;login:
THE MAGAZINE OF USENIX & SAGE
December 2001 • Volume 26 • Number 8

inside:

SECURITY

Musings

By Rik Farrow

USENIX & SAGE
The Advanced Computing Systems Association &
The System Administrators Guild

musings

by Rik Farrow

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*.

rik@spirit.com

Funny how things run in cycles. Some economists and analysts had convinced themselves (and a lot of other people) that the business cycle of boom and bust had passed with the adoption of modern computer technology. And sniffing was supposed to be a thing of the past with the adoption of switches.

Password sniffing was endemic in 1996. Attackers loved installing password sniffers at ISPs. After all, an ISP sees all the traffic coming from many sites, and any login-name/password combination collected here will work through a firewall. The only thing preventing success would be source address access control through TCP wrappers or other software that has been properly configured.

ISPs (the smart ones) changed the architecture of their internal networks so that servers sat on their own subnets and didn't get to listen to all network traffic, just local traffic (which still included lots of POP passwords). Then people started installing switches, mainly to improve performance, and secondarily to help with security. Switches do what the names implies. Instead of broadcasting, like hubs, switches are supposed to provide a "switched" connection between ports, so that packet collisions are greatly reduced, and sniffing other systems' traffic becomes impossible.

Well, that was the hope. Truth is, switches may leak information. And switches can be attacked by flooding them with ARP packets, overflowing internal tables with new IP address/MAC address mappings until the switch goes back to acting like a broadcast hub. So switches did not turn out to be the panacea they were advertised to be. David Brumley's article in the November *;login:* describes some of the information collected from a sniffer at Stanford University.

Wireless

Wireless represents the newest networking mania. If you have been to a USENIX conference in the last several years, you will have noticed many people with laptops using the wireless network to communicate. USENIX provides a number of base stations (access points), all connected to a notebook acting as a router through to the Internet. This is very convenient, and I certainly appreciate it. Wireless has also appeared in businesses and homes everywhere. Not having to wire up your house or office has great appeal, and the newer versions of 802.11b support higher speeds and 128-bit encryption.

Too bad the encryption is pretty worthless.

Peter Shipley grabbed a lot of attention during this year's RSA Conference in San Francisco with a demonstration of war driving. Peter attached a microwave antenna to a wireless card in his laptop, added a GPS receiver, and drove reporters around downtown San Francisco identifying wireless networks.

You can do this yourself. Mark Langston did, but without the GPS and using only a Libretto sitting on his dashboard. You can read his account of war driving around the Silicon Valley area at *http://www.bitshift.org/wardriving.shtml*. Mark suggests that an even nicer way of doing this (well, more sinister actually) would be to use a Compaq iPaq running Linux, with a wireless card and external power supply, and simply leave it at a site where you would like to monitor the network. Even if the site is using encryption, the way the IEEE standards committee implemented the encryption leaves it open to many attacks.

WEP also includes a 32-bit CRC at the end of each encrypted data segment to provide a check on data integrity. This doesn't work either.

Really? Yep. There have been a raft of papers about cracking WEP (Wired Equivalent Privacy), the encryption that was supposed to make wireless equivalent to a wired network. The committee produced an excellent example of what happens when smart people, using proven crypto, design a system that fails – because they did not aggressively get the crypto community to check out their design.

Broadcasts

First off, war driving works because access points (and even laptop cards) broadcast management frames. Management frames are never encrypted, and include the Service Station ID (SSID). A lot of sites put their organization name there, making it really easy to identify an interesting network. Some access point hardware includes extensions for access control, but these are generally trivial to bypass. Lucent, for example, used the SSID as the password, making things really simple for Macintosh users, where the wireless software actually presents a list of SSIDs and asks which one to join.

802.11 included only 64-bit encryption using RC4, which sounds like it should work okay, even if the key space is a bit small. Many vendors have implemented what is known as 802.1x, with 128-bit passwords, and schemes for dynamically updating keys. And none of this works very well because of the design flaws in the standard, which everyone implements for interoperability.

First, a quick reminder about RC4. RC4 is a streaming cipher invented by Ron Rivest, licensed by RSA, and publicly available for about seven years. RC4 uses the key to generate a pseudo-random keystream, then XORs the data to encrypt with the keystream. To decrypt, all you need to do is to XOR with the same keystream. RC4 is used in SSL/TLS because it is fast and secure when used with a large key space (128 bits, for example).

So, what is wrong with WEP? 802.11 specifies a shared key for encryption. When using RC4, it is important never to use the same key twice since it is possible to perform cryptanalysis on two ciphertexts and to decrypt both without cracking the key. To avoid reuse of the single, shared key, WEP appends a value, the Initialization Vector, or IV, to 40 (or 104 for the 128-bit version) bits of the key. The IV is three bytes long, 24 bits, so there are over 16 million IVs possible.

WEP also includes a 32-bit CRC at the end of each encrypted data segment to provide a check on data integrity. This doesn't work either.

Ian Goldberg, of Berkeley, was one of the authors of a paper (*http://www.isaac.cs.berkeley.edu/isaac/*wep-*faq.html*) that discusses some of the shortcomings of WEP. Ian also spoke at the Blackhat conference in 2001, outlining several different ways of defeating WEP.

WEP networks can be configured to require authentication before a station can join. When a station contacts an access point, the access point sends a 128-byte challenge, and the station responds by choosing an IV, encrypting the challenge using the key created from the shared secret and the IV, and sending back the response. Now, anyone who has sniffed this exchange can XOR the challenge and response and use the resulting keystream (and the same IV) to authenticate to the same network.

The attacker at this point does not know the shared secret, but has authenticated. Since most people rarely change the shared secret, if an attacker can come up with the

shared secret, they can sniff/use this network for a long time. Most vendors support a key generator that takes a string and converts it to a key. Timothy Newsham (of @stake) discovered that the key generator itself is very flawed, throwing away most entropy, so that the resulting keyspace is only 2 to the 21 (about two million keys). An attacker can sniff a couple of packets, then use Newsham's tool to guess the key. During Blackhat demonstrations, Newsham correctly guessed several keys in just fractions of a second.

Well, that's not very good. You can input hexadecimal values for the keys instead (highly recommended), thus placing a brute force attack at a significant disadvantage. While the 40-bit secret can be cracked by 10 systems in a day, forget about brute forcing the 104-bit shared secret.

And there are other attacks. The simplest one described by Goldberg is to send ICMP Echo Requests, padded with the data of the attacker's choice, to workstations on the wireless network and sniff the packets. Now, the attacker has both the plaintext and the ciphertext and can recreate the keystream by XORing the two together. Theoretically, the attacker would have to do this for an entire 24-bit IV space. Practically, rebooting a system, inserting a wireless card, or entering a wireless network initializes the IV to zero, so collecting a keystream for all 16 million IVs won't be necessary.

Goldberg described three other attacks. In the double encryption attack, the attacker sniffs the network, waiting until the IV for a previously sniffed packet is about to be used by the access point. The attacker then sends many copies of the encrypted packet to a workstation on the network, sniffing again. When the IV matches, the keystreams will be the same, and the attacker can sniff the plaintext – the access point having decrypted it and transmitted it.

Because WEP uses RC4 and a communications checksum, it is trivial to modify a message. The attacker can XOR bits into the message, then XOR these same bits into the 32-bit checksum, and successfully modify an encrypted message. An attacker can take advantage of this attack to redirect modified copies of encrypted messages to a system the attacker controls just by changing the destination IP address (and perhaps the port address as well). When the packet passes through the access point, it gets decrypted and sent to the attacker.

Goldberg called the third attack a reaction attack. If the attacker suspects that a target has entered a password, the attacker can send spoofed packets with small modifications to the TCP header checksum to guess bits. When a guess is correct, the attacker will see an ACK or RESET packet. An incorrect guess results in no response. Guessing a password would require at least 56 guesses (one for each bit).

Cracking WEP

As if this was not bad enough, a group of mathematicians (*http://www.eyetap.org/~rguerra/toronto200/rc4_ksaproc.pdf*) postulated an attack that reveals the shared secret. They proved that RC4 reveals information about some bits in the key in the second encrypted byte (and others of the first 256 bytes as well). By capturing four or six million packets (depending on whether 64- or 128-bit encryptions was used), the entire shared secret could be deduced. Two SourceForge projects have software to deduce the keys. And while collecting millions of packets might sound ridiculous, 24GB drives are cheap, and that quantity of packets could be gathered from a busy network in a single day.

Because WEP uses RC4 and a communications checksum, it is trivial to modify a message.

The security implications of 802.11 have thoroughly convinced me that I will treat wireless like any broadcast medium, and only use it with SSH or some other form of VPN.

The IEEE 802.11 committee was made aware of many of the failings of WEP by Jesse Walker of Intel in October 2000. You can find his paper, as well as some of the others I mentioned, through a nice page of links, *http://www.cs.umd.edu/~waa/wireless.html*, which also includes a great paper by University of Maryland researchers that explains 802.11 in great detail. The IEEE will most likely do most of what Walker suggested, which includes increasing the IV length, using a block cipher instead of RC4, and using a cryptographic checksum that includes the keying material to prevent undetectable modifications to the ciphertext.

I had already decided to wire my house with CAT 5 before I learned all this about 802.11. Bill Cheswick had mentioned to me that he wasn't interested in turning his house into a microwave oven. While Bill was exaggerating, the security implications of 802.11 have thoroughly convinced me that I will treat wireless like any broadcast medium, and only use it with SSH or some other form of VPN. Mark Langston suggests putting access points in your DMZ – since the wireless network is effectively as much outside your building as it is inside.

I don't want to sign off without mentioning something that appeared at Netcraft, the Web surveyor's site (*http:// www.netcraft.com/survey*), on Slashdot, and at the Gartner Group's site. You have probably heard of the Gartner Group: analysts who get paid for their views about technology. On September 19, they posted an interesting opinion, which I am including just in case someone convinces them to retract what was posted at: *http://www3.gartner.com/DisplayDocument?doc_cd=101034*.

"Gartner recommends that enterprises hit by both Code Red and Nimda immediately investigate alternatives to IIS, including moving Web applications to Web server software from other vendors, such as iPlanet and Apache. Although these Web servers have required some security patches, they have much better security records than IIS and are not under active attack by the vast number of virus and worm writers. Gartner remains concerned that viruses and worms will continue to attack IIS until Microsoft has released a completely rewritten, thoroughly and publicly tested, new release of IIS. Sufficient operational testing should follow to ensure that the initial wave of security vulnerabilities every software product experiences has been uncovered and fixed. This move should include any Microsoft .NET Web services, which requires the use of IIS. Gartner believes that this rewriting will not occur before year-end 2002 (0.8 probability)."

Well, Code Red actually used the Indexing Server, which runs separately from IIS, so perhaps they should rewrite all of Win2K and XP. I can wait.

Wireless insecurity URLs:

A great page of links, including Jess Walker's, the Berkeley paper, and an explanation of 802.11: *http://www.cs.umd.edu/~waa/wireless.html*

Early release of software for cracking the WEP encryption in 802.11b: *http://sourceforge.net/projects/wepcrack/* (also */airsnort*).

The Fluhrer, Mantin, and Shamir paper explaining how to crack WEP: *http://www.eyetap.org/~rguerra/toronto2001/rc4_ksaproc.pdf*

IEEE Standard 802.11 standards, paper (by order) or PDF: *http://standards.ieee.org/catalog/IEEE802.11.html*

An interview with Peter Shipley about war driving: *http://www.starkrealities.com/shipley.html*