January 15, 2000

Crypto-Gram Newsletter

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:

"Key Finding" Attacks and Publicity Attacks

A couple of weeks ago the New York Times reported a new "key finding" attack. This was a follow-up to some research discussed here some months ago, showing how to search for, and find, public and private cryptographic keys in software because of their random bit patterns.

The company nCipher demonstrated that someone who has access to a Web server that uses SSL can find the SSL private key using these techniques, and potentially steal it. nCipher's press release talked of "a significant vulnerability to today's Internet economy." Huh? Why is this news?

It's not the fact that the SSL private keys are on the Web server. That's obvious; they have to be there. It's not the fact that someone who has access to the Web server can potentially steal the private keys. That's obvious, too. It's not the news that a CGI attack can compromise data on a Web server. We've seen dozens of those attacks in 1999. Even the press release admits that "no information is known to have been compromised using a 'key-finding' attack. Neither nCipher nor the New York Times found anyone who was vulnerable. But wait . . . nCipher sells a solution to this "problem." Okay, now I understand.

I call this kind of thing a publicity attack. It's a blatant attempt by nCipher to get some free publicity for the hardware encryption accelerators, and to scare e-commerce vendors into purchasing them. And people fall for this, again and again.

This kind of thing is happening more and more, and I'm getting tired of it. Here are some more examples:

- An employee of Cryptonym, a PKI vendor, announced that he found a variable with the prefix "NSA" inside Microsoft's cryptographic API. Based on absolutely zero evidence, this was held up as an example of NSA's manipulation of the Microsoft code.
- Some people at eEye discovered a bug in IIS last year, completely

compromising the product. They contacted Microsoft, and after waiting only a week for them to acknowledge the problem, they issued a press release and a hacker tool. Microsoft rushed a fix out, but not as fast as the hackers jumped on the exploit. eEye sells vulnerability assessment tools and security consulting, by the way.

I'm a fan of full disclosure -- and definitely not a fan of Microsoft's security -- and believe that security vulnerabilities need to be publicized before they're fixed. (If you don't publicize, the vendors often don't bother fixing them.) But this practice of announcing "vulnerabilities" for the sole purpose of hyping your own solutions has got to stop.

Here are some examples of doing things right:

- The University of California Berkeley researchers have broken just about every digital cellphone security algorithm. They're not profiting from these breaks. They don't publish software packages that can listen in on cellphone calls. This is research, and good research.
- Georgi Guninski has found a huge number of JavaScript holes over the past year or so. Rather than posting scary exploits and cracking tools that script kiddies could take advantage of, and rather than trying to grab the limelight, he has been quietly publishing the problems and available workarounds. Of course, the downside is that these bugs get less attention from Microsoft and Netscape, even though they are as serious as many others that have received more press attention and thus get fixed quickly by the browser makers. Nonetheless, this is good research.
- The L0pht has done an enormous amount of good by exposing Windows NT security problems, and they don't try to sell products to fix the problems. (Although now that they've formed a VC-funded security consulting company, @Stake, they're going to have to tread more carefully.)
- Perfecto markets security against CGI attacks. Although they try to

increase awareness of the risks, they don't go around writing new CGI exploits and publicizing them. They point to other CGI exploits, done by hackers with no affiliation to the company, as examples of the problem.
- Steve Bellovin at AT&T labs found a serious hole in the Internet DNS system. He delayed publication of this vulnerability for years because there was no readily available fix.

How do you tell the difference? Look at the messenger. Who found the vulnerability? What was their motivation for publicizing? The nCipher announcement came with a Business Wire press release, and a PR agent who touted the story to reporters. These things are not cheap -- the press release alone cost over $1000 -- and should be an obvious tip-off that other interests are at stake.

Also, look critically at the exploit. Is it really something new, or is it something old rehashed? Does it expose a vulnerability that matters, or one that doesn't? Is it actually interesting? If it's old, doesn't matter, and uninteresting, it's probably just an attempt at press coverage.

And look at how it is released. The nCipher release included a hacker tool. As the New York Times pointed out, "thus making e-commerce sites more vulnerable to attack and more likely to buy nCipher's product." Announcements packaged with hacker tools are more likely to be part of the problem than part of the solution.

I am a firm believer in open source security, and in publishing security vulnerabilities. I don't want the digital cellphone industry, or the DVD industry, to foist bad security off on consumers. I think the quality of security products should be tested just as the quality of automobiles is tested. But remember that security testing is difficult and time-consuming, and that many of the "testers" have ulterior motives. These motives are often just as much news as the vulnerability itself, and sometimes the announcements are more properly ignored as blatant selfserving publicity.

New York Times article:
http://www.nytimes.com/library/tech/00/01/biztech/...

NCipher's press release:
http://www.ncipher.com/news/files/press/2000/...

NCipher's white paper (Acrobat format):
http://www.ncipher.com/products/files/papers/pcsws/...

Counterpane -- Featured Research

"A Cryptographic Evaluation of IPsec"

N. Ferguson and B. Schneier, to appear

We perform a cryptographic review of the IPsec protocol, as described in the November 1998 RFCs. Even though the protocol is a disappointment -- our primary complaint is with its complexity -- it is the best IP security protocol available at the moment.

http://www.schneier.com/paper-ipsec.html

News

You can vote via the Internet in the Arizona Democratic primary. Does anyone other than me think this is terrifying?
http://dailynews.yahoo.com/h/nm/19991217/wr/... [dead link as of 2000-02-18]

An expert at the British government's computer security headquarters has endorsed open-source solutions as the most secure computer architecture available:

http://212.187.198.142/news/1999/50/ns-12266.html

The DVD Copy Control Association is pissed, and they're suing everyone in sight.
http://www.cnn.com/1999/TECH/ptech/12/28/dvd.crack/

Moore's Law and its effects on cryptography:
http://www.newscientist.com/ns/20000108/newsstory2.html

Information warfare in the Information Age:
http://www.cnn.com/1999/TECH/computing/12/30/...
http://www.it.fairfax.com.au/industry/19991227/...

Radio pirates: In the U.K., some radios can receive a digital signal that causes them to automatically switch to stations playing traffic reports. Hackers have figured out how to spoof the signal, forcing the radio to always tune to a particular station. Good illustration of the hidden vulnerabilities in digital systems.
http://news.bbc.co.uk/hi/english/sci/tech/...
http://uk.news.yahoo.com/000106/18/d6jt.html

Well, this sure is inaccurate:
http://www.lancrypto.com/algorithms_e.htm [dead link as of 2000-04-28]

Some months ago I mentioned the Y2K notice from Hart Scientific. They now have a sequel:
http://www.hartscientific.com/y2k-2.htm [dead link as of 2000-04-28]

RSA "digital vault" software:
http://news.excite.com/news/pr/000111/...

E-commerce encryption glitch; a good example of why people are the worst security problem. A programmer just forgot to reactivate the encryption.
http://news.excite.com/news/r/000107/17/...

Become an instant cryptography portal. Encryption.com, encryption2000.com, and 1-800-ENCRYPT are for sale.
http://news.excite.com/news/bw/000111/...
http://www.encryption.com

Mail encryption utility that lets you take back messages you regret sending. Does anyone believe that this is secure?
http://www.zdnet.com:80/anchordesk/story/...

Human GPS implants:
http://www.newscientist.com/ns/20000108/newsstory8.html

Clinton's hacker scholarships:
http://chronicle.com/free/2000/01/2000011001t.htm

Microsoft is building a VPN into Windows 2000. Whose tunnel do you want to hack today?
http://www.networkworld.com/news/2000/0110vpn.html

Someone stole a bunch of credit card numbers from CD Universe, tried extortion, then posted some:
http://www.wired.com/news/technology/...
http://www.msnbc.com/news/355593.asp [dead link as of 2000-02-18]
and Cybercash's reaction (with a nice quote about how impregnable their product's security is; way to wave a red flag at the hackers):
http://www.internetnews.com/ec-news/article/... [dead link as of 2000-04-28]

An interesting three-part article about video surveillance and its effect on society:
http://www.villagevoice.com/issues/9840/boal.shtml

The system used to fund a series of anti-Bush commercials loosely resembles my "street performer protocol," using the credit card company

instead of a publisher as a trusted third party. They validate your card when you pledge, but only charge it if they get enough to run an ad:
http://www.gwbush.com/
Street performer protocol:
http://www.schneier.com/paper-street-performer.html

You can steal subway rides on the NY City system by folding the Metrocard at precisely the right point. The Village Voice and NY Times ran stories about it, but those are no longer available, at least for free. There's a copy of the NYTimes story here:
http://www.monkey.org/geeks/archive/9801/msg00052.html
The 2600 "Off the Hook" RealAudio for 2/3/98 talks about it, starting around 54:35. The RealAudio is linked from here:
http://www.2600.com/offthehook/1998/0298.html

The White House released a national plan to protect America's computer systems from unauthorized intrusions. This plan includes the establishment of the controversial Federal Intrusion Detection Network (FIDNET), which would monitor activity on government computer systems. (So far, there are no plans to monitor commercial systems, but that can change. The government does want to involve industry in this.) The plan also calls for the establishment of an "Institute for Information Infrastructure Protection" and a new program that will offer college scholarships to students in the field of computer security in exchange for public service commitments. The scholarship program seems like a good idea; we need more computer security experts.
http://www.thestandard.com/article/display/...
http://dailynews.yahoo.com/h/ap/20000107/ts/...
http://news.excite.com/news/ap/000107/01/...
http://www.msnbc.com/news/355783.asp
http://www.computerworld.com/home/print.nsf/all/...
EPIC analysis:
http://www.epic.org/security/CIP/

White House plan (PDF):
http://www.whitehouse.gov/WH/EOP/NSC/html/documents/...
White House press release:
http://www.epic.org/security/CIP/WH_pr_1_7_00.html
White House press briefing:
http://www.epic.org/security/CIP/...

New U.S. Encryption Regulations

We have some, and they're a big improvement. On the plus side, "retail" encryption products -- like browsers, e-mail programs, or PGP -- will be widely exportable to all but a few countries "regardless of key length or algorithm." On the minus side, the new regulations are complex (an unending stream of work for the lawyers) and will still make it difficult for many people to freely exchange encryption products. They also do not address the Constitutional free speech concerns raised by encryption export controls.

Major features of the new regs:

- "Retail" encryption products are be exportable, regardless of key length or algorithm, to all but the designated "T-7" terrorist nations. In order to export you need to fill out paperwork. You need to get a retail classification, submit your product to a one-time technical review, and submit periodic reports of who products are shipped to (but not necessarily report end users).
- Export of encryption products up to 64 bits in key length is completely liberalized.
- "Non-retail" products will require a license for many exports, such as to foreign governments or foreign ISPs and telcos under certain circumstances.
- Source code that is "not subject to an express agreement for the

> payment of a licensing fee or royalty for commercial production or sale of any product developed with the source code" is freely exportable to all but the T-7 terrorist countries. Source code exporters are required to send the Department of Commerce a copy of the code, or a URL, upon publication. Note that posting code on a web site for anonymous download is allowed; you are not required to check that downloaders might be from one of the prohibited countries.

One obvious question is: "How does this affect the Bernstein and Karn court cases?" I don't know yet. The free speech concerns are not addressed, but the things that Bernstein and Karn wanted to do are now allowed. We'll have to see what the attorneys think.

A more personal question is: "How does this affect the Applied Cryptography source code disks?" Near as I can tell, all I have to do is notify the right people and I can export them. I will do so as soon as I can. Stay tuned.

The actual regs (legalese):
http://www.eff.org/pub/Privacy/ITAR_export/...

EFF's press release:
http://www.eff.org/11300_crypto_release.html

Reuters story with BSA and Sun reactions:
http://news.excite.com/news/r/000112/19/...

Reuters story with EFF reaction:
http://news.excite.com/news/r/000113/13/...

AEA reaction press release:
http://news.excite.com/news/pr/000112/...

ACLU and EPIC reaction:

http://news.excite.com/news/zd/000113/18/...

Counterpane Internet Security News

Bruce Schneier profiled in Business Week:
http://businessweek.com/cgi-bin/ebiz/ebiz_frame.pl?...

Bruce Schneier is speaking at BlackHat in Singapore, 3-4 April 2000. He'll also be at BlackHat and DefCon in Las Vegas.
http://www.blackhat.com
http://www.defcon.org

Bruce Schneier is speaking at the RSA Conference in San Jose: Tuesday, 18 Jan, 2:00 PM, on the Analyst's Track. I don't know if it made it into the program, but Bruce will be on stage with Matt Blaze, Steve Bellovin, and several other really smart people.

The Doghouse: Netscape

Netscape encrypts users' e-mail passwords with a lousy algorithm. If this isn't enough, their comments to the press cement their inclusion in the doghouse:

"Chris Saito, the senior director for product management at Netscape, said that the option to save a password locally was included for convenience. Saito added that Netscape didn't use a stronger encryption algorithm to protect passwords so that 'computer experts could still access the information, in case someone forgot their password.'"

In other words, they implemented lousy security on purpose.

"Netscape's Saito said the company wasn't aware of the vulnerability and added that a 'security fix' would be forthcoming if that vulnerability were proved to exist. If the Javascript vulnerability doesn't exist, a password stealer would have to have physical access to a user's computer to figure out the algorithm."

Note the complete ignorance of viruses like Melissa, or Trojan horses like Back Orifice.

"Saito noted that Netscape already has numerous safety features, including a Secure Sockets Layer, which enables users to communicate securely with Web servers, and a protocol for encrypting e-mail messages sent."

None of which matters if the password is stolen.

http://www.zdnet.com/zdnn/stories/news/...

RST's information:
http://www.rstcorp.com/news/bad-crypto.html
http://www.rstcorp.com/news/bad-crypto-tech.html

Block and Stream Ciphers

Block and stream ciphers both transform a message from plaintext to ciphertext one piece at a time. Block ciphers apply the same transformation to every piece of the message, and typically deal with fairly large pieces of the message (8 bytes, 16 bytes) at a time. Stream ciphers apply a different transformation to each piece of the message, and typically deal with fairly small pieces of the message (1 bit, 1 byte) at a time.

Traditionally they have been separate areas of research, but these days they are converging. And if you poke around at the issues a bit, you'll see that they not very different at all.

Stream ciphers first. Traditional stream ciphers consist of three standard pieces: an internal state, a next-state function, and a plaintext-to-ciphertext transformation function. The internal state is generally small, maybe a hundred bits, and can be thought of as the key. The next-state function updates the state. The transformation function takes a piece of plaintext, mixes it with the current state, and produces the same size ciphertext. And then the stream cipher goes on to the next piece.

The security of this scheme is based on how cryptographically annoying the two functions are. Sometimes just one of the functions is cryptographically annoying. In electronic stream ciphers, a complicated next-state function is usually combined with a simple transformation that takes the low-order bit of the state and XORs it with the plaintext. In rotor machines, such as the German Enigma, the next-state function was a simple stepping of various rotors, and the transformation function was very complicated. Sometimes both are cryptographically complicated.

These ciphers could generally operate in two modes, depending on the input into the next-state function. If the only input was the current state, these were called output-feedback (OFB) ciphers. If there was the additional input of the previous ciphertext bit, these were called cipher-feedback (CFB) ciphers. (If you were in the U.S. military, you knew these modes as "key auto-key" (KAK) and "ciphertext auto-key (CTAK), respectively.) And you chose one mode over the other because of error propagation and resynchronization properties. (Applied Cryptography explains all this in detail.)

Traditionally, stream cipher algorithms were as simple as possible. These were implemented in hardware, and needed as few gates as possible. They had to be fast. The result was many designs based on simple mathematical functions: e.g., linear feedback shift registers (LFSRs). They were analyzed based on metrics such as linear complexity and correlation immunity. Analysts looked at cycle lengths and various linear and affine approximations. Most U.S. military encryption algorithms, at least the ones in general use in the 1980s and before, are stream ciphers of these sorts.

Block ciphers are different. They consist of a single function: one that takes a plaintext block (a 64-bit block size is traditional) and a key and produces a ciphertext block. The NSA calls these ciphers codebooks, and that is an excellent way to think of them. For each key, you can imagine building a table. On the left column is every possible plaintext block; on the right column is every possible ciphertext block. That's the codebook. It would be a large book, 18 billion billion entries for the smallest commonly used block ciphers, so it is easier to just implement the algorithm mathematically -- especially since you need a new book for each key. But in theory, you could implement it as a single table lookup in a very large codebook.

Block ciphers can be used simply as codebooks, encrypting each 64-bit block independently (and, in fact, that is called electronic codebook (ECB) mode), but that has a bunch of security problems. An attacker can rearrange blocks, build up a portion of the codebook if he has some known plaintext, etc. So generally block ciphers are implemented in one of several chaining modes.

Before listing the block cipher chaining modes, it's worth noticing that a block cipher algorithm can serve as any of the functions needed to build a stream cipher: the next-state function or the output function. And, in fact, that is what block cipher modes are: stream ciphers built using the block cipher as a primitive. A block cipher in output-feedback mode is simply the block cipher used as the next-state function, with the output of the block cipher being the simple output function. A block cipher in cipher-feedback mode is the same thing, with the addition of the ciphertext being fed into the next-state function. A block cipher in counter mode uses the block cipher as the output function, and a simple counter as the next-state function. Cipher block chaining (CBC) is another block-cipher mode; I've seen the NSA call this "cipher-driven codebook" mode. Here the block cipher is part of the plaintext-to-ciphertext transformation function, and the next-state function is simple.

For some reason I can't explain, for many years academic research on block ciphers was more practical than research on stream ciphers. There were more concrete algorithm proposals, more concerted analysis, and more implementations. While stream cipher research stayed more theoretical, block ciphers were used in security products. (I assume this was the reverse in the military, where stream ciphers were used in products and were the target of operational cryptanalysis resources.) DES's official sanction as a standard helped this, but before DES there was Lucifer. And after DES there was FEAL, Khufu and Khafre, IDEA, Blowfish, CAST, and many more.

Recently, stream ciphers underwent something of a renaissance. These new stream ciphers were designed for computers and not for discrete hardware. Instead of producing output a bit at a time, they produced output a byte at a time (like RC4), or 32 bits at a time (like SEAL or WAKE). And they were no longer constrained by a small internal state -- RC4 takes a key and turns it into a 256-byte internal state, SEAL's internal state is even larger -- or tight hardware-based complexity restrictions. Stream ciphers, which used to be lean and mathematical, started looking as ugly and kludgy as block ciphers. And they started appearing in products as well.

So, block and stream ciphers are basically the same thing; the difference is primarily a historical accident. You can use a block cipher as a stream cipher, and you can take any stream cipher and turn it into a block cipher. The mode you use depends a lot on the communications medium -- OFB or CBC makes the most sense for computer communications with separate error detection, while CFB worked really well for radio transmissions -- and the algorithm you choose depends mostly on performance, standardization, and popularity.

There's even some blurring in modern ciphers. SEAL, a stream cipher, looks a lot like a block cipher in OFB mode. Skipjack, an NSA-designed block cipher, looks very much like a stream cipher. Some new algorithms can be used both as block ciphers and stream ciphers.

But stream ciphers should be faster than block ciphers. Currently the fastest block ciphers encrypt data at 18 clock cycles per byte (that's Twofish, the fastest AES submission). The fastest stream ciphers are even faster: RC4 at 9 clock cycles per byte, and SEAL at 4. (I'm using a general 32-bit architecture for comparison; your actual performance may vary somewhat.) I don't believe this is an accident.

Stream ciphers can have a large internal state that changes for every output, but block ciphers have to remain the same. RC4 has a large table -- you can think of it as an S-box -- that changes every time there is an output. Most block ciphers also have some kind of S-box, but it remains constant for each encryption with the same key. There's no reason why you can't take a block cipher, Blowfish for example, and tweak it so that the S-boxes modify themselves with every output. If you're using the algorithm in OFB mode, it will still encrypt and decrypt properly. But it will be a lot harder to break for two reasons. One, the internal state is a moving target and it is a lot harder for an attacker to build model of what is going on inside the state. Two, if the plaintext-to-ciphertext transformation is built properly, attacks based on chosen plaintext or chosen ciphertext are impossible. And if it is a lot harder to break a cipher with self-modifying internals, then you can probably get by with fewer rounds, or less complexity, or something. I believe that there is about a factor of ten speed difference between a good block cipher and a good stream cipher.

Designing algorithms is very hard, and I don't suggest that people run out and modify every block cipher they see. We're likely to continue to use block ciphers in stream-cipher modes because that's what we're used to, and that's what the AES process is going to give us as a new standard. But further research into stream ciphers, and ways of taking advantage of the inherent properties of stream ciphers, is likely to produce families of algorithms with even better performance.

Comments from Readers

> **From: Markus Kuhn <Markus.Kuhn@cl.cam.ac.uk>Subject: German smart-card hack**

> The note on "German hackers have succeeded in cracking the Siemens digital signature chip" in the 1999-12-15 CRYPTO-GRAM is wrong. I have been in contact with the German Hacker (Christian Kahlo) behind this story. He discovered that one user of the Siemens SLE44 chip series included in his ROM software a routine that allowed him to upload and execute not only interpreter bytecode, but also raw 8052 assembler instructions. Using this undocumented facility, Christian uploaded a tiny assembler program that dumped the entire ROM of the card. The ROM was investigated, posted on the USENET as a documented disassembler listing in a TeX file and no vulnerabilities were found. Christian also discovered in the ROM that the SLE chips send out the chip type and serial number when the I/O line is held low during a positive reset edge and the following 600-700 clock cycles, which is a perfectly normal feature (comparable to the BIOS power-up message of a PC) that is fully documented in the SLE44 data sheets and that is not security relevant.

> No smartcard applications were hacked this way, no vulnerability was found in any smartcard application, and definitely no private keys were compromised. All this also has nothing to do with digital signatures. Any news to the contrary is the result of misunderstandings by

> journalists, who as usual fill in the gaps of the story with their limited technical background knowledge and try to formulate such reports to be more spectacular than the story behind them. The only policy that has been violated here is that Siemens -- like most other smartcard chip producers -- tries to make sure that nobody except big customers can easily get access to smartcard development kits that allow to upload assembler code directly, which might otherwise shorten the learning curve for a microprobing attacker slightly. Users of Siemens chips that allow code uploads are apparently required to use a bytecode interpreter instead. This policy seems to have been ignored secretly by one Siemens customer who left a backdoor in his byte-code interpreter to enable the later upload of high-speed crypto routines that cannot be implemented sufficiently efficient in the bytecode.

> Christian discovered this, even though he decided *not* publish the details on how he did this or the name of the Siemens customer in whose cards he had discovered this. All he published was a dump of the standard Siemens SLE ROM code (CMS = Chip Management System, comparable to a PC BIOS), a piece of code that had already been known semi-publicly for many years in the pay-TV hacking community from successful microprobing attacks on the SLE44 series. Christian's main contribution is that he has discovered a very nice low-cost assembler-level development kit for some of the SLE smartcards, which used to cost a fortune and an NDA before. This is not the first time that this has happened: Pay-TV smartcards have been shipped before with software that
> provides for uploads of EEPROM software patches with broken authentication techniques, which has been known and used in the smartcard tampering community for many years.

> **From: anonymous**
> **Subject: Re: New U.S. Crypto Export Regulations**

> In CRYPTO-GRAM of December 15, 1999 you wrote about the proposed new U.S. crypto export regulations, and I can agree with everything you said. However, I believe you missed something important: the view FROM the rest of the world.

> I work in the finance industry in Europe -- Zurich, to be precise -- and have some involvement with security. This industry (a) WILL NOT use U.S. crypto products, and (b) will certainly NOT make any long-term plans or partnerships to do so for U.S. products with consumer content, because (a) the products to date are forced by law to be weak, but more important, (b) the U.S. government can't be trusted. Even if it approved today the export of some products based on strong crypto, everyone knows that this permission could be terminated tomorrow for the same or other products. And everyone also suspects strongly that the U.S. government will in any case force providers to put trap doors into their products.

> Under the circumstances, the European finance and e-business industries would be have to be crazy to use U.S. crypto-based products. And they're not crazy.

> To play in this business in the rest of the world, the U.S. will have to have a clear, consistent, and favorable policy, and U.S. companies will have to present products that are demonstrably strong with no trap doors. (I invite you to speculate if this will happen before Hell freezes over.) In the meantime, there are plenty of non-U.S. products to choose from, and banks like UBS, Credit Suisse, Grupo Intesa, Societe General, Deutsche Bank, Generale Bank, Bank Austria, and Barclays are not sitting back anxiously waiting for U.S. products to become available. They're doing business with non-U.S. products that are just fine, thank you.

> **From: "Grawrock, David" <david.grawrock@intel.com> Subject: Electronic voting**

> All these comments regarding electronic voting and absentee voting are missing the mark. The State of Oregon has that all elections (except presidential) are done by mail. It's like the entire state is voting absentee.

> The process is actually pretty painless. You receive your voter pamphlet and then you get your ballot. It has to be in by election day. If you miss the excitement of going to the voting booth there are collection points where you can drop off your filled in ballot. It's really not that hard.

> The point here is that the state has determined that it is easier (and cheaper) to simply process the entire election via the absentee process. It now becomes a simple step to go from by mail to by electronic voting. All of the arguments regarding coercion must already have been answered (the government always thinks a process through completely). We have elected all sorts of politicians without anyone coming back and reporting problems with coercion.

> **From: Gerry Brown <gerry@liberate.com>Subject: RE: Absentee Ballots**

> I just checked some figures with a friend who has the data on Absentee Ballots for San Mateo County in California and he has compared it with the San Francisco elections held this week.

> The percentage of registered voters using absentee ballots is about 13%-15%. But the more astonishing is the fact that 35%-50% of those actually voting are done by absentee ballots. The lower figure is for national elections and the higher side corresponds to local elections.

> **From: "Hillis, Brad" <BradH@DIS.WA.GOV>Subject: PKI article--agree and disagree**

> I can't begin to tell you how much I enjoyed your article with Carl Ellison, "Ten Risks of PKI: What You're not Being Told about Public Key." I'm the lead ecommerce attorney for the state of Washington, and we are currently procuring a private PKI vendor to provide digital signatures for state and local government, similar to the federal government ACES procurement.

> What you say that PKI is not needed for ecommerce to flourish is true. It's a thought I keep having at all the digital signature law presentations I attend, and the theme I had planned to discuss at my March 7 talk in Boston on PKI. One has to keep asking oneself, why do I need a digital signature? What is the opportunity cost of setting up a PKI? (That is, what security improvements could I make if I spent the money on something besides PKI).

> However, I disagree with this statement in your article:

> "In other words, under some digital signature laws (e.g., Utah and Washington), if your signing key has been certified by an approved CA, then you are responsible for whatever that private key does. It does not matter who was at the computer keyboard or what virus did the signing; you are legally responsible."

> The law seems to say that at first reading, but my view of the law is that it sets up a "rebuttable presumption" of non-repudiation. This is the same rule that applies to physical, pen and ink signatures. Your statement reflects the views of some proponents of PKI who overstate the legal force of a "licensed digital signature" under Washington law. But if, in fact, I never applied my digital signature to a document, and I can prove it (e.g., I have an alibi), then I would not be legally responsible. I believe that is the situation in non-PKI electronic signature schemes, where a (paper and manually signed) Electronic Data Interchange Agreement or Trading Partner Agreement will state that all data submitted between the parties carries the same legal

> force as if it was manually signed.

> Having found flaws in the PKI-style laws of Washington, Utah and Minnesota, I do not find a great deal of higher or practical intelligence in the more popular electronic signature laws, either. Esignature laws have not proven any more important to ecommerce than PKI digital signature laws, so why are we in such a rush to pass UETA (uniform electronic transaction act)?

> **From: "Carl Ellison" <cme@acm.org>Subject: Re: PKI article--agree and disagree**

> You are correct. However, I believe we still need to warn against the rebuttable presumption of non-repudiation. The keyholder may have no alibi at all. The keyholder may not be aware that his key was misused (e.g., by an attacker who had gained physical or network access to his computer).

> This is similar to the position people were in in Britain when they were challenging ATM card operations. It took expert witnessing by Ross Anderson to defend some of their claims, and even then it didn't always work. There, too, the presumption was that the cardholder performed any operation when the ATM logs said he did -- whether he did or not. It was up to the cardholder to prove the negative.

> This gets even worse when the keyholder has his private key on a smartcard in his possession. It's that much harder to convince a jury that you didn't sign, if the merchant or bank can claim that the signing key never left your personal possession. When an attacker has network access to your computer, he doesn't leave a trail. You have no audit record showing the attack. It's your word against the merchant's and you have no evidence to offer on your behalf. You can't even accuse anyone else. You have no idea who to accuse.

> Meanwhile, your account has been debited until you manage to prove your point (against the presumption that you're lying). When you compare this to credit card purchases, it's radically different. With a credit card, you have not spent anything until you write the check to the credit card company. When or before you write that check, you can challenge a line item and force the merchant to prove that you were in fact the purchaser. At least with my AMEX account, the immediate result is that AMEX removes the item from my statement -- to be reinstated if the merchant is able to prove that I did do the purchase. I have had such challenges go my way once and the other times, I had simply forgotten. In one case, I thought I was being double-billed, but it turns out I had never been billed the first time (many months before).

> **From: Alfred John Menezes <ajmeneze@ cacr.math.uwaterloo.ca>Subject: Elliptic Curve Cryptosystems**

> I read with interest your recent article on ECC in the November 15 issue of Crypto-Gram. I agree with most of your statements and comments.

> Your recommendations were:
> 1) If you're working in a constrained environment where longer keys just won't fit, consider elliptic curves. 2) If the choice is elliptic curves or no public-key algorithms at all, use elliptic curves.
> 3) If you don't have performance constraints, use RSA. 4) If you are concerned about security over the decades (and almost no systems are), use RSA.

> I certainly agree with recommendations 1) and 2) -- ECC certainly cannot be worse than no security at all!

> Regarding recommendation 3), I think that most environments which call for public-key solutions will have *some* performance constraints. The limiting factor could be an over-burdened web server which needs to sign thousands of outgoing messages per minute, a handheld

> device which is communicating with a PC, etc. In such scenarios, one should select the public-key method that performs the best in the most constrained environment. If the constraints involve key sizes, bandwidth, power consumption, or speed (for private key operations), then ECC is likely the method of choice over RSA.

> Finally, I feel that your recommendation that RSA should be used (instead of ECC) in situations where you are concerned with long-term security is a bit unfair. After all, as you state in the postscript to your article, all the analysis you used on the elliptic curve discrete logarithm problem also applies to the integer factorization problem. I propose that applications which do require long-term security should consider using both* RSA and ECC -- by double encrypting a message with RSA and ECC, or by signing a message twice with RSA and ECC.

> The following are my condensed thoughts on the security and efficiencies of ECC as compared with RSA. They should be considered a supplement to your Crypto-Gram article, and not a replacement of it.

> http://www.cacr.math.uwaterloo.ca/~ajmeneze/misc/...

> ((This is a good essay, but remember the author's bias. He works for Certicom, and it is in his financial interest for you to believe in elliptic curves. --Bruce))

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

Schneier.com is a personal website. Opinions expressed are not necessarily those of BT.