musings

by Rik Farrow
<*rik@spirit.com*>

Rik Farrow provides UNIX and Internet security consulting and training. He is the author of *UNIX System Security* and *System Administrator's Guide to System V*.

Memorial Day in the U.S. has come and gone, with the usual ceremonies to remember war dead as well as the survivors. I tacked a visit to the Vietnam War Memorial onto a family visit in Washington, D.C., not so much because of Memorial Day, but rather because I had forgotten it was Memorial Day, and I had never seen the complex.

At the time of the Vietnam War, I was adamantly opposed to sending troops there. It was their civil war, and we wound up fighting our own WWII ally, Ho Chi Minh, under the illusion that if Vietnam became Communist, other countries would fall under Communism like dominoes. In no time, the Australians would have been fighting guerrillas hiding within the dense jungles in their arid interior. Right. Instead, millions of Vietnamese have suffered, and many have died, along with millions of American casualties.

What really struck me, as I walked with thousands of others past the Wall, was just how many names there were. Over 50,000 Americans died, but you don't really know how many people that is until you watch the Wall grow taller, with more names in each column, going on and on until I was numb. Rolling Thunder, a group that demonstrates for the return of Vietnam Missing in Action, was there. Their parade on Sunday consisted of over a hundred thousand motorcyclists rolling on a thunderous parade route around the Capital mall. Yet the antipathy that had existed between the vets and the anti-war protestors during the seventies appeared to have vanished.

My own feelings today about armed forces are that I am glad they are on my side. Their job is still a necessary one, and often a dangerous one, at that. If you want to get a better understanding of modern warfare, what it is like to be under fire, as well as to comprehend a failure in foreign policy, read *Black Hawk Down*, by Mark Bowden. I often wondered at the silence of vets when asked about their combat experiences, and this book explains, through the eyes of combatants, why people just don't talk about these things. My heart goes out to any veteran of any conflict anywhere, with the prayer that someday we can eliminate organized death and mayhem.

DNS, the Forgotten Story

Speaking of heroes of less violent conflicts, DNS has become a casualty quite frequently of late. Or perhaps I should say that DNS has become one of the most common ways to break into some UNIX and Linux systems remotely. And, in an embarrassing turn of events, one of the security extensions to DNS paved the way for this buffer overflow.

The latest buffer overflow is not the only one. Up until two years ago, most Linux systems were distributed with BIND version 4.9.6, with FAKE-IQUERY enabled. While many other versions of UNIX used the same version, they did not support FAKE-IQUERY. The exploit for this works by sending a request that contains a value that overflows an internal buffer in named, providing a root-owned shell. CA-98.14 covers this issue in BIND, as well as several other denial-of-service attacks, and cache corruption.

The newer releases of BIND are much more resistant to cache corruption, as they only accept responses that match outstanding requests unlike the old version, which would accept an unsolicited response. The newer cache-corruption tools must first collect a request ID, send a request for the site to spoof to the victim DNS server, and then send the phony answer with the correct request ID (based on the previously collected request ID). The OpenBSD version of named randomizes request IDs, making this much more difficult to accomplish.

What has been causing a lot of problems is a buffer overflow in BIND 8.2. If you have installed patch 5 and are running BIND 8.2P5 (or earlier than 9.2 or later versions), you do not have this problem. You can determine the version of BIND that you are running by executing named -v locally, or using dig to query a nameserver remotely:

```
dig bear.spirit.com version.bind chaos txt
```

Look in the ANSWER section of the result for VERSION.BIND, and hope to see something like "8.2.2-P5."

There has been a lot of confusion around one particular exploit (apparently written by a small group called ADM crew), as it takes advantage of a new resource record (RR) named NXT. Until recently, I had no idea what NXT meant, although I correctly assumed that you could pronounce it "next." NXT is part of the DNS security extensions, or DNSSEC, and you can find out more about this by reading RFC 2535 (<*http://www.faqs.org/rfcs/rfc2535.html*>). I am looking for someone who can provide more detailed information about DNSSEC, but I will share some of what I learned from the RFC and from some email exchanges.

No More Corruption

The only sure way to end DNS cache corruption is to create DNS servers that can include digital signatures with their answers, and servers that can validate those signatures when they are received. BIND 8.2 includes these features, supported through the use of three new RRs — SIG, PUB, and NXT. SIG RRs provide digital signatures for named resources. The signatures are multiline data structures that define what is being signed and how, and include the signature itself.

The PUB RRs provide the public key for the DNS server itself. For this to really work, a server's PUB key must be signed by a DNS server in a super zone, and the receiving server must be configured with public keys for the root servers, so that the chain of public keys can be authenticated. Note that this is a little like a PKI. There is a project going on in Europe to set up DNSSEC in Sweden, Germany, and the Netherlands by the middle of 2001, and I am very interested in seeing the results. One thing I find myself wondering about is all of those asymmetric encryptions being done to verify signatures, and how much load this will add to a busy DNS server.

The NXT RR actually stands for "non-existence record." I had to read through this section carefully, because it is not intuitively obvious (at least for me) how you could cover all non-existing records in your DNS database. The concept is actually much simpler, as it appears that instead of sending back an error when information is requested that does not appear in the zone database, two NXT RRs are sent back. The NXT RRs provide proof that the requested record really does not exist. Please read section 5 of RFC 2535 for details.

So, if you don't even know about NXT records, how can you be vulnerable to an attack using them? Well, that's easy enough. Someone will send you a NXT record with a buffer overflow in one of the fields. That way, you don't need to know anything about NXT RRs to be vulnerable. Note that vulnerable versions of BIND were shipped not only by Linux distributors but also by several BSDs, Sun, IBM, HP, SCO, and Data General. (See<*http://www.cert.org/advisories/CA-99-14-bind.html*>.)

I had to find a version of the exploit, plus RFC 2535, before I could begin to understand all of this. You can find the version I examined at<*ftp://ftp.technotronic.com/unix/nameserver-exploits/t666.c*>. This version has been modified so that it will not work without some changes to the shellcode, unless you attack a system that you control and copy a shell to `/adm/sh`. This version also does not explain how to tickle the remote DNS server into querying your evil server either, but I imagine that cache corruption tools like `erect` or `jizz` would help here.

An interesting aspect of t666.c is that the comments claim to be able to break out of a `chroot` jail. The `chroot()` system call sets a string in the process's user area that changes the process's root. Once this system call has been executed by root, all paths appear relative to the new root. If you are interested in how you would set this up for Linux and the patched version of BIND, you can read <*http://metalab.unc.edu/pub/Linux/docs/HOWTO/Chroot-BIND-HOWTO*>. Although I was not willing to decode most of the shellcode, the usual technique to escape a chrooted environment is to find an open file descriptor for a directory outside of the jail, and to use that to escape. I believe that BIND 8.2P5 fixes this by closing all file descriptors opened before the change root.

Ubiquitous

One of my favorite words, ubiquitous means "ever-present." The vulnerable version of BIND, 8.2, also appeared to be nearly ubiquitous. A survey of DNS servers, using the inverse in-addr.arpa tree by Bill Manning

(<*http://www.isi.edu/~bmanning/in-addr-data.html*>) shows that as of second quarter 2000, 13% of all DNS servers — about 19,000 of them, that is — were running 8.2.

But CERT pointed out in CA-2000-03 that this is only the tip of the iceberg. Many default installations of Linux, BSD, or UNIX will start up `named`. The unconfigured `named` will function as a local server, as well as be remotely detectable by scanning for an open port 53. Just try to imagine how many people have installed Red Hat 6 (with BIND 8.2), and left `named` running, just waiting for someone to notice the open port 53, verify the BIND version (if they even bother), and run the exploit.

CERT also pointed out another thing I find sadly funny. Besides the usual tricks done after a completed break-in (install rootkits, set up backdoors, edit logs), attackers have taken to installing patched versions of BIND. I do not consider this an act of goodwill, but merely another way to hide the intrusion (as well as making the system less likely to be "owned" by yet another attacker).

Sysadmins, sweep your networks! Commercial tools will find the buggy version of BIND, as well tools like `nmap` set to scan for an open port 53. The use of a properly configured firewall (one that, at the very least, limits incoming connections to known servers) will also do a lot to contain the damage. Note that FireWall-1 v4 (with no patches) permitted access to port 53 TCP/UDP by default, and you should change this.

VMware

I finally got a chance to install the 2.0 version of VMware on my little notebook. VMware allows me to run NT4 under Linux, and the new version does run faster. It also has a suspend feature, so you can suspend NT and resume it later without having to go through the whole reboot process. I like this: until now, before I could start teaching I had to boot first Linux, then VMware and NT4.

I have also been playing around with Java again. I am not particularly fond of PowerPoint, and I've wanted to include animations in my presentations for a long time. I downloaded a version of Java 1.1 from Blackdown.org and started writing my own display tool. Right now it runs simple animations (labeled boxes move across the screen, allowing me to demonstrate the infamous TCP three-way handshake, as well as scanning), and I can display images. Now I need to add keyboard control, as well as the ability to display text, and I might have something fun to use.

On the Microsoft front, not much has happened. Well, there are about 150 class-action lawsuits on behalf of people who had to pay for a Windows license when they bought their computer to run Linux, *BSD, or BeOS. And there have been some wonderful worms and viruses floating about, such as ILOVEYOU! and the resume virus. Microsoft had released its thirty-ninth security advisory of the year (already heading for a record-breaking year), which leads me to suggest the following to the Clinton administration:

- Microsoft should be broken up into at least two separate companies for national-security reasons.
- My argument is simple. In its present state, Microsoft can continue to embed features in its operating system that support not only the applications it writes, but copious exploits as well. Who could have imagined that an operating system and its related applications would, in the Internet age, make it so simple to download and execute code, jeopardizing the security of most desktop systems? Breaking up Microsoft should not be considered punishment, or bad for the U.S. economy, but an action taken in the hope that Microsoft will be forced to start designing secure application-language interfaces with published specifications.

I know, fat chance, but I felt I had to say something.