

Security Hardening and System Tweaking of Linux

Created May 14, 2005
by Bruce A. Westbrook

Last Revision: April 14, 2006 – BAW

Introduction

This document describes the step-by-step process of securing a base linux system, based on Red Hat. These instructions should be applicable to most versions of Red Hat, such as:

- Red Hat Enterprise Linux 3 ES / AS
- Red Hat Enterprise Linux 4 ES / AS
- Red Hat Fedora Core 2, 3, 4 and 5
- Red Hat 9.0

In addition, these instructions have been tested successfully with minor variations for other distros, such as SuSE and Ubuntu.

These instructions are meant to be followed immediately following a base installation of linux and an update process (via Red Hat up2date, YUM, or whatever mechanism has been established).

Enjoy!

System Tweaking and Hardening

✓	Description
Boot CLI	<p>Configure linux to boot up into text mode, not GUI. No reason to boot into the GUI by default on this server. To do this, login, launch a terminal session (Application => System Tools => Terminal), edit the <code>/etc/inittab</code> file and change the following line:</p> <pre>vi /etc/inittab id:5:initdefault change to id:3:initdefault</pre>
Password Protect Single-user Mode	<p>To add a level of protection to your box from being easily logged into as root by someone with physical access, perform the following:</p> <pre>vi /etc/inittab id:3:initdefault ~:S:wait:/sbin/sulogin</pre> <p>After editing the <code>/etc/inittab</code> file you should execute the following:</p> <pre>/sbin/init q</pre> <p>This will reinitialize the inittab and include your new settings. Or you can just reboot your box.</p>

	<p>Disable CTRL+ALT+DEL</p>	<p>To disable the accidental rebooting of your Fedora box with your Microsoft happy fingers, perform the following:</p> <pre>vi /etc/inittab</pre> <pre>#ca::ctrlaltdel:/sbin/shutdown -t3 -r now</pre> <pre>ca:ctrlaltdel:/bin/echo "[CTRL]+[ALT]+[DEL]disabled"</pre> <p>After editing the /etc/inittab file you should execute the following:</p> <pre>/sbin/init q</pre> <p>This will reinitialize the inittab and include your new settings. Or you can just reboot your box.</p>
	<p>User Account</p>	<p>If you didn't create a console account during the installation, you need to create one now:</p> <pre>useradd console</pre> <pre>passwd console</pre> <p>New password: <code>password</code></p> <p>After setting up the user you can hit <code>[Alt-F2]</code> and test the login.</p>
	<p>Date / Time</p>	<p>If you setup an NTP server during the installation, you can check that it is running properly by issuing the command: <code>ntpq -p</code></p> <p>The output should show your <code>*LOCAL</code> line plus one line for each of your configured NTP servers. The jitter column should show something other than 4000.00. A telltale sign that NTP synchronization is not working is a jitter of 4000.00. If this is the case, you can try to trace the problem with the following command: <code>ntptrace -vd NTP_server</code></p> <p>If you have no NTP servers setup then set your local date and time as follows:</p> <ol style="list-style-type: none"> 1. Type <code>date</code> to check the current date/time 2. Change the date/time with the following syntax: <code>date -s "06/03/2004 09:36:00"</code> 3. Now sync the hardware clock <code>hwclock --systohc</code>
	<p>NumLock</p>	<p>If you are on a workstation or server (you probably don't want to do this on a laptop) you can set the NumLock to enable on boot as follows:</p> <pre>vi /etc/rc.d/rc.local</pre> <p>Go to the end of the file and add:</p> <pre>INITTTY=/dev/tty[1-8]</pre> <pre>for tty in \$INITTTY; do</pre> <pre> settleds -D +num <\$tty</pre> <pre>done</pre>

	Warning Banners - Local -	Edit the /etc/issue file to add whatever you'd like for a warning banner. An example follows: <pre>vi /etc/issue</pre>
--	--	---

```
*****
*
*
*           This system is for authorized use only
*
*           All activity is logged and monitored
*
*
*
*****
```

```
Red Hat Enterprise Linux ES release 3 (Taroon)
Kernel \r on an \m
```

	Warning Banners - Remote -	Copy the /etc/issue file you just created to /etc/issue.net. Edit the file and remove the last two lines that identify the system, leaving only the warning banner itself. <pre>cp /etc/issue /etc/issue.net vi /etc/issue.net</pre>
	MOTD Banner	Edit the MOTD (Message Of The Day) file to display a message after a successful login: <pre>vi /etc/motd</pre> <pre>Login authenticated and logged</pre>

	<p>Secure xinetd.d Services</p>	<p>Almost every old xinetd service has been replaced by newer and more secure programs. To see if you have any running that you really need, execute the following:</p> <pre>cd /etc/xinetd.d for file in * ; do chkconfig --list \$file ; done</pre> <p>You will see a list of services and whether they're on or off. If any are on, investigate why and determine another way to accomplish the task (such as SSH).</p> <p>To disable specific xinetd services, change to the <code>/etc/xinetd.d</code> directory and then edit the service file. Within the file, simply change the <code>disable</code> from <code>no</code> to <code>yes</code>.</p> <p>Once you have disabled any xinetd services that were running, restart xinetd and verify they are all off.</p> <pre>service xinetd restart cd /etc/xinetd.d for file in * ; do chkconfig --list \$file ; done</pre> <p>Finally, once all the services are shown as <code>off</code>, disable the entire xinetd service as follows:</p> <pre>chkconfig --del xinetd</pre>
	<p>Secure Standard Boot Services</p>	<p>Back to the understanding that every system daemon (service) that does not have a clear and defined purpose on the host should be disabled, let's disable daemons that you don't need or use.</p> <p>Here is a list of commonly started services that you can disable initially:</p> <pre>for file in anacron atd avahi-daemon avahi-dnssconfd bluetooth cups cups-config-daemon firstboot hidd isdn mdmmonitor netfs nfslock portmap rpcgssd rpcidmapd sendmail ; do chkconfig --del \$file ; done</pre> <p>You can also port a list of all your services to a file and browse through it to see what else you can disable. If you don't know what a service does this would be a great opportunity to do some research and understand what your system is running:</p> <pre>chkconfig --list > /root/services</pre>

	<p>Secure SSH</p>	<p>SSH should be configured to display your warning banner and allow only the more secure protocol 2. If you have multiple users on your box you should also not permit root logins. This ensures your remote root access is logged via a user account first.</p> <pre>vi /etc/ssh/sshd_config</pre> <pre> Protocol 2 PermitRootLogin no PermitEmptyPasswords no Banner=/etc/issue.net </pre> <p>After saving this file restart the SSH daemon:</p> <pre>service sshd restart</pre>
	<p>Secure Default Firewall Ruleset</p> <p>* Explanation *</p>	<p>As a brief explanation, the firewall rules for iptables are not really kept in any editable file. That is, the rules, once loaded, exist in memory and will overwrite the file they came from. So how do you configure iptables? And how does it load it's ruleset after a reboot?</p> <p>Well, one way is to make changes to the ruleset in memory, on the fly. You then tell iptables to save the rules in memory to a file. When the box reboots, iptables reads the rules from this saved file.</p> <p>So why can't you just change the actual rules in file? Because it's overwritten any time that you save the rules. And you can't delete rules by simply re-reading the file – the file will append to the rules in memory. Instead, you should create a file of your own with all your firewall rules and comments, run your file to add, delete or modify rules in memory, and then save the iptables memory to the <code>/etc/sysconfig/iptables</code> file. Whew!</p> <p>Rather than create a script to do our changes, we will perform the changes on the fly. We'll then save the memory to a file so they get removed permanently on reboot.</p> <p>To do this, we will perform the following:</p> <ol style="list-style-type: none"> 1. Delete (flush) all of the current rules 2. Define our chains/tables in memory 3. Add our "default" rules in memory 4. Add other rules in memory as needed 5. Save the new iptables from memory to the iptables file 6. Restart iptables to verify our changes

	<p>New Default Firewall Ruleset</p> <p>* All Distros *</p>	<p>** IMPORTANT NOTE: Making these changes via a remote SSH connection WILL lock you out almost immediately, unless you do it via a script.</p> <p>In order to stay consistent with our local firewall rules, we will remove any rules that came with the distribution and set up our own.</p> <p>First, let's remove anything that may currently exist in the iptables rules by "flushing" everything as follows:</p> <pre>iptables -F iptables -F INPUT iptables -F OUTPUT iptables -F FORWARD iptables -F -t mangle iptables -F -t nat iptables -X iptables -Z</pre> <p>Define three chains, INPUT, FORWARD and OUTPUT. These three rules by themselves will drop all incoming packets and all forward packets. All packets initiated by the host will be allowed.</p> <pre>iptables -P INPUT DROP iptables -P FORWARD DROP iptables -P OUTPUT ACCEPT</pre> <p>As a way of explanation, here is the meaning of some of the most common options for iptables commands:</p> <ul style="list-style-type: none"> -i – Interface tells the kernel which interface should be filtered. -p – Protocol defines the protocol that the rule will apply to. Protocols are listed in the <code>/etc/protocols</code> file, and you can define rules for any of them. -s – Source IP address -d – Destination IP address -m – Match is a directive for matching. Commonly you can match state, protocol or both. -j – Jump to what to do if the packet matches the rule --dport – Destination Port. --sport – Source Port --state – defines packet state. There are four states: INVALID, ESTABLISHED, NEW and RELATED. The first three are pretty obvious, while RELATED is a new connection that is associated to an existing connection, such as the FTP data connection being RELATED to the FTP control connection.
--	--	--

Now let's allow some exceptions to our default of dropping all inbound packets. All of the following rules override the global DROP command that we started with.

This rule will accept anything that originates from the local loopback interface and allow it to be used by user applications:

```
iptables -A INPUT -i lo -j ACCEPT
```

This rule allows connections that have already been established and are in the connection table maintained by the kernel, such as responses to our HTTP requests (line wrapped):

```
iptables -A INPUT -m state --state ESTABLISHED -j  
ACCEPT
```

This rule allows any ICMP packets so we can perform pings and traceroutes as well as respond to pings:

```
iptables -A INPUT -p ICMP -j ACCEPT
```

The following rules are used to allow access to various applications. Use them as needed (line wrapped):

SSH

```
iptables -A INPUT -p tcp -m tcp --dport 22 -j  
ACCEPT
```

Webmin

```
iptables -A INPUT -p tcp -m tcp --dport 10000  
-j ACCEPT
```

HTTPS

```
iptables -A INPUT -p tcp -m tcp --dport 443 -j  
ACCEPT
```

Tomcat (Port 8080)

```
iptables -A INPUT -p tcp -m tcp --dport 8080 -  
j ACCEPT
```

MySQL

```
iptables -A INPUT -p tcp -m tcp --dport 3306 -  
j ACCEPT
```

The very last rule we will put in is to drop all remaining packets that didn't match any of our rules. This is simply good practice:

```
iptables -A INPUT -j DROP
```

Finally, save your revised rules to a file, restart iptables and then verify your rules are all in place:

```
service iptables save  
service iptables restart  
iptables -L
```

Other Useful iptables Commands	<p>See the full ruleset file: <code>less /etc/sysconfig/iptables</code></p> <p>Print out the current iptables with corresponding chain numbers (needed to manipulate and insert lines): <code>iptables -L --line</code></p> <p>NOTE: When you print out the current ruleset you may notice that the first line may appear to allow anything from anywhere. To see what it truly is, you'll want to look at the actual file in <code>/etc/sysconfig/iptables</code></p> <p>To insert a rule into a specific position, use: <code>iptables -I INPUT chain-number new-rule</code></p> <p>To replace a specific rule, use: <code>iptables -R INPUT chain-number new-rule</code></p> <p>To add a line at the end of the rule set, use: <code>iptables -A INPUT new-rule</code></p> <p>For multiple, sequential ports, use the <code>:</code> symbol, such as: <code>iptables -A INPUT -p tcp -m tcp --dport 11000:11099</code></p> <p>For multiple, non-sequential ports, use multiport, such as: <code>iptables -A INPUT -p tcp -m multiport --dport 80,443,8080</code></p> <p>To check what rules are being used, you can view the Byte and Packet counters: <code>iptables -vL</code></p> <p>To clear (zero-out) the counters: <code>iptables -Z</code></p> <p>To conduct port redirection – example is port 80 to port 8080: <code>iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 8080</code></p> <p>To conduct port redirection from the local box to the local box – same example: <code>iptables -t nat -A OUTPUT -d 127.0.0.1 -p tcp --dport 80 -j REDIRECT --to-port 8080</code></p>
---------------------------------------	---