations responsible for their portion of the DB
- Original intent was to simply replace HOSTS.TXT
 - Now DNS holds more such as location information (LOC) or email delivery information (MX)

DNS – Domain Name Space

- Distributed database framework



DNS – Domain Name Space (cont)

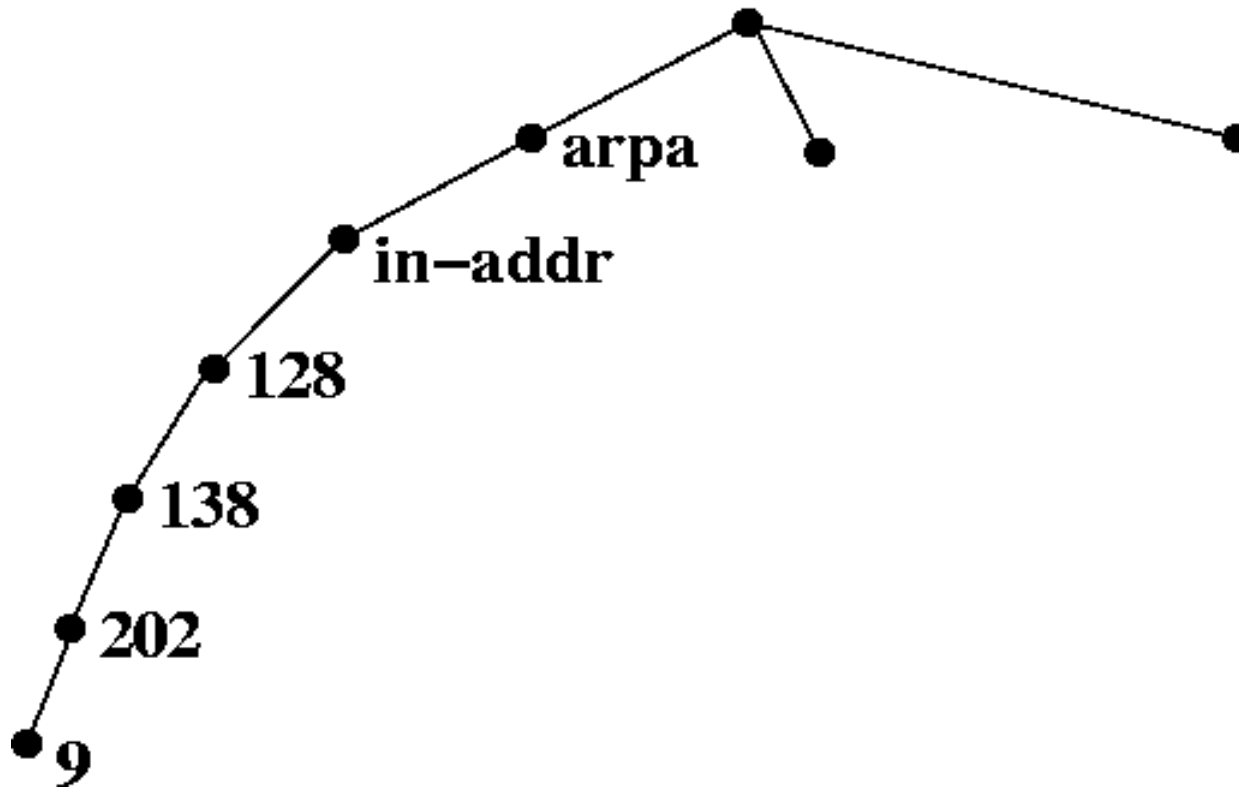
- Nodes in the tree have labels called Domain Names
 - Names consist of up to 63 characters
 - Case insensitive, but most programs propagate case anyway
 - Alphanumeric characters and hyphen '-' allowed
 - Shorter and intuitive names are useful for poor typists
- Sibling nodes **MUST** be unique
 - by definition of a tree
- Leaf nodes are still considered to be domains
 - Although most people call them 'hosts'

DNS – Fully Qualified Domain Names

- FQDN
 - Each node name from the leaf node is concatenated with dots
 - `www.cs.colorado.edu`
 - Officially the root node has the name of NULL string
 - `www.cs.colorado.edu.NULL`
 - Because NULL is usually blank (duh), we sometimes have a trailing dot
 - `www.cs.colorado.edu.`
 - Limitation of 127 node depth
 - Unlikely ever to be a problem

DNS – in-addr.arpa

- This subtree is used for reverse lookups



DNS – Zones and Delegation

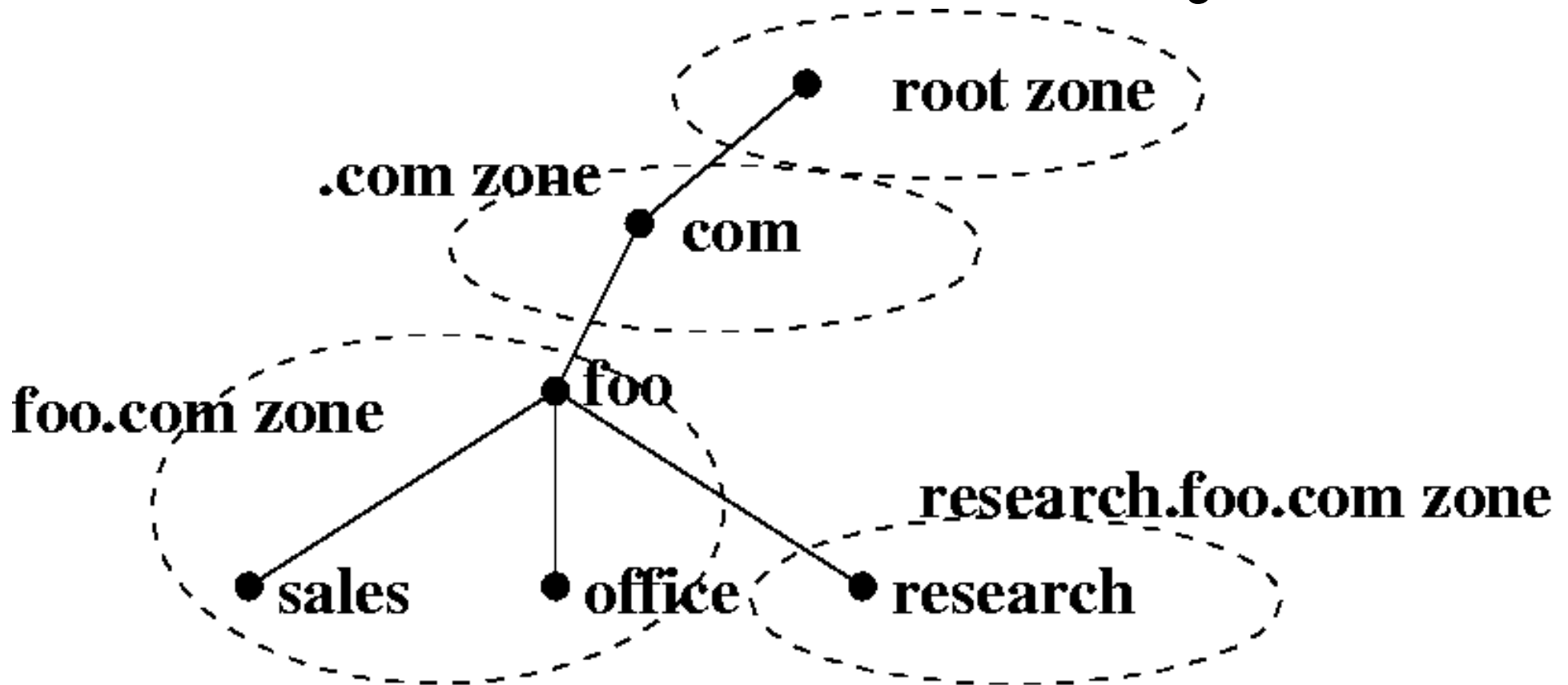
- Authority for a zone is delegated by a parent zone
 - The **root** zone delegates **edu** authority to people who manage the **edu** zone
 - The **edu** zone delegates **colorado.edu** authority to the University of Colorado
 - CU then delegates the **cs.colorado.edu** zone to Computer Science
- Delegation of authority means exactly 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

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 important records are categorized as 'zone' records
 - SOA “Start of Authority” record