RECORD, Volume 29, No. 2*

Spring Meeting, Vancouver, B.C.
June 23–25, 2003

Session 37OF
Electronic Security in the Internet Age

Track: Computer Science

Moderator: BRIAN M. SEPTON
Panelist: SHAWN MOYER†

Summary: When was the last time you used e-mail or the Internet? Are you "logged on" right now? Do you conduct business over the Web (including correspondence with colleagues and clients, document sharing, financial transactions or research activities)? Have you ever wondered what makes certain types of Internet access secure? This session is intended to improve your understanding of the history, current status and future of Internet security, and increase your awareness of common vulnerabilities.

MR. BRIAN M. SEPTON: Welcome to "Electronic Security in the Internet Age." We're going to spend an hour and a half talking about Internet security and electronic security. We're going to have an actual hacking demonstration the last five minutes.

I am an FSA with Chicago Consulting Actuaries. I mainly work as a pension practitioner but in the past couple of years my experience has branched off into the area of technology, software and actually into the database world. I am a member of the SOA's Computer Science Section board and a technophile, a kind of encryption hobbyist, you might say. My interest in encryption and Internet security started about three or four years ago when I read a book called "The Code Book" by Simon Singh, which taught me a lot of what we're going to see here today.

MR. SHAWN MOYER: I'm with Reinsurance Group of America (RGA) out of St. Louis, and I'm not an actuary. I'm the project lead for information security at RGA,

---

†Mr. Moyer, not a member of the sponsoring organizations, is project lead for information security at Reinsurance Group of America in St. Louis, Mo.

Note: The chart(s) referred to in the text can be found at the end of the manuscript.

which means that I work with our legal, information technology (IT) and some other folks on making sure that RGA is secure, dealing with legislation, some computer intrusion stuff and things like that. I'm here to talk about what I do and complement what Brian has to say.

I will first talk about information security at 50,000 feet—a high-level view of what information security is as a practice in the computer sciences. We're the only specialization that actually gets worse every year. The number of systems that are compromised, the number of intrusions and the number of virus infections continue to go up. At the very least, from year to year they'll stay the same. We're not seeing any kind of extreme downward turn in the number of compromises. What leads to this? The problem is that, as other areas grow and develop, they create new exposures. We have more connectivity; we have wireless technology; we have mobile users and people using systems from home. We have this exponential growth of the Internet (especially in the mid-1990s), although the growth has tapered off in the past couple of years. Any new endeavor introduces new risk. That's what's going on in the computer industry.

In the 1980s, computer systems were very rarely interconnected. You just had a central mainframe, maybe a small corporate network or possibly connected to another network like two universities connected together, but most of the time you didn't have this kind of interconnectivity. Now what we have is the "Web of trust." The idea of the Web of trust is that you have systems that are interconnected in this peering arrangement, which is what the Internet is. The Internet is basically a loose conglomeration of corporate networks, universities, military networks and so on. All these networks are connected in an arbitrary peering arrangement. The problem with the Web of trust is that we're only as strong as the weakest link. If there are any systems that are vulnerable, then we're sharing in that exposure, because we're all interconnected.

Another analogy is that the Internet is a bad neighborhood. The idea is that when you move out to one of these new developments, it's a bedroom community. In the early days you leave the doors unlocked and you don't have to worry too much about your car. Then that one family moves in with the kids that are a little bit rough around the edges. It's the same idea with the Internet. As we have these systems that keep being loaded up onto the Internet that aren't secure, we all share in that and we all share in that exposure. Innovation equals exposure.

There are some proposed solutions to this. There are a couple of different tacks that people are taking. My favorite, one that I will pound on over and over again when talking to folks about security, is that right now in the computer industry, features are what sell software. When you go out and buy the latest version of Microsoft Word or Microsoft Excel or something like that, you're not buying it because it says: "Now with security!" That's very rarely a purchasing decision that people are making. They're buying it based on features. In the industry, there's an impetus to increase feature for feature and meet your competitors with more

features. Security is not really a selling point on systems. In the past couple of years, as we've seen more and more press coverage about viruses and about compromises, the knowledge level is starting to increase and people are starting to ask for security. They're starting to make that part of a purchasing decision on software.

Another initiative that's been seeing some exposure in the past year or so is what's called trustworthy computing. Trustworthy computing is a system that uses encryption as a method to identify a piece of software or a piece of hardware. If you build a crypto system all the way end-to-end into a computer—from a hardware level and integrate the hardware with the software—you have what's called a public key infrastructure (PKI). That allows you to have a "stamp," an encrypted signature, on a piece of code or a piece of hardware. Most security compromises happen because things that aren't supposed to be there are running on a system. Maybe you're getting a program to load up after the fact, and it shouldn't be there. That wouldn't be able to happen, in theory, because this software that's loaded on is stamped and certified by the central authority. The whole concept of trustworthy computing is that you build that in. At a hardware level, you can't load a piece of code that's not certified.

Now, the issue with that is: Who certifies it? Who's going to make that decision? You may have encountered one of the early ideas of trustworthy computing if you've ever clicked on a Web page and it wanted to load an applet, and that applet popped up and said "This is not signed by Microsoft," or it said, "Do you always want to trust software signed by Microsoft Corporation? Click yes or no." Instead of that same concept, we're building it into the hardware and making it mandatory.

The problem is: Who's the authority? Who's going to sign it? Right now the two ideas that are being proposed are Microsoft or the government. I'm not very happy about either one of those. That's one of the issues. The other issue is standardization. You have to get every single vendor to agree, and you have to get all these things integrated. That's very hard to do as well. We'll see how that goes.

Another area is education and awareness. This is the idea of information security starting to exist and becoming a field. Just two or three years ago, you didn't even have a security role in a lot of companies. Now you're starting to see companies have a chief privacy officer or a chief security officer. You're starting to see universities such as Carnegie-Mellon and Purdue have a computer science degree with an emphasis in information security. The Certified Information Systems Security Professionals (CISSP) and Security Systems Certified Practitioners (SSCP) are similar organizations to the SOA. These are certifying boards that certify somebody in the computer science field to be a practitioner of information security. These are the things you're going to see legitimize this field and make it built into the whole process.

Legislation is not a favorite of everybody, but because there have been a number

of compromises that have been very public (Social Security numbers, credit information), the government has started to get involved. They started to release privacy regulations that are governing the holding of private data, so you have the Health Insurance Portability and Accountability Act of 1996 (HIPAA) in the United States, which anybody who's involved with health care has probably encountered. You also have the Gramm-Leach-Bliley Act (GLBA), which is similar to HIPAA, but applies more to the financial sector in terms of banking and insurance information. In the European Union and the United Kingdom, you have the Data Protection Act 1998 (DPA). In Canada, we have the Personal Information Protection and Electronic Documents Act (PIPEDA).

One of the problems with this legislation is if you're an organization that works at an international level, who do you comply with? If your company is involved globally, what privacy regulations do you need? The consensus right now is to comply with the strongest regulations in the area that you're operating. If you're operating in the United Kingdom, then comply with the DPA because it's the most stringent and that would generally put you in compliance with everything else.

Compliance is one of the problems of these regulations, and vagueness is another problem. One of the problems with writing a piece of legislation is that you can't say the specific type of encryption which must be used because two years from now that might not be as good as a control. You have to make these pieces of legislation non-specific. That creates the problem—how do we know if we're really complying? They use a lot of terms like "due care," or "due diligence" and "good faith effort." There haven't been any cases to define the case law and what those terms are going to mean. We all scramble, do the best that we can, cross our fingers and hope we're doing okay.

The big picture is that security is everybody's job. This is back on the subject of integrating security across the board. You have to have developers who license their programs. You have to have legislation. You have to have all these pieces so that everybody is aware of information security. You can't think of it as a bolt-on, plug-in solution that you just integrate in and snap onto your existing infrastructure. It's something you have to build in from the ground up.

Security in-depth is another strategy that I always emphasize. One of the problems people run into when planning information security is how they are going to secure their networks. I get this a lot in working in the security industry. I used to consult for a while. Companies would say, "We want to be unhackable. We want you to promise us we're unhackable and nobody can get into our network. That's what we want." We hear people claiming that they're unhackable. Any security person who will say he can do that for you is probably somebody you should investigate moving to another position because there's always going to be some vulnerability, some exposure. The idea of security in-depth is that you take a number of controls that aren't perfect—none of these are making you unhackable—but the aggregate of those controls, if layered on top of each other, will be stronger than one single

"unbreakable" control. That's the idea of security in-depth. Brian uses the analogy of a fish net. If you take a fish net, which is full of holes like the Internet is full of holes, and you lay this fish net on top of this fish net and on top of this fish net, eventually it's no longer transparent. It's the same idea. You just integrate this across the board and put a number of controls in place.

There's no magic bullet. There's no bolt-on solution. It's just a number of different ideas that you try to bring together into a single offering for better security.

MR. SEPTON: Let's talk about encryption. We want to talk about the history of encryption. We'll talk about the never-ending quest for unbreakable ciphers. We'll talk about the contributions made on behalf of Diffie, Hellman and Merkle. We'll look at the RSA and Pretty Good Privacy (PGP) algorithms and explain what that alphabet soup is all about. Then we'll spend a few minutes talking about the way Secure Sockets Layer (SSL) and S-HTTP encryption algorithms are actually used in the Internet nowadays.

Encryption is thousands of years old. Encryption is as old as secret communication. There are multiple ways of hiding messages. The first is steganography, or stego, as some people call it. That's concealing the existence of a message. The second is cryptography, which would be concealing the meaning of a message.

In terms of the branches of cryptography, there are codes and ciphers. What's the difference? A code is a substitution at the phrase or the word level. For example, "attack" is replaced by "retreat," "today" is replaced by "tomorrow," so "attack today" becomes "retreat tomorrow." That is actually a code. A cipher is a substitution at the letter level, which is what I usually think of when I think of cryptography. A cipher involves transposition, where you exchange letters within a given word, and substitution. We can substitute the "SOA" for the "CAS."
Stego was first used by the ancient Greeks. They would hide messages inside containers. They would write a message inside a barrel, melt wax over it and therefore hide the message. There are some occasions where they shaved the head, wrote the message, let the hair grow back and then sent the messenger in with this hidden message. This was probably used for the less urgent communication, since hair still probably grew at the slow rate it does now.

As one example in history where encryption has been used or attempted to be used to hide communication, we have Mary, Queen of Scots. She was involved in a plot to murder Queen Elizabeth, her cousin in England. Mary, queen of Scotland, was actually imprisoned and used a secret cipher to communicate with her followers and disciples. Little did she know that the chief cryptographer of England at the time was also among her jail guards. He was able to break her code, which ultimately lead to her conviction on conspiracy. I'm told she only got a life sentence, although I'm sure she was beheaded.

In terms of encryption, we have the desire for unbreakable ciphers and the desire

for ultimate secrecy. What we found throughout history is that any good algorithm or any good method of encrypting is unbreakable, but not exactly forever. One of the longer-standing encryption algorithms that we know about is the Great Cipher of the 1600s, used by Louis XIII and Louis XIV. Although mathematicians tried to crack it for centuries, it was not until the 1890s that this was encrypted. This particular encryption was actually a substitution at the syllable level, which had complex substitutions including a delete-syllable phrase and an advance-syllable phrase. So it was leaps and bounds above what had been used in the 1600s but could easily be cracked now in the 1890s and beyond that. This was the source of the story of the "Iron Mask."

Up until now we've been focusing more or less on the issue of stego, in that if something encrypted is only known by two people, the issue is that no one's going to get a hold of that message more so than people are going to be able to read it. In stego, you would have to stumble upon an encrypted message to actually know that something was being transferred. That all changed with Marconi's amazing invention—the radio—in 1894. Use of the radio created the frequent communication in the public domain. You no longer had to sit around and wait to see if you were going to stumble upon the messenger holding the message. All you had to do was eavesdrop and listen on the radio waves. It became easy to eavesdrop. Then the need for an unbreakable cipher was established all over again.

There are plenty of good stories out of World War I and World War II of some amazing encryption formulas. In World War I, the French had their top cryptographers. They broke a German code, which actually helped the Americans locate the German submarines. The Americans didn't want to appear too perfect, so they sacrificed some of their own ships during the war to avoid letting the Germans on to crack their code. But in World War I, the Allies actually did crack all the German codes. Again, World War II was really an encryption war in that both the Japanese and the German codes were broken by the Americans and the British. The Japanese codes led to the death of their defense minister, the equivalent of a Donald Rumsfeld. We cracked the Enigma by doing two things. The first is that we sank a German submarine, and we picked one up off the sunken submarine. The second way is that some people from Britain picked up on the fact that the weather report was always the first line in German encoded messages. By backtracking from the weather report, they were able to backtrack into the formula used by the Enigma.

Now we move into the days of the Internet. The predecessor of the Internet was the Advanced Research Projects Agency Network (ARPANET). In 1969, they gave birth to four connected sites. The goal was to enhance the Pentagon's—the government's—infrastructure, in order to make it indestructible in the case of nuclear war. In 1982, the Internet was born. By the end of the 1980s, non-academic and non-governmental people like myself were given access to that Internet.

Back in the early 1970s, Whitfield Diffie and Martin Hellman actually imagined e-mail and Internet commerce. They imagined two strangers meeting over the Internet, computers connected by phone lines, wanting to exchange messages and wanting to conduct business with each other. This is back in 1973. This is probably 20 years before the mainstream society caught on. Their view was that everyone should have the right to encryption and privacy. Diffie met Hellman through IBM, drove across the country to Stanford to convince this guy to spend some time with him. Well, they hit it off and realized that they might be the only couple of people in 1973 looking to talk about encryption. They thought about key exchange. They thought about how two strangers could meet over the Internet and actually verify who each other was. They viewed it as a Catch-22. I can send Shawn a message over the Internet and encrypt it with a key, so I'm sending him an encrypted message. Now I have a key I have to send him and so I have to encrypt that key with a different key, and then I have to encrypt the second key with a third key and so on and so forth. Then they came across something really revolutionary, which was the idea of a lock box. I write a message. I put it in a box. I stick a padlock on it. I send it to Shawn. He sticks his own padlock on it. He sends it back to me, I remove my padlock, I send it to Shawn and he takes off his padlock. Now, for the first time, two strangers are able to send and receive a private message back and forth without ever exchanging information on each other. That gave them the insight that maybe it's possible, without exchanging private information—my lock combination—that I can actually go through, confer and create something which is kind of a "public key," which is what we now call it. They then picked up on the idea of modular arithmetic, which is base arithmetic. The basis of all the encryption formulas was modular arithmetic, viewed as a one-way function.

Charts 1 and 2 show an example of Brian and Shawn doing a Diffie-Hellman-Merkle key exchange. We both choose a secret number. Although I'm trying not to look at Shawn's, I see he has a "3" right here. We then enter it into a function, $Y^x \pmod P$, which is a one-way function. As you see in Step 4, we've actually come across an example where we've exchanged alpha, beta, Y and P in the public domain. We've kept A and B private but we've both come up with the same answer of "8." From their perspective, this is revolutionary, because we now have a chance to use the same mathematical function with commonly agreed-upon inputs and come across with the same output using the private information.

We move on to the contributions made by Ronald Rivest, Idi Shamir and Leonard Adelman—the "RSA" team—an MIT team with computer science and math expertise. They took over a year to determine this asymmetric function. "Asymmetric" means one-way function. This is mostly a one-way function, which takes care of both having a private key and a public key. This RSA algorithm is the basis of the PGP algorithm, which is the basis of the SSL, which is now used when you have HyperText Transfer Protocol Secure (HTTPS)

We'll take a couple of steps through the RSA algorithm. You determine "N" as the product of two very large prime numbers, "p" and "q". "N" ends up being the basis

of your public key, and the factors "p" and "q" end up being the basis of your private key. Then you end up having seven steps to encryption.

Here's an example of how this works: I choose two prime numbers, "p" equals 17 and "q" equals 11. I multiply them together to get my "N" as 187. I choose another number "e" that equals 7. I publish "N" and "e" in the directory. Shawn wants to send me the letter "X." He sends me the letter "X" and we convert "X" into its ASCII equivalent, which is its binary equivalent, as "1011000." Does everyone know what "ASCII" is? It's the bits and bytes, the basis of computer communication. Shawn was telling me that ASCII isn't used as much any more; there's something called Unicode.

MR. MOYER: Yes. ASCII is a bit U.S.-centric and Latin-language-centric. In order to accommodate Khanji and some of the other character sets, you have Unicode transformation formats (UTF), which are attempts to implement a baseline that you can use for any language

MR. SEPTON: So we convert "X" to its ASCII equivalent, 1011000. We take that and convert it to the base 10 number by using the powers of two and then it turns out "X" equals 88. With the full message, you convert an entire screen of text into its ASCII equivalent, then convert to base 10, and then you have your message. Shawn determines "C" based on the formula "M^e(mod N)".You end up having a number "C" that equals 11. Shawn is going to send me an "11." I'm going to determine my private key, which is based on my "p" and my "q" and my "e", which is going to be "d", equal to 23. There's some more complexity here behind getting my private key, which equals 23, but you can work through the math and come up with this. Now what you have in Step 7 is my decryption part, where I get "X," which is the message from Shawn. This is the mathematics behind all the main encryption formulas now. Shawn has sent me the letter "X," not knowing my "p" and "q,", but knowing my "e" and my "N." We've been able to exchange information by keeping something private and something public.

Phil Zimmerman created the PGP algorithm. Phil Zimmerman is a radical. He was actually a fugitive from the law for a couple of years for exporting some encryption algorithms out of the United States. He was a liberal and a radical as much as a brilliant mathematician. He thought that encryption should protect everyone, not just the intelligentsia, not just the mathematicians and the computer science people of the world. His idea was that it should be something easy to use to protect anyone with an idea or with a thought. After publishing his PGP program, he actually received notes and e-mails from people all over the world saying, "Had it not been for this encryption, I would have been 'fill in the blank'." He thinks that the encryption algorithm has protected the rise of the democratic ideals in a number of the communist countries, as well as other parts of the world.

What we saw is that the time and the processor requirements to use the RSA algorithm were complex, way beyond the means of the computers of the day. We

looked at that simple example to send an "X." If you have an e-mail that's several hundred words, that could have taken minutes at the computers in those days to encrypt. He would say to use the RSA to encrypt an International Data Encryption Algorithm (IDEA) key, and then use this simpler algorithm to encrypt the message. You have a complex encryption for a key and a simple, quick encryption for a message. He wrapped that all up in a piece of software called PGP that can generate keys through simple mouse movements.

To sum it up, you can protect the contents of a message by encrypting with your public key and then you decrypt it with your private key. Or I can reverse it. I can authenticate my ownership, or authenticate authorship, by encrypting with my private key and then everyone can decrypt with my public key. Or you can do it both ways, where you protect the contents and you verify the authorship. The PGP software is something that's easy to use and no real knowledge of encryption is required.

Now we have SSL, which was developed by Netscape and is now supported by Navigator, Internet Explorer and very likely others. It uses the public key to encrypt a session key and that session key is then used to encode the transmissions between a client browser and a server. Then you have a secure connection between your client and your server for encryption, authentication and message integrity. Of course "https" indicates SSL is in use, which indicates that PGP algorithms are in use. This is based on the RSA algorithm, which is based on the Diffie-Hellman-Merkle idea of some type of private and public key exchange.

Then there's the concept of bits of encryption. I know that I run 120 bits of encryption. That's the length of the session key for the encrypted transactions. The larger the session key, the harder this is to crack. We commonly talk about 40-bit, 56-bit, etc. These are based on powers of 2, so 128-bit encryption is trillions of times stronger than 40-bit. It's $2^{128}/2^{40}$ powers more powerful in terms of encryption. S-HTTP just shows that secure HTTP is used. That's not used as much.

You've all heard about the eligibility encryption export requirements. You cannot export encryption algorithms to foreign countries, because they're viewed as a weapon. This is what got Phil Zimmerman in trouble. The Clinton administration relaxed some of these standards. Now it's more along the lines of the "axis of evil," such as Cuba, Iran, Sudan etc. Chart 3 shows a sample of export restrictions from Oracle. Shawn is now going to talk about vulnerabilities and exposures.

MR. MOYER: Once again we're going to start with the bad stuff and get to the good stuff. We'll begin with fear, uncertainty and doubt and go from there. In terms of vulnerabilities and exposures, one of the prevailing key concepts in information security is that there are three facets of information that we want to protect. In our role as information security officers at a company, we want to protect the confidentiality of the data. If something is intended to be private communication, we want to make sure it stays that way. We want to protect the integrity of the

data. That often means things such as the Excel rounding bug, which may be something you have run afoul of. That affects the integrity of your data because you put a number in expecting a certain result and other results come out instead. In banking transactions, you want to know that if you pay a bill online that it actually gets there. We want to protect the availability of the data. When you click on the company's Web site or a link on the company's Web site, you want to know that you can get there and that it's going to stay up and remain available. That's something that a lot of IT people have a problem doing anyway, outside of security.

The inverse of those are the disclosure of the data, the alteration of the data and the destruction of the data.

The first type of attack or common exposure of vulnerability is something that probably everybody in this room has run into at one point or another These are simply viruses or "malware," a contraction of "malicious" and "software." Malware, a more general category than a virus, is any software that doesn't have the greater good as its primary intent. Eighty-five percent of people in the last computer crime survey—Brian will go over the Computer Security Institute's (CSI) computer crime survey in more detail, but it's the most commonly referred-to piece of data that we have—reported some kind of virus infection, whether it was one or two machines or whether it was a great number of systems. It's also generally the least costly per incident. There are situations, particularly with some of the newer types of viruses, where there can be a pretty significant impact, but we've gotten pretty good at this one. We've gotten pretty good at anti-virus software which can disinfect the system. Usually, at the worst, we have to reload it and reset it back up and we'll be okay.

There are different types of viruses and malware. Network-aware viruses are becoming more common. They are viruses that replicate across a network, due to the fact that everything is so interconnected at this point. A virus gets onto a system one way or the other, and then it uses that system to infect all the other systems that are nearby on the network. It spreads through the network. Usually we'll refer to those as an Internet "worm." That's something we've seen a lot more of in the past couple of years.

A "backdoor" or a "Trojan Horse" virus is something that poses as another piece of software. In the case of the Trojan Horse—you think of the Trojan war—the virus loads up on the machine posing as someone else or something else and then actually is a virus. A backdoor would be a virus that opens up a hole that some actual person might be able to use to get back into the system. A system that's infected with a backdoor virus might send an e-mail out from that system to the developer of the software saying "Hi, I'm here. Here's how you get back to the system." Boot-sector viruses and memory resident viruses (TSR), you don't see so much any more. In the days when we all frequently shared floppy disks, there were annoying things you'd get onto your floppy, then get onto your hard drive and then

on every floppy you'd put in it. Virus writers have figured out that networks are a lot simpler to write viruses for. You just replicate across a network and you can get to 300 machines that are all in the same building.

A polymorphic virus is a virus that copies itself in a slightly different way each time. It changes its signature a little bit. That becomes more difficult to detect because when you detect the virus, it changes its behavior in a little different way and so it comes out a different signature this time. It's very hard for the anti-virus vendors to write detection routines for this kind of virus. Multipartite is just a virus that replicates in a couple of different ways. Most of the viruses that have been coming up in the past year or two—the "I love you" virus, "bug bear," "love gate" and all the viruses that you hear about on the news—are generally polymorphic and multipartite. This means that they replicate through three or four different ways; they change their signatures; they replicate over a network; and they might also copy themselves onto your hard drive in different places. It is difficult to detect, isolate and control these viruses, but for the most part we're pretty good at containing them. Generally within a couple of days, the anti-virus vendors will have their signature. Usually when you get bitten is if you're one of the first people to get infected. If your company keeps current anti-virus software, keeps it maintained and updates it regularly, you're usually okay.

The bonus virus here is spyware. This is something that I consider a virus, but not everybody views it that way. In particular, anti-virus vendors don't because they don't detect for it a lot of times. Spyware is sort of like the Trojan horse. You install a piece of software like a media player or a little Web accelerator program, thinking that it's perfectly legitimate, but it acts like a virus and lodges itself inside your system, then sends out your data to a central repository that uses the data to sell you advertising or marketing materials or to send you junk e-mail. I consider that a virus and I'm hoping the anti-virus vendors will start to get on board with doing that.

Attacks by actual people as opposed to by software are less frequent than viruses, but the cost per incident is generally much higher, because you have an individual that's directing an attack at you and usually it has specifically your network in mind and your systems in mind. It's also a lot tougher to detect. Viruses tend to be very noisy. If you've ever had a machine infected by a virus, usually what you notice is that it's crashing. Internal and insider misuse is the most common. It's generally somebody that has some internal information about your network or about your company. It is not necessarily an employee; it might be somebody who is a friend of an employee.

One of the common attacks by individuals, and this is one that probably most of you may have done whether you intended to or not, is browsing. Think of browsing in a non-electronic form. You walk into your boss's office, you look down on his desk and there's your evaluation form. There's your score. You try not to look and you don't want to look, but maybe you just kind of edge over this way or that. The

browsing attack is similar in a network context. Frequently these are mistakes made by people where they've shared out a file and they've left it in a directory somewhere where it's publicly accessible, although they didn't intend for it to be. This is a tough one to protect against. What you have to do is be vigilant and audit your systems to make sure that you have everything locked down the way it's supposed to be. One of the problems with browsing is that it might be something that could lead to another type of attack. Maybe you find a password list or something like that. So those are tough to detect.

Social engineering is another one that's tough to manage. The idea of social engineering is that someone poses as maybe your IT "help desk," calls you up, says they were just performing a password check and want to know your password for the payroll system again because they're trying to reset it right now. You see a lot of this. The biggest thing to do there is just educate your people that you never give out a password over the phone; you never give out your log-in credentials over the phone or that sort of thing. There are also things like suiting up in a telephone workman's uniform with a hard hat saying, "Hey, I'm here to fix the phones," then plugging into the network, doing some work and then pulling back out later.

Denial of service (DoS) is the idea of disrupting the resources on a Web site. You see this a lot with high-profile Internet sites that people want to target, like MSN, CNN, Yahoo or eBay. You throw so much traffic at the site that the site kind of falls over. If you've been doing a lot of e-commerce, this can be a big problem for you because you're relying on the Internet as your vessel for your business.

The idea of sniffing and Man-in-the-Middle (MITM) attacks is to get between two points of communication. This goes back to what Brian was talking about with encryption and the radio. If you can intercept the communication over the Internet and you can get in between two people who are communicating with each other, you can spoof and pose as one of the two people in that conversation. Or maybe you just intercept that conversation and maybe you'll get some privileged information between those two parties, something they wouldn't want disclosed. The best protection for that are things like PGP and Secure/Multipurpose Internet Mail Extensions (S/MIME). Encrypted e-mail is probably the simplest way to solve that problem.

Defacements are something that we're seeing a lot more of in the past three or four years as we start to rely on the Internet. The idea is of taking somebody's Web site and putting an Internet graffiti tag on that Web site that says "Bob was here" or whatever it might be. Application attacks are a new area. This is something that people just now are starting to become aware of. When you log into online banking, you pop your credit card information into the Web site and you log in to view your statement. You're running a program on that server. You're executing code on that Web site. In the early days, before you had web applications, you weren't doing that. You were just viewing static data. Sort of like an online business

card, the Web site just said, "Hi. This is our company home page." Now we're starting to publish all these programs out there to let people do these things online. The problem is that you're actually running a program on that server, which means that you have some permission, some access to that server, to execute code. If you can get that code to feed you data outside of the parameters of what it should, you might be able to take over that server or you might be able to use somebody else's information. There are a number of things you could do.

What are the solutions and countermeasures? This is the good part, hopefully. I emphasize that security is a process and not a product. What that means is technology is not going to save you. Firewalls are not going to save you. Antiviruses aren't going to save you. What's going to save you is being vigilant, building this into your day-to-day business process, making everyone aware of it and making it a part of everybody's job. The way that you solve it is with a process, but technologies do help facilitate it. Step one of the process is to define your risk. You have to figure out what systems are critical. You have to figure out what it's going to cost if this system is lost or that system is lost. You don't want to spend $250,000 securing a system that has $10,000 worth of data on it. You need to do a cost analysis. You need to define if there is a cost justification or a business case for this.

Step two of the process is policies. I just love writing policies; it's one of my favorite things to do. The idea is that you create the policies and define the road map. If you don't do that, then you don't have any kind of clear direction. You have to have those in place. They need to be vaguely specific. You don't want to write things that say, "For Windows 2000, we're going to do this." You want to say, "For all of our operating systems, we're going to have these requirements." You do that so there's some life time and so these things don't have to be rewritten. Ideally, you write them in such a way that they're going to cover you from the ground up for a long time.

Step three is rinse, lather and repeat. The idea of that is every time you have a new technology, you may have to come up with some new policies. Hopefully you were vaguely specific and you wrote them in such a way that there's already a spot where this new technology fits in your already-defined set of policies, but that may not be the case. You may need to reassess periodically. You also evaluate—and this is the whole point of the policy—the compliance with the policies. Once you have the policies in place, and they're signed by your highest-level executive you can get to sign these policies, you go back, reassess and make sure that you comply. You take the things that are out of compliance and you bring them into compliance.

There are some methodologies for doing this; they're all painfully boring. I'm not going to go into them in any great detail, but British Standard 7799 (BS7799) is an IT auditing standard and a financial auditing standard used by a lot of the Big 4 firms. ISO17799 is the same kind of thing. SAS70 is an Australian standard and you probably don't see that much. A lot of your consulting firms that do information

security consulting or business process consulting will have their own methodologies. One of those is ADDME which stands for "assess, design, deploy, manage and educate." You assess the risk, you design a solution to solve it, you deploy it, you continue to manage that deployment and educate your people. It's a cycle to go through to do this.

There are technologies that help. Again, it's the process that really matters, but these are all some tools that you should have in your organizations. These are some technologies you might want to go back and ask your IT people about. Let them know that if they don't have these things, they may be some things that they should look into.

There is log collection and analysis. The idea of this is that programmers love to write debugging. They love to write all this information that logs because when the program is running, they want to go back and try to fix the problem when it crashed. Well, you have to put the log somewhere so that you can go back and do those. If the system is hacked, the very first thing that happens is that the logs get wiped. The logs are done. You can't trust the logs on a hacked system. The first thing that you do is you set up all of your systems to log to a central point. You have a system that has the log of logs. It's just a big hard drive that stores all this data. You need to go back and you can analyze that data after the fact. You can have forensic information to provide to investigators.

I think we're all familiar with firewalls. In the early days the idea of a firewall was something to protect you from the Internet, because the Internet is a bad neighborhood. The firewall keeps the bad guys out of the neighborhood. Well, sometimes your own neighborhood, your own network, has some bad guys too. Or maybe there are just some things going on that don't need to go on around the rest of the network. The idea now is that not only do you firewall from the inside of the company to the outside world, but you might also firewall between partner companies, between departments or between floors of the building. That gives us a little more control of what can go on. That's the idea of firewalling and multiple security zones. There are different zones on your network where you can protect the different departments and different areas from each other.

Network intrusion detection is essentially a flight recorder or a black box. This is something you plug into your network, and you log all the traffic. You log everything that goes on and then you can respond or react to any kind of anomalies that are detected. In our field we're starting to do simulations, modeling and statistical analysis against traffic and then trying to make determinations, trying to make educated guesses that this might look like bad traffic. This might be something that you should investigate. Intrusion prevention, as opposed to intrusion detection, is the idea that we're going to alert on these things and then we're going to actually try to stop them. We're going to stop them short; we're going to cut this traffic off. It sounds really good. This product and this idea are selling like hot cakes right now. The problem is when it's wrong. If you have a false

positive, if this thing predicts something as being not legitimate that is legitimate, then you've just killed legitimate traffic. You've just cut off one of your customers from your Web site. That's something that, again, as we all depend on e-commerce and these things, we don't want to do.

Vulnerability scanning is basically a canned way of auto-hacking your site. With the vulnerability scanner, you run this canned set of attacks against your Web site. You throw all these things at it and try to simulate hacking. The idea is that you can predict some of these things and find them before the other guys do. Anti-virus, host intrusion detection system (IDS) and personal firewalls are the local system version of the same things. These are things you'd run on your local computer. Host IDS is the same thing, but we're doing intrusion detection on the PC itself. Personal firewalls are the same kind of thing. We're going to firewall our own machine off from everybody else. There's some software for that called ZoneAlarm and another piece of software called BlackICE. These are probably things you should have on your home computers if you don't already.

MR. SEPTON: You covered it all. Let's spend a little time talking about the 2002 CSI Computer Security Crime Report. We're going to talk about who responded to this, what some of the common breaches were and what some of the common protections are. We'll touch on worms and viruses again, and we'll give a case study from our own industry on the CSI.

The CSI conducts an annual survey, which, as Shawn mentioned, is one of the leading surveys of computer security crime. I think of computer security in terms of three questions. Who are you? Can I verify this? Do you have clearance? Once you answer all these questions, you still don't know what somebody is going to do with that clearance. I may be okay to go, but I might be mad that day and I'll cause some trouble. In terms of the respondents to the survey, as shown in Chart 4, we have a good cross-section of U.S. industry. Manufacturing, financial and high tech are certainly up there in terms of respondents. In Chart 5, we have some small companies, under 100 people, as well as some larger organizations, over 10,000 people. Of course, in Chart 6 we have a variety of gross incomes among those companies.

In the 2002 report, 90 percent of the companies reported some security breach. There are about 480 companies that responded to this survey, 90 percent of whom reported some breach. Among the 223 that reported losses, they had $455 million in losses—proprietary information loss was at $171 million and financial fraud loss was at $116 million. That $171 million value of proprietary information is what you might receive in a settlement or a court trial for actually stealing proprietary information. Only 34 percent of the breaches were actually reported to law enforcement. Of the respondents, 40 percent reported invasions from outside. Forty percent reported denial-of-service attacks, and 85 percent reported computer viruses.

The 2002 report shows that it went up for a couple of years, and then it's been going down. In 2003 I believe the trend is stabilizing, if I understand this correctly. We have many "Yes" responses, some "No" responses and people who don't know if they've had a breach.

MR. MOYER: It probably would be valid to say that most of those "No" responses are actually "Don't know" responses. I think that would be a pretty fair assessment.

MR. SEPTON: In terms of points of attack, obviously the Internet is growing. Remote dial-in is kind of fading from the Internet access world and so naturally the point of attack of the remote dial-in is decreasing. In terms of sources of attacks, we have disgruntled employees and independent hackers accounting for a huge portion of the attacks. "U.S. competitors" is on the chart, but what's most surprising to me is that foreign corporations and foreign governments are being reported as the sources of attacks in a number of cases. In terms of financial impact of crime in 2002, laptop theft loss was $12 million. In the 2003 report that just came out, these numbers are actually down. The financial impact of the crime is actually decreasing, although as I understand it, the number of occurrences is actually staying level. The containment measures are getting better.

MR. MOYER: We still have just as many incidents, and we still have just as many people being compromised, but we seem to be responding to it a little better. We seem to be mitigating the losses a little better. So we're starting to improve. We'll have to see what 2004 holds for us, but I think we're getting a little better at this.

MR. SEPTON: How can the government crack down on this? What gives the government the authority to crack down on computer crime? The Economic Espionage Act (EEA) of 1996, passed under the Clinton administration, gives the government the authority to crack down on computer crime. It gives the government the power to prosecute, and it gives the government the forum to prosecute. The mere fact that the government has this power alerts industries that the government takes this threat seriously and alerts would-be hackers that there is a potential federal criminal penalty. It actually may act as a deterrent. In terms of the security technologies used, there are a whole bunch of technologies out there. We've talked about access control, anti-virus software, firewalls, encryption and some other security measures. This is what people are actually using. I don't believe they're mutually exclusive. People are using multiple versions of security to tighten up that fish net.

Worms and viruses are the simple pieces of code that can be devastating. Among the respondents, the losses from the worms and viruses are increasing, with the average loss going from $75,000 in 1997 to over $280,000 in 2002. This is just the average cost per respondent. The actual loss estimation in the world economy is quite different. The world impact of the "I love you" virus was $8.7 million, and the "Melissa" virus loss was over $1 billion. These are some of the worldwide

impacts of these things.

Here's a short excerpt from the 2002 crime report from Prudential Insurance, a part of our own industry and a company with a number of actuaries. "In 2002, U.S. federal agents, working with the New York Electronic Crimes Task Force arrested Donald Matthew McNeese on charges of identity theft, credit card fraud and money laundering after he stole a computer database containing personnel records for as many as 60,000 employees of the Prudential Insurance Company and attempted to sell the data over the Internet." This affects us. This is not about other companies. This is not about technology companies or manufacturing companies. Arguably every company has human resource (HR) data on its employees, and many of us working here have tons of information on non-employees—firms that we either do business with or consult with. This can actually happen to us.

MR. MOYER: The SysAdmin, Audit, Network, Security (SANS) Institute is a lot like the SOA within the information security field. It's an organization that tries to promote awareness.

FROM THE FLOOR: I'm an independent consultant. I consult with physicians. I send e-mails and spreadsheets back and forth with a lot of information. What do I need to worry most about as far as things being compromised?

MR. MOYER: So you have one PC that you sort of take with you as you go to these other companies and you correspond with them over the Internet? The one thing I would suggest for home users, as I mentioned earlier, is personal firewalls, which protect a single system. This runs on a single PC and protects your PC against other machines. It restricts the kind of traffic that's allowed into and out of your machine. When somebody tries to connect to your machine—you'll see a lot of this low-level noise over the Internet which is intruders scanning around looking for systems that have vulnerabilities—an alarm pops up and says that somebody is trying to connect to you and asks you if this is okay. Then you can give a "yes" or a "no." Or you can turn all that stuff off and it will just not allow anything aside from the traffic you initiate.

Also, you'll want to talk to the companies you deal with, particularly if the material is health care and physicians because there are may be some HIPAA questions. As I understand, both HIPAA and GLBA in terms of U.S. regulations, the onus is on the data owner. The data owner is the physician. When the physicians are sending that to you, they should be fully cognizant of the fact that that data is not encrypted. They should be aware that they're sending things in an insecure fashion. On a HIPAA audit down the road, that might end up becoming an issue. Something that a lot of people run into is that each one of those companies might use a different technology to encrypt, so then you have to have on your system the ability to talk to every one of those networks.

MR. SEPTON: Obviously the password is a big component of that. When you are

sending attachments via e-mail, you want to make sure those attachments have good passwords on them. A simple eight-digit password can be cracked in a matter of minutes with some good software. The password should contain capitals, lower cases, a number and even a punctuation mark. The bigger the potential character set is, the larger number of potential passwords a brute force attack would have to go through.

You can use encryption on your e-mail. You can use PGP through your e-mail, which I believe Microsoft Outlook and some other higher end e-mail systems do support.

MR. MOYER: The one that I tend to advocate a lot, and I wish I saw more companies using it, is S/MIME. It's the thing that allows you to click an attachment in an e-mail and have it open up. If somebody sends you a movie, that's what you click on to open the movie up. S/MIME is an attempt to integrate encryption into e-mail in that kind of fashion. The thing that's nice about S/MIME is that it's an open standard. It's free and it's built into nearly every e-mail program. PGP is actually a private company, and you do have to purchase PGP software to use it. If you want to encrypt e-mail with PGP, you need to purchase a license for that.

FROM THE FLOOR: I have a very small consulting practice. When you send an e-mail, my opinion is that you should assume there will be a copy of that e-mail in somebody's computer forever. What I finally did with my clients is that I wrote my own algorithm, my own encryption program.

MR. SEPTON: There is a common misconception that I want to address. In Microsoft Word, if you put a password on a file, it's not encrypted. The file is not encrypted in any way, shape or form. The *password* is encrypted in the file, but it's encrypted very weakly. It's something that would take you a couple of hours to crack. There are actually a number of tools that would let you crack that. The other thing is that you can actually use another program besides Word, like a Hex editor or some program that will let you read the file directly without the Word layer. This is the problem when you put the security on the user instead of putting the security into the whole process. You can still read the document, it's just that there's a little field at the top that says "Password = XXXXX." The rest of the document is still there; there's no encryption. Actually PKZip is starting to be big. They're going to have the feature that when you pick a file and make it a password-protected ZIP file, it does encrypt it with RSA.

FROM THE FLOOR: You talked about false positives. I live in Japan. A lot of people in North America have their spam detectors set to reject e-mails from Asia. There are actuaries who do not get my messages because of their spam detectors.

MR. MOYER: False positives are a bane to my existence as much as anything. I always say that logs are my life, and false positives are my life. The problem is that unsolicited mail has become out of control, in the past six or eight months

especially. With the "dot-bomb," the dot-com fallout, a lot of these people that had Internet-based business models went into this spamming business so that they could stay afloat. It's just gotten out of control. I have users who can't read their e-mail because there's so much junk that they get.

We talked about spyware earlier. The anti-virus vendors unfortunately don't agree about the whole spyware concept. I think they are afraid of stepping on toes. Some of the spyware vendors are linked with a lot of big media companies like Viacom and Miramax. There's a community-developed program called Ad-aware that you can run on your machine and it acts like anti-virus. It scans your hard drive, looks for spyware and deletes it. Probably about 80 percent of the time you run it you'll find something that is on the edge of being spyware. You can find Ad-aware at www.lavasoft.com. That's a free software that you can download and run.

FROM THE FLOOR: Now we have wireless Internet. What are the problems there?

MR. MOYER: Wireless is a big one. This goes back to the idea that every new innovation creates a new path of risk. Nobody thought about it when they designed wireless networks. These engineers were coming up with all these great ideas and this great product. Wherever you are, you can get connected. You don't need wires. You don't need anything. The other day I needed a hardwired Ethernet card. I went to the computer store to find one that just plugs in and I couldn't find it. There were shelves and shelves of wireless cards because that's what everyone's starting to do. The problem with wireless access is that it was a weak crypto implementation. They did actually make an effort to put encryption into the wireless standard. I won't pretend to be a cryptographer and go into all of it; you can research it yourself. Look for "wire equivalent privacy (WEP) vulnerability" on Google.

There's a key space that's secure and there's a key space that's less secure. With any given set of crypto keys, there are a couple of keys that will be weak. They're not quite as good and they're susceptible to brute forcing. When you implement a crypto system, cryptographers know that you pull a couple of those weak keys. You just have to know about those. When they implemented wireless encryption, they forgot to pull those out. What that means is that every once in a great while, you'll see one fly across there as a weak key. If you can decrypt that, you can get the rest of the traffic and then you can decrypt all of it. Now the vendors have since gone back and fixed it. If you buy a new wireless product today, it doesn't have that problem. The problem is all of the existing people that already have it that haven't applied the patch. The other problem is that most of the access points, especially the ones geared toward home users that you plug in, don't require you to set any security on it. Again, security doesn't sell the product—"Now featuring security!" is not what makes people take it off the shelf. If you read your documentation on your wireless access points or your wireless cards, it will tell you how to secure it. You just have to go through that process.

MR. SEPTON: Shawn is running Unix up here on his laptop. He actually rigged up a Web site. He'll walk through how a hacker might find some low-hanging fruit here.

MR. MOYER: What this is meant to be more than anything is just a demonstration of the standard process that somebody might go through to hack a Web site. This is just an illustration of the technique that most people use in compromising a site, if it's an individual. There are a number of worms and things that use different methods.

This is a company's Web site. It says "authorized access only." Nobody is supposed to be going to this site. There are no passwords or anything on the site. There is not a lot of data on there. If we click "More Information," it just tells us there is nothing to see here and please move along. We click on the "More Information" link again and it says again that there's nothing there to see. That's all that's on this site, so you don't have much to go on in terms of what's going on.

One of the first things a hacker, or somebody performing a penetration test, might do is see what kinds of errors the Web site throws at you. A penetration test, by the way, would be an exercise where you actually pay somebody to do this for you as a kind of test to see what your exposure level is. In this case, there's actually a lot of information here. It tells us that that particular URL, the Internet address, which is just some random characters, was not found, but it gave us an error code. It also tells me that if we throw out some other errors, it might throw different error codes. We might be able to infer from a Web server that throws this error but not this error, that it's doing something a little differently or that it's interpreting the data a little bit differently. Again, developers love to write debugging information. They love to throw all this data out and they throw it to the user on a Web site. One of the first things that I tell people to do in securing Web sites is to set up a canned Web page error, so it gives you the same error every time. No matter what you throw at it, it just says that something is wrong, please contact their help desk and they're sorry for the inconvenience. It logs the actual debugging data to a log somewhere that the developer can go back and look at. Don't throw it out on the Web site, because somebody else can find that.

The other one we see here is the version number of the Web server we're running. One of the things I can do then is go out and look on some of the vulnerability databases on the Internet because you just gave me your version number of your Web server. I can go find some bugs that way. Maybe I can find a hole. A lot of times these advisories that point out the different security holes that are found go into some detail on how that bug works and how to exploit that bug. If you haven't patched your system because you're not following through on your security process, I might be able to go back and find the old bugs that I can run against your site.

Another area where people like to throw stuff around is in the HTML code.

Frequently what I'll do on a site is take a look at the code. A lot of times there won't be a whole lot of stuff there, but sometimes I might find something useful. In this case, I have a comment that the developer put at the top of the Web page to remind himself to back the site up. It says "Remember to add this site to the sites backed up in the /admin directory. Also need to mail a copy to Joel at 'such-and-such'.com." Now I have two things. I have an e-mail address that I might be able to use for a spoofing or for a MITM attack. Maybe I'll send an e-mail posing as this Joel address or maybe I'll send Joel something saying, "Hey, by the way, can I get that password again for that Web site so I can log into this?"

I also have another directory to look in. That /admin directory isn't linked anywhere on the Web site, but I can infer from that comment that it's probably there, so I go look at the /admin directory. One of the big things that people make a mistake about regarding Web sites is that they think because there's not a link to it, it's not there. It doesn't work that way. There was a case with a publicly traded company that released a report every quarter. Everybody's quarterly reports kind of fit the same format. Well, the Web developer was lazy. He wanted to go home early. What he did was he put the quarterly report file out there on the site, but he didn't link to it. He didn't put the link on the Web page. He had something written as to how he was going to go publish that link later on in the day on the date that they release the report. But he put the report out a day or two early. Well, a reporter with Reuters figured out because he looked at the old reports, that it said "report2303.html". The next one said "report2403.html". He could just pull it up. He went ahead and released the story to the wire with everything that was in the report and scooped it by about six or eight hours of the actual release. They're still in court on whether that was legitimate or not. Was that hacking or was that a failure on the company's part? It wasn't a password or anything; it just wasn't linked.

There's another mistake that a lot of developers and other people make. If there's not an index file, a main file in the directory, don't give me the listing of the directory. Don't tell me all the files that are there because then I might be able to infer something. I might be able to look for other things that I can view. In this case I have a /admin directory, and then I have mirrored sites. I have the names of some other Web sites that they've mirrored here, that they have copies of. From that I can look for other sites to go after. There are some links there that I could use. Then I have a directory called "Commands," so that's lovely. This is a contrived exercise to a certain extent, but this is actually pretty common. There are a lot of Web sites that will leave executable programs lying around on the Web site to be run. The Code Red bug exploited this. That's essentially all it did. There was an executable command left lying around in the Microsoft IAS Web server that you could run on a server to get it to copy to other servers and launch code. Anyway, we have our little set of commands here, and there are a couple of doodads that are kind of nice. There is a utility here that's a tool to pull down copies of other Web pages. It's probably used by the admin in that mirror site to pull down the copies of the other Web sites that they mirror. Like most folks who pop on to Web sites and

do this kind of stuff, I have a little repository set up which is my little archive of backdoor tools that I would want to pull down to the server. On there, I can launch a command and pull down a copy of data.

The fundamental idea of all web attacks is that, because you're running programs on your server, if you have any kind of active, dynamic content, whether it's generating pages to give press releases or something else, you're running code on a server. If I can get your code to fall over in some way or find something that you left lying around, I can push that code up and I can do other things with it if I want to.

Now I have my backdoor loaded onto your Web server and I've now made the backdoor executable, so it's downhill from here. I can go ahead and start this backdoor application. Now, if you have a firewall, I'd have to find a port that I could listen on that was open to the firewall. But there are also some utilities that will traverse firewalls. They'll go out and connect back in. The server actually connects out to the Internet to another host and then tunnels you back in through that. The Web servers don't need to go to the Internet. They don't need to get out. Don't let them out. It only lets certain ports in. Get after your IT people and tell them to go through the firewall policies and make sure that they're corrected.

At this point, I'm going to watch a show. I can use the other end of that same utility and I can go to the company site on the port that I opened from my backdoor. Now I'm connected and inside the Web server. At this point I'm typing on your Web server, so I have full control. I'm going to pull down my own Web page to put on your Web server and a nice little graphic as well. I'm going to be a nice Web defacer. I'm just going to move your default Web page over for you so that I don't actually delete it or wipe anything out. You can restore it later, whenever you realize what's happened. A lot of these people that do this are actually pretty good about that. They'll leave it backed up for you.

When a site gets compromised, the first thing that the hacker will sometimes do is go patch all your security holes for you because they don't want anybody else to get in this site. They don't want to share. So you hope it's a nice hacker. When you come into work in the morning, because I'm probably doing this in the middle of the night while you're sleeping soundly, you check your company Web site and my page is waiting for you.

Essentially all of these things work this way. It's basically getting the Web server to run some code, that we can then push a backdoor to the system to get onto the Web server and then do whatever it is we might want to do. If you run programs on your Web server and you do things in an insecure fashion, or you don't follow the process, or you're not careful and you leave something lying around or you don't go through and actively secure it, somebody will use that same ability that you put out there to run code to do something malicious.

Chart 1

Encryption: Diffie–Hellman–Merkle Example of Key Exchange

	Shawn	Brian
Step 1: Choose secret #	A = 3	B = 4
Step 2: Y^x (mod P) One way function	11^A (mod 9) $\alpha = 8$	7^B (mod 16) $\beta = 1$
Step 3: Exchange results	$\alpha = 8$ and $\beta = 1$	$\alpha = 8$ and $\beta = 1$

$11^3 = 1331$; 1331 / 9 = 147 R 8; therefore 11 A (mod 9) = 8

Chart 2

Encryption: Diffie–Hellman–Merkle Example of Key Exchange

	Shawn	Brian
Reverse	β^α (mod Y)	α^β (mod Y)
Variables	$\alpha = 8$, $\beta = 1$ Y = 11, P = 9	$\alpha = 8$, $\beta = 1$ Y = 9, P = 16
Step 4	1^8 (mod 11) = 8	8^1 (mod 9) = 8

Both parties end up with the same result!

communicate α, β, Y and P in public domain

keep A and B private

Chart 3

Encryption – SSL and S-HTTP

Limitations from Oracle.com

ELIGIBILITY EXPORT RESTRICTIONS

I am not a citizen, national or resident of, and am not under the control of, the government of: Cuba, Iran, Sudan, Iraq, Libya, North Korea, Syria, nor any other country to which the United States has prohibited export.
I will not download or otherwise export or re-export the Programs, directly or indirectly, to the above mentioned countries nor to citizens, nationals or residents of those countries.
I am not listed on the United States Department of Treasury lists of Specially Designated Nationals, Specially Designated Terrorists, and Specially Designated Narcotic Traffickers, nor am I listed on the United States Department of Commerce Table of Denial Orders.
I will not download or otherwise export or re-export the Programs, directly or indirectly, to persons on the above mentioned lists.
I will not use the Programs for, and will not allow the Programs to be used for, any purposes prohibited by United States law, including, without limitation, for the development, design, manufacture or production of nuclear, chemical or biological weapons of mass destruction.

Chart 4

CSI Crime Report – Respondents by Industry Sector

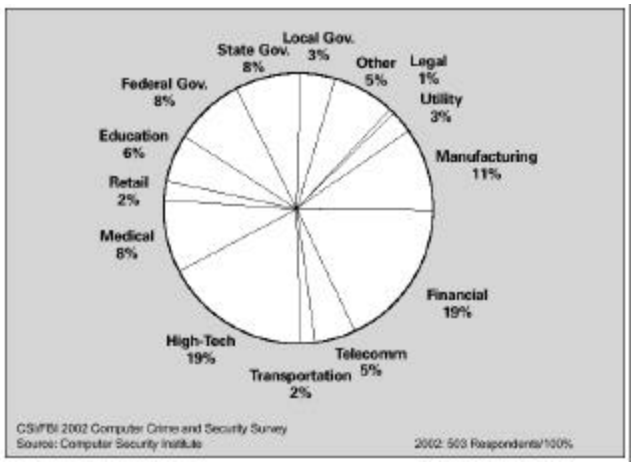

Chart 5

CSI Crime Report – Respondents by Number of EEs

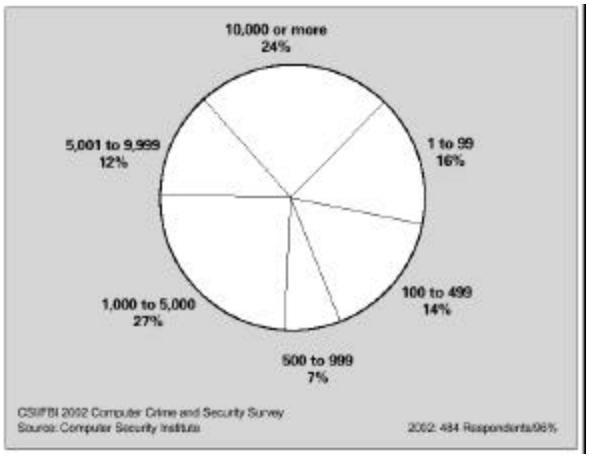

Chart 6

CSI Crime Report – Respondents by Gross Income