[IN]SECURE

OPEN. INFORMATIVE. TO THE POINT. Issue 5 - January 2006

WEB APPLICATION FIREWALLS PRIMER

WRITING AN ENTERPRISE HANDHELD SECURITY POLICY

THREAT ANALYSIS USING LOG DATA

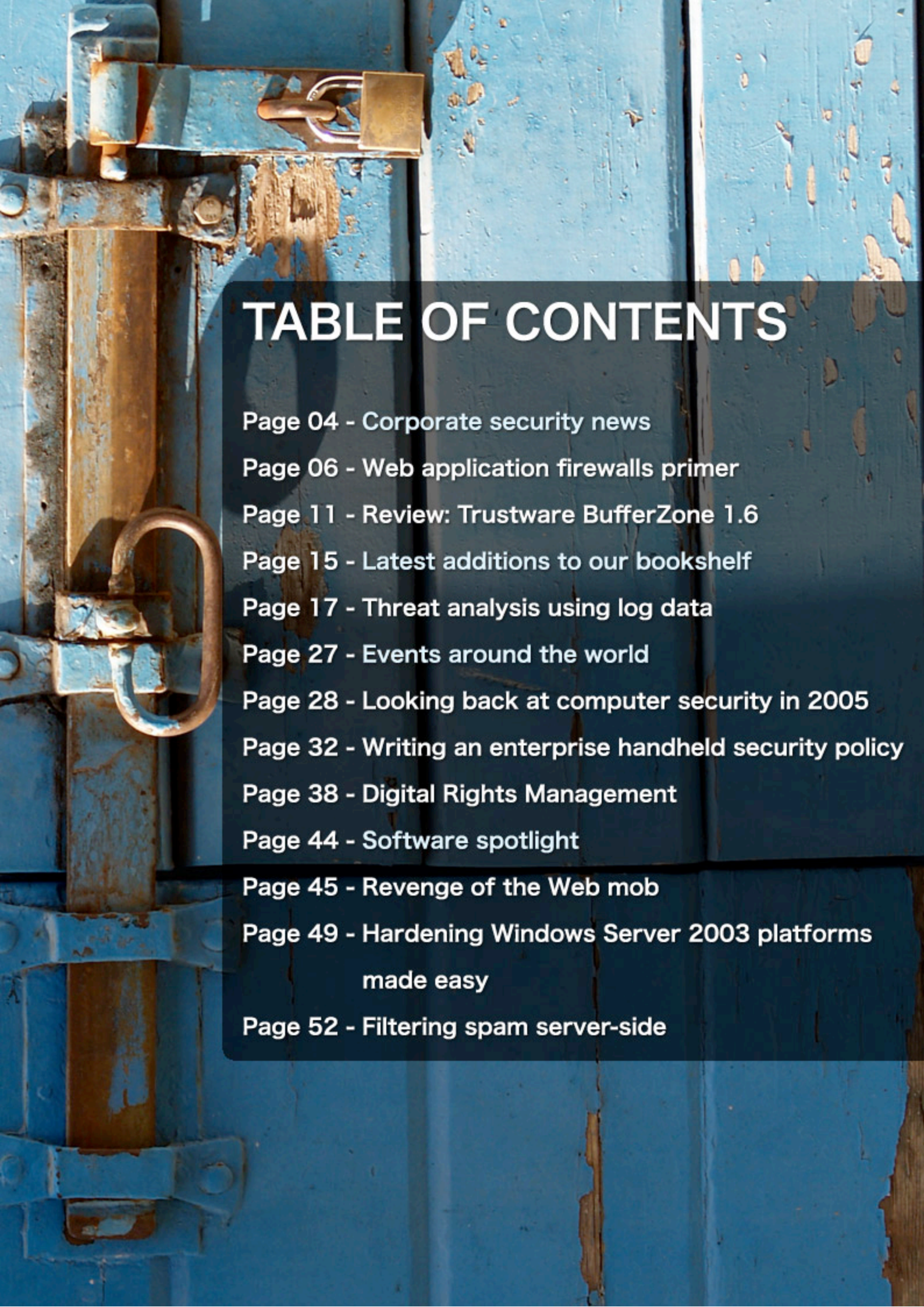


TABLE OF CONTENTS

With 2006 just starting off many are happy to have left a myriad of security problems behind in 2005. I hope you see more security awareness and experience a cleaner network environment this year.

This issue brings more technical articles and reflections on the past year. Many of you e-mailed us asking us to review software titles and hardware, so you'll be glad to know that we have a review in this issue.

Keep those comments coming and thanks for the support!

Mirko Zorz
Chief Editor

Visit the magazine website at www.insecuremag.com

(IN)SECURE Magazine contacts

Feedback and contributions: Mirko Zorz, Chief Editor - editor@insecuremag.com

Marketing: Berislav Kucan, Director of Marketing - marketing@insecuremag.com

Distribution

Web application firewalls primer

By Ivan Ristic

The popularity of web application firewalls is on the rise. These tools used to be reserved for a very small percentage of high profile deployments, but, with the number of less costly products appearing on the market and an open source option available for anyone to try out (ModSecurity - modsecurity.org), they are finally available to the majority.

In this article I will describe what web application firewalls do and give you a quick overview of their most useful features. After reading the article you should leave fairly familiar with the subject matter. You should also have enough information to determine whether or not web application firewalls have a place in your life.

What is a web application firewall?

The interesting thing about web application firewalls is that no one really knows exactly what they are. Or, to say that correctly, it is difficult to get different people to agree to a common definition. In very broad terms, web application firewalls are specialized tools whose purpose is to increase security in web applications. But try pressing a few people to give you a more specific definition and they'll give you more questions than answers. Some web application firewalls work are hardware devices, some are software applications. Where some are network-based, the others work embedded in web servers. You can try to compile a list of web application firewall vendors and visit their web sites in order to learn more, but chances are you will only get more confused reading what you find there.

Jeremiah Grossman, spokesman for the *Web Application Security Consortium* (webappsec.org), approached me in late 2004 with an idea of starting a project to figure out the web application firewalls. Having been involved with WAFs for some time I thought it was a beautiful idea. Other people must have liked the idea too because by mid 2005 we have had formed a team consisting of some of the very knowledgeable people in the web application firewall market. The *Web Application Firewall Evaluation Criteria* project was born. (It's home page is at webappsec.org/projects/wafec) The name of the project is a mouthful so we are usually referring to it by the abbreviation WAFEC. To save me some typing I will also refer to web application firewalls as WAFs from now on.

This text is not about the work we've done for the project, although my opinions have, without any doubt, been heavily influenced by it. I mention this project now (although it was bound to be mentioned sooner or later instead of later) is because I remembered one of the emails sent to the project list, which illustrates my point about market confusion beautifully. Achim Hoffmann, one of the fellow team members, had sent a list of names various people and organizations used to refer to web application firewalls over time. With Achim's permission, I am providing the full list here (with a couple of additions I thought were appropriate):

- Adaptive Firewall
- Adaptive Proxy
- Adaptive Gateway
- Application Firewall
- Application-level Firewall
- Application-layer Firewall
- Application Level Gateway

- Application-level Security Gateway
- Application Security Gateway
- Application Security Device
- Stateful Multilayer Inspection Firewall
- Web Adaptive Firewall
- Web Application Firewall
- Web Application Security Device
- Web Application Proxy
- Web Application Shield
- Web Shield
- Web Security Firewall
- Web Security Gateway
- Web Security Proxy
- Web Intrusion Detection System
- Web Intrusion Prevention System

There are 22 names on the list and *none* of them are entirely adequate, for the reasons I will soon discuss. Adequate or not, of all the names only one survived (guess which one). I verified my suspicions by using Google to search for each of the terms. Only the term "web application firewall" had any adverts associated with it.

Enough talk; what is a web application firewall?

The main reason it is often difficult for people to agree on a single definition for a web application firewall is simply because the name is used to refer to too many things. If you look at the lower network layers (web application firewalls are situated at layer 7) you will find that they are occupied by many devices, each specialized for a specific purpose. We have routers, switches, firewalls, intrusion detection systems, intrusion preventions systems, and so on. In the HTTP world, however, we are seeing roughly the equivalent functionality crammed into a single device type (and thus only one name): the web application firewall.

THE MAIN REASON IT IS OFTEN DIFFICULT FOR PEOPLE TO AGREE ON A SINGLE DEFINITION FOR A WEB APPLICATION FIREWALL IS SIMPLY BECAUSE THE NAME IS USED TO REFER TO TOO MANY THINGS.

Roughly speaking, it is possible to identify four distinct functionality types within what is today called a web application firewall (the names in the bracket refer to the equivalent devices on the lower network layers):

1. *Audit device.* Used to capture full content data (Network Security Monitoring) or only transactions that match some criteria (Intrusion Detection Systems).
2. *Access control device.* Used to control access to web applications, either in positive security mode (Network Firewalls) or negative security mode (Intrusion Prevention Systems).
3. *Architectural/Network design tool.* When operating in reverse proxy mode, used to distribute functionality, centralize access, virtualize infrastructure and so on.
4. *Web application hardening tool.* Features that increase web application security either by resolving the weaknesses inherent to web applications, or by targeting the programming errors in the applications they are protecting.

Because of the multi-faceted nature of WAFs people with different backgrounds tend to view them in different light. For example, people with background in network intrusion detection are likely to view WAFs as an IDS devices that just happen to operate on the HTTP level. Of course, to be entirely honest, the lower network layers are not without their problems either. For example, the distinction between intrusion detection and intrusion prevention if often not quite clear. But Richard Bejtlich summed it up well: “... an "IPS" is a layer 7 firewall that inverts the access control best practice of "allow some, deny everything else." (In other words, an IPS performs a "deny some, allow everything else" function.) I absolutely detest the IPS label and wish access control devices were simply identified as such, and not confused with audit devices (e.g., IDSs).”

Additional confusion is often introduced when the term *deep-inspection firewalls* is involved. Deep-inspection firewalls are devices which, some claim, have features equivalent to those of web application firewalls. But, although there is some similarity, the difference is profound. Deep-inspection firewalls generally make some effort to look beyond level 3 and into higher levels. Web application firewalls, on the other hand, are tools that are built from the ground up to handle HTTP and they understand it very well. A very good (and entertaining) overview of this subject has been provided by Marcus J. Ranum in his 'What is "Deep Inspection?' text, available at ranum.com/security/computer_security/editorials/deepinspect/.

It is important to accept one fact though: it is not necessary for a device to implement all four types of functionality in order for it to be called a web application firewall. As long as there is a clear

understanding of the possible variations, anything that increases web application security can be called a WAF as far as I am concerned. What end users need to do is first determine what their needs are and then find a tool that fulfils them.

Evolution of Web Intrusion Detection

Intrusion detection, as a concept, has been with us for many years. Its purpose is to detect attacks by looking at the network traffic or by looking at operating system events. The term intrusion prevention is used to refer to systems that are also capable of preventing attacks.

Today, when people mention intrusion detection, in most cases they will be referring to a network intrusion detection system (NIDS). An NIDS works on the TCP/IP level and is used to detect attacks against any network service, including the web server. The job of such systems, the most popular and most widely deployed of all IDSs, is to monitor raw network packets to spot malicious payload. Host-based intrusion detection systems (HIDSs), on the other hand, work on the host level. Though they can analyse network traffic (only the traffic that arrives to that single host), this task is usually left to NIDSs.

Host-based intrusion is mostly concerned with the events that take place on the host (such as users logging in and out and executing commands) and the system error messages that are generated. An HIDS can be as simple as a script watching a log file for error messages. Integrity validation programs (such as Tripwire) are also a form of HIDS. Some systems can be complex: one form of HIDS uses system call monitoring on a kernel level to detect processes that behave suspiciously.

Using a single approach for intrusion detection is insufficient. Security information management (SIM) systems are designed to manage various security-relevant events they receive from agents, where an agent can listen to the network traffic or operating system events or can work to obtain any other security-relevant information.

Because many NIDSs are in place, a large effort was made to make the most of them and to use them for web intrusion detection, too. Though NIDSs work well for the problems they were designed to address and they can provide some help with web intrusion detection, they do not and cannot live up to the full web intrusion detection potential for the following reasons:

• NIDSs were designed to work with TCP/IP. The Web is based around the HTTP protocol, which is a completely new vocabulary. It comes with its own set of problems and challenges, which are different from the ones of TCP/IP.
• The real problem is that web applications are not simple users of the HTTP protocol. Instead, HTTP is only used to carry the application-specific data. It is as though each application builds its own protocol on top of HTTP.
• Many new protocols are deployed on top of HTTP (think of Web Services, XML-RPC, and SOAP), pushing the level of complexity further up.
• Other problems, such as the inability of an NIDS to see through encrypted SSL channels (which most web applications that are meant to be secure use) and the inability to cope with a large amount of web traffic, make NIDSs insufficient tools for web intrusion detection.

Vendors of NIDSs have responded to the challenges by adding extensions to better understand HTTP. The term *deep-inspection firewalls* refers to systems that make an additional effort to understand the network traffic on a higher level. Ultimately, a new breed of IDSs was born. *Web application firewalls* (WAFs) are designed specifically to guard web applications. Designed from the ground up to support HTTP web application firewalls often work as reverse proxies. Instead of going directly to the web application, a request is rerouted to go to a WAF first and only allowed to proceed if deemed safe. Web application firewalls were designed from the ground up to deal with web attacks and are better suited for that purpose. NIDSs are better suited for monitoring on the network level and cannot be replaced for that purpose.

Though most vendors are focusing on supporting HTTP, the concept of application firewalls can be applied to any application and protocol. Commercial products have become available that act as proxies for other popular network protocols and for popular databases. (Zorp, at balabit.com/products/zorp/, available under a commercial and open source license at the same time, is one such product.)

Isn't it better to just fix the code?

Of course it is. But it is not that simple. Sometimes there is a controversy as to whether we are correct to pursue this approach to increasing security. A common counter-argument is that web intrusion detection does not solve the real problem, and that it is better to go directly to the problem and fix weak the vulnerable web applications. I tend to agree with this opinion. However the reality is preventing us from letting go from web application firewalls:

• It is not possible to make anything 100% secure - humans have limited capabilities and make mistakes.
• Attempting to approach 100% security is not even done in most cases. Today, in real life, those who direct application development usually demand features, not security. Attitudes are changing, but slowly.
• A complex system always contains third-party products (components, libraries) whose quality (security-wise) is not known. If the source code for the products is unavailable, then you are at the mercy of the vendor to supply the fixes. Even in the cases when the source code is available you are unlikely to have the resources to review it.
• We must work with existing vulnerable systems. Some of these legacy systems can not be touched.

It is, therefore, necessary to adopt a dual-approach strategy to achieve best results. You should work hard to raise awareness among management and developers. In the meantime do what you can to increase security straight away.

Life becomes much easier once you accept you will fail. To deal with the problem (in this case "deal" means minimize the chance of total failure) people invented an approach called *defense in depth*. By now, defense in depth is a well-known and a widely accepted security principle. The basic idea is that you don't want to put all your eggs into the same basket. Instead, assuming any part of the system can fail, you look for ways to configure other parts, or introduce new parts to limit the effect of the failure. Web application firewalls are just another piece in the security puzzle. Treat them as such.

You should work hard to raise awareness among management and developers. In the meantime do what you can to increase security straight away.

Web Application Firewall Features

The following sections describe some of the more interesting features frequently found in web application firewalls.

Protocol anomaly detection

If you read through various RFCs, you may detect a recurring theme. Most RFCs recommend that implementations be conservative about how they use protocols, but liberal with respect to what they accept from others. Web servers behave this way too, but such behavior opens the door wide open for all sorts of attacks. Almost all WAFs perform some sort of sanity check on incoming requests and refuse to accept anything that is not in accordance with the HTTP standard. Furthermore, they can narrow down the features to those that are acceptable to the application and thus reduce the attack surface area. Some will even go further than that, restricting certain aspects of the HTTP protocol that were left too loose or completely unspecified.

Enforcing input validation

A frequent web security problem occurs where the web programming model is misunderstood and programmers think the browser can be trusted. If that happens, the programmers may implement input validation in the browser using JavaScript. Since the browser is just a simple tool under control of the user, an attacker can bypass such input validation easily and send malformed input directly to the application.

A correct approach to handling this problem is to add server-side validation to the application. If that is impossible, another way is to add an intermediary between the client and the application and to have the intermediary reinterpret the JavaScript embedded in the web page.

Negative versus positive security models

If you have ever worked to develop a firewall policy, you may have been given advice to start with a configuration that denies access to everything, and only then proceed to allow the traffic you know is safe. That is a very good example of a positive security model. Negative security model does the opposite - access is allowed by default and some dangerous traffic patterns are denied.

The two approaches each ask a question:

• Positive security model: *What is safe?*
• Negative security model: *What is dangerous?*

A negative security model is used more often. You identify a dangerous pattern and configure your system to reject it. It is simple, easy, and fun. But it is not foolproof. The concept relies on you knowing what is dangerous. If there are aspects of the problem you are not aware of (which happens from time to time) then you have left a hole for the attacker to exploit.

A positive security model (also known as a white-list model) appears to be a better approach to building policies and works well for firewall policy building. In the realm of web application security, a positive security model approach boils down to enumerating every script in the application. For each script in the list, you need to create a list such as this one:

• Allowed request methods (e.g., GET/POST or POST only)
• Allowed Content-Type
• Allowed Content-Length
• Allowed parameters
• Which parameters are mandatory and which are optional
• The type of every parameter (e.g., text or integer)
• Additional parameter constraints (where applicable)

This list is just an example. A real-life positive security model typically includes more elements. Positive security model actually attempts to do externally what programmers are actually supposed to internally: verify every bit of information that goes into a web application. Using the positive security model is better if you can afford to spend the time to develop it. One difficult aspect of this approach is that the application model changes as the application evolves. You will need to update the model every time a new script is added to the application or if an existing one changes. But it works well to protect stable, legacy applications that no one maintains anymore.

Automating policy development can ease problems:

• Some WAFs can observe the traffic and use it to build the policy automatically. Some can do it in real time.
• With white-list protection in place, you may be able to mark certain IP addresses as trusted, and configure the WAF to update the policy according to the observed traffic.
• If an application is built with a comprehensive set of regression tests (to simulate correct behavior), playing the tests while the WAF is watching will result in a policy being created automatically.

So it turns out neither approach is entirely satisfactory on its own. Negative security model works well to deal with known problems. Positive security model works well for stable web applications. A combination of both approaches is ideal to use in real life.

Just-in-time Patching

The positive security model has the potential to work well because the communication between a browser and the application is well defined in the HTML specification. Scripts that make web applications are essentially designed to process requests that consist of a number of parameters. This is true for almost any application out there. Since these parameters are visible to the web application firewall it is possible to make decisions based on its observations. This leads to a interesting WAF capability I like to call just-in-time patching. Is it very simple, really.

When a vulnerability in an application is discovered in most cases forces are put in motion to close the hole in the code. Depending on the circumstances (the size of the application, availability of developers, legal arrangements and such) the process can last anywhere from a couple of minutes to, well, infinity. This is the window of opportunity for the attacker to exploit.

If you can close a vulnerability in code quickly then you don't have anything to worry about. But what if the length of the window of opportunity is measured in days or weeks? Web application firewalls are ideal tools to help with this: give a decent WAF to a security professional together with enough information about the security problem and he will likely be able to close the security hole in under one hour. Naturally, this type of protection is not foolproof and there is always a danger the fix is not complete but in case when you have no other choice any protection is better than no protection. Still, the advantages are worth restating:

1. Just-in-time patching is applied as a separate security layer.
2. It can be implemented straight away. (Some web application firewalls require changes to the network layout but some are happy to work embedded in the existing web server.)
3. It is work executed by a different user profile. Your typical developer may not be very skilled in application security, but chances are the security professional that discovered the problem is.

The principle of just-in-time patching is even easier to apply to the XML-based applications because the application communication is much

better documented. On the other end of the range are various AJAX applications, which tend to use their own unique protocols to exchange information between browsers and the application code. This is where just-in-time patching without custom programming becomes much difficult.

Rule-based versus anomaly-based protection

Rule-based WAFs comprise the majority of what is available on the market today. They subject every transaction to a series of tests. Each test can consists of one or more inspection rules. If the test fails, the request is treated as invalid and possibly rejected.

Rule-based WAFs are easy to build and use and are efficient when used to defend against known problems. They are also easy to use when the task is to build a custom defence policy. But since they must know about the specifics of every threat to protect from it, these tools must rely on using extensive rule databases. Vendors maintain rule databases and distribute their tools with programs to update WAF installations automatically.

This approach is unlikely to be able to protect custom applications or to protect from zero-day exploits (exploits that attack vulnerabilities that are not yet publicly known). This is where anomaly-based IDSs work better.

The idea behind anomaly-based protection is to build a protection layer that will observe legal application traffic and then build a statistical model to judge the future traffic against. In theory, once trained, an anomaly-based system should detect anything out of the ordinary. With anomaly-based protection, rule databases are not needed and zero-day exploits are not a problem. Anomaly-based protection systems are difficult to build and are thus rare. Because users do not understand how they work, many refuse to trust such systems, making them less popular than their rule-based counterparts.

State management

The stateless nature of the HTTP protocol has many negative impacts on web application security. Sessions can and should be implemented on the application level, but for many applications the added functionality is limited to fulfilling business requirements other than security. Web application firewalls, on the other hand, can throw their full weight into adding various session-related protection features. Some of the features include:

• *Enforcement of entry points.* At most web sites, you can start browsing from any site URL that is known to you. This is often convenient for attackers and inconvenient for defenders. An IDS that understands sessions will realise the user is making his first request and redirect him back to the default entry point (possibly logging the event).

• *Observation of each user session individually.* Being able to distinguish one session from another opens interesting possibilities, e.g., it becomes possible to watch the rate at which requests are made and the way users navigate through the application going from one page to another. Looking at the behavior of just one user it becomes much easier to detect intrusion attempts.

• *Detecting and responding to brute-force attacks.* Brute-force attacks normally go undetected in most web applications. With state management in place, an IDS tracks unusual events (such as login failures), and it can be configured to take action when a threshold is reached. It is often convenient to slow down future authentication attempts slightly, not enough for real users to notice but enough to practically stop automated scripts. If an authentication script takes 50 milliseconds to make a decision, a script can make around 20 attempts per second. If you introduce a delay of, say, one second, that will bring the speed to under one attempt per second. That, combined with an alert to someone to investigate further, would provide a decent defense.

• *Implementation of session timeouts.* Sessions can be expired after the default timeout expires, and users would be required to re-authenticate. Users can be logged out after a time of inactivity.

• *Detection and prevention of session hijacking.* In most cases, session hijacking results in a change of IP address and some other request data (that is, request headers are likely to be different). A stateful monitoring tool can detect the anomalies and prevent exploitation from taking place. The recommended action to take is to terminate the session, ask the user to re-authenticate, and log a warning.

• *Allowing only links provided to the client in the previous request.* Some tools can be strict and only allow users to follow the links that have been given in the previous response. This seems like an interesting feature but can be difficult to implement. One problem with it is that it prevents the user from using more than one browser window with the application. Another problem is that it can cause incompatibilities with applications using JavaScript to construct links dynamically.

Other protection techniques

Other security-hardening features specific to web application firewalls aim to remedy the problems that arise when the developers place trust in the input data. For example:

• *Hidden form fields protection.* Internal application data is sometimes exposed via hidden form variables, which are not hidden at all. Programmers often use hidden form fields to preserve process state, send data to the user and expect such data back with no modifications. Such data is very easy to change. This is a tough problem to solve, but WAFs usually employ selective cryptographic signing to deal with it.

• *Cookies protection.* Similar to the problems with the hidden form fields, cookies are sometimes used to transport private application data. Unlike hidden form fields, some cookies may contain very sensitive data so WAFs usually encrypt their contents altogether. Another approach is to completely virtualise the cookie mechanism. In such a setup the end users only see cookie tokens (similar to session tokens), while the cookies are kept safely in the web application firewall itself.

Anti-evasion techniques

One area where network-based IDSs have had trouble with web traffic is with respect to evasion techniques. The problem is there are so many ways to alter incoming (attack) data, so that it keeps the original meaning as far as the application is concerned, but to still have it modified sufficiently to sneak under the IDS radar. This is an area where being able to understand HTTP completely results in significant improvement.

For example, just by looking at whole HTTP requests at a time, an entire class of attacks based on request and packet fragmentation is avoided. And because they understand HTTP well and can separate dynamic requests from requests for static resources (and so choose not to waste time protecting static requests that cannot be compromised), they can afford to apply many different anti-evasion techniques that would prove too time consuming for NIDSs.

NO MATTER HOW YOU CALL THEM, WEB APPLICATION FIREWALLS ARE VERY USEFUL SECURITY TOOLS WITH A SECURE POSITION (NO PUN INTENDED) IN EVERY SECURITY PRACTITIONER'S TOOLBOX. THEY ARE HERE TO STAY.

Response monitoring and information leak prevention

Information leak prevention is a fancy name for monitoring of the outgoing HTTP traffic. In principle it is identical to request monitoring, and its goal is to watch the output for suspicious patterns and prevent the response from reaching the client when such a pattern is detected. The most likely candidates for patterns in output are credit card numbers and social security numbers. Another use for this technique is to watch for signs of successful intrusions.
It is not really possible to prevent a determined and skillful attacker from retrieving a piece of information, since she will always be able to encode the information in some a way as to prevent detection. Still, this technique can protect in the cases when the attacker does not have full control over the server but instead tries to exploit a weakness in the application.

Conclusion

No matter how you call them, web application firewalls are very useful security tools with a secure position (no pun intended) in every security practitioner's toolbox. They are here to stay. Whether the name itself will remain is a matter for debate. Many believe the name is not entirely adequate and that the major vendors will force a name change in order to get a bigger slice of the application market. This article only scratches the surface of this subject. If you care to learn more about it I suggest you visit the Web Application Firewall Evaluation Criteria project, which has recently had its first official release. It's a concise a document of twenty pages which lists various aspects of web application firewalls. Reading it probably won't be as entertaining as reading this article was but it can at least claim to be a serious attempt to provide a comprehensive analysis framework.

Ivan Ristic is a web security specialist and the author of ModSecurity (modsecurity.org), an open source web application firewall. He is the founder of Thinking Stone (thinkingstone.com), a web application security company. Ivan wrote "Apache Security" for O'Reilly (apachesecurity.net), a concise yet comprehensive web security guide for administrators, system architects, and programmers. Ivan is an active participant in the web application security community, and a member of the Web Application Security Consortium.

TAOSECURITY

THE WAY OF DIGITAL SECURITY

Know your network before an intruder does.

Understanding the network is the key to success in the digital realm. TaoSecurity works with customers to assess the threats to their organizations and improve digital situational awareness.

Using customized network security monitoring solutions, our clients stop most intruders, contain the ones that exploit remaining vulnerabilities, and implement rapid, efficient recovery actions.

We build on defenses already deployed in the enterprise and augment them with the knowledge, processes, and data needed for effective incident prevention, detection, and response.

Let us assist your enterprise with any of the following services.

Network Security Operations Training

TaoSecurity offers half-day, full-day, and week-long classes on all of the core competencies presented here, at your location with hands-on technical labs.

Network Security Evaluation

Do you wonder how your protection, detection, and response plans and processes compare to industry best practices? TaoSecurity can tell you.

Defensible Network Architecture

Using our monitor-control-minimize-current approach to enterprise administration, TaoSecurity can help your organization reduce its risk of compromise.

Network-Centric Forensics

Host-centric forensics discovers evidence on hard drives. Network-centric forensics finds compromised systems, intruder activity, and incident scope.

Enterprise Network Instrumentation

TaoSecurity selects, configures, and deploys sensors and wire access technologies to gain maximum insights into network traffic.

Network Incident Response

Do you suspect a breach, or have you found evidence of compromise? If so, TaoSecurity will help you detect, contain, and remove intruders from your network.

Network Security Monitoring

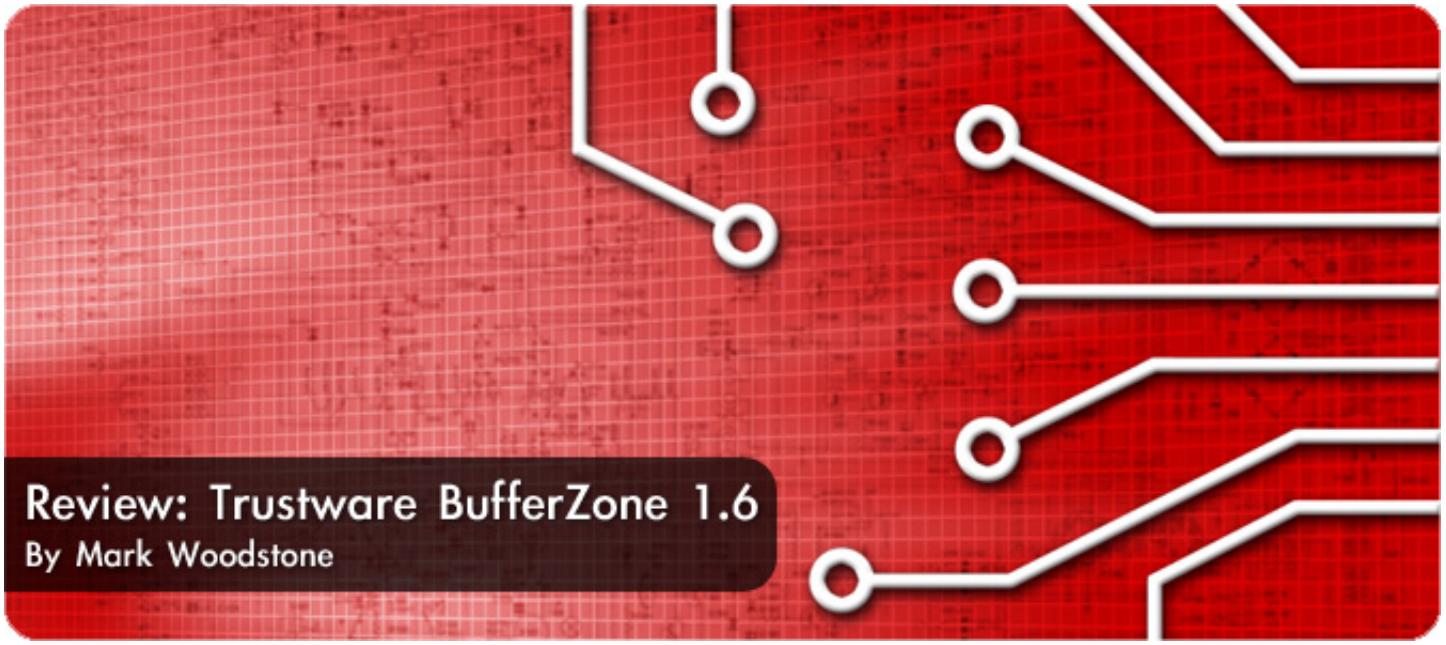
We augment access control or monitoring solutions to collect the alert, full content, session, and statistical data needed to identify and contain advanced intruders.

Traffic Threat Assessment

A vulnerability assessment finds systems with holes. A traffic threat assessment improves risk estimates by finding adversaries abusing or exploiting those holes.

TaoSecurity LLC

www.taosecurity.com - contact@taosecurity.com
9532 Liberia Ave Suite 141 - Manassas VA 20110



Over the years we have seen a number of different concepts that were trying to help the state of security of an average Windows PC user. Earlier, the only major problems were viruses, than we saw Trojans, worms, spyware, malicious scripting, etc. Antivirus software nowadays incorporates scanning for all the mentioned types of pests, but the approach that is based on signature updating and therefore on human intervention is not a perfect way to secure a PC user.

Security company Trustware (www.trustware.com) has a product that takes a new approach on protecting the end users. BufferZone is centered on a concept of virtualization technology, that creates a whole new secluded environment on your computer.

After installing the software, you are guided through a mini presentation that introduces you to the process of setting up your BufferZone. Although usage of terms like "virtualization" and "buffer" might be a bit complicated for the average PC user, the concept is very easy to comprehend and to setup.

Fighting the malware

Your connection to the Internet has probably the biggest potential of damaging your computer in any way. Using a non patched browser and visiting a site with malicious code can very fast compromise your computer. Downloading and starting a file without any proper checking by a 24/7 updated antivirus product could generate a massive infestation that will soon hurt your computer in many ways. These are just some of the constant threats PC users are susceptible to.

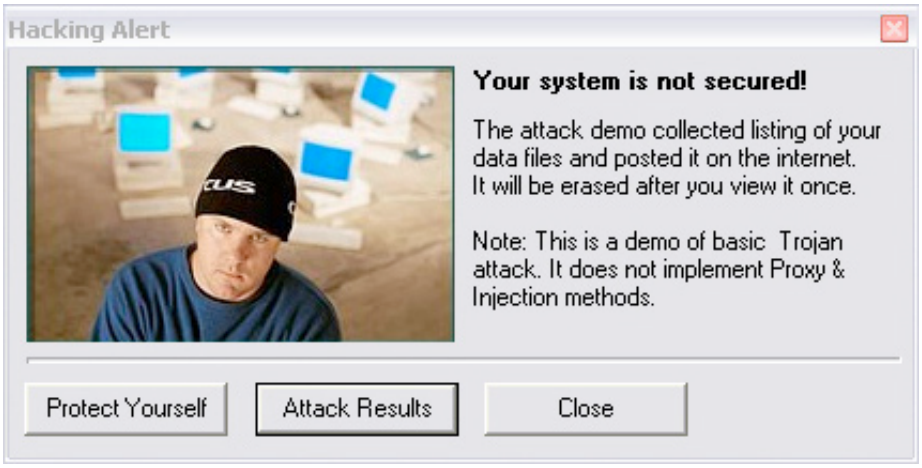
BufferZone comes to the rescue – with only a few of clicks you could create a defensive shield around all the pieces of software that interact with remote computers over the Internet. For instance, if you are still using Microsoft Internet Explorer, you are probably well aware of the problems unpatched versions of this software could generate. Never mind, just add Internet Explorer into the BufferZone and every potential malicious script will execute in this simulated environment and therefore won't have any impact on your real computer files.

From my perspective, the real power of BufferZone is not just real-time protection from the problems that can occur while browsing, but the possibility of reassuring that downloaded files are secure for running.

In the test case scenario, I tried to download a Trojan that gets a list of all my files and sends it to an online web page. I downloaded the file and started it while it was placed outside the BufferZone. The Trojan did its payload and very soon I could see my details online. I then sent the file to the BufferZone and started it once again.

This time the test Trojan encountered an internal error as he couldn't list my files, and it reported that my computer was secure. I usually download a lot of different files from the Internet, especially from sites like Sourceforge and Freshmeat.

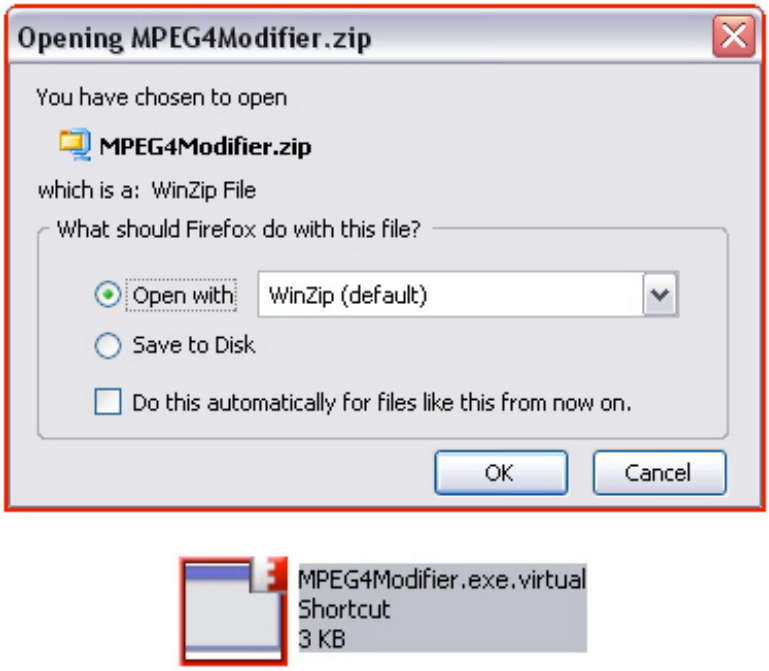
Although they have different methods of taking care of file integrity and security, you never know when you will come across an “evil” developer that will create some kind of a unsafe file.



Test Trojan output when application is run outside the BufferZone

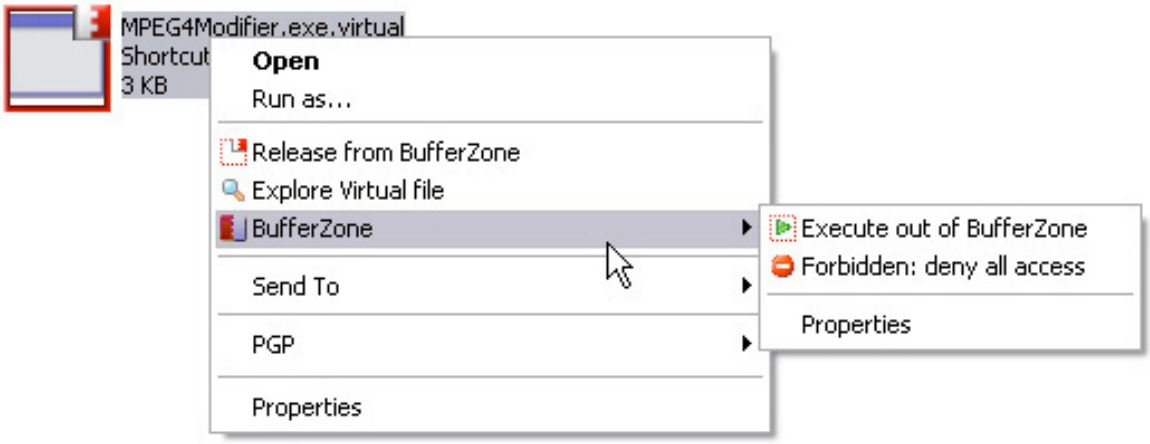
BufferZone main screen

During my tests, I ran different programs in the BufferZone, from simple SCP clients and Instant Messengers to an mpeg4 modifier program that I used for editing a couple of gigabytes of digital video files. All programs worked like a charm, I didn’t come across any potential problem. There is a very nice visual touch – all programs that are in the BufferZone have a red border around their icons and windows (see an example on the following page). This way you always know if you are working within an insecure or secure environment. If the border annoys you, you can disable it from the configuration menu.



The red border indicates that this file is in the buffer zone

The program hosts a couple of customizing features. You can group specific files under several categories including Web, Mail and P2P. This helps a bit as the most popular software is predefined. When you start BufferZone out of the box, it will immediately add the popular Internet related tools into its environment. You can also add your own software into these categories, making it easy to enable or disable a specific set of programs.



If you want, any file can be sent for execution outside of the protected zone

From the enterprise point of view, BufferZone 1.6 incorporates advanced management tools for monitoring, controlling and enforcing user activity throughout the LAN. These include an enterprise-wide, automated, scheduled BufferZone technology re-set that removes BufferZone values from Windows registries without data loss. Also, there is a tool that controls and prevents installation anywhere on the LAN of software not originating from designated servers and lets managers define acceptable filename extensions. Managers could also monitor all BufferZone activity in real time.

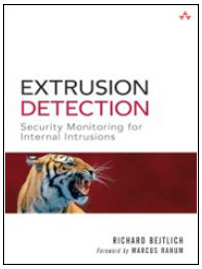
Overall, BufferZone is a must have software for Windows users. Its powerful virtualization engine creates a trusted environment that you will very soon fall in love with. The software is very easy to setup, manage and use.

Mark Woodstone is a security consultant that works for a large Internet Presence Provider (IPP) that serves about 4000 clients from about 30 countries worldwide.

Extrusion Detection: Security Monitoring for Internal Intrusions

by Richard Bejtlich

Addison-Wesley Professional, ISBN: 0321349962



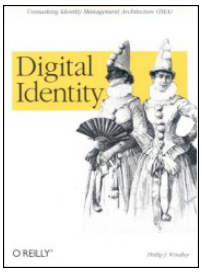
Extrusion Detection is a comprehensive guide to preventing, detecting, and mitigating security breaches from the inside out. Bejtlich teaches you how to assess threats from internal clients, instrument networks to detect anomalies in outgoing traffic, architect networks to resist internal attacks, and respond effectively when attacks occur.

If you've enjoyed Bejtlich's previous publications, especially The Tao of Network Security Monitoring, you will love this one.

Digital Identity

by Phillip Windley

O'Reilly, ISBN: 0596008783



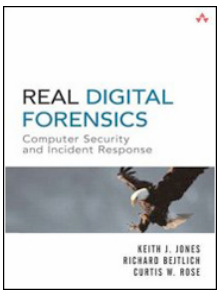
The author shares his extensive knowledge on the ideas, issues, and technologies behind a key concept known as Identity Management Architecture" (IMA).

Focused on upper management and IT professionals working in this field, the book covers in details set of standards, policies, certifications, and management activities that enable companies to manage digital identity effectively.

Real Digital Forensics : Computer Security and Incident Response

by Keith J. Jones, Richard Bejtlich, Curtis W. Rose

Addison-Wesley Professional, ISBN: 0321240693

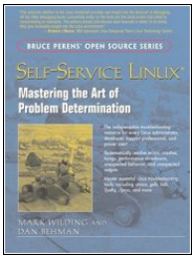


If you are into forensics, this book is probably already on your book case. If not, you should definitely check this out.

The authors provide five different scenarios and show you what steps to take and what tools to use in the process of incident response. The book is complemented with a DVD with all the evidence collected for each of the scenarios, which makes the educational perspective of this book much more interesting.

Self-Service Linux: Mastering the Art of Problem Determination

by Mark Wilding, Dan Behman
Prentice Hall PTR, ISBN: 013147751X

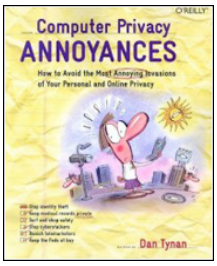


In Self-Service Linux, two of IBM's leading Linux experts introduce a four-step methodology for identifying and resolving every type of Linux-related system or application problem: errors, crashes, hangs, performance slowdowns, unexpected behavior, and unexpected outputs.

If you're involved with deploying or managing Linux in the enterprise, it can help you significantly reduce operation costs and enhance availability.

Computer Privacy Annoyances

by Dan Tynan
O'Reilly, ISBN: 0596007752

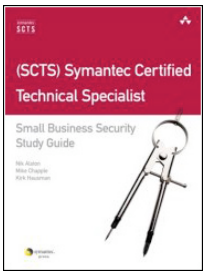


This is a very interesting little book that will make you think a bit more about your privacy, both at home, work and online. It contains a myriad of good tips, each of them containing information on the actual annoyance and a possible solution.

Although the title of this book implies that it focuses on computers, the author also managed to give a lot of good tips on various real life stations, including dealing with the IRS, US government, postal service, etc.

(SCTS) Symantec Certified Technical Specialist: Small Business Security Study Guide

by Nik Alston, Mike Chapple, Kirk Hausman
Addison-Wesley Professional, ISBN: 0321349946

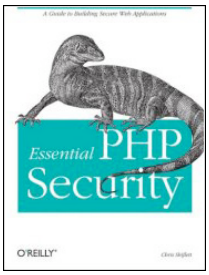


Symantec's Certified Technical Specialist (SCTS), Small Business Security certification allows security professionals to validate their knowledge of today's most crucial information security techniques in combination with Symantec's security products.

This guide covers the exam objective in depth; everything you need to know to pass your exam the first time. The book comes with a CD that contains a couple of SCTS sample exams.

Essential PHP Security

by Chris Shiflett
O'Reilly, ISBN: 059600656X



This hundred pager should be "a must" for every self-conscious PHP developer. A large majority of PHP web applications had some kind of a security vulnerability, so developers, start your engines. "Essential PHP Security" is a straight-forward book, it has 100 pages and hosts a precise problem/solution type of content. This could also be a good read to penetration testers, as it will definitely broaden their knowledge on the subject.

Threat analysis using log data

By Kevin J. Schmidt

System and security administrators have long known the value in capturing and analyzing log data. Systems administrators tend to focus on operating system logs, while security administrators focus on router, firewall, and similar log data. Unfortunately these groups rarely see how system logs and security log data can be used together to paint a better overall picture of what is going on in the environment.

The goal of this article is to educate system and security administrators, and others, on the value in analyzing disparate log data to discover potentially malicious behavior.

This article is broken up into the following sections:

- Log Data Basics
- Log Gathering Architecture
- Prepare Log Data for Analysis
- Analyzing Events for Threats
- Threat Analysis Example
- Tools of the Trade

Let's begin by summarizing the threat analysis process. The following five steps briefly discuss the process.

Step 1. Configure

Generally you will need to configure your systems to begin emitting log data. This is system- and device-specific so this step will not be discussed in this article.

Step 2. Understand

Understanding what sort of log data your systems can emit is critical. It is through this understanding that you are then able to effectively analyze data for interesting behavior. It may be that you are well versed on the nature of your systems and you can formalize what sort of things you will want to look for. Or it may be that you are somewhat new to administration of security systems and may need input from others in your organization or from online resources. For example, your accounting group may have ideas on the sort of things to look for with respect to financial systems and such.

Step 3. Collect

Aggregation is often used to describe the act of collecting log data to a central location. By collecting all log data to a central you are able to look at things as a whole and perform effective analysis.

Step 4. Prepare

Preparing log data for analysis is performed through *normalization*. The end result of normalization is the creation of an event, which is used in the analysis process.

Step 5. Analyze

The fifth and final step is analysis. Here we are concerned about discovering potentially malicious behavior.

Analysis is generally performed against events, but in some instances analysis is performed against raw log data.

Let's now begin the journey down the path to threat analysis.

Log Data Basics

Log data is a general term used to identify information which can be used to better understand what is going on with a particular system, set of systems or network. Some systems, like operating systems, store log data on disk. Others like routers or hardware-based systems don't have internal disks to store log data on; they simply emit log data.

Most people think of a file on disk when they hear the term log data or log messages. UNIX administrators tend to think of `/var/log/` or `/var/adm` as repositories for log data. Windows people, on the other hand, are used to dealing with the Event Log. As for log data sources, again most people think of UNIX or Windows servers, routers, and such.

While these are accurate, the source of log data is not limited to a certain type of system. The following is a partial list of system/device classes which are capable of generating log data.

- Operating System (OS)
- Firewall
- Network Intrusion Prevention System (NIPS)
- Host Intrusion Prevention System (HIPS)
- Authentication Systems (Kerberos, Radius, etc.)
- Vulnerability Assessment (VA)
- Anti-Virus (AV)
- Anti-Spam
- Router
- Switch

The type of information contained in log data varies greatly. Some examples of the type of information which could be used for threat analysis include:

- Login/Logout Messages
- User Account addition, modification, or deletion
- Disk Full Messages
- Firewall allow/deny Messages
- NIPS Messages
- Web Server Logs
- Many others

Log Data Transmission

Log data transmission deals with how data is sent and received. This includes knowing what transmission protocols your systems support and configuring them to send data. The most common protocol is Syslog. Generally speaking some piece of software is run to gather log data sent via one of these transmission protocols.

The following list of Internet-protocol specific methods are utilized by many devices and systems systems:

Syslog

Syslog (System Logger) is a simple UDP-based protocol. Syslog logging comes in two flavors: local logging and remote logging. Local logging is where logs are generated and stored on the same machine. Remote logging is where one machine generates a log message but forwards it on to another machine for storage. By default all data transmission is sent in clear-text. Since Syslog is the most commonly used mechanism, let's delve into it a bit more. Syslog messages generally have the following simple format:

```
<timestamp><hostname><message source><message>
```

The format is as follows:

Timestamp
This is the time and date of when the message was created. If local logging is used then it's based on the local machine's time. If the message was received from a remote machine then the time is based on the remote machine.

Hostname
The hostname can be an IP address (if no DNS information for the IP address exists), fully qualified domain name (FQDN) or simply a hostname.

Message source
The message source indicates, as you might have guessed, the source of the message. For operating-system components or applications, the source is usually a process name. Note that when receiving messages from devices like routers, switches, and firewalls, you generally will not get a process ID as part of the source. Often times all you get is something along the lines of the vendor name.

Message
The message contains specific detail on what happened. Note that message formats between applications, systems, devices, vendors, etc., all differ, e.g. it is very free-form in nature.

SNMP

SNMP (Simple Network Management Protocol) is also UDP based.

There are three versions of SNMP: SNMPv1, SNMPv2, and SNMPv3. SNMPv3 adds security to the protocol in the form

of authentication and encryption. Many systems employ SNMP traps for sending of log data.

SMTP

SMTP (Simple Mail Transfer Protocol) is TCP-based. While many systems support SMTP for notification, it is rare to see a system which uses SMTP as its log data emission mechanism, but a few do exist. The default is to send in clear-text.

Beyond these Internet-standard methods, proprietary ones also exist. These are protocols and APIs that commercial vendors have created. Some these include the following:

Checkpoint OPSEC LEA

The Open Platform for Security (OPSEC) is an open and extensible framework for managing all aspects of network security. The Log Export API (LEA) is one API under the OPSEC umbrella. LEA can be used to gather Checkpoint firewall logs from Checkpoint's SmartCenter management platform. The LEA API uses encryption to securely transmit data.

Cisco RDEP/SDEE

The Remote Data Exchange Protocol (RDEP) is Cisco's first generation protocol for gather log data from its IPS product. The Security Device Event Exchange (SDEE) extends and updates RDEP. Both clear-text and encrypted modes are supported.

Sourcefire E-Streamer

E-Streamer is a protocol used to gather log data from SourceFire IPS. Both clear-text and encrypted modes are supported.

Windows Event Log

The Windows Event Log is Microsoft's central source for logging. There are three main log types: System, Application, and Security.

Log Gathering Architecture

Gathering log data requires you to configure your systems to emit log data. Once this is done, you then need a place to capture all this data. At a minimum you need a server that will act as the central collector or log server. Aggregation is used to describe the act of gathering log data in one place. A log data architecture generally has a number of components which are discussed now.

Collector

The collector is used to collect and aggregate log data from log data sources.

Analysis Server

The analysis server does the actual work of analyzing log data for threats. Note that the collector and analysis server may be on the same machine or different machines for efficiency.

Archive Server

The archive server is used to store log data, either in raw form, normalized form or both, so it can be analyzed at a later time or for report generation. Typically a database is used to store this information.

Administrator Console

The administrator console is generally some piece of software which is used for viewing log data, events, alerts, reports, etc.

Reporting

Reporting is generally performed on data in the archive database.

Figure-1 (on the following page) depicts a basic log-gathering architecture.

There are actually two architectures within Figure-1. The fist one is the most basic. The firewall, router, database and Web server are all configured to send their log data to the central log server. The central log server is responsible not only for gathering log data, but for preparing the data for analysis.

The second architecture is one where a remote collector is used at a remote network site. It is called a collector because it is responsible for collecting log data, but it forwards what it receives to the central log server. There are several reasons for using a remote collector:

- You may not want all log data flowing over your Internet link. You can filter out messages you don't care about or want to analyze and save on bandwidth.
- The collector could prepare the data before it is sent to the central server. This would allow the server to not have to work as hard and spend more time analyzing critical events.
- You may want to encrypt the data sent over the Internet.

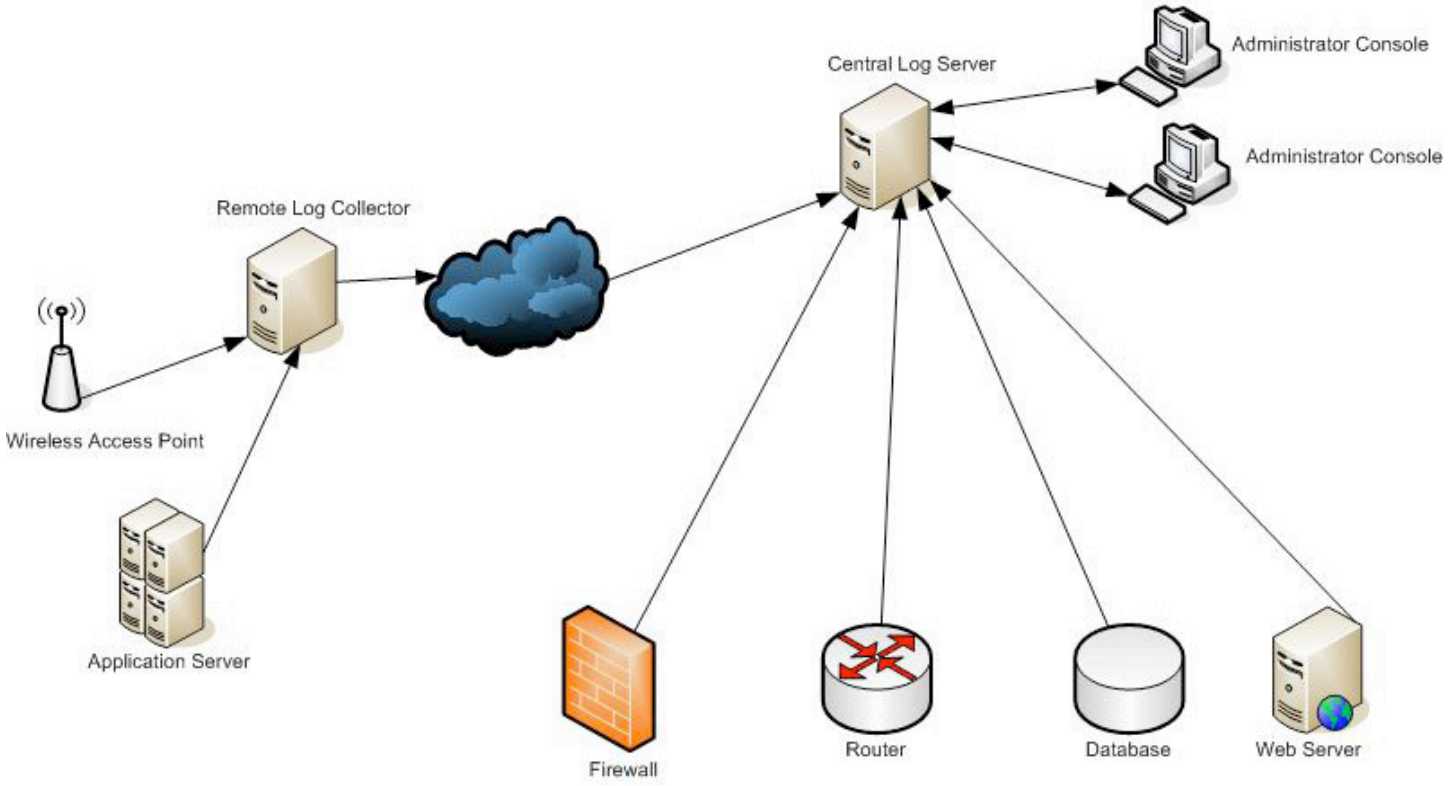


Figure-1. Log gathering architecture

The final component in the architecture is the administrators/analysts who are consumers of the gathered log data and analyses. It is their job to be aware of what's going on both at the network level and at the machine level. For more information on creating a logging server, see Anton Chuvakin's article Advanced Log Processing (www.securityfocus.com/infocus/1613). He also discusses how to secure Syslog transmission.

Prepare Log Data for Analysis

While Syslog's protocol has a common format, the device- or system-specific data format itself varies widely. This is because no two vendors pick the identical format for messages and log data.

Preparing Log Data

Recall that Step 4 in the threat-analysis process is to prepare log data. This is a critical and important step in the threat analysis process. Unfortunately, it is also tedious and error prone. This is because you have to know and understand the format of the log data you gather. Some vendors provide excellent documentation on their message formats, while others either inaccurately document or provide no documentation at all. Normalization is the process of going from a specific format to a common one without loss of precision. The output of normalization is an event. This event is what is used during the analysis phase. But what is an event?

Event Creation

Think of an event as the currency used within an analysis system. How an event comes into existence depends upon the underlying locomotion mechanism used for normalization. Here are some example mechanisms.

Database
A schema can be created which embodies an event format. Log data is received, normalized and inserted into an event table. Standard SQL can then be used to analyze the event table. The use of a database is quite common and can make life a little easier.

Programming language-specific Structure
The C programming language allows the programmer to create user-defined types using structures. Object oriented languages like Java and C++ facilitate the creation of user-defined types via classes. Regardless of the language used, the goal is the same as with a database. The main difference is that a database isn't used. Instead you have a system, written in one particular language, broken up by components. One component gets the log data, normalizes it. Events are created in a language specific manner and handed off to an analysis component. The analysis component may be on the same machine as the normalization component or it may be on a remote machine (for efficiency purposes). This means some sort of inter-process communication

(IPC) will need to be used to pass information from one component to the next.

Flat-file
A simpler way to create events is to use flat-files where each line contains a CSV line of text. Each value would map to specific field in your normalized event. Similar to a database, the flat file would be processed for analysis.

Event Fields

Now that we have established what an event is, what sort of fields should we have in our event structure? The following list outlines some of the more common fields that are of interest to system and security administrators.

Date and Time
It is important to record the time the message was received by the collector or server. Some vendors also include the time the message was created, which should be captured in another field. Some products and systems emit epoch timestamps while others generate normal month, day, time, etc. It is often easier to deal with epochs, so consider normalizing all date and times to an epoch.

Application/File Names
This could either be the application which generated the message or an application which attempted to perform an illegal action, e.g. a host intrusion prevention system can identify potentially malicious applications.

Application Exit Codes
Some log messages may contain exit codes, which could help point out dubious behavior.

Source and Destination IP Addresses and Ports
Firewall and NIPS systems generate source and destination IP address and port information which can be used to discover malicious behavior patterns, among other things.

Taxonomy Type
A Taxonomy is a set of types which are aligned around conceptual boundaries. For example, one firewall vendor may emit a message which uses the word “accept”, but a different firewall vendor may use the word “allow”. Through use of a taxonomy, these two messages could be normalized simply as accept. But taxonimification doesn’t stop at homogenous device normalization. It can also be used for heterogeneous normalization.

For example, let’s say IPS vendor A emits log Message-A. Firewall vendor B emits log Message-B. It turns out that Message-A and Message-B are the same conceptually. This means that two distinct messages are now collapsed down to one single concept.

Priority
The priority is used to determine how severe an event is. Some of the log data you normalize will have its own priority value as part of the message, while others will not. It is generally a good idea to establish your own priority scheme and map your log data’s priorities to this scheme. One simple scheme is to use low, medium, and high.

Protocol
The most common are UDP and TCP.

ICMP Type and Code
When the protocol is ICMP, don’t forget to record the type and code.

Username
Capturing username information is very useful in tracking down malicious attackers who log into a machine and attempt to do something nasty like fill up hard disks, crash running programs, etc. Unfortunately, username information is generally only available in certain log messages.

Domain
Domain could refer to Windows domain or the domain name portion of an email address, etc.

Email Address
An email address may be present in SMTP/POP messages.

The Mechanics of Normalization

Normalization in the case of log data is sometimes referred to as parsing. The most common way parsing is performed is through regular expressions. It allows for the most flexible processing possible. Care must be taken, however, to ensure you create regular expressions which perform as optimal as possible. Jeffrey E. F. Friedl’s book Mastering Regular Expressions, 2nd Edition (www.oreilly.com/catalog/regex2) is one of the best resources for learning everything you need to know about regular expressions.

The following sections provide actual parsing examples. Even though Perl is used with the examples, the concept would be the same regardless of the language or technique used.

UNIX Login
When someone logs into a UNIX system using SSH, this activity is recorded in the form of a log message. The following is an example Syslog message from a Linux machine:

```
Jan 10 14:57:29 server login(pam_unix)[2653]: session opened for user root by LOG-
IN(uid=0)
```

The following Perl snippet shows how to process this log messages.

```
# Assume $message contains the message
my($month, $day, $time, $host, $process,
  $pid, $user, $rest) = $message =~
  m/^(\w{3}) (\d{2}) (\d{2}:\d{2}:\d{2}) (.*?) (.*?)\[(\d+)\]: session opened for
user (\w+) by (.*?)$/g;
```

Here I grab month, day, time, host, process, pid (process id), user, and rest.

Cisco PIX
Cisco PIX is Cisco's firewall product. It is capable of generating an extensive amount of valuable log messages. Here are two such examples.

```
Jan 11 2006 10:00:03: %PIX-4-106023: Deny tcp src dmz:10.0.3.4/36637 dst
outside:10.0.2.2/25 by access-group \"in_dmz\"

Jan 11 2006 16:21:25: %PIX-6-106015: Deny TCP (no connection) from 10.3.3.15/80 to
10.21.1.3/41063 flags SYN ACK on interface outside
```

This illustrates and interesting point. Both of these are TCP deny messages, yet neither of them have exactly the same format. This is a common problem in the real world and you need to be aware of it. So let's look at how we might parse these.

```
# Assume $message contains the message
my($month, $day, $year, $time, $type, $protocol) = $message =~ /^(\w{3}) (\d{2})
(\d{4}) (\d+:\d+:\d+):.*?: (.*?) (.*?) /g;

my($srcIp, $srcPort, $dstIp, $dstPort) = $message =~
/(\d+\.\d+\.\d+\.\d+)\/(\d+)/gc;
```

I use two regular expressions to process the message. The first regular expression gets the month, day, year, and time. Notice how the PIX messages, unlike the UNIX message, has a year as part of the date. Next I grab the type of event and protocol. The type for both messages is Deny. Protocol is the same for both, but one is uppercase and the other is lowercase.

The second regular expression obtains the source and destination IP addresses and ports. I used the `/gc` modifier, which allows Perl's regular expression engine to keep matching after `/g` fails. This is why I only specify a single pattern but I am able to get both IP addresses and both ports. Unfortunately, this will not work for many PIX messages. Some PIX messages will contain NAT addresses, too, which would cause our regular expression to miss some or all of the information we need.

It's worth while at this point to discuss some things that can go wrong with parsing. The PIX type (in this case Deny) just happens to be in the same place in both messages. The regular expression I wrote takes advantage of this fact. However, there are many PIX messages that either have no type or have the type string some place else in the message. Your parsing should be able to handle these conditions.

One approach is to create message-specific parsers. Notice how both messages have a token which looks like `%PIX-X-YYYYYY`. Each PIX message has this token. The six-digit number is a unique identifier for the event. Fortunately Cisco documents the format for all PIX messages and they do so by the version of PIX.

Analyzing Events for Threats

Richard Bejtlich has written that "the process by which the intentions and capabilities of threats are assessed is called threat analysis." I really like this definition because it simply makes sense.

It should be noted that threat analysis cannot replace the human factor in network security. Threat analysis techniques provide better information on

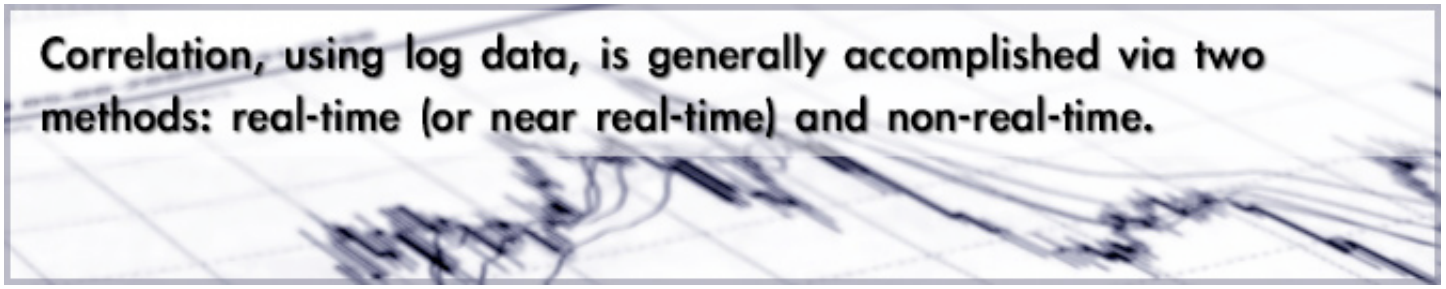
what is going on in an environment, but an analyst or administrator may still need to investigate further by firing up a packet capture tool, inspect OS logs, etc., to make a determination that something malicious really happened. Let's now look at techniques used for analyzing log data for potential threats.

Correlation

Correlation is the buzzword used in conjunction with threat analysis. It is often times touted as a panacea. In reality it isn't. What people generally mean when they mention correlation is: looking at all events in aggregate and grouping similar things together to see if they relate to each other in interesting ways. It is the aggregation of log data to a single place which allows correlation to take place. In the context of log data, we want to group similar events together to discover possible malicious behavior. The real goal of correlation is to provide better intelligence to administrators so they can investigate possibly dubious behavior.

Some people may say "why should I care about correlation since firewall, IPS, and other systems can aid in the detection and prevention of malicious behavior?" This is certainly true to a certain extent. But what about the situation where a certain IPS event combined with a certain OS event constitutes a higher-level threat which cannot be detected solely by the IPS event or OS event? This is where correlation is beneficial.

Correlation, using log data, is generally accomplished via two methods: real-time (or near real-time) and non-real-time. Generally speaking, real-time relies on rules to specify what things to look for; non-real-time analysis deals with methods which can either supplement real-time analysis or stand alone in its own right, but are generally done after the fact, i.e. forensically. These topics will now be discussed.



Real-time Analysis

Rules are simply a prescribed set of checks or conditions whose entire truth value is evaluated. There are many ways rules can be written. For example, you may use the if-then structure of a programming language to embed rules in application code. Or you way wish to purchase a standalone rule engine which evaluates events fed to it with an externally created rule set.

The purpose of this section is simply to present the idea of rule-based analysis. A few pseudocode examples will be provided to drive home this analysis technique.

Recall that our definition of correlation stipulates that we group events together. The first kind of grouping involves events from the same device type. Consider the following example:

```
Event A = accept
Event B = deny

IF (10 B's are followed by 1 A)
THEN
Possible scan success
```

This rule works on two firewall events. Event A is a normalized event with taxonomy accept. Event B is also normalized to deny.

The rule itself is looking for 10 B's followed by a single A, in other words 10 accepts followed by a single deny. This form of analysis is useful, but we also want to group across event types.

For example:

```
Event A = portscan
Event B = accept

IF (A's destination IP == B's destination
IP) AND
(A's destination port == B's destination
port)
THEN
Possible destination breach due to open
firewall rule
```

Here Event A is a normalized IPS port scan event. Event B is a firewall accept. The rule uses destination IP address and port fields in Events A and B to identify a possible scan breach due to an open firewall hole.

Non-real-time Analysis

The techniques discussed in this section can be used to supplement real-time analysis or as standalone methods. It is the nature of these methods that they operate over data which has been accumulated for some time period.

Statistical Methods

Statistical methods can be employed to discover interesting things that rules generally cannot. A few of these methods are discussed now.

Baseline
A baseline is simply a set of data which represents normal values. Trends can be discovered by evaluating new data against an established baseline.

Thresholds and Windows
With windowing we wish to discover possible dubious behavior that occurs outside of a certain range, e.g. time of day, etc. For example, you may want to know any time your router's configuration changes outside of scheduled maintenance windows. The comparison of some event against a simple baseline is considered thresholding. For example, you may wish to know when a user fails to login five times in a row. The value five is a threshold.

Never Before Seen (NBS) Detection
NBS detection deals with determining when something hasn't happened before. Marcus Ranum's NBS (www.ranum.com/security/computer_security/code) tool can aid in this endeavor.

Other Statistical Techniques
The following statistical techniques can be used by themselves or in conjunction with the other methods discussed in this section.

- Standard Deviation
- Moving Averages
- Ratios
- Range
- Interquartile Range
- Variance Analysis

Vulnerability Correlation
Vulnerability correlation is the use of vulnerability assessment data to determine how valid an event is. For example, if your IPS detects that an attempt has been made to exploit some sendmail vulnerability on a Windows server, which isn't running sendmail, then you can disregard this event or set its priority lower.

External Data Correlation
This technique is similar to vulnerability correlation in that it uses external data sources to validate events. For example, you may take in a feed from some security Web site and correlate an increase in a certain type of event with the outbreak of some new worm.

Final Thoughts

Never underestimate the value in learning from others. Here's a brief list of books I have found useful for helping me think about threat analysis:

- Building a Logging Infrastructure (www.sage.org/pubs/12_logging/) by Tina Bird and Abe Singer
- Schaum's Outline of Statistics by Murray R. Spiegel and Larry J. Stephens
- The Tao of Network Security Monitoring: Beyond Intrusion Detection by Richard Bejtlich
- Extrusion Detection: Security Monitoring for Internal Intrusions by Richard Bejtlich.

Threat Analysis Example

A walk through of how real-time analysis can be used to aid in threat analysis is probably in order. Let's say the following events show up in `/var/log/auth.log` on a Unix system:

```
Dec 20 15:00:35 host PAM_unix[11351]: check pass; user unknown
Dec 20 15:00:35 host PAM_unix[11351]: authentication failure; (uid=0) -> **un-
known** for ftp service
Dec 20 15:00:43 host PAM_unix[11351]: check pass; user unknown
Dec 20 15:00:43 host PAM_unix[11351]: authentication failure; (uid=0) -> **un-
known** for ftp service
```

But moments before this the following showed up in `/var/log/daemon.log`:

```
Dec 20 15:00:30 host in.ftpd[11351]: connect from 10.0.3.4
```

To understand what is going on here we need to have access to messages written to both log files. But more importantly than this is that we have the proper knowledge and experience to know what constitutes a possible attack, break in, etc.

We first need to normalize the events. Beyond this, however, it is important to properly map the messages to a taxonomy. For example, the first message in `/var/log/auth.log` could be classified as `credential-check`. The second message could be `auth-failure`. We notice that the third and fourth messages are really the same as the first and second messages respectively. Finally, the message from `/var/log/daemon.log` could be classified as `access-attempt`.

The next step is to write a rule to combine or correlate these normalized events. Writing a rule to catch such behavior is tricky. In the case of our FTPD messages, we see that the process ID (the number between the square brackets, "[11351]") is the same. This is because the `in.ftpd` process (in `/var/log/daemon.log`) spawned a subprocess to handle the incoming connection. We know the process name and the process ID because of the string "in.ftpd[11351]". Notice, however, that `/var/log/auth.log` shows the same process ID but different process names ("PAM_unix[11351]"). This means when crafting a rule to tie these events together we will need to make sure the process ID is used to tie together these messages into a single session. So how would we detect something like this? A rule can be used to specify how to determine that something has happened and then what to do about it. We can express it in pseudo-English with the following:

```
Event A = credential-check
Event B = auth-failure
Event C = access-attempt

IF((B.count >= 2 ) AND
(A.processID == B.processID) AND (B.processID == C.processID) AND
(TimeDifferenceBetween(A,B,C) <= 10)
THEN
Create and investigate an event with process ID, hostname, etc.
```

This rule attempts to detect a situation where there are two authentication failures occur. It also makes sure the process IDs for events A, B and C are the same and the time difference between them is no greater than 10 seconds. In other words, make sure all three events are tied together by process ID and also make sure they occur in close proximity to each other (less than 10 seconds all together). What if we wanted to use this rule for systems where we don't get process ID information? It is the case that we may very well have non-OS message taxonomized exactly the same way. We could rewrite the previous rule as follows:

```
A = credential-check
B = auth-failure
C = access-attempt

IF((B.count >= 2) AND
(A.srcIp == B.srcIp) AND (B.srcIp == C.srcIp) AND
(TimeDifferenceBetween(A,B,C) <= 10))
THEN
Create and investigate an event with sourceIP of attacker, etc.
```

Here we are using the source IP address to ensure that events A, B and C are from the same attacker. Of course there will be times when process ID and source/destination IP address information will not be available. Sometimes you have to do the best you can with what you have.

Tools of the Trade

Now that you have a firm foundation for the steps involved in threat analysis, how do you actually go about achieving this goal? Fortunately you have several options at your disposal to help you. They range from building your own solution to using open source software.

Roll Your Own Solution

If you have software development expertise you can opt to build your own log data gathering and analysis system. This is probably the least desirable approach, but it is an option.

Commercial Solutions

Security Information Management (SIM) and Security Event Manage (SEM) companies have sprung up over the last five or so years to meet the growing emphasis on log data analysis. SIM software is delivered either as an appliance or shrink-wrapped software and utilizes a three-tiered architecture. The first tier is a collector which is used to gather and normalize log data. The second tier is an analysis and storage system. The storage system is used to store events in long-term storage. This is done for forensic purposes as well as historical reporting. The console administrators use to view events is the third and final layer. One advantage of commercial vendors is they tend to support a wide variety of devise and systems out of the box. This makes your job easier, i.e. because you don't have to spend time worrying about log data format issues.

Some of the bigger players in the SIM market include:

- Intellitactics (www.intellitactics.com)
- ArcSight (www.arcsight.com)
- netForensics (www.netforensics.com)
- GuardedNet (www.guraded.net)
- LogLogic (www.loglogic.com)
- LogRhythm (www.logrhythm.com)

Open Source Solutions

In the open source realm you have a lot of different tools to choose from. These tools range from Syslog daemon replacements to log analysis programs to full-blown SIM solutions. The following list is by no means exhaustive. It is simply meant to give you a feel for what sort of open source tools are out there.

syslog-ng

syslog-ng (www.balabit.com/products/syslog_ng) is a replacement for the standard UNIX Syslog daemon. It is unique in that it is highly configurable and supports TCP transmission and extensive filtering. It also supports customizable data mining and analysis features.

High Performance Syslog

The San Diego Supercomputer Center (SDSC) maintains a high-performance Syslog replacement (security.sdsc.edu/software/sdsc-syslog/). It boasts the following features:

- Input modules for socket, UDP network connections, TCP/BEEP, etc.
- Message switch to perform log message routing
- Multiple output modules for UDP, TCP/BEEP, "syslog classic" files, structured files
- Multi-processing - handles more input syslog steams, provides better scalability
- Support for draft standards such as "syslog-reliable" (RFC 3195, syslog messages over BEEP).

Simple Event Correlator (SEC)

SEC (estpak.ee/~risto/sec/) is a Perl-based system for analyzing data via several different methods like regular files, named pipes, and standard input. It uses rules to instruct it how to analyze and react to events. External programs or analysis modules can be spawned from rules for greater flexibility.

Open Source Security Information Management (OSSIM)

OSSIM (ossim.net) is an open source SIM tool that aims to be as feature-rich as its commercial counterparts. It supports normalization, correlation, and risk assessment among many other features.

LogAnalysis.Org

LogAnalysis.Org (loganalysis.org) is not an open source tool, but is a resource for all things related to log data analysis. It includes mailing lists and a comprehensive resource library on log data analysis tools, systems, software, etc. This site should be one you visit to learn more about what open source (and commercial) tools are available.

Conclusion

Threat analysis involves gathering, normalizing, and analyzing log data.

The end goal is to correlate data from many sources to better detect dubious behavior, which may not be detected by a single source, and alert on it. Administrators and analysts use alerts to determine if a given situation requires further investigation or not.

Kevin J. Schmidt is a senior software developer at SecureWorks, Inc. (secureworks.com), an Atlanta, Georgia based MSSP. He is a member of a dedicated software team who take security, threat analysis and correlation very seriously. This team provides software tools and systems which allows the Security Operation Center (SOC) to ensure SecureWorks' clients are well protected 24X7X365.

InfoSec World 2006
3 April-5 April 2006 – Disney's Coronado Spring Resort, Orlando, USA
www.misti.com

Infosecurity Europe 2006
25 April-27 April 2006 – Olympia, London, UK
www.infosec.co.uk

RSA Conference 2006
13 February-17 February 2006 – McEnery Convention Center, San Jose, CA, USA
2005.rsaconference.com/us/C4P06/

Black Hat Europe 2006 Briefings and Training
28 February-3 March 2006 – Grand Hotel Krasnapolsky, Amsterdam, the Netherlands
http://blackhat.com

LayerOne 2006
22 April-23 April 2006 – Pasadena Hilton, Los Angeles, California, USA
www.layerone.info/

InfoSeCon 2006
8 May-10 May 2006
Hotel Croatia, Dubrovnik, Croatia
www.infosecon.org

iTrust 2006
16 May-19 May 2006 – Piza, Italy
www.iit.cnr.it/iTrust2006/index.htm

Eurocrypt 2006
28 May-1 June 2006 – St. Petersburg, Russia
www.iacr.org/conferences/eurocrypt2006/

Looking back at computer security in 2005

By Mirko Zorz

What follows are some of the biggest events of 2005 with comments by:

- **Bruce Schneier - CTO of Counterpane Internet Security and acclaimed security technologist and author.**
- **Howard Schmidt - former Special Adviser for Cyberspace Security for the White House, was CSO of eBay and Microsoft.**
- **Dr. Gerhard Eschelbeck - CTO of Webroot, named one of Infoworld's 25 Most Influential CTO's in 2003 and 2004.**
- **Mikko H. Hyppönen - Chief Research Officer at F-Secure.**
- **Fyodor - acclaimed security researcher and author of nmap.**
- **Ira Winkler - author of "Spies Among Us".**

An increasing number of techniques and easier access to computer equipment enhances the knowledge of both the malicious users and the security professionals. However, it always seems that the "dark side" has much more free time on their hands since they tend to be ahead of the industry.

Windows users are fighting with all sorts of malware and security holes year after year. "I know it is popular to blame Microsoft for security woes, but they really deserve it this year! From remotely exploitable vulnerabilities in Windows core services like UPnP and MSDTC, to a barrage of severe IE vulnerabilities, Windows users were constantly under attack." said Fyodor. "Microsoft spends many marketing dollars touting their security, but they need to start backing this up with action." he added.

The media tends to spread FUD by writing stories where large percentages of Internet users are very afraid to shop online, we see exceptionally big numbers when it comes to identity theft and yet e-commerce is booming and everyone and their mother are getting gifts for the holidays online. The truth is always somewhere in between - despite the media trying to publish "horror stories" in order to increase readership.

When it comes to all these reports where we see average users very paranoid Ira Winkler has another view on the situation: "As time goes on, people will only be more comfortable with computers. They will use it for more and more applications. Security is at best an afterthought, and the more ubiquitous the computer becomes, the less they will consider the threats involved with its usage."

Every year analysts inform us that this year was the worst yet and that a bleak digital future awaits just around the corner. I tend to be skeptical about such predictions so I'm going to let you decide what to make of 2005. The events depicted in this article all left a mark on both the industry and the users. As repercussions go, some are evident and some will be seen in the upcoming months. All in all, it was an interesting year.

Not a great year for credit cards

CardSystems processed payments for multiple credit card companies. In May the company suffered the largest data security breach to date when around 40 million credit card numbers were stolen. The affected companies were MasterCard, Visa, American Express and Discover.

The problem was not only in the fact that the incident occurred in the first place but in the fact that CardSystems did not comply with the regulations that their customers had in place. Audits showed that they weren't as secure as they had to be. The result? Not surprisingly, even after complying to the demands of increased security the company was sold in October.

Bruce Schneier comments on this situation: "Every credit card company is terrified that people will reduce their credit card usage. They're worried that all of this press about stolen personal data, as well as actual identity theft and other types of credit card fraud, will scare shoppers off the Internet. They're worried about how their brands are perceived by the public. And they don't want some idiot company ruining their reputations by exposing 40 million cardholders to the risk of fraud."

Howard Schmidt said: "I think that anytime a breach of security of any size, especially one that contains consumer private information causes executives to ask "Can this happen to us and if so how do we fix it?" With the compliance issues taking a bigger role in corporate governance world wide I would expect this to continue to be a board room discussion which will increase security."

And just in time for the holidays, Guidance Software (a self-proclaimed leader in incident response and computer forensics) suffered a breach that will probably get a lot of people fired. The incident during which some 3,800 customer credit card numbers have been stolen, occurred on November 25th but wasn't discovered until December 7th. Did Guidance Software contact their customers immediately? No. In the age where even children use mobile phones, IM and e-mail, they chose to send out notices of the breaches via regular mail. Why? They claim people change e-mail addresses too frequently while the location of the offices stays the same. I guess they think these companies also change their phone numbers all the time. Even if they do, shouldn't they keep an up-to-date database with contact information?

To make things even worse, the company stored customer records in databases that were not encrypted and if that wasn't bad enough they also kept the three digit Card Value Verification (CVV) numbers despite the guidelines by MasterCard and Visa that prohibit the storage of the CVV numbers after a transaction and require the databases to be encrypted. The company says they didn't know these numbers were stored for a longer period of time. I don't know if this makes things better or worse.

Rootkits go mainstream

On October 31st Mark Russinovich posted an entry on his blog entitled "Sony, Rootkits and Digital Rights Management Gone Too Far" that sparked a media frenzy. Russinovich discovered that Sony was using a rootkit as a method of control for some of their CDs.

Sony got under much fire as both privacy advocates and the users were raging against such vile control actions and started boycotting certain Sony titles, bad reviews were starting to show up on shopping sites and Amazon.com contacted their customers and offered them a complete refund if they returned the "infected" CDs. At least now the public is much more aware of certain problems.

Assorted malware

Not surprisingly this year had thousands of pages filled with reports of various types of malware wrecking havoc. So, are things getting any better or just worse when it comes to virus outbreaks? "It seems better. In 2003 we had tons of large outbreaks. In 2004 we saw some. This year only a handful." says Mikko H. Hyppönen. "However, the transformation from hobbyist virus writers to professionals also means more targeted attacks. These stay under the radar and don't become front page news - the criminals don't want to end up on the front page. We're seeing less outbreaks - so the situation seems to be getting better. It's actually getting worse" he adds.

The most talked about virus of 2005 is certainly Sober which caused a lot of problems and disrupted e-mail traffic for both MSN and Hotmail. F-Secure cracked the code and learned how Sober activates. More than 20 variants of the virus have been found since October.

Other "popular" viruses in 2005 were Zafi.D and several variants of Zotob. When it comes to numbers, Hyppönen says the situation seems better: "All of these cases were smaller than cases like the Mydoom/Bagle/Netsky war or the Sasser outbreak from 2004."

Is there any hope in sight for 2006? "We're afraid of several things. Automatic mobile phone viruses. WLAN viruses. Skype viruses. I'm afraid it's not going to get better" according to Hyppönen.

Ciscogate

A lot of media attention was on the Black Hat Conference in Las Vegas this year. Michael Lynn, a researcher working for ISS, did a presentation on a security hole in Cisco's IOS. Since Cisco threatened to shut down the conference Lynn first resigned from his position at Internet Security Systems but wouldn't back down from the presentation. What was a sad example of bad PR is everything that Cisco did. They instructed the people behind the conference to get the promotional material and rip out the pages containing the slides of Lynn's presentation. So 1984 of them.

Cisco claims the presentation was dangerous since it contains information on IOS and that the information was obtained illegally. Lynn found the problem while working for ISS under specific instructions to reverse-engineer the Cisco operating system. He noted that the release of information was necessary since the IOS source code was already stolen earlier and it was only a matter of time before someone decided to engage in some illegal activity. To get his perspective on things I suggest you read this interview at Wired (elfURL.com/141z).

I'm positive that if they hadn't made all this noise, much less interest would have surrounded this presentation. Immediately after the conference Cisco released a patch for the IOS vulnerability. Lynn was hired by Juniper Networks in November.

Common Vulnerability Scoring System allows IT managers to create a single standardized and prioritized ranking of security vulnerabilities across multiple vendors and platforms.

Common Vulnerability Scoring System (CVSS)

The issues surrounding the scoring of vulnerabilities got a possible solution this year with the creation of the CVSS (first.org/cvss/). Gerhard Eschelbeck said: "CVSS allows IT managers to create a single standardized and prioritized ranking of security vulnerabilities across multiple vendors and platforms. CVSS is relevant in all stages of the vulnerability lifecycle, from the time a vulnerability is identified by a researcher to the time a vulnerability needs patching within an enterprise. For computing the vulnerability score, CVSS considers not only the technical aspects of a vulnerability, but also how widely a vulnerable technology is deployed within an enterprise. A multitude of vendors have indicated their commitment to support CVSS in their products, and enterprises are currently introducing CVSS into their environments. By utilizing this scoring system, organizations can patch critical issues quicker, spending less resources on low priority issues."

Phishing

This is the year when phishing stopped being confused with fishing and basically everyone knows what it means. Howard Schmidt comments: "I agree that the number of phishing scams is on the increase all indications are that LESS people are falling for the scams. In some cases the international law enforcement have made arrests of people who are running these scams which has proven that people can be caught and will be prosecuted. Also, MANY technology steps have been taken to reduce the likelihood one will even see the phishing emails. There was a period of time where some people were scared away from online commerce because of phishing but all indications that there is limited "if any" impact at all."

Opinions on top problems in 2005

The security related event that defined 2005

Fyodor: "I think the continued rise of botnets has been the year's greatest trend. The Honeynet Project has been researching these and identified more than 100 botnets containing at least 226,585 unique compromised hosts. Much of this excellent work was done by the German Honeynet Project, and we released a paper. In the months since then, we've seen several people arrested for running botnets of more than 100,000 machines each. Increasingly, they have been using these for extortion: threatening crippling distributed denial of service (DDoS) attacks unless companies pay up."

The biggest online security threats in 2005

Gerhard Eschelbeck: "The security research community as well as vendors identify and publish on average 40 new security vulnerabilities per week. These vulnerabilities provide a multitude of avenues for attack and originate from many different areas. Incorrectly configured systems, unchanged default passwords, product flaws, or missing security patches are among the most

typical causes. Security vulnerabilities linger and consequently create a breeding ground for attacks, leading to security breaches. Improperly patched systems not only endanger themselves, but also put other users at risk.

It is not the security holes and vulnerabilities we know about and can respond to that are the biggest concern – it is the security holes and vulnerabilities we do not know about and will be the target of tomorrow."

Final thoughts

Was it worse than 2004? Better? Or did it just evolve to what you expected a year ago? It depends on how you look at it, how much influence a certain event had on your job, on your home computer or on your neighbor that just won't patch his machine and you have to help every weekend. We all rate the importance of an event based on how it affected us. The industry will take care of itself. Its revenue has been rising every year and you can look at it like this - more incidents, more compliance or both.

Mirko Zorz is the Chief Editor of (IN)SECURE Magazine and Help Net Security (www.net-security.org).

Writing an enterprise handheld security policy

By Jon Read, Seth Fogie and Cyrus Peikari

Most IT administrators will agree that writing a security policy is like passing a kidney stone. It's a glacially slow, excruciating process. And it's even worse when it comes to dealing with handheld devices such as personal digital assistants (PDAs) and Smartphones. Unlike legacy platforms, we still haven't found what the threats are to these powerful new devices. Worse, these tiny, embedded operating systems currently have little or no native security.

We have recently been helping a major telecom carrier in the USA create an enterprise security policy for Windows Mobile 5.0 devices. But the lessons learned can apply broadly to other handheld platforms.

This paper draws on our experience in the field, but you should recognize our limitations, too. You know your own systems better than we ever will. We hope this template can help give you a head start.

Note that if you are planning to roll out PDAs across the enterprise, you will first need to invest in 3rd party management software.

95% of the large customers that we deal with use Intellisync. SOTI has a new program out as well (Mobicontrol), but they have not yet returned any of our requests for evaluation. Still, we respect their programming skills and we recommend you check them out.

We will begin with a general background, and will then provide you with the skeleton of a sample template for your organization.

What is a Security Policy and why are they needed

A security policy can be defined as specific rules or laws that are put in place by an organization's management to ensure the security of information. A security policy is essential for any organization from large commercial companies to small non-profit groups.

Most countries have laws that concern the privacy of its individual citizens; these laws are a factor in security policies. Other factors are the commercial sensitivity of information, national security and personal security. Good security policies address all these factors.

Security policies are also essential for insurance purposes. Large insurers expect that a company will have done its very best to avoid an incident. Security policies are necessary to prove this, in the same way insurance companies expect you to lock your house when you leave.

The ultimate responsibility lies with the CEO of any company. Her job is to ensure that the security policy is written to a professional standard.

The CEO does not have to write the policy, but she does have to ensure the quality of the policy. When the policy fails a company, it is ultimately the CEO who will shoulder the responsibility. The chief information officer should also have a great deal of input into the policy as it is his or her job to be fully conversant with the information systems of the organization.

Policies require planning and research before they can be implemented. When researching what type of policy will be implemented in your organization it is important to consider the following factors:

• Threats: What are the potential threats faced by the enterprise or mobile field you want to protect? A threat is something that has the potential to damage. It can be an activity by a user or an event such as a virus infection (or even an earthquake).

• Vulnerabilities: Does the mobile field you are trying to protect have flaws that would allow an internal or external attack? Does the operating system frequently crash? Vulnerabilities pose a major threat. Systems with very policies are only as strong as their weakest flaw. It is important that you fix all known vulnerabilities before implementing policies.

• Risk: A risk can be described as the likelihood of a threat occurring and the damage sustained by an individual or company, financial or otherwise, from the occurrence of that threat. It is important to differentiate between likely threats and unlikely threats.

Mobile devices have revolutionized that way that people do business. Mobile phones in particular are becoming increasingly common, not only for business users but for the ordinary householder. With the increased advances in technology also come the increased risks of threats to the information that is stored and or passes through these devices.

With the emergence of malware for mobile devices, companies and organizations are now being forced to adequately implement Mobile Device / PDA security policies. Effective policies will ensure that the confidentiality, integrity and availability of every individual device in the enterprise are maintained.

Implementation: What needs to be secured?

First and foremost the actual device needs to be secured physically. Theft and loss play a major factor in mobile device security breaches. Employee carelessness needs to be addressed along with inventory control. The following URL has some useful information on Linux-specific tools that can help protect against loss and theft - tuxmobil.org/stolen_laptops.html

Replacing a stolen or lost device is not cheap; more importantly, the data on a device may be irreplaceable, and security policy needs to reflect this. The small size of devices makes them easy targets for thieves. While it may be hard for a thief to stroll into an office and steal a desktop pc, a Pocket PC can easily be removed without much effort. Misplacing a device is also relatively easy; Security company PointSec conducted research into this and came up with astounding results. In Chicago (United States), during the 6 months that the study was conducted, 85,619 mobile phones, 21,460 PDAs/Pocket PCs, and 4,425 laptops where left in the back of taxis.

Employees need to be held responsible for their device. Accidents can and do happen, but loss can be minimized when staff are encouraged to take ownership of their device. Stickers placed on the outside of the device with the owner's details can help the police or finder of a lost device to contact the owner. If a device has proper security measures in place this may be the only way for someone to find out who owns the device.

Proper physical storage of an unused device needs to also be addressed. This may seem a trivial issue, but don't take it for granted that an employee will know how to look after a device. Simple measures can extend the life of your mobile device inventory.

Data storage media needs to me secured. This embraces all types of storage, either removable or other. SD cards have the ability to hold gigabytes worth of data and they are the size of a postage stamp. The data on these devices is also vulnerable to damage by heat, RFI and electro magnetic damage. This needs to be addressed.

Protocols for data transfer need to be secured. Packet sniffing wifi transmissions can provide invaluable information to a hacker or to a rival company.

How it needs to be secured: Access control and Authentication

The use of access control mechanisms among your mobile device field is essential to prevent unauthorized access of data. Access control needs to be strong and well tested. The device on a data is only as safe as the ability of someone to access it. If strong access control software is used the data remains relatively safe.

Authentication processes need to be easy to use and not time consuming. End users are not interested in security; they only want to be able to access their work easily. If the process is time consuming or difficult you may find that employee's will look for ways of disabling the access control mechanisms, thus defeating the whole purpose.

Consideration should also be made to the pre-authentication state of the device. Some devices do have debugging features that can be accessed by manufacturer codes. A lot of mobile phones have these debugging features hard coded into the device. Thieves can use these codes to by-pass authentication measures and gain access to the protected data. Essentially this is a backdoor to the data stored on the device.

If you are worried that your device may have some sort of debugging feature it may be wise to contact the manufacturer, or even try to reverse engineer it yourself.

Authentication passwords should be changed regularly. Staff should be made aware of password security issues including social engineering.

Encryption

Even if an attacker manages to break an access control system, properly encrypted data will protect it from being of any use. As with access control mechanisms, it is important to insure that the encryption process is not difficult for the end user to use. Passwords and passphrases should be kept separate from the device; staff training will help ensure this.

Some security companies unwisely place hidden backdoors into their encryption software. Thoroughly research the software you intend to use. Do not take the software company's word at face value: read newsgroups and forums of actual user impressions. Backdoored software serves no purpose to your organization; it may help in instances of lost passwords, but it will also help hackers attain information and data.

Countries such as the USA also have rules that pertain to the allowable encryption strength companies are allowed to export. It is best to research your options before you purchase; you might be able to find stronger encryption solutions from less restrictive countries. One important factor to remember is that buying software from companies that are not well established may not be advisable. If a company goes under (bankrupt) it may be impossible to get further upgrades or technical support for your software.

Firewall and IDS protection

Just like a desktop machine, mobile devices are also vulnerable to remote attacks and intrusions. A proper firewall will protect not only the end users device, but also will protect any corporate network to which this device has access. Hackers realize that a lot of mobile devices do not have firewalls. An unprotected device is literally a backdoor into your network.

Firewall settings need to be easily administrated and mandatory settings need to be fully transparent. It may not be a good choice to allow employees to access the firewall settings. Employees who find the firewall hinders them may disable it or remove it altogether. It is also preferable that firewall rulesets be updated regularly to include newly discovered trojan ports and worm-vulnerable ports into the database until a fix is ready.

IDS or intrusion detection system's do not offer the literal blocking that firewalls provide; they do, however, alert a user to an intrusion attempt. Intrusion detection systems monitor file changes and security breaches: these are logged and can be reviewed by administrators.

IDS software is not common on mobile devices, but there are a few products around. Some antivirus software also have IDS type functions and a good IDS can help an administrator when it comes to advanced debugging of a device.

Antivirus and Malware protection

Mobile devices are actively being targeted by malware coders. The range of software available for mobile devices varies but software should be easy to use and not interfere with the end users' work. Mandatory settings should be enforced so that end users cannot disable the antivirus software. The software should also be easily updateable preferably over the air. Malware protection should also include protection against web based malicious scripting. Some mobile devices are vulnerable to these attacks.

Antivirus software should not try and detect malware that does not target mobile devices. It is not sensible to detect desktop PC malware on a mobile device as these can not damage a mobile device. The company's desktop PCs are protected with their own antivirus software. Scanning for non-specific malware is a giant drain on resources of embedded devices.

Active scanning ("real time scanning") for malicious files is a way to distinguish a good mobile antivirus program from one that is mediocre. If the software offers this benefit then it will probably provide better security then "scan on demand" type programs.

Data Erasure

Data erasure software is required for all devices that handle confidential information. This software should at least meet the US Department of Defense guidelines as outlined at the following URL - www.dss.mil/isec/nispom_0195.htm

This type of security will protect old data from being retrieved, which is essential if your organization sells its old devices to a third party.

Data erasure should also be controlled so that end users do not accidentally wipe important information. This can be addressed with restrictive settings as well as with staff training. Software should also ask a user at least once if they really want to wipe the information.

When purchasing data erasure software, make sure that the software lives up to the manufacturer's claims. You should test it with proper forensic software to ensure that information or data is not leaked.

Proper Synchronization Controls

When a device is synchronized with a workplace computer it is vital that measures are put in place to stop it from retrieving confidential business documents. Normal synchronization software will allow the device to grab any new documents without checking if they are confidential or not.

The problem occurs when an employee takes his or her device home and perhaps syncs it with their home computer. The confidential documents are now transferred to the user's home machine, which may not have the security mechanisms in place that the organizations computers do. It is wise to restrict the synchronization ability of devices, and perhaps to not allow them to synchronize at all.

Automatic, silent wifi synchronization should be disabled. If a device mistakenly synchronizes with the wrong host computer, all sorts of problems could ensue. In general, this is not a major problem, but for security reasons it is best to have this disabled by default.

Staff Training

Staff training is an essential aspect of all good policy implementation. Make your training experiences enjoyable; don't overload the staff with highly technical jargon. A system of friendly reminders such as slogans printed on coffee cups is also an advantage. You walk a fine line, so be careful not to go overboard as staff members may switch off and become complacent.

Proper staff training is also the best insurance against employee's pleading ignorance to security measures. If that employee has been shown what to do, then the onus is on them in the event of a security breach or incident.

Strict policy guidelines should be made clearly available to staff. Policies should be enforced using a system of warnings. If a staff member received too many warnings against his or her name then it is probably time to let them go.

Mentoring systems can help new staff adjust to an organizations security policy. New staff can be teamed up with responsible older staff from whom they can learn. It may be a good idea to rotate mentors so the new staff member does not pick up bad habits.

Staff should be rewarded for good practice. Pay increases may not be financially viable, but it doesn't cost anything to tell an employee that you appreciate their adherence to company security policy. People like to be praised and happy employees are generally more likely to be security-conscious.

Mandatory Device Settings

All devices in your mobile fleet should have mandatory security settings in place that are impossible to change by the average user. If the device does not need to be wifi aware, then this should be disabled, along with Bluetooth and infrared. Viruses such as Cabir exploit Bluetooth and simply disabling this feature will provide protection.

If your employees have no reason to play media files such as mp3's, mpg's, etc. it is recommended that media players be removed from the device. This not only stops the device being used as an expensive mp3 player it also protects your organization from potential legal problems often surrounding these types of media files.

Malware has been found in pirated software for mobile devices, especially software for Symbian mobile phones. It is imperative that employees are fully educated on the risk involved in installing this type of software. As with mp3s, there is also a legal aspect as to why these files should be avoided. Mandatory device settings can stop unauthorized software from being installed.

Mass Device Management

You will find many tools on the market that allow for mass management of devices. This can allow the administration to implement security updates, as changes to the organizations policy are made. These tools can also push out virus definition updates for antivirus software.

Mass device management ensures that all devices are homogenized, and they protect a network from rogue devices that may have been modified by staff members (e.g. devices with pirated games installed, etc). Management software can also allow administration to remotely control a device viewing its screen on a desktop PC. This type of administration is invasive but protection of company assets should be a main priority.

The following gives a sample template for you to start writing your own PDA security policy.

All devices in your mobile fleet should have mandatory security settings in place that are impossible to change by the average user.

Sample PDA security policy template

A. Purpose

This document outlines the accepted use policy related to personal digital assistant (PDA) devices. Any existing user or future user of a PDA should acknowledge their understand of this document and that they will abide by its content.

B. Definition of PDA

A Personal Digital Assistant is any device that provides a mobile handheld computing platform. In general, these devices contain the ability to keep track of tasks, contact information, provide internet and email access, in addition to various other features, such as games, music, time keeping, and more. Common devices include, but are not limited to, the Palm, Treo, Blackberry, Compaq iPaq and Dell Axim. Some PDAs are merged with other mobile devices, such as a GPS or Cell phone, which are also considered applicable for this policy.

C. Failure to Comply with Policy

Upon the first violation of this policy, the PDA owner will be given a written warning of their infraction, and be required to read and acknowledge the PDA policy by signing a copy of the policy, which will be placed in their administrative record. If a second violation occurs, the owner of the PDA will be given a second written warning, and asked to remove the PDA from company property, or if the PDA is company property, then the device will be taken from the violator. Any further violations of this policy will result in termination. Managerial approval is required for any second time violator to be permitted to use a PDA on company property.

D. Policy Guidelines

The following section will outline the acceptable use guidelines that apply to PDA devices. This includes synchronization procedure, data storage and encryption, network use, authentication/ identification measures, and loss of control guidelines.

1. *Data Storage* - A PDA can store data in persistent memory, external storage (ie. Compact Flash), or internal RAM. Each of these types of memory present a security challenge that will be covered in this section

a. Internal RAM: All sensitive data in use must be stored in this part of the memory while decrypted.

D. Policy Guidelines

The following section will outline the acceptable use guidelines that apply to PDA devices. This includes synchronization procedure, data storage and encryption, network use, authentication/identification measures, and loss of control guidelines.

1. *Data Storage* - A PDA can store data in persistent memory, external storage (ie. Compact Flash), or internal RAM. Each of these types of memory present a security challenge that will be covered in this section

a. Internal RAM: All sensitive data in use must be stored in this part of the memory while decrypted.

b. Persistent Memory: This part of memory maintains it status in the even of a loss of power, which eliminates the need for reinstallation if there is a loss of power. PDA owners can install their applications in this section of memory, as long as no sensitive data is stored in the program directories.

c. External Memory: External memory is only to be used for data that has no security risk (ie. Music). If data is encrypted according to policy requirements, this section of memory can be used to store that information, this includes encrypted backup files.

2. *Network Use (Email/Internet/Etc.)* - The PDA may be used to access web sites provided the content is not a security risk. Email may be downloaded to the PDA over a security link. The security link should consist of a VPN connection to the company network, or via an SSL protected connection to the secure website. Use of public hotpots is acceptable only for generic web browsing.

3. *Authentication/Identification* - Each PDA device must have an alphanumeric login enabled. The password must contain letters and numbers. In addition, there must be a check in place to prevent brute force password guessing.

4. *Loss of Control (Lost or stolen device)* - In the even that a PDA is lost or stolen, the owner must immediately contact the IT department and report the incident. An inventory of programs and data must also be included with the report.

5. *Third Party Applications* - Only approved third party applications can be installed on PDA's. The approved list can be obtained by contacting your IT department. If there is a desired program that is not on the list, a request can be submitted to the IT department. If the program meets internal testing requirements (stability/security), it will be added and at that point it can be installed.

6. *Synchronization* - Synchronization of the PDA to the host PC can only occur locally or via a secure connection to the company.

E. 3rd party security software

1. *VPN* - A VPN is required to connect from the handheld to the corporate environment from outside. This includes remote email, remote sync, etc.

2. *Encryption* - Encryption software should conform to currently accepted, strong cryptographic algorithms

3. *Data Wiping* - Any encryption program should likewise include a security file-overwriter/data wiper. This secure bit-overwriting must conform to DoD standards (up to seven passes of secure file overwriting) and must allow wiping of internal and external memory cards as well.

4. *No hard reset code* - Security software will not implement remote hard reset ability. Bit wiping of RAM with a remote hard reset is now considered an obsolete security practice, and may actually increase the danger from worm attacks.

5. *Firewall* - A host-based personal firewall is mandatory on all PDAs.

6. *Antivirus* - Virus scanning programs, updated regularly, are required on all PDAs.

7. *Remote security management* - Security software must provide central control policy that allows the administrator to set features such as frequency of virus signature updates, remote change of firewall rules, enforcing password strength, etc.

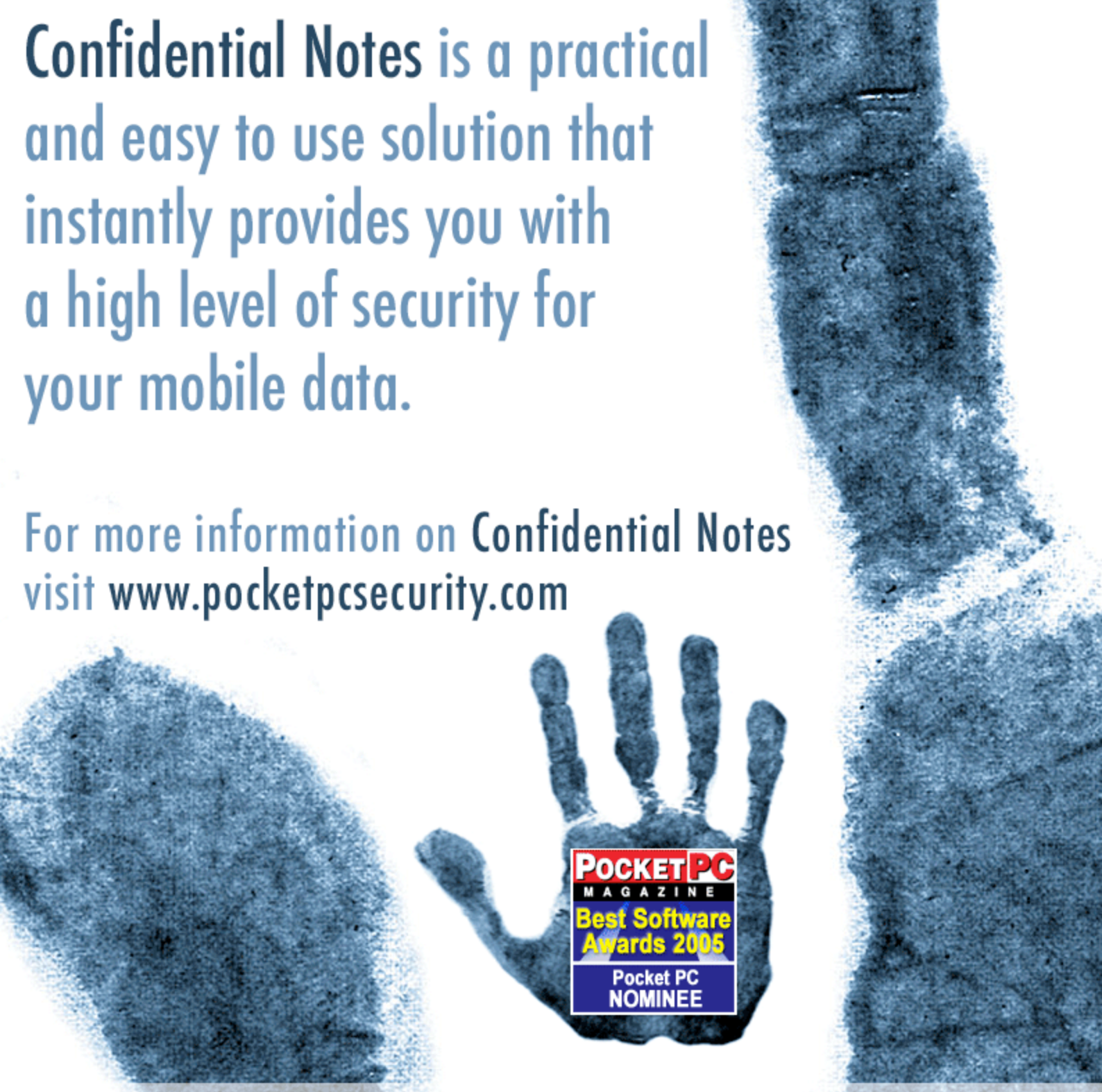
This paper completes our rather long, three-part series on handheld security written for (IN)SECURE Magazine.

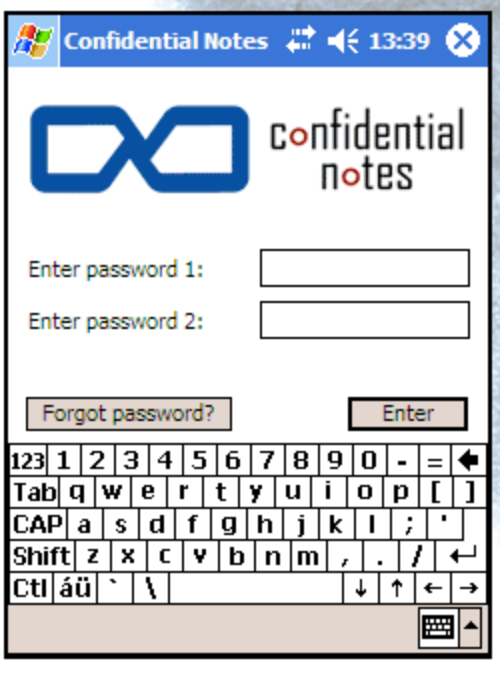
The field is evolving rapidly so we hope you will keep up with our blogs and news releases at **www.airscanner.com**.

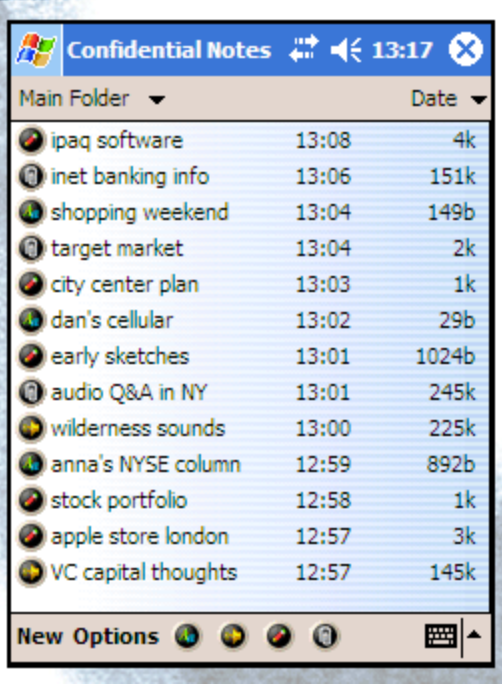
Jon Read, CISSP, Seth Fogie, and Cyrus Peikari are members of the Airscanner Mobile Security Team. They focus on exploring security threats and on reverse engineering malware for embedded and handheld wireless platforms.

Confidential Notes is a practical and easy to use solution that instantly provides you with a high level of security for your mobile data.

For more information on Confidential Notes visit www.pocketpcsecurity.com







Access control using authentication and authorization works well for limiting how people use digital resources in a controlled environment, such as the corporate network. But traditional access control schemes do not work as well when the people or resources are outside of the organization's direct control.

Documents released under non-disclosure agreements illustrate this problem. Once the document has been released to someone outside your organization, that person could make unlimited copies, send the document to your competitor, and so on. Encrypting or password protecting the document does little to deter this unwanted behavior, because the person receiving the document must unlock it to use it. The authorization schemes we've discussed don't address the problem either, because access control depends on a trusted environment. Absent another solution, we're left with trust and legal remedies.

Digital rights management (DRM) is an attempt to address these problems. Rather than merely controlling the actions that an entity can perform on digital resources, DRM provides mechanisms for controlling the particular uses to which a digital resource can be put. This is a tough problem, and as we'll see, good solutions are sufficiently draconian that they impose a significant burden on users and have raised the ire of digital rights activists.

Digital Leakage

Digital leakage is the loss, whether intentional or inadvertent, of confidential data in digital form. The loss might take the form of a trade secret sent to a competitor, the premature release of financial data to an analyst or market, or the leak of embarrassing information to the media. Digital leakage occurs from seven primary sources:

- Employees sometimes steal valuable confidential information for personal use or to sell.
- Confidential information is sometimes accidentally distributed. This can happen when an email containing confidential data is addressed to the wrong person.
- Computer theft and hacking results in the release of confidential data despite the best efforts of computer security professionals.
- Employees, partners, and customers often do not understand the real value of information that your organization has shared with them and do not adequately protect it.
- Changing alliances result when relationships between the organization and employees, partners, and customers end, leaving these entities in possession of information to which they are no longer entitled.
- Lost or stolen devices can result in the loss of information more valuable than the device itself. Often, companies sell used computers that contain confidential data.
- Disgruntled employees and others may maliciously redistribute or otherwise release confidential information.

Digital leakage is costly. A survey published by PricewaterhouseCoopers and the American Society for in Industrial Security in 1999 found that on average, large organizations lost confidential or proprietary information 2.45 times in a year and each incident cost an average of $500,000. The survey estimated that the cost of digital leakage in the Fortune 1000 in a single year was $45 billion.

The DRM Battle

With those kinds of statistics, you'd think DRM would be a technology that everyone could love, but it has been at the heart of some of the most acrimonious debates of the digital age. The movie and recording industries are worried that electronic distribution of their works will result in violations of their copyrights and, as a result, diminish their bottom line.

This is a classical problem for digital rights management. The producers of the digital goods want to release them to people beyond their control and give them only specific rights (e.g., to listen to, but not copy, the music).

The problem is that the needs of the copyright holders are in direct conflict with the wants of their customers. Consumers of movies and songs want open access, open formats, and access to works no longer for sale in traditional distribution channels. Napster illustrated the powerful drivers in this market. People want to be able to share music and movies with others.

Needless to say, this is a complex issue. The reason for bringing it up here is that the battle over copying movies and music has colored many people's view of DRM and created an atmosphere where any discussion of DRM creates strong feelings. DRM might be the right technology for solving critical access-control problems for your organization's digital resources. Unfortunately, DRM has become synonymous with the battle over copyrighted music and movies. You can probably avoid the DRM battle and the emotions it engenders if your motive is to control activities on digital resources rather than to use DRM as part of a business plan that restricts customer actions.

Apple iTunes: A Case Study in DRM

Apple's iTunes and its associated music store provide a real-world example of DRM in action. iTunes will play unprotected MP3 format audio files, but when a user purchases music form the Apple music store, the audio file is downloaded in a format called AAC. Apple wraps the AAC file in a DRM system called Fairplay. The standard rights allow a purchaser to listen to the song in an unlimited way on up to three computers and to burn the song to a CD.

This set of rights was chosen to try to match the value that user's place on the audio file to the price Apple wanted to charge. For example, Apple could disallow burning AAC format songs to CD, because they control the client, but that would decrease the value of the file for many people.

Apple also has to be able to administer rights remotely to provide customer service. For example, if I purchased a song, installed it on three computers, and then sold one of those computers and bought another, I can contact Apple to have the rights reset on my music collection, allowing it to be installed on my new computer. Without this ability, audio files purchased on iTunes would quickly lose their value as people upgraded their computers.

This case study is a good example of the additional burden placed on a company in controlling access to content using DRM. Restricting rights for content costs real money, because the content has to be administered and it reduces the value of the content to users.

Features of DRM

Digital rights management is, of course, about more than just protecting music and movies. DRM is a technology all of us would like to use in certain circumstances. For example, when I send my Social Security Number to my bank, I'd like to be able to control how it is used. As another example, wouldn't it be nice to be able to send your credit card number to an online merchant and attach specific rights to it: the right to use it to facilitate a single purchase and not be stored or transferred. All of us have digital information that we'd like to be able to send to someone else without giving them unlimited rights.

SealedMedia, a DRM vendor, lists some important features that DRM systems should have to be effective.

- Persistent security, wherever the digital resource exists.
- Separation of right of access from the information itself.
- Management of discrete rights (viewing, printing, editing, print-screen).
- Dynamic allocation and withdrawal of rights.
- Support for both online and offline work.
- Audit trail of actions and activities performed on the document.

• Support for multiple common document formats using the same security tools.
• Simple integration with existing workflow and applications.
• Integration with document/content management systems.

A perfect DRM system with all of these features does not exist. You will likely have to prioritize these properties for your particular application and evaluate solutions on that basis. The next section describes a reference architecture that shows how some of these features can be accommodated.

DRM Reference Architecture

Figure 1 shows a reference architecture for a digital rights management system. The reference architecture is by Bill Rosenblatt, et al., and is discussed in some detail in the book *Digital Rights Management: Business and Technology* (John Wiley & Sons). The reference architecture points out the key interactions in a DRM system and also exposes some of the weaknesses in current DRM schemes.

In Figure 1, there are three primary participants. The client is an application invoked on behalf of the entity wanting to access and use a digital resource. The content server is the application that is invoked on behalf of the entity supplying the digital resource. The license server is the application invoked on behalf of the person who owns or controls the rights associated with the good. The license server and content server might be operated by the same entity or they might not. For example, the owner of the goods may contract with multiple distributors to supply the good but control the licensing centrally.

In the reference architecture, the client, on behalf of the user, requests a specific resource from the content server. The content server uses a content repository along with product information to create a content package. The content repository might be part of the content server itself or a standalone content management system. The product information specifies price and other product-specific information. It makes sense to separate the product information from the content, because the same content may be sold as different products differentiated by customer class, additional services or warranties, and so on.

The content is delivered in an encrypted content package that contains both the content and metadata. Whether the content is streamed or delivered as a single package is inconsequential to our discussion. The metadata usually includes information such as title, author, rights holder, and other information about the content as well as information that uniquely identifies this content and transaction.

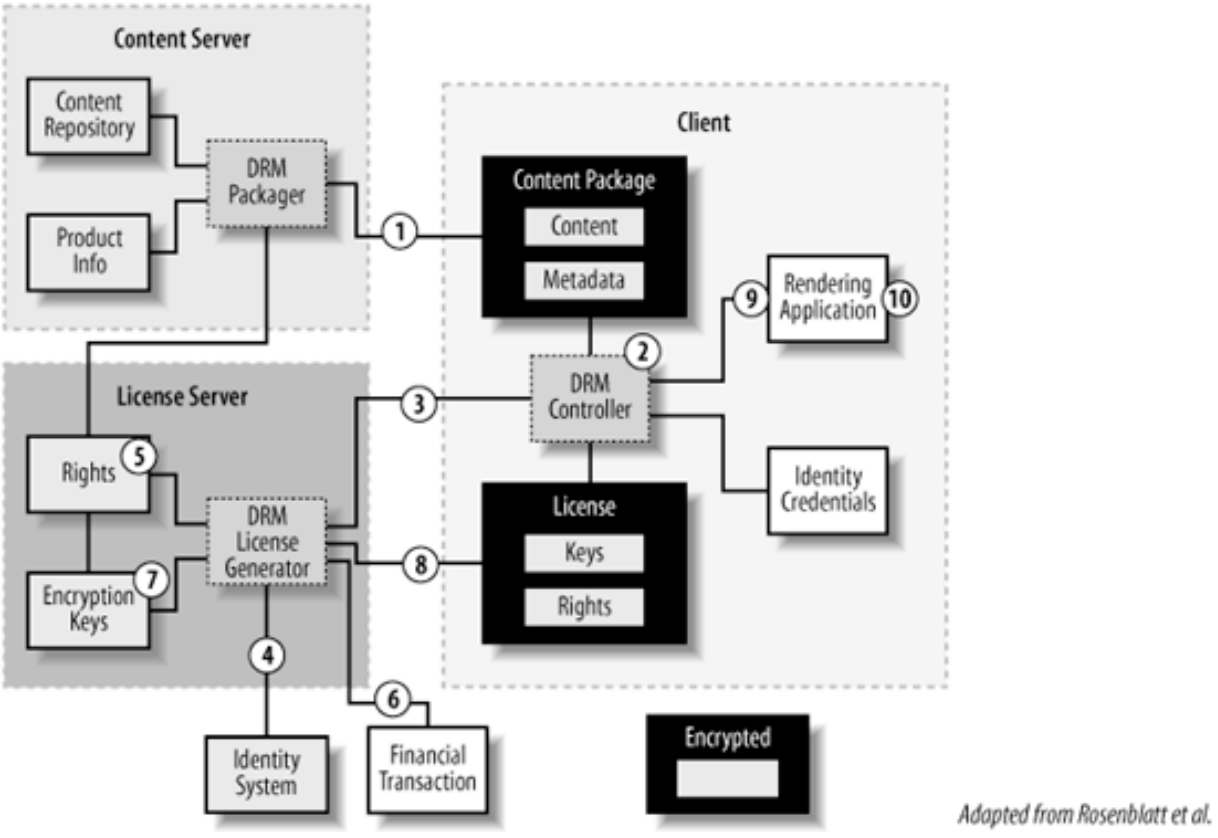


Figure 1. A DRM reference architecture

The DRM controller (2) contacts the license server (3) associated with the content package to retrieve a license for the content. The DRM controller sends the license server information about the content package and the identity credentials for the entity that invoked it.

The license server checks the identity credentials (4) using whatever identity infrastructure is available to it for authentication and then consults a rights database (5) for authorization. We'll see an example of an XML language for expressing rights later in this chapter.

There may be a financial transaction (6) if the user is required to pay for the license. Alternately, the license server may require some other consideration such as registering and providing contact information. Once consideration for the license has been received, encryption keys (7) are retrieved based on the content package identity and these keys are used to generate a license that is sent back to the DRM controller (8) as an encrypted package containing a statement of rights and a set of keys for unlocking the content package. The license is encrypted to protect the keys used to unlock the content package and to ensure its integrity. The DRM controller decrypts the license and uses the keys contained in it to unlock the content. The content can be sent to a rendering application (9) for display (10).

The client, in managing rights, may store information about usage in the content package. Usage data is stored with the package, rather than separately, to ensure that even if the content package and license are moved to another client, the usage restrictions are honored and audit trails are continuous.

The client application is a critical component in this scheme, because the client application, rather than the client, receives and processes the keys. This overcomes, in part, the problem of a user unlocking the digital content and then using it in an uncontrolled fashion, because the client application can ensure the permissions carried in the license are honored.

Trusted Computing Platforms

If you were applying a little creative thinking during the preceding discussion of the DRM reference architecture, you probably thought of several ways that the scheme could be defeated. That issue is the chief weakness of DRM. As we've seen, for the user to view or otherwise use the content, it has to be rendered in a usable format, and that allows ample opportunity for the content to be redirected to a use that wasn't specifically authorized.

The iTunes example illustrates some of the problems:

- Once an audio file has been put on a CD, it can be ripped in another format, such as MP3, without any DRM, because the DRM wrapper can't be transferred to the CD.
- Remote rights administration, needed for customer service, opens up further opportunities for people to exploit the system and get around DRM.
- The Fairplay system has been cracked, and methods for playing Fairplay-protected files outside of iTunes have been published on the Internet.
- Even if all of these problems were solved, the analog feed going to the speakers could always be redirected to a recording device, and the audio file could be re-encoded in another format.

These examples show just how hard it is to really protect content in a digital format. In addition to the tradeoffs made by Apple that were examined in the case study, Apple is inconveniencing their legitimate users while still allowing the rights of copyright holders to be undermined by determined crackers.

These problems have led to numerous calls for trusted computing platforms that would ensure that the DRM client was run in an environment that kept even determined attackers from gaining access to protected content illegitimately. The basic idea is to protect every component of the end-user system in a way that disallows illegitimate use. When we say "every component," we're being literal - right down to the keyboard. Building a trusted computing platform requires the cooperation and coordination of both hardware and software manufacturers in very sophisticated way.

The nature of trusted computing systems and the debate surrounding them is beyond the scope of this article, but they are being advanced by companies as powerful as Microsoft and Intel and countered by numerous user advocacy groups. Because trusted computing platforms are still in the future, DRM will remain an exercise in making the theft of unauthorized rights sufficiently inconvenient that most users will only access content legitimately.

Specifying Rights

One of the most important features of a DRM system is the ability to specify and manage rights. Rights are a special kind of authorization.

The differences lie in the fact that DRM is meant to restrict actions on a much finer-grained scale than we typically deal with in a standard authorization system. Authorization rights typically center around whether a subject is allowed to read, modify, or create objects. As we saw, we usually specify the rights for classes of users against classes of objects in order to make the task manageable. In DRM, we often want to give specific rights (for example, the right to view but not copy) to specific people (Ted in accounting or a particular customer) for specific time periods (for the next two hours or three more times). That makes the task much more difficult. The problem can be made tractable by being able to build general licenses and derive specific licenses from them automatically.

Separating authorization rights from the objects being protected increases the ability of operators to take protective action. Specifically, when rights are associated with objects, removing rights for a particular user means visiting each object the user might have had rights to. The goal of systems like RBAC is to specify rights separately, so we can remove access rights across the board with a single, reliable action.

In DRM systems, the nature of the problem does not make this solution possible. Since our problem statement is to protect content that is not directly under our control, the rights will generally be sent outside the systems we directly control to some other system that will enforce them. Offline access to protected content is usually a requirement, and so it is not practical for the client application to check back with the license server each time the content is accessed. Thus, rights can be difficult or impossible to revoke once they've been issued.

XrML

There are several proprietary DRM systems and most of them have proprietary languages or systems for specifying rights. XrML is an XML-based rights management language. We'll discuss it briefly because it is gaining widespread acceptance as an open format for specifying rights and because it illuminates the kinds of features that you want in a rights language. XrML is a large standard, and this section is not intended to be a tutorial. More detailed information on the XrML standard can be found at xrml.org. The examples given here are based on the Example Use Cases document that accompanies the XrML 2.0 specification. The following simple example gives us a feel for XrML. In English, the license grants a specific RSA public-key holder the right to print the contents of an object identified by a URI as many times as she wants before Christmas day, 2005.

```
<license>
  <grant>
    <keyHolder>
      <info>
        <dsig:KeyValue>
          <dsig:RSAKeyValue>
            <dsig:Modulus>Fa7wo6NYf...</dsig:Modulus>
            <dsig:Exponent>AQABAA=  =</dsig:Exponent>
          </dsig:RSAKeyValue>
        </dsig:KeyValue>
      </info>
    </keyHolder>
    <cx:print/>
    <cx:digitalWork>
      <cx:locator>
        <nonSecureIndirect
           URI="http://www.contentguard.com/sampleBook.spd”/>
      </cx:locator>
    </cx:digitalWork>
    <validityInterval>
      <notAfter>2005-12-24T23:59:59</notAfter>
    </validityInterval>
  </grant>
</license>
```

You can see that the `<keyHolder/>` elements contain the user's key. The `<print/>` element gives the allowed action. The `<digitalWork/>` element identifies which resource the license applies to. The `<validityInterval/>` element specifies the interval for which the license is valid. This is an example of an XrML end-user license. More complicated license specifications are possible. As an example, suppose that PDQ Records wishes to allow university libraries to allow their

patrons to check out digital music. There are three types of rights that might be specified.

The first, very general type concerns one entity granting rights to a class of content to a class of entities. Here's an example:

PDQ Records allows university libraries to issue limited end-user licenses within certain parameters for any content they have purchased.

The second type is a policy specification. In a policy specification, an entity spells out specific rights that classes of users have regarding classes of content. Here's an example:

Brigham Young University will grant faculty the right to check out any PDQ Records song in its collection for up to six months. Student may checkout any PDQ Records songs in the BYU collection for three weeks. Anyone may play any PDQ Records song in the BYU collection on a library computer at any time.

The third, and most specific, type is an end-user rights license. Here's an example:

BYU grants Alice the right to play "When the Thistle Blooms" for three weeks.

Notice the hierarchy of rights contained in these examples. The first is very general and grants very broad rights to a class of entities. The rights specified in the second policy statement must fit within the rights granted in the first statement. Similarly, the rights granted in the third policy statement fall within the rights granted in the second statement, and hence those granted in the first as well.

In addition to specifying rights, this example assumes that Brigham Young University can uniquely identify itself and assert that it is a university in a way that is trusted by PDQ Records, and that Alice can uniquely identify herself and assert that she is a student in a way that BYU can trust. XrML can be used to specify each of these cases, although the XML documents for each are lengthy and not included in the interest of brevity. Interested readers are referred to the Example Use Cases document that accompanies the XrML specification.

Conclusion

The battle over DRM and many of the controversies surrounding it are unimportant to your organization, provided that your intent in using DRM is to control confidential and sensitive data rather than using it to control the actions of your customers. The key point to remember is that DRM is not usually effective against determined attacks. The more valuable the content, the more difficult it is to adequately protect. Consequently, the cost of DRM increases linearly with the value of the content. Because no DRM system is perfect, any DRM system should be carefully evaluated for its particular application and the trade-offs examined thoroughly.

WINDOWS - Deep Network Analyzer
http://www.net-security.org/software.php?id=646

DNA (Deep Network Analyzer) is an open, flexible, and extensible deep network analyzer server and software architecture for passively gathering and analyzing network packets, network sessions, and applications protocols.

LINUX - ProShield
http://www.net-security.org/software.php?id=282

ProShield is a security program for Debian Linux. It helps ensure your system is secure and up-to-date by checking many different aspects of your system. Regular use is recommended.

MAC OS X - Fugu
http://www.net-security.org/software.php?id=629

Fugu allows you to take advantage of SFTP's security without having to sacrifice the ease of use found in a GUI. Fugu also includes support for SCP file transfers, and the ability to create secure tunnels via SSH.

POCKET PC - AirFix
http://www.net-security.org/software.php?id=647

With the release of ActiveSync 4.0, you can no longer maintain a network connection while synced. According to Microsoft, they removed this useful feature because some corporate customers had thought it was a security risk. However, Airscanner believe that taking it out is a much greater risk. The result of this feature removal is that any ActiveSync session (via Bluetooth, IR, or USB) will immediately disable your network connection. AirFix will give the control back to you, which is where it belongs.

If you want your software title included in the HNS Software Database e-mail us at software@net-security.org

Revenge of the Web Mob

By Melisa Bleasdale

2005 was a banner year for high-profile big money security breaches. One violated company after another notified law enforcement while news trickled to the press in slow and reluctant batches. Called into question were management styles, internal process, security infrastructure, lack of vigilance, diligence and common sense. While companies struggle with break-ins, hijacks, and nearly invisible infiltration the ever expanding legions of would-be cyber thieves are gearing up for their next attack.

In recent years we've given excess weight to the criminal element – making them bigger, stronger and more intelligent than normal folk. In fact, research shows that today's cyber-crime is most often perpetrated by a far less threatening group - the everyman.

Once the domain of savvy hackers and progressive crime lords, online crime is now anybody's game. Online crime has quickly devolved from an elitist game of network infiltration to plug-and-play theft kits readily available to anyone that's interested. Ease and accessibility make the buying and selling of illicit information increasingly attractive.

Forward thinking criminals have long used the evolution of technology to their advantage, upgrading their nefarious activities in line with company upgrades of applications. Joining their ranks are those with time on their hands and the newfound means to dig up data.

Much like the magic pyramid, hackers, phishers, crackers and social engineers recruit others to help them build their wealth. The buying and selling of products is buoyed by the black market's supply and demand. If it's valuable, it can be taken, if it's taken, it's for sale. Yet even criminal consumers are a fickle bunch. What was popular last year is not so golden by this year's standards. According to Bindview's RAZOR research team - a group of people focused on incorporating the latest up-to-date changes in the threat, vulnerability, and regulatory landscape into Bindview's products - Credit card numbers that were worth approximately $25 each wholesale and $100 each retail in 2002 are now worth $1-$5 wholesale and $10-$25 retail. Where identities used to net a tidy profit, email addresses and letters of credit are gaining ground.

A fly-by-night hacker could easily procure tens of thousands of email addresses in one night of focused pharming and a well-programmed bot could find many more. Each email address netting between $.01 and $.05 cents per name adds up to a tidy sum by the end of the take.

The cyber criminal profile is rapidly shifting from the devious black hat to the enterprising capitalist.

"The ease with which data can be stolen depends on the tools being used and the thief's level of sophistication in traversing through the network," says Jim Hurley, Senior Director, RAZOR Research, for Houston, Texas-based Bindview, "Creating a breach ranges in difficulty from been intimately familiar with the innards of OS design,

construction and network protocols to having absolutely no knowledge - because you don't need it with the vast availability of pre-built tools."

"Sniffers, keyloggers, rootkits, loaders, Trojans and virus kits are but a few of the many offerings on thousands of accessible sites" he adds.

Tools of the Trade

According to a recent presentation by Hurley at the Computer Security Institute's 32nd Annual Conference in Washington D.C., Windows and Linux tools enable a vast amount of attacks using the Least Common Denominator (LCD) approach. A snapshot of the choices include:

Ping	Whois	Finger	Traceroute
Dig	Host	DNSwalk	Netstat
Procman	Portscan	NBscan	SNMPAudit
NMBlookup	Who	Route	Rsh
ARP	Rarp	Nmap	NDIS
Promiscuous mode		Rsh	Useradd/mod

Sniffers provide a whole different realm of opportunity with tools such as:

NetBus	Strobe	Msscan	SATAN	SAINT
SARA	Nmap	Super Scan	Stealth	Back Orifice
Win scan	Port Scan	Airsnort	Snort	Proxy Hunter
Snare	Honeyd	Nessus	ScanIP	Tcpdump
Windump	Whisker	IP Tracer	Kismet	FTP scanner

Keyloggers and crackers, often used by criminals with a more hearty technical background can include:

KLOG	BIOS cracker	CMOS killer
Home page worm mailer generators	MS Word cracker	VNC cracker
Linux W0rmxx.xn generators	Brutus	PwDUMP
IRC worm generators	SID tracer	SID dump
Firewalk - ACLs on network devices	Rainbow Tables/Crack	
l0pht	SID2USER	(USER2SID)
John the Ripper	Brute force	Chntpwd

Trojans are used by more sophisticated black marketers and readily available examples include:

Sub Seven	Sub7CRK	Sub7PC	Rat Cracker	Back Orifice
Silk Rope	Netbus	Moosoft	Admin Trojan	AcidSchivers

Boing	GRUB	VICE	Infiltrator	Deep Throat
Apache BD	Evil	Hack Attack	NetController	
SubRoot	Telnet Trojan	Donald Dick		

Determined criminal programmers have easy access to ready-made worms and rootkits waiting for their creative skill set to turn them loose. Examples include:

- VBS worm generators
- Home page worm mailer generators
- Linux W0rmxx.xn generators
- IRC worm generators
- Firewalk – determines ACLs on network devices
- FU
- Shadow Walker
- LRK 5
- Adore LKM
- Adore BSD
- Knark
- Root evil
- Adore MS

In the recent past online theft and criminal activity relating the Web poured forth from highly advanced or conversely, severely disadvantaged nations – but today's online crime is far from country specific. Just as online auctions launched a flurry of overnight entrepreneurs, so has the availability of criminal source kits.

Once a realm that depended on well-cultivated contacts between buyers and sellers, today's online black market requires little more than access to Web sites, bulletin boards, IM, email and cell phones. If its volume sales they're looking for, the first stop is the lucrative Web auction ring.

Well-organized Web Mobs run in a similar vein to organized crime families. They're efficient machines that include many layers of people performing very specific roles and functions. From the top down they include the inner ring, evaluators, inspectors, enforcers/contacts, trusted fences and the buyer and seller.

Web Mobs have proved to be a very nasty problem for Federal and international investigators due to their cross-country logistics and distributed operations. Once sufficient evidence has been gathered to crack an auction ring, authorities must work within international boundaries, time zones and with foreign legal statutes to make an arrest.

"What's not well known is the scope, intelligence and capabilities of these Web auction rings. They're not in the business of stealing things and they're not hackers," says Hurley, "It's best to think of them as a fence between the buyer and the seller. They're not technologists and they don't care to be, they just want to make sure that their activities are not traceable and these are the organizations that are operating around the world."

So what's for sale in this more accessible market? Falsified deeds, birth and death records, letters of credit, health insurance cards, source code, diplomas and even people are available for the right price. The anonymity and relative ease of criminal activity is gaining in attractiveness to the barely skilled programmers looking to cash in.

The modus operandi of today's cyber criminal includes commonly known tricks of the trade, starting with the path of least resistance, i.e., social engineering.

According to Hurley, criminals go after their victims using a predictable set of steps: reconnaissance, target, evaluate the environment, install new service or backdoor, cover your tracks, hit pay dirt and run or decide to hang around to exploit and reuse the target, keep ownership of the device - or not, and then move on to the next victim.

With so much information so relatively easy to get to, it's a feast of sorts for the would-be Web Mobber. Using established channels spanning international date lines, and employing thousands of zombie machines, it's more difficult than ever to locate these extensive criminal networks but easier than expected to join one.

So what can be done to protect ourselves from this type of infiltration?

"There's what I'll call best practices and then there's reality. Based on our research over the past 2-3 years there are significant differences in performance results that companies are experiencing with their security programs based on a number of factors such as the strategic actions they take, how they're organized and structured to deal a breach, how they share data and knowledge to minimize security losses, the procedures they use, their policies, and how much active employee training they do," says Hurley.

"There's a whole lot of different things that distinguish one company's performance results from another. There are some common things that are done very well among the best class enterprises that are suffering the least amount of breaches and damages but even having said that, there's probably no way to defeat a serious security threat today and it wouldn't matter what the tool is. The only way to do it would be to unplug the computers."

According to Hurley, the firms that have a good chance of avoiding victimization are the ones with a very active risk management program in place, "An executive team devoted to solving security issues, where the IT security function isn't buried in a hole somewhere in IT but rather implemented as far as a risk management function, cross-company and cross-functional."

Although international governments have joined forces to dismember online Web Mobs, they continually form and disband in a constant game of hide and seek. With many thousands of converts, the seething side of the Web is a thriving economy for those willing to cross over to the dark side. While our indictments are a win, we've only just touched the iceberg.

Melisa Bleasdale would like to thank Jim Hurley, Senior Director, RAZOR Research, Bindview (www.bindview.com) and Executive VP of Research for Security Compliance.com for providing access to his presentation "The Not-So-Unorganized World of Online Crime" which supplied the statistics and framework presented in this article.

Melisa Bleasdale is a San Francisco area communications consultant and strategist specializing in the security industry. Her focus is on emerging and innovative security technology, current events and market concerns. Visit www.superheated.com to find out more about Melisa.

Hardening Windows 2003 platforms made easy

By Alessandro Perilli

More than ever, today, the battle for security is played on the application field.

Years ago, attacking Windows 95 or 98 boxes was not that easy. Few network services to target, few complex networking applications to pull about. Instead of exploiting those, attackers considered as the best way to reach their victims was creating new engaging points. So Trojan horses like SubSeven started spreading on Windows, arriving mainly by e-mail and chat file exchanges.

Since that time a lot of things changed: the firewall "culture" reached the masses, new and improved security tools were developed, modern Windows operating systems got a huge amount of network services, every application became network oriented, people's security awareness increased. Now, with Trojan horses no more effective, attackers needed to find a new way to reach targets. Fortunately for malicious users, Windows 2000 and XP offer a large number of services ready to receive malicious input and provide unauthorized access. Not to mention the thousands of applications, from news aggregators to P2P clients and MMORPG games, where one could use to send malformed network traffic in order to gain remote computer control.

The days of Trojan horses are not over yet but the large majority of attacks are now based on exploiting application vulnerabilities. Why? Have developers started producing more insecure applications? No, quite the opposite. The attackers focused on them, plainly exposing what has always been there - crucial development errors. Development errors are here and will most certainly always exist. They are the product of a typical human brain behavior: taking things for granted. Developers sometimes don't check applications inputs, assuming users will provide data in the correct form, and malformed inputs crash their applications, and in some cases give access to the underlying system with full permissions. These validation input errors are quite probable in modern networked applications. The more complex the application, the easier it will be to forget something. Even if today's vendors apply secure development frameworks to reduce errors, we'll likely have to handle validation input errors for many years to come.

How Hardening Can Help

The best way to mitigate the inherent application insecurity is to harden our systems, hoping endpoint security methods will soon offer something more defenses.

Hardening means reducing the amount of services listening on the system, the amount of installed applications and the way applications handle inputs. In other terms hardening means reducing the attack surface area. Typically hardening is something applied to Operating Systems but it should be considered an approach valuable with any back-end server and desktop application as well.

Today we have hardening guidelines written by well-known security experts and organizations (like NIST), and have partially automated hardening tools, covering several aspects of an OS. Microsoft released its official tool for hardening within the Windows Server 2003 Service Pack 1: Secu rity Configuration Wizard (SCW) that addresses a

lot of problems. Other OSes have their semi-automated hardening tools like JASS for Sun Solaris or Bastille for Red Hat Linux distributions.

Hardening Can Be A Risky Business

Hardening practices exist since a lot of years but are hard to apply. Before stopping a service or modifying a registry key, people should have a deep knowledge of the system. And even in that case, a hardening set of modifications could break an installed application, requiring infrequent access to what you disabled. A hardened configuration could work for a system doing a specific task and not for another. Every platform needs its hardening tuning which is time-consuming and error-prone. Just consider that even when hardening two identical systems you can always miss something. And if the platform role or base of installed application changes, you'll need to review the hardening procedure and adjust it accordingly. It's a hard security life-cycle to achieve, even on a small server farm.

The bottom line is that hardening a system can invalidate vendor support for an installed product because it essentially changes the supported environment.

Exploring the SCW

SCW lets you approach hardening in two ways: per-role or custom.

Hardening with a per-role approach means you just explain the wizard what servers and applications your operating system is going to run. For example, you can choose to declare the SQL Server 2000 role and the ISA Server 2004 role, but also to declare the system will act as a DNS client. Depending on which roles you selected the wizard will submit you a hardened configuration where unneeded services are stopped and registry keys are disabled. This is the best way to start with for a hardening novice.

Hardening with a custom approach means you details every single setting modification of your system. The resulting configuration will be a hybrid-role model tailored for a specific environment. This is the expert way to work with the SCW and should be adopted carefully.

Services and registry keys aren't the only settings SCW can modify. You'll be asked to choose how to setup Local Policies, IPSec filters, Windows Firewall ingress filters and IIS web extensions (if you are going to harden a web server). The whole amount of things you can control is impressive and will require a lot of time and testing before reaching optimal configurations.

SCW explains every setting and therefore enables the user to make the correct choice and becomes a sort of a learning too.

SCW also offers a rollback feature, able to revert your system to its pre-hardening state. This feature is a must-have since troubleshooting a problematic service or application after a hardening can be highly complex. When something you disabled or removed prevents the proper starting of a depended service, it's not always reported on the Windows event log, or if reported, it's not always declared in a clearly. Starting back from a working environment can save a lot of time and availability problems, otherwise the rollback feature always summarizes how the previous state was configured, so you could eventually invoke it just to check and find where the problems could lay.

One of the best parts of SCW is its configuration file. When you finish producing your hardening template it's saved as an XML file. This permits you to deploy it on every single machine in your server farm equipped with SCW, without restarting the template creation process, avoiding mistyping errors and saving lot of time. The whole procedure is done just typing a single command:

```
scwcmd.exe configure /p:my_policy.xml
```

If you work in an Active Directory environment you can assign the XML configuration file to a Group Policy and deliver hardening to all servers within an Organizational Unit (OU) at once.

SCW is distributed as free tool but it won't work on anything but Windows Server 2003 SP1 platforms. A bad decision from Microsoft that hopefully will change its mind for the next version.

Best practices

Even if SCW greatly simplifies the hardening procedure, many things can go wrong. Before hardening a system be sure to study and check service dependencies and applications needs. Custom applications are particularly important to verify.

In Active Directory environments, a hardening configuration applied to apparently similar servers can produce different results and eventually cause services down-time (for example because similar servers weren't installed in unattended ways). So, if you want to deploy the SCW template to a whole OU, you better define a subset of hardening modifications, commons to every OU member and then apply specific hardening settings to every

single server. Do a lot of testing in a lab environment with cloned productions servers before deploying SCW templates.

Remember to document every choice and update documentation on changes.

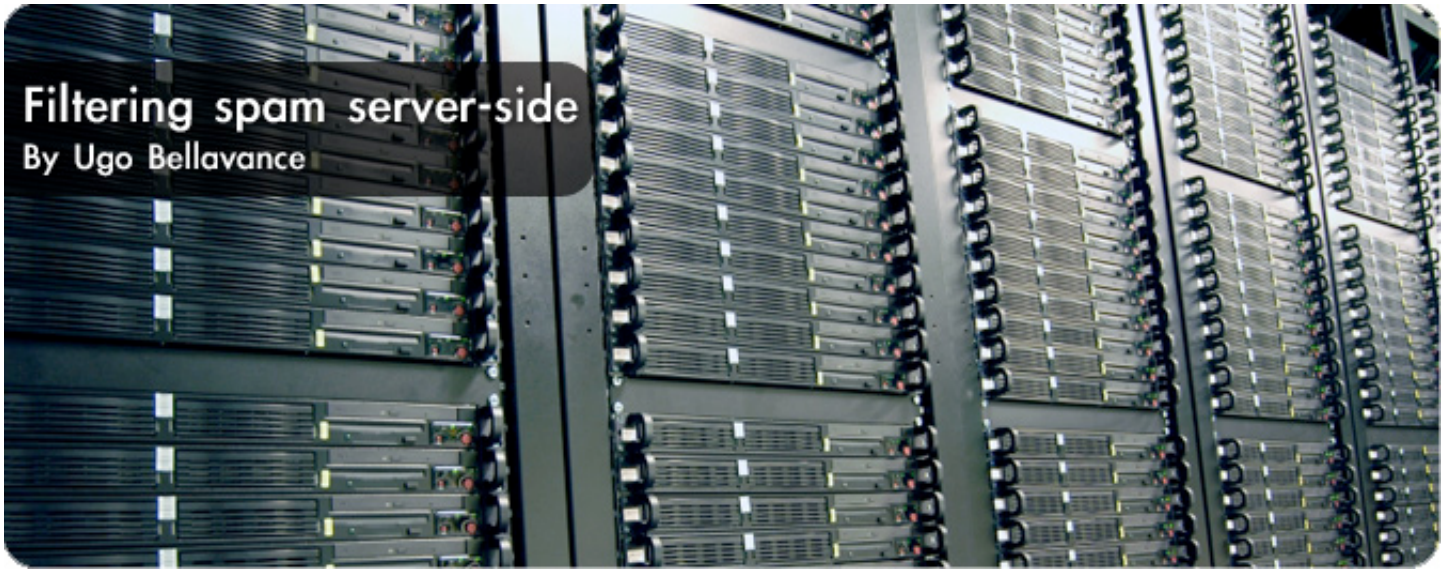
Finally plan a periodical review of hardening templates to adapt them with new knowledge and new needs.

Conclusion

SCW is a great step forward in securing Windows platforms. It does the large part of the job, offers you a basic documentation of what you're modifying and addresses some distribution problems you'll have when dealing with multiple servers.

It requires a good knowledge of Windows behavior and a fair amount of testing before deploying in production. I'd still consider it a tool for experts.

Alessandro Perilli (alessandro.perilli@falsenegatives.com) is founder of False Negatives (falsenegatives.com) and the technology focused blogs Security Zero (securityzero.com) and virtualization.info.



Filtering spam server-side

By Ugo Bellavance

There is no need to give any statistics to claim that spam is now a real and important problem in computing. It is a pain for everyone, even for my grandmother, and this threat is universal and cross-platform. It has consequences on many levels and affects different people in many ways.

There is no need to give any statistics to claim that spam is now a real and important problem in computing. It is a pain for everyone, even for my grandmother, and this threat is universal and cross-platform. It has consequences on many levels and affects different people in many ways. For individuals, the consequences are difficult to assess, but for a business they are clearly harmful on many levels: Management/Financial, Law/Ethics, Technical.

I'll quickly go through some "grey-suit" content, so if you want to go directly to the technical details, feel free to skip to the "Installing a spam-filtering server with MailScanner" section. I added this managerial-related content mainly because I wanted to offer technical persons some arguments to convince their manager/clients that spam-filtering is needed and will reduce costs in the end.

Management/Financial

The most evident consequence of spam is loss of productivity and employee frustration. An organization of 100 employees who each receive 10 spam messages per day will cost approximately $1300 yearly in lost productivity (Google for "spam cost calculator" to get numbers from different sources). 100 employees? $13 000. You get the picture? There is also a chance of discarding legitimate messages, for which the cost could be a lot higher. It is easy to conclude that most spam-filtering solutions pay for themselves fairly quickly, especially when used together with virus filtering. Since many modern viruses can be detected by spam filters, having spam filtering systems translate directly into lowered risks of early infection by new strains of virus, even if they're used without an anti-virus engine.

Law/Ethics

This aspect of spam is often overlooked but is quite significant. In particular, messages leading to an ethical/legal consequence are those that contain pornographic content. Another type that is growing faster than any e-mail-related malware is fraud or phishing. See www.antiphishing.org for details. Many spam filters and anti-virus engines (ClamAV) can detect phishing characteristics in a message. Without being used exclusively for spam, address forgery (using a false address to send an e-mail) can harm an organization's reputation. These elements are all 'contingencies' but when they hit an organization their impact can be financially and operationally large.

Technical

It is becoming harder than ever to live without spam filtering. In medium to large organizations the most cost-effective method for spam-filtering is on the server side, whether internal or outsourced (spam-filtering service). Individual or small businesses will usually rely on the usually not-very-effective ISP spam filters or try client-side software. Be it client- or server-based, adding new technology in an organization always means increased monetary costs as well as increased soft costs incurred in teaching employees how to use new tools.

Client-side spam-filtering solutions usually show a lower initial cost than server-side solutions. However, they can be more complex to install and maintain in an organization with more than 10 clients unless it is provided with a management console. A client-side solution may allow for more granular preferences for each user but may be less effective than a server-side solution (please don't flame me for that, this opinion is based on general assumptions). Finally, server-side solutions can reduce the amount of costly resources needed (bandwidth, CPU time, storage space) to process e-mail on corporate servers.

Installing a spam-filtering server based on MailScanner

My goal in this article is to make it easier for people to install a secure server-side solution that is free, robust and effective. The solution described is based on MailScanner, used with SpamAssassin and its plugins; and virus scanners. I chose this solution because MailScanner allows one to do security check on many levels (file names, file types, dangerous HTML code, Web Bugs, phishing attacks), while using SpamAssassin's very good spam-detection engine. It is constantly evolving and can now suppress dangerous HTML code and Web Bugs in messages; and has a "phishing net" that warns when it suspects a phishing fraud, with a global whitelist that is updated automatically once a day. It can also work with up to about 15 different anti-virus engines and updates them automatically. It is robust, reliable, highly configurable and efficient. A MailScanner server typically uses 2 instances of the MTA; the first receives the messages and queues them. MailScanner then picks the messages in the incoming queue and will process them (spam/virus/threat detection) and will drop them in the outgoing queue when completed. The second instance of the MTA will then pick up messages in the outgoing queue and deliver them.

To increase the reader's knowledge about spam-filtering techniques, I'll throw in some explanations about other technologies that can help make this solution even more effective (DNSBLs, SPF, DomainKeys, Greylisting).

Now, let's get our hands dirty. Please try this on a test server first, and read all the licenses. Not all software mentioned here is free to use in all situations: Razor, DCC, and F-Prot have restrictive licenses.

Of course, there is more than one way to do things, so please don't scream at me if I did not choose your favorite. If you have a better way to do things, you are probably knowledgeable enough so that you can adapt these instructions to fit your needs. I will give explanations rather than exact commands, so I presume basic knowledge of the operating system. Before putting your server in a production environment, make sure you have secured your OS and tested the system thoroughly. More importantly: READ messages printed on screen. During MailScanner's installation you can use CTRL-S to stop output messages from scrolling and CTRL-Q to start it again (install.sh output). These procedures were all correct at time of writing. You may need to adapt them over time.

My goal in this article is to make it easier for people to install a secure server-side solution that is free, robust and effective.

Before installation: Make sure your operating system is up to date and you have a supported MTA installed and configured correctly. Here are generic installation instructions for FreeBSD, Debian Sarge Linux and Red Hat Enterprise Linux 4. I'm more familiar with the RHEL platform, so the directions are obviously more complete for this platform.

Installation on FreeBSD

- Minimal install of FreeBSD
- Make sure you enable SSH login
- Create a user in the group wheel in the installation so that you can su
- Answer yes when you're asked if you want to install the ports.
- After the OS install process, install these ports: clamav, bdc, f-prot, p5-Mail- SpamAssassin, razor-agents, dcc-dccd, pyzor, spamass-rules, spamass-rules_du_jour, (search at www.freebsd.org/ports/) and configure them correctly
- Install ports: mailscanner, then mailscanner-mrtg
- Make sure you read carefully what is printed on screen at the end of the installation of the MailScanner port.
- Look in /usr/local/etc/rc.d and edit
    1. mailscanner.sh.sample
    2. mta.sh.sample
- and rename them without the .sample. Those are the startup scripts.

Installation on Debian Sarge

- Using APT or Aptitude, install: mailscanner, pyzor, razor, spamassassin, clamav (volatile), dcc-client.

Installation on Red Hat Enterprise Linux 4 (or one of its clones)

- Minimal install
- Change, in /etc/sysconfig/i18n, the language to "en_US" and reboot
- Download the MailScanner and the ClamAV + SpamAssassin packages from mailscanner.info
- Download Razor from razor.sourceforge.net/ (agents and sdk).
- Download Pyzor from pyzor.sourceforge.net/
- Download DCC from: rhyolite.com/anti-spam/dcc/source/dcc.tar.Z.
- If possible, check GPG signatures/MD5
- Untar all the packages and install them, following the instructions given in the tarball.
- Install MailScanner using the install.sh script inside the MailScanner package. There is also an install.sh script for the ClamAV + SpamAssassin package to ease this installation as well.
- You may need to install dependencies like gcc, rpm-build and zlib-devel. They are easy to install with yum.

Once you're done with the installation, take a few minutes to read the file "MailScanner.conf" (in /etc/MailScanner on RHEL and Debian, in /usr/local/etc/MailScanner on FreeBSD). The default settings are rather safe, but you must change two parameters to activate SpamAssassin and virus scanning.

1. Change "Use SpamAssassin = no" to Use SpamAssassin = yes"
2. Change "Virus Scanners = none" to "Virus Scanners = clamav bitdefender fprot" (depends on which engines you choose).

This will give you a basic, functional spam- and virus-filtering mail server. The default settings are for Sendmail, so you will have to make a few edits to make it works with your favorite MTA. There are many tweaks you can do to improve the spam-catching effectiveness and processing speed. MailScanner is highly customizable, using the parameters in the MailScanner.conf file. Rulesets are at the core of MailScanner's flexibility. They allow one to designate different parameters, depending on many conditions (source IP, from:, to:, etc).

I strongly encourage you to visit the MailScanner Wiki (wiki.mailscanner.info) and especially its "Most Asked Questions" section (wiki.mailscanner.info/doku.php?id=maq:index). Along with optimization tips, the wiki contains information about how to test and use the software. There is even a full section about the "Best Practices of e-mail server administration, which covers topics from reverse-lookup records that matches EHLO string to confidentiality disclaimers.

Another good move would be to buy the MailScanner book (mailscanner.info/store/). It is written by MailScanner's author, Julian field. It helps making the learning curve less steep and encourages the continuing development of MailScanner.

MailScanner servers are mainly used as mail-filtering gateways that are "in front of" corporate mail servers (Exchange, Domino, Groupwise). A search for "gateway" in the wiki will lead you to instructions for your MTA. It is even possible to configure the MTA on the spam-filtering system so that it accepts messages only for valid e-mail addresses. This is done by performing a request to the corporate mail server before accepting the message and saves a lot of resources on both the MailScanner and corporate server.

Once you've got that working and read through some documentation, I suggest you subscribe to the MailScanner-announce list to get announcements and the Mailscanner list where you can get answers from knowledgeable users.

What now? You need to manage this server, and your manager/customer can't wait to see stats and reports. For statistics, you should be able to install Vispan (while.homeunix.net/mailstats/) (from source) and MailScanner-MRTG (mailscannermrtg.sourceforge.net/) (packages/ports available for RHEL and FreeBSD, source install for Debian). To help you with the configuration of MailScanner, you may want to use the MailScanner Webmin Module (sourceforge.net/projects/msfrontend/).

Finally, the ultimate management interface for MailScanner is MailWatch, available at mailwatch.sourceforge.net. It is slightly complex to install and requires an AMP setup (Apache MySQL PHP) to work, but it is really worth it. It allows you to see all the messages that have been processed, get real-time statistics about message processing and about your system (load, queue size, etc.), manage black/white lists, create highly-customizable reports. Instructions are included in the tarball and on the website.

To conclude, a few words on 3 spam-fighting tools: DNSBLs, SPF, DomainKeys and GreyListing.

DNSBLs (DNS black lists) have been used since the early ages of spam filtering. On a MailScanner server, DNSBLs can be used at 3 levels, with different consequences. First, they can be used at the MTA level. At this level, it is black or white; there is no 'maybe': If the originating server is on a black list, the message is immediately rejected. That makes it a bit risky for false positives. At least, the sender has a reject message explaining why the message has been rejected.

When used at the MailScanner level, it needs more processing than at the MTA level, but at least the system administrator can decide what to do with a message that comes from a blacklisted server: consider it as spam, or high scoring spam, after x RBL hits all configurable by rulesets. It is more flexible than at MTA level, but not as flexible as when used in SpamAssassin.

SpamAssassin adds a (configurable) score, depending on the specific RBL that has a positive result. Of course, this requires more processing power than the other methods, but lowers considerably the risk of rejecting or deleting a legitimate message.

SPF (spf.pobox.com) is a tool based on DNS records stating from which IP address outgoing messages from a certain domain should come from. For example, it will help identify a spam message coming from joe@yahoo.ca that comes from a compromised server in China.

Its main advantage is that it doesn't cause extra delays or many false positives. However, its success depends directly from the number of domains that have SPF DNS records. There are different ways to test messages against DNS records, including SpamAssassin and MTA-level tools like Sendmail milters, but if you don't want to filter based on SPF records, please at least put up SPF records for your domain.

DomainKeys (antispam.yahoo.com/domainkeys) is similar to SPF, but instead of using DNS records, it uses public key cryptography to prove that a server is allowed to send mail for a specific sending domain. As a side effect, it guarantees message integrity between two servers. DomainKeys is dependent, like SPF, of how many sites use the technology. However, it offers a more reliable mechanism.

Greylisting (projects.puremagic.com/greylisting/) is rather new and aims at reducing the load on servers while improving the anti-spam effectiveness of the whole system. It is implemented at the MTA level and here is a quick, very simplified overview:

1. If the "triplet" of the message (originating server IP, envelope sender, and envelope recipient) is not known, the MTA sends a "temporary failure" message, saying to the originating server to try again later.

2. If the "triplet" is known (has been seen before), the message is delivered as usual.

The idea behind the idea of Greylisting is that zombies and spammers will not retry the delivery, and legitimate mail servers will. It features white listing, different configuration parameters and a "learning" mode (depending on the implementation).

It usually helps block a lot of messages at MTA-level. Since post-processing (MailScanner, SpamAssassin, etc.) are more expensive on resources, Greylisting usage reduces the load on the server considerably. Many see an increase of over 80% in rejected spam at MTA level, and a significant decrease of undetected spam messages (~60%).

All of these concepts can be implemented on a MailScanner-based server and most of them depend on MTA features. Most MTAs have features that have not been discussed here, but can be configured independently, since MailScanner is never involved in the SMTP transaction.

Since spam and viruses are constantly evolving, the best way to keep a high spam-catching rate is to use many techniques and update their implementations often.

Ugo Bellavance (www.lubik.ca) is an independent consultant in computer security. He's an expert in e-mail filtering servers, but he also appreciates playing with intrusion detection systems web and database servers. In his spare time he enjoys telemark skiing, mountain biking, hiking, cycling, and playing acoustic guitar.

o3 magazine

The Open Source Enterprise Magazine

http://www.o3magazine.org

FREE monthly Enterprise Magazine

FOCUS on Enterprise Data Networking

Over 500,000 readers in 140 countries

Regular Features Include:

- **Security**
- **Internet Technologies**
- **Web Technologies**
- **Networking**
- **Network Applications**
- **Network Security**
- **Open Source Business**
- **VoIP (Voice over IP)**

http://www.o3magazine.org