Linux Security

Arun Eapen

RHCA,CISSP,CSPFA

SeniorTechnical Consultant & Certification Poobah

Red Hat India

Agenda

Security Aspects & Updates

Hardening

SELinux

Intrusion Detection & Prevention

Security Tools & Best Practices

redhat

“ The only truly secure system is one that is powered off, cast in a block of concrete and sealed in a lead-lined room with armed guards - and even then I have my doubts. ”

Eugene H. Spafford, director of the Purdue Center for Education and Research in Information Assurance and Security.

Aspects of the Security Problem

- **Security is a *technical issue*—**New technology can make systems more impervious to common attacks by default. It can also provide organizations with more control over who can access which resources.
- **Security is a *human issue*—**Compliance and failure to keep systems up to date are a major source of security vulnerabilities. Administrators need help understanding the state of their systems and managing the complexity of securing their network.
- **Security is an *economic issue*—**In an interconnected environment, you are only as secure as your neighbor. Access to strong security must be affordable enough to be ubiquitous

Linux Lowers the Cost of Security

- Cost should not be a prohibitive factor in providing a secure computing environment.
- Traditionally, proprietary “trusted operating systems” have been priced out of reach for most customers. Red Hat is making secure operating system architectures an affordable, mainstream solution.
- As affordable and secure solutions scale, the entire Internet becomes more safe and reliable.

Red Hat Enterprise Linux
The most certified operating system available

- Through its history, Red Hat Enterprise Linux ® has passed the Common Criteria process 13 times on 4 different hardware platforms.
- RHEL 5 has even received Common Criteria certification at Enterprise Assurance Level 4 (EAL4+) under the:
  * Controlled Access Protection Profile (CAPP)
  * Label Security Protection Profile (LSPP)
  * Role-Based Access Control Protection Profile (RBACPP)
- Red Hat Enterprise Linux provides a level of security and a feature set that was previously unheard-of from a mainstream operating system

Linux Security - Transparent

- Open source has made Linux the superior solution for secure operating systems.
- No more “security by obscurity”.
- The “many eyes” principle: everyone can view the code and suggest improvements.
- Customers do not have to rely solely on a single software vendor to acknowledge and address vulnerabilities.
- Software updates can be examined to enable IT departments to easily assess the impact, perform QA, and quickly distribute.
- Modular (package-based) design makes it easier to identify and resolve issues.

Linux Security - Vigilant

- RHEL Security Response Team provides a consistent point of contact for security issues.
- Dedicated Engineering team monitoring and responding to threats.
- Defined processes for addressing, prioritizing, and fixing security vulnerabilities in a timely manner.
- RHN provides enterprise-class delivery of fixes
  - Packages are tested and digitally signed by Red Hat
  - Customized notification
  - Built-in processes to facilitate internal testing
  - Powerful management features
  - Single update system across entire platform

Linux Security - Innovative

- Red Hat is the leader in Linux security.
- Developing technology that brings advanced security concepts to a mainstream operating system.
- Code and Protocol auditing
- Innovation that delivers real security benefits to customers.
  - SELinux
  - Exec-shield
  - Position Independent Executables
  - NX support

Linux Security - Inclusive

- Red Hat Partners can provide security solutions in many areas, including access management, anti-virus, monitoring, encryption, and authentication.

redhat.

Linux Security

Security Updates

Linux Worms

- Security issues rated with a “Critical” impact are those that can be exploited remotely without user interaction
  - Worms take advantage of Critical vulnerabilities in order to spread with huge consequences
- Linux is not immune from Critical vulnerabilities
  - But few are actively exploited by worms
  - Linux worms to date have targeted known, fixed, vulnerabilities

Name	*Date worm released*	*Date flaw Fixed*
Slapper	Sep 2002	July 2002
Adore	Apr 2001	Jan 2001
Lion	Mar 2001	Jan 2001
Ramen Noodle	Jan 2001	Sep 2000

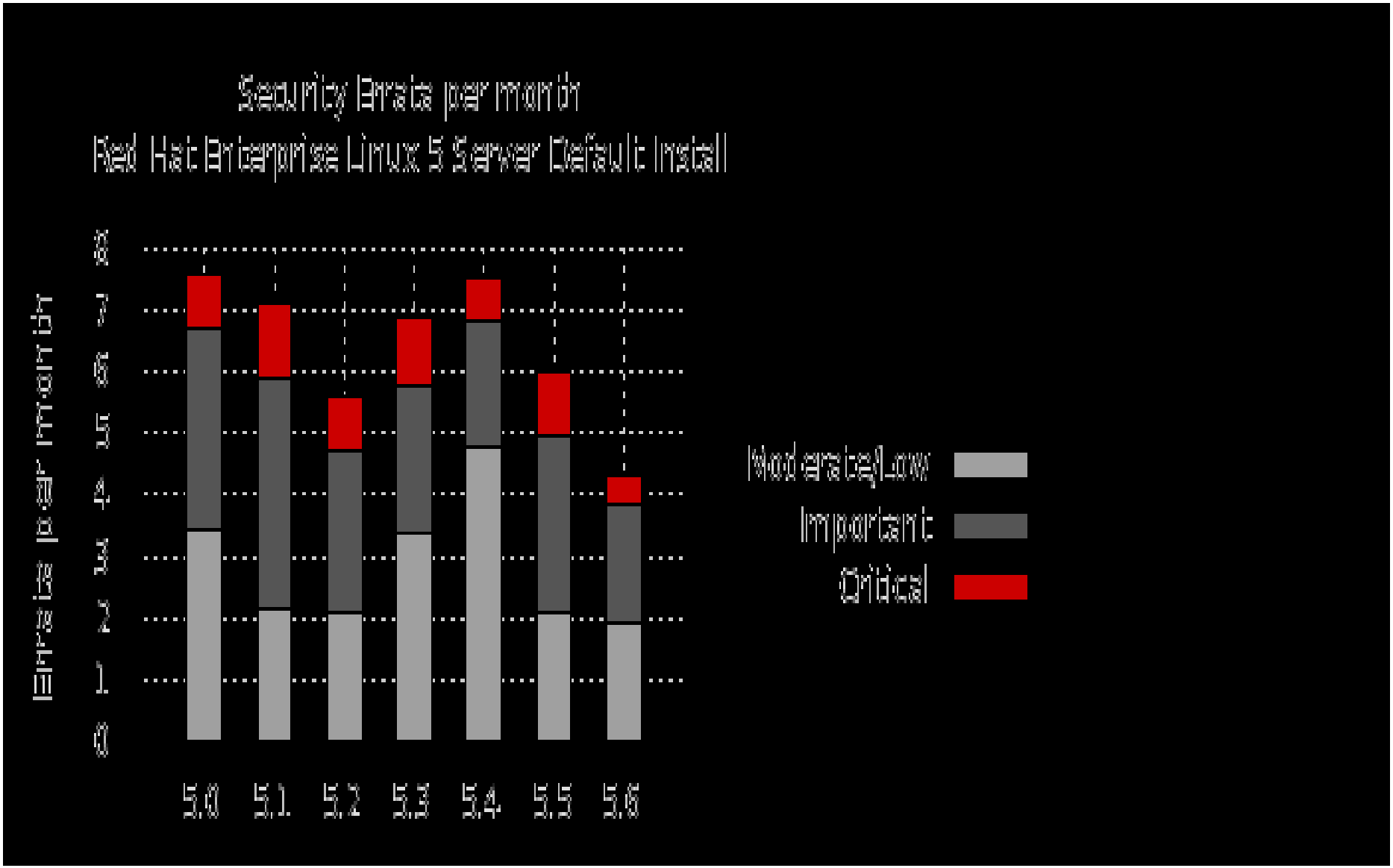
Reducing the threat of Linux worms

- In the past users who are vulnerable are ones that didn't upgrade their systems in the 1-2 month window
- Users don't upgrade for a number of reasons
  - Machines are forgotten, ignored, or lost
  - “Cry wolf” with too many vulnerabilities all saying “Urgent”
  - Incorrect or misleading information on the flaws
  - Users have too many diverse systems to manage
  - Too hard to upgrade
  - Can't figure out which updates to apply
  - Multiple update services for different parts of their OS
- It's our job to solve these problems and help protect users
- If we can't get users to upgrade can we find ways of reducing the impact of security issues, removing some “Critical” vulnerabilities?

Red Hat's Security Commitment

- Dedicated Security Response Team
- Effective communication
- Expertise dealing with Open Source projects and authors
- Easy updating with Red Hat Network
- Powerful management capabilities with Red Hat Network
- Long term support for security updates
- Consistent vulnerability naming

Flaws Affecting Enterprise Linux v.5



redhat

Buffer Overflow Protection

- Most common vulnerability used in exploits across all platforms.
  - Sends carefully-crafted data into memory where it overwrites return address information in buffer, which then allows the execution of hostile code.
- Red Hat provides support for NX (No eXecute) hardware from AMD and Intel.
  - Separates read and execute permissions in memory so hostile code will not run.
- Red Hat contributions to the Linux kernel such as Exec-Shield and PIE (Position Independent Executables) also prevent buffer overflow exploits.
  - Exec-Shield is similar to NX, but works on existing hardware.
  - PIE allows applications to load into random memory locations, removing predictability.

redhat.

Linux Security

Hardening System Security

What are you protecting?

There are essentially four things to protect against

1. An intruder reading your confidential data
2. An intruder changing your data
3. An intruder removing your data
4. Denial of Service
5. An intruder launching other attacks from your site

Who are your enemies?

1. Crackers
2. Disgruntled current employees
3. Disgruntled Former employees
4. Competitors
5. Spies
6. Criminals
7. Extremists

Installation Issues

- BIOS password
- partitions
- Boot Loader Password
- Firewall Selection
- File Integrity Verifiers
- up2date
- Bastille Linux hardening script - http://bastille-linux.org
- Enable logging — centralized logging

- Do not even connect your systems to any networks, including any type of modem, until you have completed all installation, configuration, hardening procedures, and backed up system

The Seven Most Deadly Sins

1. Weak Passwords
2. Open Network Ports
3. Old Software Version
4. Poor Physical security–Prevent CTRL-ALT_DEL
5. Stale and Unnecessary Accounts
6. Procrastination – I meant to install ...
7. Insecure CGIs

LINUX Tests – Logging

- What was recorded recently in the systems event log? /var/log/messages
- What other logs are available?
- Who can alter the log file?

LINUX Tests – Services

- What services were loaded at startup?
- What processes are currently running?
- What services are set to run?
- What modules are loaded?
- What is accessing the CPU currently?
- What jobs are scheduled to run?

LINUX Tests – Connections

- What networking devices are attached?
- What other hosts can connect to the system under review?
- What communication protocols are used?

LINUX Tests – File Systems

- What file systems are in use?
- Which files and directories are world writeable?
- What are the permissions on sensitive files & directories?
- What files were changed in the last day?
    1. Who changed it?
    2. Why, was that authorized?
    3. Was the change tested?

LINUX Tests – Security Parms

- What automated password controls are in place?
  - Min days
  - Max days
  - Length
- What environment controls are in place?
  - Last users
  - shells

LINUX Tests – Applications

- What applications are installed?
- What malware is present?
- Are there any monitoring tools?

Hardening

Apply Latest Patches

Setup Logging & Log Analyzis

When mounting remote Shares use nodev, nosuid, noxec, nousers

Restrict Root logins to System Console

Require authentication for single user mode

~~:S:wait:/sbin/sulogin

Hardening

User Accounts/Environment

Remove accounts with no passwords

Set account expiration parameters on active accounts

Verify that no uid 0 accounts exists other than root

No '.' or world writable directory in root's $PATH

User home directories should be mode 750 or more

No user dot-files should be world writable

Set default umask for users

Disable core dumps

File and Directory Permissions

Very few files should have world write access - /tmp, /var/tmp, /usr/tmp,... protected using Sticky bit

#find / ! -fstype proc -perm -2 -ls &> | mail -s 'world writable' aeapen@redhat.com

World writable directories should have sticky bit set

Using umask in Startup Scripts

grep umask /home/*/{.bash_profile,.bashrc}

Find unauthorized SUID/SGID executables

Use Extended bits available and set ACL's

Use RPM integrity checking flags

Check using AIDE database

File Integrity

File Integrity can be verified:

- Size and timestamp – can be modified to fool the auditor
- MD5 hashes – secured method, but tedious
- RPM commands
- File Integrity Software:
  - Must be used immediately after the installation
  - Create a database of MD5 hashes of all critical files
  - Monitor changes to these files and send alerts
  - AIDE – open-source, reasonably enterprise-level

redhat.

Linux Security

SERVICES HARDENING

The Attacker's Plan

- Gather intelligence on the target system
- Identify vulnerabilities of the target
- Gain any access to the target required
- Use a vulnerability to successfully compromise the system
- Avoid retaliation for the attack

System Security

- End system compromises include
  - Attacks on system availability
  - Unauthorized access
  - Inappropriate use by authorized users

- These compromises may involve
  - Exploiting flawed or misconfigured software
  - Exploiting protocol weaknesses
  - Exploiting trust relationships

Service Availability & Info Leakage

- Applications must remain available
- Denial of Service (DoS) attacks
  - May simply be used to disrupt service
  - May be used so attacker can spoof clients
- Monitoring is critical
  - Service status and network activity

- Services leak information
  - Banner messages (software and version)
  - Hosts or accounts that may use the service
  - Resources that are available
- This information may be used by an attacker to prepare an intrusion attempt

Service Authentication and Trust

- Secure services need to authenticate client identities and authorize access
- Weak authentication
  - Credential capture attacks
  - Leads to mis authentication, compromise
- Inappropriate trust
  - Stolen data or authentication credentials
  - Compromise of one trusted host can lead to compromise of many hosts

Service Protection Mechanism

- Application Specific hardening
- Firewalls
  - Netfilter
  - Proxy
  - Local packet filters
- TCP Wrappers
- Xinetd
- Pluggable Authentication Modules
- Security Enhanced Linux

Services Hardening

No trusted hosts features: .rhosts, .shosts or /etc/hosts.equiv

Create appropriate banner for any network interactive service

Minimize Boot & Xinetd Services

- Disable telnet & r* services. Use SSH suite
- Enable secure versions of applications
- Disable kudzu
- chkconfig --list

View system activity using SAR(System Activity Report)

Restricted use of Cron & At

redhat

Apache (Web Services)

- The web server is responsible for providing access to content via the HTTP protocol. Web servers represent a significant security risk because:
  - The HTTP port is commonly probed by malicious sources
  - Web server software is very complex, and includes a long history of vulnerabilities
  - The HTTP protocol is unencrypted and vulnerable to passive monitoring.

redhat

Apache Vulnerabilities

- Default Apache installation is usually more open than desired.
- Careless use of CGI scripts can cause major security failures.
- HTTP messages are sent unencrypted by default

redhat

Apache Hardening

- Avoid common misconfigurations.
- Establish policy to deny all access that is not specifically allowed.
- Use SSL when possible
- 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
- Use Authentication for restricted contents

redhat

Apache Hardening(Cont'd)

- Use of some Security Features can lead to problems
  - Indexes
  - ExecCGI
  - FollowSymLinks
  - Includes
  - IncludesNoEXEC
- Enable SELinux
- Remove unnecessary modules
- Attach certificates with your Web servers for SSL

redhat

BIND & DNS Security

- DNS is a critical piece of infrastructure
  - Almost every site must run a name server.
- Information stored in DNS is required by many network services.
  - Spoof DNS responses can lead to subversion of hostname-based security.
  - An attack on DNS may be the first stage of attack on something else.

redhat

Types of Attack on DNS

- Host enumeration
  - Determining existing hosts, possible targets
- Cache poisoning (name spoofing)
  - Bogus information fed to server cache through packet interception, client flooding, or control of s legitimate name server.
- Attacking the parent zone
  - Causing NS deligation to b e changed to hijack control of your zone.

redhat

BIND Security Measures

- Running BIND in chroot() jail
- Address matching list and ACL
- TSIG Key: Transaction Signature
- Restricting zone transfer
- Restricting recursive queries.
- Bogus servers and Blackholes
- Selinux policies for BIND
- Using Split namespaces and views

RPC Security

RPC and Portmap

Issues - trust based on IP Address only,No data encryption,Dynamic port assignment firewall problem

Securing - TCP Wrapper support, protect portmap port with iptables

NIS

Issues - Anyone can bind,NIS Data in clear text

Securing - securenets, do not use BROADCAST for discovery, secure portmap

NFS

Issues - UID/GID issues

Securing - Use NIS/OpenLDAP for UID issues,Use Static ports, root_squash, secure portmap, use NFSV4 for User Authentication

NFS Hardening

- Using Static Ports with NFSv3

  • Simplify firewall configuration by setting static ports for auxiliary NFS services
- nfsd uses TCP and UDP port 2049 by default
- Edit /etc/sysconfig/nfs:

  LOCKD_TCPPORT=888

  LOCKD_UDPPORT=888

  MOUNTD_PORT=889

  STATD_PORT=12001
- Verify ports with rpcinfo -p
- If possible, use NFSv4 with Kerberos supporting ticket-based machine and user authentication, integrity checking via secure checksums, privacy via data encryption

Remote Login Vulnerabilities

- Unencrypted network connections prone to sniffing, spoofing, and hijacking
- Need to protect authentication
  - Defend against credential capture attacks
  - Permit alternative authentication methods
- Need to protect data
  - Provide interactive login security
  - Provide security for other data channels

Resolutions

- Use SSH for interactive login sessions
- Configure public key user authentication between trusted hosts
- Protect unencrypted data channels with TCP port forwarding
- Use protocol version 2 whenever possible
- Use of StrictHostKeyChecking on Clients
- User / Group access controls
  - AllowUsers/AllowGroups
  - DenyUsers/DenyGroups
  - PermitRootLogin

SSH TCP port forwarding

Local Port Forwarding

- $ ssh -L 3025:mail.example.com:25 -N S1.example.com

Tells sshd on station1.example.com:

“I, the ssh client, will listen for traffic on my host's port 3025 and send it to you, sshd on station1. You will decrypt itand forward that traffic to port 25 on mail.example.com as if it came from you.”

Remote Port Forwarding

- $ ssh -R 3025:mail.example.com:25 -N S1.example.com

  Tells sshd on station1.example.com:

  “You, the sshd on station1, will listen for traffic on your port 3025 and send it to me. I, ssh, will decrypt it and forward that traffic to port 25 on mail.example.com as if it came from me.”

Sendmail Vulnerabilities

- Denial of service attacks
- • Users need access to the mail spool
  - • Users have local account on mail server or need to access the server externally
- • E-mail is sent as clear text
- • Need to control transmission of "spam"
  - • May still need to let legitimate hosts relay

Sendmail Resolutions

- Protect and tune the Sendmail service
- • Require users to access the mail server through a secure IMAP or POP service
- • Encrypt e-mail messages
- • Filter e-mail with a blacklist, blackhole, and/or procmail configuration

Sendmail Hardening

- confSMTP_LOGIN_MSG
  - Change version banner
- • confPRIVACY_FLAGS
  - Suppress information leakage
- sendmail.mc options to control DoS attacks
  - confMAX_MESSAGE_SIZE
  - confMAX_DAEMON_CHILDREN
  - confCONNECTION_RATE_THROTTLE
  - confMIN_FREE_BLOCKS

Sendmail Hardening (Cont'd)

STARTTLS

- TLS protection for SMTP communication
- • If offered by the remote server, Sendmail will always attempt to use TLS
- • Very useful, but there are limitations
  - • Only protects the next hop; no guarantee of end-to-end protection if relayed
  - • Messages not protected in queues or spool

Anti-Spam Mechanisms

- • Local mechanisms

  /etc/mail/access and blacklisting
- • Remote mechanisms

  dnsbl: MAPS and other DNS blackholes
- • Local delivery filtering

  procmail and SpamAssassin

Sendmail Hardening (Cont'd)

DNS Blackhole Lists

- DNS can be used as a database to store information about sources of “spam”
  - FEATURE(`dnsbl',`nospam.example.com')
- DNS lookup using reversed octets of sender IP address, with the domain of the blackhole list appended
- Can implement locally or use public or subscription-based blackhole services

Authentication

Authentication consists of four Steps

- proof of identity (Authentication)
- grant of access (Authorization)
- Update of Credentials
- Session Initialization

Pluggable Authentication Modules

PAM Operation

- Application calls **`libpam.so`** for authentication
- Additional libraries are called based on config
- Config decides how the individual libraries' exit codes result in overall success or fail:
  - required success required, other libs still checked
  - requisite failure exits authentication immediately
  - sufficient access granted immediately on success
  - optional result irrelevant

PAM Steps

1. Users invoke PAM aware application
2. Application calls underlying PAM library to perform Authentication
3. PAM Library looks in /etc/pam.d for application configuration file
4. PAM lib loads required auth modules
5. These modules communicate with Application conversation functions
6. Conversation function require user info
7. User provides required information
8. PAM auth modules supply application with with status via PAM Library
9. If auth process successfull, grants required privileges to user
10. If auth process fails, informs process failure

sudo

- Users listed in **/etc/sudoers** execute commands with:
  - an effective user id of 0
  - group id of root's group
- An administrator will be contacted if a user not listed in **/etc/sudoers** attempts to use **sudo**

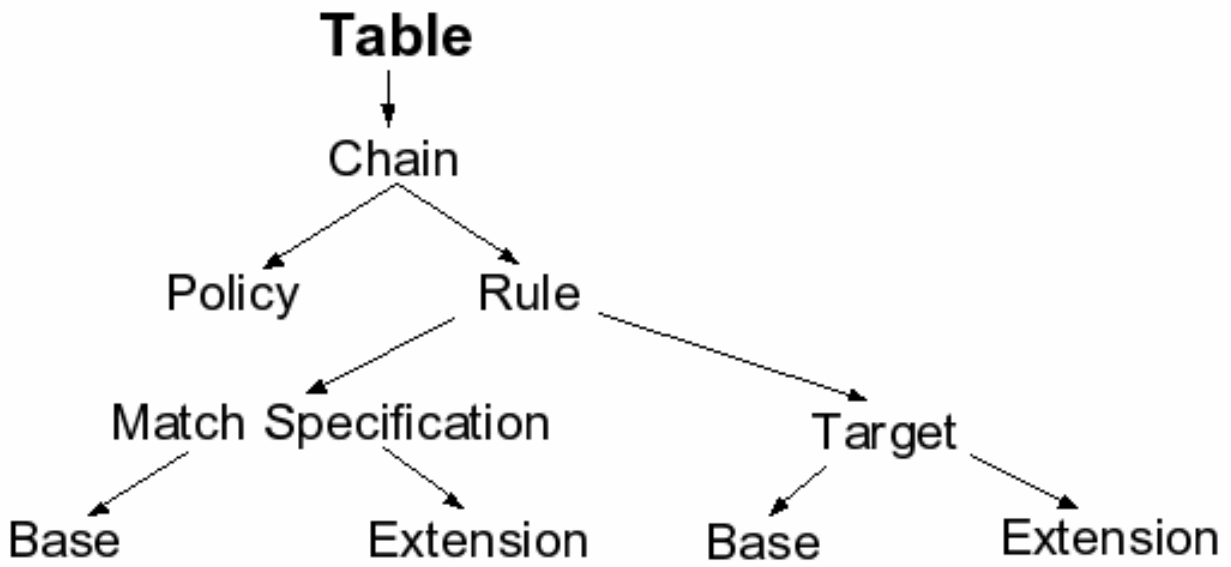
Authentication Troubleshooting

- PAM produces extensive log file entries
  - **`/var/log/messages:`** general messages
  - **`/var/log/secure:`** specific messages
- If the root account is no longer available use Single User Mode to bypass PAM settings.
- **`getent`** prints administrative databases

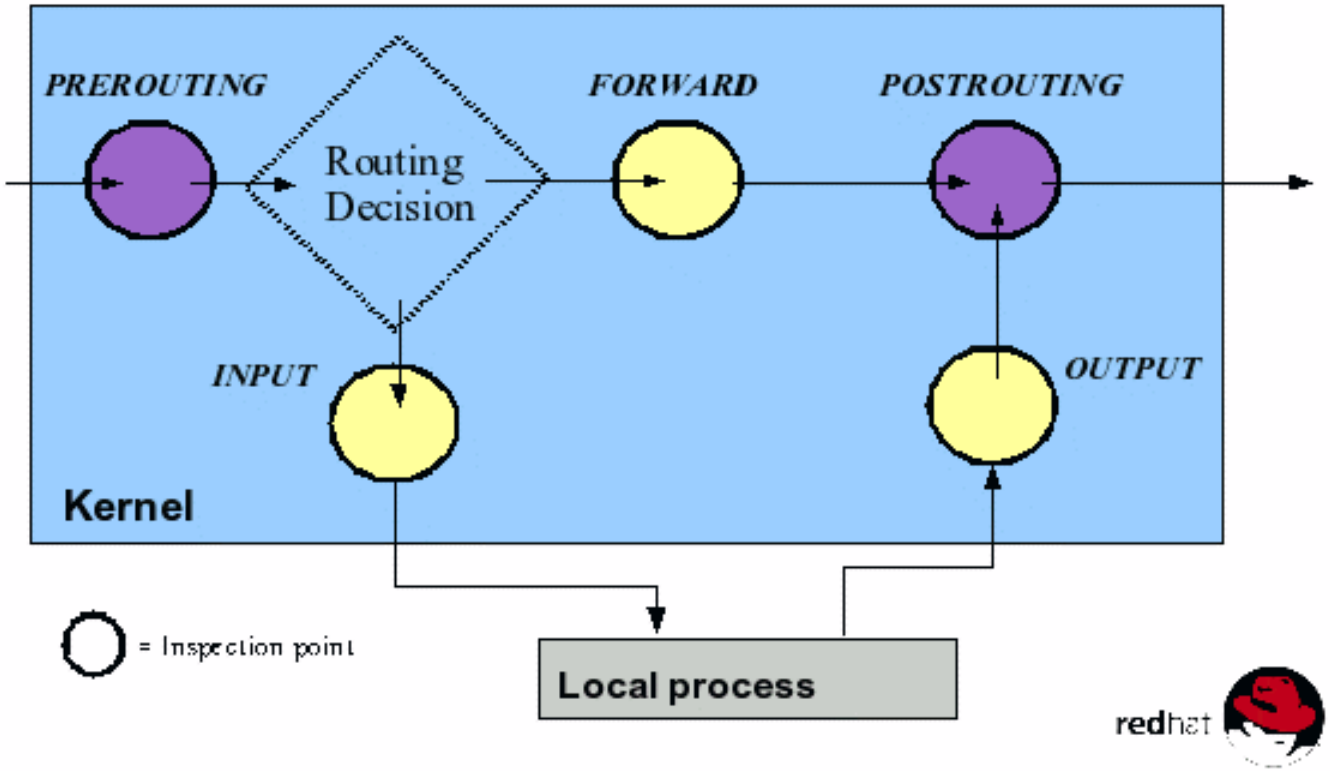
Netfilter Overview

- 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



Netfilter Packet Flow



Packet Flow

A packet forwarded from one Network to another traverses

1. mangle PREROUTING
2. nat PREROUTING
3. filter FORWARD
4. nat POSTROUTING

A packet sent to firewall host traverses

1. mangle PREROUTING
2. nat PREROUTING
3. filter INPUT

A packet sent by firewall host traverses

1. mangle OUTPUT
2. filter OUTPUT
3. nat POSTROUTING

Sample rules

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

iptables -I OUTPUT -p tcp -d 61.1.1.1 –dport 80 -j REJECT

iptables -t nat -A POSTROUTING -s 192.168.0.0/24 -j SNAT –to-source 100.1.1.1-10

iptables -t nat -A PREROUTING -p tcp –dport 80 -j DNAT –to-destination 192.168.0.10

redhat.

Red Hat Enterprise Linux Security

Selinux

Red Hat Enterprise Linux Security

Intrusion detection & recovery

Intrusion Risks

System Downtime

Theft of Data

Modification or destruction of data

Installation of Hostile Software

Bad Publicity and Financial Impacts

Security Policy

- Your organization should have a policy on
  - Detection of possible intrusions
  - Verification and investigation of intrusions
  - Recovery from intrusions
  - Reporting of intrusions
  - Documentation of the policy's execution

- • The policy should be developed with the support of management and legal counsel

Detecting Possible Intrusions

- Regular monitoring of...
- Log files
  - Systems should log to local files and a dedicated remote host
  - Logs should be analyzed on at least a daily basis using logwatch
  - Logwatch results should be delivered to a separate system
  - Look for signs of subverted services
- Network traffic
  - Intrusion Detection Systems
  - Use iptables rules to log suspicious access attempts
  - Use tools like tcpdump and wireshark to capture and store suspicious traffic
  - Implement a honeypot

Detecting Possible Intrusions(Cont'd)

- Open ports
  - netstat (from the local system)
  - Show listening daemons: $ netstat -tulpn
  - Show active connections: $ netstat -tupn
- nmap (from a remote system)
  - Make sure you are authorized to scan first!
  - Scan a host: $ nmap -P0 server1.example.com
  - Scan a subnet: $ nmap -sV 192.168.0.0/24
  - GUI front-end: nmapfe
- Modified files
  - md5sum, aide, rpm

Investigating and Verifying Detected Intrusions

- • Work from a secure environment
  - • Rescue mode
  - • Using a trusted system
  - • Linux-on-CD custom distributions
- • Image suspect block devices for analysis
- • dd if=/dev/sdb1 of=/evidence/sdb1.img conv=noerror,sync
- • $ mount -o loop sdb1.img /mnt/victim-sdb1/
- • Perform file integrity checks

Detecting and Defeating Backdoors & Rootkits

- Strict inbound and outbound firewall rules
- Regular checks of listening ports and files compared to a known baseline
- Network traffic monitoring and recording with tools like tcpdump or Snort
- Monitoring open files with lsof and fuser

- Regular checks of log files and network traffic for suspicious activity
- Regular checks for promiscuous network interfaces
- Test system binaries on a trusted system
- Break automated root kit installs
- Root kit detectors(chkrootkit detector,...)

Recovering from an Intrusion

- Restore from a known-good backup
  - Do not attempt to repair the compromised system!
- Monitor the system for further attacks
  - The attacker may attempt to regain access
  - The attacker may succeed in regaining access
- Reporting and Document the Intrusion
  - How and when the intrusion was detected
  - What actions were taken by whom
  - What the nature of the intrusion was
  - What was done to recover from the intrusion
    - Were policy changes needed?
    - Can you detect similar future intrusions like this?
  - Notification of law enforcement if necessary
    - What evidence was gathered?

Procedural: Incident Response Plan

- Are the six Incident Response steps covered?
  - Preparation
  - Identification
  - Containment
  - Eradication
  - Recovery
  - Lessons Learned (if there are no lessons learned documents either the plan isn't followed or no incidents have occurred).

Open Source Security Tools

Nessus , SARA,NMAP, Snort, TCPDump,SAINT

Shadow (Secondary Heuristic Analysis for Defense Online Warfare)

Jhon the Ripper, Crack

DTK, AntiSniff, HostSentry, PortSentry, ttysnoop

FreeSwan, Bastille-Linux, LIDS,Tiger

Best Practices

Linux mat not be secure in your default configuration

- Security can be added to a very high level, but must be balanced with functionality
- The correct Linux distribution must be chosen, and minimum installation done
- Patches must be diligently applied
- Syslog logs must be exported and analyzed periodically
- Network Services must be kept to a minimum
- User and groups must be periodically audited
- File/folder access control lists must be set
- File Integrity software may be used in high-security installations
- Application-specific security measures are also a must
- Obtain service to audit your security and perform penetration testing

Questions?

arun@redhat.com
+91 93421 66634

redhat