RED HAT®
ENTERPRISE LINUX®

redhat.

WHITEPAPER

RISK REPORT
SIX YEARS OF RED HAT ENTERPRISE LINUX 4

Mark J Cox, Red Hat Security Response Team

ABSTRACT

Red Hat® Enterprise Linux® 4 was released on February 15th, 2005. This report takes a look at the state of security for the first six years from release. We look at key metrics, specific vulnerabilities, and the most common ways users were affected by security issues. We will show some best practices that could have been used to minimise the impact of the issues, and also take a look at how the included security innovations helped.

This report is an update to the five-year risk report published in 2010.

TABLE OF CONTENTS

LEGAL NOTICE

1801 Varsity Drive
Raleigh, NC 27606-2072 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588 Research Triangle Park, NC 27709 USA

1. INTRODUCTION

We measure the overall risk of running Red Hat Enterprise Linux 4 as a function of two factors: vulnerabilities and threats. Our first section covers the security vulnerabilities found in packages that are part of Enterprise Linux 4 and the advisories that address them. Our second section covers the threats by examining actual exploitation of those vulnerabilities through exploits and worms.

All the data used to generate this report, tables, and graphs, apply to Red Hat Enterprise Linux 4 AS from release day, 15 February 2005 to 14 February 2011 unless otherwise stated.

Red Hat Enterprise Linux 4 reaches its end of life in February 2012.

2. VULNERABILITIES

At first sight it may appear that Red Hat has released a lot of updates for Enterprise Linux 4 — in the last twelve months of this report publishing a total of 92 security advisories[1] to address 266 individual vulnerabilities. But in reality this is by far a worst-case count, as it treats all vulnerabilities as equal, regardless of their severity, and assumes a system that has installed every available package — which is not a default or even a likely installation.

With the release of Enterprise Linux 4, we started publishing severity levels with package errata to help users determine which advisories were the ones that mattered the most. Providing a prioritised risk assessment helps customers to understand and better schedule upgrades to their systems, being able to make a more informed decision on the risk that each issue places on their unique environment. Red Hat rates the impact of individual vulnerabilities on a four-point scale[2] designed to be an at-a-glance guide to how worried Red Hat is about each security issue. Since 2009 we also started publishing Common Vulnerability Scoring System (CVSS) scores for every vulnerability addressed.

2.1. Vulnerability Counts

There are four variants of Red Hat Enterprise Linux 4; two targeted at server solutions with Enterprise Linux 4 AS and ES, and two targeted at client solutions with Enterprise Linux 4 WS and Red Hat Desktop. The package set available in Enterprise Linux 4 WS and Red Hat Desktop is a subset of that available in Enterprise Linux 4 AS.

1 rhn.redhat.com/errata/rhel4as-errata-security.html

2 access.redhat.com/security/updates/classification/

During Enterprise Linux 4 installation, the user gets a choice[3] of installing either the default selection of packages, or making a custom selection. Table 1 shows the vulnerability counts (unique CVE names), for some selected configurations.

3 docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/index.html

TABLE 1. VULNERABILITIES BY SEVERITY, 6 YEARS

Severity	Enterprise Linux 4.0 AS default install	Enterprise Linux 4 WS default install	Enterprise Linux 4 AS all possible packages
Critical	20	247	252
Important	366	419	463
Moderate	357	497	659
Low	253	268	414
Total	996	1431	1788

A default install of Enterprise Linux 4.0 AS was vulnerable to just 20 critical flaws in the six years. This is because most of the critical flaws have been in web browsers and their plug-ins; Firefox and Mozilla/SeaMonkey packages are not installed by default on distributions intended for server systems. Client systems (Enterprise Linux 4 WS and Red Hat Desktop) do include Firefox, Mozilla, and Helix Player by default, leading to 247 critical vulnerabilities. A custom installation of Enterprise Linux 4.0 AS, selecting every available package, would result in a system affected by the maximum possible number of critical vulnerabilities for the six years, 252.

For the purposes of this study we consider the full six years of vulnerabilities, basing the counts on a version of Red Hat Enterprise Linux 4 obtained on the day of release. During the first six years, eight Update releases were made (Update 1 in June 2005, Update 2 in October 2005, Update 3 in March 2006, Update 4 in August 2006, Update 5 in May 2007, Update 6 in November 2007, Update 7 in July 2008, and Update 8 in May 2009). The Update releases are similar to a "service pack" and contain a roll-up of all security advisories. For example, a user who installed Enterprise Linux 4 in June 2009 would use Update 8 and be affected by only a subset of the issues.

One important thing to note when looking at the default install is that Update 1 changed the default set of installed packages; specifically, it included the Konqueror web browser. Konqueror has had a number of critical issues fixed in it over the years; therefore, the exact critical vulnerability count for a given default install system does depend on which release was first installed.

For Table 1, we counted vulnerabilities not advisories; it's usual for a single security update of a package to fix a number of vulnerabilities at the same time, so the number of advisories, and hence updates needed to be installed, is far lower.

NOTE

You can cut down the number of security issues you need to deal with by carefully choosing the right Red Hat Enterprise Linux variant and package set when deploying a new system, and ensuring you install the latest available Update release.

2.2. Critical Flaws

Vulnerabilities rated critical severity are the ones that can pose the most risk to an organisation. By definition, a critical vulnerability is one that could potentially be exploited remotely and automatically by a worm. However we also stretch the definition to include those flaws that affect web browsers or plug-ins where a user only needs to visit a malicious (or compromised) website in order to be exploited. Since the vast majority of critical severity issues occurred due to web browsers or plug-ins, this is why there is such a difference between the number of critical issues that affect a default install of Enterprise Linux 4 AS and WS.

For the purposes of the severity classification we ignore what privileges the attacker would be able to gain: a remote root compromise via something like Samba would be of a much higher risk than a user-complicit Firefox flaw that results in running code as an unprivileged user, but both would be rated as critical on this scale.

To help qualify the risk we've split up the critical vulnerabilities into those that require some minimal user interaction to be exploitable (such as if a user visits a malicious web page), and those that require no user interaction at all (and therefore could potentially be exploited by a worm).

For Enterprise Linux 4 AS with every package installed, Table 2 summarises all critical issues, and Table 3 breaks out the critical, non-browser flaws.

TABLE 2. ALL CRITICAL FLAWS, SIX YEARS

Type	Number of flaws	"Days of Risk" (median)
Mozilla products (Firefox, Mozilla, SeaMonkey, Thunderbird)	194	0
Media player plug-in (Helix Player)	25	4 (See Note)
Other browsers (Lynx, Links, KDE, QT)	8	0
Non-browser (see Table 3)	25	0
Total	252	0

TABLE 3. NON-BROWSER CRITICAL FLAWS

Package affected	Default installed?	References	Description	"Days of Risk"
exim	No	CVE-2010-4344 RHSA-2010:0970	Heap-based buffer overflow from crafted SMTP connection.	4
samba	Yes	CVE-2010-3069 RHSA-2010:0697	Stack-based buffer overflow triggered by remote malicious client.	0
samba	Yes	CVE-2010-2063 RHSA-2010:0488	Buffer overflow triggered by a remote malicious client.	0
krb5	Yes	CVE-2009-4212 RHSA-2010:0029	Integer underflow.	0
pidgin (was gaim)	No	CVE-2009-2694 RHSA-2009:1218	Overwrite from crafted MSN messages.	0
nss	Yes	CVE-2009-2404 RHSA-2009:1190	Heap overflow in certificate parsing.	1
dhcp	Yes	CVE-2009-0692 RHSA-2009:1136	Stack overflow triggered by connecting to untrusted DHCP server.	0
ntp	Yes	CVE-2009-1252 RHSA-2009:1040	Stack overflow if NTP authentication is enabled.	0
krb5	Yes	CVE-2009-0846 RHSA-2009:0409	Uninitialised pointer free.	0
openssh	Yes	CVE-2008-3844 RHSA-2008:0855	Mitigate an intrusion into certain Red Hat computers where a small number of signed tampered packages were created but not distributed on Red Hat Network.	0
samba	Yes	CVE-2008-1105 RHSA-2008:0288	Heap-based buffer overflow handling over-sized packets.	0
krb5	Yes	CVE-2008-0062 RHSA-2008:0180	Use of an uninitialized pointer.	0
samba	Yes	CVE-2007-6015 RHSA-2007:1114	Stack-based buffer overflow if the "domain logons" option is enabled.	0
samba	Yes	CVE-2007-5398 RHSA-2007:1016	Stack-based buffer overflow if operating as a WINS server.	0
samba	Yes	CVE-2007-2446 RHSA-2007:0354	Heap-based buffer overflows.	0
krb5	Yes	CVE-2007-2446 RHSA-2007:0354	Authentication bypass if the krb5 telnet daemon is enabled.	0
sendmail	Yes	CVE-2007-0956 RHSA-2007:0095	Race condition in the handling of asynchronous signals.	0

Package affected	Default installed?	References	Description	"Days of Risk"
kdenetwork	Yes	CVE-2006-0058 RHSA-2006:0264	Integer overflow in Kopete triggered by a malicious Gadu-Gadu message.	1
evolution	No	CVE-2005-1852 RHSA-2005:639	Stack-based buffer overflow handling iCalendar attachments if the Itip Formatter plug-in is disabled.	0
evolution	No	CVE-2008-1108 RHSA-2008:0516	Format string vulnerability in Evolution triggered by receiving a malicious message.	0
tog-pegasus	No	CVE-2008-0003 RHSA-2008:0002	Stack-based buffer overflow in the OpenPegasus CIM management server.	0
gnomemeeting	No	CVE-2007-1007 RHSA-2007:0086	Format string vulnerability.	7
mod_auth_pgsql	No	CVE-2005-3656 RHSA-2006:0164	Several format string vulnerabilities if mod_auth_pgsql is used for user authentication.	0
gaim (now pidgin)	No	CVE-2005-2103 RHSA-2005:627	Buffer overflow triggered by a malicious away message on AIM or ICQ networks.	2
gaim (now pidgin)	No	CVE-2005-1261 RHSA-2005:429	Buffer overflow triggered by a malicious URL.	0

We have included in these tables the "days of risk" metric. This is commonly defined as the number of calendar days it takes for a vendor to produce updates that correct a flaw after the flaw is first known to the public.

Of note in the data above is the non-default HelixPlayer package. In previous reports we highlighted HelixPlayer as a package that had a number of fixes that were significantly delayed from when they were originally disclosed. In 2010 we deprecated the package and removed it[4] due to numerous security issues in RealPlayer.

4 rhn.redhat.com/errata/RHSA-2010-0981.html

Updates to address 85% of all critical flaws were available from the Red Hat Network the same day or next calendar day after public disclosure of the flaw. This fast response time is a deliberate goal of the Red Hat Security Response Team and forms an essential part of reducing customer risk from critical flaws.

2.3. Expanding "days of risk"

The "days of risk" metric has its limitations and so it isn't particularly useful for comparing different software vendors against each other. The software that makes up the Enterprise Linux 4 distribution is open source, so we're not the only vendor shipping each particular application. Unlike companies shipping proprietary software, Red Hat is not in sole control over the date each flaw is made public. This is actually a good thing and leads to much shorter response times between flaws being first reported to being made public. It also keeps us honest; Red Hat can't play games to artificially reduce our "days of risk" statistics by using tactics such as holding off public disclosure of important flaws for a long period, or until some regularly scheduled patch day.

A more useful metric to help assess risk would also take into account the amount of time that each issue was known to the vendor in advance. As part of our security measurement work since Enterprise Linux 4, we've been tracking how the Red Hat Security Response Team first found out about each vulnerability we fix. This information is interesting as it can also show us which relationships matter the most to us, and identify trends in vulnerability disclosure.

For each of the 1,788 total vulnerabilities, across every package in the six years, we determined if the flaw was something we knew about a day or more in advance of it being publicly disclosed, and how we found out[1] about the flaw. The results are summarised in Figure 1 and Figure 2.

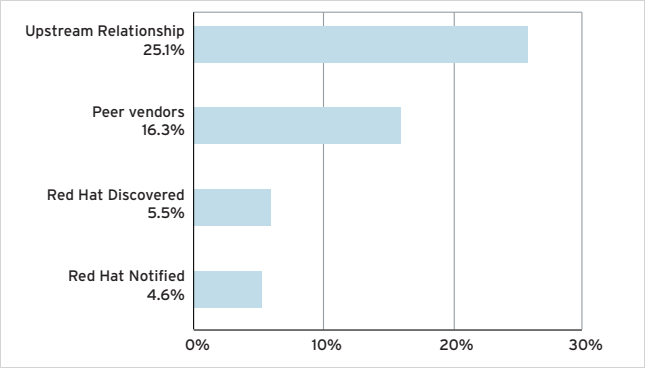

Figure 1. Source of vulnerabilities known in advance, 51.5% of the total

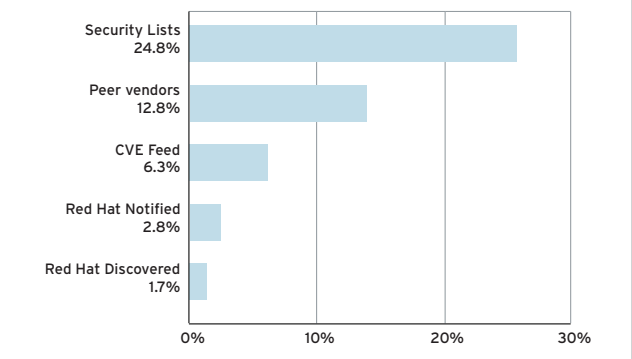

Figure 2. Source of vulnerabilities already public, 48.5% of the total

Red Hat knew about 51.5% of the security vulnerabilities that we fixed at least a day in advance of them being publicly disclosed. For those issues, the average notice was 23 calendar days, although the median was much lower, with half the private issues having advance notice of 10 days or less. Figure 3 shows the distribution of notice periods in more detail.

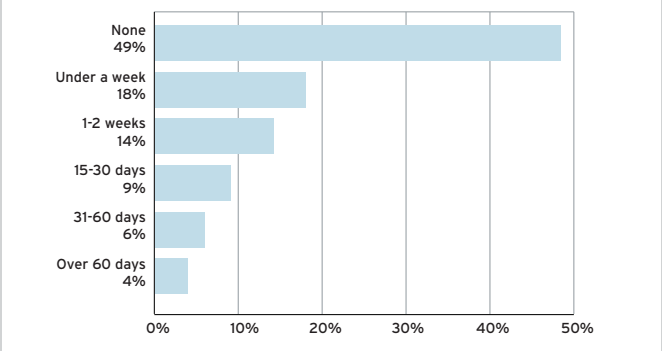

Figure 3. How much time in advance Red Hat knew about issues before they were publicly disclosed

2.4. RISKIEST PACKAGES

In our work tracking and fixing vulnerabilities it sometimes seems like we produce a security advisory for the same packages every month. We therefore analysed Enterprise Linux 4 to find out which packages were responsible for the most vulnerabilities, weighting them [2] to take into account their severity. The results are shown in Table 4, which lists the top 10, ranked across all six years.

TABLE 4. TOP 10 PACKAGES WITH THE WORST SECURITY HISTORY, 6 YEARS

Rank	Package	Number of vulnerabilities			
		Critical	Important	Moderate	Low
1	firefox	185	34	127	36
2	mozilla/seamonkey	148	26	107	23
3	thunderbird	49	22	160	15
4	kernel	0	154	82	47
5	HelixPlayer	25	0	1	0
6	cups	0	34	13	6
7	samba	6	2	4	3
8	krb5	4	11	4	2
9	kdegraphics	0	31	2	5
10	xpdf	0	30	3	5

These top 10 packages together totaled 82% of all the weighted vulnerabilities. The kernel, cups, krb5, and samba packages are part of the default installation of Enterprise Linux 4 AS.

Note

You can reduce the number of vulnerabilities that will affect your systems by removing packages that you don't need or don't use, particularly those that have the worst security history. For example, if you don't use Thunderbird on a machine you could just remove the thunderbird package.

2.5. Advisory Workload

In our original risk reports we graphed the vulnerability workload, a measure of the number of vulnerabilities that security operations staff would need to worry about every day, weighted by severity. But the actual effort in maintaining a Red Hat Enterprise Linux system is more related to the number of advisories we released, rather than the number of vulnerabilities: a single Firefox advisory may fix ten different issues of critical severity, but takes far less total effort to manage than ten separate advisories each fixing one critical Samba vulnerability.

Our Advisory Workload index gives a measure of the number of important advisories that users would need to worry about every day. The higher the number, the greater the workload, and the greater the general risk represented by the vulnerabilities addressed. This workload index is calculated in a similar way to the NIST workload index.[5]

5 nvd.nist.gov/home.cfm?workloadindex

For a given month, Advisory Workload = weighted number of advisories[3] / days in the month. A workload of 1.0 would mean one important advisory a day.

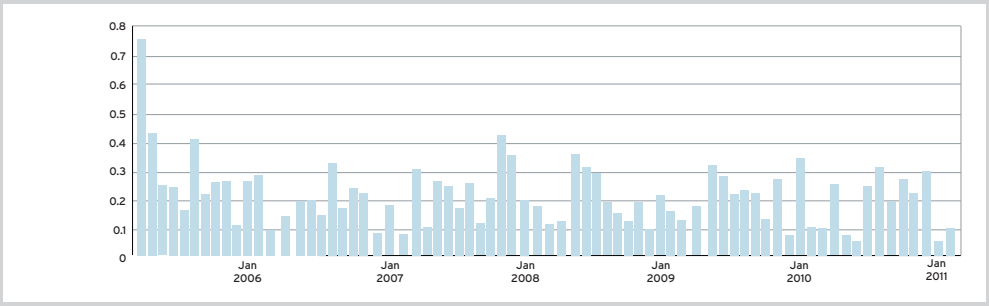

Figure 4. Advisory Workload

Figure 4 shows the advisory workload index for an installation of Enterprise Linux 4 including every package. The initial peak during the first month looks surprising, but is easily explained, as the packages for Enterprise Linux 4 had a code freeze a few months prior to release. This led to a backlog of security issues that were fixed with updates on the date of release. The average Advisory Workload across all dates was 0.21.

Note

Cut down on the number of alerts you receive. Register your systems with the Red Hat Network to get customised notifications for security updates for the packages your systems have installed. If you want to see all security updates for every Red Hat Enterprise Linux version and package, subscribe to the enterprise-watch-list mailing list.[6]

6 access.redhat.com/security/updates/advisory/

3. THREATS

The first part of this report analysed the total vulnerabilities found affecting Enterprise Linux 4. But to get a better evaluation of platform risk we also need to take into account the threat. This part therefore looks at exploits and worms written to take advantage of the vulnerabilities.

Red Hat is continually developing technologies to help reduce the risk of security threats, and a number of these were consolidated into Red Hat Enterprise Linux 4. The most significant technologies were SELinux and Exec-Shield. Exec-Shield is a project which includes support for the No eXecute (NX) memory permission, simulating NX via segment limits, Position Independent Executables (PIE), gcc, and glibc hardening. For more details, a table of the major security technology innovations in Enterprise Linux 4[7] is available.

7 www.awe.com/mark/blog/200801070918.html

3.1. Exploits

An exploit is the way that an attacker makes use of a vulnerability. The Red Hat Security Response Team monitor numerous sources to track which vulnerabilities are being exploited. For this report we compiled a list of the publicly available exploits for the vulnerabilities that affected the first six years of Enterprise Linux 4.

We are interested in those exploits that have the potential to cause remote damage to the confidentiality or integrity of a system and we therefore don't include exploits for vulnerabilities that are limited to a denial of service (affecting availability). We do, however, include exploits which are labeled "proof of concept". A proof of concept exploit may only cause a crash or not quite work properly without modification, but in theory the vulnerability could be exploited properly leading to greater consequences. These proof of concept exploits often show techniques that a skilled attacker can turn into a full exploit.

We found public exploits for 80 vulnerabilities affecting Enterprise Linux 4 in the six years.

3.1.1. Privilege Escalation exploits

We found exploits for 15 vulnerabilities that had the potential to allow an unprivileged user to gain privileges on an unpatched Enterprise Linux 4 system. Of the 15, one required the target system to be using bluetooth drivers (CVE-2005-0750), one was exploitable only on systems with more than one CPU (CVE-2005-0001), one affected only AMD64 and Intel 64 architectures (CVE-2007-4573), and one required a writable sgid directory (CVE-2008-4210).

The remainder (CVE-2010-4170, CVE-2010-3081, CVE-2009-3547, CVE-2009-2698, CVE-2009-2692, CVE-2009-1337, CVE-2006-3626, CVE-2006-2451, CVE-2005-0736, CVE-2004-1235, and CVE-2005-0531) could work on any default, unpatched system. Five of those exploits need unpublished source code adjustments in order to work against a vulnerable Enterprise Linux 4 kernel.

3.1.2. Browser exploits

Exploits for 22 web browser vulnerabilities were found; all but three targeted Mozilla applications (SeaMonkey/Mozilla, Firefox, Thunderbird). These are detailed in the Exploits for browser flaws table.

For each exploit, any resultant code execution would be limited to being run with the same rights as the user that is running the vulnerable browser. It is best practice to never use a web browser or email client as root. Some of these exploits are also blocked if JavaScript is disabled.

EXPLOITS FOR BROWSER FLAWS

Vulnerabilities	Description
CVE-2009-1169, CVE-2009-3382, CVE-2009-0689. CVE-2008-0016, CVE-2006-3677, CVE-2006-0295, CVE-2005-2871, CVE-2005-0399	Exploits for flaws where a malicious web page could run arbitrary code. (The public exploit for CVE-2005-2871 was designed for Windows platforms.)
CVE-2005-1476, CVE-2005-1531, CVE-2005-2264, CVE-2005-1160, CVE-2005-1155, CVE-2005-1157	Exploits for flaws where a malicious web page could run arbitrary JavaScript, doing things like stealing cookies, cross-site scripting, or creating files on the system.
CVE-2009-3076	Exploit for a flaw which could trick a user into installing a malicious module.
CVE-2009-2408	An exploit that can allow man-in-the-middle attacks to spoof arbitrary SSL certificates.
CVE-2007-0981	An exploit that can bypass the same-origin policy, allowing cookie or cross-domain attacks.
CVE-2005-2262, CVE-2005-2269	Exploits for two user-complicit overflow flaws that require the victim to use the 'set as wallpaper' option on a malicious image.
CVE-2005-2710	An exploit for a format-string vulnerability in Helix Player. Helix Player can run as a web browser applet.
CVE-2005-3120	An exploit in the Lynx optional text-based browser. The public exploit is a proof of concept only.
CVE-2006-5925	An exploit in the Links text web browser which could allow a malicious web page to execute arbitrary commands.

3.1.3. Other user-complicit exploits

The next class of exploits are those we term 'user-complicit', in that they need some involvement from the victim to be exploited. Some examples of user involvement would be opening a malicious file with a vulnerable application, or viewing an instant message from an unknown user. Exploits for user-complicit flaws lists the exploits we discovered that require some user involvement.

EXPLOITS FOR USER-COMPLICIT FLAWS

Vulnerabilities	Description
CVE-2009-3605	An exploit for an xpdf flaw which could lead to code execution if a victim opens a malicious PDF document using xpdf.
CVE-2008-2383	An exploit for a flaw in xterm. A malicious text file (or log entry, if unfiltered) that could run arbitrary commands if read by a victim inside an xterm window.
CVE-2008-2292	Proof of concept exploit for a buffer overflow in the Perl bindings for Net-SNMP. This could be triggered if an attacker could convince an application using the Net-SNMP Perl module to connect to a malicious SNMP agent.
CVE-2008-1801	Proof of concept exploit for an integer underflow flaw in rdesktop. If an attacker can convince a victim to connect to a malicious RDP server, the attacker could cause the victim's rdesktop to crash or possibly execute arbitrary code.
CVE-2008-1105	Proof of concept exploit for a heap overflow in Samba. If an attacker can convince a victim to connect to a malicious server, the attacker could cause the client to crash or possibly execute arbitrary code.
CVE-2007-3103	An exploit for a flaw in the X.Org font server. If a local attacker can get the xfs service to be restarted by root they could gain privileges.
CVE-2007-2356	An exploit for a GIMP image editor flaw which could lead to code execution if a victim edits a malicious image using GIMP.
CVE-2006-2656	An exploit for a flaw in libtiff. If an attacker can force a victim to run the 'tiffsplit' executable with a malicious file name they could cause code to run as that user.
CVE-2006-1542	An exploit for a flaw in Python. A victim would need to be tricked into running Python with a very long script name, an unlikely scenario.
CVE-2010-0304, CVE-2005-3243, CVE-2005-2367, CVE-2005-1461, CVE-2005-0699	Exploits for several vulnerabilities in Wireshark. In order to be exploited, a victim with privileges would have to be analysing network packets using Wireshark from a network into which an attacker could inject carefully crafted malicious packets. The protocols affected by the vulnerabilities (SLIMP3, AFP, SIP, LWRES, and RADIUS) are unlikely to be allowed through a border firewall, so the ability to exploit this flaw remotely is restricted.
CVE-2005-1704	An integer overflow in the GDB Binary File Descriptor (BFD) library could allow a malicious executable to execute arbitrary code. However the attacker needs to convince the victim to run the malicious binary (and such a binary could perform arbitrary actions anyway).
CVE-2005-1261	Proof of concept exploit for a flaw in the Gaim instant-messaging client. For some protocols, an attacker could send a carefully crafted message which could lead to code execution.
CVE-2005-0156	An exploit for a flaw in the setuid Perl package. Where perl-suidperl is installed, an unprivileged local user could gain root privileges. (Published exploit needs changes to work against Enterprise Linux 4).

3.1.4. PHP exploits

During March 2007, the "Month of PHP bugs" uncovered a number of issues, some of which affected the PHP packages as distributed with Enterprise Linux 4. The PHP interpreter does not offer a reliable sand-boxed security layer (as found in, say, a JVM) in which untrusted scripts can be run, so any script run by the PHP interpreter must be trusted with the privileges of the interpreter itself. Therefore, in analysis of these issues, exploits which relied on an "untrusted local attacker" were not classified as security-sensitive since no trust boundary was crossed.

This leaves us with the exploits shown in the Exploits for PHP flaws table. These exploits rely on the victim having PHP scripts installed that use the vulnerable PHP functions in a particular way or with untrusted data. In each case the default SELinux targeted policy for the Apache HTTP Server would restrict what a successful exploit is able to do.

EXPLOITS FOR PHP FLAWS

Vulnerabilities	Description
CVE-2007-1286	Exploit for a flaw in the unserialize function. Although unserialize is used by some PHP scripts with untrusted data, the input string required to exploit this issue must exceed ~512K in length, so default Apache line length limits will prevent this from being remotely exploited via input data carried in the HTTP request headers or URI.
CVE-2007-1287	Exploit for a cross-site scripting issue in the phpinfo function. Generally, the phpinfo function should never be used in publicly-accessible PHP scripts.
CVE-2007-1701	Exploit for a flaw in the session extension which allows super-globals to be overridden by an attacker. This flaw is exploitable if session data is taken from an untrusted source.
CVE-2007-1718	Exploit for a flaw in the mail function, which could allow a remote attacker to inject arbitrary headers into PHP generated mail if the mail Subject comprises of user-supplied data.
CVE-2007-1885	Exploit for an integer overflow in the str_replace function, which can be triggered remotely if a script passes large untrusted strings to particular arguments of this function.
CVE-2007-0906	Exploit for a heap overflow in the imap_mail_compose function, which can be triggered if a script uses the function to create a new MIME message based on an input body from an untrusted source.
CVE-2006-4020	Exploits for a flaw in the sscanf function. If a PHP script passed data under an attackers control to sscanf it could result in a buffer overflow.
CVE-2005-1921, CVE-2005-2498	Exploits for flaws in the PEAR XML-RPC code. These exploits require a server to be running a third-party PHP application that exports an XML-RPC interface.

3.1.5. Servers and services exploits

Our final class of exploits are those that affect server applications and services, in Exploits for flaws in servers and services. These are the most serious threats.

EXPLOITS FOR FLAWS IN SERVERS AND SERVICES

Vulnerabilities	Description
CVE-2010-4344	An exploit for a flaw in the Exim Internet Mailer. A remote attacker could make a carefully crafted connection and execute arbitrary code on the server.
CVE-2011-1173	An exploit for a flaw in the kernel SCTP handling. A remote attacker could make a carefully crafted connection resulting in a machine crash.
CVE-2010-0013, CVE-2009-2694, CVE-2009-1376	Exploits for multiple flaws in the Pidgin MSN protocol handler. If a user receives a carefully crafted MSN message it could lead to file disclosure or code execution.
CVE-2009-0692	An exploit for a flaw in the DHCP client which could allow a malicious DHCP server to cause a buffer overflow.
CVE-2008-1679	An exploit for a flaw in the Python imageop module. If a Python application used this module to process untrusted images it could lead to code execution.
CVE-2008-2936	An exploit for a flaw in Postfix. A local attacker could gain root privileges in the unlikely event they have write access to a mail spool directory with no root mailbox.
CVE-2008-1891	An exploit for a Ruby WEBrick flaw. A remote attacker could read arbitrary CGI files, but only if the files were being served from a NTFS or FAT file system.
CVE-2008-0960	An exploit to bypass authentication in Net-SNMP. A remote attacker could execute arbitrary commands if they can connect to a system using Net-SNMP.
CVE-2008-1447, CVE-2007-2926	Problems with BIND not having sufficient randomisation. Exploits were released to use these flaws to poison the DNS cache.
CVE-2007-0957	An exploit for a buffer overflow in the Kerberos administration daemon. A remote authenticated user could execute arbitrary code as root on the Kerberos server.
CVE-2007-6015	An exploit for a buffer overflow in Samba. In order to exploit this flaw, the "domain logons" option would need to be enabled.
CVE-2005-0022	A remote exploit for a buffer overflow in the Exim mail server which could lead to arbitrary code execution. In order to exploit this flaw Exim needs to be installed and SPA authentication specifically enabled, which is not a usual configuration.
CVE-2005-0710, CVE-2005-0709	Exploits for flaws in the MySQL server. A remote authenticated user with privileges to insert or delete from a database table could execute arbitrary code on the MySQL server. The default SELinux targeted policy for MySQL would restrict what a successful exploit is able to do.

Note

The way to reduce your risk from exploits is to make sure your systems have all applicable security updates installed. Red Hat Network can help keep track of this.

3.2. Worms

Worms take advantage of vulnerabilities in order to compromise systems, then use the compromised system to seek out other systems to infect. By our definition, any vulnerability that could be exploited in this way would be classed as critical severity. In the first section of this report we listed every vulnerability that was rated as critical severity and showed that only a subset of those vulnerabilities could be actually used by worms. This is because we also class as critical some browser vulnerabilities where a victim has to take action (for example, visiting a malicious web page) and therefore are not exploitable by a worm.

Worms affecting Linux platforms have been quite scarce in recent years, and the anti-virus vendors who track malware recorded only two (although some variants of each exist) during the six year period of this study:

- Linux/MARE was discovered in November 2005 and was a worm that spread by exploiting a flaw in PHP-Nuke. PHP-Nuke is not shipped as part of Red Hat Enterprise Linux.
- Linux/Lupper was also discovered in November 2005 and was a worm designed to exploit CVE-2005-1921,[8] a flaw in the PHP PEAR XML-RPC server package exploitable through a number of third-party PHP applications. None of the affected third-party applications were shipped as part of Red Hat Enterprise Linux. Additionally, a PHP update in July 2005 fixed the underlying flaw in PHP. Even users that had not patched were also protected from this worm by the default SELinux configuration.

8 www.awe.com/mark/blog/200801070918.html

4. CONCLUSION

The aim of this report was to get a measure of the security risk to users of Red Hat Enterprise Linux 4 during the first six years since release. We've shown that although on the surface it looks like Red Hat released a large number of security advisories, many of them do not apply to usual or default installations, and only a very small subset are a high risk. We've shown:

- A default installation of Enterprise Linux 4 AS was vulnerable to twenty critical security issues over the first six years.
- A customised installation of Enterprise Linux 4, selecting every package, would have been vulnerable to 227 critical browser security issues, and 25 in non-browser packages in the six years. 85% of those vulnerabilities had fixes to correct them available from the Red Hat Network within one calendar day of them being known to the public.
- Red Hat knew about 51.5% of the security issues affecting the first six years of Enterprise Linux 4 in advance. The average time between Red Hat knowing about an issue and it being made public was 23 days (median 10 days).
- We found public exploits for 80 vulnerabilities that could have affected a customised full installation, although the majority relied on user interaction or non-default settings. Attempts to use many of the exploits would be caught by standard Enterprise Linux 4 security innovations.
- The most likely successful public exploits on a default configuration allowed a local, unprivileged user to gain root privileges on an unpatched Enterprise Linux 4 machine. The most serious public exploit was for the non-default Exim Internet Mailer where a remote attacker could easily gain root privileges.

- Two worms targeting Linux systems were found during the six years, but both were from 2005 and affected third-party PHP applications not shipped in Red Hat Enterprise Linux 4.
- "Brute force" attacks against SSH can be successful when systems exposed to the Internet have weak passwords.

It would be foolish to draw conclusions about the future state of security in Red Hat Enterprise Linux 4 solely on the basis of this analysis of the past. However, what we've tried to do is to enumerate the level of vulnerability and threat, and hence overall platform risk. Red Hat treats vulnerabilities in our products and services seriously and the policies of the Red Hat Security Response Team are specifically designed to reduce the risk from security vulnerabilities, such as:

- We place an emphasis on providing the fastest possible highest quality turnaround for critical vulnerabilities. We have a Security Response Team distributed globally which can draw on significant engineering and quality resources to get the things that matter the most fixed quickly.
- We release updates for critical and important security issues as soon as possible rather than batching them into monthly or quarterly updates.
- We provide transparency in the handling of vulnerabilities, our methods, and our metrics.

All of the raw data used to generate the statistics in this report along with some tools to analyse them are available[9] from the Red Hat Security Response Team. We also provide other tools and data that can help security measurement including CVE and CVSS mappings for all our advisories and OVAL definitions.

9 www.redhat.com/security/data/metrics/

5. FURTHER READING

What's new in security for Red Hat Enterprise Linux 4
www.redhat.com/magazine/006apr05/features/security/

Vulnerability Types for Enterprise Linux 4
www.awe.com/mark/blog/200610241300.html

Security Features in Red Hat Enterprise Linux and Fedora Core
www.awe.com/mark/blog/200801070918.html

SELinux: A New Approach to Secure Systems
www.redhat.com/apps/webform.html?event_type=whitepaper&eid=315

Statistics and data from the Security Response Team
www.redhat.com/security/data/metrics/

6. ABOUT THE AUTHOR

Mark J Cox lives in Scotland and is the Director of Red Hat Security Response. Over the last 16 years, Mark has developed software and worked on the security teams of some of the most popular open source projects, including Apache, mod_ssl, and OpenSSL. Mark is a founding member of the Apache Software Foundation and the OpenSSL project, and a board member of the MITRE CVE and OVAL projects. In his spare time he automates his home[10] and plays music.[11]

10 www.awe.com/ha/

11 www.sonik.co.uk/

A. NOTES

[1] We count the first place that the Red Hat Security Response Team heard about a security issue. 'Peer vendors' are other distributors of open source software who are part of vendor-sec. 'Upstream relationship' covers issues told to us because we work on the upstream projects or they contacted us to tell us about an issue. 'Red Hat discovered' are issues Red Hat employees discovered. 'Red Hat Notified' are where some customer, researcher, or other third-party told us about an issue through email, Bugzilla, or other means. 'Security Lists' includes public lists like Bugtraq and Full-Disclosure as well as blogs and other Internet sites. 'CVE feed' is a feed of newly allocated CVE names for public issues from MITRE.

[2] To rank the riskiest packages we use a weighting of "Critical + Important/5 + Moderate/25 + Low/100".

[3] To weight the effort of dealing with advisories, Critical and Important advisories are scored as 1.00, Moderate advisories as 0.20, and Low advisories as 0.05. This is designed to be similar to the way that NIST calculate their workload metrics.

www.redhat.com
#5732067_0311