Keep It Simple

Making your enterprise more secure with less effort.

by Dr. Peter S. Tippett

TruSecure

Much of the conventional wisdom in corporate information security is seriously flawed. As Dr. Peter Tippett sees it, too many IT managers spend too much time and money on the wrong issues, while ignoring simple-to-exploit, simple-to-fix problems.

This executive brief, a collection of articles originally published in Information Security, unveils some of these common misperceptions, delivers a method for accurately assessing an enterprise's true risk, and suggests many small, easy measures that will improve your security immediately.

TABLE OF CONTENTS

Peter S. Tippett has led the computer security industry for more than 15 years as an expert on both the theoretical and practical aspects of security. A leader in risk-based strategies, Dr. Tippett has authored several studies and security cost models and is a frequent speaker on computer and Internet security, privacy and security ethics issues.

PETER TIPPETT, M.D., PH.D.
EXECUTIVE PUBLISHER OF *INFORMATION SECURITY*
VICE CHAIRMAN AND CHIEF TECHNOLOGIST, TRUSECURE CORPORATION
ptippett@trusecure.com

Keep It Simple

Virtually all of the recent high-profile Internet attacks were successful because network managers continue to overlook simple security problems.

As IT executives, we often get our priorities backwards and concentrate our efforts on resolving complex technical issues. As airplane pilots often say, “It almost always takes three mistakes to kill you.” Luckily, any interruption in a chain of errors can usually avert a crash. I believe the same holds true for network security.

Over the past year, the American Israeli Public Affairs Committee, Western Union and CD Universe were among the many companies that lost several hundred thousand customer credit cards to malicious hackers. Each of these attacks was successful because at least three simple security mistakes were exploited. First, all sites had unnecessary services running on their Web servers. Second, all failed to apply security patches (at least during the three months prior to the incident). And third, all kept sensitive information either in a database or a “temporary” file on their Internet-exposed Web servers or shared drives. Having analyzed hundreds of similar e-commerce sites, I would bet that each also had a handful of other simple-to-exploit, simple-to-fix problems. There’s no denying that anything connected to the Internet is constantly being pinged, scanned or probed. In recent weeks, about two dozen Web sites have been defaced a day, split equally between NT and Unix servers. The amazing thing is that the great majority of these exploits were based on two common, easy-to-prevent security vulnerabilities:

RDS vulnerability

More than 80 percent of the successful attacks on NT-based Web servers exploited a vulnerability in the RDS, a Microsoft Data Access Component that allows for particular database access methods. The service is installed by default on NT-based IIS Web servers and is not commonly used at most Web sites. The RDS vulnerability is more than two years old and has had good patches available since July 1999. It became the top attack vector for NT servers in October 1999, when 44 percent of Web sites were still vulnerable. Despite the publicity and the availability of a patch, more than a one-quarter of NT servers are still vulnerable today. That’s amazing when you consider how easy it is to fix the problem: delete a file, turn off the RDS service, or patch the RDS service.

Wu-ftp

More than 70 percent of the successful attacks on Unix-based Web servers exploit wu-ftp, one of the popular FTP daemons installed in many Unix boxes. In almost every case, the FTP service can be turned off or patched (the preferred solution is doing both). Honestly, there’s no real need for running FTP at most sites.

The easy solutions to both the RDS and wu-ftp vulnerabilities underscore the fact that three basic things will keep most malicious hackers out of your networks: (1) Turn off all unneeded services on boxes exposed to the Internet; (2) patch Internet-exposed computers often (every three months is a good start); and (3) never use Web servers for anything but their intended purpose—no DNSes, databases, e-mail or other functions.

Malicious Code, Too

Lest you think the simplicity rule only applies to hacking, let’s look at malicious code issues.

The Melissa virus surprised us all. Remember that it was a .doc Macro virus that caused the local mail client to mail it to the top 50 people in the local address book. Talk about pervasive: 20 percent of all North American companies were touched by Melissa, and 15 percent suffered a “disaster.” Likewise, the LoveLetter worm, which e-mailed with a visual basic script (VBS) attachment that used the desktop Windows scripting facility to mail itself to others, was experienced by 61 percent of North American companies, causing a disaster at about 42 percent.

We learned a lot from these viruses, but paid little attention to resolving the core problem. If we had, then Happy 99—now the most persistent re-mailer virus—wouldn’t still be plugging along over the past year, slowly causing as much or more damage than Melissa.

In the 2000 TruSecure Virus Prevalence Survey, **less than half of responding companies said they use perimeter antivirus products**, and **less than 10 percent use generic virus protection** (like file filtering) at the gateway. Guess what? Filter out e-mail attachments, including .exe, .scr, .pif and .vbs, and you'll have no problem from these "surprise" viruses—even if you haven't updated your AV definitions in months. In rare cases, users have a legitimate business need for receiving such attachments; but in most cases, they do not. Therefore, filtering them all out at the gateway prevents big problems. Users who actually need these file types can get the sender to zip them or get their e-mail administrator to manually forward them.

This stuff is drop-dead easy, requires almost no maintenance and, best of all, it works. Even better, it doesn't require buying any fancy hardware, installing or maintaining PKI, providing users with tokens, monitoring IDSes, enforcing complex password policies or doing anything else of marginal added value.

Wait: There's More

Simplicity also works in preventing internal attacks. The most common inside attack is one person using another person's "already-logged-in" machine to gain higher access privileges. Is "hacking" involved? No. Would hardening the database server help? No. What about IDS? *Fagettaboutit*.

What is the easiest, strongest way to stop this kind of inside attack? Use the screen-saver lockout provided with every desktop OS. Ask users to use any password at all. Does it need to be a secure screen saver? No. Does it need to be an eight-character, mixed alphanumeric password? Hardly. A four-character PIN will do nicely. Ask users to make it something they can remember and to change it yearly—if at all. You're not trying to prevent a brute-force or tool-oriented attack; you're trying to prevent a random person in the office from using a computer that's already logged in to the network before they get physically caught. While the Windows 95/98 screen saver is easily defeated by rebooting, this would require logging into the network again, which would defeat the average inside attacker.

Collectively, the five simple preventive measures below will reduce risk in an average organization by 10-fold or more. None of these measures is particularly invasive, expensive or high maintenance. Together, they are stronger than any new technology, firewall, architecture, IDS or almost any other defense we can throw up on the digital perimeter.

Do the simple stuff. Then do it again. And again. When you're satisfied that the simple controls are in place and working, then test them. Once you have proved they are working, then find a way to keep up with the changes in these simple approaches.

After the easy stuff is finished, working and still working after a month or two, then, and only then, should you think about additional security measures.

FIVE EASY PIECES

1. Turn off unneeded services in boxes attached to the Internet.
2. Never use a Web server for anything else.
3. Regularly apply security patches to critical machines.
4. Block all executable attachments at the gateway.
5. Use screen saver lockouts.

The Great AV Myth

It's one of the most widely accepted beliefs in computer security: the more often you update your virus signatures, the better you're protected against new malware. Unfortunately, like many other myths in security, it's a perception that, in practice, is completely false.

Here's a familiar scenario to illustrate the point. Let's say that once a day, every day, you update the virus signatures on your mail server, two file servers and your own desktop. Once a week, every week, you also pester your desktop end users to make sure their AV is updated. (You've heard of folks who manage to get automated updates working, but you're not one of them.)

Your assumption—a common one—is that this routine schedule of updates will protect you and your user base from getting taken down by a new virus or worm. Is this an accurate assumption? No. To be sure, if you update your mail server antivirus daily instead of weekly, you'll improve your company's defenses against known viruses. But the amount of improvement is so small—perhaps 2 or 3 percent—that it will have little or no practical effect on your ability to defend against serious malware attacks.

For example, no matter how often you updated your AV signatures, you were probably vulnerable to most of the mass mailing worms we've seen this year. That's because a large proportion of fast, mass mailing-type worms penetrate more than half of the sites they will ever infect *before* anyone has a chance to update their signatures. Occasionally, such a worm reaches its circulation peak even before the AV vendors have a chance to develop and release a signature update to defend against it.

A related assumption about virus prevention is that AV products are your only defense against malware attacks. This is also false. For instance, if you use Microsoft Word, the application's own Macro virus protection settings will protect you. Though older versions of Word shipped with this protection disabled, in new versions it's enabled by default.

O.K., so what happens when a Macro virus targeting an older version of Word isn't caught through signature scanning? Another AV technology—heuristics—often comes to the rescue. Most people assume that either their AV software doesn't include heuristics, or that it's not enabled. But in virtually all of the AV products on the market today, heuristics is working all of the time. And since there's a limited number of ways to create a virus in VBA (the Word Macro language), heuristics is very successful at capturing most new Macro variants.

Beyond these defenses, one of the best ways to close the "new virus gap" is to fine-tune your e-mail gateway to filter out e-mail attachments such as .vbs, .exe, and .pif.[1] According to ICSA Labs' Annual Virus Prevalence Survey, only 10 percent of corporations perform active attachment filtering at or near the mail gateway. Filtering for even some of these attachments would dramatically decrease your company's risk of infection.

The lesson here is that the best way to fight malware is to take an overlapping, "synergistic" approach. Updating your AV definitions is important, but doing so without adopting other defensive strategies won't decrease your overall risk.

[1] For a complete guideline to corporate virus controls, including a complete list of attachments to drop, see the TruSecure Anti-Virus Policy Guide at www.trusecure.com/knowledge/whitepapers/

Calculating Risk

When interviewing me for security-related stories, reporters frequently ask me to describe the primary goal of information security in terms everyone can understand. Here's what I say: Infosecurity is about mitigating risk.

There are many ways to define and evaluate risk, and many subtle and substantial differences in the application of risk-related terms. The most effective way I've found to define risk is with this simple equation:

Risk = Threat x Vulnerability x Cost

This equation is fundamental to all that we do in information security. But before we discuss the equation itself, let's take a look at these terms individually.

Threat is the frequency of potentially adverse events.

Since threat (by this definition) is always a frequency, it's always potentially measurable. And since the events are only *potentially* adverse, threat *per se* is not necessarily dangerous or detrimental.

Here are some examples. The threat rate of southern California earthquakes greater than 4 on the Richter Scale is 21 per year. The threat rate of hurricanes hitting Florida is 1.4 per year. The threat rate of insiders who use somebody else's logged-in PC to inappropriately access restricted information is approximately four per 1,000 users per day. The threat rate of virus encounters by a 1,000-PC organization is 88 per day. The threat rate of "attack-related scans" against a single IP address is seven per day. And so on.

Threat rates can be categorized into "global threat rates" and "local threat rates." A local organization's geography, status, political stance or any other factor may expose it to more or less threat than that of the global rate. The key to thinking about this is to determine (or at least estimate) the rate of whatever threats face your organization.

Of course, many threat rates change constantly, particularly those driven by humans. The rate of "attack-related scans" is up more than 20-fold in the past year, while the rate of virus encounters has nearly doubled in each of the past five years.

Vulnerability is the likelihood of success of a particular threat category against a particular organization.

Notice that if this were the likelihood of success of a particular attack (e.g. the Ping of Death) against a particular machine, the likelihood would be either 0 or 1 (0 percent or 100 percent likely to succeed against that machine). But since we are concerned about vulnerability at an organizational level (with, say, 1,000 PCs and 50 servers configured and architected in a particular way) to an entire class of threat, binary terms don't work. Instead, vulnerability has to be quantified in terms of a probability of success, expressed as a percent likelihood.

The likelihood of success is not easy to measure, but a related term, "vulnerability prevalence," is. Vulnerability prevalence is simply the number of machines of a particular type (say NT-based Web servers running IIS that are exposed to the Internet) that exhibit the particular vulnerability as currently installed and operating in their current environment.

Many factors work together to make some, but not all, machines vulnerable in their current environment—even if the software, hardware and data is identical across machines. Router rules, firewall configuration, proxy settings, NAT, location on a subnet, OS type, co-existence of other running processes, existence of data of certain types, existence of sample code or files, secondary connections of certain types—these factors and many others change the likelihood of success of a particular threat.

Event Cost is the total cost of the impact of a particular threat experienced by a vulnerable target.

Cost is measured in both "hard" and "soft" dollars. Hard dollar costs are measured in terms of "real" damages to hardware or software, as well as quantifiable IT staff time and resources spent repairing these damages. Semi-hard dollars might include such things as lost business or transaction time during a period of downtime. Soft costs include such things as lost end-user productivity, public relations damage control, a decrease in user or public confidence or lost business opportunities.

For the two weeks before and after the Melissa virus catastrophe in 1999, TruSecure did a study where the person most responsible for virus security in 300 organizations was asked to assess the cost of his or her company's "most recent virus event." Nearly one in five companies in the survey said their most recent virus event was Melissa. Of these companies, 79 percent experienced a "disaster" from it. The average "disaster" company had 1,120 employees and averaged 196 infected PCs and 8.7 infected servers (including e-mail, e-commerce and other servers) per site, which were down for an average of just over two days. Yet the average technician whose company experienced a disaster related to Melissa said the organizational cost was only $1,700. The actual total costs were probably more than seven-fold higher. Why? Because almost none of the technicians surveyed added in second-order hard costs or semi-soft or soft costs.

Risk. It's not threat, vulnerability or cost alone that really matters, but risk.

As you can see from the risk equation, for there to be any risk there must be at least some threat *and* vulnerability *and* cost. The concept we all learned in sixth grade—that anything multiplied by zero is zero—means that if any one of the three components of risk is zero, then risk is also zero.

This concept comes in very handy when evaluating a vendor's or the media's suggestion that "XYZ risk" *must* be addressed. If you can determine that XYZ risk poses no threat to your organization…or if you determine that your organization is not vulnerable to it…or that if it is vulnerable to it, the cost of fixing or repairing the problem is zero—you automatically know that XYZ risk doesn't pose a risk to your organization.

In most instances, you won't be able to say for sure that any of the three risk factors is zero. Instead, you'll need to estimate or measure each component of risk. For instance, let's say you want to determine if your intranet Web server is vulnerable to the "gichagoombi" attack, and if so, to determine the level of the threat. To do this, you need to evaluate the *threat rate* in other spheres (like the Internet), and determine how that translates to a likely threat rate in *your intranet*. What tools, knowledge and access are required to make it a threat? What human motivation is necessary? Who in your company has all the ingredients (tools, knowledge, access, motivation) to exploit the vulnerability? By drilling down into each component, you'll very often conclude that there's no risk—or at least no imminent risk—because at least one component of risk is zero or near zero. After this analysis, you'll often conclude that there are many other things that are far more risky and therefore should be addressed first.

Vulnerability is often the first thing to address, since that's where you typically have the most control. There are always many, many places where you can at least partially reduce vulnerability, and do so easily and inexpensively. We call these partial solutions "synergistic controls." They are overlooked by almost everyone, but are exceedingly useful, especially when used together with other synergistic controls.

The Crypto Myth

What's our biggest obstacle in implementing good corporate security? Lack of funding? Lack of management support? Network configuration complexity? Lack of end user awareness? Keeping up with patches and updates? The answer, I would argue, is none of these, but rather misdirected focus.

We devote much more time and effort to putting out fires than to making our systems flame-retardant. Worse, when we do pay attention to prevention, we often misdirect our actions toward what the security "experts," auditors, regulators, our peers and "common sense" deems important. Unfortunately, for most things related to security, what seems like the right thing to do isn't.

For more than 15 years we have been deluged with the idea that Internet encryption, SSL in particular, is *sine qua non*—an absolutely indispensable component of enterprise and e-commerce security. The argument goes like this: Because the Internet uses packet switching rather than circuit switching, our traffic is part of giant party lines—easily sniffed (eavesdropped, snooped, wiretapped) by almost anyone with a packet sniffer and a little ambition. Because most of us in the infosecurity community regard Internet encryption as a given, we, in turn, pester partners, end users and anyone else who will listen to make sure their browsers are in secure mode whenever transmitting sensitive information (address, credit card number, etc.).

On a more technical level, security geeks constantly remind us that the paltry 40-bit encryption in default browsers can easily be broken with an old desktop PC in a day. We should really use 56-, 64- or 128-bit encryption, they argue, because it would take a week of 1,000 computers (56 bit) or a century of all the computers on the planet (128 bit) to break.

Yes, data encryption is a fundamental concept in security, and I'd be a fool to say it's not important for many applications and in many environments. But all this brouhaha about Internet transaction encryption misses a much larger point: The risk of having your credit card number sniffed on the public 'Net is next to nothing. I'm not talking about sniffing on slow network segments, or on a corporate subnet—where the risk is real—but rather on the public Internet.

In "Calculating Risk", I discussed a basic equation: *Risk = Threat x Vulnerability x Cost*. Let's apply this equation to the concept of Internet encryption. The risk we are mitigating by encrypting our e-commerce transactions is the risk of someone sniffing our traffic somewhere between us and the destination site. To quantify this risk, we only need to measure or estimate the vulnerability (likelihood of success or *vulnerability* prevalence), *threat* (frequency of attempts or successes) and *cost* (of a successful security breach by this mechanism) of this alleged problem.

OK, so what is the vulnerability of Internet sniffing? I would argue that the likelihood of success of sniffing somewhere between your home or office and an e-commerce Web server is incredibly low, perhaps as low as 10^{6} (meaning the likelihood of success would be one in 1,000,000 sniffing attempts).

A few years ago, I hosted a TruSecure ISP Backbone Security (ISPsec) Consortium meeting, where we discussed a problem MCI and other ISPs were having trying to fulfill an FBI wiretap request. The court order wanted MCI to write to disk a week's worth of data from an OC3 Internet pipe for later analysis. After many months of complex technical work, using the fastest processors, tools and disk arrays obtainable, MCI was only able to sniff the *headers* from the wire.

Three years have passed, and Moore's Law tells us that processors are perhaps three times faster, and disk drives perhaps two times faster. However, bandwidth has also increased; today's OC192 pipes are more than 60 times faster than OC3. Translation: As difficult as sniffing was three years ago, it's 20 to 30 times more difficult today, even if you're a backbone ISP.

Of course, there are other contributing factors that further reduce the vulnerability, including the problem of identifying which fiber to sniff and the fragmentation of transmitted packets. The point is this: the vulnerability for sniffing public Internet traffic is low, whether it's encrypted or not.

Now, what about the threat rate? We read lots of news reports about this-or-that Web site losing thousands of credit card numbers to a database cracker, but have you ever once heard about a cracker obtaining such information by sniffing the public Internet? Neither have I. That's because, for credit cards at least, *it hasn't happened*.

Proponents of Internet encryption might cite this fact as a "crypto success story." The reason no one has sniffed credit card numbers on the public 'Net is because everyone's using encryption.

Baloney. In 2000, less than half of the credit card numbers traveling across the Internet were encrypted at all. For the other half, more than 70 percent of browsers in North America and Western Europe only support 40-bit encryption. Most B2B sites still use private (unencrypted) lines or 56-bit DES. All of this is to demonstrate that the threat is lower than low. In fact, it appears to be *zero*.

That brings us to cost. This one is quick. For consumers, the loss of credit card information is somewhere in the "minor hassle" category. If someone steals your credit card and charges four new radial tires on it, you're only liable for $50 under the worst of circumstances. (Most credit card issuers waive *all* liability if you contact them within 24 hours). A new card arrives within a day or two, and you're back in business. No hassles, no headaches…and no cost.

So, when we consider all these factors together, here's what our risk equation looks like: The *risk* of credit card fraud by sniffing the public Internet has a very low *vulnerability* multiplied by a *threat* rate near zero multiplied by a very small *cost*. When you extrapolate this out to the millions of people transmitting credit card numbers across the 'Net, the risk is darn near zero. In fact, I would argue that it's not even in the top 1,000 real risks worth worrying about. This hasn't always been the case, but as each year passes and bandwidth and traffic and processor speed all increase exponentially, the risk of such a breach is less and less, with or without the use of encryption.

This is what I mean about misdirected focus. No part of the credit card theft problem relates to Internet sniffing. No amount of transit encryption has any real value once you get outside the DMZ. The number one e-credit card problem has always been the insecurity (both physical and electronic) of servers and databases storing this information. But instead of addressing well-known and easily fixed server vulnerabilities, or setting up basic programs to make sure Internet-facing machines are regularly patched and updated, we spend money and resources on the most pervasive and least provable Internet security myth.

In the 14th Century, Philip VI of France ordered doctors at the University of Paris to explain why Europe was getting hit so hard by the bubonic plague. Their answer was that Saturn, Jupiter and Mars were in an unusual alignment in the 40th degree of Aquarius.

In 1600, years before Galileo was ostracized, Dominican friar Giordano Bruno was burned at the stake in Rome for insisting that the Earth traveled around the Sun.

Nuff said?

Stronger Passwords...Aren't

In "The Crypto Myth," my main point was this: The cost and effort to maintain an infrastructure that supports Internet encryption far outweighs any possible gain. Now I'll focus on another security "necessity" that, in reality, has a minimal impact on risk reduction: strong passwords.

Most of us are intimately familiar with the recipe for a "strong" password: it's seven or eight characters in length, uses mixed alphanumeric characters (or maybe even upper and lower case letters or Alt-key characters), and is changed every 60 days or so. The reason we're told to adopt such a password policy is to prevent crackers from easily guessing an end-user's password, which could be used to gain access to a corporate network.

Sounds simple enough, but unfortunately this type of password policy is a red herring. In real life, a "strong" password is really no more secure than a "weak" one.

Typically, a password file stores neither native nor "encrypted" passwords. Rather, passwords are usually hashed with SHA or MD5 and stored with corresponding user IDs. Hashes are truly one-way functions. In other words, you could hash the entire Bible and represent it as 8 bytes of gibberish. There's no way to use these 8 bytes of data to get the Bible back.

The reason we're told to use strong passwords boils down to this: Someone might steal the password file—or sniff the wire and capture the user ID/password hash pairs during logon—and run a password cracking tool on it. Such tools come in many sizes and shapes; the most popular include Crack, L0phtcrack and John the Ripper. Since it's impossible to decrypt a hash back to a password, these programs first guess a password—say, "helloworld." The program then hashes "helloworld" and compares the hash to one of the hashed entries in the password file. If it matches, then that password hash represents the password "helloworld." If the hash doesn't match, the program takes another guess.

Depending on the utility, a password cracker will try all the words in a dictionary, all the names in a phone book, the names of football teams and so on—and for good measure, throw in a few numbers and special characters to each of the words it guesses. (I've even heard of password crackers guessing words found only in the Klingon language). If any of the guessed words match any of the passwords in the password file, game over.

By using random alphanumeric characters in lengthy strings, strong passwords supposedly thwart these so-called dictionary attacks. But there are at least three problems with this assumption.

1. Strong password policies only work for very small groups of people.

   In real companies they fail miserably. Suppose you have the aforementioned strong password policy in your 1,000-user organization. On average, only about half of the users will actually use a password that satisfies your policy. Let's say your company constantly reminds your employees of the policy, and compliance increases to 80 percent. Maybe you use special software that won't allow users to have "bad" passwords. It's rare that such software can be deployed on all devices that use passwords for authentication, but for the sake of argument, let's say it gets you to 90 percent compliance.

   Great, right? Sorry. Even if 900 out of 1,000 employees use strong passwords, Crack can still easily guess 100 user/ID password pairs. Is 100 better than 500? No, because either way, the attacker can log in. When it comes to strong passwords, anything less than 100 percent compliance is necessarily weak. And as we all know, nothing is 100 percent when it comes to security.

2. With modern processing power, even strong passwords are no match for current password crackers.

   The combination of desktop Pentium III processors and good hash dictionaries and algorithms (to deal with numbers and special characters and capitalization issues) means that, even if 100 percent of these 1,000 users had passwords that meet the policy, Crack will still win. Why? Because after it finishes its dictionary attack, it can conduct a brute-force attack. While some user ID/password pairs may take days or weeks to crack, approximately 150, or 15 percent, can be brute forced in a few hours. It's only a matter of time.

3. Strong passwords are incredibly expensive.

   Organizations spend a lot of money trying to support strong passwords. The second or third highest cost to help desks is related to resetting forgotten passwords. The stronger the password, the harder it is to remember. The harder it is to remember, the more help desk calls. Many companies have full-time help desk employees dedicated to nothing more than password resets. These companies are also suffering real productivity loses from users who struggle for minutes or hours before calling the help desk. And then there's the cost of training users and promulgating the password policy in the first place. It all adds up.

What Should You Do?

So, we're left with an unwieldy password policy that, among other things, requires expensive training, expends lots of valuable help desk time and results in lost end user productivity. And at the end of the day, no one achieves 100 percent policy compliance anyway, and anything less than 100 percent is scarcely better than no policy at all. Not very logical, is it? A lot of time, effort and expense for little or no security gain.

What's the answer? One solution is to augment passwords with another form factor, such as biometrics, smart cards, security tokens or digital certificates. But each of these solutions is expensive to deploy and maintain, especially for large, distributed organizations with heterogeneous platforms.

Or, we could recognize that 95 percent of our users could use simple (but not basic) passwords—good enough to keep a person (*not* a password cracker) from guessing it within five attempts. I'm talking about four or five characters, no names, initials or teams, changed at least once a year. For all intents and purposes, this type of password is equivalent to our current strong passwords. The benefit of these passwords is that they're much easier and cheaper to maintain. Fewer calls to the help desk, fewer password resets, less of a productivity hit—all at no measurable security degradation. Under this scenario, we could reserve the super-strong passwords or tokens for the 5 percent of system administrators who really yield large span of control over many accounts or devices.

If you want to improve on that, you could make the password file mighty hard to steal. You could also introduce measures to mitigate sniffing, such as network segmentation, desktop automated inventory for sniffers and other malicious electronic tools. For the truly paranoid, you could encrypt all network traffic with IPSec on every desktop and server.

If the promised land is robust authentication and access control, you can't get there using passwords only, no matter how "strong" they are. Simply put, strong passwords aren't. If you want to cut costs and solve practical problems that impact the everyday operations of your organization, think clearly about the vulnerability, threat and cost of each risk, as well as the costs of the purported mitigation. Then find a way to make it cheaper with more of a security impact.

Defense-in-Breadth

How to reduce risk using "synergistic security."

Just about everyone involved in infosecurity has heard of "defense-in-depth," the practice of building multiple layers of security into a given system or network. Most security books, trade magazines, conferences and workshops trumpet defense-in-depth as a fundamental principle of security management and administration.

Defense-in-depth is, indeed, a key security concept. But I would contend that most of us think of "depth" in rather shallow ways. We don't do a very good job of implementing depth at an organization-wide level. Worse, we don't use the defense-in-depth concept to simultaneously simplify and improve security.

Binary vs. Synergistic Controls

First, a working definition. There are five different control types that can be applied to any given security threat (or attack) scenario: protect, detect, recover, deter and transfer. Some people define defense-in-depth as the ability to respond to each threat or attack with at least one control from each of these five categories.

For example, think about how a bank protects itself from a robbery. It uses vaults, armed guards and bulletproof glass dividers to protect against break-ins; alarm systems and security cameras to detect unauthorized entry; recovery plans and alternate facilities to help it recover in the event of a theft; laws and marketing to deter robbery attempts; and insurance to transfer the residual risk.

Such an approach guarantees security redundancy; should any one of the controls fail, others are there to back it up. If the bank security guard falls asleep, the alarm system will detect unauthorized activity. If the alarm system is disabled, the security cameras will record the break-in. And so on.

The problem with most approaches to digital defense-in-depth is that they assume that each control has "binary effectiveness"—that is, it works either all of the time or not at all. And, as we all know, perfect security is impossible. We all pay lip service to the idea that "no security is perfect," but most of us translate that into a belief that good security controls will still be in excess of 99 percent effective.

It's laudable to try to achieve this level of effectiveness with any one security control, but it's totally unrealistic. Trying to achieve even 90 percent effectiveness in some controls is incredibly costly, time-consuming and counterproductive.

A better (and broader) approach to defense-in-depth is one that I call "synergistic security." Like traditional conceptions of defense-in-depth, the success of synergistic security hinges on the redundancy of security controls. But unlike binary security controls, synergistic controls are not either "on" or "off." Each synergistic control is purposefully understood to be (significantly) less than 100 percent effective, making it more practical to maintain while also reducing cost, infringement, management and maintenance burdens.

Let's go back to the bank example. What is it that keeps banks from being robbed? And how effective are each of the controls? If you think about it, a security guard, a surveillance camera and a vault are independently only about 70 percent or 80 percent effective in preventing a robbery. For example, the vault in most banks is kept open during banking hours. For the vault to be 90 percent effective or better, it would have to be kept closed almost all the time. This, in turn, would force a bank manager to open and close it every time he needed to get some money or help a customer with a safe deposit box. Imagine how much time that would take; the bank would have to dedicate a manager to doing nothing else but opening and closing the vault!

My point is, any single control that's 99 percent effective would cripple the bank's business productivity. What keeps most banks from being robbed is that each of its controls works synergistically with the others to achieve an additive effect. The same holds true for enterprise infosecurity. Single controls with binary effectiveness often kill our ability to do business efficiently. We should actively look for a better way.

Bayes's Theorem

The statistical theory I use behind the concept of synergistic security is called Bayes's Theorem, which describes a "new" probability (control effectiveness) given a "prior" probability (see chart, opposite). If one control is 80 percent effective, then it fails one out of five times. Two controls, each 80 percent effective, together will fail one out of 25 times. Three 80 percent effective controls, operating together, will fail one out of 125 times. In other words, they will succeed with a likelihood of 99.2 percent.

Now, suppose that in addition to implementing good primary controls, we look for and implement a suite of complementary controls that are nowhere near as good—not robust, not fundamental, not whiz-bang, not even sold by any vendors, and perhaps not even thought of as relating to security at all. These controls need to be cheap, easy and non-infringing; effective enough (say, more than 60 percent) against an important category of risk; and able to operate independently versus other controls.

We should optimize our primary controls and then add depth with simple, manageable, low-infringement, synergistic controls.

For example, to protect an IIS server from external hacks, you could implement multiple complementary controls at different levels. At the perimeter, you could configure border routers and firewalls to default-deny traffic. On the IIS box itself, you could delete sample files, move or rename the command shell .exe and delete the scripts directory. On the policies and practices level, you could specify only local management of the server and insist on a quarterly tune-up cycle. And so on.

These are only a few examples for a very specific application. For any protected resource, it's easy to look around at the practices, policies, procedures, configurations, architectures and other protective mechanisms already in place at your organization and use Bayes's Theorem to determine where your real needs are.[1] At a bare minimum, for most categories of risk, you should have either two protective primary controls (defined as controls with greater than 90 percent effectiveness), or a primary and at least three synergistic controls. Failure of any one control in a scenario like this would still leave better than 99 percent effectiveness.

Efficacy of combined controls

		Efficacy of each control				
		50%	**60%**	**70%**	**80%**	**90%**
# of synergistic controls	**1**	50.0%	60.0%	70.0%	80.0%	90.0%
	2	75.0%	84.0%	91.0%	96.0%	99.0%
	3	87.5%	93.6%	97.3%	99.2%	99.9%
	4	93.8%	94.7%	99.2%	99.8%	100.0%
	5	96.9%	99.0%	99.8%	100.0%	100.0%

Mind you, I'm not advocating we abandon the excellent primary controls most of us already use: firewalls, IDSes, AV scanners, crypto, door locks, etc. But even in the best of all worlds, the effectiveness of these controls (used by themselves) is in the low 90 percent range. Our task should not be finding ways to increase the protection levels of these controls beyond their capabilities. Instead, we should optimize these primary controls and then add depth with simple, manageable, low-infringement, synergistic controls.

Allowing yourself to give some value to things which are only partially effective can be liberating. For every threat category, force yourself and your team to list at least a dozen synergistic controls. Then look at your list and narrow it not for controls that are the strongest, but instead for those with the least business impact on your organization. Be realistic about just how poor these kinds of controls can be. Put enough of them together to take the appropriate bite out of your risk-worry. Don't forget to include the things that may already be in place.

[1] For example, the "TruSecure Antivirus Policy Guide" offers an example of how to use four primary controls and more than 20 synergistic controls to protect against viruses and other malware.

To determine the total effectiveness of one or more synergistic controls, use the following equation (E = effectiveness of a single control). As the chart indicates, using multiple ineffective controls together results in effective control overall.

Bayes's Theorem: $E_{total} = 1 - ((1-E1)*(1-E2)*(1-E3)...)$

TruSecure

TruSecure® Corporation is a worldwide leader in security assurance solutions for Internet-connected organizations. Hundreds of leading companies rely on TruSecure to help them identify, correct, and continuously and pragmatically manage risks to mission-critical systems and information. TruSecure's cost-effective programs generate improved ROI on security investments, and provide the assurance that organizations can confidently and safely pursue their Internet-based initiatives.

TruSecure provides a comprehensive suite of services integrated by TruSecure's **Lifecycle Risk Management** model, which addresses all the key phases of an ongoing risk management program. Unlike audits or consulting projects, this lifecycle model intelligently integrates existing technology, internal staff, expert support and real-time intelligence into a coherent, continuous and prioritized enterprise security plan.

Security Assurance Services provide a comprehensive and essential baseline of effective security across the organization. Focused on proactive and preventative risk reduction, these services address hacking, viruses, insider threats, physical risks, downtime, and more.

Enhanced Services provide in-depth expert support for the security infrastructure, such as application security, documentation, security design and investigative response.

Managed Security Services enable organizations to fully outsource the complex and demanding tasks of managing, monitoring and maintaining critical security systems such as firewalls, intrusion detection systems, and email scanning.

Powered by Trusted Security Authority ICSA Labs

TruSecure brings its customers an unparalleled body of expertise through its ICSA Labs® division—the security industry's central authority for research, intelligence, and product certification for over a decade.

Supported by Industry Leader *Information Security* Magazine

Information Security magazine, published monthly by TruSecure Corporation, is the security industry's leading publication, with more than 49,000 subscribers. The magazine includes in-depth features, case studies, timely news coverage, authoritative commentary, and product reviews authored by recognized experts.

TruSecure also publishes NT Bugtraq, the de facto resource for intelligence, perspectives and solutions to Microsoft-related security issues.

For more information about how TruSecure Corporation can help secure your business, call **1-888-396-8348** or email **info@trusecure.com**.

TruSecure

TruSecure • 13650 Dulles Technology Drive, Suite 500 • Herndon, VA 20171-4602 • www.trusecure.com • 1-888-396-8348