# **Wormsign: Predicting the Next Outbreak**

By Joe Stewart, GCIH

It's no secret that Internet worms are becoming more and more advanced. With each incarnation, worms are able to disrupt the Internet as never before. Up to this point, the damage has been relatively small in comparison to what a truly nasty worm could do. Sure, the denial-of-service effect of Slammer put some companies out of commission for a day or two and stopped some ATMs and other services from working. But for the most part simply blocking UDP port 1434 on core routers quickly contained it. The worm writer witnessed this, and you can bet the next worm will not be so easily countered. That's why it is important to begin preparing now for the next outbreak.

Worms have been around for a couple of decades, but their frequency of occurrence is rapidly increasing due to two things: the population explosion of vulnerable hosts and the knowledge needed to write a worm is more widely available; literally a few clicks away from someone with the malice of forethought to write one. Much of the work needed to write a worm is actually done for them, by legitimate security researchers. Worms are an unwanted side effect of the full-disclosure policy of many security institutions. However, the additional boost they get also becomes their weakness, as the worm writers lose the element of surprise.

All of the worms in the past 3 years have followed a particular pattern. They share common factors that contributed to their development. These factors are like pieces of a puzzle. If all the pieces are not present, worm emergence is unlikely in a given scenario.

# Factor 1: Security Advisory

The first factor needed is an advisory released by a vendor or an independent researcher. An advisory is considered serious enough to be republished by CERT and other institutions.



Factors in Worm Development

## Factor 2: Exploit Code

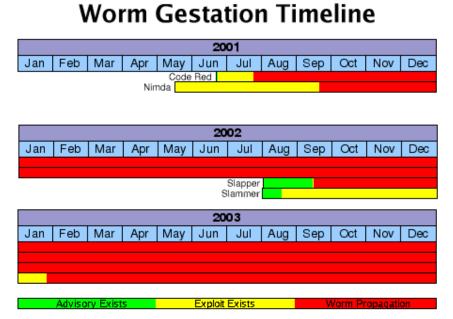
The second factor is working exploit/proof-of-concept code, either by the researcher or a third party. Worm writers could develop the exploit code themselves, but the time invested in exploit development can be considerable; perhaps even more so than the actual spreading code of the worm. It's easier to wait for someone else to do the initial legwork. So far this seems to have been the case for each worm in the past five years.

# Factor 3: Targets

A third and very important factor in worm propagation is target selection. Sometimes a target may be several versions of the same vulnerable program. In these cases, the worm must find common hooks in the code it is overflowing since the environment where the worm gains control may change with each release. The program might even run on multiple operating systems; in which case the worm code might need to be radically different in order to execute on each platform. The more a worm has to adapt to fit different versions, the more code it must contain. This bloat is detrimental to the rapid propagation of the worm. It also requires greater expertise and a longer testing cycle. Worm writers are then more likely to look for vulnerabilities in software that that is single platform and prolific in major versions. For this reason IIS and MSSQL have been prime targets in more than one worm. While Apache is more widely used than IIS, it exists on so many platforms that writing a single buffer overflow exploit that would be guaranteed to work on a large enough subset of them would be extremely difficult.

### Factor 4: Time

The final factor in worm development is, like any other software development project; time. A worm writer needs time to code and test the worm before releasing it in the wild. The length of time varies; however, the past four most notable worms, Code Red, Nimda, Slapper and Slammer all had a gestation period of one to six months. Below is a timeline of the development of all four worms.



The timeline extends into the future, because it is a good bet that all four worms will still be alive and well at the end of 2003 and beyond. It is important to note that the worm writer must try and release working code as soon as possible. The longer they wait, the more systems that will have been upgraded or patched. There is a definite time frame in which worms are viable. We need to look back in that time frame and guess where the next attack will come from.

If we were to try and predict the next vector of worm infection, we should look at the past six months' advisories and take note of how each one fits into the typical worm mold. Here are some notable releases for the past six months taken from CERT:

- <u>CA-2003-01</u> :Buffer Overflows in ISC DHCPD Minires Library
- <u>CA-2003-02</u> :Double-Free Bug in CVS Server
- <u>CA-2003-03 :Buffer Overflow in Windows Locator Service</u>
- <u>CA-2002-36</u>: Multiple Vulnerabilities in SSH Implementations
- <u>A-2002-34: Buffer Overflow in Solaris X Window Font Service</u>
- <u>CA-2002-33: Heap Overflow Vulnerability in Microsoft Data Access Components</u>
  (MDAC)
- <u>CA-2002-31: Multiple Vulnerabilities in BIND</u>
- <u>CA-2002-29: Buffer Overflow in Kerberos Administration Daemon</u>
- <u>A-2002-26: Buffer Overflow in CDE ToolTalk</u>

These are serious flaws, but are they potential sources for new worms? Applying the worm development factors to recent vulnerabilities, we can eliminate some or all of them as new worm vectors. Below are the CERT advisories linked with the most current information available (which is of course subject to change without notice):

Advisory	Exploit	Adequate Targets	Time Since Advisory
Buffer Overflows in DHCPD	Yes	Yes	1 Month
Double-Free Bug in CVS	No	No	1 Month
Buffer Overflow in Windows	Private	No	1 Month
Locator Service			
Multiple Vulnerabilities in SSH	No	No	2 Months
Implementations			
Buffer Overflow in Solaris X	Yes	No	3 Months
Window Font Service			
Heap Overflow Vulnerability in	No	No	3 Months
Microsoft Data Access Components			
(MDAC)			
Multiple Vulnerabilities in BIND	No	Yes	3 Months
Buffer Overflow in Kerberos	Yes	No	4 Months
Administration Daemon			
Buffer Overflow in CDE ToolTalk	No	No	6 Months

At the time of this writing, the only reasonable current worm vector is the DHCPD vulnerability. It fills all the criteria need for worm development. If it were to be the next worm, it may not be as prolific as an attack on IIS or MSSQL, but I believe there are enough broadband users with a Linux firewall who are running vulnerable versions of DHCPD to achieve critical mass. This doesn't necessarily mean there "will" be a DHCPD worm. However, the elements necessary for a DHCPD worm are present.

#### Conclusion

Because of the predictability of worm development, we should not be caught totally unprepared. While there is little the average sysadmin can do to mitigate the effect of a worm on the Internet as a whole, one should plan for protecting their own network as best as possible. An outbreak on the Internet is an inconvenience; an outbreak on your internal network is potentially devastating. In order to mitigate this risk, you should do the following:

- Employ a patch management solution to track software and versions within the organization and follow major patch release mailing lists.
- Perform regular vulnerability scans
- Ensure all security devices are in line with corporate security policy

- Deploy anti-virus on the desktop and the mail gateway
- Have staff to monitor the network and alert you of problems 24 hours a day

This list could go on and on, as protecting your network against worms is essentially just following general security "best practices". However, one countermeasure - 24x7 monitoring, can be especially useful. If you don't have a 24-hour staff monitoring your network, consider outsourcing this service to a managed security services provider (MSSP). They will not only be able to alert you at the first sign of trouble, but can do so at any time, day or night. You can also utilize their expertise in dealing with the outbreak. In any event, make sure you have a plan in place, because it's not a matter of if the next worm will strike, but when.

#### About the Author

Joe Stewart, GCIH, is a Senior Information Security Analyst with managed security services provider <u>LURHQ Corporation</u>. He may be reached at <u>jstewart@lurhq.com</u>.