

Windows, and avoid most of the issues involved with running a Web server on Windows NT or 2000. There is even support for running .asp scripts (although that might open you up to some of the same exploits that will appear on IIS servers). Apache is available through [www.apache.org](http://www.apache.org) (with mirrors worldwide).

Rik Farrow can be reached via e-mail: [rik@spirit.com](mailto:rik@spirit.com).



## ESPIONAGE

### Arrested development, Part I by Richard Power

*This is the first in a two-part look at some serious problems that don't seem to be getting any better. Next month, "Arrested Development, Part II: Mitnick preaches to the software developers..."*

On Friday, October 27, the news wires caught fire with reports that Microsoft had been hacked and its source code had been compromised in the attack.

According to Reuters, Microsoft characterized the incident as "a deplorable act of corporate espionage."

Reuters quoted Steve Ballmer, Microsoft's President and Chief Executive, directly: "It is clear that hackers did see some of our source code."

By the end of the day, I had done interviews with the *Los Angeles Times*, the *Washington Post*, the *New York Daily News*, the *San Jose Mercury News*, *USA Today*, *Newsweek*, the *BBC*, the *Associated Press*, *Reuters* and others.

These follow-up stories hit the newsstands over the weekend. They highlighted several key points.

#### Information-age espionage not industrial espionage

Microsoft itself, referred to the incident as an act of "espionage." Indeed, for several years now, it has been obvious to some of us that industrial age espionage (for example, the turning of an insider through bribes, blackmail or seduction) was rapidly giving way to information age espionage (for example, hacking the internal networks of major corporations to engage in digital trade secret theft).

Whether or not, if and when, the actual motives of the perpetrator(s) are ever known, the Microsoft source code hack underscores the urgent need for those responsible in vital sectors of the economy to take information age espionage far more seriously than they have so far.

#### Microsoft not alone in anti-malware failure

The initial news reports indicated that one of the hacking tools used was a Trojan horse/worm, QAZWSX.HSQ (derived from the keys farthest to the left of the QWERTYY-style keyboard). It is commonly referred to as QAZ.

QAZ was a known piece of malicious code. It was discovered in China in mid-July. All of the major anti-virus software vendors, including those used by Microsoft, had

updated their scanners' signature databases to identify QAZ.

Of course, the annual CSI/FBI survey illustrates the problem. Each year, over 90% of the respondents indicate that they use anti-virus software; but over 90% also indicate that they have suffered virus contaminations within the last twelve months. My guess is that although some of these virus incidents are the result of new strains, many of the incidents are the result of a break-down in anti-virus defenses. Typically, such breakdowns are either the result of a lack of commitment to updating the signature databases for scanners enterprise-wide, or the failure to update (or in some cases even use) the scanners for telecommuters and other remote access users.

#### Is spin control a countermeasure?

It is hard to get over the fact that it was Microsoft's source code that was compromised in this incident. It is also hard to get over the fact that remote access to Microsoft's source code was "secured" with reusable passwords. After all, any organization can be hacked, any organization can suffer from a breakdown in their information security—but Microsoft's customers, partners and stockholders might have expected a higher level of security surrounding the source code. And those customers, partners and stockholders with some knowledge of information security would certainly have expected something more than passwords to allow remote access to that source code.

If I am wrong, I welcome Microsoft's rebuttal and clarification. Unfortunately, all I hear coming out of Redmond through the media is spin and misdirection reminiscent of the recent presidential election campaign. For example, both on Friday and then again on Monday, Microsoft emphatically denied that any harm was done to its source code. Well, frankly, such blanket statements rely on the naivete of the audience.

All Microsoft could really say with any degree of certainty is that the copy of the source code resting on that particular server had not been altered. Okay. But was it copied in some way? Was it downloaded from that server? Was it studied for a long time even if it wasn't downloaded? If so, the source code is at risk in numerous ways.

If the source code fell into the hands of the underground it would be analyzed for vulnerabilities that could be exploited in future attacks.

The source code could be analyzed by competitors to more easily mimic Microsoft product features and/or improve on such features. Such analysis could save these competitors R&D time and money as well as deal Microsoft a blow in the marketplace.

The downloaded source code could also be Trojanned and then compiled, shrink wrapped and sold to the unwitting.

Another far more troubling example of how Microsoft used spin and misdirection was how the published story changed from Friday to Monday. If nothing else, there was at least a P.R. failure at Microsoft.

On Friday, the media carried the comments of Steve Ballmer, Microsoft President and CEO, and others making statements that would be difficult to misinterpret. But on

Monday, several of these statements were contradicted.

For example, on Friday (and over the weekend) the press reports stated that the intrusions had gone on anywhere from five weeks to three months. But on Monday, Microsoft said that the intrusions had only gone on for twelve days and that they had known about it all along and had been monitoring the intrusions the entire time.

There was an uncanny sense that Monday's version of the incident had more to do with spin control based on the comments of information security experts quoted in the media over the weekend than it had to do with some natural progression of the story from Friday's initial reports.

Of course, as a result of Monday's "clarifications," the story went away. It was deep-sixed. My phone went dead.

### Passwords: the Undead strike again

No, I won't digress into the ongoing debate about the relative merits of going open source or holding your source code as a proprietary secret. I just want to point out that Microsoft for better or worse has chosen to hold its source code as a proprietary secret. Indeed, it has spent a fortune in legal fees protecting its source code from the probing eyes of the U.S. government. And yet, it could not thwart the probes of some hacker using an e-mail drop in Russia.

One of the perennial pleasures of CSI's annual conference and exhibition (held this year in Chicago) is talking with William Hugh Murray of Deloitte and Touche LLP, one of the giants in the field. This year, I reminded him that way back in 1995, I had transcribed a presentation he had given on the problems surrounding the authentication of users and ran the piece on the cover of the *Computer Security Alert*.

I wanted to title the excerpt: "Death of the password."

Indeed, in the course of his presentation, Murray did pronounce the reusable password dead. He gave it a proper eulogy ('it served us well') and consigned it to oblivion ('but its usefulness is at an end').

Murray is an eminently practical man. He knew back then that passwords wouldn't be going away. He just wanted people to know that they should have gone away.

That talk of Murray's had a big impact on me. Ever since I have been telling people that passwords were just a "placebo," just something to make people "feel better."

Unfortunately, the harsh reality is that most information security practitioners still toil in environments in which their struggles involve questions like, "How long a period of time should users be allowed before they have to change their passwords? Thirty days? Sixty days? Ninety days?" Incredible.

So after reminding Murray about "Death of the Password," I brought up the subject of the Microsoft hack.

"Isn't the whole point of that story—wherever it leads from here—that Microsoft was using reusable passwords to control access to its source code?"

Murray smiled, sipped his champagne and said, "yes."

### What to do

For another dose of reality, I dropped into Rik Farrow's post-conference two-day class on Intrusion Techniques and Countermeasures (an excellent course he teaches exclu-

sively for CSI and originally developed at the request of the National Security Agency). Of course, Farrow discusses a broad range of threats and countermeasures, but just as a teaser and a practical assist here are some random excerpts from his course notes on password attacks and countermeasures.

"The easiest way to get into any system is the normal way: entering a username and password.

"Passwords may be acquired via non-technical means: social engineering, and dumpster diving.

"Compromised accounts may also provide usernames and passwords: an administrator's or user's list of accounts and passwords, files such as .netrc, found on UNIX system, password caches found on Win 9x, or via the LSAsecrets exploit for NT systems.

"In years past, it was common to set up a new system without passwords: account names such as 'guest' and 'tutor' were common. Systems still appear without passwords—for example, NT systems with MSQL server (accountname sa, admin privileges), Silicon Graphics IRX (accountname lp), UNIX systems in general (adm, bin, uucp, nuucp).

"Password crackers use common passwords, dictionaries and speeded-up hashing algorithms. The success of any crack program is based on having extensive and relevant dictionaries, making good guesses based on information about the account and modifying both of the above according to a set of rules (e.g., 1->1,E->3).

"A copy of the hashed password is essential for cracking. Many UNIX exploits exist for capturing UNIX passwords.

"NT servers and workstations store their passwords in the System Accounts Manager (SAM).

"With SP3, Microsoft added SYSKEY, the ability to encrypt stored password hashes with a 128 bit key. SP6 includes a better version of SYSKEY. If you choose to do this, you have three choices for providing this key: keep it stored on a floppy, which must be used during booting; by entering a passphrase during the reboot process; or, let NT hide the key in the registry.

"NT passwords are used as part of network authentication. Older Windows versions (and Samba) may transmit plaintext passwords. More commonly, a challenge-response method for authentication is used. If both the challenge and the response can be sniffed from the network, they can be used in password cracking. Internet Explorer can be tricked into performing challenge-response with a rogue Web server. PPT, Microsoft's remote encryption algorithm, includes the challenge-response (and uses the password as the base for the encryption key). There is a tool called pptpsniff.

"The existence of the backward compatible Lanman passwords make brute force cracks much easier.

"The best possible defense against password sniffing or cracking is to use one-time passwords: e.g., passwords are not stored or re-used, authentication devices can provide much greater security. Second-best are proactive password checking solutions: e.g., passwords are checked when users attempt to change them. Hiding passwords (shadow password files and SYSKEY) are better than nothing.

How much of your security relies on reusable passwords?

# ALERT

THE NEWSLETTER FOR INFORMATION PROTECTION PROFESSIONALS

COMPUTER  
SECURITY  
INSTITUTE  
600 HARRISON STREET  
SAN FRANCISCO  
CALIFORNIA 94107  
TEL: (415) 947-6320

## INSIDE INFORMATION



### Crime

Hacker defense  
attorney speaks out

2



### Tools & Techniques

Infosec experts take  
hard look at Carnivore

5



### Government

Bush v. Gore fallout  
highlights infosec  
challenges

6



### Industry Update

Dow Chemical's  
Dan Ervin honored  
by CSI

7



### Calendar

New offerings  
for 2001

7



### Bonus Item

Move systems into  
production consistent  
with an architecture

9

## Arrested Development, Part II

By Richard Power

At the Software Development East conference in Washington, D.C. in October 2000, Kevin Mitnick (aka Condor), fresh from 46 months in Federal prison, was interviewed via satellite (he was not allowed to leave California, where he is on parole). Mitnick, "looking more like an investment banker than a hacker" as one reporter phrased it, espoused a purely vanilla flavor of computer security during the hour-long session. Several hundred software developers crammed the auditorium to hear Mitnick. The Mitnick interview was followed a panel of experts—Dr. Dorothy Denning of Georgetown University, Keith Alan Rhodes of the U.S. General Accounting Office and Ron Moritz of Symantec—none of whom have spent any time on the FBI "Wanted" list. Of course, as I had predicted to my colleagues (I chaired the panel), the crowd of developers shrank from several hundred to a few dozen. Further evidence that software developers, in general, just don't get it. Here are excerpts from the post-Mitnick panel.

Keith Rhodes, U.S. GAO: "When I was asked to speak here and I found out that Kevin Mitnick was going to be the keynote speaker, I thought, 'Well, okay, I've known Richard for a long time so I'll do it anyway.' I want to make certain that people understand how individuals do break-ins. Mitnick is a very good example because he's not the Einstein of intruders, he was just very good at dealing with people.

"Even though as engineers and scientists we may like to think that we aren't really dealing with people—we are dealing with people. For example, at the Center for Technology and Engineering of the U.S. General Accounting Office, we often break in simply because people make up bad passwords. People do that.

"One of the great ways to break into an organization is to photograph all the license plates of the cars and then put that in your password cracking dictionary. But that's still more sophisticated than trying to break the passwords against Webster's."

"It's the human side (e.g., social engineering, elicitation) that we often exploit.

"There's nothing more helpful than a "help desk" because the help desk people are at the low end of the food chain in terms of the organization. They're hated, they know they're hated, they're treated like crap, they're given bad tools, and you don't really have to do much to get around them. All you have to do is call the help desk and say, "Oh I'm sorry, I'm new here..." and they're more than willing to help you. Soon you get three user ID passwords changed and suddenly you have root. So yeah, it may not seem like it to most software developers but you're actually building code for people. Somebody is going to end up owning that code. I think it's very valuable, as much as I dislike having an rock star intruder be the keynote speaker at something that's supposed to be serious, it's very good to understand that what he did was take advantage of the weaknesses of people. We find those weaknesses exploited again and again.

"On the Internet, we're using some techniques that are about ten or twelve thousand years old. They still work because people still have to do system administration, network administration and security administration. People do it.

"Time after time, when a U.S. GAO team goes into an organization, we find that security isn't taken seriously—and it's certainly not taken very seriously amongst most

continued on page 8

*continued from page 1*

organizations that are developing software.

"If there's one message that I want you to take away from what you heard from Kevin Mitnick, it's that he didn't have to bust your system wide open because he usually had maybe one or two user ID's and passwords, and he used some very standard exploitations. What I would like you to work into your equation is to realize that the systems you are designing are being built for people and people can be manipulated.

*Ron Moritz, Symantec:* "How many of you are actually developing code for the Web today? Can I have a show of hands? I'm looking at about half of the crowd (25 out of 50 attendees). Two years ago I got interviewed by one of the San Francisco papers after some attack against some Web site, and the questions posed to me were "Why isn't the Web secure?" and "Will it ever become secure?" My answer was (I think it's still appropriate today) is that we, as software developers, have forgotten thirty years of data processing. How many of you came from the mainframe developing community? Very few. How many of those who have developed software for the mainframe world are now developing for the Web? Almost none. The principles we learned in thirty years of mainframe data processing were simple—for example, preventive controls, detective controls, response, deterrence. These concepts are common ones in the accounting world. They flow from many of the systems that we developed—for example, general ledger systems, accounts payable systems.

"We understood what it meant when data had to be validated, when input such as date fields had to be validated. You couldn't put a date with a month of "13." Many of these principles have been thrown aside in the rush to get things to the Web.

"I come from a software development background, and I run a software development company. I'm sorry I've got to do this, but I've got to point the finger back at the software development community. When we look at why we have problems on the Web today it's often because the developers are creating technologies that have defects.

"We've rushed to get our software out and we've forgotten some of the principles that allow our software to be of high quality. We can't do that any longer in this connected world. We've got to provide confidence for users in this connected world, and this isn't going to happen if the underlying security principles is forgotten.

"I was invited to a S.F. Giants baseball game a few months back in PacBell Park in San Francisco. (The Giants won that night.) I was invited because Symantec and Yahoo! have a relationship. Thirty-eight million Yahoo! users have their email attachments scanned in both directions by an array of Symantec servers. My guys invited me to attend that baseball game because Yahoo!'s Chief Technical Officer (CTO) was going to be there and we'd have a lot of things to talk about—two CTO's getting together. We spent nine evenings trying to find something to talk about. We could not find a common ground. This is a very telling experience. There you had one of the people who is on the frontier of the Web, pushing that frontier ever farther, not wanting to be restrained by at least some of the security

principles that I've grown up with.

"He was pushing. I was pulling back, asking him to consider some of the principles of security. This is what's happening in the industry today. This was very indicative of the situation.

"If you look at the Computer Emergency Response Team (CERT) advisories over the last five years, you will see that the majority of them have been related to buffer overflows. Who's responsible for buffer overflows? You, the software developers, are responsible.

"That doesn't make sense to me. This is a real challenge for us in the industry. Why aren't we building quality software? A lot of it has to do with our business partners. We're not developing software as engineers just for the sake of developing software. We're developing software because we have some business needs and the market pressures often cause us to rush faster than we'd like to as software engineers. How many of you actually paid attention in your software engineering classes to the section on reliability, black box testing, white box testing? Two or three. I slept through that one. That was not my favorite topic—but it is one of the keys, one of the foundations of our ability to create reliable and secure and safe software today. We should not be allowed to develop software for the Web without having those principles as part of our core, our foundation, right alongside an understanding how to write C code or Java code or any of the script languages.

*Dorothy Denning, Georgetown University:* "A lot of the software that's out there gives the impression that it was thrown together in a rush job in order to get it out to the market. When it gets thrown out there what happens is that tens of thousands of hackers all over the world go at it and they find the weaknesses. Some of them are even pretty good about the process of exposing these weaknesses. They will contact the vendor first and try to work with the vendor to get it fixed before going public with it, but then eventually going public with it for various reasons. One reason is that they probably want to get some credit for it amongst their peers. But also I think many of them have a legitimate feeling that by publishing the vulnerabilities and making them public, they will in the end alert people to the problems so that they will better protect their systems.

"This process goes on and on. The number of vulnerabilities is growing all the time. Every day, at least a couple of notices on new vulnerabilities show up in my e-mail boxes. To some extent this model works—systems are getting better. People are fixing the vulnerabilities. One of the big breakdowns with the model is that even if the vulnerabilities get fixed and patches are developed, many end users simply don't install them. So they end up running systems that still have vulnerabilities because they're not using the latest versions of everything.

"What we really need to do here, and I call upon you, I'm not a software developer, is to somehow take that process of vulnerability testing and put that into the life cycle of the software's development—before it's release. Hopefully, if you do this when the initial versions of these products come out that they will in better shape".