Introduction

RH253
Red Hat Network Services and Security Administration
Copyright

- The contents of this course and all its modules and related materials, including handouts to audience members, are Copyright © 2005 Red Hat, Inc.
- No part of this publication may be stored in a retrieval system, transmitted or reproduced in any way, including, but not limited to, photocopy, photograph, magnetic, electronic or other record, without the prior written permission of Red Hat, Inc.
- This curriculum contains proprietary information which is for the exclusive use of customers of Red Hat, Inc., and is not to be shared with personnel other than those in attendance at this course.
- This instructional program, including all material provided herein, is supplied without any guarantees from Red Hat, Inc. Red Hat, Inc. assumes no liability for damages or legal action arising from the use or misuse of contents or details contained herein.
- If you believe Red Hat training materials are being used, copied, or otherwise improperly distributed please email training@redhat.com or phone toll-free (USA) +1 866 626 2994 or +1 919 754 3700.
Red Hat Enterprise Linux

- Enterprise-targeted operating system
- Focused on mature open source technology
- 12-18 month release cycle
  - Certified with leading OEM and ISV products
- Purchased with one year Red Hat Network subscription and support contract
  - Support available for seven years after release
  - Up to 24x7 coverage plans available
Red Hat Network

- A comprehensive software delivery, system management, and monitoring framework
  - **Update Module**, included with Red Hat Enterprise Linux, provides software updates
  - **Management Module** adds more scalable management capabilities for large deployments
  - **Provisioning Module** provides bare metal installation, configuration management, and multi-state configuration rollback capabilities
Red Hat Desktop

- High-quality, full-featured client system based on Red Hat Enterprise Linux
  - Includes desktop productivity applications
- Available in packages of 10 or 50 units for mass deployments of desktop systems
- Clients entitled to RHN Management Module
  - Package may also include RHN Proxy Server or Satellite Server
Red Hat Applications

- Open source applications provided separately from Red Hat Enterprise Linux
- Include a range of support options
- Installation media and updates provided through Red Hat Network
- More information on specific products at http://www.redhat.com/software/rha/
The Fedora Project

- Red Hat-sponsored open source project
- Focused on latest open source technology
  - Rapid four to six month release cycle
  - Available as free download from the Internet
- An open, community-supported proving ground for technologies which may be used in upcoming enterprise products
  - Red Hat does not provide formal support
Objectives of RH253

- Learn skills of the system administrator who can configure Red Hat Enterprise Linux common network services and security at a basic level
Audience and Prerequisites

- **Audience:** Linux or UNIX operators who can perform system administration tasks to a level where he/she can install, configure, and attach a new Red Hat Linux workstation to an existing network.
- **Prerequisites:** experience in Linux or UNIX administration at the single-workstation level
# Classroom Network

<table>
<thead>
<tr>
<th></th>
<th>Names</th>
<th>IP Addresses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Our Network</td>
<td>example.com</td>
<td>192.168.0.0/24</td>
</tr>
<tr>
<td>Our Server</td>
<td>server1.example.com</td>
<td>192.168.0.254</td>
</tr>
<tr>
<td>Our Stations</td>
<td>station\textit{X}.example.com</td>
<td>192.168.0.\textit{X}</td>
</tr>
<tr>
<td>Their Network</td>
<td>cracker.org</td>
<td>192.168.1.0/24</td>
</tr>
<tr>
<td>Their Server</td>
<td>server1.cracker.org</td>
<td>192.168.1.254</td>
</tr>
<tr>
<td>Their Stations</td>
<td>station\textit{X}.cracker.org</td>
<td>192.168.1.\textit{X}</td>
</tr>
</tbody>
</table>

(For Stations, \textit{X} is a number between 1 and 20)
UNIT 1

Introduction to System Services
Objectives

- Understand how services are managed
- Learn common traits among services
- Introduce service fault analysis methods
Agenda

- Service management concepts
- System V-managed services
- \texttt{xinetd} managed services
- The /etc/sysconfig files
- Fault Analysis
Service Management

- Services are managed several ways:
  - by `init`
  - by System V scripts
  - by direct command
  - by `xinetd`
Services Managed by init

- Typically non-TCP/IP services, for example dial-in modems
- Provides respawn capability
- Configured in /etc/inittab
System V Service Management

- Processes are “wrapped” by System V (‘SysV’) initialization script methods
- More than one script, and several configuration files are often used, per service
- The service command is a “wrapper of wrappers”
  - /etc/init.d/cups start
  - service cups start
chkconfig

- Manages service definitions in run levels
- To start the cups service on boot: `chkconfig cups on`
- Does not modify current run state of System V services
- List run level definitions with `chkconfig --list`
**xinetd Managed Services**

- Services are started by `xinetd` in response to incoming request
- Activated with `chkconfig`
  - `chkconfig cups-lpd on`
- Uses files in `/etc/xinetd.d/`
The `xinetd` daemon

- Manages network-specific resources and authentication
  - less-frequently needed services
  - host-based authentication
  - service statistics and logging
  - service IP redirection
- Replaces `inetd`
- Linked with `libwrap.so`
- Configuration files: `/etc/xinetd.conf`, `/etc/xinetd.d/service`
xinetd default controls

- Top-level configuration file
  - /etc/xinetd.conf
**xinetd service controls**

- Service specific configuration
  - `/etc/xinetd.d/<service>`
The /etc/sysconfig files

- Some services are configured for *how* they run
  - named
  - sendmail
  - dhcpd
  - samba
  - init
  - syslog
Fault Analysis

- Determine the severity of the fault
  - Is it the data?
  - Is it the program or application?
  - Is it the operating system?
  - Is it the hardware?
- Inspect logs *before* configuration files
- Use command options for debugging
- Document your investigation
Security Enhanced Linux

- Who can do what to which files?
  - Mandatory access control (SELinux)
    - Under the control of the security administrator
  - Discretionary access control (Traditional Linux)
    - Under the control of the user
SELinux

- Each process or object (file, directory, network socket) also has a SELinux context
  - identity:role:domain/type
- The SELinux policy controls
  - What identities can use which roles
  - What roles can enter which domains
  - What domains can access which types
SELinux Installation Options and Control

- Installation Options
  - Disabled
  - Warn (Permissive)
  - Active (default) (Enforcing)
- Control Options when SELinux is enforced
  - Targeted (default)
  - Strict
Controlling SELinux

- `system-config-securitylevel`
- `setenforce` and `setsebool`
- `/etc/sysconfig/selinux`
- `enforcing=0`
- `/selinux virtual filesystem`
SELinux Contexts

- List process contexts: `ps -Z`
- List file contexts: `ls -Z`
- Change file contexts: `chcon`
  - `chcon -t http_sys_content_t index.html`
  - `chcon -reference=/var/www/html index.html`
Troubleshooting SELinux

- What is the error?
  - Check `/var/log/messages` for `avc` denials
- Is the process doing something it should not?
- Does the target have the right context?
- Does a “boolean” setting need adjustment?
End of Unit 1

- Address questions
- Preparation for Lab 1
  - Goals
  - Sequences
  - Deliverables
- Please ask the instructor for assistance when needed
Objectives

- Explain networked systems organization
- Describe the Domain Name System (DNS)
- Explain the BIND DNS service
- Learn how to configure BIND
- Understand BIND utilities
- Explain the DHCP service
Agenda

- DNS operational overview
- Configuring BIND
- Creating BIND databases
- Additional DNS methods
- Using BIND tools
- Configure DHCP services
Domain Name System (DNS)

- Resolves hostnames into IP addresses (forward lookup)
- Resolves IP addresses into hostnames (reverse lookup)
- Allows machines to be logically grouped by name *domains*
- Provides email routing information
Zones, Domains & Delegation

- A domain is a complete sub-tree of the hierarchical namespace.
- A zone is the part of the domain managed by a particular server.
- Subdomains may be delegated into additional zones.
- A zone may directly manage some subdomains.
Name Server Hierarchy

• Master name server
  • Contains the master copy of data for a zone.

• Slave name server
  • Provides a backup to the master name server
  • All slave servers maintain synchronization with their master name server
The DNS Server

- Server receives request
- If server doesn't have answer, either asks root server or forwards request
- Response from upstream server may be final answer or referral to another name server
- lwresd
Berkeley Internet Name Domain (BIND)

- BIND is the most widely used DNS server on the Internet
  - Red Hat Enterprise Linux uses BIND 9
  - Provides a stable and reliable infrastructure on which to base a domain's name and IP address associations
  - Runs in a chrooted environment
Service Profile: DNS

- **Type:** System V-managed service
- **Packages:** bind, bind-utils, bind-chroot
- **Daemons:** named, rndc
- **Script:** named
- **Ports:** 53 (domain), 953(rndc)
- **Configs:** (Under /var/named/chroot)
  /etc/named.conf,
  /var/named/*, /etc/rndc.*
- **Related:** caching-nameserver, openssl
bind-chroot

- Config file:
  /etc/sysconfig/named

- Define chroot directory:
  ROOTDIR=/var/named/chroot
Configuring BIND

- Default configuration file is
  `/var/named/chroot/etc/named.conf`
- Read by `named` (BIND daemon) during startup or service `named reload`
- Text-file specifying directives: zones, options, access control lists, etc.
- Comments can be in C, C++ or shell style
Global Options

- Declared with the `options` directive:

```plaintext
acl "mynetwork" { 192.100.100/24; }
options {
    directory    "\var\named";
    forwarders   { 203.50.0.137; }
    allow-query  { mynetwork; }
    allow-transfer { mynetwork; }
}
```
Access Control Lists (acl)

- Access control list is a list of semi-colon separated IP addresses, networks, or named access control lists.
- Can use acl directive to create a custom named access control list:
  ```
  acl "mylist" { 192.168.0/24;
  192.168.1.12; }
  ```
- Trailing, non-significant zeros may be dropped.
- Makes the configuration easier to read and maintain.
Name Daemon Control Utility (rndc)

- Provides secure and remote management of running name server
- Uses TSIG security

```bash
include "/etc/rndc.key";
controls {
inet 127.0.0.1 allow { localhost; } \n  keys { rndckey; };
);
```

- `rndc` only listens to the loopback interface, or "localhost" by default
Master and Slave Zones

- Declared with the `zone` directive:

  ```
  zone "redhat.com" {
    type master;
    file "redhat.com.zone";
  };
  
  zone "kernel.org" {
    type slave;
    masters { 192.168.192.168; };
    file "slaves/kernel.org.zone";
  };
  ```

- File name should indicate the zone
Reverse Lookup Zones

- Zone name ends with special domain: `.in-addr.arpa`
- Declared with the `zone` directive:

```plaintext
zone "10.100.172.in-addr.arpa" {
    type slave;
    masters { 172.100.10.1; }
    file "slaves/172.100.10.zone";
};
```
Special Zones

• Root zone: "."

```plaintext
zone "." {  
    type         hint;
    file         "named.ca";
};
```

• Loopback zone: "0.0.127.in-addr.arpa"
  • Specified like other reverse lookup zones
Zone Files

- Files usually reside in
  /var/named/chroot/var/named
- Begins with $TTL (time to live)
- First resource record is zone's start of authority (SOA)
- Zone data in additional resource records
Resource Records (RR)

- Syntax:
  \[ \text{domain} \] \[ \text{ttl} \] \[ \text{class} \] \text{<type>} \text{<rdata>} \]
  
  - \[ \text{domain} \] specify domain or use current
  - \[ \text{ttl} \] how long record will be cached
  - \[ \text{class} \] record classification (usually IN)
  - \text{<type>} record type (SOA, MX, A, etc.)
  - \text{<rdata>} specific data for record
SOA (Start of Authority)

- Every zone file must have one

```plaintext
@ IN SOA ns.redhat.com. root.redhat.com. (2001042501 ; serial number
300 ; refresh
60 ; retry
1209600 ; expire
43200 ; minimum TTL for negative answers)
```

- Values no longer need be in seconds
NS (Name Server)

- There should be an NS record for each master or slave name server serving your zone.
- NS records point to any slave servers that should be consulted by the client's name server if the master should fail.

```plaintext
@       IN NS ns.redhat.com.
```
Main Record Types

- **A records** map hostname to IP address
  - mail IN A 192.100.100.3
  - login.redhat.com. IN A 192.100.100.4

- **CNAME records** map address aliases
  - pop IN CNAME mail
  - ssh IN CNAME login.redhat.com.

- **PTR records** map IP address to hostname
  - 4.100 IN PTR login.redhat.com.

- **MX records** map mail servers for a domain
  - redhat.com. IN MX 5 mail.redhat.com.
Example Zone File

- SOA record
- NS records
- A records
- CNAME records
- MX records
Round-Robin Load Sharing Through DNS

- Load balancing can be achieved through the simple use of multiple A records:

  www    0    IN    A    192.168.34.4
  www    0    IN    A    192.168.34.5
  www    0    IN    A    192.168.34.6

- DNS traffic will increase as a TTL of 0 is never cached
Delegating Subdomains

- Configure the subdomain as a zone on the new server
- On delegating server, set up NS record for the subdomain pointing to the new server
- If new server is in subdomain it manages, on delegating server need a "glue" A record for new server
BIND Syntax Utilities

- BIND will fail to start for syntax errors
  - named-checkconf
    - Inspects /var/named/chroot/etc/named.conf by default
  - named-checkzone
    - Inspects a specific zone configuration

```bash
named-checkzone redhat.com
   /var/named/chroot/var/named/redhat.com.zone
```
Caching-only Name Server

- The caching name server configuration; forwards queries and caches results.
  - `caching-nameserver` RPM package provides a working `named.conf` BIND configuration
  - Also provides Internet root server "hints" or references via `named.ca`
BIND Utilities

- Many useful utilities are included in the bind-utils RPM package, including:
  - `host`: gather host/domain information
    ```
    host -a ns.redhat.com
    host -al redhat.com
    ```
  - `dig`: send queries to name server directly
    ```
    dig @ns redhat.com any
    ```
  - `nslookup`
Advanced BIND Features

- Integration with `dhcppd` to implement Dynamic DNS (DDNS) updates from the DHCP server
- DDNS updates directly from clients
- Transaction Signatures (TSIG) for secure exchanges between name servers
DHCP Overview

- DHCP: Dynamic Host Configuration Protocol, implemented via dhcpd

- dhcpd provides services to both DHCP and BOOTP clients
Service Profile: DHCP

- **Type:** SystemV-managed service
- **Packages:** dhcp
- **Daemons:** dhcpd
- **Script:** dhcpd
- **Ports:** 67 (bootps), 68 (bootpc)
- **Configuration:** /etc/dhcpd.conf, /var/lib/dhcp/dhcpd.leases
- **Related:** dhclient
Configuring a DHCP Server

- Configure the server in
  `/etc/dhcpd.conf`
- Sample configuration provided under
  `/usr/share/doc/dhcp-<version>/`
- There must be at least one subnet block, and it must correspond with configured interfaces.
End of Unit 2

- Address questions
- Preparation for Lab 2
  - Goals
  - Scenario
  - Deliverables
- Please ask the instructor for assistance when needed
UNIT 3

Network File Sharing Services
Objectives

- Explain Network File Sharing
- Describe the NFS service
- Describe the FTP service
- Describe the SMB/CIFS service
- Use client tools with each service
Agenda

- Introduction to NFS
- Configuring the NFS service
- Introduction to FTP
- Configuring the FTP service
- Introduction to Samba (SMB)
- Configuring the SMB service
Network File Service (NFS)

- The Red Hat Enterprise Linux NFS service is similar to other BSD and UNIX variants
  - Exports are listed in `/etc/exports`
  - Server notified of changes to exports list with `exportfs -r`
  - Shared directories are accessed through the `mount` command
  - The NFS server is an RPC service and thus requires `portmap`
- Red Hat Linux supports NFS version 3.0 on the client, and most 3.0 features on the server
Service Profile: NFS

- Type: System V-managed service
- Packages: nfs-utils
- Daemons: nfsd, lockd, rpciod, rpc.{mountd,rquotad,statd}
- Scripts: nfs, nfslock
- Ports: Assigned by portmap (111)
- Configuration: /etc/exports
- Related: portmap (mandatory)
NFS Server

- Exported directories are defined in /etc/exports
- Each entry specifies the hosts to which the filesystem is exported plus associated permissions and options
  - options should be specified
  - default options: (ro, sync)
  - root mapped to UID 65534 (nfsnobody)
Client-side NFS

- implemented as a kernel module
- `/etc/fstab` can be used to specify network mounts
- NFS shares are mounted at boot time by `/etc/rc.d/init.d/netfs`
- `autofs` mounts NFS shares on demand and unmounts them when idle
File Transfer Protocol (FTP)

- **vsftpd** - the default RHEL ftp server
- No longer managed by **xinetd**
- Allows anonymous or real user access only
- The anonymous directory hierarchy is provided by the **vsftpd** RPM
- **/etc/vsftpd/vsftpd.conf** is the main configuration file
Service Profile: FTP

- Type: SystemV-managed service
- Packages: vsftpd
- Daemons: vsftpd
- Script: vsftpd
- Ports: 21 (ftp), 20 (ftp-data)
- Configuration: /etc/vsftpd/vsftpd.conf
  /etc/vsftpd.ftpusers
  /etc/pam.d/vsftpd
- Logs: /var/log/vsftpd.log
Samba services

• Four main services are provided:
  • authentication and authorization of users
  • file and printer sharing
  • name resolution
  • browsing (service announcements)

• Related
  • `smbclient` command-line access
  • `smbfs` Linux can mount an SMB share
Samba Daemons

- `smbd`: SMB/CIFS server
  - authentication and authorization
  - File and printer sharing

- `nmbd`: NetBIOS name server
  - resource browsing
  - WINS server
Service Profile: SMB

- Type: System V-managed service
- Packages: `samba{,-common,-client}`
- Daemons: `nmbd, smbd`
- Script: `smb`
- Ports:`(netbios) 137(-ns), 138(-dgm), 139(-ssn)`
- Configuration: `/etc/samba/*`
- Related: `system-config-samba`
Configuring Samba

- Configuration in `/etc/samba/smb.conf`
  - Red Hat provides a well-commented default configuration, suitable for most situations
- Configuration tools are available
  - `system-config-samba`
  - Hand-editing `smb.conf` is recommended
Overview of *smb.conf*

Sections

`smb.conf` is styled after the `.ini` file format and is split into different `[]` sections

- `global` : section for server generic or global settings
- `homes` : used to grant some or all users access to their home directories
- `printers` : defines printer resources and services
Configuring File and Directory Sharing

- Shares should have their own [ ] section
  - Some options to use:
    - `public` - share can be accessed by guest
    - `browseable` - share is visible in browse lists
    - `writable` - resource is read and write enabled
    - `printable` - resource is a printer, not a disk
    - `group` - all connections to the share use the specified group as their primary group
Printing to the Samba Server

- All printers defined in `/etc/cups/printers.conf` are shared as resources by default.
- Can be changed to allow only explicitly publicized printers.
Authentication Methods

- Specified with `security = <method>`
- Valid methods are:
  - `user`: validation by user and password (this is the default)
  - `share`: user validation on per-share basis
  - `domain`: a workgroup with a collection of authentication data is used
  - `ads`: acts as an “Active Directory” member with Kerberos authentication
Passwords

- Encrypted password considerations
  - Stored in `/etc/samba/smbpasswd`
  - Users managed with `smbpasswd`
  - Users must have local accounts, or implement `windbind`, a separate service
Samba Client Tools: `smbclient`

- Can be used as an `ftp`-style file retrieval tool
  - `smbclient //machine/service`
    - `cd` directory
    - `get` file
- Allows for simple view of shared services
  - `smbclient -L` hostname
- `user%password` may be specified with `-u` or by setting and exporting the `USER` and `PASSWD` environment variables
nmblookup

- list specific machine
  nmblookup -U server -R 'name'
- list all machines
  nmblookup /*
smbmount

- The SMB file system is supported by the Linux kernel

- Use `smbmount` to mount a SMB-shared resource:
  `smbmount service mountpoint -o options`
Samba Mounts in /etc/fstab

- Samba mounts can be performed automatically upon system boot by placing an entry in /etc/fstab

- Specify the UNC path to the samba server, local mount point, smbfs as the file system type, and a user name.
End of Unit 3

- Questions and Answers
- Preparation for Lab 3
  - Goals
  - Scenario
  - Deliverables
- Please ask the instructor for assistance when needed
UNIT 4

Electronic Mail Services
Objectives

- Understand electronic mail (email) operation
- Review email transmission
- Basic Sendmail server configuration
- Evaluate the m4 macro language
- Learn debugging techniques for email servers
- Evaluate Postfix
- Learn to configure Procmail
Agenda

- Sendmail features
- Email overview
- Basic Sendmail configuration
- Using the m4 macro language
- Debugging Sendmail
- Basic Postfix configuration
- Configuring Procmail
Sendmail Features

- Allows many different types of email addresses to be routed
- Supports virtual domains and users
- Allows masquerading of users and machines
- Provides automatic retry for failed delivery and other error conditions
Security and "Anti-spam" Features

- Many security features and options:
  - rejects email from unresolvable domains
  - full access control for users, machines, and domains
  - default configuration allows only local connections
  - no longer a setuid root program

- "Anti-spam" features
  - no relaying by default
  - access databases
  - Email header checks
  - interoperability with spamassassin
An Email Review

- Mail user agent (MUA) passes message to mail transport agent (MTA)
- MTA routes message to destination, giving to other intermediate MTAs as necessary
- Domain MTA passes message to mail delivery agent (MDA)
- User receives message
Server Operations

- User's email agent connects to the local MTA as an unprivileged mail submission program (MSP)
- Local MTA queries DNS for destination's MX
- Local MTA opens a TCP/IP connection to port 25 of the target MX
- Both email servers negotiate a SMTP (Simple Mail Transport Protocol) connection
- Target MX allows or rejects email delivery or relaying based upon its own rulesets
Service Profile: Sendmail

- **Type:** System V-managed service
- **Packages:** `sendmail{-cf,-doc}`
- **Daemons:** `sendmail`
- **Script:** `sendmail`
- **Ports:** 25 (smtp)
- **Configuration:**
  `/etc/mail/sendmail.cf,`  
  `/etc/mail/submit.cf,`  
  `/etc/aliases,/etc/mail/`  
  `/usr/share/sendmail.cf/`
- **Related:** `procmail`
Main Configuration Files

- `/etc/mail/sendmail.cf` is the main configuration file for Sendmail:
  - Contains domain alias directives, header rewriting directives, relaying rules, etc.
  - Edit this file with care and comprehension
- `/etc/mail/submit.cf` is used when Sendmail is called by a user program
  - Normally does not need modification
Other Configuration Files

- `/etc/aliases` defines local user aliases
  - needs to be hashed to `aliases.db` with the `newaliases` command
- `/etc/mail/` contains access control, virtual user database, and configuration source files
  - `local-host-names`
Sendmail Configuration with the m4 Macro Language

- m4 is a macro language that can help configure the sendmail.cf file
- Red Hat’s default Sendmail configuration is generated from the m4 specification in /etc/mail/sendmail.mc
- Red Hat recommends configuring Sendmail with m4 using sendmail.mc as a starting point
Sendmail m4 Macro File: Introduction

- All `sendmail.mc` macro configuration files should define the OS type, file locations, desired features, and mailer and user tables
- Step through header and definitions in the `sendmail.mc` below
Sendmail m4 Macro File: Features

- Investigate the features enabled and disabled in the continuing example below:
Sendmail Client Configuration

- Often, clients do not accept incoming mail themselves
  - A central mail server accepts all incoming mail and relays all outgoing mail
    - MAIL_HUB, SMART_HOST defines
    - Central mail server must allow relaying from the client and have local-host-names set up
  - Useful for client to “masquerade” as the server in From: addresses
    - MASQUERADE_AS(`example.com')
Other Valuable m4 directives

- FEATURE(`dnsbl')
  - checks a DNS implemented blackhole list to block email spammers
- FEATURE(`relay_based_on_MX')
  - Automatically allows relaying if sendmail server is listed as the target domain’s MX record
Additional Sendmail Configuration Files

- `/etc/mail` is now considered the default Sendmail configuration directory
- `virtusertable` maps virtual addresses to real addresses
- `access` specifies rejection or acceptance criteria for email from specified domains
/etc/mail/virtusertable

Allows multiple virtual domains and users to be mapped to other addresses:

- admin@123.com
- admin@xyz.org
- pageme@he.net
- @cba.com
- @dom1.org
- shopper
- jdj
- 1miwtc@pg.com
- cba@aol.com
- %1@dom2.org
/etc/mail/access

Used to accept or deny incoming email:

90trials spammer@aol.com REJECT
spamRus.net REJECT
204.168.23 REJECT
10.3 OK
virtualdomain1.com RELAY
user@dom9.com ERROR:550 mail discarded
nobody@ ERROR:550 bad name
Blacklisting Recipients

- FEATURE('blacklist_recipients')
  - Block mail destined for certain recipients

- Any entry in the access file that has a REJECT or returns an error code will be a blacklisted recipient
Debugging Sendmail

- `/etc/mail/local-host-names`
  - must contain server's name and aliases
- `mail -v user`
  - view SMTP exchange with local relay
- `mailq and mailq -Ac`
  - view messages queued for future delivery
- `tail -f /var/log/maillog`
  - View log in real-time
Using alternatives

- alternatives configures the server software through a generic name
  - generic name is a link to a link in /etc/alternatives/
  - only the links in /etc/alternatives/ are modified
- alternatives displays and sets link groups
  - alternatives --display name
  - alternatives --config name
- system-switch-mail
Postfix

- A replacement for Sendmail
- Project goals:
  - Sendmail-compatible
  - Speed
  - Ease of Administration
  - Security
- Efficient application design based on a modular suite of programs
Service Profile: Postfix

- **Type:** SystemV-managed service
- **Packages:** postfix
- **Daemons:** master, nqmgr, smtpd, pickup, (others)
- **Script:** postfix
- **Ports:** 25 (smtp)
- **Configuration:** /etc/postfix/main.cf
  /etc/postfix/master.cf
- **Related:** procmail
Configuring Postfix

- Activate with alternatives
- Set up minimal configuration directives
  - using postconf
  - using a text editor
- Start with service
Additional Postfix Configuration

- /etc/postfix/ files share syntax and function with those of /etc/mail/
  - virtual - virtual domain mapping
  - access – mail routing controls
- /etc/aliases can be used by postfix, as is
- Postfix command utilities
  - postmap
  - postalias
Enhanced Postfix Configuration

- Pre-receipt header and body checks
- Multiple transports (uucp, X.400)
- Virtual domain support
- UCE controls (blacklists, helo/sender)
- Table lookups (SQL, LDAP)
Procmail Delivery

- Procmail is a very powerful delivery tool
- Different uses include
  - sorting incoming email into different folders or files
  - preprocessing email
  - starting an event or program when email is received
  - Automatically forwarding email to others
- Additional MTA configuration may be required
Procmail Sample Configuration

• Usually located in a user’s home directory:
  /home/bob/.procmailrc

• To forward mail from Joshua about ADSL to Jim, but also copy to the ADSL folder:
  :0
  *^From:*joshua
  *^Subject::*ADSL
  { :0 c
    Jim@somedomain.org
  :0:
  ADSL
  }
End of Unit 4

- Address questions
- Preparation for Lab 4
  - Goals
  - Scenario
  - Deliverables
- Please ask the instructor for assistance when needed
UNIT 5
The HTTP Service
Objectives

- Learn the major features of the Apache HTTP server
- Be able to configure important Apache parameters
- Learn per-directory configuration
- Learn how to use CGI with Apache
- Identify key modules
- Understand proxy web servers
Agenda

- Introduce Apache Features
- Apache configuration files and important parameters
- Using CGI with Apache
- Key modules
- Squid proxy server
Apache Overview

- Process control:
  - spawn processes before needed
  - adapt number of processes to demand

- Dynamic module loading:
  - run-time extensibility without recompiling

- Virtual hosts:
  - Multiple web sites may share the same web server
Service Profile: HTTPD

- Type: SystemV-managed service
- Packages: httpd, httpd-devel
- Daemons: httpd
- Script: httpd
- Ports: 80(http), 443(https)
- Configuration: /etc/httpd/*, /var/www/*
- Related: system-config-httpd, mod_ssl
Apache Configuration

- Main server configuration stored in
  /etc/httpd/conf/httpd.conf
  - controls general web server parameters,
    regular virtual hosts, and access
  - defines filenames and mime-types
- Module configuration files stored in
  /etc/httpd/conf.d/*
- DocumentRoot default
  /var/www/html/
Apache Server Configuration

- Min and Max Spare Servers
- Log file configuration
- Host name lookup
- Modules
- Virtual Hosts
- user/group
Virtual Hosts

NameVirtualHost 192.168.0.100

<VirtualHost 192.168.0.100>
    ServerName virt1.com
    DocumentRoot /path-to-document-root
</VirtualHost>
Apache Namespace Configuration

- Specifying a directory for users' pages:
  
  `UserDir public_html`

- MIME types configuration:
  
  `AddType application/x-httpd-php .phtml`
  `AddType text/html .htm`

- Declaring index files for directories:
  
  `DirectoryIndex index.html default.htm`
Apache Access Configuration

- Apache provides directory- and file-level host-based access control
- Host specifications may include dot notation numerics, network/netmask, and dot notation hostnames and domains
- The `order` statement provides control over "order", but not always in the way one might expect
Using `.htaccess` Files

- Change a directory's configuration:
  - add mime-type definitions
  - allow or deny certain hosts
- Setup user and password databases:
  - `AuthUserFile` directive
  - `htpasswd` command:
    `htpasswd -c /etc/httpd/mypassword bob`
CGI

- CGI programs are restricted to separate directories by ScriptAlias directive:

  `ScriptAlias /cgi-bin/ /<path>/cgi-bin/`

- Apache can greatly speed up CGI programs with loaded modules such as `mod_perl`
Notable Apache Modules

- mod_perl
- mod_php
- mod_speling
Apache Encrypted Web Server

- Apache and SSL: https (port 443)
  - mod_ssl
  - /etc/httpd/conf.d/ssl.conf
- Encryption Configuration:
  - certificate: conf/ssl.crt/server.crt
  - private key: conf/ssl.key/server.key
- Certificate/key generation:
  - /usr/share/ssl/certs/Makefile
  - self-signed cert: make testcert
  - certificate signature request: make certreq
Squid Web Proxy Cache

- Squid supports caching of FTP, HTTP, and other data streams
- Squid will forward SSL requests directly to origin servers or to one other proxy
- Squid includes advanced features including access control lists, cache hierarchies, and HTTP server acceleration
Service Profile: Squid

- Type: SystemV-managed service
- Packages: squid
- Daemons: squid
- Script: squid
- Ports: 3128(squid), (configurable)
- Configuration: /etc/squid/*
End of Unit 5

- Address questions
- Preparation for Lab 5
  - Goals
  - Scenario
  - Deliverables
- Please ask the instructor for assistance when needed
UNIT 6

Security Concerns and Policy
Objectives

- Be able to define security
- Understand Security Components
- Be able to develop a Security Policy
Agenda

- Define Security
- Where are the Vulnerabilities?
- Developing a Security Policy
  - System Activity
  - Human Activity
- Response Strategies
Definition of Security

- Types of security
  - Network (external)
  - Local (internal)
  - Physical
Attacks from the Network

- Exploits and “script kiddie” attacks
- Denial of Service (DoS) attacks
- Distributed Denial of Service (DDoS) attacks
- Hijacking, “Man-in-the-Middle” attacks
- Trojans and “Root Kits”
Principles of Security

- No such thing as 100% protection
- Myth: “We're too small to be at risk”
- Every service is a liability
- Processes running as root are a liability
Security Practices

- Do not run services you do not need, lock down services you do need
- For processes that run as root:
  - "Do I need to be running this?"
  - "Does it need to be running as root?"
  - "Have I applied all relevant security updates"
- Regularly scan for vulnerable files
- Compromising a user often leads to root
  - Educate users!
Diagnostic Utilities

- Port scanners (nmap)
  - Show what services are available on a system
- Packet sniffers (tcpdump, ethereal)
  - Stores and analyzes all network traffic visible to the “sniffing” system
  - Availability is also a liability
Which Services Are Running?

- Use `netstat -taupe` for a list of:
  - active network servers
  - established connections
Remote Service Detection

- nmap scans for active services
  - Advanced scanning options available
  - Offers remote OS detection
  - Scans on small or large subnets
- Used by intruders for the same purpose
- Do not use without written permission of the scanned system's admin!
- Graphical front-end available (nmapfe)
Isolate Vulnerabilities

- Isolate processes
  - Process runs as own user (RHEL default)
  - System users should only have access to service's files and nothing else
- Isolate networks
  - Implement a “firewall”
  - Avoid services that authenticate without encryption
    - telnet, pop, imap, authenticated ftp
    - alternatives: ssh, apop, imaps, sftp, anonymous ftp
- Keep systems 'up2date'
Security Policy: the System

- Managing system activities
- Regular system monitoring
  - Log to an external server in case of compromise
  - Monitor logs with logwatch
  - Monitor bandwidth usage inbound and outbound
- Regular backups of system data
Security Policy: the People

- Managing human activities
  - includes Security Policy maintenance
- Who is in charge of what?
- Who makes final decision about false alarms?
- When is law-enforcement notified?
Response Strategies

- Assume suspected system is untrustworthy
  - Do not run programs from the suspected system
  - Boot from trusted media to verify breach
  - Analyze logs of remote logger and "local" logs
  - Check file integrity against read-only backup of rpm database
- Make an image of the machine for further analysis/evidence-gathering
- Wipe the machine, re-install and restore from backup
Additional Resources

- Security Education
  - Red Hat Security Guide (on Documentation CD and at redhat.com's docs section)

- Keeping up with vulnerabilities
  - Red Hat Network
  - Red Hat Errata
  - Bugtraq mailing list

- Keeping track of "the other side"
End of Unit 6

- Address questions
- Preparation for Lab 6
  - Goals
  - Scenario
  - Deliverables
- Please ask the instructor for assistance when needed
UNIT 7

Authentication Services
Objectives

- Understand the basics of authentication
- Understand the roles of NSS and PAM
- Use NIS to centrally manage user information and authentication through NSS and PAM
Agenda

- User Information and NSS
- Authentication and PAM
- Network Information Service (NIS)
  - Configuring NIS master servers, slave servers, and clients
User Authentication

- Two types of information must always be provided for each user account
  - **Account information**: UID number, default shell, home directory, group memberships, and so on
  - **Authentication**: a way to tell that the password provided on login for an account is correct
Account Information

- Name services accessed through library functions map names to information
- Originally, name service was provided only by local files like /etc/passwd
- Adding support for new name services (such as NIS) required rewriting libc
Name Service Switch

- NSS allows new name services to be added without rewriting libc
  - Uses /lib/libnss_service.so files
- /etc/nsswitch.conf controls which name services to check in what order
  - passwd: files nis ldap
getent

- getent database
  - Lists all objects stored in the specified database
  - getent services
- getent database name
  - Looks up the information stored in the specified database for a particular name
  - getent passwd smith
Authentication

- Applications traditionally authenticated passwords by using *libc* functions
  - Hashes password provided on login
  - Compare to hashed password in NSS
  - If the hashes match, authentication passes
- Applications had to be rewritten to change how they authenticated users
PAM

- Pluggable Authentication Modules
- Application calls \texttt{libpam} functions to authenticate and authorize users
- \texttt{libpam} handles checks based on the application's PAM configuration file
  - May include NSS checks through \texttt{libc}
- Shared, dynamically configurable code
PAM Operation

- `/lib/security` PAM modules
  - Each module performs a pass or fail test
  - Files in `/etc/security` may affect how some modules perform their tests
- `/etc/pam.d` PAM configuration
  - Service files determine how and when modules are used by particular programs
/etc/pam.d Files: Tests

- Tests are organized into four groups:
  - **auth** authenticates that the user *is* the user
  - **account** authorizes the account may be used
  - **password** controls password changes
  - **session** opens, closes, and logs the session

- Each group is called as needed and provides a separate result to the service
/etc/pam.d/ Files: Control Values

- Control values determine how each test affects group's overall result
  - **required** must pass, keep testing even if fails
  - **requisite** as **required**, except stop testing on fail
  - **sufficient** if passing so far, return success now
    - if fails, ignore test and keep checking
  - **optional** whether test passes or fails is irrelevant
Example `/etc/pam.d/ File`

<table>
<thead>
<tr>
<th>Type</th>
<th>Requirement</th>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>auth</td>
<td>requisite</td>
<td><code>pam_securetty.so</code></td>
</tr>
<tr>
<td>auth</td>
<td>sufficient</td>
<td><code>pam_unix.so</code></td>
</tr>
<tr>
<td></td>
<td>required</td>
<td><code>likeauth</code></td>
</tr>
<tr>
<td>account</td>
<td>required</td>
<td><code>pam_unix.so</code></td>
</tr>
<tr>
<td>password</td>
<td>required</td>
<td><code>pam_cracklib.so</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>retry=3</code></td>
</tr>
<tr>
<td>password</td>
<td>sufficient</td>
<td><code>pam_unix.so</code></td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>use_authtok</code></td>
</tr>
<tr>
<td>password</td>
<td>required</td>
<td><code>pam_deny.so</code></td>
</tr>
<tr>
<td>session</td>
<td>required</td>
<td><code>pam_unix.so</code></td>
</tr>
<tr>
<td>session</td>
<td>required</td>
<td><code>pam_limits.so</code></td>
</tr>
<tr>
<td>session</td>
<td>optional</td>
<td><code>pam_console.so</code></td>
</tr>
</tbody>
</table>
pam_stack

- Special module that bases result on the tests in another /etc/pam.d service file
- system-auth is widely used
  - Contains standard authentication tests
  - Shared by many applications on the system
  - Allows easy, consistent management of standard system authentication
pam_unix

- Module for NSS-based authentication
  - **auth** gets hashed password from NSS and compares it to hash of entered password
  - **account** checks for password expiration
  - **password** handles password changes to local files or NIS
  - **session** records login and logout to logs
Network Authentication

- Central password management
  - pam_krb5 (Kerberos V tickets)
  - pam_ldap (LDAP binds)
  - pam_smb_auth (old SMB authentication)
  - pam_winbind (SMB through winbinddd)
- Some services use NSS/pam_unix
  - NIS, Hesiod, some LDAP configurations
auth Modules

- pam_securetty fails if logging in as root from a terminal not in 
  /etc/securetty
- pam_nologin fails if the user is not root and the file /etc/nologin exists
- pam_listfile checks a characteristic of the authentication against a list in a file
  - A list of accounts can be allowed or denied
Password Security

- pam_unix MD5 password hashes
  - Makes password hashes harder to crack
- pam_unix shadow passwords
  - Makes password hashes visible only to root
  - Makes password aging available
- Other modules may support password aging mechanisms
Password Policy

- Password history
  - `pam_unix` with `remember=N` argument
- Password strength
  - `pam_cracklib`
  - `pam_passwdqc`
- Failed login monitoring
  - `pam_tally`
session Modules

- pam_limits enforces resource limits
  - Uses /etc/security/limits.conf
- pam_console sets permissions on local devices for console users
  - Can be used as an auth module as well
- pam_selinux helps set SELinux context
Utilities and Authentication

- Local admin tools need authentication
  - su, reboot, system-config-* etc.
- pam_rootok passes if running as root
- pam_timestamp for sudo-like behavior
- pam_xauth forwards xauth cookies
PAM Troubleshooting

- Check the system logs
  - /var/log/messages
  - /var/log/secure
- PAM mistakes can lock out the root user
  - Keep a root shell open when testing PAM
  - Single-user mode bypasses PAM
  - Boot the system using a rescue disc
NIS Overview

- Simple directory service for system and account information
- All NIS servers and clients are members of a named NIS domain
  - Single master server, multiple slave servers
- Minimal network security
- Support for NIS version 1 and 2
Service Profile: NIS

- Type: System V-managed services
- Packages: yperv
- Daemons: yperv, rpc.yppasswdd, rpc.ypxfrd
- Scripts: yperv, yppasswdd, ypxfrd
- Ports: Dynamically assigned by portmap
- Configuration: /var/yp/*, /etc/ypserv.conf
  (/etc/yp.conf for ypbind)
- Related: portmap, ypbind, yp-tools
NIS Server Configuration

- Install the *portmap* and *ypserv* RPMs
- Set the NIS domain name
  - Run `nisdomainname mydomain`
  - In `/etc/sysconfig/network` insert the line: `NISDOMAIN=mydomain`
- In `/var/yp/securenets`, specify the networks that may use your server
- Start *ypserv*
Configuring a Master Server

- To share only user, group, and host name information, edit `/var/yp/Makefile`
  ```plaintext
  all: passwd group hosts netid
  ```
- Build the NIS maps from local files by using the makefile:
  ```plaintext
  /usr/lib/yp/ypinit -m
  ```
- Start `yppasswdd` to allow password updates
Configuring a Slave Server

- Include the names of all slave servers in the master's `/var/yp/ypservers` file
- On the slave, transfer the initial NIS maps from the master server:
  
  ```bash
  /usr/lib/yp/ypinit -s master
  ```
- To rebuild and push NIS maps from master to slave, on the master run
  
  ```bash
  cd /var/yp; make
  ```
NIS Client Configuration

- Must install *ypbind* and *portmap* RPMs
- `system-config-authentication`
  - Enable NIS to provide "User Information"
  - Specify NIS server and NIS domain name
  - Keep default "Authentication" (using NSS)
- What does this actually do?
  - Modifies four configuration files
NIS Troubleshooting

- Is the default firewall still turned on?
- Are services running and registered with portmap?
  - `rpcinfo hostname`
- Use `ypwhich` to verify which server a client is bound to, if any
- Use `ypcat` and `getent` to verify that NIS data is available
End of Unit 7

- Address questions
- Preparation for Lab 7
  - Goals
  - Scenario
  - Deliverables
- Please ask the instructor for assistance when needed
UNIT 8

System Monitoring
Objectives

- Learn to identify file statistics
- Ensure filesystem integrity
- Understand system log configuration
- Learn log file analysis
- Understand process monitoring
Agenda

- File system analysis with `find`
- Common log files
- Configuration of `syslogd` and `klogd`
- Process monitoring and accounting
Introduction to System Monitoring

- Security breaches can be detected with regular system monitoring
- System monitoring includes:
  - File system monitoring
  - Log file analysis
  - Process monitoring
File System Analysis

- Regular file system monitoring can prevent:
  - Exhausing system resources
  - Security breaches due to poor access controls
- File system monitoring should include:
  - Data integrity scans
  - Investigating suspect files
- Utilities: `df`, `du`, `logwatch`
Set User and Group ID Permissions

- Programs owned by root with SUID or SGID permissions can be dangerous
- Security policy should include monitoring SUID programs
- Prevent SUID and SGID permissions on filesystems with nosuid mount option
Typical Problematic Permissions

- Files without known owners may indicate unauthorized access:
  - Locate with: `find / \( -nouser -o -nogroup \)`

- Files/Directories with "other" write permission (o+w) may indicate a problem:
  - Locate with: `find / -type f -perm -2`
  - Locate with: `find / -type d -perm -2`
EXT2/3 Filesystem Attributes

- EXT2/3 supports several special attributes that affect the behavior of files
- Show attributes with `lsattr`
- Set attributes: `chattr <file>`
- Some attributes not currently supported by the Linux kernel
- Some attributes unavailable for users
System Log Files

• Why monitor log files?
• Which logs to monitor?
• Logging Services:
  • Many daemons send messages to `syslogd`
  • Kernel messages are handled by `klogd`
syslogd and klogd
Configuration

- syslogd and klogd are configured in
  /etc/syslog.conf
- Syntax:
  \texttt{facility.priority \ log\_location}
- Example:
  mail.info /dev/tty8
Advanced syslogd Configuration

- **Operators**
  - `facility.priority`
    - facility messages with equal or higher priority
  - `facility.=priority`
    - facility messages with exact priority
  - `facility.=!priority`
    - facility messages except those with priority
  - `facility1,facility2.priority`
    - priority messages from `facility1` and `facility2`
  - `*.priority`
    - messages with equal or higher priority, regardless of facility

- **Special Targets**
  - Comma-separated list of users
  - Remote machine (`@hostname`)
Log File Analysis

- Should be performed on a regular basis
- `logwatch` can be installed to run by `crond` every hour to report possible issues
- When looking for anomalies, `logwatch` uses negative lists
  - Discard everything normal
  - Analyze the rest
Monitoring Processes

- Monitor processes to determine:
  - Cause of decreased performance
  - If suspicious processes are executing
- Monitoring utilities
  - top
  - gnome-system-monitor
  - sar
Process Monitoring Utilities

- **top**
  - view processor activity in real-time
  - interactively kill or renice processes
  - watch system statistics update through time, either in units or cumulatively

- GUI system monitoring tools:
  - `gnome-system-monitor`: GNOME process, CPU, and memory monitor
  - `kpm`: KDE version of `top`
System Activity Reporting

- Frequent reports, over time
  - `cron` spawns `sa1` and `sa2`
  - `sar` reads and generates “human friendly” logs
- Commonly used for performance tuning
  - more accurate statistics
    - binary “database” collection method
    - regular intervals
  - Evidence of pattern establishes “normal” activity
Limiting Processes

- Use PAM to set resource limits for processes:
  - `pam_access.so` can be used to limit access by account and location
  - `pam_time.so` can be used to limit access by day and time
  - `pam_limits.so` can be used to limit resources available to process
Process Accounting Tools

- `history` shell built-in command listing
- `last` displays user's login history
- Process accounting
  - provided by `psacct` package
  - Activated by `accton`
  - Potential performance impact
  - `ac` displays user connect times from 
    `/var/log/wtmp`
End of Unit 8

- Questions and Answers
- Preparation for Lab 8
  - Goals
  - Scenario
  - Deliverables
- Please ask the instructor for assistance when needed
UNIT 9

Securing Networks
Objectives

- Explain packet filtering architecture
- Explain primary filtering command syntax
- Explain Network Address Translation
- Provide examples
- Show how to maintain configuration
Agenda

- Introduce packet filtering architecture
- Describe Netfilter configuration
- Demonstrate rules by example
- Describe NAT
- Making rules persistent
IP Forwarding

- Effectively makes your Linux box a router
- Usually used with two network interfaces
- Can be used with dynamic routing and firewall services
- Configure by setting `net.ipv4.ip_forward` kernel variable
  - `/etc/sysctl.conf`
  - `/proc/sys/net/ipv4/ip_forward`
    - (not persistent)
Routing

- Routers transport packets between different networks
- Each machine needs a default gateway to reach machines outside the local network
- Additional routes can be set using the `route` command
- Permanent entries can be placed in `/etc/sysconfig/static-routes`
- Dynamic routing protocols are used for greater flexibility
Netfilter Overview

• Packet filter architecture for 2.4 kernel
• Filtering in the kernel: no daemon
• Assert policies at layers 2, 3 & 4 of the OSI Reference Model
• Only limited capacity to inspect packets
• Consists of netfilter modules in kernel, and the iptables user-space software
• Supercedes ipchains
• See http://www.netfilter.org/
Netfilter Architecture

Table
- Chain
  - Policy
  - Rule
    - Match Specification
      - Base
      - Extension
    - Target
      - Base
      - Extension
Netfilter Tables and Chains

- Built-in Chains:

<table>
<thead>
<tr>
<th>Filtering point</th>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>filter</td>
</tr>
<tr>
<td>INPUT</td>
<td>X</td>
</tr>
<tr>
<td>FORWARD</td>
<td>X</td>
</tr>
<tr>
<td>OUTPUT</td>
<td>X</td>
</tr>
<tr>
<td>PREROUTING</td>
<td>X</td>
</tr>
<tr>
<td>POSTROUTING</td>
<td>X</td>
</tr>
</tbody>
</table>
Netfilter Packet Flow

- PREROUTING
- FORWARD
- POSTROUTING

Kernel

- Routing Decision
- INPUT
- OUTPUT

= Inspection point

Local process

RedNet
Rule Matching

- Rules in ordered list
- Packets tested against each rule in turn
- On first match, the target is evaluated: usually exits the chain
- Rule may specify multiple criteria for match
- Every criterion in a specification must be met for the rule to match (logical 'and')
- Chain policy applies if no match
Rule Targets

- Built-in targets: DROP, ACCEPT
- Extension targets: LOG, REJECT, custom chain
  - REJECT sends a notice returned to sender
  - LOG connects to syslogger kernel facility
  - LOG match does not exit the chain
- Target is optional, but no more than one per rule and defaults to the chain policy if absent
Simple Example

- An **INPUT** rule for the filter table:

  ```
  iptables -t filter -A INPUT -s 192.168.0.1 -j DROP
  ```

  Where the rule is checked (**CHAIN**)

  The rule

  The match part of the rule

  The target part of the rule
Basic Chain Operations

- Append a rule to the chain (-A)
- Insert a rule to the chain (-I)
  - -I CHAIN (inserts as the first rule)
  - -I CHAIN 3 (inserts as rule 3)
- Delete an individual rule (-D)
  - -D CHAIN 3 (deletes rule 3 of the chain)
  - -D CHAIN RULE (deletes rule explicitly)
- Flush all rules of a chain (-F)
- List rules in a chain or table (-L or -vL)
Additional Chain Operations

- Assign chain policy (-P CHAIN TARGET)
  - ACCEPT (default, a built-in target)
  - DROP (a built-in target)
  - REJECT (not permitted, an extension target)
- Zero byte and packet counters (-z [CHAIN])
  - Useful for monitoring chain statistics
- Manage custom chains (-N, -X)
  - -N Your_Chain-Name (adds chain)
  - -X Your_Chain-Name (deletes chain)
Rules: General Considerations

- Defaults to mostly open (ACCEPT). Mostly closed is more appropriate
  - `iptables -P INPUT DROP` or
  - `iptables -A INPUT -j DROP`

- Criteria also apply to loopback interface
  - The example rules above will have the side effect of blocking localhost!

- Rules, like routes, are loaded in memory and must be saved to a file for persistence across reboots
Match Criteria (filter table)

- A rule can match many characteristics of a packet:
  - Incoming interface (-i)
  - Outgoing interface (-o)
  - Layer 4 protocol (-p)
  - Source IP address (-s)
  - Destination IP address (-d)
- The above are base capability
TCP Match Extensions (filter table)

- Additional criteria can be used as the basis for packet matching:
  - Protocol -p
  - Source port --sport
  - Destination port --dport
  - TCP flags --tcp-flags
  - Initial connection request --syn
UDP and ICMP Match Extensions

- Match source and destination ports with UDP extensions:
  
  ```
  iptables -A INPUT -m udp -p udp --sport 123 -j DROP
  ```

- Match ICMP types:
  
  ```
  -p icmp --icmp-type destination-unreachable
  ```
Match Arguments

- Matches may be made by:
  - IP address, or host name
  - Port number, or service name
  - Arguments may be negated with `!`
- Inclusive port range may be specified '0:1023'
- Masks may use VLSN or CIDR notation
Chain Criteria

- Outgoing interface (-o) may only be used in the FORWARD, OUTPUT and POSTROUTING chains
- Incoming interface (-i) may only be used in FORWARD, INPUT and PREROUTING chains
- Owner match (- - * - owner) may only be used in the OUTPUT chain
Directional Filtering

Scenario: HostA to HostB

CASE 1
No filtering

CASE 2
iptables -A INPUT -s HostA -j DROP

Refused
Directional Filtering II

Scenario: HostB to HostA

CASE 3

iptables -A INPUT -s HostA -j DROP

CASE 4

iptables -A INPUT -p tcp -s HostA -j DROP
Connection Tracking

- Provides inspection of packet’s “state”
  - a packet can be tested in a specific context
- Simplifies rule design
  - without connection tracking, rules are usually in pairs (inbound & outbound)
- Implemented in state match extension
- Recognized states: NEW, ESTABLISHED, RELATED, INVALID
- Requires much more memory
Connection Tracking Example

- One rule to permit established connections:
  ```
  iptables -A INPUT -m state \n  --state ESTABLISHED,RELATED -j ACCEPT
  ```

- Many rules; one for each permitted service:
  ```
  iptables -A INPUT -m state --state NEW \n  -p tcp --dport 25 -j ACCEPT
  ```

- Lastly, one rule to block all others inbound:
  ```
  iptables -A INPUT -m state --state NEW \n  -j DROP
  ```
Network Address Translation (NAT)

- Translates one IP address into another (inbound and/or outbound)
- Allows "hiding" internal IP addresses behind a single public IP
- Rules set within the `nat` table
- Network Address Translation types:
  - Destination NAT (DNAT)
    Set in the PREROUTING chain where filtering uses translated address
  - Source NAT (SNAT, MASQUERADE)
    Set in the POSTROUTING chain where filtering never uses translated address
SNAT Examples

- MASQUERADE
  ```bash
  iptables -t nat -A POSTROUTING \
  -o eth0 -j MASQUERADE
  ```
- SNAT
  ```bash
  iptables -t nat -A POSTROUTING \
  -j SNAT --to-source 1.2.3.45
  ```
DNAT Examples

- **INBOUND**
  
  `iptables -t nat -A PREROUTING \
  -p tcp --dport 80 -j DNAT \ 
  --to-dest 192.168.0.20`

- **OUTBOUND** (with port redirection)
  
  `iptables -t nat -A OUTPUT \
  -p tcp -j DNAT \ 
  --to-dest 192.168.0.200:3128`
Rules Persistence

- `iptables` is not a daemon, but loads rules into memory and exits.
- Rules are not persistent across reboot:
  - `service iptables save` will store rules to `/etc/sysconfig/iptables`
  - System V management may be used, and is run before networking is configured.
- Conflicts with `ipchains`
Example

- Sample /etc/sysconfig/iptables

```
*filter
:INPUT DROP [573:46163]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [641:68532]
-A INPUT -i lo   -j ACCEPT
-A INPUT -p tcp -m tcp --dport 143    -j ACCEPT
-A INPUT -p tcp -m tcp --dport 22    -j ACCEPT
-A INPUT -p tcp -m tcp --dport 25    -s 123.123.123.1 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 53    -j ACCEPT
-A INPUT -p udp -m udp --dport 53    -j ACCEPT
-A INPUT -p udp -m udp --dport 123    -s 123.123.123.1 -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -p tcp -m tcp --dport 113    -j REJECT --reject-with tcp-reset
COMMIT
```
End of Unit 9

- Address questions
- Preparation for Lab 9
  - Goals
  - Sequences
  - Deliverables
- Please ask the instructor for assistance when needed
UNIT 10

Securing Services
Objectives

- Analyze service activity
- Implement security policy
  - within the service
  - with tcp_wrappers
  - with xinetd
Agenda

- Inspect local network services
- Configure tcp_wrappers
- Secure xinetd managed services
System V Startup Control

- Determine which services are running from SysV startup scripts or xinetd
- `chkconfig --list`
  - shows which services should run.
  - cannot be used to get a list of running services
- Disable all unneeded services
Securing the Service

- Service-specific configuration
  - Daemons like `httpd` provide special security mechanisms

- General configuration
  - All programs linked with `libwrap.so` use common configuration files
  - Because `xinetd` is linked with `libwrap.so`, its services are affected
  - Checks for host and/or remote user name
tcp_wrappers Configuration

- Three stages of access checking
  - Is access explicitly permitted?
  - Otherwise, is access explicitly denied?
  - Otherwise, by default, permit access!

- Configuration stored in two files:
  - Permissions in /etc/hosts.allow
  - Denials in /etc/hosts.deny

- Basic syntax:
  \[
  \text{daemon\_list: client\_list [[:options]]}
  \]
Daemon Specification

- Daemon name:
  - Applications pass name of their executable
  - Multiple services can be specified
  - Use wildcard ALL to match all services
  - Limitations exist for certain daemons

- Advanced Syntax:
  
  `daemon@host: client_list ..`
Client Specification

- Host specification
  - by IP address (192.168.0.1, 10.0.0.0.)
  - by name (www.redhat.com, example.com)
  - by netmask (192.168.0.0/255.255.255.0)
  - by network name
Advanced Syntax

• Wildcards
  • ALL, LOCAL
  • KNOWN, UNKNOWN, PARANOID

• EXCEPT operator
  • Can be used for client and service list
  • Can be nested
Options

• Syntax
  `daemon_list: client_list [:option1 :option2 ..]`

• Spawn
  • Can be used to start additional programs
  • Example: `in.telnetd: ALL : spawn echo "login attempt from %c to %s" | mail -s warning root`
  • Special expansions are available (%c, %s)

• DENY
  • Can be used as an option in `hosts.allow`
  • Example: `ALL: ALL: DENY`
Example

Consider the following example for the machine 192.168.0.254 on a class C network:

```
# /etc/hosts.allow
vsftpd: 192.168.0.
in.telnetd, portmap: 192.168.0.8

# /etc/hosts.deny
ALL: .cracker.org EXCEPT trusted.cracker.org
vsftpd, portmap: ALL
pop3d: 192.168.0.0. EXCEPT 192.168.0.4
```
Securing `xinetd`-managed services

- `xinetd` provides its own set of access control functions
  - host-based
  - time-based
- `tcp_wrappers` is still used
  - `xinetd` is compiled with `libwrap` support
  - If `libwrap.so` allows the connection, then `xinetd` security configuration is evaluated
xinetd Access Control

• Syntax
  • Allow with `only_from = host_pattern`
  • Deny with `no_access = host_pattern`
  • The most exact specification is authoritative

• Example
  ```
  only_from = 192.168.0.0/24
  no_access = 192.168.0.1
  ```
Host Patterns

- Host masks for \texttt{xinetd} may be:
  - numeric address (192.168.1.0)
    - rightmost zeros are treated as wildcards
  - network name (from /etc/networks)
  - hostname or domain (.domain.com)
  - IP address/netmask range (192.168.0.0/24)
Advanced Security Options

- Access by time
  - Syntax: `access_times = 9:00-18:00`
  - `pam_time.so` for more advanced scenarios

- Number of simultaneous connections
  - Syntax: `per_source = 2`
  - Cannot exceed maximum instances
End of Unit 10

- Address questions
- Preparation for Lab 10
  - Goals
  - Sequences
  - Deliverables
- Please ask the instructor for assistance when needed
UNIT 11

Securing Data
Objectives

- Understand fundamental encryption protocols
- Describe encryption implementations in Red Hat Enterprise Linux
- Configure encryption services for common networking protocols
Agenda

- Introduction to data encryption
- Contrast encryption methods
- Red Hat encryption implementations
  - OpenSSH
  - RPM
The Need For Encryption

- Susceptibility of unencrypted traffic
  - password/data sniffing
  - data manipulation
  - authentication manipulation
  - equivalent to mailing on postcards
- Insecure traditional protocols
  - telnet, ftp, pop3, etc.: insecure passwords
  - sendmail, nfs, etc.: insecure information
  - rsh, rcp, etc.: insecure authentication
Cryptographic Building Blocks

- Random Numbers
- One Way Hashes
- Symmetric Algorithms
- Asymmetric (Public Key) Algorithms
- Public Key Infrastructures
- Digital Certificates
- Implementations:
  - openssl, gpg
Random Numbers

- Pseudo-Random Numbers and Entropy Sources
  - keyboard and mouse events
  - block device interrupts
- Kernel provides sources
  - `/dev/random`:
    - best source
    - blocks when entropy pool exhausted
  - `/dev/urandom`:
    - draws from entropy pool until depleted
    - falls back to pseudorandom generators
- `openssl rand [-randbase64]`
One-Way Hashes

- Arbitrary data reduced to small "fingerprint"
  - arbitrary length input
  - fixed length output
  - If data changed, fingerprint changes ("collision free")
  - data cannot be regenerated from fingerprint ("one way")

- Common Algorithms
  - md2, md5, mdc2, rmd160, sha, sha1

- Common Utilities
  - md5sum [ --check ]
  - openssl, gpg
  - rpm -V
Symmetric Encryption

- Based upon a single Key
  - used to both encrypt and decrypt
- Common Algorithms
  - DES, 3DES, Blowfish, RC2, RC4, RC5, IDEA, CAST5
- Common Utilities
  - passwd (modified DES)
  - gpg (3DES, CAST5, Blowfish)
  - openssl
Asymmetric Encryption I

- Based upon public/private key pair
  - What one key encrypts, the other decrypts
- Protocol I: Encryption without key synchronization
  - Recipient
    - generate public/private key pair: P and S
    - publish public key P, guard private key S
  - Sender
    - encrypts message M with recipient public key
    - send P(M) to recipient
  - Recipient
    - decrypts with secret key to recover: \( M = S(P(M)) \)
Asymmetric Encryption II

- Protocol II: Digital Signatures
  - Sender
    - generate public/private key pair: P and S
    - publish public key P, guard private key S
    - encrypt message M with private key S
    - send recipient S(M)
  - Recipient
    - decrypt with sender’s public key to recover \( M = P(S(M)) \)
- Combined Signature and Encryption
- Detached Signatures
Public Key Infrastructures

- Asymmetric encryption depends on public key integrity
- Two approaches discourage rogue public keys:
  - Publishing Key fingerprints
  - Public Key Infrastructure (PKI)
    - Distributed web of trust
    - Hierarchical Certificate Authorities
      - Digital Certificates
Digital Certificates

- Certificate Authorities
- Digital Certificate
  - Owner: Public Key and Identity
  - Issuer: Detached Signature and Identity
  - Period of Validity
- Types
  - Certificate Authority Certificates
  - Server Certificates
  - Self-Signed certificates
Generating Digital Certificates

- X.509 Certificate Format
- Generate a public/private key pair
- Define Identity
- Two Options:
  - Use Certificate Authority
    - generate signature request (csr)
    - send csr to CA
    - receive signature from CA
  - Self Signed Certificates
    - sign your own public key
OpenSSH Overview

- OpenSSH replaces common, insecure network communication applications
- Provides user and token-based authentication
- Capable of tunneling insecure protocols through port forwarding
- System default configuration (client and server) resides in `/etc/ssh`
OpenSSH Authentication

- The `sshd` daemon can utilize several different authentication methods
  - password (sent securely)
  - RSA and DSA keys
  - Kerberos
  - s/key and SecureID
  - host authentication using system key pairs
The OpenSSH Server

- Provides greater data security between networked systems
  - private/public key cryptography
  - compatible with earlier restricted-use commercial versions of SSH
- Implements host-based security through libwrap.so
Service Profile: SSH

- Type: System V-managed service
- Packages: openssh{-clients,-server}
- Daemons: sshd
- Scripts: sshd
- Ports: 22
- Configuration: /etc/ssh/*, $HOME/ssh
- Related: openssl, openssh-askpass, openssh-askpass-gnome
OpenSSH Server Configuration

- SSHD configuration file
  - /etc/ssh/sshd_config

- Options to consider
  - Protocol
  - ListenAddress
  - PermitRootLogin
  - Banner
The OpenSSH Client

- Secure shell sessions
  - `ssh hostname`
  - `ssh user@hostname`
  - `ssh hostname remote-command`
- Secure remote copy files and directories
  - `scp file user@host:remote-dir`
  - `scp -r user@host:remote-dir localdir`
- Secure ftp provided by `sshd`
  - `sftp host`
  - `sftp -C user@host`
Protecting Your Keys

- **ssh-agent**
  - manages key passphrases
- **ssh-add**
  - collects key passphrases
Applications: RPM

- Two implementations of file integrity
- Installed Files
  - MD5 One-way hash
  - `rpm --verify package_name` (or `-v`)
- Distributed Package Files
  - GPG Public Key Signature
  - `RPM-GPG-KEY`
  - `rpm --checksig package_file_name`
End of Unit 11

- Address questions
- Preparation for Lab 11
  - Goals
  - Scenario
  - Deliverables
- Please ask the instructor for assistance when needed