Bruce Schneier

Crypto-Gram Newsletter

February 15, 2000

by Bruce Schneier
Founder and CTO
Counterpane Internet Security, Inc.
schneier@schneier.com
http://www.counterpane.com

A free monthly newsletter providing summaries, analyses, insights, and commentaries on computer security and cryptography.

Back issues are available at http://www.schneier.com. To subscribe or unsubscribe, see below.

In this issue:

Distributed Denial-of-Service Attacks

Suddenly, distributed denial-of-service (DDS) attacks are big news. The first automatic tools for these attacks were released last year, and CERT sent out an advisory in November. But the spate of high-profile attacks in mid-February has put them on the front pages of newspapers everywhere.

Not much is new. Denial-of-service attacks have been going on for years. The recent attacks are the same, only this time there is no single source of the attack. We've seen these for years, too. The attacker first breaks into hundreds or thousands of random insecure computers (called "zombies") on the Internet and installs an attack program. Then he coordinates them all to attack the target at the same time. The target is attacked from many places at once; his traditional defenses just don't work, and he falls over dead.

It's very much like the pizza delivery attack: Alice doesn't like Bob, so she calls a hundred pizza delivery parlors and, from each one, has a pizza delivered to Bob's house at 11:00 PM. At 11, Bob's front porch is filled with 100 pizza deliverers, all demanding their money. It looks to Bob like the pizza Mafia is out to get him, but the pizza parlors are victims too. The real attacker is nowhere to be seen.

This sounds like a complicated attack on the Internet, and it is. But unfortunately, it only takes one talented programmer with a poor sense of ethics to automate and distribute the attacks. Once a DDS tool is publicly available, an attacker doesn't need skill; he can use a simple point-and-click interface to infect the intermediate sites, as well as to coordinate and launch the attack. This is what's new: easy-to-use DDS tools

like Trin00 and Tribal Flood Network.

These attacks are incredibly difficult, if not impossible, to defend against. In a traditional denial-of-service attack, the victim computer might be able to figure out where the attack is coming from and shut down those connections. But in a distributed attack, there is no single source. The computer should shut down all connections except for the ones it knows to be trusted, but that doesn't work for a public Internet site.

Other defenses also have problems. I've seen proposals that force the client to perform an expensive calculation to make a connection. (RSA pre-announced such a "solution.") This works against standard denial-of-service attacks, but not against a distributed one. Large-scale filtering at the ISPs can help, but that requires a lot of effort and will reduce network bandwidth noticeably.

At least one report has suggested that a lack of authentication on the Internet is to blame. This makes no sense. The packets did harm just by the attempt to deliver them; whether or not they were authenticatable is completely irrelevant. Mandatory authentication would do nothing to prevent these attacks, or to track down the attackers.

There have been two academic conferences on DDS attacks in recent weeks, and the general consensus is that there is no way to defend against these attacks. Sometimes the particular bugs exploited in the DDS attacks can be patched, but there are many that cannot. The Internet was not designed to withstand DDS attacks.

Tracing the attacker is also incredibly difficult. Going back to the pizza delivery example, the only thing the victim could do is to ask the pizza parlors to help him catch the attacker. If all the parlors coordinated their phone logs, maybe they could figure out who ordered all the pizzas in the first place. Something similar is possible on the Internet, but it is unlikely that the intermediate sites kept good logs. Additionally, it is easy to disguise your location on the Internet. And if the attacker is in some Eastern European country with minimal computer crime laws, a bribable police, and no extradition treaties, there's nothing you can do anyway.

So far, these attacks are strictly denial-of-service. They do not affect the data on the Web sites. These attacks cannot steal credit card numbers or proprietary information. They cannot transfer money out of your bank account to trade stocks in your name. Attackers cannot gain financially from these attacks. Still, they are very serious. And it is certainly possible that an attacker can use denial of service as a tool for a more complicated attack that IS designed to steal something.

This is not to say that denial-of-service attacks are not real, or not important. For most big corporations, the biggest risk of a security breach is loss of income or loss of reputation, either of which is achieved by a conspicuous denial-of-service attack. And for companies with more mission- or life-critical data online, a DOS attack can literally put a person's life at risk.

The real problem is that there are hundreds of thousands, possibly millions, of innocent naive computer users who are vulnerable to attack. They're using DSL or cable modems, they're always on the Internet with static IP addresses, and they can be taken over and used as launching pads for these (and other) attacks. The media is focusing on the mega e-corporations that are under attack, but the real story is the individual systems.

Similarly, the real solutions are of the "civic hygiene" variety. Just as malaria was defeated in Washington, DC, by draining all the swamps, the only real way to prevent these attacks is to protect those millions of individual computers on the Internet. Unfortunately, we are building swampland at an incredible rate, and securing everything is impracticable. Even if personal firewalls had a 95% market penetration, and even if they were all installed and operated perfectly, there would still be enough insecure computers on the Internet to use for these attacks.

I believe that any long-term solution will involve redesigning the entire Internet. Back in the 1960s, some people figured out that you could whistle, click, belch, or whatever into a telephone and make the system do things. This was the era of phone phreaking: black boxes, blue boxes, Captain Crunch whistles. The phone company did their best to defend against these attacks, but the basic problem was that the phone system was built with "in-band signaling": the control signal and the data signal traveled along the same wires. In the 1980s, the phone company completely redesigned the phone system. For example SS7, or Signaling System 7, was out-of-band. The voice path and data path were separated. Now it doesn't matter how hard you whistle into the phone system: the switch isn't listening. The attacks simply don't work. (Red boxes still work, against payphones, by mimicking the in-band tones that count the coins deposited in the phones.)

In the long term, out-of-band signaling is the only way to deal with many of the vulnerabilities of the Internet, DDS attacks among them. Unfortunately, there are no plans to redesign the Internet in this way, and any such undertaking might be just too complicated to even consider.

Discussion of DDS attacks:
<http://staff.washington.edu/dittrich/talks/cert/>

CERT Advisory:
<http://www.cert.org/incident_notes/IN-99-07.html>

Tool to check if Tribal Flood Network or Trin00 is installed on your computer:
<http://www.nfr.net/updates/> [link dead; NFR updates are now at https://support.nfr.com/]

Tutorial on DOS attacks:
<http://www.hackernews.com/bufferoverflow/00/...>

Trin00 Analysis:
<http://staff.washington.edu/dittrich/misc/...>

Tribal Flood Network Analysis:
<http://staff.washington.edu/dittrich/misc/tfn.analysis>

Stacheldraht Analysis:
<http://staff.washington.edu/dittrich/misc/...>

Declan McCullagh's essay on the topic:
<http://www.wired.com/news/politics/0,1283,34294,00.html>

New Chinese Cryptography Regulations

Claiming that they are trying to prevent state secrets from being stolen, China has issued some strict Internet cryptography controls. First, everyone who uses encryption has to register with the government and give the details of what software they are using and the algorithm, users' names, e-mail addresses, as well as the location of their computers. Second, Chinese companies are prohibited from buying products containing encryption software that was designed overseas. (So if a U.S. software company wants to sell encryption in its product, it needs to rip out whatever it has now and install something Chinese-made.)

This is a weird one, and I have a few observations. One, China is probably afraid that foreign security products have back doors. This is possible, and something that the U.S. has done before. But I don't see how enforcing requirements on crypto algorithms help here. Back doors are usually much more subtle than a broken crypto algorithm.

Two, this could easily be a prelude to key escrow. Certainly the first step toward requiring people to give a copy of their encryption keys to the government is to find out who is using encryption, and what kind.

Three, even by itself this information is useful for Chinese espionage. Traffic analysis is a very difficult problem, and knowing who is using encryption software (and where they are physically located) makes it a lot easier to know who to target.

People with a lot more political expertise than I have said that this is really nothing. China demanded that all fax machines be registered a decade ago, and many didn't bother to comply.

<http://www.wired.com/news/print/0,1294,33910,00.html>
<http://www.usatoday.com/life/cyber/tech/cth217.htm>
<http://www.currents.net/newstoday/00/01/27/news6.html>

Counterpane Internet Security News

Lots of excitement; still lots of secrecy. We're up to 45 employees and still growing. If anyone knows of a good VP of Marketing who likes commuting to San Jose, send him my way.

The Counterpane Web site has a new look. Visit it:
<http://www.counterpane.com>

Bruce Schneier is speaking at PC Forum, on March 15th:
<http://www.edventure.com/PC2000/>

Publicizing Vulnerabilities

Last month I discussed publicity attacks, and the tendency of vendors to hype security vulnerabilities as publicity for their products and services. My essay generated a lot of commentary (see the end of the article for some URLs). This is a complicated issue, with gray areas, slippery slopes, and a lot of room for debate. My position has changed over time. I'd like to revisit it.

There are really two issues here, intertwined. If someone discovers a vulnerability in a product, should he quietly alert the vendor or should he make it public? And when is a vulnerability important and when is it trivial?

The first issue involves some history.

In 1988, the Morris Worm illustrated how susceptible the Internet is to attack. The Defense Advanced Research Projects Agency (DARPA) funded a group to coordinate responses to these kinds of attacks, increase security awareness, and generally do good things for Internet security. The group is known as CERT -- more formally, the Computer Emergency Response Team -- and its response center is at Carnegie Mellon University in Pittsburgh.

Over the years CERT has acted as kind of a clearinghouse for security vulnerabilities. People are supposed to send vulnerabilities they discover to CERT. CERT then verifies that the vulnerability is real, quietly alerts the vendor, and publishes the details (and the fix) once the vendor has fixed the vulnerability.

This sounds good in theory, but worked less well in practice. There were three main complaints. First, CERT got a lot of vulnerabilities reported to it, and there were complaints about CERT being slow in verifying them. Second, the vendors were slow about fixing the vulnerabilities once CERT told them. Since CERT wouldn't publish until there was a fix, so there was no real urgency to fix anything. And third, CERT was slow about publishing reports even after the fixes were implemented.

The "full disclosure" movement was born out of frustration with this process. Internet mailing lists like Bugtraq (begun in 1993) and NT Bugtraq (begun in 1997) became forums for people who believe that the only way to improve security is to understand and publicize the problems. This was a backlash against the ivory tower attitude of CERT. As one hacker wrote: "No more would the details of security problems be limited to closed mailing lists of so-called security experts or detailed in long, overwrought papers from academia. Instead, the information would be made available to the masses to do with as they saw fit."

Today, many researchers publish vulnerabilities they discover on these mailing lists, sometimes accompanied by press releases and the usual flurry of factoids. The press trolls these mailing lists, and writes about the vulnerabilities in the computer magazines: both paper-based and online. (This is why there have been so many more press stories about computer vulnerabilities over the past few years.) The vendors scramble to patch these vulnerabilities as soon as they are publicized, so they can write their own press releases about how quickly and thoroughly they fixed things. The full disclosure movement is improving Internet security.

At the same time, hackers use these mailing lists to learn about vulnerabilities and write attack programs (called "exploits"). Sometimes the researchers themselves write demonstration exploits. Sometimes others do. Some of these attacks are complicated; but as soon as someone writes a point-and-click exploit, anyone can exploit the vulnerability.

Those against the full-disclosure movement argue that publishing vulnerability details does more harm than good by arming the criminal hackers with tools they can use to break into systems. Security is much better served, they counter, by not publishing vulnerabilities in all their gory details.

Full-disclosure proponents counter that this assumes that the researcher who publicizes the vulnerability is always the first one to discover it, which simply isn't true. Sometimes, vulnerabilities have been known by attackers (sometimes passed about quietly in the hacker underground) for months or years before the vendor ever found out. The sooner a vulnerability is publicized and fixed, the better it is for everyone.

That's the debate in a nutshell: Is the benefit of publicizing an attack worth the increased threat of the enemy learning about it? (In the language of the intelligence community, this is known as the "equities issue.") If

vulnerabilities are not published, then the vendors are slow (or don't bother) to fix them. But if the vulnerabilities are published, then hackers write exploits to take advantage of them.

In general, I am in favor of the full-disclosure movement, and think it has done a lot more to increase security than it has to decrease it. Publicizing a vulnerability doesn't cause it to come into existence; it existed even before it was publicized. Given that most vendors don't bother fixing vulnerabilities that are not published, publicizing is the first step towards closing that vulnerability. Punishing the publicizer feels a lot like shooting the messenger; the real blame belongs to the vendor that released software with the vulnerability in the first place.

The second issue -- the one that generated most of the discussion last month -- involves the agenda of the researcher. Publishing a security vulnerability is a play for publicity; the researcher is looking to get his own name in the newspaper by successfully bagging his prey. The publicizer often has his own agenda: he's a security consultant, or an employee of a company that offers security products or services. This is especially true if the vulnerability is publicized in a press release. Services like PR Newswire and Business Wire are expensive, and no one would do it unless they thought they were getting something in return.

All researchers are guilty of courting publicity. I am guilty of this. It was fun when my Microsoft PPTP break hit the press. Calling the protocol "kindergarten cryptography" was a hoot. On the other hand, I objected to nCipher's publication of their key finding attack last month. The differences are subtle and there's a lot of gray area, but here are my rules.

First, I am opposed to attacks that primarily sow fear. Publishing vulnerabilities that there's no real evidence for is bad. Publishing vulnerabilities that are more smoke than fire is bad. Publishing vulnerabilities in critical systems that cannot be easily fixed and whose exploitation will cause serious harm (e.g., the air traffic control system) is bad.

Second, I believe in giving the vendor advance notice. CERT took this to an extreme, sometimes giving the vendor years to fix the problem. I'd like to see the researcher tell the vendor that he will publish the vulnerability in a month, or three weeks (no fair giving the vendor just seven days to fix the problem). Hopefully the vulnerability announcement can occur at the same time as the patch announcement. This benefits everybody. (Admittedly, I did not do this with Microsoft PPTP.)

Third, I believe that it is irresponsible, and possibly criminal, to distribute exploits. Reverse-engineering security systems, discovering vulnerabilities, and writing research papers about them benefits research; it makes us smarter at designing secure systems. Distributing exploits just make us more vulnerable. For example, Mixter is a German hacker who wrote the Tribal Flood Network tool used in some of the distributed denial-of-service attacks. I believe he has a lot to answer for. His attack tool served no good. It enabled criminals and cost a lot of companies a lot of money. Its existence makes networks less secure.

This is not clear-cut: there are tools that do both good and bad. Vulnerability assessment tools can be used both to increase security, and to break into systems. Remote administration tools look a lot like Back Orifice. Publishing tools can also help; Microsoft has lied to the press and denied that a published vulnerability is real, until an actual exploit appeared.

I like Marcus Ranum's "be part of the solution, not part of the problem" metric. Full disclosure is part of the solution. Convincing vendors to fix problems is part of the solution. Sowing fear is part of the problem. Handing computer weaponry to clueless teenagers is part of the problem.

These are my opinions; they have changed over time, and are probably still changing. I came to this field via cryptography. Cryptography is by nature adversarial, even in the ivory towers of academia. Someone proposes a new scheme: an algorithm, a protocol, a technique. Someone else breaks it. A third person repairs it. And so on. It's all part of the fun, and this is how I learned. I first came to network security with this philosophy. But when it comes to fielded systems, things can get a lot trickier. Publishing vulnerabilities can cause significant damage to networks, and considerable pain and suffering for network administrators. If an announcement, product, or exploit makes things worse, it's bad. If it makes things better, it's good.

My original essay:
<http://www.schneier.com/...>

One response:
<http://www.securityfocus.com/templates/...>

A response to that response:
<http://www.securityfocus.com/templates/...>

The discussion on SlashDot:
<http://slashdot.org/articles/00/01/17/0839211.shtml>

Marcus Ranum's essay on the topic:
<http://www.clark.net/pub/mjr/pubs/dark/>

See also the reader comments at the end of this issue.

Counterpane -- Featured Research

"A Twofish Retreat: Related-Key Attacks Against Reduced-Round Twofish"

Niels Ferguson, John Kelsey, Bruce Schneier, Doug Whiting

The Twofish AES submission document contains a partial chosen-key and a related-key attack against ten rounds of Twofish without whitening, using 256-bit keys. This attack does not work; it makes use of a postulated class of weak key pairs which has the S-box keys and eight successive round keys equal, but no such pairs exist. In this report we analyze the occurrence of this kind of weak key pair and describe how such pairs may be used both to mount attacks on reduced-round Twofish and to find properties of reduced-round Twofish that are not present in an ideal cipher. We find that related-key and chosen-key attacks are considerably less powerful against Twofish than was previously believed.

<http://www.schneier.com/paper-twofish-related.html>

News

One of the nicer things about living in Minneapolis:
<http://www.ag.state.mn.us/home/files/news/...>

GCHQ (the British NSA equivalent) is looking for recruits. Take the test on their Web site.
<http://www.gchq.gov.uk/>

Various pundits have said that the government is still winning the crypto battle as long as Windows is shipping without strong crypto. Guess what?
Microsoft has said that it would release Windows 2000 worldwide with strong cryptography.
<http://www.wired.com/news/technology/...>

A brute-force machine for a combination lock:
<http://vv.carleton.ca/~neil/robotics/locraker.html>

Would you hire hackers? Some backlash to the @stake announcement:
<http://www.zdnet.com/enterprise/stories/security/...> [link dead; try http://www.zdnet.com/eweek/stories/general/...]

Why security policies fail:
<http://www.cdc.com/working/security/whitepapers/...> [link dead; try http://www.cdc.com/acrobat/208770.pdf]

Another distributed attack against a 56-bit cipher, one called the CS-Cipher. This one took 62 days on about 38,000 machines, and happened to require searching 98% of the keyspace.
<http://www.wired.com/news/print/0,1294,33695,00.html>

Nice article on how easy it is to hack into Web sites:
<http://www.pcworld.com/current_issue/article/...>

The NSA has contracted with Secure Computing Corp. for a secure version of Linux. Personally, I don't know if the Linux license allows the NSA to make a secure version of the operating system if they are not going to freely distribute the results.
<http://www.fcw.com/fcw/articles/...>

The U.S. government and cyber crime:
<http://www.currents.net/newstoday/00/01/17/news4.html>
<http://www.fcw.com/fcw/articles/web-fbi-01-14-00.asp>
<http://www.computerworld.com/home/print.nsf/all/...>
<http://washingtonpost.com/wp-srv/business/feed/...>

The new export rules and the Bernstein case:
<http://www.wired.com/news/print/0,1294,33651,00.html>

In a nice piece of irony, the new U.S. crypto regulations will benefit federal agencies:
<http://www.fcw.com/fcw/articles/web-export-01-14-00.asp> Other commentary on the new encryption regulations:
<http://www.computerworld.com/home/print.nsf/all/...>
<http://www.usatoday.com/life/cyber/tech/cth136.htm>

New service monitors what radio station you're listening to:
<http://www.wired.com/news/technology/...>

Windows 2000 has its first virus:
<http://www.computerworld.com/home/print.nsf/all/...> and its first security holes:
<http://dailynews.yahoo.com/h/zd/20000130/tc/...>
Remember, it hasn't even been released yet.

I'm not the only one who thinks Internet voting is a dumb idea. The "California Internet Voting Task Force" agrees.
<http://www.ss.ca.gov/executive/ivote/>

This is the most clever piece of credit-card fraud I've seen in a long time:
<http://www.zdnet.com/zdnn/stories/news/...>

More information on the French smart card hack:
<http://www.msnbc.com/news/361936.asp>
<http://www.theregister.co.uk/000123-000005.html>
<http://www.parodie.com/english/smartcard.htm>

Someone is suing Yahoo for violating Texas's anti-stalking law by using cookies to track computer users' every move without their consent:
<http://news.cnet.com/category/0-1005-200-1533164.html>

Twofish on the AS/400:
<http://www.news400.com/features/newmf/Article.cfm?...>

Snake-oil alert. Remember, it is possible -- although unlikely -- that this is as good as NEC's PR department makes it sound. But it will take years to know.
<http://www.theregister.co.uk/000127-000025.html>
<http://www.cnn.com/2000/TECH/computing/01/24/...>

Good summaries of the DVD break and deCSS. An important point is that DVDs can be copied and pirated without using deCSS or any other decryption, which certainly makes the original claim of "prevents piracy" look either astoundingly ignorant or brazenly deceptive.
<http://www.fool.com/portfolios/rulemaker/2000/...>
<http://www.latimes.com/news/comment/20000207/...>
<http://linuxtoday.com/stories/16556.html>

Recently declassified NSA documents. 9 and 12 mention ECHELON:
<http://www.gwu.edu/~nsarchiv/NSAEBB/NSAEBB23/...> General information:
<http://www.gwu.edu/~nsarchiv/NSAEBB/NSAEBB23/index.html>

Worth reading. EPIC's testimony on digital infrastructure protection.
<http://www.epic.org/security/cip/...>

House passes digital signature legislation:
<http://www.cnn.com/2000/TECH/computing/01/31/...>

France sues the U.S. and U.K. over ECHELON:
<http://www.the-times.co.uk/news/pages/tim/2000/02/...>

Former CIA director John Deutch brought classified information home on his unsecured laptop.
<http://www.fcw.com/fcw/articles/2000/0131/...>
<http://www.wired.com/news/print/0,1294,34105,00.html>

New vulnerabilities in e-commerce. Some shopping carts allow shoppers to alter fields in HTML forms and URLs to alter prices of items they are buying.
<http://www.computerworld.com/home/print.nsf/all/...>
<http://www.usatoday.com/life/cyber/nb/nb2.htm> [link dead; try http://www.usatoday.com/life/cyber/tech/cte965.htm]
<http://www.theregister.co.uk/000203-000006.html>

Mitnick Case Yields New Crypto Twist

When Kevin Mitnick was captured, federal agents seized two of his laptop computers. Several files on those computers were encrypted. During pre-trial discovery, the prosecution refused to give copies of the encrypted files to the defense unless Mitnick gave them the key. The defense argued that the Constitution required the government to turn over any documents that might help Mitnick defend himself. The prosecution argued that since they had no idea what was in the files, they could be potentially dangerous. Unfortunately, the judge agreed with the prosecution.

<http://www.nytimes.com/library/tech/00/01/cyber/...>

The Doghouse: X.com

A bank where you can withdraw money not just from your account, but from anyone's account. My favorite quote from X.com's CEO: "I don't think a mistake was made." Sadly, I believe him.

<http://www.zdnet.com/zdnn/stories/news/...>
<http://www.currents.net/newstoday/00/01/31/news4.html>
<http://www.nytimes.com/library/tech/00/01/biztech/...>

Cookies

Cookies are a clever programming trick built into WWW browsers. Basically, a cookie is a little bit of data that a Web server gives to a browser. The browser stores the data on the user's computer, and returns it to the server whenever the browser returns to the server. Cookies can do all sorts of useful and good things. Unfortunately, they can also do all sorts of useful bad things. First I'll explain how they work; then I'll talk about the problems.

The WWW is basically a stateless protocol. This means that the server doesn't know who you are from one click to the next. All the server does is serve up Web pages. A browser asks for a Web page; the server gives it to it. The server has no idea if this is the same browser as before or a different browser, nor does it care. This works great for simple, static, Web sites that just contain informational pages.

More complex Web sites are dynamic. Retail Web sites often have shopping carts, which travel with you as you browse the site. Paid access informational sites have usernames and passwords, which travel with you as you go from page to page. (I would find it annoying to have to type my username and password in every time I wanted to see another New York Times article.) Cookies are a way to handle this.

Remember that the WWW protocols are stateless; there is no way for the server to remember who you are from one mouse click to the next. By giving the browser a cookie and then asking for it back, the server can remember who you are. "Oh yes, you're user 12345657; this is your shopping cart." Cookies allow the browser to add state to the WWW protocols. You can think of them as a large distributed database, with pieces stored on millions of browsers throughout user-land.

So far, so good. And mostly, cookies are good, if the server placing the cookie plays by the rules. The server can set how long the cookie lasts before it expires: a few days seems like a good number. A server can set restrictions on who can access the cookie. They usually limit access to servers in the same domain; this

means that if your cookie comes from random-merchant.com, than only random-merchant.com can access the cookie.

The problems come when they are abused. Some servers use cookies to track users from site to site, and some use them to uncover the identity of the user. Here's an easy example: there are companies that resell advertising space on popular sites. DoubleClick is a company that does that; often the ads you see are placed there by DoubleClick in arrangement with the site. If you're browsing on sex-site.com, you're going to see a portion of that window that comes from DoubleClick.com. DoubleClick.com gives you a cookie. Later (that day, or maybe another day), when you're browsing on CDNow.com, there might be another DoubleClick-placed ad. DoubleClick can request the cookie from your browser and, because the cookie says that it was created while you were visiting a sex site, send you targeted ads while you're browsing CDNow. Because DoubleClick is on a bunch of commerce sites, its cookies can be used to track you across all of those sites.

Even worse, if you type your e-mail address in at any of those sites and DoubleClick gets it, DoubleClick can now attach an e-mail address to your browsing habits. All it needs is for you to type that address in once -- that's ordering only one thing -- and it has it forever. (Or, for as long as that cookie has not expired, which can be years.)

DoubleClick freely admits they collect data and use that data to target ads to users, but until recently they denied building an identity database. Two weeks ago, USA Today exposed their duplicity. They have since admitted that they try to collect names and attach cookies to off-line identities. The implications for private Web browsing are profound.

There's more. Sites can send you a cookie in e-mail that it can use to identify you if you later visit that site with your browser. Here's how it works: the site sends you a piece of HTML e-mail. (This implies you're using an e-mail program that supports HTML messages, such as Microsoft's Outlook and Outlook Express, Netscape Messenger, or Eudora.) The message contains a URL to a graphic, which the site can use to send you a cookie. Now, when you browse the site at some later date, the site can use the cookie to link the browsing with the e-mail, and hence the e-mail address. Supposedly this has been used by some sites to track Web surfers.

Some things cookies cannot do: Cookies cannot steal information from your computer. A cookie is simply some data that the server gives the browser, and the browser later returns. A cookie cannot grab your passwords or files. (ActiveX, Java, and JavaScript are much more dangerous in this regard.) Cookies cannot steal your credit card numbers.

The lesson here is that cookies are not bad, but there are some very malevolent uses of them. There are ways in most browsers to turn cookies off completely, and third-party programs that help you manage them better. Managing is a better solution, since some Web sites refuse entry to people who don't accept cookies.

"Opt out" of DoubleClick. They can't keep your personal information if you tell them not to. DO THIS NOW.
<http://www.cdt.org/action/doubleclick.shtml>

Cookie blocking software:
<http://www.junkbusters.com/ht/en/cookies.html>
<http://www.ecst.csuchico.edu/~atman/spam/adblock.shtml>

Anonymous Web browsing:
<http://www.zeroknowledge.com>

Stories on DoubleClick's duplicity:
<http://www.usatoday.com/life/cyber/tech/cth211.htm>
<http://www.hackernews.com/arch.html?012600#1>
<http://news.cnet.com/news/0-1005-200-1531929.html?...>

Lawsuit against DoubleClick:
<http://www.wired.com/news/print/0,1294,33964,00.html>

CDT's testimony on online profiling:
<http://www.cdt.org/testimony/991108mulligan.shtml>

Comments from Readers

From: Andrew D. Fernandes <andrew@cryptonym.com>
Subject: Publicity Attacks

I have a couple of comments regarding your "Publicity Attack" article in the January Crypto-Gram.

First, you insinuated that my company, Cryptonym, being a PKI vendor, had somehow profited by publicizing the Microsoft/NSA issue. In fact, we didn't make a penny off of the information. Heck, I couldn't sell you anything even if I wanted to. We are at least a year from releasing any products whatsoever -- products that will be fully open source -- and we only take on a very limited amount of consulting.

We thought that we were doing the "Right Thing" by publicizing the fact that all US companies wanting to export crypto software had to involve themselves somehow with the NSA. It was never our intention to make money off of our finding, and so far, we haven't.

Second, you discuss publicity attacks. Apparently, we cannot trust nCipher or eEye to discuss security because they sell security products and consulting. In fact, at the end of the Crypto-Gram is a note indicating that we should be wary of claims about elliptic curves made by Alfred Menezes because he has financial interest in Certicom. I think that this is going too far.

Yes, perhaps nCipher and eEye (and maybe even Cryptonym) did a bad job publicizing security holes. Maybe Alfred wants to fool everybody about the security of elliptic curves for his own nefarious gain. But get real!

Everybody in this field -- including you and Counterpane -- has strong monetary ties to the industry, therefore everything everybody says has to be taken in context. These financial ties do not mean that a company's advice or press releases cannot be trusted. Taking advice about cryptography is no different than taking advice from your lawyer. You have to know how your lawyer will benefit from you taking their advice.

But I think it's presumptuous to imply that we lack professional integrity if you disagree with us. For instance, why are you advising people to use RSA over ECC? Why not ElGamal or Authenticated Diffie-Hellman? After all, RSA is patented, while ElGamal and ADH are not... Wait! Does Counterpane hold any shares of RSA?! Could Counterpane be making money by advising people that ECC isn't as good as RSA?

I certainly hope not. I don't agree with your ECC vs. RSA advice (and neither does Alfred Menezes). However, I do believe in you and your company's professional integrity.

From: Geoff Thorpe <geoff@eu.c2.net>
Subject: nCipher's Key-Finding Attack

I was interested to read your thoughts on the nCipher-announced key-scanning attacks. As someone who works for a company that produces an open-source based secure web-server I would have as much temptation as anyone to play down the importance of such attacks -- especially as the desired inference seems to be that the web-server can't be secure unless one shells out on hardware crypto too! However, I feel it's dangerous to disregard what the underlying work may have to tell us about our security and configurations, regardless of whether or not that involves taking it to the extent that press release may like us to (which is to conclude hardware crypto is the solution to the problem).

I do generally agree with the sentiment that these key-scanning attacks aren't nearly as sensational as the press might like (or be led) to believe -- key scanning is nothing new and this work has simply sped it up quite a bit, and there are only very specialized kinds of web-server usage where these attacks can be mounted against a server that has been set up competently. However it is incompetently set up web-servers that are mostly likely to be discussed to illustrate and justify attacks so we're best to look under the hood to see what we can take and what we can discard from all this.

IMHO, the research behind the announcement is legitimate and worthy of a look -- if for no other reason, to get a more healthy respect for key-protection. In particular, the research uses some interesting ideas on scanning large blocks of data for areas of high-entropy before using

more refined search techniques to track down actual keys in the "haystack". It rather effectively illustrates why one must have a keen eye for scenarios where unencrypted private keys lie in any media subject to brute-force scanning, no matter how impossibly resource intensive the scanning attacks may at first appear. The research in question illustrates how a large hard-disk could be scanned quite quickly to search out a private key that may lie in it (e.g., a swap partition, a core dump, etc.). Everyone knows an unencrypted private key is always a target -- but the research gives some considerable warning to those who might have previously thought 1 private key in 2 Gb of data was too difficult a thing to find.

Also, the attack has at least led us to consider situations where this kind of attack is or is not an issue rather than allowing hackers to do it for us. The truth is that this family of attacks, at least in the case of web-servers, is only a serious threat for multi-hosting servers that run multiple virtual hosts in the same web-server application that is running a secure virtual host (and does not use any setuid on the web-server child processes). In that scenario, an administrator who can upload and activate native (e.g., CGI) programs in their virtual host can not only scan for and find their own private key (if they have one), but also the private keys of any other loaded virtual hosts running in the same processes (or as the same user). By attaching to a running process (as debuggers do) or using some "/proc"-style mechanism to get direct access to a process's virtual memory, you can begin your scanning as you would on a regular file. There are not many servers running in scenarios like this (where an "administrator" can compromise anyone except him/her-self), but the research has at least allowed to us to consider exactly who is and isn't vulnerable (helping the former, and providing some peace-of-mind to the latter) before the lessons are learnt the hard way.

From: Nicko van Someren <nicko@ncipher.com>
Subject: Re: Key Finding

First, I would like to point out a significant inaccuracy in your report. You state in the penultimate paragraph that "[the] nCipher release included a hacker tool". This is incorrect. We have built a tool that efficiently finds SSL server keys and we have shown it to a limited number of web server vendors, but we have NEVER released the code; nor do we have any intention of doing so.

On the broader issue of what you call "publicity attacks," I feel I must defend nCipher's issuance of a press release on this topic. An essential aspect of developing security solutions is finding the weaknesses in existing systems, and when those weaknesses are found it is reasonable to let those who will be affected know. After making the theoretical attacks known in February 1999, we found that many web server vendors felt that the attacks were impractical and ignored the issue. Thus we went on to prove the practicality of these attacks and let the server vendors have the details of our new results. We then went on to let the rest of the world know.

While you are right to say that nCipher has some interest in the results of this research, it is rare for any company to carry out research without having some interest in the results.

We stand by our stance that publication of potential modes of attack is preferable to obscuring them and allowing them to be employed on an unsuspecting world.

From: ruth@innocent.com
Subject: Radio Pirates

The "Radio pirates" story originated in New Scientist, where for all I know it was told in a more balanced way, and probably had a side panel explaining RDS technology and the reason for this vulnerability.

RDS (Radio Data System) lets consumer radios decode a low bit rate digital signal from compatible FM broadcasts. This signal can include a station ID (such as "BBC R4" or "KISS FM", and various other info, but there is also an additional feature which drivers can switch on at their option called TP (Traffic Programme) which switches to local stations when they are broadcasting a travel update.

Just why is this a "Good illustration of the hidden vulnerabilities in digital systems"? Even if you believed the misleading articles from the BBC which says that "radio stays tuned in until ... the driver switches off the RDS feature," you'd still realise that this system degrades nicely to

ordinary FM radio service at the push of a button. Actually those reports are an exaggeration, all the RDS car radios I've ever seen had a button labelled "TP", which when pressed enables or disables just the traffic programme service. This leaves all the other benefits of RDS intact.

I cannot think of a mechanism that would permit radios to identify and ignore pirate TP signals, without going to fully fledged DAB (as the UK inevitably will in the next decade anyway)? Offering an optional additional service which is subject to a potential DOS doesn't seem like such a vulnerability to me.

From: anonymous
Subject: French Smart Card Break

in the December 15, 1999 issue of Crypto-Gram, you wrote:

> A French engineer succeeded in factoring the 640-bit
> RSA key stored in the chip on the card (all French
> "CB" credit cards have had a chip since 1990).

What is clearly established [agreed by Serge Humpich and the Groupement des Cartes Bancaires in court] is that Humpich made some counterfeit Smart Cards, with incremental account numbers, and used them to buy metro tickets in an automatic machine, as a demonstration to the Groupement. This is basically what he is charged for. The judge is expected to release a verdict on February 25, 2000. A summary (in French) of the audience is at <http://www.legalis.net/legalnet/actualite/...>

>From several sources, including Humpich himself on a TV show and this audience report, he was trying to sell his expertise to the Groupement, through a lawyer in an attempt to remain anonymous.

The 640-bit claim originated in the "Pirate Mag" magazine (also known for promoting the idea that PGP has a backdoor). They have an interview <http://www.acbm.com/pirates/num_05/interview.html> [link dead; try http://www.virus.ldh.org/pirates/num_05/interview.html] of Serge Humpich where he claims: - He broke a "fairly solid 96 digits code" [i.e. 320 bits] used by the Groupement for the "CB" credit cards, with details consistent with an RSA signature scheme with simple redundancy. - He made a spectacular demonstration to experts, factoring some special format 640-bit public modulus, guessing the factors had been chosen close to the square root of the modulus and with some special properties. He is clear his method does not work in a general case. He makes no explicit claim 640-bit signatures are used in the counterfeit Smart Cards. I failed to find any independent confirmation of a 640-bit factorization by Humpich, or even any other statement attributed to Humpich that he ever factored a 640-bit RSA key.

Based on undeniable evidence that the Groupement des Cartes Bancaires originally used a systemwide 321 bits public key, and the lack of evidence of wider keys in general use as of early 1999, many believe that the card fraud Humpich demonstrated is a combination of:
- Factoring the systemwide public modulus of 321 bits, which corresponding secret key is used at card issuance to produce a static 320-bit RSA signature held in the card, certifying the 16 digits card number and expiry date; this static signature being checked by POSTs as a simple off-line validation of the card. - Making working Smart Cards holding counterfeit card number, expiry date, and signature thereof.

<http://humpich.com/> [link dead; try http://www.parodie.com/monetique/], a recent Web site on the case with sources apparently close to Serge Humpich, describes how he factored a 321-bit key in 1997, citing the classical MPQS factoring algorithm, modest computing resources, and a Japanese program. They present 640-bit keys as a reasonable choice the Groupement des Cartes Bancaires could have made, but did not.

The same Web site contains on-line scans of publications where French expert Louis C. Guillou, known to be involved in the design of the Smart Card system used by the Groupement des Cartes Bancaires, warns as early as 1988 [in French] then again in 1990 [in English] that the 320 bit key then in still-experimental use by CB is obsolete.
<http://humpich.com/LCguillou_AnnTelecom_No43_1988.jpg>

<http://humpich.com/SmartCard19900810-2.jpg>

It could be that Humpich demonstrated he can break an obsolete system, tried to make money out of it, and as a retaliation is being sued using whatever legal means available.

Make your own opinion.

CRYPTO-GRAM is a free monthly newsletter providing summaries, analyses, insights, and commentaries on computer security and cryptography.

To subscribe, visit http://www.schneier.com/crypto-gram.html or send a blank message to crypto-gram-subscribe@chaparraltree.com. To unsubscribe, visit http://www.schneier.com/crypto-gram-faq.html. Back issues are available at http://www.schneier.com.

Please feel free to forward CRYPTO-GRAM to colleagues and friends who will find it valuable. Permission is granted to reprint CRYPTO-GRAM, as long as it is reprinted in its entirety.

CRYPTO-GRAM is written by Bruce Schneier. Schneier is founder and CTO of Counterpane Internet Security Inc., the author of "Applied Cryptography," and an inventor of the Blowfish, Twofish, and Yarrow algorithms. He served on the board of the International Association for Cryptologic Research, EPIC, and VTW. He is a frequent writer and lecturer on computer security and cryptography.

Counterpane Internet Security, Inc. is a venture-funded company bringing innovative managed security solutions to the enterprise.

http://www.counterpane.com/

next issue
previous issue
back to Crypto-Gram index
