



# Global Information Assurance Certification Paper

Copyright SANS Institute  
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

## Interested in learning more?

Check out the list of upcoming events offering  
"Security Essentials: Network, Endpoint, and Cloud (Security 401)"  
at <http://www.giac.org/registration/gsec>

# Development of Exploits for CVE-2000-0666

**Jon Lasser**

CVE-2000-0666 is the Linux `rpc.statd` remote root exploit used against Red Hat 6.x systems to great effect by the Ramen Worm. [1] Discovered in the wake of CAN-2000-0573 (the WU-FTPD site-exec format string vulnerability), the `rpc.statd` format string vulnerability has been exploited widely. Because this hole was discussed prior to the public availability of exploits, and because much about it has been posted to popular mailing lists, CVE-2000-0666 makes a fascinating case study regarding full-disclosure discussion of security holes. In this paper I will discuss the history of this bug with an eye on the effects of full disclosure on the course of events.

Because the full-disclosure / limited-disclosure disagreement is still a heated one, and because any history is informed by one's personal perspective, I must lay my biases on the table before delving into the story: I am, in general, strongly supportive of full disclosure. As the security administrator for my site, I have found 'hacker' tools invaluable for testing and verifying the security of my site, and I believe that full disclosure is sometimes the only way to force vendors to release necessary security fixes. That said, I agree that there are harmful effects to full disclosure, as the history of the `rpc.statd` exploit makes clear.

## Uncovering the Hole: April 2000 - June 2000

The `rpc.statd` format string vulnerability was first discussed on the Linux Security Audit Project mailing list in June of 2000. [2] However, Chris Evans (one of the primary instigators on that mailing list) brought up the subject of `rpc.statd` in a message on April 21st, asking what the `rpc.statd` daemon is responsible for. [3] The question was answered by Ed Franks and Olaf Kirch who explained that the `rpc.statd` daemon's primary responsibility is NFS file locking, though Kirch added that the protocol used is "bletcherous" and could be greatly simplified. [4] [5] Several days later, Matthew Kirkwood asked if `rpc.statd` needed to run as root, and Kirch answered that it probably did not need root privileges. [6] [7] In May, the thread was briefly picked up on by Anton Opperman, but an exchange with Kirch resulted in Opperman's agreement that he had confused `rpc.statd` and `rpc.rstatd`, a totally different service. [8] [9] [10]

On June 23rd (a Friday), a message was posted to Bugtraq with a remote root exploit for a wu-ftpd format string vulnerability. [11] Posted by "tf8," the exploit code (also allegedly from tf8) was dated October 15th 1999. There is of course no way to verify either the authorship of the exploit or the date it was written. Not surprisingly, a firestorm of posting ensued, with several fixes rapidly provided by different listmembers, including one by Daniel Jacobowitz. [12]

The following Tuesday, Chris Evans posted a message to the security-audit mailing list suggesting that `rpc.statd` was vulnerable to a format string bug similar to the `wu-ftpd` vulnerability. [13] (This message was apparently in response to a message from H. D. Moore, but that message was never archived and may have been private mail.) Daniel Jacobowitz responded that he could confirm the `rpc.statd` format string vulnerability was remotely exploitable. [14] Olaf Kirch responded to a question in that message, indicating that Jeff Uphoff was the original author and that H. J. Lu maintained most of the Linux kernel `nfs` daemon utilities. [15] Chris Evans responded to [15], indicating that he had a patch for the hole, to which Jeff Uphoff responded, asking for a copy of the patch, so that he could check it into the CVS repository. [16] [17] Daniel Jacobowitz also responded to [16], asking for a copy of the fix. Jacobowitz also indicated that he had a working exploit for Linux/PowerPC. [18]

By the end of June, the hole had been discovered and an exploit had been written but was not publicly available. A search of the standard mailing lists did not turn up any obvious attempts to exploit this hole, though Chris Evans had suggested in [13] that exploits would soon follow, due to the open source nature of the Linux `rpc.statd` code. (This was raised as an argument for full disclosure, not as an argument against open source software.)

The fix was committed to the CVS repository for the Linux NFS utilities on June 29. [19] The fix was present in the `nfs-utils` 0.1.9 release of July 3. [20] By conventional measures, then, the bug was fixed as of that date. However, because most Linux users rely on the distributors for updates, the fix was not widely disseminated among end-users.

### **Widening the Hole: July 2000 - August 2000**

Nothing seemed to happen regarding this hole for another two weeks, until Daniel Jacobowitz posted his non-functional exploit to the Bugtraq mailing list on July 16. [21] At this point, full disclosure must have seemed eminently reasonable: the current release of the software had the bug excised, and after all the exploit was not fully functional. Not only was the shell code to execute missing, but due to PowerPC architecture issues the code would only run in a debugger. Furthermore, the offsets hardcoded in the exploit were for the Debian Linux/PowerPC version, which is not widely used. Attached to the exploit was what Jacobowitz characterized as a "rant" attributed to Chris Evans regarding the problems with the `rpc.statd` implementation. In particular, Evans suggests that the level of privilege provided to `rpc.statd` was unnecessary, and that the daemon should be `chroot`'ed.

In his role as a Debian Linux maintainer, Jacobowitz apparently released a new Debian NFS package approximately one hour before his Bugtraq post. The current release version of Debian did not include `rpc.statd` and was not vulnerable, but the two development versions were vulnerable and fixed packages were released. [22] Due to the lack of accurate datestamping in the SecurityFocus Bugtraq archive, I can't confirm the timing of this with utter certainty; it is enough to say that the fix was released at virtually the same day as the full disclosure of the bug.

The following day, Red Hat released an updated NFS package that fixed the bug. [23] Other Linux distributors soon followed, including Mandrake and Caldera. [24] [25] By the 21st of July, clearly spurred by the disclosure of the bug, virtually every vulnerable Linux distribution had a patched version available for download and had publicly announced their fix. Full disclosure had done its job, and it was now the users' responsibility to fix their systems.

The first known exploit for Linux on the x86 architecture was posted to the Bugtraq list on August 1 by jdoing@teleline.es. [26] Doing followed up the next day by documenting how to find the offset necessary for the exploit to provide a remote root shell, a somewhat technical procedure that the average script kiddie might not have been able to accomplish without such help. [27] Doing's exploit was explicitly based on Jacobowitz's Linux/PowerPC code, gently suggesting the danger of full disclosure.

A much more user-friendly exploit, targeting all Red Hat 6.x systems but also permitting manual offsets, was posted by "ron ln" to Bugtraq on August 4. [28] This exploit, named 'statdx.c,' has been seen in the wild with great frequency, and based on my observation was the tool most often used by hackers through the end of December. A follow-up message from ron ln suggests that his version of the exploit was completed at approximately the same time as Doing's exploit, though he claims to have started it on July 24. [29] I have not examined the code closely enough to see if they share a common heritage, but it is clear that ron ln's exploit was based at least in principle upon Jacobowitz's disclosure.

In [29], ron ln defended his disclosure by noting that there was already working exploit code available and that 'crippled' exploit code is usually 'fixed' within days of posting. [26] would seem to validate this opinion.

Finally, on August 17, CERT posed their advisory regarding the rpc.statd exploit. [30] The CERT advisory provided additional information about blocking rpc.statd and identifying exploit attempts. The announcement referenced only the Debian, Red Hat, and Caldera announcements, and added no useful vendor-specific information. (It did mention that non-Linux systems with different rpc.statd daemons were not affected. This should not have been news to anyone.) The announcement came almost exactly one month after the exploits were patched by the referenced vendors, and approximately two weeks after a user-friendly exploit was released.

Although this inaction makes it tempting to place blame on CERT for whatever successful exploitation occurred, that would be unrealistic in several regards: first, vendor announcements had been released prior to the availability of any working x86 exploit; second, successful exploitation accelerated following the CERT announcement.

## **Digging Deeper: September 2000 - February 2001**

In the following months, the rpc.statd hole was exploited widely. Although ron ln released an updated version of the exploit, little changed. [31] On December 15,

marc@zounds.net posted excerpts from a wrapper script (dated October 30th) that was used to automate the exploit. [32] At my own site, all of our systems had long since been patched, though customer systems continued to be affected.

In mid-January, the Ramen worm exploited the rpc.statd hole, along with other commonly-used remote root exploits. The most recently repaired of the exploits was the Red Hat 7 LPRng exploit, patched on September 26. [33] Despite the age of these exploits, the Ramen worm was notably successful, hinting at the wealth of unpatched systems running on the Internet.

The hole continues to be exploited, though the shell code currently being used does not trip any rules available for the Snort IDS other than generic x86 code rules, as can be seen in this excerpt from my logs:

```
[**] IDS428/portmap-listing-111 [**]
02/25-08:47:18.295228 0:30:A3:F1:B8:A0 -> 0:3:6B:FA:6:C2 type:0x800
len:0x62
AAA.BBB.160.230:973 -> CCC.DDD.0.23:111 TCP TTL:50 TOS:0x0 ID:13599
IpLen:20 DgmLen:84 DF
***AP*** Seq: 0xB9C324DC Ack: 0x416627BA Win: 0x7D78 TcpLen: 20

[**] IDS428/portmap-listing-111 [**]
02/25-08:47:18.295234 0:3:6B:FA:6:C2 -> 0:10:5A:E7:6A:B3 type:0x800
len:0x62
AAA.BBB.160.230:973 -> CCC.DDD.0.23:111 TCP TTL:49 TOS:0x0 ID:13599
IpLen:20 DgmLen:84 DF
***AP*** Seq: 0xB9C324DC Ack: 0x416627BA Win: 0x7D78 TcpLen: 20

[ . . . ]

[**] IDS428/portmap-listing-111 [**]
02/25-08:47:24.495049 0:30:A3:F1:B8:A0 -> 0:3:6B:FA:6:C2 type:0x800
len:0x6E
AAA.BBB.160.230:817 -> CCC.DDD.70.87:111 TCP TTL:50 TOS:0x0 ID:19011
IpLen:20 DgmLen:96 DF
***AP*** Seq: 0xBA0A822A Ack: 0xC2D8E587 Win: 0x7D78 TcpLen: 32
TCP Options (3) => NOP NOP TS: 44212029 638435012

[**] IDS428/portmap-listing-111 [**]
02/25-08:47:24.495338 0:3:6B:FA:6:C2 -> 0:1:29:0:A:F8 type:0x800
len:0x6E
AAA.BBB.160.230:817 -> CCC.DDD.70.87:111 TCP TTL:49 TOS:0x0 ID:19011
IpLen:20 DgmLen:96 DF
***AP*** Seq: 0xBA0A822A Ack: 0xC2D8E587 Win: 0x7D78 TcpLen: 32
TCP Options (3) => NOP NOP TS: 44212029 638435012

[**] IDS428/portmap-listing-111 [**]
02/25-08:47:24.535493 0:30:A3:F1:B8:A0 -> 0:3:6B:FA:6:C2 type:0x800
len:0x6E
AAA.BBB.160.230:818 -> CCC.DDD.70.88:111 TCP TTL:50 TOS:0x0 ID:19016
IpLen:20 DgmLen:96 DF
***AP*** Seq: 0xBAB92487 Ack: 0xC2B83E14 Win: 0x7D78 TcpLen: 32
TCP Options (3) => NOP NOP TS: 44212033 638435017
```

[\*\*] IDS428/portmap-listing-111 [\*\*]  
02/25-08:47:24.536116 0:30:A3:F1:B8:A0 -> 0:3:6B:FA:6:C2 type:0x800  
len:0x6E  
AAA.BBB.160.230:819 -> CCC.DDD.70.89:111 TCP TTL:50 TOS:0x0 ID:19018  
IpLen:20 DgmLen:96 DF  
\*\*\*AP\*\*\* Seq: 0xBA7152CA Ack: 0xC29EEAFD Win: 0x7D78 TcpLen: 32  
TCP Options (3) => NOP NOP TS: 44212033 638435017

[\*\*] IDS428/portmap-listing-111 [\*\*]  
02/25-08:47:24.536129 0:3:6B:FA:6:C2 -> 0:1:29:0:A:F8 type:0x800  
len:0x6E  
AAA.BBB.160.230:819 -> CCC.DDD.70.89:111 TCP TTL:49 TOS:0x0 ID:19018  
IpLen:20 DgmLen:96 DF  
\*\*\*AP\*\*\* Seq: 0xBA7152CA Ack: 0xC29EEAFD Win: 0x7D78 TcpLen: 32  
TCP Options (3) => NOP NOP TS: 44212033 638435017

[ . . . ]

[\*\*] IDS10/portmap-request-rstatd [\*\*]  
02/25-08:48:19.422012 0:3:6B:FA:6:C2 -> 0:1:29:0:A:F8 type:0x800  
len:0x62  
AAA.BBB.160.230:691 -> CCC.DDD.70.89:111 UDP TTL:49 TOS:0x0 ID:39329  
IpLen:20 DgmLen:84  
Len: 64

[\*\*] IDS10/portmap-request-rstatd [\*\*]  
02/25-08:48:19.823288 0:30:A3:F1:B8:A0 -> 0:3:6B:FA:6:C2 type:0x800  
len:0x62  
AAA.BBB.160.230:699 -> CCC.DDD.70.90:111 UDP TTL:50 TOS:0x0 ID:39337  
IpLen:20 DgmLen:84  
Len: 64

[\*\*] IDS10/portmap-request-rstatd [\*\*]  
02/25-08:48:20.058655 0:30:A3:F1:B8:A0 -> 0:3:6B:FA:6:C2 type:0x800  
len:0x62  
AAA.BBB.160.230:703 -> CCC.DDD.70.88:111 UDP TTL:50 TOS:0x0 ID:39342  
IpLen:20 DgmLen:84  
Len: 64

[\*\*] IDS362/shellcode-x86-nops-udp [\*\*]  
02/25-08:48:20.336247 0:30:A3:F1:B8:A0 -> 0:3:6B:FA:6:C2 type:0x800  
len:0x1EA  
AAA.BBB.160.230:708 -> CCC.DDD.70.67:929 UDP TTL:50 TOS:0x0 ID:39347  
IpLen:20 DgmLen:476  
Len: 456

[\*\*] IDS362/shellcode-x86-nops-udp [\*\*]  
02/25-08:48:20.336288 0:3:6B:FA:6:C2 -> 0:1:29:0:A:F8 type:0x800  
len:0x1EA  
AAA.BBB.160.230:708 -> CCC.DDD.70.67:929 UDP TTL:49 TOS:0x0 ID:39347  
IpLen:20 DgmLen:476  
Len: 456

[\*\*] IDS10/portmap-request-rstatd [\*\*]  
02/25-08:48:20.688563 0:30:A3:F1:B8:A0 -> 0:3:6B:FA:6:C2 type:0x800  
len:0x62

```
AAA.BBB.160.230:715 -> CCC.DDD.70.88:111 UDP TTL:50 TOS:0x0 ID:39352
IpLen:20 DgmLen:84
Len: 64
```

```
[**] IDS10/portmap-request-rstatd [**]
02/25-08:48:20.688565 0:3:6B:FA:6:C2 -> 0:1:29:0:A:F8 type:0x800
len:0x62
AAA.BBB.160.230:715 -> CCC.DDD.70.88:111 UDP TTL:49 TOS:0x0 ID:39352
IpLen:20 DgmLen:84
Len: 64
```

```
[**] IDS362/shellcode-x86-nops-udp [**]
02/25-08:48:20.778767 0:3:6B:FA:6:C2 -> 0:1:29:0:A:F8 type:0x800
len:0x1EA
AAA.BBB.160.230:716 -> CCC.DDD.70.88:929 UDP TTL:49 TOS:0x0 ID:39354
IpLen:20 DgmLen:476
Len: 456
```

```
[**] IDS10/portmap-request-rstatd [**]
02/25-08:48:20.891766 0:30:A3:F1:B8:A0 -> 0:3:6B:FA:6:C2 type:0x800
len:0x62
AAA.BBB.160.230:719 -> CCC.DDD.70.87:111 UDP TTL:50 TOS:0x0 ID:39356
IpLen:20 DgmLen:84
Len: 64
```

(Note that the snort rule confuses `rpc.statd` and `rpc.rstatd`, as did Anton Opperman. Personal investigation on my part has confirmed that the `portmap-request-rstatd` rule is triggered by `rpc.statd` traffic.)

## Conclusions

The `rpc.statd` format string vulnerability is still a serious headache for system administrators nearly eight months after it was first disclosed. The history of the development of exploits for this hole offers ample ammunition both to proponents and opponents of full disclosure. Partial disclosure --- that of only partially-functional exploits --- is a clear failure and should be considered the equivalent of full disclosure.

In favor of full disclosure, it can be said that the Linux distributors rapidly fixed the mistake only when it was demonstrated to be exploitable with publicly-available code. It can also be said that marc@zounds.net's principled refusal to release the `std.pl` script did not make it visibly more difficult for hackers to scan sites. Against full disclosure, the hole is not known to have been exploited until after a proof-of-concept exploit was released, and the first fully functional exploit was based on the same code. Subsequent implementations of the exploit may or may not have been developed independently, but the damage was clearly done. Furthermore, the CERT announcement was released late enough to be fundamentally useless.

The history of this exploit certainly demonstrates the importance of regular patching of systems, disabling unused services, and firewalling off necessary internal services from the outside world. The vendors did an admirable job of responding to this incident, but the end-user system administrators still have lessons to learn.

---

## References

- [1] Lestat, Morgan. "The Ramen Worm and its use of rpc.statd, wu-ftpd and LPRng Vulnerabilities in Red Hat Linux. February 7 2001. URL: <http://www.sans.org/infosecFAQ/malicious/ramen.htm>
- [2] Linux Security Audit Project. URL: <http://lsap.org/>
- [3] Evans, Chris. "rpc.statd." Linux Security Audit Project mailing list. 21 April 2000. URL: <http://www.securityfocus.com/archive/59/56392>
- [4] Franks, Ed. "Re: rpc.statd." Linux Security Audit Project mailing list. 22 April 2000. URL: <http://www.securityfocus.com/archive/59/56453>
- [5] Kirch, Olaf. "Re: rpc.statd." Linux Security Audit Project mailing list. 22 April 2000. URL: <http://www.securityfocus.com/archive/59/56458>
- [6] Kirkwood, Matthew. "Re: rpc.statd." Linux Security Audit Project mailing list. 25 April 2000. URL: <http://www.securityfocus.com/archive/59/56795>
- [7] Kirch, Olaf. "Re: rpc.statd." Linux Security Audit Project mailing list. 25 April 2000. URL: <http://www.securityfocus.com/archive/59/56805>
- [8] Opperman, Anton. "Re: rpc.statd." Linux Security Audit Project mailing list. 15 May 2000. URL: <http://www.securityfocus.com/archive/59/60074>
- [9] Kirch, Olaf. "Re: rpc.statd." Linux Security Audit Project mailing list. 15 May 2000. URL: <http://www.securityfocus.com/archive/59/60077>
- [10] Opperman, Anton. "Re: rpc.statd." Linux Security Audit Project mailing list. 15 May 2000. URL: <http://www.securityfocus.com/archive/59/60090>
- [11] tf8. "WuFTPd: Providing \*remote\* root since at least 1994." Bugtraq mailing list. 23 June 2000. URL: <http://www.securityfocus.com/archive/1/66367>
- [12] Jacobowitz, Daniel. "Re: WuFTPd: Providing \*remote\* root since at least 1994." Bugtraq mailing list. 23 June 2000. URL: <http://www.securityfocus.com/archive/1/66566>
- [13] Evans, Chris. "Re: format bugs, in addition to the wuftp bug." Linux Security Audit Project mailing list. 27 June 2000. URL: <http://www.securityfocus.com/archive/59/67156>
- [14] Jacobowitz, Daniel. "Re: format bugs, in addition to the wuftp bug." Linux Security Audit Project mailing list. 27 June 2000. URL: <http://www.securityfocus.com/archive/59/67202>

- [15] Kirch, Olaf. "Re: format bugs, in addition to the wuftp bug." Linux Security Audit Project mailing list. 28 June 2000. URL: <http://www.securityfocus.com/archive/59/67219>
- [16] Evans, Chris. "Re: format bugs, in addition to the wuftp bug." Linux Security Audit Project mailing list. 28 June 2000. URL: <http://www.securityfocus.com/archive/59/67220>
- [17] Uphoff, Jeff. "Re: format bugs, in addition to the wuftp bug." Linux Security Audit Project mailing list. 28 June 2000. URL: <http://www.securityfocus.com/archive/59/67237>
- [18] Jacobowitz, Daniel. "Re: format bugs, in addition to the wuftp bug." Linux Security Audit Project mailing list. 28 June 2000. URL: <http://www.securityfocus.com/archive/59/67414>
- [19] nfs-utils CVS repository. URL: <http://cvs.sourceforge.net/cgi-bin/cvsweb.cgi/nfs-utils/utils/statd/log.c?cvsroot=nfs> (2001 Feb 26)
- [20] nfs-utils 0.1.8.2 - 0.1.9 patch. URL: [HREF="http://download.sourceforge.net/nfs/nfs-utils-0.1.8.2-0.1.9.diff.gz"](http://download.sourceforge.net/nfs/nfs-utils-0.1.8.2-0.1.9.diff.gz) (2001 Feb 26)
- [21] Jacobowitz, Daniel. "Lots and lots of fun with rpc.statd." Bugtraq mailing list. 16 July 2000. URL: <http://www.securityfocus.com/archive/1/70306>
- [22] Jacobowitz, Daniel. "New Debian nfs-common packages released." Debian security-announce mailing list. 16 July 2000. URL: <http://lists.debian.org/debian-security-announce-00/msg00019.html>
- [23] Red Hat Linux. "[RHSA-2000:043-02] Updated package for nfs-utils available." Bugtraq mailing list. 17 July 2000. URL: <http://www.securityfocus.com/archive/1/70417>
- [24] MandrakeSoft. "MandrakeSoft Security Advisory MDKSA-2000:021 : nfs-utils." 18 July 2000. URL: <http://www.linux-mandrake.com/en/security/2000/MDKSA-2000-021.php?dis=7.1> (2001 Feb 26)
- [25] Caldera Systems, Inc. "Security Advisory: rpc.statd is not a problem on OpenLinux." Bugtraq mailing list. 19 July 2000. URL: <http://www.securityfocus.com/archive/1/70895>
- [26] jdoing@teleline.es. "rpc.statd remote root xploit for linux/x86." Bugtraq mailing list. 1 August 2000. URL: <http://www.securityfocus.com/archive/1/73193>
- [27] jdoing@teleline.es. "rpc.statd remote root xploit for linux/x86 (little fix)." Bugtraq mailing list. 2 August 2000. URL: <http://www.securityfocus.com/archive/1/73365>
- [28] ron ln. "Redhat Linux 6.x remote root exploit." Bugtraq mailing list. 4 August 2000. URL: <http://www.securityfocus.com/archive/1/74148>

[29] ron ln. "the rpc.statd exploit." Bugtraq mailing list. 7 August 2000. URL: <http://www.securityfocus.com/archive/1/74255>

[30] CERT. "Advisory CA-2000-17 Input Validation Problem in rpc.statd" 18 August 2000. URL: <http://www.cert.org/advisories/CA-2000-17.html> (2001 February 26)

[31] ron ln. "statdx2 - linux rpc.statd revisited." Bugtraq mailing list. 11 October 2000. URL: <http://www.securityfocus.com/archive/1/138850>

[32] marc@zounds.net. "More info regarding: std.pl, the rpc.statd linux mass rooter." Incidents mailing list. 15 December 2000. URL: <http://www.securityfocus.com/archive/75/151194>

[33] Red Hat Linux. "RHSA-2000:065-06" Red Hat Linux web site. 4 October 2000. URL: <http://www.redhat.com/support/errata/RHSA-2000-065.html> (2001 February 26)

© SANS Institute 2000 - 2002, Author retains full rights.