



Classification of Security Issues

By Mark J Cox

Abstract

Red Hat has implemented a scheme from Red Hat Enterprise Linux 4 to publicly classify the impact of security issues found in our products and services. Customers want a simple way to judge the severity of security updates so they can see which issues matter the most. The media needs guidance to understand the priority and criticality of the updates Red Hat releases. The classification scheme chosen gives a rating of security impact on a four point scale and is designed to be an at-a-glance guide to how worried Red Hat is about each security issue. This paper looks at why Red Hat has started classifying vulnerabilities, the details of the scheme we are using, and how well the classification matches our response times for Red Hat Enterprise Linux 3 vulnerabilities.

February 2004

Table of Contents

Why rate security flaws?	2
Aims of the severity classification scheme	2
Red Hat Severity Classification	3
Implementation Details	5
Example classifications	5
Critical flaws in Red Hat Enterprise Linux 3	7
Known issues with the classification system	9
Best Practices for severity classification	12
Comparison with the Microsoft impact scale	13
Metrics for Red Hat Enterprise Linux 3	15



Why rate security flaws?

In the past few years increasing attention has been paid to software flaws that have security consequences. Software developers find and fix bugs on a daily basis. When bugs have a security consequence, updates are produced by affected vendors and advisories explaining the flaws are written. To date no widely-accepted method for rating the severity or impact of these flaws exists, so vendors, researchers, and the press pick arbitrary schemes designed to try to help users understand how important security issues are.

Since the release of the first Red Hat Enterprise Linux, Red Hat has rated the severity of flaws internally, using that rating to determine the order in which fixes are prepared. These priorities influence every aspect of the product's life cycle: investigation, engineering work, testing, QA, and all the way through to the final release. Until Red Hat Enterprise Linux 4, Red Hat did not make these ratings available to the public.

Customers have unique environments and every software flaw can potentially affect every customer in a different and unique way. A vendor could misclassify important issues by underestimating or overestimating the severity of flaws, devaluing a rating scheme altogether. At the time of writing there are still many well known vendors that do not give severity ratings in public advisories for software updates. These include companies such as Apple¹, Cisco², and Sun³.

In addition to the work on classification, we've also looked at other aspects of security advisories that could be improved. The aim of our security advisories is to give users everything they need so that they can assess each flaw's unique effect on their particular environment. Red Hat already provides good textual descriptions of potential risks from flaws, but wanted to improve those descriptions. Advisories now require links to trackable bugs for each security issue, and in many cases these lead to the exact source code patch that was applied to correct the flaw, aiding in transparency.

Aims of the severity classification scheme

Our overall aim in creating a public classification scheme was to give our customers a quick and simple rating to show them how worried they should be about an issue, based on how worried we are about it. In addition, Red Hat wishes to create a classification scheme that:

- is based on a technical analysis of the type of flaw. This determines the likelihood that the flaw is exploitable and is a measure of risk if the flaw is left unpatched.
- helps customers make a risk assessment of security issues that affect them. It would be nice if all users of our software applied our security

1 <http://lists.apple.com/archives/security-announce/2004/Oct/msg00000.html>

2 <http://www.cisco.com/warp/public/707/cisco-sa-20041202-cnr.shtml>

3 <http://sunsolve.sun.com/search/document.do?assetkey=1-26-57652-1>

updates as soon as they are published, but in reality this doesn't happen. Customers have their own procedures for rolling out software updates, and they need to weigh the overall risk of the unfixed flaw against the risk of performing an update. We can help reduce the risk inherent in updating by providing back-ported patches and automated update services such as the Red Hat Network, but we still must accept that some customers are going to assume the risk of not applying an update.

- helps our customers prioritize issues without having to rely on ratings from third parties such as the press or vulnerability aggregation services. Many security research companies report on Linux flaws without doing any additional investigation or analysis of the issue themselves. We'd like to think that because the code is open that these companies can look at the flaw, look at the patch, and come up with a useful and accurate assessment of the flaw and its impact. In reality, this rarely happens. When vendors such as Red Hat provide guidance to the severity of security issues, this can be used directly by third parties reporting on the vulnerabilities, leading to more accurate reports.
- matches our current internal process for prioritizing and responding to vulnerabilities. Our Security Response Team constantly monitors and investigates issues that affect our products, assessing the impact and severity of each issue. We already have internal processes in place to deal with vulnerabilities and expedite issues based on severity through the entire life-cycle. By matching this internal process, customers will be able to understand the decisions we make when prioritizing and releasing updates.
- is independent of the current threat landscape. We need the rating to stay constant through the life of the advisory and not change our published ratings except in exceptional circumstances. We must assume that any vulnerability that looks technically feasible to exploit will be exploited. However we will take into account any security technologies that can remove or reduce the threat of particular issues.
- allow customers to rank security updates. In the first six months of 2004, Red Hat released updates to correct 81 individual vulnerabilities. For Red Hat Enterprise Linux 3 this required 64 security advisories on 42 separate dates. Users will always have to read the advisories in full in order to make an accurate assessment of how the flaw affects their unique environment; but a user faced with a list of 64 advisories could be forgiven for not knowing which ones are most important and should be dealt with immediately, and which could be implemented at a later time.

Red Hat Severity Classification

Red Hat rates the impact of individual vulnerabilities on a four point scale. The scale takes into account the potential risk of a flaw based on a technical analysis of the exact flaw and it's type, but not the current threat level. Therefore the rating we give to an issue will not need to change if an exploit or worm is released for a flaw, or if one is available before release of a fix.



Figure 1. Security classification scale.

Anything ranked as critical impact clearly needs urgent and immediate attention. Anything ranked as low impact can be dealt with in a more leisurely fashion.

It is important to note that this classification is not replacing the need for customers to read about and investigate how the issue uniquely applies their environment. An issue might be marked critical but the customer might not have those packages installed, or might be using them in a non-vulnerable way.

Critical Impact

This rating is given to flaws that could be easily exploited by a remote unauthenticated attacker and lead to system compromise (arbitrary code execution) without requiring user interaction. These are the types of vulnerabilities that can be exploited by worms.

For example, overflows in image handling libraries could be critical impact because a carefully crafted malicious image could be embedded in a web page or sent to a user as HTML email. Flaws that require an authenticated remote user, a local user, or an unlikely configuration would not be classed as critical impact.

Important Impact

This rating is given to flaws that can easily compromise the confidentiality, integrity, or availability of resources. These are the types of vulnerabilities that allow local users to gain privileges, allow unauthenticated remote users to view resources that should otherwise be protected by authentication, allow authenticated remote users to execute arbitrary code, or allow local or remote users to easily cause a denial of service.

Moderate Impact

This rating is given to flaws that may be harder or more unlikely to be exploitable but given the right circumstances could still lead to some compromise of the confidentiality, integrity, or availability of resources. These are the types of vulnerabilities that could have had a critical impact or important impact but are less easily exploited based on a technical evaluation of the flaw, or affect unlikely configurations.

Low Impact

This rating is given to all other issues that have a security impact. These are the types of vulnerabilities that are believed to be require unlikely circumstances to be able to be exploited, or where a successful exploit would give minimal consequences.

Implementation Details

A Red Hat security advisory may contain fixes for more than one security issue as long as all issues are affecting the same package. It's quite common for security audits to turn up several different issues in a single package simultaneously, and these issues may well have different severities. Red Hat use the Common Vulnerabilities and Exposures (CVE) project⁴ at Mitre as a way to document the distinct security issues in each advisory. All Red Hat security advisories since 2000⁵ have relevant CVE names listed.

For each CVE named issue in an advisory the Red Hat Security Response Team will determine the impact rating based on a technical analysis of the issue. The overall severity of an advisory is then taken as the highest severity of all the individual issues. Therefore if an advisory covers an important impact issue as well as four low impact issues the overall advisory impact will be important. It's also possible that the severity of an issue will be different for different versions of a distribution, or even different architectures, but the same rule applies.

In order to simplify the security advisories we do not currently list the individual impact ratings for each issue in the advisory text. Instead, each advisory contains links to relevant tickets in Red Hat's bug tracking system where the impact as well as any additional commentary is given.

For the release of Red Hat Enterprise Linux 4 and beyond, the overall advisory impact will be listed as part of the advisory text as shown:

This update has been rated as having low security impact by the Red Hat Security Response Team.

Example classifications

To further clarify the impact levels, here are some examples taken from Red Hat Enterprise Linux 3 advisories:

The very first advisory for Red Hat Enterprise Linux 3 was RHSA-2003:324⁶ affecting Ethereal, a package for monitoring network traffic.

4 <http://cve.mitre.org/>

5 <http://www.redhat.com/security/transparent/cve/>

6 <http://rhn.redhat.com/errata/RHSA-2003-324.html>

“A number of security issues affected Ethereal. By exploiting these issues, it may be possible to make Ethereal crash or run arbitrary code by injecting a purposefully-malformed packet onto the wire or by convincing someone to read a malformed packet trace file.” (CAN-2003-0925, CAN-2003-0927)

These flaws could therefore allow a remote attacker to run arbitrary code as root. However, the system administrator would have to be using Ethereal to monitor network traffic, most likely interactively. The attacker would have to inject packets onto the network that would trigger the flaw while the administrator was using the Ethereal tool. The attacker would need to know when the admin would be online or send packets continually over a long period of time. These issues only affected protocols that would normally not be open to the public, therefore the attacker would need to be on the same network as the machine they wished to attack. So although the flaws have a serious consequence (remote root), they are actually quite unlikely to be exploitable in practice. These issues were therefore given a moderate rating.

In RHSA-2004:066⁷ a flaw affecting the Linux kernel was fixed:

“...a flaw in return value checking in mremap() in the Linux kernel versions 2.4.24 and previous that may allow a local attacker to gain root privileges. No exploit is currently available; however this issue is exploitable.” (CAN-2004-0077)

This flaw affected all users of Red Hat Enterprise Linux 3 and at the time of disclosure it was technically exploitable. In fact, an exploit was distributed publicly a short time after disclosure. For systems with untrusted local users this flaw was a significant issue. Even without untrusted local users, the flaw could be part of a blended threat, used after an attacker takes control using a non-root remote flaw. This issue was therefore given an important rating.

In RHSA-2004:402⁸ libpng flaws were fixed:

“...several buffer overflows in libpng. An attacker could create a carefully crafted PNG file in such a way that it would cause an application linked with libpng to execute arbitrary code when the file was opened by a victim.” (CAN-2004-0597)

The libpng library is used by a large number of applications that display graphics files. It's likely that a malicious image sent to a Red Hat Enterprise Linux 3 user would be opened with some application that was vulnerable to this flaw. A working and reliable exploit would need to target a particular application, but since the flaw was a straightforward stack-based overflow the technical chances of being able to create such an exploit was very high. An additional attack vector would be graphical email clients that parse and display in-line png files. This could allow exploitation without specific user action--a Linux worm. Exploitation via malicious web pages specific to a browser using the vulnerable library would also be possible. This issue was therefore given a critical rating.

In RHSA-2004:259⁹ a flaw in Samba was fixed:

7 <http://rhn.redhat.com/errata/RHSA-2004-066.html>

8 <http://rhn.redhat.com/errata/RHSA-2004-402.html>

9 <http://rhn.redhat.com/errata/RHSA-2004-259.html>

“...a flaw in the internal routine used by the Samba Web Administration Tool (SWAT)... could allow an attacker to execute arbitrary code” (CAN-2004-0600)

Flaws affecting specific and unlikely configurations are less serious than flaws that affect likely configurations. This flaw was not classed as a critical impact because it relied on the attacker being able to connect to the administration port--a port which isn't enabled by default and is documented as needing to be protected by firewall rules. Having a Samba installation with a SWAT port open to the Internet is a significant security issue by itself, independent of this flaw. This issue was therefore given an important rating.

Critical flaws in Red Hat Enterprise Linux 3

In recent years, Linux distributions have had few flaws that would fall into our rating as critical impact. Flaws in services such as Samba and Apache with OpenSSL¹⁰ have led to Linux worms, and these vulnerabilities are among those that would be classified as critical impact.

The following issues in the first 12 months of Red Hat Enterprise Linux 3 released October 2003 would be classed as critical impact. However, as of the time of writing none of these issues had been exploited by worms.

(See table 1 on next page.)

¹⁰ <http://securityresponse.symantec.com/avcenter/venc/data/linux.slapper.worm.html>

CVE name, RHSA	Description of flaw and classification reasoning
CAN-2004-0904 (RHSA-2004:486)	An integer overflow in the BMP handling code in Mozilla. An attacker could create a carefully crafted BMP file which would cause Mozilla to execute arbitrary code if opened by a victim. This flaw is critical as a user could visit a web page that contained an malicious in-line BMP image.
CAN-2004-0687 CAN-2004-0688 (RHSA-2004:478)	A stack overflow flaw in the X.Org libXpm library used to decode XPM images. An attacker could create a carefully crafted XPM file which would cause an application to execute arbitrary code if opened by a victim. This issue is borderline on being important since it most likely will require user interaction in order to be exploited.
CAN-2004-0817 (RHSA-2004:465)	Heap overflow flaws in the imlib BMP image handler. An attacker could create a carefully crafted BMP file which could cause an application linked with imlib to execute arbitrary code if the file was opened by a victim.
CAN-2004-0785 (RHSA-2004:400)	Buffer overflow flaw in the Gaim (instant message client) RTF message parser. A remote attacker could send carefully crafted data leading to arbitrary code execution. This issue is borderline on being important since it requires a user to be running Gaim and to receive a carefully crafted malicious instant message.
CAN-2004-0642 CAN-2004-0643 (RHSA-2004:350)	Double-free bugs in the Kerberos 5 KDC and libraries. A remote attacker could exploit these flaws to execute arbitrary code.
CAN-2004-0597 (RHSA-2004:402)	Buffer overflows in libpng. An attacker could create a carefully crafted PNG file which would cause an application linked with libpng to execute arbitrary code if the file was opened by a victim.
CAN-2004-0414 (RHSA-2004:233) CAN-2004-0396 (RHSA-2004:190)	Flaws reading malformed "Entry" lines in CVS. An attacker who has access to a CVS server may be able to execute arbitrary code under the UID on which the CVS server is executing. This issue is more likely to be only important as it does require an authenticated attacker - however some open source projects give untrusted public users anonymous access to CVS servers, which could then be remotely and automatically exploited using these flaws.
CAN-2004-0006 (RHSA-2004:033)	Buffer overflow in the Gaim Yahoo! YMSG packet. It is possible that a remote attacker could send carefully crafted data to a vulnerable client and lead to arbitrary code execution.
CAN-2003-0962 (RHSA-2003:399)	A heap overflow flaw in rsync. On machines where the rsync server has been enabled, a remote attacker could use this flaw to execute arbitrary code as an unprivileged user. This issue is more likely to be only important as it requires that the administrator has enabled rsync access - however some open source projects give untrusted public users anonymous read-only access to rsync servers, which could then be remotely and automatically exploited using these flaws.

Table 1: Critical impact issues in Red Hat Enterprise Linux 3

Known issues with the classification system

Effect of technology innovations

Not all users can apply all fixes for all security issues in a timely manner, so our security engineering teams research, develop, and code a number of technologies for Linux and Red Hat Enterprise Linux to help mitigate certain types of security flaws. Innovations that reduce the impact of certain common vulnerabilities include:

- Exec-Shield¹¹ technology. Found in Fedora Core, Red Hat Enterprise Linux 3 (Update 3 and later), and Red Hat Enterprise Linux 4. This technology does a good job of preventing common stack buffer overflow exploits on x86 architectures.
- Package changes made to mitigate flaws. For example: removing the need for risky setuid executables, or dropping root privileges by default (tcpdump¹²).
- Updates to glibc made in Red Hat Enterprise Linux 4. These updates can block certain types of flaws, such as exploits for double free memory bugs and some types of overflows.
- SELinux¹³. Found in Red Hat Enterprise Linux 4, SELinux can mitigate threats against specific applications with a targeted policy.

When a technology--enabled and most likely used by default-- completely blocks the exploitation of a particular vulnerability across all architectures, we will adjust the severity impact classification level. When a technology reduces the risk of a security issue, we may adjust the severity impact level and give an explanation of the the decision in the tracking bug entry.

Blended threats

One difficulty with assigning a severity rating is that it's sometimes hard to predict all the ways a particular flaw could be abused. Many exploit tools have made use of multiple vulnerabilities in order to accomplish a single goal. The component flaws aren't particularly serious in isolation, but when used together they can have significantly more impact.

One recent example of this is an issue in PHP, an HTML-embedded scripting language commonly used with the Apache HTTP server. Researcher Stefan Esser discovered a flaw when "memory_limit" is enabled in versions of PHP 4 before 4.3.8 (CVE name CAN-2004-0594). If a remote attacker could force the PHP interpreter to allocate more memory than the memory_limit setting before script execution begins, then the attacker may be able to supply the contents of a PHP hash table remotely. This hash table could then be used to execute arbitrary code as the 'Apache' user. By itself this flaw doesn't look too serious – it requires that an attacker finds some way to exhaust memory before script execution starts.

However, a few months earlier a flaw was found in Apache: a remotely

11 http://www.redhat.com/f/pdf/rhel/WHP0006US_Execshield.pdf

12 https://bugzilla.redhat.com/bugzilla/show_bug.cgi?id=49635

13 http://www.redhat.com/apps/webform.html?event_type=whitepaper&eid=315

triggered memory leak in the Apache HTTP Server (earlier than version 2.0.50) allowed a remote attacker to perpetuate a denial of service attack against the server, forcing it to consume large amounts of memory (CAN-2004-0493). By itself this Apache flaw isn't too serious – an Apache child process will consume memory until it reaches system configured limits, then will usually get killed and replaced. But together, these two issues form a blended attack that could allow a remote attacker to gain the ability to run arbitrary code as the 'apache' user.

An article in *The Register* reaches the same conclusion about blended threats¹⁴:

"[Attacks] are getting more and more complex, and attackers are using multiple vulnerabilities to carry them out. It also represents what I consider a flaw in the way the IE security team looks at and rates vulnerabilities. The "mitigating factors" in these vulnerabilities have always been determined by looking at the problems in singularity. Things like "an attacker would have to be able to write files locally" or "this would only work if code was run in the Local Intranet Zone.... When Microsoft then uses these factors to schedule hot fix development and deployment, we find ourselves in the position we' e in today: insufficient ranking is given to these vulnerabilities, attackers piggyback exploits together - leveraging one against the other to fully compromise a machine - and here we are sitting around with no patch available."

Cross-site scripting vulnerabilities (sometimes called XSS) pose a unique challenge. It's hard to make generic statements about the impact of these flaws. If a website is using HTTP cookies for storing authentication data and that site also has a cross-site scripting flaw, an attacker could trick a user into revealing those authenticating cookies. But cross-site scripting isn' t just about cookie stealing; A banking site with a XSS flaw could potentially be used as part of a phishing attack¹⁵. Many vendors simply describe XSS issues as "can run arbitrary script" without going into potential specific risks. Red Hat rates cross-site scripting flaws as important unless they are significantly mitigated.

The same issue affects vendors in different ways

The nature of open source allows multiple companies to ship the same or nearly the same software. This means that for each flaw, users will see advisories from different organizations describing the same issue in different ways, depending on how it impacts that vendor' s version. These advisories are frequently released at different times and/or over a long period of time. This can make it hard to see that vendors are talking about the same flaws, especially if multiple flaws in the same package are found in quick succession. Red Hat uses the Mitre Common Vulnerabilities and Exposures (CVE) project in an effort to help customers track issues across multiple vendors.

CVE is a dictionary of issues, not a vulnerability database. Most Linux vendors now use CVE names in their advisories, even if they do not use the

14 http://www.theregister.com/2004/06/28/ie_is_complex/

15 http://news.netcraft.com/archives/2004/12/06/suntrust_site_exploited_by_fraudsters.html

actual CVE description verbatim. Even with the same flaws, however, a vulnerability can affect different vendors in different ways; some of these differences can be quite significant.

An example of a vulnerability that affects vendors differently is the “Apache Chunked Encoding” flaw, given CVE name CVE-2002-0392. The Apache Week security center gives this issue a critical impact severity rating.¹⁶

CVE-2002-0392	Apache 1.3 through 1.3.24, and Apache 2.0 through 2.0.36, allows remote attackers to cause a denial of service and possibly execute arbitrary code via a chunk-encoded HTTP request that causes Apache to use an incorrect size.
---------------	--

Table 2: Apache Chunked encoding flaw

The technical details of this flaw are a little more involved¹⁷ than the one paragraph CVE description. Basically, the flaw allowed an attacker to cause the heap to be overwritten via a `memcpy()` call with a large length parameter. On 32-bit Linux systems, such a flaw caused a heap overflow and lead to a SIGSEGV when attempting to write out of the bounds of mapped memory. The Apache default SEGV handler didn't call functions or do anything unsafe with the resultant broken heap. On Linux systems, exploiting the flaw would cause the Apache process handling the request to die. On systems using Apache 1.3 or Apache 2.0 with the default MPM process model, another process would get created to replace any that die. Therefore for Linux systems this vulnerability has little security consequence - isn't even a denial of service.

On some BSD-based systems, however, the impact was different. The internal implementation of the `memcpy()` syscall allowed the heap overflow to be controlled--to some extent--by the attacker. This meant that the flaw could lead to arbitrary code execution. This quickly became more than a theoretical possibility, as exploits were coded and made public that took advantage of the flaw on those systems.

It's easy with the benefit of hindsight to say that this flaw on Red Hat Linux systems (which were at the time shipping Apache 1.3) was of very low impact, but on other systems could be critical impact. However, the flaw was disclosed by a third-party research company the same day they reported it to the Apache Software Foundation. This did not give the Apache security team time to complete a full technical analysis, so many Linux vendors rushed to complete and push out updates, just in case it was found to be a flaw with a critical impact. Researchers and the media then added to the confusion by reporting that Linux was affected by a remote arbitrary code execution vulnerability.

This particular example highlights a general problem when the initial reporting about a potential security flaw comes from the researcher that found the flaw rather than the software authors or vendors. In working with the Apache Software Foundation, we've found many cases where the researcher has deliberately misstated the severity of a flaw – possibly in order to gain more publicity for themselves or their companies. There are

¹⁶ <http://www.apacheweek.com/features/security-v1.3.24>

¹⁷ <http://www.apacheweek.com/issues/02-06-21#security>

significant rewards for finding a flaw in the Apache Web server that affects millions of sites around the world. Companies that make money from security advisories have a vested interest in getting as much publicity as possible for their flaw, and in our experience flaws that are less than critical can get sensationalized to receive press attention. Unfortunately this practice leads to the devaluation of words such as “critical”. For example, one company that provides vulnerability advisory aggregation rates each issue on a scale from “less critical” through to “highly critical”, so even the flaws with minimal security impact are now “critical” in some way.

Best Practices for severity classification

When designing the Red Hat classification system we took a look at the best practices for the security severity ratings performed by other vendors.

In their vulnerability notes, CERT/CC use a metric field¹⁸ comprising of a number between 0 and 180 derived from factors including threat, risk, and impact. The exact formulae used to calculate the metric is not given, and the field itself isn't particularly obvious. In some vulnerability notes, the short overview of the flaws will contain words like “critical”, but these are not defined. One of the problems with a simple numeric scheme is that customers make assumptions about the relative severity of issues – an issue rated 10 sounds twice as serious as an issue rated 5.

Novell SUSE gives a severity rating on a numeric scale in their Linux security advisories¹⁹. The scale is calculated by adding and subtracting points based on criteria relating to the risk and threat. Each advisory also mentions if the flaw is in a SUSE default package and gives a vulnerability type (such as “local privilege escalation”). SUSE fixes multiple issues in multiple software packages in the same advisory²⁰, sometimes only giving brief details of the lesser flaws. These additional issues mentioned in the advisories do not consistently get a separate severity rating or CVE name which can be confusing.

OpenPKG advisories gives a vulnerability type, and a “OpenPKG Specific” flag²¹. Vendors use these flags to show if the flaw has been introduced by the vendor or is in an upstream package.

Debian advisories published on the Debian web site do not contain any severity information. The versions of the advisories sent by email do contain the vulnerability type, a “local/remote” flag, and a “Debian-specific” flag²². In general, Red Hat finds it less useful to make a distinction between a local and a remote flaw – a local attacker is most likely an authenticated remote attacker.

Gentoo has a well-defined process for rating the severity of a security

18 <http://www.kb.cert.org/vuls/bymetric>

19 http://www.novell.com/linux/security/advisories/2004_43_cyrus_imapd.html

20 http://www.novell.com/linux/security/advisories/2004_03_sr.html

21 <http://marc.theaimsgroup.com/?l=bugtraq&m=110175142028823>

22 <http://marc.theaimsgroup.com/?l=bugtraq&m=110253088903481>

vulnerability and using that to give a target response time²³. The target response time is the time between an upstream fix being made publicly available and the publishing of the corresponding Gentoo advisory. The severity of the issue depends on a large number of factors including how widely a package is likely to be in use.

ISS X-Force rates the risk of issues that affect Linux and open source software according to the following scale:

High	Security issues that allow immediate remote or local access, or immediate execution of code or commands, with unauthorized privileges. Examples are most buffer overflows, backdoors, default or no password, and bypassing security on firewalls or other network components.
Medium	Security issues that have the potential of granting access or allowing code execution by means of complex or lengthy exploit procedures, or low risk issues applied to major Internet components. Examples are cross-site scripting, man-in-the-middle attacks, SQL injection, denial of service of major applications, and denial of service resulting in system information disclosure (such as core files).
Low	Security issues that deny service or provide non-system information that could be used to formulate structured attacks on a target, but not directly gain unauthorized access. Examples are brute force attacks, non-system information disclosure (configurations, paths, etc.), and denial of service attacks.

Table 3: ISS X-Force rating scale.

Comparison with the Microsoft impact scale

The system of classification that we use for issues in Red Hat products and services looks on the surface to closely match the current severity rating scale used by Microsoft²⁴. This isn' completely by coincidence. This parity has an immediate advantage for customers with heterogeneous environments who appreciate consistent risk classification, even if it isn' t always possible to directly compare vulnerabilities that have the same impact rating.

²³ http://www.gentoo.org/security/en/vulnerability-policy.xml#doc_chap3

²⁴ <http://www.microsoft.com/technet/security/bulletin/rating.msp>

Impact	Description
Critical	A vulnerability whose exploitation could allow the propagation of an Internet worm without user action
Important	A vulnerability whose exploitation could result in compromise of the confidentiality, integrity, or availability of users data, or of the integrity or availability of processing resources.
Moderate	Exploit ability is mitigated to a significant degree by factors such as default configuration, auditing, or difficulty of exploitation
Low	A vulnerability whose exploitation is extremely difficult, or whose impact is minimal

Table 4: Microsoft Impact Ratings (November 2002)

One example of the difficulty comparing issues is the difference in response to local denial of service attacks, where a simple program can be written which will crash a machine. On a Red Hat Enterprise Linux system, these attacks are treated as an important impact, since there is an expectation that users should not be able to completely crash the systems they are using (assuming that the systems administrator is using sensible limits for users, otherwise a one line fork-bomb program could have the same effect²⁵). Such expectations will vary on other operating systems. For example Windows operating systems still find it hard to escape from the legacy single user design.²⁶

Direct comparison of the number of flaws at a particular impact level is also misleading, even when Linux's flaws are normalized to the number of CVE-named issues. Open source software is easily examined to determine the number of distinct CVE-named issues to apply.

For example: a new version of Etherreal is released that fixes a buffer overflow, an integer overflow, and an out-of-bounds read. Even though all the flaws may have the same security consequence, they will get separate CVE names--one for each flaw type. With closed source software there is often no transparency as to what exactly has been fixed and in what way it was fixed, so it can be hard for independent researchers to work out the exact nature of every flaw. This makes it difficult to assign the right number of CVE names, and therefore unreliable to compare the number of vulnerabilities in open and closed source software by using only a simple numeric count

Comparisons of metrics also have to take into account the diverse package set available with popular Linux distributions, many of which may not be installed by default on the majority of machines. In fact, out of the twelve critical impact issues affecting Red Hat Enterprise Linux 3, three were for issues unlikely to affect all but a small number of users. Very few users set up public facing anonymous CVS or RSYNC servers. None of the twelve critical impact issues affected services enabled in a default installation.

²⁵ http://en.wikipedia.org/wiki/Fork_bomb

²⁶ http://www.theregister.co.uk/security/security_report_windows_vs_linux/#singleuser

Metrics for Red Hat Enterprise Linux 3

The Red Hat Security Response Team examined all security advisories for Red Hat Enterprise Linux 3 for the first 12 months following its release in order to evaluate the severity classification system.

We fixed 165 CVE-named issues by releasing 95 security advisories. For each CVE-named issue we used the classification system described earlier, taking into account the impact specific to the Red Hat Enterprise Linux 3 distribution. We did not adjust the ratings for Exec-Shield, as this was introduced towards the end of the period of analysis.

To find out if the classification scheme was close to our internal processes for responding to security issues, we looked at how long it took Red Hat to produce a fix for each issue. The start date was taken as when the issue was first known to the public. The end date was taken as when Red Hat released a security advisory that contained a fix for the issue. The difference between these two dates gives the “days of risk”, which is not adjusted to take account of weekends or holidays. A zero “days of risk” means “on the same day”. For all the issues in each impact classification, we took a median (the time to fix half the issues), as well as a metric of how many issues were fixed within a day (on the same day or the day after the issue was public). The results are shown in the table:

Classification	CVE matches	Advisory matches	Median “days of risk”	% with fixes within a day
Critical	12 (7%)	10 (11%)	0	58.00%
Important	50 (30%)	42 (44%)	5	44.00%
Moderate	59 (36%)	27 (28%)	16	23.00%
Low	44 (27%)	16 (17%)	31	22.00%
All severities	165	95	13	32.00%

Table 5: Time required to fix issues

Note that Red Hat often delays the release of low impact vulnerabilities until the next regularly scheduled Update release, or releases them with other issues in the same package. This has an effect on the data for low impact issues and for the total of all severities.

We can see that the severity classification scale closely matches how quickly we responded to issues over the period, with critical impact issues being fixed fastest. The chosen classification scheme therefore in practice provides a good measure of how worried Red Hat was about a vulnerability. Providing a prioritized 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.