Introduction to System Security

Leonard Rogers

(Capstone Submission for MINT 709 Course)

Capstone Advisor: Dr. Mike MacGregor

Table of Contents

Table of Figures

Chapter 1: Security Basics

General Security Concepts

Preface

Systems Administrators, Security Administrators, Network Administrators; whatever title they have, they are tasked with building, maintaining and most importantly, protecting the IT systems from malicious software, intentional intrusions from both internal and external people and even accidental activity that can destroy the organization's data integrity.

From this point forward we will simply refer to these individuals as the "Administrator" or "Administrators". In the end, you must take this title to mean "you", since at some point in your professional's career "you" will be an Administrator.

Likewise, many of the topics discussed here are based on a common sense approach and experience. The problem with common sense is that in reality it is not that common, many Administrators simply follow "best practice" guidelines yet don't really take the time to fully understand if the solution they are implementing is the best one for their overall systems design.

As for experience, an Administrator just starting their career, has very little and while they can gain much wisdom and knowledge from the more senior "seasoned" staff they work with, their best experience they will likely gain through their career is the experience of failure. Not failure in doing the job wrong, but failure by assuming that they have done everything right and that their systems are impenetrable.

There really is no such thing as an impenetrable system that can still be considered useful. Yes, encasing a computer in concrete and submerging it at the bottom of the ocean would make it relatively impenetrable, but it is no longer useful. Experience comes from implementing solutions to meet your organization's critical business needs, and it comes from testing and re-testing the security implementations and it eventually comes from finding out that someone successfully penetrating the defenses that they have so carefully implemented.

With so many tools and techniques available to the attackers, they will eventually get in…, it is just a matter of time. There is no perfect security solution, there are general solutions that every organization puts in place and then there are the better solutions; those that are tailored to the organizations specific needs, systems design and infrastructure, from the oldest device to the newest.

By no means is the information here to be considered a complete manual for preventing security breaches. Just as there are an endless number of corporate infrastructure and systems design solutions that can be put in place, there are an endless number of security risks and likewise, security solutions that can be implemented to help prevent or delay a successful attack. The information here should be taken as a guide to help along the way, and modified, or ignored, as needed for each specific situation you may encounter.

Introduction: Hollywood vs. Reality

War Games, The Net, Sneakers, Swordfish are all movies that glorify the world of hackers and hacking and every day you can find news stories from all over the world about how hackers have compromised a company's website, its databases, or other internal systems to steal confidential information or personal records. Every industry has been affected by the activities of these "hackers"; hospitals, educational institutions, governments, private and public businesses, utilities and even charities.

However, Hollywood and the news media have never truly been able to bring the real world of hackers and hacking to the silver screen or the headlines. Traditionally, a hacker, by definition is someone who enjoys the art of programming, modifying applications to provide unique solutions to problems or modifying programs to improve a system's performance. The portrayal of hackers by Hollywood and the news media is more in line with the definition of "cracker".

By definition a cracker is a person who accesses computers and computer networks by exploiting security holes or by circumventing its security systems. Be that as it may, we will use the popular media term of "hacker" throughout this course as we discuss computer security issues. To further complicate the terminology, there are different types of hackers (White Hats, Grey Hats, and Black Hats) and there is a significant difference between professional hackers and amateur hackers.

Straight from Hollywood's old black and white movies, White Hats (the good guys) are people who perform their hacking activities for the purposes of helping organizations find and close security risks to their computer systems. Black Hats (the bad guys) are people who exploit the security weaknesses in computers and computer networks for malicious or criminal purposes and the Grey Hats (neutral) are generally Black Hats turned good or people that have their own agenda for breaking into computer systems that are (in their minds) not usually malicious or criminal.

As for professional vs. amateur hackers; professional hackers (good, bad or otherwise) generally have only three goals when hacking:

- Find the security and exploit weaknesses in the systems
    - This can for the purpose of helping a client, stealing an organization's data or exposing a weakness for others to use.
- Avoid getting caught
    - Again this can be to verify and validate a client's security system and its detection/alerting capabilities or simply ensure that the authorities are not able to trace the hacker back to his/her origin by covering their tracks through the deletion or editing of log files.
- Get paid
    - This can be in the form of a cheque from a contracting client, the sale of stolen information (such as patentable, personal or credit card information), or payment can be extracted by embarrassing the attacked organization or harming its business reputation.

The professional hacker may even be using many of the same tools as the amateur hacker, however their skill and experience with using the tools is far greater. They can generally extract more information and specifically target more valuable information. Simple rule here, the more valuable the information, the more they will get paid.

Amateur hackers are often younger individuals who are trying to make a name for themselves in the online community or trying out software they have downloaded online without care or concern for the consequences. They can often do as much damage to their own systems as they can to the systems that they direct their attacks against.

Most often amateur hackers lack focus in their attacks and when they do have focus it is often only for personal reasons, such as hacking the computer systems of a local school or business computer system. They have little concern about covering their tracks and in reality, very few get caught.

The reason they don't get caught is because, in the case of a school, the Administrator has very little time to really review the system access logs and find out where the attacks are coming from or is so overwhelmed with other issues that they aren't even aware of the attacks. For the small business, they usually don't have their own IT staff so anything that goes wrong is fixed by a local contracted tech or service provider whose sole purpose is to get in, fix the problem and get out as fast as they can.

The more service repairs the tech can perform in a day they more money they make. In most cases these techs also have very little in the way of security training and even if they do they often lack the knowledge, tools and time to track the attacker down.

In either case, unless serious damage to the systems is caused, they law is usually not brought in. Even if the attackers are discovered and caught by the organization, most organizations would prefer to sweep it under the rug and have their techs patch the security holes so it doesn't happen again.

Why? There are really three reasons for this; first the organization often doesn't want others to know of the incident as it may damage their business reputation with their customers. Second, the organization often doesn't think the issue is serious enough to bring in the law, a stern talking to the offender and their parents should resolve the issue.

Finally, the unfortunate third reason has more to do with the requirements and costs associate with the legalities of reporting the issue. Often the burden of reporting the issue to the authorities, followed by the costs associated with an audit and any disclosure requirements placed on the organization are more than most small organizations could afford.

In some cases, if the auditors deem that the organization was negligent in its responsibilities for putting adequate security practices in place, it may also face hefty fines for the incident. All, of this is placed squarely on the shoulders of the attacked organization, and they were the victim.

For large multi-national, billion dollar organizations, that were the victims of professional attacks (see the links provided below), a large fine if they were found negligent may be warranted, however for a small family run business it can easily be what closes their doors for good.

Cost of Sony Hacker Attack – Los Angeles Times
http://articles.latimes.com/2011/may/24/business/la-fi-ct-sony-20110524

VG24/7 – Post regarding potential fines against Sony
http://www.vg247.com/2011/04/26/sony-issues-statement-on-psn-outage/

To be honest, it is believed that nearly every website, business and home internet connection has been probed at some point for weaknesses. There are very few safe havens left on the internet, however there are even fewer serious (professional) hackers who are looking to break into your personal home PC. The risk versus reward just simply isn't there.

In most cases these probes are the result of amateurs, who wouldn't really know what to do if they did get in, or the result of automated probes generated by automated attack systems which includes PCs infected by botnets or worms, looking to expand to new systems.

For most people a good firewall and up-to-date anti-virus software is enough to prevent most of the attacks, the same cannot be said for larger organizations that come under fire from direct professional efforts. Given enough time and resources, every system is vulnerable to their attacks and will eventually be compromised.

Common data gathering techniques and tools

Footprinting and reconnaissance

Footprinting is the process of gathering as much information about a computer system as can be gathered without actually gaining access to the system itself. This process also gathers information about other systems the target may be connected to or associated with and can even detect the presence of a firewall.

Generally, it doesn't take much practice or even very high tech tools to perform footprinting of a system. It is deployed through the use of a wide range of basic tools that are available to nearly every person whether they are connected to the Internet or not. These tools or techniques include:

- Reconnaissance and site visit
    - Nothing quite beats a good old fashion site visit and general reconnaissance. It allows the hacker to literally see who the target organization is buying their systems from.
        - What brand of desktop or laptop does the staff use?
        - What operating system are they running?
        - How old are their systems?
    - Depending on how close they can get, they may even be able to tell what software is running on the systems for antivirus and office applications, it can even reveal any additional security or connectivity tools that the organization has deployed.
- DNS queries and IP Address lookups

- These are used to determine what domain the system is connected to or is a part of. The information gathered here can provide the hacker with country, city and even address locations associated with the system(s).
- As shown in Figure 1, the WHOIS protocol, commands and available websites provide easy access to this information. So what does it tell us?
  - It tells us who the DNS registrar is.
  - It tells us who the administrative contact is.
  - It provides two different room numbers in the same building and the building address for the contact.
  - It provides phone numbers and an email address for the contact.
  - It provides us with two different IP Addresses for two different Name Servers used by the organization.

```
Domain name:            ualberta.ca
Domain status:          registered
Creation date:          2000/10/04
Expiry date:            2012/12/20

Registrar:
    Name:               easyDNS Technologies Inc.
    Number:             88

Registrant:
    Name:               University of Alberta

Administrative contact:
    Name:               Kevin Watts
    Postal address:     AICT 352 General Services Building University
                        Edmonton AB T6G 2H1 Canada
    Phone:              +1 780 492-9583
    Fax:                +1 780 492-1729
    Email:              kevin.watts@ualberta.ca

Technical contact:
    Name:               Kevin Watts
    Postal address:     University Computing Systems 103H General Services
                        of AB T6G 2H1 Canada
    Phone:              +1 780 492-9583
    Fax:                +1 780 492-1729
    Email:              kevin.watts@ualberta.ca

Name servers:
    name.ualberta.ca    129.128.5.233
    nom.ualberta.ca     129.128.76.233
```

Figure 1: WHOIS results for ualberta.ca

Network scanning and queries

Network scanning can provide the hacker with a great deal of information including specific attack vectors. Gathering this in depth information allows the hacker to focus their network

scanning activities looking for weaknesses in the victim's defense systems. Generally, professional hackers will take greater care when performing these types of scans since they will often set off any intrusion detection or prevention systems in place.

This will alert the organization's Administrators or their security team; as a result the in depth scanning process is usually slow and methodical to prevent detection. An alarm or alert set off here is usually of great concern to the Administrators simply because scanning almost always precedes an attack.

There are a wide range of tools employed in network scanning. They can be as simple as traceroute and ping sweeps which look for target hosts and the path to them. For getting at specific information, vulnerability and port scanners such as nmap, nessus and Metasploit are used to find open ports and systems that are most vulnerable to attack and least likely to alert the Administrators to the attack.

Network queries are also part of the scanning arsenal. DNS queries can provide the attacker with a large amount of host related information. The simple fact is, that retrieving a host resource record through a standard DNS query, is a pretty good indication that there is a live target that can be checked for vulnerabilities.

Administrators often use DNS zone transfers to configure their zone files (essentially a local DNS database) on a single local master server and often allow this data to be transferred to one or more local secondary name server. Review Figure 1 again, there are two DNS servers; one for English (name.ualberta.ca) and one for French (nom.ualberta.ca). Are they both configured the same? Are they both patched the same? Are they both even running the same operating system? Which is the master and which is the secondary?

This is often the weakest link in the process of protecting host records and many of today's Administrators only allow zone transfers to occur to specific servers and will use some type of authentication or digital signature to verify the receiver. However, often these servers are not necessarily managed or controlled by the Administrator; such as those used by the organization's ISP. This is where the vulnerability comes in, as many ISPs provide little to no control over zone transfers and will allow them to any host.

Enumeration and Operating System identification

Network enumeration goes hand-in-hand with network scanning. It is used to gather Information about the user accounts and their format, user groups the accounts may be members of, network shares accessible to the user account and even network services that are available.

Again, many of the same tools (nmap, Nessus, packet sniffers, etc.) are deployed here simply because the responses from the scanning processes can often provide this information through details provided in the response packets. While the information contained in response packets may seem innocuous, an in-depth review of the data reveals that different operating systems respond with slightly different details in the data.

Taken separately, these differences may not mean much. But when analyzed against a cross-reference database, the responses are used to differentiate one operation system from another such as Red Hat Linux version 4.1versus Red Hat Linux 7.3 versus MS Windows Server 2000 versus MS Windows 7. It can sometimes even determine major OS patch levels such as MS Server 2000 Service Pack 1 versus MS Server 2000 Service Pack 3.

Social Engineering

IT people are not known for their flamboyant social skills, so those that do have it, tend to have it in abundance.

For those that do have it, social engineering is often a way to get detailed systems information directly from an organization's Administrators simply by talking to them. If done properly, the Administrator won't even realize that they are passing along valuable confidential or semi-confidential information on how their systems are configured.

The premise of this type of attack is as follows:

- The attacker shows up at a local vendor provided function or even a coffee shop frequented by the target organization's systems administrative staff.
- Through the process of casual conversation, the attacker will isolate one or more of the target organization's Administrators that the attacker feels they can get the most information out of, often a junior staff member who is eager to show off his knowledge while having his ego stroked.
- Eventually, the discussion is directed towards a falsified issue that the attacker is having with a server configuration.
    - Previous reconnaissance, port scanning, enumeration and other queries have gotten the attacker enough information about the target organization's internal systems to be able to fake a realistic scenario that is close enough to the target organization's systems that the victim believes they are helping a fellow system's Administrator.
- In an attempt to help, the victim Administrator discloses how they resolved the issue or why their systems design doesn't have the issue.
    - Often, the attacker presses enough, by providing false information about how his own fake organization's systems are configured, to get the victim Administrator to reveal detailed configuration information.
    - This is information that they otherwise would not have gotten until they had already hacked inside the target organization's systems and can include IP addresses, VLAN configuration and even drawings of how the systems are interconnected.

With today's blogging and social media connections, a hacker with these social skills can perform much of this social engineering online, without the victim ever having seen their attacker's face. It is easy to set up a fake online profile with fake photograph (for that real personal touch), and over the course of time get the victim Administrator to reveal much about the internal workings and configurations of their systems.

This process gets even easier if the attacker finds that the victim Administrator has posted online for help on a problem. The attacker tries to come to the rescue but just needs slightly more information on the victim's system configuration.

Through this process, the attacker may actually help the victim resolve their problem and provide a solution while getting the victim to reveal the information that the attacker was really looking for. It also provides a great foundation for future conversations and more information gathering.

Common attack vectors

To be honest, there are hundreds of different ways and tools that have been developed to attack today's modern computer system. There are attacks designed for the desktop and server operating systems (Linux, Windows, Apple it doesn't matter), attacks for the hardware and drivers, attacks designed for the desktop and server applications, attacks designed networking equipment and protocols, etc.

This list pretty much goes on and on, there are even attacks based on systems previously thought relatively impervious to attack; modern industrial control systems and PLCs (Stuxnet anyone?). Because the list is so long and varied, IT Security specialists have become a mainstay in large organization IT Departments, to the point where there are now specialists for many of the different areas of security needed.

There are network security specialist, firewall and intrusion detection specialist, server security specialist, and application security specialists. Many larger organizations also hire third party security auditors to test their security implementations, which has led to the title of Penetration Tester and certifications such as the "Certified Ethical Hacker".

If your interest lies in the IT Security field there are many organizations which can provide extensive training in this area. The SANS/GIAC courses and certifications are a way that many people get started, once they actually begin their IT career.

However, for the purposes of this document we will look only a brief cross-section of the many attack vectors used to gain unauthorized access to an organization's computer systems, those that have been persistent and continuously used during the last 10 years or more and will like still be in use beyond the next 10 years.

Viruses and worms

Computer viruses are small applications that inject themselves into other computer programs. The common thread of all computer viruses is that they cannot spread by themselves in the wilds of the internet. Instead they must be (initially) placed on a host or mobile storage media which is then manually moved from computer to computer.

Once the virus has infected a host, any removable media or messaging applications (email, messaging services, etc.) can be infected and used to spread the virus. In general, viruses cannot move from host to host by themselves, they rely on other modes of transport to transmit themselves and infect other hosts.

Viruses can create very little damage to a computer system, to extensive damage depending on the author's intentions. Nuisance viruses such as the "Stoned" virus would infect the boot sector of a PC's hard disk drive and when activated (a one in eight chance) would pop-up with the message "Your PC is now Stoned!"

More dangerous viruses, like "agent.btz" spread through USB thumb drives and once it infected a PC would steal data from the PC by transmitting it to storage servers on the internet. The largest claim to fame for this virus is that it forced the Pentagon to ban thumb drives and resulted in the creation of the U.S Cyber Security Department; a division of the U.S. Military.

Unlike viruses, worms have the capability to move from system to system on their own. They do this by taking control of and using network-based applications on the infected PC. While a virus waited for the PC's user to spread it through use of the network-based applications or manual transfer through removable media, the worms would literally take over the PCs email or TCP stack and begin sending itself out to other computers.

Another significant difference in the worm is that it may have multiple malware applications or functions associated with it. Major worm infections such as "conficker" created a massive botnet whose function is still not fully known and which may still infect millions of PCs worldwide. Others, like Fizzer, have a known purpose – to spread spam email – took over the infected PCs email application and used the address book to both spread itself and send out millions of spam emails.

Trojans and backdoors

Trojans, also referred to as a Trojan horse, is a standalone program that does not generally infect other files of the computer, but instead masquerades itself as a legitimate application that the computers user is convinced is useful to them.

One of the most common Trojan's today is the "Security Shield" anti-malware software. This is a fake anti-malware application that uses other viruses or infected websites to spread. Each year new versions of this application are created, and users who encounter this Trojan are usually greeted with fake security alerts and crippled use of their real anti-virus/malware application. The user's web browser application is also typically crippled and prevents the user from getting to legitimate anti-malware sites and redirects the user to a "pay to purchase the full version of the software" website.

The payment website is real, and they will take the user's credit card information but they really don't process anything. The software itself is fake and cannot actually remove any viruses or other malware from your system. But what do the creators of the software care, their objective was reached and they are going shopping, after all they have the user's credit card information.

Other Trojans often carry other applications with them. The "Zeus" Trojan would install keystroke logging software onto the infected computer and steal personal and banking information from the computer user. This information is often used for financial and identity theft by the criminal organizations that implemented it.

The “Beast” Trojan would install the equivalent of a complete “Remote Administration Tool” in which the Trojan’s creator could remotely access and take over the victim’s computer. This included:

- being able to upload and download files onto the system
- edit the victim computer’s registry
- turn off anti-malware and firewall services
- remotely take over the victim computer’s webcam.

This is where the “backdoors” come in. These Remote Administration Tools (aka RATs) are really backdoor applications that allow its creator to remotely access the victim’s infected computer whenever they want. In most cases, the owner/user (victim) may not even know that there is someone else accessing their computer. It is difficult to detect that someone else copied files off of a system without monitoring software, and with the right tools installed, the victim may not even be allowed to see any of the hidden directories and files that the attacker created and/or placed on the system.

There are several reasons that these tools are created. Primarily it gives the attackers the ability to steal personal data off of the system without the user’s knowledge. It can even give the attacker a place to remotely hide any files that they have stolen from other systems. The ability to remotely, fully, control another computer also allows the attacker to install other software for performing attacks against other computers, and keeps them safely away from the action.

After all, if someone traces an attack back, it is the victim’s computer that all of the attacks came from, and until the Remote Administration Tool is discovered, it is the remote computer’s owner, or victim, that is left trying to explain to the authorities that they had nothing to do with the attacks.

DoS and DDoS

DoS (Denial of Service) and DDoS (Distributed Denial of Service) attacks have been around for a long time and really their name describes their function pretty well. These attacks are designed to deny a legitimate user access to a legitimate service, such as FTP or HTTP services being provided by a company, through the use of a SYN Flood attack.

Denial of Service attacks is the original form of this attack. It was nothing more than a single computer attempting to open up thousands of sessions to a service but never completing the standard TCP Three-way Handshake connection process. Instead, the attacking system would send the initial SYN packet to the service’s providing server, such as a web server (HTTP), using a faked or spoofed IP Addresses (why get caught using your own real IP Address) which would result in the web server responding back with a SYN-ACK packet.

The system receiving the SYN-ACK packet from the server would never respond with the final SYN-ACK packet of the Three-way Handshake, simply because it never initiated the connection in the first place and as a result does not have a partly-open socket pairing to match the incoming SYN-ACK packet to. It also doesn’t respond with a RST-ACK packet, which would kill the

server-side connection, again because it doesn't have a connection related to the incoming SYN-ACK from the server.

This leaves the connection on the server in a half-open state. Due to the numerous timeouts, built into TCP's connection-oriented design, a server that receives a SYN packet must maintain a listen state for at least 75 seconds. On some older TCP stacks, such as those in Windows NT, it could take as much as three and a half minutes for the server to timeout the half open connection and shut it down.

During this timeout period, the attacking system sends out more and more fake connection requests, eventually causing the server's TCP stack to overflow and crash due to either a lack of socket pairs available or to a shortage of buffer memory in which to track the connections.

One must also remember that, as this attack is occurring, legitimate user connection requests are either getting dropped because the server is too busy handling the fake requests, or because the server has simply hung-up due to the overload. End result, the legitimate user is denied access to the service.

It is difficult for a single workstation to shut down or create a DoS attack against the multi-server, redundant connection, globally distributed solutions implemented today. There are also coalitions of organizations, such as those working with Akamai Technologies, that are actively involved in preventing DoS and DDoS attacks.

As a result of the changes in technology, such as more powerful, resilient and redundant systems hosting services on the Internet, the DoS attack also changed into the DDoS attack. The premise of the attack is the same, however instead of one system attempting to perform the attack. The organizers of the attack use tools such as RATs, Worms and Botnets, which they have managed to get onto thousands of victim computers. These applications are all configured, either at a specific time or through direct commands sent from the attacker, to perform a massive coordinated attack against a targeted host's services.

Again, DoS and DDoS attacks are always being fought against by many organizations and through advancements in technology. For their part the attackers continue to compromise large numbers of systems to in order to coordinate an attack and try again.

Botnets

Bots (often referred to as zombies or drones) are small applications that provide the attacker with a remote control mechanism to control the infected victim system. Through this remote control mechanism, the attacker can issue any command programmed into the bot, including taking complete control over the victim computer.

The attacker's initial purpose is to get hundreds or even thousands of victim computers infected with their bot, referred to as a botnet. With all of the bots in the botnet working under a common remote control mechanism, referred to as the "Command and Control" infrastructure, the attacker can then increase the effectiveness of their attack through a "strength in numbers" approach.

Initially, these botnets were used to create the initial DDoS attacks on IRC networks. However, today's bots are most often used by cybercriminals for the purposes of financial theft, identify theft, mass spamming, etc. They have also altered the Command and Control infrastructure to prevent the attacker from being easily traced.

Traditionally, the bots were all controlled through IRC communications for Command and Control, with one central and traceable IRC server sending out all of the commands. These traceable IRC servers allowed authorities to track down the central control server and put a stop to any criminal activities that were being performed by the attackers. As a result the attackers changed how their Command and Control infrastructure communicated and began to use covert channels for controlling how their bots.

These covert channels would include HTTP or DNS tunneling. For example the attacker could send commands to the bots inside HTTP requests or DNS.TXT records. For the most part, the bots would still randomly contact the central Command and Control server through these communications protocols, so to a certain extent these central servers could be still traced back. However, an HTTP website can be set up anywhere on the planet through many legitimate web hosting companies, so tracking down the actual attacker proved significantly more difficult for the authorities.

The bots could also be designed to request commands from multiple Command and Control servers, thereby making it more difficult to shut down a botnet's activities since if the authorities shut down once Command and Control website, the bot would revert to its backup site for any new commands and possibly be re-programmed with the backup site as the new primary site and the address of a new backup site. With this capability, it is proving extremely difficult for the authorities to shutdown botnets without actually removing the bots from the infected systems and nearly impossible to track down the attackers themselves.

Another option for the attacker controlling the bots was to use P2P communications systems to avoid the single point of failure of a central Command and Control server. Through this method, the attacker places a file up on any one of the many P2P or .torrent providers for the bots to access. The file contains all of the commands the attacker wants the bots to carry out and the next file location, on some other P2P network, where the bots are to get their next set of commands.

Regardless, of the communications methods used by the Command and Control mechanisms, there are two types of commands that the attackers generally send their bots; attack and update. The attack commands can be nearly anything that the attacker needs the machine to do from performing a DDoS attack, sending out large quantities of spam, gathering and stealing user information such as passwords, credit card numbers or even back account information through website activity monitoring and key stroke recording.

The update commands often inform the bot to download and execute a file from the internet. These files can contain updates such as patches and bug fixes to full upgrades that allow new types of attacks to be performed against newly discovered OS vulnerabilities or allow the bots to spread to new systems through theses vulnerabilities. After all, the attackers are well aware that

many people do not update or patch their systems regularly and for large corporate organizations with many complex and mission critical systems it can be months before new patches rigorously tested and applied, if they can be applied at all.

Today's bots and botnets have become increasingly difficult to track and shutdown. Many of the people that have computers infected with bots from those Command and Control servers that the authorities have shut down, still don't know and may never know that their computer was infected. The danger in this is that it leads to the possibility that someone, sooner or later manages to retake control of these bots, update them and start the process all over again.

Common system hacking methods

Password Cracking (brute force, dictionary and rainbow tables)

Password cracking is often on of the easiest methods of cracking a system password, with an abundance of open source applications freely available on the Internet. There are applications to crack password files take from a system, from packet sniffing on the network and even bootable password "recovery or reset" software that is available in CD or USB format.

The brute force attack pretty much describes itself. Generally used to defeat login passwords, the attacker tries every combination of characters (upper and lower case alpha, numbers and symbols) in an attempt to log into a system with an Administrative ID.

Today's modern authentication servers typically require some form of "complex" password based on a minimum character length and a combination of upper/lower case alpha, number and symbols. They also often require that it be changed every 90 to 120 days and will not allow the user to use the password again for some period of time. That period of time can be an actual measure of time (i.e. cannot reuse the password for 6 months or a year) or can be based on the number of password changes (i.e. cannot reuse the password for the next 3, 6, 9 or 12 passwords).

However the users, out of habit and to ease the remembering and recall of a password, will often make their login IDs based on some combination of word, number and symbol combinations that have some meaning to them.

So the user generally picks:

- a word that have some meaning to them:
    - their son's/daughter's name,
    - their pet's name,
    - their favorite music band,
    - their favorite fruit, etc.,
- a number that has some meaning to them:
    - their son's/daughter's birth year or month/day combination
    - their year their pet was born,
    - the year they got married, etc.,
- a symbol or two that is easy to remember and most importantly easy to type
    - # or ##

    - $#
    - ++, etc.

Under most complex password systems all of the following will pass the test for a user login password with a minimum of 8 characters and alpha/numeric/symbol combination.

- `P@ssw0rd` or `P@55w0rd`
- `Adam123#` or `@dAm123#`
- `Eve1234$` or `Ev31234$`
- `F1uFFy##` or `F1uffy##`
- `We1come?` or `Welc0me?`
- `App1e!23` or `@pp1E!23`

Of the above written passwords; `Adam123#`, `Eve1234#` and `Apple!23`, are particularly bad examples for legitimate passwords. These passwords contain a complete word without any modifications, making a dictionary-based attack very quick and efficient at cracking them.

A dictionary-based attack is implemented by an application which uses a dictionary of every word in a language, generally the English language, and can add additional number/symbol combinations to the dictionary word to crack the password. Like a brute force attack, Dictionary-based attacks can be very fast (just a few minutes) to taking a relatively long period of time depending on the parameters that the application is running under. Often dictionary-based attack applications allow the attacker to perform any or all of the following options.

- Define a dictionary file.
    - This file could contain a relatively small subset of common words to a complete list of words for the entire language they are working with. It can even contain only 3, 4 or 5-letter words for that language depending on expected minimum and maximum password lengths.
- Allow for the definition of minimum and maximum password lengths to try.
- Allow for the replacement of common letter number combinations such as a number "`0`" for the letter "O" or "o", or the number "`1`" for the letter "L" or "l".
- Allow for the insertion of number/symbol combinations, both in front of, or after the dictionary word.

The more options that an attacker turns on the longer the cracking process generally takes. With all of the options set to a maximum setting and using a large dictionary file, it could take a very long time to run through the complete set of word/number/symbol combinations.

Finally the rainbow tables method uses pre-calculated hashes for Microsoft Windows LM or NTLM-based passwords. These hashes are stored in indexed files called rainbow tables and they can be downloaded from the Internet and often comes with the password cracking software itself. They can also be generated by the attacker using many different Rainbow Table generators.

For the attacker to use the rainbow tables they often must have either direct access to the system they cracking or have somehow gotten the Microsoft Windows SAM (Security Accounts Manager) file off of the system. The SAM file is normally a system protected file and while the Microsoft OS is running is generally locked and prevented from being copied. However, as with

the password cracking software itself there are many tools with which allow an attacker to remotely access and copy these files off of their target systems.

Today, with many changes made by Microsoft to help protect system passwords, including the option of using non-reversible encryption of the passwords stored in the SAM file. The capabilities of cracking system passwords remotely, without direct access to the target system, has been somewhat reduced.

On the other side of that coin however, the attackers have also modified their attack strategy to allow them to collect the password information they need. Reality is that they attackers don't even need the Administrator password; all they really need is a low-level account to gain initial access to a system.

Really, if the attacker is only there to remotely steal data from of a system, gaining Administrative rights is often overkill. The low-lever user account that they have access to, really only has to be granted rights to the data store where the data is located or be able to talk to another system on the network with a user account that does have access to the data they need. From there the attacker can take whatever they want.

Two of the most common applications that are used to crack passwords are "Cain and Abel" and "Ophcrack". Cain and Abel is able to crack passwords using multiple methods including brute force, dictionary and rainbow tables, while Ophcrack just uses rainbow tables. As seen in Figure-2; Ophcrack took less than 5 minutes to crack the majority of user passwords taken from this SAM file which was "captured" from an unprotected Microsoft Windows 2003 test server, and it only used the two smallest rainbow file sets that are available from the Ophcrack developers. At this speed the attacker doesn't even have time to get himself a fresh cup of coffee.

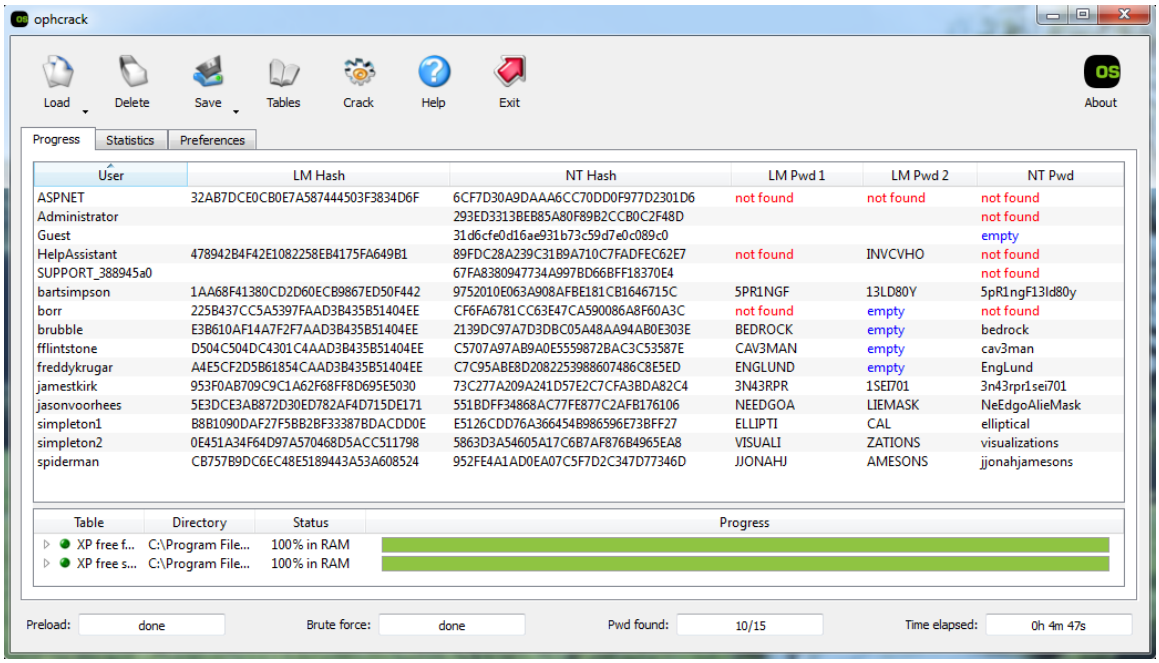


Figure 2: Ophcrack Password Cracker

Buffer overflows

It doesn't matter what hardware, operating system or application the IT department in any organization is working with; buffer overflows still remains one of the biggest problems that have to be addressed. Basically, a buffer overflow is the result of poor programming (application design standards), poor data validation and poor memory management on behave of the software developers. So what is a buffer and a buffer overflow?

Nearly every IT person will, including programmers, will define a "buffer" as a defined amount of space that has been set aside in memory for the use of the running application that set up the buffer. While this is true at the basic level, it is not just user applications that have buffer overflow issues. The operating systems (even at the kernel level) use many buffers in providing the desktop experience to the users, core operating system functions like the TCP/IP stack use buffers and even the system hardware such as network/graphics cards and serial/parallel ports will create and use buffers on the system.

Really, at its basic level, the Microsoft "Ping-of-Death" attack was a simple buffer overflow vulnerability. For the attackers, the whole purpose for issuing this type of attack was to either crash the system's TCP/IP stack or the entire system, effectively creating a simple DoS attack. Until it was patched, the attacker only needed to issue a single packet to the target system to invoke the attack. Resulting in huge headaches for Administrators who had find and implement a work around to prevent the attacks while still allowing ICMP messaging to work.

It is the "defined amount of space" where we start to visualize the buffer overflow issues. Basically a buffer overflow is caused when the "defined amount of space" is sent more data than it was "designed" for. In other words, someone entered 25 bytes of data into a buffer that was only defined to hold 20 bytes. So what happens to the remaining 5 bytes of data? Well the real answer to this question is "It depends…,"

Earlier, we stated that a buffer overflow is the result of poor programming (application design standards), poor input validation and poor memory management on behalf of the software developers. That said, there are really four scenarios that can come out of our example:

- The program will report back an error to the user, and possibly the system admin or DBA, and writes nothing to the buffer.
- The program only writes 20 bytes of data and drops the remaining 5 bytes.
- The program verifies that the data entered into the field is too long and creates a new buffer to handle the excess data.
- The program writes the entire 30 bytes of data to memory.

The belief in the first scenario is what we all hope for; it's what helps the DBAs and Administrators sleep at night. That the organization that designed the software had great application design standards (every module has appropriate error checking, etc.), excellent input validation (user cannot enter more data into the field than the buffer can hold) and fantastic memory management (every buffer is designed with canary values, discussed later, in place for each buffer), the user would most likely receive a message pop-up from the system identifying where the issue was and why their record cannot be saved and it writes nothing to the buffer(s).

However, the existence and even proliferation of buffer overflows in systems and applications still makes the first scenario a rarity. It's not that the application developers don't want to prevent buffer overflows, it's that the complexity of the applications and operating systems (and with today's move towards cloud-based services and application/data integration), and the tight deadlines for getting the applications to market often leave out the opportunity for truly thorough testing that might have discovered these vulnerabilities. This can be seen in the number of patches and updates released by many of the software vendors today.

The second scenario is really a stop-gap or short-term solution to resolving the issue. Rather than fixing the problem fully, the developers patch the issue with a quick fix error checking subroutine that drops any data that extends beyond the buffers specifications. This practice is usually put in place to fix the issue "now" and a full and proper fix is often implemented in either a larger service pack or major application upgrade release.

The third and fourth scenarios are truly the result of the poor programming practices, input validation and memory management. These are the true buffer overflows that allow attackers to gain access to a system and take it over. In the third scenario the creation of a new buffer is often a violation of data validation and data integrity rules of the database designed to store the data long-term. If the buffer is only 20 bytes in length then it is likely that the database field that stores this data is also only 20 bytes in length.

While the data entry application may be able to create a new buffer at will to handle its overflow issues, when it goes to write the data to the database it now causes new issues on another system, mainly the database server, which cannot accept the data. These issues can include crashing both the data entry system/application and/or the database server itself, causing deadlocks on the database server (the equivalent of a basic DoS attack on the database server) and even resulting in the database server suffering from its own buffer overflow.

Even if the original data entry system/application that accepted the data only writes the expected 20 bytes to the database server (as it should), it often abandons the "new" buffer that contains the addition 5 bytes of data because it cannot do anything with it. If this application doesn't have a proper clean-up routine, it results in a slow memory leak on the system which eventually causes it to run out of memory and crash.

The fourth option is what the hacker is really trying to exploit and specifically gives the hacker the ability to destroy data integrity and/or gain full root-level access to the system. Most often when an attacker attempts to use a buffer overflow vulnerability, they have an understanding that they are really trying to "crash" the application and cause it to give them shell (or root) level access without having to authenticate on the system.

This is done by specifically sending command string information in the data that over flows the buffer. This excess data is written into area usually reserved for special variables such as the return value and the Stack Frame Pointer, and is most often located between the user-defined stack variables in memory and the user-defined heap variables in memory.

The return value being of critical importance here in that, it provides the application with the return value (application memory location) from the function that called it, and to where the program needs to return to, in order to continue executing the next line of code in the application. With this return value overwritten with a different command string the attacker has now pointed the return value to a different memory location (the first part of the command string) and attempts to execute the remaining shellcode (the remaining command string) that was entered as part of the buffer overflow data through the system's command interpreter.

The attackers also know that this form of attack doesn't always work the first time. Once they can actually perform a buffer overflow, they know that it will cause either the program or the system to crash with somewhat predictable results. It is the somewhat predictability that allows them to continue their attack often unnoticed by the DBA or Systems Admin staff who are trying to get the system or application back online as quickly as possible. These predictable results are:

- The memory location that was written in the buffer overflow was not correct and not linked to any critical system function. This would most likely cause the application to crash resulting in minor data entry loss if it was a local (desktop) application or resulting in loss of access to the application by all users if it is a server-based application.
    - Often the crashing of the application does not provide the administrative staff enough information to determine what caused the crash itself. Typically it was recorded into the system logs or appeared as an error message on the screen as an "Unhandled Exception", "General Protection Fault" (GPF Error), or a "Segmentation Fault", depending on the operating system that application was running on.
    - The lack of effective error messages to identify the cause of the crash as a buffer overflow attack leaves the attacker with the ability to try again once the application is back up and running.
- Caused the entire system to crash because the memory location that was written into the buffer overflow was not correct and remaining command string in the buffer overflow data overwrote critical system memory.
    - Again the various error messages that resulted on the system from this would provide little to no information for the administrative staff to readily identify the crash as being caused by a buffer overflow attack. Reality is most DBA's or other Administrator's first and primary task will be to get the system back up with as little downtime as possible.
    - Again the lack of effective error reporting leaves the attacker with the ability to try again once the system is back up and running.
- The memory location was correct and the remaining shellcode executed without issue, resulting it the attacker having the ability to issue command to the system remotely as a root-level user without having to authenticate on the system.
    - With this level of access, the attacker can install Remote Administration Toolkits (RATs) which will allow them access onto the system at any time. They can also create new users with elevated system rights, which would again give them access to do whatever they wanted to the system and its data.

Really the process to fix these kinds of vulnerabilities is through more stringent programming practices, input validation and memory management. Application developers and must start writing cleaner, more secure code that has better error handling and processing techniques.

They must also work more closely together with the operating system and hardware developers to develop better memory management and partitioning of the memory so that application buffer space cannot be used to pass executable code to the systems command interpreter.

The implementation of "Stack Canaries" is one method of further partitioning the memory with a protective barrier against buffer overflow. A canary is a small random value that is placed between the user data (buffer) and the stored return value. The application needs to be designed to check if the canary value has changed before proceeding with processing the buffer, if the value has changed (i.e. the buffer has overflowed) the application halts and exits the processing routine.

Stack canaries have been in introduced within Microsoft's Visual Studio .Net as part of the development suite and gcc by patching it with "StackGuard". Implementation of these tools will help in the longer term; however the developers must actually use these tools before they can do any good.

SQL Injection

SQL injection attacks are a form of injection attacks that specifically target SQL database servers through web-based applications. The objectives of the attack are most often to retrieve confidential data, such as customer names, credit card numbers, social security or social insurance number from the database. Although these attacks can also be used to cause DoS attacks by shutting down a server or destroying data by sending commands that delete data or even tables within the database.

The stealing of data is typically the greatest concern to organizations since the information can be used for identity and/or financial theft. Business losses, caused by the shutting down of the server or the destruction of data, are also of great concern. However since most organizations have modern backup systems and can recover relatively quickly from these attacks, the loss is mainly restricted to financial impacts.

The ability to implement these forms of attacks really comes down to how the systems are configured and secured, and how well the web-based application is written to prevent these forms of attacks. To conduct an attack, the attacker would first assess if the application and database are susceptible to this type of attack. This can quickly be determined just by entering a single quote character ('), into the query string of a URL.

If the system is open to an SQL injection attack the attacker will receive an ODBC error message back. Typically, the return of such an error message means that the scripts that are running on the web-based application are open to modification and thus, the attacker can modify them to inject SQL query strings and SQL server commands that does whatever they need to do.

Basic examples of an SQL injection include:

```
SELECT Username FROM Users \o /<enter webserver folder path here>/…/ .user1 ; --‘
```

- This statement directs the system to print out all of the Usernames in the User table and store them in a folder on the webserver at the path that they have defined.
    - If the attacker has done their preliminary scouting well enough they would have found a folder on the webserver which contains documents that they can easily access because it is under the document root of the webserver and therefore not protected.

```
SELECT * FROM Customer_CC \o /<enter webserver folder path here>/…/ .custcc1 ; --‘
```

- Like the statement above it, this statement directs the system to output the data out to a file. However, this file now contains all of the customer credit card numbers stored in the table, which we are going to hope was properly encrypted.

```
SELECT * FROM Users WHERE USERID = ‘’ ; DROP TABLE Customers -‘
```

- This statement appears to be asking for all of the user information from the User table for a particular User ID. However, where a normal User ID would be entered (e.g. USERID = 12345) an SQL command has been inserted. It is a legitimate and valid SQL statement in most database engines and likely won’t even generate an error when the statement is processed.
    - If an error is generated it is most likely going to be the result of the Customers table is being modified when the statement executes, or the user does not have access to the DROP TABLE command and/or the Customer table.
    - From the attacker’s point of view this command is wholly destructive. They have just removed the Customers table from the database and literally brought any legitimate business to a halt. Customer orders cannot be entered, new customers cannot be created on the system, billing and invoicing cannot be performed, etc.
    - This didn’t have to be a destructive statement, it could have just as easily been any legitimate SQL statement, including statements that can alter product prices or inventory levels, shut down the server, reveal the database admin user name and password or execute stored procedures on the system.

Like buffer overflows the process of eliminating SQL injection vulnerabilities requires more stringent programming and data validation practices and implementing better database administration such as:

- the “sa” account is given a proper password (many are left blank)
- verbose messaging from the server, which provide the attacker with information on the results of their commands, is disabled
- ensures that the web applications that connect to the server have the minimum amount or privileges they need to perform their routines
- preventing invalid data from being entered into the web application through data analysis and validation processes
- using predefined data selection fields where possible and enforcing only valid SQL statements which contain the validated data from these fields.

Common System Protection Methods

Intrusion Detection/Prevention Systems

IDS generally stands for Intrusion Detection System, which is a hardware device or software application that is specifically designed to monitor a network segment or system looking for malicious activity and generate a report and/or alarm for the Administrator to investigate.

The initial IDS solutions were developed in the early 1990s when hubs were still the mainstay of network connectivity. These IDS's were passive devices that only monitored the network (broadcast domain) it was on, much like a network sniffer, and did nothing to stop an attack if it identified one.

With the advent of switching technologies, more modern IDS systems were designed like firewalls with the data being able to pass through the device so it could monitor the traffic of a particular backbone segment or internet link without having to configure port mirroring on the network switch. These solutions still did nothing to stop and attack, they just continued to monitor, report and alarm.

Eventually they evolved to the current form of IDS which is now called an IDS/IPS or most often just IPS (Intrusion Prevention System). This new form could both monitor the network and under the right circumstances, halt/prevent or limit the effects of an attack as it is occurring.

Today there are many forms of IDS solutions available and they are often being included in firewall and anti-malware solutions. Most mid to high-end firewall manufacturers (Cisco, Cyberoam and CheckPoint) have included some form of IDS into its features, and most off-the-shelf anti-malware (anti-virus) software such as McAfee and Symantec now includes an IDS module within their application.

It is nice to think that we are becoming more capable in protecting, monitoring, reporting/alerting and even preventing attacks with these powerful solutions. However, as these devices became better, the hackers adjusted and evolved their attacks, finding ways to avoid and/or circumvent these solutions.

As one can imagine, with so many vulnerabilities and design flaws in modern IT systems today, there are many ways to fool a system designed to look for these attacks by modifying the "signature" of the attack. These signatures are basically a fingerprint of what an attack looks like to the system.

IDS/IPS systems generally detect attacks using one of three methods:

- Behavior-based Detection (also referred to as Anomaly-based Detection)
    - The IDS/IPS learns what "normal" network/system behavior pattern are over a period of time. And defines a baseline set of parameters such as bandwidth loads, CPU utilization, user login/logout patterns, etc., that it expects the network or system to operate within.
    - If any of these activities falls outside of the "normal" behavior patterns then an alarm is generated identifying a problem to the system Administrator, and if it has

the ability will perform whatever pre-defined actions are built into the system to stop the attack or limit its impact.

- Pattern-based Detection (often referred to as Signature-based Detection)
    - The more common of the two types of IDS/IPS, pattern files that relate to various attacks are stored in a database. As the IDS/IPS monitors for an attack it compare current traffic patterns or system activities to the database of "known" attack patterns.
    - Like the behavior-based system, if it gets a match against the database, it reports/alerts on the attack and if it has the option, will perform the pre-defined actions for stopping the attack or limiting the attack's impact.
- Protocol-based Detection (also referred to as Protocol Anomaly-based Detection)
    - Usually installed on Web servers, this IDS/IPS looks only for abnormal protocol activity which could identify a protocol-based attack, such as a SYN Flood attack.

Based on the differences in these forms of detection, the attack signatures for each of these solutions are greatly different and even the activity that they will report and alert on are also different. For the Behavior-based solution, a dramatic change in increased traffic loads or number if login request would set it off. However this could be legitimate traffic created by the merger of two city offices into one office thus effectively doubling the staff, network traffic patterns and login request. This would force the Administrator to put the system back into learning mode until it re-establishes a new baseline for what is "normal" behavior.

Yet, if an attack is implemented slowly enough, over a period of several weeks or months, the attacker may be able to alter the slowly adjusting "normal" behavior patterns to allow their attack activity to go unnoticed by the IDS/IPS. For this to occur, an attacker would be looking for a large return (monetarily or otherwise) to invest so much time performing their attack.

For the Pattern-based solution, the recording of previous attack signatures into a database provides very few false alerts. The signatures stored in the database are like fingerprints in the fingerprint database used by law enforcement agencies all over the world. Like fingerprints, the patterns stored, contains very specific details about each attack and when a match is made it is very likely that an attack is actually occurring.

Pattern-based IDS/IPS solutions do have their own inherent weaknesses. Unknown attacks that haven't been entered into the database and "Zero-day" attacks that have just emerged can be completely missed by these systems. As a result, many of these solutions also have a heuristic module built in.

This module basically looks for traffic patterns that contain similarities to the signatures in the database. However, the drawback is that processing requirements are often increased and there is also a corresponding increase in false-negatives (alarms that are not really attacks) generated by the system.

These false negatives often have the Administrators looking for problems that are not really there, until they realize that the problem was a "phantom" of the system. If there are too many of

these alarms the Administrators will eventually stop relying on or trusting the system. They may even turn the heuristic module off, if they are able.

Finally, Protocol-based solutions detect abnormalities in the protocol activity. It does this by keeping and continuously inspecting the state information of a communications stream. As this system looks into the data stream itself, it has high-level access to look at each of the protocol fields in a packet and can identify violations of the protocol rules.

This type of inspection is really based on the protocol specifications as defined in the RFCs written about the protocol. However, these RFCs do not always contain a complete specification, but more of a good "base" from which to build the protocol and further refine its specifications.

Even after the protocol has been around for a while and implemented on a large scale, there is always the opportunity to enhance the specification and as a result the Protocol-based IDS/IPS may not have a complete set of protocol "rules" to verify or validate a new form of attack. Because of weaknesses or lack of definition in a protocol's specification, this system may not have a rule to measure against when it comes to a new form of attack.

One thing that can be certain is that IDS/IPS solutions, like the attacks they try to defend against, are ever evolving and the features of each type of IDS/IPS are overlapping into each other. The example provided earlier under the Pattern-based, regarding the Christmas Tree attack, could also be found by a Protocol-based solution, simply because the attack breaks the rules of the protocol's specifications. Both solutions find the attack by using different methods. These solutions are also being implemented to work with the ever-evolving changes in communications as well.

The move towards Unified Communications is forcing the IDS/IPS solutions to now detect attacks in VOIP and SMS messaging systems. Protocol-based systems are now, not just protecting Web servers, but have evolved into Application Protocol-based IDS (APIDS) solutions which inspect the protocols at the application layer and now monitor the traffic between the Web server and the SQL server behind it.

In recent years, with the advent of new attacks that can compromise SCADA and Industrial Control systems, such as Stuxnet, IDS/IPS systems are venturing into areas, which were traditionally thought of as "safe" from attack. The truth is, attacks on these systems, were inevitable as more and more of them were being connected to the internet and remotely monitored and managed.

Security specialists had been warning industry and the government for years regarding the eventuality of these attacks, yet people were surprised and even caught off-guard when these attacks were found to exist, there has been a scramble ever since to provide solid IDS/IPS and other security solutions for these systems.

Firewalls

At its basic level, a firewall is a software application or a hardware appliance that acts as a filtering device between two networks. It allows packets from approved sources through its

interfaces and blocks packets from unapproved sources. A source can be any system on the network or internet that can communicate with any other system through standard communications protocols.

The approach used to approve or deny a source's communications traffic are varied, based on the design of the firewall, from very basic protocol filtering to complicated multi-method systems based on protocols, IP Address Ranges, system certificates, etc. However, all of them use some form of rule-based or policy-based (or combination) approach. Most firewalls have a default set of rules or policies defined on initial start-up, and these are adjusted or added to by the organization's security or Administrators.

Firewalls, like IDS solutions, are broken down into several standard categories;

- Proxy-based Filtering (also referred to as Application-based Filtering)
    - These systems fully terminate both ends of a communications session on the firewall itself, which gives the firewall full access to the communications stream and allows it to perform validation and policy enforcement of the communications stream as defined in its firewall rules and/or policies.
- Stateless Packet Filtering
    - These systems provide filtering on specific protocol header values, with each packet inspected separately and considered independent of other packets. If the specific header values are not met the packet is simply dropped. Commonly, this type of firewall looks at:
        - IP source or destination address fields
        - IP option fields
        - Transport layer protocol fields
        - Packet fragmentation fields
        - TCP/UDP source or destination port fields
        - ICMP type and code fields
    - One of the most important things to remember about these types of firewalls is that they do not retain the session state information of the communications streams. This allows a performance boost over other types of firewalls but limits their effectiveness as a high-end firewall.
- Stateful Packet Filtering
    - An improvement over the previous Stateless Packet Filtering firewall, this type of firewall maintains the full state information of the protocols that pass through it, for both session and non-session oriented protocols. It maintains this state information for two primary reasons:
        - The prevention of unsolicited requests for the external (unprotected) network to the internal (protected) network.
        - To correctly inspect packet data that resides at and above the network and transport layers such as IP Fragmentation and out-of-order TCP sequence numbers, as well as application-level states such as control channel and data channel information.
- Deep Packet Inspection Filtering

    - These types of firewalls are even more adept at tracking the state of the application layer and the validation, inspection and filtering of the data at the application layer.
    - For many of these firewalls, it performs these higher-layer functions in much the same way as an Intrusion Detection System, which gives this type of firewall the advantages of a Stateful Packet Filtering firewall and IDS combined.
        - The primary drawback on this type of firewall is that it is often more CPU and/or memory intensive and can reduce the speed of the communications stream as the systems spends more time inspecting and validating the packet's data.
- Web/XML Filtering
    - A specialty firewall, as their name implies, these types of firewalls are specifically designed to protect one or more web servers and their infrastructure from HTTP and XML-based vulnerabilities.
    - The increased complexity of web-based applications, the increased use of the HTTP protocol and the increased use of XML services for structuring and sending data has resulted in the rise of this form of firewall.
        - The most important thing to remember here is that these firewalls are function specific. They are great at what they do, protecting web servers and services. Beyond that, they do not offer much protection.

Honeypots

Honeypot systems are "designed" systems, used for the purpose of attracting, monitoring and reporting on the activities of hackers. But what do we mean by "designed"?

Generally, these environments are developed by Administrators to provide the hackers with an easy target and distracts the attacker from looking for the "real" critical systems. The design is such that there is no real data to gather or steal on the systems, but they are complex enough to keep the attacker occupied with trying to get at this artificial data.

The honeypot will typically have enough alerting capability to notify the Administrators that someone is searching around, and allows them to analyze the attacker's methods. As stated earlier, the systems may look very real and may use the actual applications employed by the organization to prevent the attacker from discovering too quickly that they have been duped.

Getting onto the honeypot is generally easy. Once inside, if the organization uses their real applications and normal security routines, the attacker may find that their progress has slowed significantly. However, from the attacker's perspective, they have a way to get to the internal systems and now have all the time they need to harvest the data for the big score. The data itself may also be simulated well enough to not have any real value, but can give the attacker enough motivation to keep trying to get more.

All honeypots also log all or nearly all information related to the attack for later analysis, however the data that is gathered also depends on how much the honeypot interacts with the attacker. If the honeypot provides minimal interaction with the attacker, referred to as a low-

interaction honeypots, there is only so much information the attacker can gather from the system and a limited number of actions that the attacker can take against the system.

This is because low-interaction honeypots only emulates actual systems, as a result the responses from these emulations are limited. This limitation also makes it easier for seasoned attackers to determine that they are interacting with or have gotten onto a honeypot, at which point they will quickly cease their attack, resulting in limiting the amount of data that can be acquired for analysis of attacks.

High-interaction honeypots, will typically use real systems, and provide for more responsive targets and feedback to the attacker. These systems can gather much more information on the attacker, the techniques and tools they use, etc. With the great increase in virtualized systems, these honeypots can simulate the real environment very quickly and realistically using a minimal set of hardware.

Honeypots themselves, really have two modes of operation:

- Research Mode
    - This mode is employed to collect data on the motivations, tools, and techniques employed by the attacker. It can also be used to identify threats to the real systems by helping to identify new or emerging attack trends.
- Production Mode
    - This mode is employed to detect, respond to and prevent future attacks to the real systems. To accomplish this, the honeypot is designed to impede the attacker's capability to get to the data and causes them to interact more with the honeypot's systems.

High-interaction honeypots that have been expanded or grown large enough to become interconnected sets of honeypots are called "honeynets". These honeynets provide a greater attack surface and more open access ports for attackers to find, it also makes it more difficult for the attacker to determine that they are in a honeynet since there are other areas (networks of systems) for them to break into and explore.

IDS/IPS, Firewalls and Honeypot: Issues that allow for discovery and evasion

The first issue is that nearly all of the advances that come with the latest and greatest IDS/IPS and firewall solutions come from the attacks of the past. Reality is that these devices can only defend against that which is already known, while attackers are always developing new methods, techniques and tools to perform their attacks.

This leaves the defensive systems, and those that develop them, in a continuous reactive state. The attackers create a new attack, the defenders create a new signature, firewall rule, etc., to stop the attack, the attackers modify their attack, the defenders modify their signature to stop the "new" attack…, it really is a repetitive cycle.

It is also not surprising that the attackers also spend a great deal of time and effort on trying to evade these defensive systems in the first place. After all, they know the IDS/IPS and firewall is likely to be there, so even before attempting an attack, they may try to determine the defensive

capabilities of the systems they are going up against. Perhaps, the IDS/IPS doesn't have the latest signatures, maybe the firewall has an open TCP/UDP port that can be used to their advantage…, etc.

To top all of this off, there are dozens of software solutions available on the Internet which are designed to perform "penetration tests" of IDS/IPS and firewall systems. These tools generate packets that simulate various attack situations or vectors to determine the capabilities of the IDS/IPS and firewall systems. They can also be used to test the Administrator's capabilities in identifying a root cause and responding to different attack scenarios.

Many of these tools can perform numerous attack scenarios and which can include;

- Session splicing
- Fragmentation overlapping or overwriting
- DoS and DDoS
- Insertions and
- Payload obfuscation
- TCP/UDP Port scanning

This is just a shortened list, which depending on the tools used can be hundreds of attacks long. However, for the purposes of "testing" these tools can provide the Administrator with a much greater understanding of their IDS/IPS and firewall solutions, helping them "tweak" their system to achieve the best possible performance vs. security stance. It also gives them a better understanding of how an attack works, how their systems respond/react to the attack and even what to look for in identifying an attack.

However, the difference between a "testing tool" and a "hacking tool" is whose hands it is in. Unfortunately, it is not in the hands of the Administrator often enough.

A second issue is that the attackers also know that many organizations do not regularly update or replace their defensive systems. Many organizations will replace these systems in much the same way that they replace their servers, giving these systems a three to five year life cycle, even longer with smaller organizations that look at five to seven year life cycles on their IT equipment.

These older defensive systems can only perform so well against newer attacks, and even a slightly modified older attack may make it through some of them. Add to that, the continuously increasing size of the IDS/IPS signature databases, firewall rules, etc.; it leaves these older systems in a vulnerable state that may make it easier to detect, evade, bypass and even directly attack them.

As the signature databases and rules continue to increase, the processing and memory requirements needed to evaluate every packet that passes through the device also increases. This causes the system to lag behind when under heavy traffic pressure, leaving the system with two real options; pass the traffic through uninspected or drop the traffic causing the connection to drop or forcing the end systems to resend the traffic again.

Neither of these options is ideal since allowing the traffic through uninspected means that a legitimate attack may get through because the system just couldn't process it in time. The reason that the system may allow uninspected traffic to be pushed through may be a result of the type of traffic and the fact that the system is trying to prevent the communications streams from dropping. Time sensitive traffic, such as VOIP, is especially vulnerable to this issue, since failure to move the traffic along quickly results in the call having severe quality issues or dropping completely.

The other option, dropping the packet either causes a communication breakdown between the end-points likely generating many system errors on both ends causing the connections to be dropped, or forces the systems to resend the missing packets. If the system is already under heavy traffic pressure, dropping packets and having the end-points resending traffic is not going to relieve that pressure it's going to make it worse.

A third issue is the attackers also know that, for many organizations, updating these devices generally requires maintenance windows and possible down time for applying patches, new software releases and even in updating the databases, rules and policies. For critical systems or systems that are operating on a 24x7 basis, getting this down time is often difficult since it costs the organization money when the systems are not operating. As a result, many of these systems may be months behind in patches, updates, new feature releases and even signature, rule and protocol updates.

A fourth issue is that attackers are always looking for things to attack, even systems that we have traditionally thought of as not attackable. SCADA and Industrial control systems were in this category for many years. Security analysts had for years stated that these systems were very vulnerable, while the manufacturers and implementers of these systems stated it is not possible.

SCADA and Industrial Control system communications, which are also considered time sensitive or control critical are installed everywhere. For many of these systems, the control signals must get through to avoid system shutdown or even a catastrophe and we see them every day without paying much attention to them. They control:

- the monitoring of flow rates on natural gas pipelines (commercial or residential),
- the electricity draw demand on main power transmission stations and substations,
- the remote monitoring of radiation levels in nuclear power plants,
- the control and manufacturing of steel or other metal smelting and forging plants,
- the control and manufacturing of oil into gasoline, diesel and other by-products,
- the control and manufacturing of common electronic and computer components,
- the control of heating and cooling (HVAC) systems in large buildings.

The list goes on and on, nearly every industry today has some of these control systems installed somewhere. Then along came Stuxnet, a computer worm identified in 2010 that is believed to have been created by the United States and Israel governments for the purpose of delaying of stopping the Iran government from being able to create nuclear weapons. This malware showed the world that the ability to attack SCADA and Industrial Control systems was very real.

Since Stuxnet was discovered, several variants and even new versions of this malware have begun to surface, leaving Administrators with the larger problem of now having to find appropriate security tools to protect the organization's SCADA and Industrial Control systems as well. Time will only tell whether these rapidly installed solutions will stand up to the pressure of future attacks.

The end result is that any organization, large or small, running a SCADA and/or Industrial Control system that cannot provide the resources needed to implement and maintain proper security solutions, will most often turn them off when they become the bottle-neck to efficient communications flow or put at risk that ability of the organization to run and/or monitor critical control systems effectively. This will leave the organization, and more importantly, their control systems at risk of attack.

Security methodologies

Security by obscurity

This is not an uncommon approach and is used both in the real and digital worlds. It is the art of blending in, of being unknown or unimportant. The objective is of course not to present yourself as a target, like blending into a large crowd where there are many other and possibly better, targets to attack.

This is possible one of the simplest approaches to security; however it doesn't really protect your systems. Automated hacking applications, viruses, worms, etc. really don't care how obscure you are, they attack all systems equally trying to gain access. In most cases, these automated tools attack every system they encounter to increase the size and capability of their botnets.

This solution may keep you safe for a small period of time, but in the end you will need additional solutions to protect your systems for long-term security. That said, many systems security departments will use this tool in their arsenal after they have their other security tools in place.

Why? The answer is simple; obscuring the importance of the systems that are internet facing and open to attack generally reduces the number of attacks that they often see. With the exception of automated tools which attack everything, people, i.e. hackers, may not detect or find obscured interfaces that do not readily respond to standard service requests.

Security by isolation/separation

This technique is rarely used and where it is, it is most often used by government, military and research installations. These facilities literally have no connection to the Internet. If they do have any connection to an outside network at all, it is setup as an on-demand type connection that is generally initiated by the isolated network and not the network it is connecting to.

This is considered the most secure type of systems implementation simply because the only way to attack it is to gain direct physical access to it. On the other hand there are considerable issues

with maintaining a solution like this. One of the biggest issues is in updating and patching the Operating Systems, software, firmware, etc.

Patches, firmware updates, service packs, etc., are all released by their vendors through the internet. If the Administrators are performing regular maintenance on their systems, they would need to manually download, scan and test each patch, service pack, firmware release, IOS upgrade, etc., for compatibility purposes, before they could install them on any system.

So what if they decide they don't need to perform this maintenance? After all nothing electronically can gain access to their systems. They are "off the net" and can't be hacked in the traditional sense.

While this is true, it again generally means more work for the IT Department. After all, implementations like this are rare and set up for a specific purpose, such as military research. They are most likely running highly customized or purpose specific "in-house" applications for whatever they are doing.

Generally, situations such as this will require that the programmers and developers of this software are also part of the research team and whenever there is a problem with their "custom" application that is rooted in the Operating System or some other off-the-shelf software that they are using; they must now build custom patches to work around the issues, not always the easiest thing to do.

While this solution is the most secure, it does have its drawbacks and is only used for very specific purposes. It is not something that is done by most organizations, simply because they need their Internet connection as a part of their business solutions. Separation simply means no business or loss of business.

Security by encryption

This is a piece of an organization's overall security solution. By itself, encryption only buys you security of data in transmission or data that is sitting in storage somewhere. It does not protect the systems that are transmitting or storing the data.

The systems themselves must be protected by other means and without these other security methods in place the data, no matter how well encrypted is open to attack if the systems that are sending or storing them are compromised.

Assuming these systems are properly secured, then encryption is "possibly" a safe means of sending or storing data without fear that someone can steal and use it for their own legitimate or illegitimate purposes. The reason that "possibly" is used here is that the strength of the encryption methods used can be very weak or very strong.

We will cover more on encryption later, however at this point we will leave it that encryption, in and of itself, is only a piece of a full security implementation.

Security by Authorization

Security by Authorization is also a piece of an overall security implementation. It cannot, by itself, provide overall security protection and must have other supporting security practices working with it.

Most people recognize Security by Authorization for both its physical and digital forms. Physically, it is the security procedures or processes that are put in place to prevent someone from gaining access to something. Security guards, swipe cards and tokens are one example of this type of security, they allow a person to gain access to a building, or area within a building depending on the "authorization" level that the person has.

For digital authorization, the most recognizable example is MS Active Directory's User Rights and Policies. With an estimated +80% of the world's business desktops and servers, it is easy to see why it is the most recognized, however it doesn't mean that it is the most secure. User rights and policies, as they are configured and more importantly, maintained by Administrators may have security holes and weaknesses.

Often these holes and weaknesses come from organizational changes. As a person within the organization moves from one role to another, their access rights and needs change. However, since they may be training the person replacing them in their old role, they gain the access rights of their new role while still maintaining the rights for their old role, and no follow-up is done later to clean it up.

A few years later, and another new role, the process happens again, and unless the role is significantly different than their current roll, the Administrator often will not remove existing rights that have been granted to a user simply because they don't know if the user still needs them or not.

The reality is, that it is the Administrator's responsibility to ask, but in a large organization with so many people doing so many jobs it is often not questioned. You may ask, so "What's the problem anyway?" or "Does it matter?"

It actually does.

Improperly managed user rights and accesses, if left unchecked, become nearly impossible to clean up later. It also leaves weaknesses in the overall security of the Active Directory structure because a compromised user account now has more access to data and systems then it should. Allowing more data to be compromised, or additional, more privileged accounts to be identified and compromised.

The simple rule here is; give only the access needed to do the job and no more. Just because the user thinks they may still need access doesn't mean they actually do. If a user really does need to maintain some or all of their former role's access, clear timelines should be put in place to identify how long they need these access privileges for and follow-ups must be performed to remove the access once the timeline has expired. Otherwise take it away immediately; it can always be given back later if required.

Defense-in-Depth (Multiple layers of security)

There are numerous defense-in-depth models available for an individual to search through on the Internet. There are also many whitepapers and books written on the subject. Some take are based on a specific focus and are based on specific equipment (Cisco Unified Defense-in-Depth) or systems (Industrial Control Defense-in-Depth-USCert). Others are based on the TCP/IP or OSI Models (OSI Defense in Depth-GIAC).

Still others are based on specific regions, countries (Critical Infrastructure Protection-Australia) or branches of government or military (Dept of Defense-USA). These documents may provide a great deal of information or a brief over view of the topic and others will delve deeper into specific areas of the various defense-in-depth model's individual layers.

Despite all of the information available, individual circumstance defines the approach a person must take when implementing any defense-in-depth strategy. There is no "one size fits all" model that will work for you or your organization and each layer of defense must be evaluated and implemented to work with each other layer in the model.

There is also no single defense-in-depth model from which to choose. There are 4-layer, 5-layer, 7-layer, even 13-layer models from which to choose. However, most models are based on a common theme and it really depends on how the author of the model chose to incorporate or break-out specific areas of defense in their model. A 7-layer model will be covered in more depth later.

References

Texts

Footprinting and Reconnaissance

Kimberly Graves. *CEH Certified Ethical Hacker Study Guide*, Wiley Publishing Inc., 2010. ISBN: 978-0-470-52520-3 [Pages 38 – 48]

Network Scanning and Queries

Kimberly Graves. *CEH Certified Ethical Hacker Study Guide*, Wiley Publishing Inc., 2010. ISBN: 978-0-470-52520-3 [Pages 64-81]

Enumeration and Operating System Identification

Kimberly Graves. *CEH Certified Ethical Hacker Study Guide*, Wiley Publishing Inc., 2010. ISBN: 978-0-470-52520-3 [Pages 81-86]

Social Engineering

Kimberly Graves. *CEH Certified Ethical Hacker Study Guide*, Wiley Publishing Inc., 2010. ISBN: 978-0-470-52520-3 [Pages 48-54]

Viruses and Worms

Victor Oppleman, Oliver Friedrichs and Brett Watson. Extreme Exploits: Advanced Defenses Against Hardcore Hacks, McGraw-Hill/Osborne, 2005. ISBN: 0-07-225955-8 [Pages 352-353]

Kimberly Graves. *CEH Certified Ethical Hacker Study Guide*, Wiley Publishing Inc., 2010. ISBN: 978-0-470-52520-3 [Pages 141-145]

Trojans and Backdoors

Kimberly Graves. *CEH Certified Ethical Hacker Study Guide*, Wiley Publishing Inc., 2010. ISBN: 978-0-470-52520-3 [Pages 126-141]

Dos and DDoS

Susan Young and Dave Aitel. *The Hacker's Handbook: The Strategy behind Breaking into and Defending Networks*, CRC Press LLC, 2004. ISBN: 0-8493-0888-7 [Pages 58, 96-98, 107, 142]

Kimberly Graves. *CEH Certified Ethical Hacker Study Guide*, Wiley Publishing Inc., 2010. ISBN: 978-0-470-52520-3 [Pages 174-182]

Botnets

Niels Provos and Thorsten Holz, *Virtual Honeypots: From Botnet tracking to Intrusion Detection*, Addison-Wesley, 2008. ISBN: 978-0-321-33632-3 [Pages 359-390]

Victor Oppleman, Oliver Friedrichs and Brett Watson. Extreme Exploits: Advanced Defenses Against Hardcore Hacks, McGraw-Hill/Osborne, 2005. ISBN: 0-07-225955-8 [Pages 138, 360-362]

Password Cracking

Kimberly Graves. *CEH Certified Ethical Hacker Study Guide*, Wiley Publishing Inc., 2010. ISBN: 978-0-470-52520-3 [Pages 102-105, 213-214]

Ronald L. Krutz and Russell Dean Vines. The *CEH Prep Guide: The Comprehensive Guide to Certified Ethical Hacking*, Wiley Publishing Inc., 2008. ISBN: 978-0-470-13592-1 [Pages 154-155, 387-391)

Buffer Overflows
Victor Oppleman, Oliver Friedrichs and Brett Watson. Extreme Exploits: Advanced Defenses Against Hardcore Hacks, McGraw-Hill/Osborne, 2005. ISBN: 0-07-225955-8 [Pages 386-392]

SQL Injection
Victor Oppleman, Oliver Friedrichs and Brett Watson. Extreme Exploits: Advanced Defenses Against Hardcore Hacks, McGraw-Hill/Osborne, 2005. ISBN: 0-07-225955-8 [Pages 384-386]

Ronald L. Krutz and Russell Dean Vines. The *CEH Prep Guide: The Comprehensive Guide to Certified Ethical Hacking*, Wiley Publishing Inc., 2008. ISBN: 978-0-470-13592-1 [Pages 328-338]

Intrusion Detection/Prevention Systems
Ronald L. Krutz and Russell Dean Vines. The *CEH Prep Guide: The Comprehensive Guide to Certified Ethical Hacking*, Wiley Publishing Inc., 2008. ISBN: 978-0-470-13592-1 [Pages 462-467]

Firewalls
Victor Oppleman, Oliver Friedrichs and Brett Watson. Extreme Exploits: Advanced Defenses Against Hardcore Hacks, McGraw-Hill/Osborne, 2005. ISBN: 0-07-225955-8 [Pages 76-91]

HoneyPots
Ronald L. Krutz and Russell Dean Vines. The *CEH Prep Guide: The Comprehensive Guide to Certified Ethical Hacking*, Wiley Publishing Inc., 2008. ISBN: 978-0-470-13592-1 [Pages 469-471]

Niels Provos and Thorsten Holz, *Virtual Honeypots: From Botnet tracking to Intrusion Detection*, Addison-Wesley, 2008. ISBN: 978-0-321-33632-3 [Pages 7-13]

IDS/IPS, Firewalls and Honeypot: Issues that allow for discovery and evasion
Victor Oppleman, Oliver Friedrichs and Brett Watson. Extreme Exploits: Advanced Defenses Against Hardcore Hacks, McGraw-Hill/Osborne, 2005. ISBN: 0-07-225955-8 [Pages 118-119]

Kimberly Graves. *CEH Certified Ethical Hacker Study Guide*, Wiley Publishing Inc., 2010. ISBN: 978-0-470-52520-3 [Pages 308-310]

White Papers and Web References

TechRepublic: Hacker vs. cracker page (Chad Perrin, July 29, 2011), http://www.techrepublic.com/blog/it-security/hacker-vs-cracker/#http://googleads.g.doubleclick.net/xbbe/pixel?d=CL0EEJA-GIWoSA

Technews Daily: The 10 Worst Computer Viruses in History page (John R. Quain, July 20, 2011), http://www.technewsdaily.com/2909-10-worst-computer-viruses-history.html

Computer Archeology: Stoned Bootsector Virus page (Chris Cantrell, last modified: April 22, 2013), http://www.computerarcheology.com/wiki/wiki/Virus/Stoned

Reuters: old worm won't die after 2008 attack on U.S. military page (Phil Stewart and Jim Wolf, June 16, 2011), http://www.reuters.com/article/2011/06/16/us-usa-cybersecurity-worm-idUSTRE75F5TB20110616

Conficker Working Group: Lessons Learned, (Conficker Working Group, January 2011), http://www.confickerworkinggroup.org/wiki/uploads/Conficker_Working_Group_Lessons_Learned_17_June_2010_final.pdf

Symantec: Introduction to Trojans and Backdoors page, (Farzad, September 20, 2010), http://www.symantec.com/connect/articles/introduction-trojans-and-backdoors

IEEE Spectrum: The Real Story of Stuxnet page, (David Kushner, February 26, 2013), http://spectrum.ieee.org/telecom/security/the-real-story-of-stuxnet

Voice of America: Stuxnet: An Effective Cyber Weapon page, (George Putic, June 28, 2013), http://www.voanews.com/content/stuxnet-an-effective-cyberwar-weapon/1691311.html

CERT: Software Engineering Institute, Carnegie Mellon University: Denial of Service Attacks page, (Author not published, June 4, 2001), http://www.cert.org/tech_tips/denial_of_service.html#history

Zimbio: Security Shield 2012 page, (Jack Lazara, February 19, 2012), http://anti-virus-software-review.toptenreviews.com/my-shield-security.html

Techspot: Zeus Trojan returns: Facebook being used to spread the infection (David Tom, June 5, 2013), http://www.techspot.com/news/52795-zeus-trojan-returns-facebook-being-used-to-spread-the-infection.html

Wikipedia: Beast (Trojan horse) page, (multiple contributors, last modified: July 25, 2013), http://en.wikipedia.org/wiki/Beast_(Trojan_horse)

Sourceforge: ophcrack page, (multiple contributors, last modified: June 13, 2013), http://ophcrack.sourceforge.net/

Oxid.it: Cain and Abel home page, (Massimilliano Montoro and Sean Babcock, last modified: May 27, 2013), http://www.oxid.it/cain.html

Speed Network Lab: Evasion Techniques: Sneaking through Your Intrusion Detection/Prevention Systems (Tsung-Huan Cheng, Ying-Dar Lin, Yuan-Cheng Lai and Po-Ching Lin, October 10, 2011), http://speed.cis.nctu.edu.tw/~ydlin/pdf/Evasion_Techniques_Sneaking_through_Your_Intrusion_Detection_Prevention_Systems.pdf

2012 IEEE Network Operations and Management Symposium: An Evasive Attack on Snort Flowbits, (Tung Tran, Issam Aib, Ehab Al Shaer and Raouf Boutaba, May 15, 2012), http://rboutaba.cs.uwaterloo.ca/Papers/Conferences/2012/TranNOMS12.pdf

TechNet Magazine: The Great Debate: Security by Obscurity, (Jesper M Johansson and Roger Grimes, June 2008), http://technet.microsoft.com/en-us/magazine/2008.06.obscurity.aspx

Learnthat: Introduction to Active Directory (Jeremy Reis, July 2, 2008), http://learnthat.com/2008/07/introduction-to-active-directory/

Chapter 2: Encryption Primer

History

Brief historical uses of cryptography

The word cryptography is derived from the Greek words "kryptos", which means hidden, and logos, which means word. Today's modern cryptology is its own science and has several subfields including: cryptography, cryptanalysis and steganography as shown in Figure-2 below.

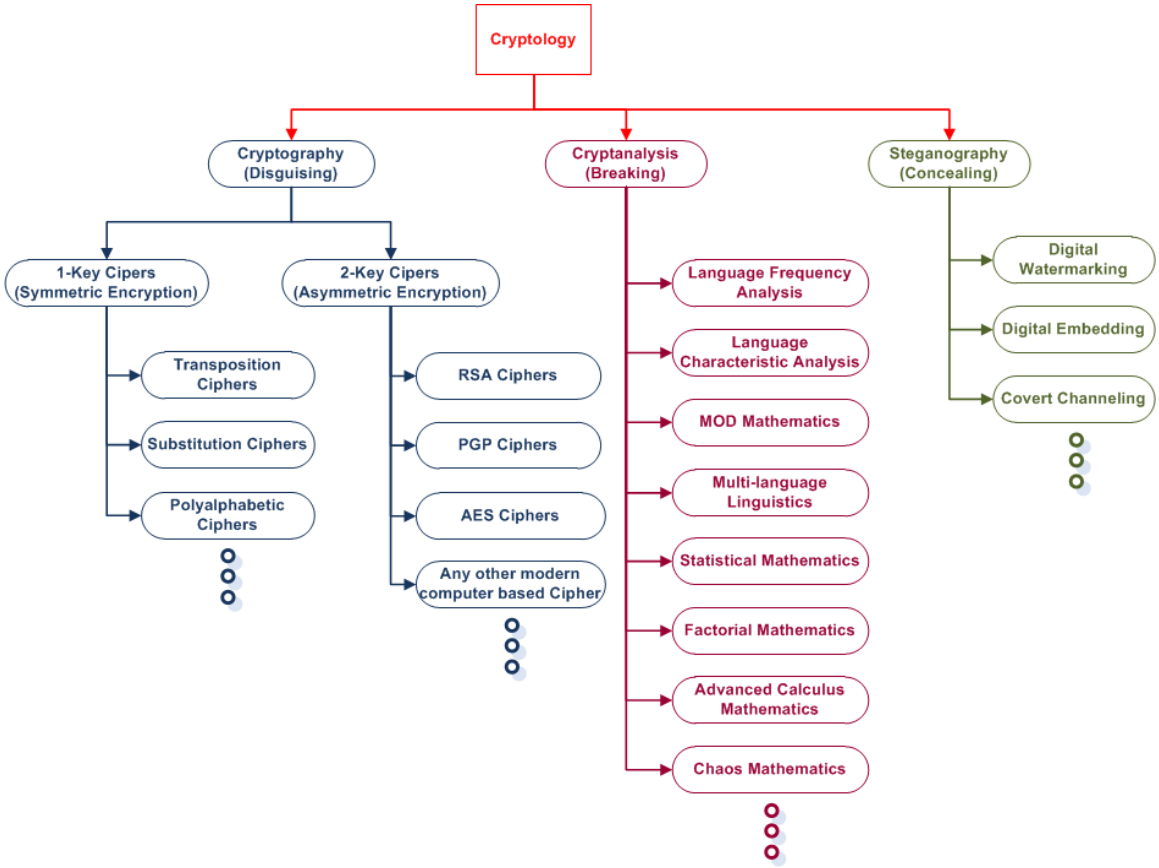


Figure 3: Cryptography Subfields

Cryptography is as old as the written language itself, and has been mostly employed for military, religious and diplomatic communications purposes and has continuously grown more complex as time has passed. To provide a brief breakdown of the history of cryptographic use we need to look back at least 4500 years, so here it goes:

- The ancient Egyptians enciphered some of their writing on monuments to disguise or obscure the proper names and titles of the people the monuments referenced. This was probably a tactic used to shroud either the name of the person or the monument itself in mystery.
- Ancient Hebrews enciphered certain words and texts in their scriptures. Again, this was an attempt to obscure the proper name of a person or place that the text referenced. Remember, "Freedom of speech" wasn't always a right as we enjoy it today.

- About 6 BC the ancient Greeks used a type of staff called a "scytale" to create ciphered messages by wrapping a strip of cloth around the staff and writing the message on it. Once unwound from the staff the cloth appeared to have nothing more than random letters written all over it.
- Julius Caesar used a simple substitution cipher that is still used today by children and in puzzles. This simple cipher technique is now called the Caesar cipher or the Caesar-shift cipher.
- Roger Bacon, an English monk, wrote about seven different cipher methods in one of his documents in the middle of the 13th century.
- In the mid-1300s Geoffrey Chaucer an English poet, often called the "father of English literature" wrote several passages of his works in cipher.
- In the mid-1400s the Italian architect and philosopher Leon Alberti devised a cipher wheel and also appears to be the first person to discuss and describe the use of frequency analysis as a tool that could be used to "break" ciphered documents.
- In 1587 Mary Stuart – Queen of Scotland was executed by Queen Elizabeth of England because an advisor of Queen Elizabeth's uncovered a plot by Mary to escape her English prison and take over the English throne.
- The 16th and 17th centuries both bore a proliferation of documents and books by philosophers, architects, mathematicians, physicists, engineers, monks, and others who designed new methods of enciphering messages or revised older methods to improve upon them. Including the book written by Blaise de Vigenère in 1585 that describes the first use of a polyalphabetic substitution cipher. Today we refer to this as the Vigenère cipher.
- The 18th and 19th centuries saw this same growth in complex updating of existing ciphers and development of new cipher methods. Including the Jefferson cylinder developed in the 1790's and the Wheatstone disc developed in the early 1800's. As always, these developments were mostly the result of a need to secure diplomatic and war-time communications.

This pattern continued until the late 19th and early 20th centuries, when a fundamental change in the communications technologies led to an exponential increase in the complexity of the encryption methods used today. That change, was the use of radio and telegraphy/telephony communications to send messages very quickly (almost instantly) over great distances.

When the methods and speed at which messages could be sent greatly increased, the need and ability to secure these messages also greatly increased. The need to be able to "break" these messages also took a higher priority for governments in the advent of both world wars; however the "code breakers" were always one, if not two, steps behind the code makers.

This could never be more evident, then in World War II, when the Germans began using a mechanical device called the Enigma Rotor Machine to encipher and decipher their war-time communications. It took the allies a long time, through the use of counterintelligence, espionage, and a large group of people with a wide variety of skills, to break the encrypted German messages. Their success was one of the keys to defeating the German army and one is left to wonder what the world would be like today had they failed.

Electronic and mechanical technology rapidly changed in the 20th century. The computer, born of these technologies, became a mainstream device used by nearly everyone today. It is the use and speed of today's computer communications that drives the encryption industry today. While still used to ensure the security of military and diplomatic correspondence, the business and financial worlds needing to keep corporate secrets and/or transactions secure, are the main users of encryption technologies today.

The complexity of the today's computer-based encryption applications leaves the cryptanalyst with little hope of breaking the encrypted messages within reasonable time frames. Which is literally the objective of encryption; while not impossible to break it would take so long to decipher that any relevant use of the information contained in the message would have long since expired.

Main Goals of Cryptography

There are really four main goals of cryptography as follows;

1. User Authentication
2. Data Authentication
    a. Data Integrity
    b. Data Origin Authentication
3. Non-repudiation of Origin
4. Data Confidentiality

User Authentication

If you are logging onto a computer, there should be some way that you are able to identify yourself to the computer to verify that you are, who you say you are. At the simplest level this is usually done through some type of UserID and password scheme that is built into the operating system. Once this information is enter the computer can verify whether or not you are entitled to log into the system and even determine, through your login rights, what networked services you are allowed to access.

This same principle applies to the communications process when one person or system tries to communicate with another person or system. The first step is for each end in the communications process to verify the identity of the other person or system that is at the remote end. As a result there must be some way for each end to prove their identity to the other end.

There are several ways to obtain user authentication. Let's take a basic example, suppose you place a phone call to your mother. When she answers the phone at the other end, her usual response is "Hello". This simple statement authenticates that the person on the other end is your mother because you recognized her voice, tone, accent, etc.

All of these minute details of a person's voice are what allows each of us to recognize the person we are talking to, even if we don't or can't see them. Typically your response back would be "Hi Mom. How are you?" Again the identification and recognition process is repeated for your mother and she would use the same technique to recognize you as her son or daughter.

While this is a simple example, and is by no means foolproof, it does illustrate the process that must be followed for basic authentication to occur. There are several ways in which computer systems can authenticate you as a user. Generally, you would give the computer information that only you would know or possess such as a UserID and Password combination, a PIN code, a smart card that has been assigned to you, a thumbprint scan, and so on.

Once the user has completed their authentication process then the computer should provide authentication back to the user who just logged on. This is even more important if you are accessing a remote system with your computer, and is basically the same as you responding to your mother's "Hello", in the above example.

This process of both parties identifying themselves is referred to as *mutual authentication* and it helps reduce the probability that one end or the other is faking its identity as to who or what it is. Again, none of these techniques are foolproof and identities can be "faked", but this takes time and effort. The more complex the authentication process the more time and effort will be needed to "fake" the user's identity.

Data Authentication

Data authentication is as important as user authentication and consists of two parts; data integrity (ensuring the data hasn't been modified) and data origin authentication (ensuring you know who the sender of the data is).

Data Integrity

Data Integrity guarantees that the contents of the messages that are being sent back and forth between the two communicating users or systems is unchanged and has not been tampered with. A prime example of this today is the email systems that are used around the world to send email messages across the Internet. For the most part the Internet, basically the communications channel, offers no security to the users of these email systems.

This means that anyone can "tap" into the communications channel and pick up these messages as they travel across the Internet. Once they have the messages it requires very little skill and knowledge to read and modify the message. It is important to keep this in mind, as most people and organizations consider these messages to be private and even confidential; this is despite the fact that they are sending them over public communications channels.

Consider the following example of Bob and Alice sending an email to each other.

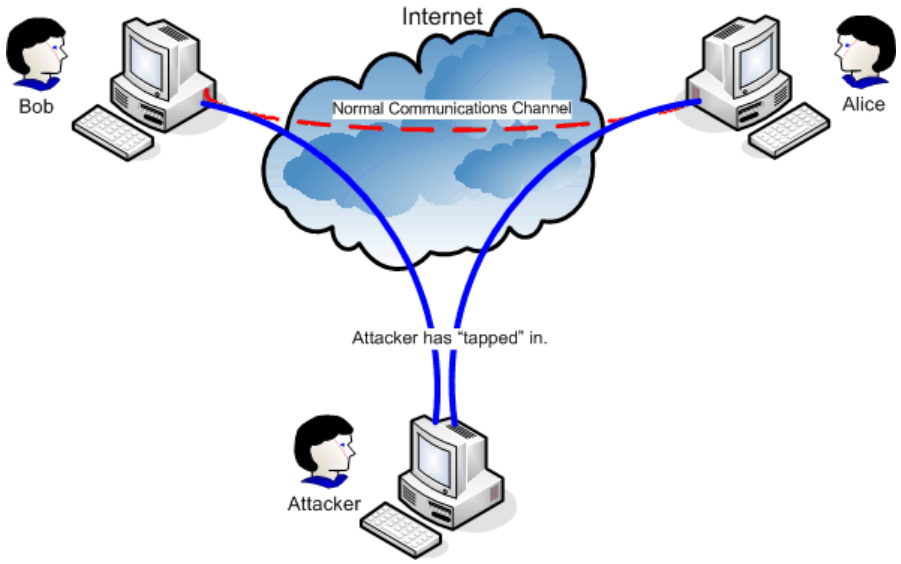


Figure 4: An active "man-in-the-middle" attack

Bob sends a message to Alice and it should pass through the Internet's infrastructure untouched on the normal communications channel. However, an attacker has tapped into the path that Bob and Alice's information travels using some form of active "man-in-the-middle" attack, as seen in Figure 3.

The attacker's intentions and purpose for tapping into Bob and Alice's communications stream are not readily known, be it for legal or illegal purposes. They could have tapped in for the purpose of reading the information to learn what is going on or modifying the message which would cause the information that Alice is receiving to be incorrect.

Assuming Bob and Alice supported data integrity, then the attacker could not change the message and forward it on to Alice as there will be indications that the message was tampered with. However if they don't support it, then Alice will receive the message and assume that the information contained in it is exactly what Bob sent.

By itself, data integrity only helps you determine that the data is unchanged and does nothing to ensure that it came from the right person or system. For that reason it must always be combined with the data origin authentication process.

Data Origin Authentication

Data integrity by itself is great in ensuring that the messages sent between Bob and Alice remain unchanged however it does nothing to verify that the messages received by Alice actually came from Bob and vice versa. As a result the attacker could be simply picking up the messages that Bob and Alice are sending to each other and creating their own messages that they send to Bob or Alice.

This defeats the data integrity checking as the messages are not modified, but are original messages. For example, Bob sends Alice a message that reads:

Hi Alice,
Meet me at Joe's Diner on 4th Avenue and Main Street at 10 PM!
Bob
This message is picked up by our attacker who is pretending to be Alice for any message that Bob is sending out. The attacker then creates and sends a new message to Alice, while pretending to be Bob. The new message reads:

Hi Alice,
Meet me at Joe's Diner on Main Street and 4th Avenue at 10 PM!
Bob

Without data origin authentication, Alice will believe that the message she is getting from the attacker is coming really coming from Bob. The attacker is pretending to be Bob and the message is passing the data authentication checking process because it is an original and unmodified message. Likewise Bob believes that the messages are getting to Alice safely because any message he receives in reply from Alice are also unmodified and passing the data authentication process.

The problem is that the attacker, hiding in the middle of the communications path, has properly pretended to be Bob from Alice's point of view, and Alice from Bob's point of view. To verify that he was successful in tapping the communications between Bob and Alice, the attacker only modified the order of the address for the meeting. The attacker only has to now wait at Joe's Diner to see if Bob and Alice meet at 10 PM as the message indicated.

If Alice and Bob had employed both data origin authentication and data authentication the attacker would have been quickly detected. This is because the complexity of defeating the dual process of data origin authentication and data authentication increases dramatically for the attacker.

Why?

Well, if the attacker only modifies the original message from Bob and then forwards it onto Alice, it may pass the data origin authentication process but it would fail the data authentication process (it's not the same message). If the attacker intercepted Bob's message and then creates a new copy to send to Alice, it fails the data origin authentication process (it's not from the correct sender) and passes the data authentication process. Basically, the attacker has to somehow defeat both systems for his attack to be successful.

Note: Some might argue that it is difficult to perform this type of active man-in-the-middle attack or wire-tapping attack and they would be correct in their argument. However, these types of attacks have been successful in the past and even today it is only a matter of money and time that prevents these types of attacks from occurring on a broad scale. Money and time is something

that is not a typically an obstacle for many organizations; be they criminal, legitimate or government.

Non-repudiation

Non-repudiation prevents Bob and Alice from denying that they were involved or participated in any communications with each other. Basically, non-repudiation with proof of origin protects Alice against any attempt by Bob in declaring that he didn't send the message to Alice. Likewise, non-repudiation with proof of receipt protects Bob against any attempt by Alice in denying that she never received the message from Bob.

Let's look at a simple example of how this works. We will assume that Alice owns an online Widget sales company. Bob visits Alice's website and places an order for 100 Widgets using his credit card. To place the order Bob has to enter in several pieces of personal information, including his email address and shipping address.

To complete the transaction Alice's website server processes the order and sends a confirmation number to Bob's personal email address that requires Bob to go back to Alice's website and enter the confirmation number manually to finalize the order.

For Alice it is very important that she can show to an independent third party that Bob really ordered the 100 Widgets. Hence the reason, that Bob must enter the confirmation number that was sent to his email address in order to complete the order, otherwise it would be easy for Bob to deny the purchase of the goods.

Similarly, for Bob it is very important that he be able to show an independent third party that Alice received the order for 100 Widgets. The receipt of the confirmation number, sent to Bob's email address prevents Alice from denying that she every got an order from Bob.

Data Confidentiality

Data confidentiality is used to protect against unauthorized disclosure of the information contained in the message. Let's go back to Figure 3, and assume that Bob and Alice have been sending each other messages with confidential business information in it.

Due to the confidential nature of the correspondence Bob has chosen to encrypt the message in a manner that has been previously agreed to by both Bob and Alice. The reason for this is to prevent anybody else from reading these messages.

So even though the attacker has now come along and has inserted himself between Bob and Alice, Bob can be reasonably sure that Alice will receive the message intact even without data origin authentication and data authentication, because Alice can properly decrypt it. He is also not concerned that the attacker may have a copy of it because the encryption ensures a reasonable level of protection against the attacker ever understanding the messages contents.

The need to encrypt the information contained in many of today's internet transactions is very important. Billions of bytes of financial, medical, legal, business and diplomatic information pass

through the Internet every day. If there was no way to reasonably ensure confidentiality in the transfer of this information the Internet would not be the large, globally used, and open network it is today.

Basically, anybody would be able to see who had purchase what, who has what illness, who is being investigated for what, what the specs of a company's new products are, and what country is planning a trade agreement with what other country just by watching the data as it passes through the Internet. Clearly, many individual and corporate rights to privacy are violated without the use of some type of data encryption that protects the information as it travels across the Internet.

However, one has to ensure that the encryption method employed for this data transfer is sound and strong so that it prevents easy unauthorized decryption, but it must also not violate the many laws that are in place as to how strong an encryption process can be.

Encryption Techniques

Key Methods Substitution Ciphers

In substitution ciphers we are replacing the letters of the plaintext with other letters or symbols. Substitution ciphers can be broken into two classes; monoalphabetic – in which a single plaintext letter is replaced with a single substitution letter or symbol, and polyalphabetic – in which a single plaintext letter can be replaced with several different substitution letters or symbols. In this section we will look at the Multiplication Cipher which is a form of monoalphabetic cipher.

Multiplication Cipher

Today, messages are normally encoded and decoded using a computer, but as we all know computers understand numbers better then they understand letters or words. For that purpose we will number each letter. Normally, our number system starts at zero (0) so we will use that starting point for our conversion, as a result our conversion set is: {a=0, b=1, c=2,…, z=25}, as seen below.

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**

The Multiplication cipher is a modification of the Caesar cipher which uses addition of a constant number (the key) instead of multiplication to encrypt the message. Instead of adding a constant number we multiply the plaintext letter value by a common constant number. The results of this cipher and how it is broken is not much different than the Caesar cipher.

We are still limited by the same two things that create the weaknesses in the Caesar cipher; the key never changes and there are functionally only 25 of them that we can use. In fact because of the principles of multiplication and the use of modular arithmetic there are even less keys available for use.

To start we multiply each plain letter by our secret key ***b***. Since each plain letter turns into 0 for ***b=0*** and remains unchanged for ***b=1***, we start with **b=2**. We will multiply mod 26 as we are using the 26 letters of the English alphabet. We get the following encoding and decoding table.

PLAINTEXT LETTER:	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Secret key: ***b=2***	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
	0	2	4	6	8	10	12	14	16	18	20	22	24	0	2	4	6	8	10	12	14	16	18	20	22	24
cipher letter:	**a**	**c**	**e**	**g**	**i**	**k**	**m**	**o**	**q**	**s**	**u**	**w**	**y**	**a**	**c**	**e**	**g**	**i**	**k**	**m**	**o**	**q**	**s**	**u**	**w**	**y**

Notice, that only every other cipher letter appears, and that it appears twice. This is not a useful encryption system since it may yield ambiguous messages. For example, let's encode and decode NAT and ANT.

PLAINTEXT LETTER	N	A	T		A	N	T
Secret key ***b=2***	13	0	19		0	13	19
	↓	↓	↓		↓	↓	↓
	0	0	12		0	0	12
cipher letter	**a**	**a**	**m**		**a**	**a**	**m**

You can see the dilemma of this message. Decoding "**aam**" can either yield NAT or ANT as the plain text. What would you do? Of course, you don't want to receive any more ambiguous messages. Let's simply test all possible keys of the multiplication ciphers mod 26 as seen in the table depicted in Figure 4.

We know already that the key ***b=2*** (as can be seen in the 3rd row) does not produce a unique encryption. So which ones do? Each row that contains each integer from 0 to 25 exactly once and therefore yields a unique cipher letter will serve. Simply by looking at the table, we find that the following keys (whose rows are bold) produce a unique encryption and therefore call them the **good keys**:

$$\boldsymbol{b} = 1,3,5,7,9,11,15,17,19,21,23,25$$

Why those and what do they have in common? They seem to not follow any apparent pattern. Are they the odd numbers between 1 and 25? No, 13 is not a good key. Are they possibly the primes between 1 and 25? No, since 9, 15, 21 and 25 are not prime numbers and the prime 13 is missing.

So, let's understand why the **bad keys** such as:

$$\boldsymbol{b} = 2,4,6,8,10,12,13,14,16,18,20,22,24$$

does not produce a unique encryption.

Encoding Key *b*	PLAIN TEXT																									
	A	B	C	D	E	F	G	H	I	J	K	L	M	**N**	O	P	Q	R	S	T	U	V	W	X	Y	Z
0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**	0	0	0	0	0	0	0	0	0	0	0	0
1	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**
2	0	2	4	6	8	10	12	14	16	18	20	22	24	**0**	2	4	6	8	10	12	14	16	18	20	22	24
3	**0**	**3**	**6**	**9**	**12**	**15**	**18**	**21**	**24**	**1**	**4**	**7**	**10**	**13**	**16**	**19**	**22**	**25**	**2**	**5**	**8**	**11**	**14**	**17**	**20**	**23**
4	0	4	8	12	16	20	24	2	6	10	14	18	22	**0**	4	8	12	16	20	24	2	6	10	14	18	22
5	**0**	**5**	**10**	**15**	**20**	**25**	**4**	**9**	**14**	**19**	**24**	**3**	**8**	**13**	**18**	**23**	**2**	**7**	**12**	**17**	**22**	**1**	**6**	**11**	**16**	**21**
6	0	6	12	18	24	4	10	16	22	2	8	14	20	**0**	6	12	18	24	4	10	16	22	2	8	14	20
7	**0**	**7**	**14**	**21**	**2**	**9**	**16**	**23**	**4**	**11**	**18**	**25**	**6**	**13**	**20**	**1**	**8**	**15**	**22**	**3**	**10**	**17**	**24**	**5**	**12**	**19**
8	0	8	16	24	6	14	22	4	12	20	2	10	18	**0**	8	16	24	6	14	22	4	12	20	2	10	18
9	**0**	**9**	**18**	**1**	**10**	**19**	**2**	**11**	**20**	**3**	**12**	**21**	**4**	**13**	**22**	**5**	**14**	**23**	**6**	**15**	**24**	**7**	**16**	**25**	**8**	**17**
10	0	10	20	4	14	24	8	18	2	12	22	6	16	**0**	10	20	4	14	24	8	18	2	12	22	6	16
11	**0**	**11**	**22**	**7**	**18**	**3**	**14**	**25**	**10**	**21**	**6**	**17**	**2**	**13**	**24**	**9**	**20**	**5**	**16**	**1**	**12**	**23**	**8**	**19**	**4**	**15**
12	0	12	24	10	22	8	20	6	18	4	16	2	14	**0**	12	24	10	22	8	20	6	18	4	16	2	14
13	**0**	**13**	**0**	**13**	**0**	**13**	**0**	**13**	**0**	**13**	**0**	**13**	**0**	**13**	**0**	**13**	**0**	**13**	**0**	**13**	**0**	**13**	**0**	**13**	**0**	**13**
14	0	14	2	16	4	18	6	20	8	22	10	24	12	**0**	14	2	16	4	18	6	20	8	22	10	24	12
15	**0**	**15**	**4**	**19**	**8**	**23**	**12**	**1**	**16**	**5**	**20**	**9**	**24**	**13**	**2**	**17**	**6**	**21**	**10**	**25**	**14**	**3**	**18**	**7**	**22**	**11**
16	0	16	6	22	12	2	18	8	24	14	4	20	10	**0**	16	6	22	12	2	18	8	24	14	4	20	10
17	**0**	**17**	**8**	**25**	**16**	**7**	**24**	**15**	**6**	**23**	**14**	**5**	**22**	**13**	**4**	**21**	**12**	**3**	**20**	**11**	**2**	**19**	**10**	**1**	**18**	**9**
18	0	18	10	2	20	12	4	22	14	6	24	16	8	**0**	18	10	2	20	12	4	22	14	6	24	16	8
19	**0**	**19**	**12**	**5**	**24**	**17**	**10**	**3**	**22**	**15**	**8**	**1**	**20**	**13**	**6**	**25**	**18**	**11**	**4**	**23**	**16**	**9**	**2**	**21**	**14**	**7**
20	0	20	14	8	2	22	16	10	4	24	18	12	6	**0**	20	14	8	2	22	16	10	4	24	18	12	6
21	**0**	**21**	**16**	**11**	**6**	**1**	**22**	**17**	**12**	**7**	**2**	**23**	**18**	**13**	**8**	**3**	**24**	**19**	**14**	**9**	**4**	**25**	**20**	**15**	**10**	**5**
22	0	22	18	14	10	6	2	24	20	16	12	8	4	**0**	22	18	14	10	6	2	24	20	16	12	8	4
23	**0**	**23**	**20**	**17**	**14**	**11**	**8**	**5**	**2**	**25**	**22**	**19**	**16**	**13**	**10**	**7**	**4**	**1**	**24**	**21**	**18**	**15**	**12**	**9**	**6**	**3**
24	0	24	22	20	18	16	14	12	10	8	6	4	2	**0**	24	22	20	18	16	14	12	10	8	6	4	2
25	**0**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**

Figure 5: Mod 26 - Multiplication Cypher Table

We have to understand why multiplying by a bad key (***b***) mod 26 yields some integers more than once and others not at all. An extreme example would be when ***b=0***: all plain letters are translated into 0's which are all **a**'s so that no decryption is possible.

If ***b=1*** is used as a key, each cipher letter equals its plain letter which shows that it does produce a unique encryption. However, it yields the original text. This is not very useful.

The bad key ***b=2*** yields an ambiguous message as we saw in the introductory example: each (***b***) turns into 0 (=**a**) since 2*0 = 0 mod 26 just as each "N" turns into 0 since 2*13 = 26 = 0 mod 26. Also, each "B" and each "M" turn into 2 (=**c**) since 2*1 = 2 mod 26 and 2*14 = 28 = 2 mod 26. When you study the ***b=2*** row precisely, you will see that the original 26 plain letters are converted into 13 "even" cipher letters (the even cipher letters are those whose numerical equivalent is an even number.) each occurring exactly twice. This is the reason why ***b=2*** yields an ambiguous decryption.

If ***b=4, 6, 8,…, 24***, we encounter the same dilemma as for ***b=2.*** We can see in the table that an "A" will always translate into 0 (=**a**) since the product of any such key ***b*** with 0 (=**a**) yields 0. "N" (=13) translates into **a** for any even key (***b***) as well because

even keys **N**

```
4  * 13  = 2*(2*13) = 2*0 = 0 mod 26,
6  * 13  = 3*(2*13) = 3*0 = 0 mod 26,
8  * 13  = 4*(2*13) = 4*0 = 0 mod 26, etc.
```

Notice in all three equations that **because *b=2*** turns the 13 (=N) into 0 in 2*13 = 0, all the multiples of ***b=2*** translate the "N" into 0 (=**a**). That means:

Because *b=2* is a bad key all the multiples of (*b*) must be bad keys also.

Consider an alphabet length of *M=35*: the bad key ***b=5*** will translate the "H" (=7) into **a** (=0), because 5*7 = 35 = 0 mod 35. Therefore, all the keys that are multiples of 5 such as ***b=10,15,20,…,30*** will also translate the "H" into 0 (=**a**). Similarly, the multiples of ***b=7*** will translate an "F" (=5) into 0 (=**a**) because 7 does.

b=13 yields an ambiguous message since each even plaintext letter is translated into **a** (=0):

```
b=13    even letters
  13 *  0            =  0 mod 26,
  13 *  2            =  0 mod 26,
  13 *  4            =  (13*2) * 2 = 0 * 2 = 0 mod 26,
  13 *  6            =  (13*2) * 3 = 0 * 3 = 0 mod 26, etc.
```

Each odd plain letter translates into 13 (=**n**):

```
b=13    odd letters
  13 *  1          = 13 mod 26,
  13 *  3          = 13*2 + 13*1 = 0 + 13 = 13 mod 26,
  13 *  5          = 13*4 + 13*1 = 0 + 13 = 13 mod 26,
  13 *  7          = 13*6 + 13*1 = 0 + 13 = 13 mod 26, etc.
```

Moreover, since ***b=13*** is a bad key its multiples (i.e. 26, 39, 52…) must also be bad keys. However, we don't need to consider keys that are greater than 26 since each of them has an equivalent key less than 26 that yields the same encryption: the even multiples of 13 (i.e. 26, 52, 78, ...) have its equivalent key in ***b=0***, a very bad key, since 26 = 52 = 78 = 0 mod 26. The odd multiples of 13 (i.e. 39, 65, 91, …) have its equivalent key in ***b=13***, another bad key, since 39 = 65 = 91 = 13 mod 26.

Let's summarize what we know about which keys now yield a unique encryption?

A key ***b*** does not produce a unique encryption,

If ***b*** divides 26 evenly

Or

If ***b*** is a multiple of such divisors

We can combine these two criteria into one easy criterion.

Criteria for Good Keys

A key ***b*** produces a unique encryption,
If the greatest common divisor of 26 and ***b*** equals 1,
Which we write as: gcd (26, ***b***) = 1

Using what we know, 26 has a greatest common divisor equal to 1 with each of these good keys ***b = 1,3,5,7,9,11,15,17,19,21,23,25***. As a result, except for 2 and 13, all prime numbers less than 26 are among the keys.

However, there are some additional integers that are not prime numbers (i.e. 9, 15, 21 and 25). These numbers are those that don't have a common divisor with 26. Thus, being prime is not quite the reason for a good key, but almost. As a result we refer to the set of good keys as "*Relative Prime*" numbers.

Definition of numbers that are relative prime

Two integers are called *relative prime* if their greatest common divisor equals 1.

Examples: 4 and 5 are relatively prime because gcd (4, 5) = 1. So are 2 and 3, 2 and 5, 3 and 10, 26 and 27, 16 and 45.

Counter examples: 45 and 18 are **not** relative prime since gcd (45, 18) = 9 and not 1.
343 and 14 are **not** relative prime since gcd (343, 14) = 7.

From now on we will use a handy notation for the set of possible and good keys:

1) All the **possible** keys for an alphabet length of 26 are clearly all the numbers between 1 and 26, denoted as Z_{26}. Where: Z_{26} = {0, 1, 2, 3, …, 24, 25}.
   - Generally: An alphabet of length *M* has the keys: Z_M = {0, 1, 2, 3, …, *M*-2, *M*-1}

2) Now, the **good** keys are the ones that are relative prime to 26 as listed above and are denoted as Z_{26}*****. Where: Z_{26}***** = {1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25}.
   - Generally: The good keys are those ***b***'s that are relative prime to *M* and are denoted as Z_M*****.

We are now able to summarize how to encrypt a message using the multiplication cipher:

To encrypt a plain letter P to the cipher letter C using the Multiplication Cipher, we use the encryption function:

$$f: P \rightarrow C = (b * P) \bmod 26$$

If (*b*) is a "good" key, that is if (*b*) is relatively prime to 26, then *f* produces a one-to-one relationship between plain and cipher letters, which therefore permits a unique encryption.

Among the 12 good keys we pick ***b=5*** to encode our secret message as follows:

PLAINTEXT	S	M	O	K	E	O	N	T	H	E	W	A	T	E	R
	18	12	14	10	4	14	13	19	7	4	22	0	19	4	17
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
	12	8	18	24	20	18	13	17	9	20	6	0	17	20	7
ciphertext	**m**	**i**	**s**	**y**	**u**	**s**	**n**	**r**	**j**	**u**	**g**	**a**	**r**	**u**	**h**

So our encoded message reads: **misyusnrjugaruh**

Breaking the Cipher

Now that the secret message is encoded in a unique manner how can it be decoded? First of all, you need to know which one of the 12 good keys was used. In some secret manner, the sender and the recipient had to agree on the encoding key (***b***). Let's say ***b=5*** was chosen. Now, how do you decrypt the above message? To do so, we have to look at the encryption equation ***C = b * P mod 26*** and solve it for the desired plaintext letter P.

In order to solve an equation like ***23 = 5 * P*** for ***P*** using the rational numbers, we would divide by 5 or multiply by 1/5 to obtain the real solution ***P = 23/5***. However, when using mod arithmetic and solving ***23 = 5 * P mod 26***, we don't deal with fractions but only integers. Therefore the division is performed slightly different: instead of dividing by 5 or multiplying by 1/5, we first write 5^{-1} (instead of 1/5) where 5^{-1} now equals an integer and multiply both sides by that integer 5^{-1}. We denote 5^{-1} as "the *inverse* of 5". Just as 5 * 1/5 yields 1, 5 * 5^{-1} equals 1 mod 26.

Definition of an inverse number:

> A number b^{-1} that yields 1 when multiplied by ***b*** is called the *inverse of* ***a***.
>
> Mathematically: $b^{-1} * b = b * b^{-1} = 1$.

Example1: When using fractions,

> $5^{-1} = 1/5$ is the inverse number to 5,
> $3^{-1} = 1/3$ is the inverse number to 3,
> $3/2$ is the inverse number to 2/3.

Example2: Now, let's look at examples for mod arithmetic:

> The inverse of ***b=3*** is b^{-1} = 2 mod 5 because $b * b^{-1} = 3 * 2 = 6 = 1 \bmod 5$.
> We find that $b^{-1} = 2$ by simply testing the integers in Z_5^* = {1, 2, 3, 4}.

```
(3 * 1) mod 5  =   3 mod 5 =   3
(3 * 2) mod 5  =   6 mod 5 =   1
(3 * 3) mod 5  =   9 mod 5 =   4
(3 * 4) mod 5  =  12 mod 5 =   2
```

b=4 is inverse to itself modulo 5 since $b * b^{-1} = 4 * 4 = 16 = 1 \bmod 5$.

```
(4 * 1) mod 5  =   4 mod 5 =   4
(4 * 2) mod 5  =   8 mod 5 =   3
(4 * 3) mod 5  =  12 mod 5 =   2
(4 * 4) mod 5  =  16 mod 5 =   1
```

Example3:

Doing arithmetic mod 7, the inverse of ***b=3*** is $b^{-1} = 5$ because

$b * b^{-1} = 3 * 5 = 15 = 1 \bmod 7$.

We found the inverse of ***b=3*** by again testing the integers in $Z_7^* = \{1, 2, 3, 4, 5, 6\}$

```
(3 * 1) mod 7  =   3 mod 7 =   3
(3 * 2) mod 7  =   6 mod 7 =   6
(3 * 3) mod 7  =   9 mod 7 =   2
(3 * 4) mod 7  =  12 mod 7 =   5
(3 * 5) mod 7  =  15 mod 7 =   1
(3 * 6) mod 7  =  18 mod 7 =   4
```

The inverse of ***b=4*** is 2 since $b * b^{-1} = 4 * 2 = 8 = 1 \bmod 7$.

```
(4 * 1) mod 7  =   4 mod 7 =   4
(4 * 2) mod 7  =   8 mod 7 =   1
(4 * 3) mod 7  =  12 mod 7 =   5
(4 * 4) mod 7  =  16 mod 7 =   2
(4 * 5) mod 7  =  20 mod 7 =   6
(4 * 6) mod 7  =  24 mod 7 =   3
```

And finally, ***b=6*** is inverse to itself mod 7 since $b * b^{-1} = 6 * 6 = 36 = 1 \bmod 7$.

```
(6 * 1) mod 7  =   6 mod 7 =   6
(6 * 2) mod 7  =  12 mod 7 =   5
(6 * 3) mod 7  =  18 mod 7 =   4
(6 * 4) mod 7  =  24 mod 7 =   3
(6 * 5) mod 7  =  30 mod 7 =   2
(6 * 6) mod 7  =  36 mod 7 =   1
```

Now, back to the secret message. It is encoded with mod 26. Since our mod 26 number set ranges from 0 to 25, we now have to find an integer b^{-1} among those numbers that yields 1 mod 26 when multiplied by 5 as in $\mathbf{5 * b^{-1} = 1 \bmod 26}$. We know we can pretty much rule out any number in out number set that is going to provide us a with a resultant that is less than 26, in this case the numbers 1, 2, 3, 4 and 5. So let's perform the calculations:

```
(5 * 6) mod 26 =  30 mod 26 =   4
(5 * 7) mod 26 =  35 mod 26 =   9
(5 * 8) mod 26 =  40 mod 26 =  14
(5 * 9) mod 26 =  45 mod 26 =  19
```

```
(5 * 10) mod 26 =  50 mod 26 =  24
(5 * 11) mod 26 =  55 mod 26 =   3
(5 * 12) mod 26 =  60 mod 26 =   8
(5 * 13) mod 26 =  65 mod 26 =  13
(5 * 14) mod 26 =  70 mod 26 =  18
(5 * 15) mod 26 =  75 mod 26 =  23
(5 * 16) mod 26 =  80 mod 26 =   2
(5 * 17) mod 26 =  85 mod 26 =   7
(5 * 18) mod 26 =  90 mod 26 =  12
(5 * 19) mod 26 =  95 mod 26 =  17
(5 * 20) mod 26 = 100 mod 26 =  22
(5 * 21) mod 26 = 105 mod 26 =   1
(5 * 22) mod 26 = 110 mod 26 =   6
(5 * 23) mod 26 = 115 mod 26 =  11
(5 * 24) mod 26 = 120 mod 26 =  16
(5 * 25) mod 26 = 125 mod 26 =  21
```

Now we know that the answer is 21 because 105 divided by 26 leaves a remainder of 1. Excellent, now we have the answer we need to solve our encryption function ***C= b * P mod 26*** for the plaintext letter ***P*** in order to decode the secret message:

Multiplying both sides of our encryption equation yields the following:

$$b^{-1} * C = b^{-1}*(b*P) \quad [1]$$
$$= (b^{-1}*b)*P \quad [2]$$
$$= 1*P \quad [3]$$
$$= P \bmod 26 \quad [4]$$

Where: solving this equation requires the following 4 group properties:

[1] The existence of an inverse
[2] The associative property
[3] The inverse property
[4] The unit element property

We are only guaranteed to solve this equation for numbers that form a group of good keys with respect to multiplication mod 26. This would be the group of relative prime numbers that we saw earlier: $Z_{26}^* = \{1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25\}$. Therefore, we can always find b^{-1} for a given good key (***b***).

Our decoding function $P = b^{-1}*C \bmod 26$ tells us to simply multiply each cipher letter by the inverse of the encoding key ***b=5***, because of the decoding key $b^{-1} = 21 \bmod 26$ and we can eventually decode:

ciphertext	**m**	**i**	**s**	**y**	**u**	**s**	**n**	**r**	**j**	**u**	**g**	**a**	**r**	**u**	**h**
	12	8	18	24	20	18	13	17	9	20	6	0	17	20	7
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
	18	12	14	10	4	14	13	19	7	4	22	0	19	4	17
PLAINTEXT	S	M	O	K	E	O	N	T	H	E	W	A	T	E	R

Let's verify the decoded message for the first five letters by multiplying our ciphertext letter with $b^{-1} = 21$:

$(b^{-1} * C) \bmod 26 = (x) \bmod 26 = P$

```
(21 * 12) mod 26 = 252 mod 26 =  18 = S
(21 *  8) mod 26 = 168 mod 26 =  12 = M
(21 * 18) mod 26 = 378 mod 26 =  14 = O
(21 * 24) mod 26 = 504 mod 26 =  10 = K
(21 * 20) mod 26 = 420 mod 26 =   4 = E
```

Symmetric Key Ciphers

Symmetric key cryptography (aka. secret key cryptography) relies on using the same key for both the encryption and decryption and decryption process and is considered a shared-key system. This form of encryption can use either:

- stream ciphers - ciphers that encrypt each plain text character individually with the symmetric key or a corresponding symmetric keystream
- block ciphers - ciphers that encrypt blocks (fixed lengths of bits) of plaintext with the symmetric key.

Of the two variations the stream cipher is generally faster at both the encryption and decryption processes.

Symmetric key encryption has been used in many systems since the 1970s when DES (Data Encryption Standard) a block cipher was selected as the encryption standard by the NSA for the United States government. Since then, several other symmetric key algorithms have been introduced including:

- 3-DES - a modified and considered much stronger version of DES
- RC4 - a widely used stream cipher implemented in SSL and WEP
- Twofish - a block cipher that has been released to the public domain and included in the OpenPHP standard
- AES (Advanced Encryption Standard) - the current block cipher used as the encryption standard for the United States government

One of the main issues associated with symmetric key ciphers is that they are considered inherently weak due to the single key and key-length limitations implemented in the current algorithms. However, the algorithms used to create the cipher text varies greatly and some of these ciphers have only been theoretically broken, i.e. calculation on what it would take to break the cipher-text have been performed but no real world test has been implemented to actually break the cipher.

Another issue with these ciphers is that they cannot be used for non-repudiation. Non-repudiation is an assurance that the sender is who they say they were and that the receiver is who they say they are. In the case of a symmetric key system, because the same key both encrypts and decrypts the message and because all key holders have the same key, one of the key holders can actually fake a message and say they received it from the other key holder.

Without some other means of verifying the origin or receipt of a message, the key holders have no true way of determining who sent or who received a message other than blind trust.

Public Key Ciphers

Public Key cryptography which relies on two keys - one public and one private, is often referred to as two-key or asymmetric cryptography. This form of cryptography had to first overcome one of the long standing issues with cryptography had to be initially overcome; the issue of key transfer.

In order to communicate secretly the sender had to pre-deliver the secret key to the recipient, otherwise the recipient of the enciphered messages would not be able to convert them back to plain text. How can the key be sent from one party to another securely enough to remain a secret? If you sent it over an insecure channel (i.e. telephone, internet, etc.) what's to stop someone from intercepting it and using it to decode all subsequent messages?

Diffie-Hellman Key Exchange

The Diffie-Hellman Key Exchange provided a process that overcame the key transfer dilemma. Whitefield Diffie and Martin Hellman's idea of Public-Key Cryptography was described in their paper "New Directions in Cryptography", in which they showed how two individuals, located in different places, can publicly create a common encryption key without the fear that a third party could obtain the key, even if they observed the key generation process. It enabled the two individuals to establish secure communication *without* having to pre-deliver the secret key.

To understand the Diffie-Hellman Key Exchange let's return to an example between Alice and Bob easily. Here it is in 4 simple steps:

Step 1: Alice and Bob publicly pick two (2) integers.
- a. Alice picks a prime number which we will refer to as p and sends it to Bob
- b. Bob picks an integer between 1 and p that we will refer to as s and sends it back to Alice.

Step 2: Alice and Bob pick random numbers that are both less than p
- a. Alice picks a random number we will call a
- b. Bob picks a random number we will call b

Step 3: From these numbers Alice and Bob calculate two new numbers
- a. Alice calculates $A = s^a \text{ MOD } p$ and sends A to Bob
- b. Bob calculates $B = s^b \text{ MOD } p$ and sends B to Alice

Step 4: Alice and Bob then calculate their keys based on the information they have sent each other
- a. Alice calculates her key as $K = B^a \text{ MOD } p$
- b. Bob calculates his key as $K = A^b \text{ MOD } p$

Now let's look at these 4 steps using actual numbers:

Step 1: Alice picks $p = 17$ and Bob picks $s = 8$ and they send these numbers to each other

Step 2: Alice picks $a = 3$ and Bob picks $b = 6$ as their random numbers

Step 3: Alice calculates the number she is going to send to Bob as:

$$A = s^a \text{ MOD } p = 8^3 \text{ MOD } 17 = 2$$

and Bob calculates the number he is going to send to Alice as:

$$B = s^b \text{ MOD } p = 8^6 \text{ MOD } 17 = 4$$

Step 4: Finally, Alice calculates her key as:

$$K = B^a \text{ MOD } p = 4^3 \text{ MOD } 17 = 13$$

and Bob calculates his key as:

$$K = A^b \text{ MOD } p = 2^6 \text{ MOD } 17 = 13$$

Now we have to answer two very important questions:

a) Why do Alice and Bob always end up with the same key K?
b) Why can't an eavesdropper compute the key K?

The answer to the first question is simple mathematics, Alice and Bob both calculate the key K in the final step. But if we write the math using the expanded version of the formulas it looks as follows:

Alice: $K = B^a = s^{ba} \text{ MOD } p$

Bob: $K = A^b = s^{ab} \text{ MOD } p$

Since $s^{ba} \text{ MOD } p = s^{ab} \text{ MOD } p$ both Alice and Bob calculate the same key K.

The answer to the second question requires us to go beyond this simple example that we have used here. Even if an eavesdropper did manage to intercept the values of A, B, p and s, he still has to manually calculate the key K himself. If the numbers were small as in the above example, the eavesdropper probably could successfully calculate the key. However, by making the numbers incredibly large, i.e. hundreds or even thousands of digits in length, for a, b, p, and s it becomes mathematically impossible for the eavesdropper to reverse the calculations in a reasonable or timely fashion.

The reason? Well, although it is quite simple to compute $A = s^a \text{ MOD } p$, however, solving this equation for a is impossible (remember the eavesdropper only has A, B, p and s so he must therefore calculate both a and b). Functionally, while the "*discrete exponential function*" can be calculated, its inverse, the "*discrete logarithm function*" cannot be. This is simply because the

sheer size of the numbers involved in the calculation turns the "*discrete exponential function*" into a "One-Way" function.

Let's assume that the numbers selected for a and b are 100 digits in length. The only possible way to reverse the calculation would be to try all the possible combinations of 100-digit numbers for a and b until you calculate the same result as A and B using the known numbers for p and s. This is not a practical approach simply because the time involved would be tremendous.

Even with advances in computing power over time, assuming a doubling in processing capability every 12 months, the time it would take to try all possible combinations between 1 and of 100 digit numbers would likely take far longer than the useful lifespan of any information that was in the encrypted message. It would also be important to note that as computing power increases, Alice and Bob could also increase the length of the values used for a, b, p, and s, thereby staying ahead of any possible way to discover future keys.

Still there was another problem. While the issue of key exchange was resolved, there was still the issue of how to use it effectively. Diffie Hellman proposed a theoretical solution where each party shall possessed a **key pair**, a **public** and a **private key**. I.e. for Alice and Bob; Alice's public key is used by Bob to encrypt the message for Alice whereas her private key is used to decrypt Bob's encrypted message.

The problem of finding an appropriate one-way function which could do this was resolved by the scientists from MIT; Ronald Rivest, Adi Shamir and Leonard Adleman, who created the RSA Cipher.

RSA Cipher

The RSA Cipher is actually quite easy to understand and uses a three step process to execute as follows:

Step 1: **Preparation:** We begin by choosing two prime numbers, p and q, so that their product n is greater than the used alphabet length M.

Step 2: **Encryption:** We then choose a public encoding key e that has to be relative prime to $\varphi(n)$. Where $\varphi(n) = (p - 1) * (q - 1)$. We then encrypt each plaintext letter P with the formula $C = P^e \text{ MOD } n$ for our calculations

Step 3: **Decryption:** The private decoding key d is chosen as the inverse of $e \text{ MOD } \varphi(n)$, this is similar in function to when we found the inverse in the Multiplication Cipher. In other words $d^{-1} = e \text{ MOD } \varphi(n)$, which is the same as $e * d = 1 \text{ MOD } \varphi(n)$. We then decrypt using the formula $P = C^d \text{ MOD } n$.

Now let's look at these 3 steps using actual numbers, the word "LAUNCH" and we will also assume that the alphabet has been converted, like our previous example, to a number series between 0 and 25 for A through Z respectively:

Step 1: For our example we will use the following prime numbers:

$p = 11$ and $q = 23$
Therefore $n = p * q = 253$

Step 2: Now we will choose our public encoding key e - remember it has to be relative prime to $\varphi(n)$. Possible values for e include:

{1, 3, 7, 9, 13, 17, 19, 21, 23, 27, 29, 31, 37, 39, 41, 43, 47, 49, 51, 53, 57, 59, 61, 63, 67, 69, 71, 73, 79, 81, 83, 87, 89, 91, 93, 97, 101, 103, 107, 109, 111, 113, 117, 119, 123, 127, 129, 131, 133, 137, 139, 141, 143, 147, 149, 151, 153, 157, 159, 161, 163, 167, 169, 171, 173, 177, 179, 181, 183, 189, 191, 193, 197, 199, 201, 203, 207, 211, 213, 217 and 219}

For our purposes we will choose $e = 31$.

Now we will encrypt using the formula $C = P^e \text{ MOD } n$ as follows:

L = 11	so	11^{31}	MOD 253 = 88
A = 0	so	0^{31}	MOD 253 = 0
U = 20	so	20^{31}	MOD 253 = 97
N = 13	so	13^{31}	MOD 253 = 13
C = 2	so	2^{31}	MOD 253 = 167
H = 7	so	7^{31}	MOD 253 = 84

Step 3: Now for the decoding part. Similar to how the Multiplication cipher worked, we decode by finding an inverse number to the encoding key. But there is a twist: since we are using two numbers to make up our key (11 and 23) the decoding key d is the inverse of $e \text{ MOD } \varphi(n)$. Remember above, we stated that $\varphi(n) = (p - 1) * (q - 1)$, so calculated out $\varphi(n) = 220$.

So, $d^{-1} = e \text{ MOD } 220$ or written another way $e * d = 1 \text{ MOD } 220$. Now let's create a small table to determine the values of d, using the relative prime numbers that we listed earlier.

e	d	Formula	Result	e	d	Formula	Result
31	1	31 * 1 MOD 220	31	31	113	31 * 113 MOD 220	203
31	3	31 * 3 MOD 220	93	31	117	31 * 117 MOD 220	107
31	7	31 * 7 MOD 220	217	31	119	31 * 119 MOD 220	169
31	9	31 * 9 MOD 220	59	31	123	31 * 123 MOD 220	73
31	13	31 * 13 MOD 220	183	31	127	31 * 127 MOD 220	197
31	17	31 * 17 MOD 220	87	31	129	31 * 129 MOD 220	39
31	19	31 * 19 MOD 220	149	31	131	31 * 131 MOD 220	101
31	21	31 * 21 MOD 220	211	31	133	31 * 133 MOD 220	163
31	23	31 * 23 MOD 220	53	31	137	31 * 137 MOD 220	67
31	27	31 * 27 MOD 220	177	31	139	31 * 139 MOD 220	129
31	29	31 * 29 MOD 220	19	31	141	31 * 141 MOD 220	191
31	31	31 * 31 MOD 220	81	31	143	31 * 143 MOD 220	33
31	37	31 * 37 MOD 220	47	31	147	31 * 147 MOD 220	157
31	39	31 * 39 MOD 220	109	31	149	31 * 149 MOD 220	219
31	41	31 * 41 MOD 220	171	31	151	31 * 151 MOD 220	61

31	43	31 * 43 MOD 220	13		31	153	31 * 153 MOD 220	123
31	47	31 * 47 MOD 220	137		31	157	31 * 157 MOD 220	27
31	49	31 * 49 MOD 220	199		31	159	31 * 159 MOD 220	89
31	51	31 * 51 MOD 220	41		31	161	31 * 161 MOD 220	151
31	53	31 * 53 MOD 220	103		31	163	31 * 163 MOD 220	213
31	57	31 * 57 MOD 220	7		31	167	31 * 167 MOD 220	117
31	59	31 * 59 MOD 220	69		31	169	31 * 169 MOD 220	179
31	61	31 * 61 MOD 220	131		31	171	31 * 171 MOD 220	21
31	63	31 * 63 MOD 220	193		31	173	31 * 173 MOD 220	83
31	67	31 * 67 MOD 220	97		31	177	31 * 177 MOD 220	207
31	69	31 * 69 MOD 220	159		31	179	31 * 179 MOD 220	49
31	***71***	***31 * 71 MOD 220***	***1***		31	181	31 * 181 MOD 220	111
31	73	31 * 73 MOD 220	63		31	183	31 * 183 MOD 220	173
31	79	31 * 79 MOD 220	29		31	189	31 * 189 MOD 220	139
31	81	31 * 81 MOD 220	91		31	191	31 * 191 MOD 220	201
31	83	31 * 83 MOD 220	153		31	193	31 * 193 MOD 220	43
31	87	31 * 87 MOD 220	57		31	197	31 * 197 MOD 220	167
31	89	31 * 89 MOD 220	119		31	199	31 * 199 MOD 220	9
31	91	31 * 91 MOD 220	181		31	201	31 * 201 MOD 220	71
31	93	31 * 93 MOD 220	23		31	203	31 * 203 MOD 220	133
31	97	31 * 97 MOD 220	147		31	207	31 * 207 MOD 220	37
31	101	31 * 101 MOD 220	51		31	211	31 * 211 MOD 220	161
31	103	31 * 103 MOD 220	113		31	213	31 * 213 MOD 220	3
31	107	31 * 107 MOD 220	17		31	217	31 * 217 MOD 220	127
31	109	31 * 109 MOD 220	79		31	219	31 * 219 MOD 220	189
31	111	31 * 111 MOD 220	141					

From the results of our table it appears that 71 is the number we are looking for as our inverse. Now we can apply our decryption formula of $P = C^d \text{ MOD } n$ to this process and decrypt our encrypted message, which is currently 88-0-97-13-167-84.

88 becomes 88^{71} MOD 253 = 1 = L
0 becomes 0^{71} MOD 253 = 0 = A
97 becomes 97^{71} MOD 253 = 8 = U
13 becomes 13^{71} MOD 253 = 18 = N
167 becomes 167^{71} MOD 253 = 19 = C
84 becomes 84^{71} MOD 253 = 18 = H

For this example we chose the word LAUNCH for a specific reason; two of the letters have a problem, similar to that in the Multiplication Cipher. The letters "A" and "N" are represented by 0 and 13 respectively. The number 0 will always equate to 0 in its calculations, as a result the letter "A" will always be represented by a 0 in both plain text and cipher text.

The letter "N" likewise will always be represented by the number 13. This is because in the English language the alphabet has 26 characters and the number 13 is the GCD of 26. So, like the letter "A" the letter "N" will be represented by the number 0 and 13, respectively, in both plain and encrypted (cipher) text.

To compensate for this situation; where the character set in use contains an even number of characters, the character set can be padded with a filler or Null character. This results in an odd number character set and eliminates the situations where one character, like the letter "N", is the same in plain and cipher text form.

To eliminate the situation created by the letter “A” always being zero, there are two options:

- Shift the number set to start at a different number, for example instead of 0 through 25, the number set could start at 237 for “A” and end at 262 for “Z”.
- The other option is to randomly generate a number for each character. In this way the numbers do not follow a standard counting pattern, but each character would have a unique identifier. To do this, the encryption and decryption process would have to use the same number set to properly identify the characters after decryption.

References

Texts

Brief Historical Use

Simon Singh. *The Code Book: The Science of Secrecy from Ancient Egypt to Quantum Cryptography*, Anchor Books, 1999. ISBN: 0-385-495323 [Pages 1-25]

Main Goals of Cryptography

H.X.Mel and Doris Baker. *Cryptography Decrypted*, Addison Wesley, 2001. ISBN: 0-201-61647-5 [Pages 53-63]

Public Key Ciphers

H.X.Mel and Doris Baker. *Cryptography Decrypted*, Addison Wesley, 2001. ISBN: 0-201-61647-5 [Pages 75-87]

Neils Ferguson and Bruce Schneier. *Practical Cryptography*, Wiley Technology Publishing, 2003. ISBN: 0-471-22357-3 [Pages 26-30]

Diffie-Hellman Key Exchange

H.X.Mel and Doris Baker. *Cryptography Decrypted*, Addison Wesley, 2001. ISBN: 0-201-61647-5 [Pages 300-304]

RSA Cipher

H.X.Mel and Doris Baker. *Cryptography Decrypted*, Addison Wesley, 2001. ISBN: 0-201-61647-5 [Pages 267-288]

White Papers and Web References

NKU Faculty: Caesar Ciphers, (Chris Christensen, August 22, 2006), http://www.nku.edu/~christensen/section%202%20caesar%20ciphers.pdf

SANS Institute: History of Cryptography, (author not published, 2001), http://www.sans.org/reading-room/whitepapers/vpns/history-encryption-730?show=history-encryption-730&cat=vpns

Crypto Challenge: A Brief History of Cryptography page (author unknown, publish date unknown), https://www.cryptochallenge.com/home/history

All Net: A Short History of Cryptography page, (Fred Cohen, 1995), http://all.net/edu/curr/ip/Chap2-1.html

Cryptozine: A Brief History of Cryptography page, (author unknown, May 16, 2008), http://cryptozine.blogspot.ca/2008/05/brief-history-of-cryptography.html

Symantec: An Introduction to Cryptography, (John Callas, 2009), http://www.google.ca/url?sa=t&rct=j&q=&esrc=s&frm=1&source=web&cd=2&cad=rja&ved=0CEEQFjAB&url=http%3A%2F%2Fwww.symantec.com%2Fbusiness%2Fsupport%2Fresources

%2Fsites%2FBUSINESS%2Fcontent%2Flive%2FTECHNICAL_SOLUTION%2F149000%2FTECH149738%2Fen_US%2Fintrocrypto.pdf&ei=vfhEUo2oC4qgiAKKjIHgCw&usg=AFQjCNGLIDq5g4K71RcLMs-buUGlTdGyeg

PGP: An Introduction to Cryptography, (author unknown,1990), ftp://ftp.pgpi.org/pub/pgp/7.0/docs/english/IntroToCrypto.pdf

Northern Kentucky University: Multiplicative Ciphers, (Chris Christensen, Fall 2006), http://www.nku.edu/~christensen/section%206%20multiplicative%20ciphers.pdf

The Crypto Tutorial: Chapter 2: Multiplication Cipher, (N. Hahnfeld, 2001), http://www.ti89.com/cryptotut/text/Multi.doc
Itillious: Encryption Basics page, (author unknown, 2001), http://www.itillious.com/insight/articles/encryptionbasics.html

Yahoo Voices: Advantages and Disadvantages of Symmetric and Asymmetric Key Encryption Methods, (Dr. Chukwumah Ezeobika, July 2, 2010), http://voices.yahoo.com/comparing-symmetric-asymmetric-key-encryption-6329400.html

The Crypto Tutorial: Chapter 4: RSA Cipher, (N. Hahnfeld, 2001), http://www.ti89.com/cryptotut/text/RSA.doc

John Hopkins University: New Directions in Cryptography, (Whitfield Diffie and Martin E. Hellman, June 3, 1976), http://www.cs.jhu.edu/~rubin/courses/sp03/papers/diffie.hellman.pdf

Massachusetts Institute of Technology: New Directions in Cryptography: Twenty Some Years Later, (Shafi Goldwasser, 1997), http://groups.csail.mit.edu/cis/pubs/shafi/1997-focs.pdf

McGill University: Security Issues in the Diffie-Hellman Key Agreement Protocol, (Jean-Francios Raymand and Anton Stiglic, 2003), http://crypto.cs.mcgill.ca/~stiglic/Papers/dhfull.pdf

Chapter 3: Implementation

Implementing Security Measures

Defense-in-Depth Model

Defense-in-Depth Basics

Presented here is a 7-layer model which has been developed over time by this document's author. With more than 26-years of IT experience in all facets of the IT industry, this model has proven itself to be generic enough to assist in the defining of defense-in-depth strategy for most organizations regardless of their infrastructure and IT strategies, including the now popular "cloud-based" services that many organizations are moving towards including as part of their critical IT service solutions.

Again, any defense-in-depth model, including the one presented here, is really nothing more than a template or guideline to follow in which to implement security measures to protect your organization's systems, technology, people, buildings, applications and data. No matter how advanced the security measures are there are two things that one must always remember.

1. The implementation of security must not come at the expense of preventing the user from being able to perform their job effectively.
2. Without proper management and continuous monitoring, all security solutions will eventually fail.

There is really no way around these two important issues. All too often, security has been compromised or simply failed as a result of one or both of these issues being ignored. Why?

The answers are simple. First, if you overly complicate how the user must perform their job in order to get their work done, they will just find ways to circumvent the security measures to make their job easier and more efficient. To many passwords, security fobs, access restrictions, inactivity timeouts, etc., will simply force the user to find a way around all of the security in order to do their work.

Second, failure to properly manage and monitor the security systems, will eventually lead to a breach and most likely a breach that you will remain unaware of for some time. Like any system, regular updates and configuration tuning must take place to keep up with current security threats. The reports, alerts and logs (monitoring) from the security systems must be regularly reviewed and verified.

Just putting in security measures and walking away only lulls the organization into a false sense of security. Continuous diligence in monitoring is the only way you can be sure that the systems are functioning and reporting properly.

Now, let's discuss the 7-layer Defense-in-Depth model represented in Figure-2.

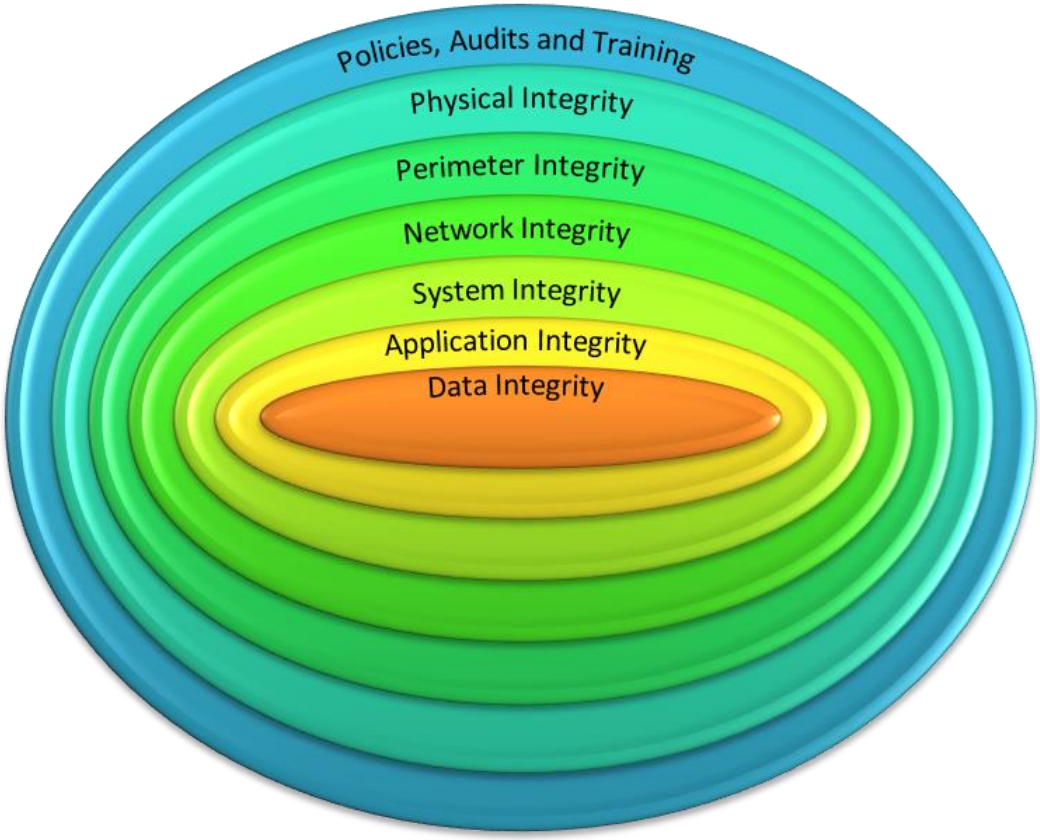


Figure 6: 7- Layer Defense-in-Depth

As you can see, the layers are overlapping one another towards the center of the model. At the center is Data Integrity. This is where the real wealth in any IT system lies. It is the customer's orders and order history. It is their credit card information, their billing information, their bank account and social security numbers. It contains the organization's next latest project information, new product designs and patents.

It is everything that can be used to steal your customer's money or identity, or your organization's integrity and competitive advantage. In short, for simple 1's and 0's it is the most important thing you must protect. Systems can be rebuilt, applications can be re-installed and yes destroyed data can be recovered from back-up. But stolen data…, data that could ruin your company's reputation, customer confidence, lead to lawsuits, etc., is never truly recoverable.

Policies, Audits and Training

In this model, we will start at the first layer; "Policies, Audits and Training". Security policies have been around for many years. In many cases, the larger the organization, the more complex and often more burdensome the policies became. However, over the past decade, policies and guidelines on how to properly create them have seen great improvements. This was mainly due to the many industry-based standards that have been introduced by both government and independent bodies.

The purpose of the policies to provide the reader, typically and organization's employee, with information about what is legitimate use of the organization's IT resources, versus what is considered inappropriate or misuse of these resources. Standards, such as the ISO-17799 Information Security Standard and the ISO-27000 family of standards have helped many organizations build better security practices and implement better policies.

Depending on the size of the organization, the IT policies can be as simple as a single-page Acceptable Use Policy, which defines nothing more than what is considered as "acceptable" use of the organization's IT resources to a full policy manual with glossary of terms. In many cases the user is expected to sign a document which states that they have read and understood the policy(ies) and the consequences of enforcement should the policy be breached. Also, in most cases, especially if there is a large policy manual, the users actually never read the policies.

That said, for many mid to large size organization's today, a full set of policies can be expected to encompass some, or all, or even expand on the following list of policies:

- IT Resource Acceptable Use Policy
- IT e-Mail Use Policy
- IT Internet Use Policy
- IT Password Policy
- IT Anti-virus and Anti-malware policy
- IT Resource Disposal Policy
- IT Encryption Policy
- IT Physical Environment Protection Policy
- IT Wireless Security Policy
- IT Logical Access Management Policy
- IT trusted and Untrusted Device Management Policy
- IT Remote Access Policy

Along with these policies, there may also be other documents written by the IT Department staff including a Standards of Practice and an IT Governance Manual. All of these documents are brought together to define the organization's security stance and how it will manage eventual breaches or inappropriate use of its IT resources.

Which leads us to the second part of the first level; Audits. Within any IT Department there should be many different types of audits being performed, both by the organization's IT staff and by 3rd party contractors who are knowledgeable in performing these security audits.

The audits themselves should really be designed around testing that the policies are being properly monitored and enforced as well as testing each of the controls or procedures in each of the Defense-in-Depth's layers. This includes verifying:

- systems patches are performed regularly,
- firewall or intrusion prevention system rules are properly implemented and optimized,
- access controls into restricted IT resource centers are properly implemented,

- user account password strengths are appropriate and that they are regularly forced to be changed
- system access rights and controls are properly implemented and reviewed regularly
- network traffic flows from secure networks and unsecure networks are properly filtered and managed, etc.

Basically, any service, application, or physical and electronic access to the IT Resources should be monitored and audited to ensure that there are no security issues, and auditing is a key part of ensuring the success of this monitoring. Reality check here, you can never patch or close all of the security holes in a complex IT solution, however with proper audits you can verify where the security weakness exist in your systems, set up more effective monitoring for these security weaknesses to alert you faster to potential breaches. Regular auditing also helps to identify and potentially close or repair new security weaknesses that come along.

Internal audits should be performed on both a regular and ad-hoc basis. The regular audits, typically performed once a year, should focus on the big picture or overall security. User rights, access controls, system logs, application logs, firewall logs, etc., should be reviewed and any questionable or unrecognized activity should be followed up on.

Configuration audits should also be a part of the regular audits, to ensure that system configurations are properly managed and backed up and to ensure that appropriate change management processes were followed. A proper configuration audit will quickly determine if anything has changed with respect to network systems such as if the router configurations match the backup configuration files or if the firewall rules have been adjusted without going through proper change management processes.

Ad-hoc audits, when performed, should only focus on a single auditing area. Occurring more frequently than regular audits, they pick on only one area of the regular, overall, security audit and are most often performed after a significant change has happened. These changes can be anything from a major application upgrade, a firewall replacement, or a substantial staff change.

Performing the ad-hoc audit, basically closes the security loop for the significant change and ensures that auditing procedures for the regular audit are updated properly. It also provides the Administrators with information regarding to potential security weaknesses that the change may have closed or introduced.

With all of this internal auditing, why perform 3rd party audit? Well, the answer here is quite simply, “Human Nature”. While internal staff do spend a great deal of time working on their system’s security, they also make assumptions about how secure their system is without thoroughly performing the tests.

They know how the systems are designed; most often because they designed it and they know how the systems are configured because they configured it. They have performed the security audit in the past and it passed, why should the next audit be any different? Again, the assumptions made by knowledgeable, competent and train staff are often any organizations greatest weakness.

The 3rd party auditors are basically a fresh set of eyes looking at the organization's security practices. They will test and retest the systems without making assumptions, based on "internal" knowledge of the systems. Often, the 3rd party auditors also have knowledge related to new attack vectors in their arsenal and will use them as required to test that the systems are properly secure.

All too often, old attacks come back because of weaknesses introduced by new patches and application upgrades, or the old attack only need a slight modification to work again. So, while the organization's Administrators assume that they have gotten everything, the 3rd party auditor will often surprise them. The 3rd party auditors can also provide professional external advice on current best practices and even assist in training internal staff on how to mitigate against know security weaknesses in their systems.

The training leads to the third item. It is not just the Administrators that need training, but all staff. While the training of general staff is different than that of the Administrators, they must still be put through a security training session.

This training session doesn't have to long, it just has to be effective, i.e. it has to teach the staff what to do in different security related scenarios. This can include providing them with training in the following situations:

- What questions to ask when someone they cannot identify contacts them over the phone to ask for information or requests that they send, through email or fax, documents out to either the caller or someone else. (Possible social engineering.)
- What to do when they believe that information in a file has been altered in a file. (Possible data integrity issue or access rights violation.)
- Who to contact when they believe that their system, or another system in their department, has been compromised by malware. (Possible trojan, botnet or virus infection.)

The list can continue endlessly, however at a certain point you will lose your audience. So defining the training into a concise half-day training session, which uses historical examples either from the company's experiences or the IT staff's experiences. In this way the staff gain a better understanding of how an attack can be implemented and what to do about it. In most cases, this training should end with an "*If you are not sure, contact the IT Department*" statement.

Another way to get the information out to the company's general staff is to update a daily dashboard on the company's Intranet Website, which identifies the most current security issue(s) that the IT Department is dealing with, and have a "Current Security Posture" color banner or icon. In most cases this is the fastest way to make employees aware of a security issue.

Awareness and knowledge of current or potential issues can be two of your biggest tools in preventing a security breach. When the general staff know what is going on, what they can do, or be looking out for, they can often help eliminate or reduce the impact of an attack or malware outbreak.

Physical Integrity

From here, we as IT Administrators, must take a step back from our focused view of looking strictly at the systems, networks, applications and other devices that we interact with daily as part of our jobs and take in our surroundings. Physical integrity is the implementation of physical barriers and controls that act as a deterrent or countermeasure to any threat against the organization's critical resources. These critical resources include the organization's equipment, property and most importantly, staff.

Physical security is critical to the success of every other layer in our security model. If someone has physical access to your systems, networks, applications, etc., they can compromise every other layer in the model. As such, physical security needs to be logically and methodically planned, implemented and controlled – just because a lock was put on the door does not mean that it is secure.

The easiest way to start defining your physical security is to break it onto two categories; external measures and internal measures. External measures are those performed outside the physical building(s) that house your IT equipment and can include:

- Physical barriers
- Outside lighting
- Lockable gates, doors and windows
- Fire escapes
- Security guards

External physical barriers are often identified under two categories; natural and structural. Natural barriers include trees, hedges, rivers and mountains. They provide us with obstructed views of the location or provide us with difficult terrain to cross to reach the location. With that in mind we must also consider that an obstructed view from the outside looking in may also mean an obstructed view from the inside looking out.

The same goes for items that provide difficult terrain to cross, if we have made it difficult for someone to get to a location, we have also generally made it difficult for us to get to get to a location. Very few of us work in isolated, difficult to reach locations, so for the most part this is an area we will rarely spend a lot of time on.

However, many of us do work in a location where there may be many trees or hedges within our vicinity. The placement of these barriers should be reviewed on a regular basis to ensure that they provide the maximum inside-to-outside viewing capabilities and the minimum outside-to-inside viewing capabilities. Often to accomplish this, we combine these natural barriers with our structural barriers.

Structural barriers can include walls, fences, bars and security gates. Today, perimeter fencing is the most common structural barrier we see, with what is often referred to as chain link fencing being the most common. This type of structural barrier is far more cost effective then building strong solid walls, and is often combined with natural barriers such as hedges or vines which is used to block the outside-to-inside view while giving us the greatest view of all activity within the perimeter.

The drawback with this combination is that it also blocks our view of what is going on outside of the perimeter. If being able to view the activity outside the perimeter is a requirement, the perimeter chain link fencing can be combined with a "top guard" which is the addition of support arms, positioned at the top of each post in the fence, that are angled outwards at 45 degrees. These arms are used to suspend and support three to four strands of tightly strung barbed wire.

Additional security measures such as lighting, cameras, and security guards and bars on the windows can be added to increase the level of security. However, we must also look at the location's purpose and the staff's overall work environment. Are we protecting a top secret research facility or a basic remote site office with minimal staff and systems? Are the staff working in a comfortable, adequately secure location or a prison?

It is easy to go overboard on the implementation of physical security and generally if the security controls and environment are extreme for the type of facility you are protecting, the staff will likely be unhappy and not want to work there. The simple rule of thumb is if the staff feel safe you likely have adequate security, if they feel imprisoned you have gone too far.

As mentioned previously, outside lighting, cameras and guards may also be part of your external security needs. If your organization is looking at these options for a facility, the outside lighting is one of the more important parts of the three and should not be short changed in the budget. The location of the lighting should also be taken into consideration because proper placement means improved visibility for the guards at the security gate and patrolling the perimeter.

The location of the lighting is also important for the implementation of the security cameras. Inadequate lighting means that the camera may not be able to get clear video of an intruders activities or may result in shadow zones where an intruder at night may be able to slip by the camera unnoticed.

It would be extremely rare that any of us work for an organization which provides its guards with night vision goggles or has implemented infrared cameras everywhere. So, spending the extra time, and possibly money, to ensure that the lighting meets the needs of the facility and is optimized to work with other external security measures is always worth it.

As we work our way from the location's perimeter inward towards the facility itself, the last line of defense that meets the intruder is includes the exterior doors and window of the building. At a minimum, the doors should be controlled through some form of electronic controls such as security swipe cards/fobs, and depending on the security needs of the facility can also require an additional PIN code.

The exterior doors should also be strong enough that they cannot be easily broken through, while overhead doors used for vehicle access to the facility should be controlled by an overhead door lift. These overhead doors lifts, like the exterior doors, can also be controlled by security swipe cards/fobs.

As for the windows. The ground floor windows of many of today's commercial buildings cannot be opened and generally buildings with more than three floors have few, if any, windows that can be opened. However, older facilities or facilities that have been converted from another function, such as a low-rise apartment building that has been converted to an office building, may have many or even all of its windows capable of opening.

Again, the main or ground level floor should have its windows either replace with a non-opening variety or should be secured with internal locks and bars. These windows should also be made of a strong shatterproof glass or from "safety glass", which provides high strength yet has a reduced injury risk because the glass does not break into sharp or jagged shards.

Once we have gotten past the exterior of the facility and are now inside, we begin to deal with the interior security risks. Generally, the interior physical security technologies, must deal with the day-to-day operational activities of the staff working inside the facility. It must restrict their movements without limiting their capability of performing their work functions, at the same time it must limit the intruder's capability of moving freely throughout the facility and either contain or prevent the intruder from accessing, altering, destroying or stealing the organization's critical resources.

Most often, internal physical security is achieved through access controls which implement some type of electronic access measure such as the security swipe cards/fobs as well as "Area Designation" levels. These Area Designations are basically restricted access level definitions such as:

- Access Area #1: General/Public Access
    - These areas include common areas such as a building lobby or entrance and most reception areas.
    - Public washrooms are also included in these areas.
- Access Area #2: Controlled Area
    - General public access is prevented to this area.
    - Admittance is controlled to staff or personnel how have business functions within the area and the area may be monitored by cameras.
    - These areas can include business document or file rooms and archive storage rooms, mail rooms, maintenance or electrical closets and staff offices
- Access Area #3: Restricted Area
    - Restricted access to personnel that have been assigned to work in the area, with tighter access rights and restrictions, these areas are almost always monitored by cameras.
    - These areas are typically designated to house personnel or equipment that are critical to the operation of the organization and/or facility.
    - Access is controlled on a 24/7 basis and can even include restrictions on which doors a staff member may enter and leave from, and additional security areas may be implemented within this area. I.e. some staff may be able to enter an interior room within this area while others may not. For example:
        - A restricted area may be further broken into the interior zones containing the server room, mainframe room or data storage room, network operations center, and building environmental control room.

        - While the staff members of this area may work and collaborate together outside of the additional restricted zones, the Server Analyst does not have access to the network operation center zone, mainframe room zone or the environmental control room zone, but can access the server room zone and data storage room zone.
    - These further "zone" restrictions are designed to contain and limit the damage that can be performed by an intruder.

Beyond these restricted zones, many organizations have strong rooms or vaults to provide for onsite data backups, spare critical equipment, and software storage. These strong rooms are designed to prevent forced entry and to protect the items stored inside from floods, fire and other environmental damage.

Once you have moved beyond the physical elements of strong doors, walls and restricted zones we can move back into the more familiar electronic side of physical security. To do that we will discuss alarm systems with security access controls, closed circuit camera systems, aka CCTV and finally environmental controls including backup power systems and fire suppression.

For most IT people these systems are still not part of their everyday IT support routines, however this is changing for more and more organizations as they are commonly becoming linked to the corporate network and controlled by applications running on traditional IT servers. These areas, once the sole domain of the Facility or Building Management Department's, are now partly coming under the control of the IT Department and are often monitored, managed or can only be accessed directly from the organization's Network Operations Center (NOC).

Alarm systems are now often merged with security access controls systems to form a complex solution which provides 24/7 monitoring of all areas of the facility with the ability to control who can access which area or zones. Security swipe cards, fey fobs and card/fob readers (with and without keypads), like those in Figure 7, are all examples of security access control devices which can be used to unlock a door or restrict a staff member or intruders access to parts of the facility.

Figure 7: Samples of Access Control Devices (Security Fobs and Reader)

Along with window and door sensors, most alarm systems are also equipped with motion or intrusion detection sensors. These systems can be used to track the movements of an intruder throughout the facility as the sensors are set off. There are three primary types of intrusion detection systems that are often implemented with alarm system:

- Perimeter sensors
  - Used for doors, windows and skylights.
  - Advantage: they are simple to design and install.
  - Disadvantage: they only detect intrusion that occur through an existing opening.
- Volumetric sensors
  - Used to detect the presence of an intruder in a room.
    - Can include infrared, ultrasonic and photoelectric (beam or laser) sensors
  - Advantage: highly sensitive and can be hard to detect if properly installed
  - Disadvantage: improper installation, large environmental fluctuations, and improper sensitivity calibration can cause significant false alarms.
- Proximity sensors
  - Used to provide direct security for items such as equipment, safes and other security containers.
    - Can include magnetic field, vibration and weight variation sensors
  - Advantage: highly sensitive and very effective for small items.
  - Disadvantage: Can be set of by accidental touching (magnetic field/weight) or large passing vehicles (vibration) and as a result without proper calibration for the environment can induce a significant number of false alarms.

Alarms are common place and for some businesses they can go off quite often. So much so, that in many cities, an alarm permit is required to have one. There are also hefty fines from the police department for false alarms and if there are too many false alarms the police will serve the organization with a notice that they will no longer respond to the alarm, i.e., they will revoke the alarm permit.

To ensure that the alarm is properly responded to, there are some general rules of thumb regarding alarm systems, as follows:

- Alarm systems are **not** designed to prevent an intrusion.
  - They are designed to alert security personnel or the police of an actual or attempted intrusion.
  - Prevention of an intrusion comes from the strong doors, windows, gates, access controls and other physical barriers that you have put in place.
- By themselves, alarm systems do **not** act as a deterrent
  - Intruders are often not worried about the alarm system, they are worried about the response and more importantly how long the response takes
    - Think about it, how many times have you heard a car alarm or home alarm go off and not even looked in the direction of the alarm.
- An alarm system is worthless if there is no response.

    - Many alarm monitoring organizations will provide security guard response to alarms for an additional monthly fee. This guard response is often cheaper than numerous fines from false response fines from the police department.
        - The responding guard will typically try to determine if the alarm is genuine or false.
        - If it is a genuine alarm they will contact the police who will actually move their response to the alarm up in priority because the guard has verified that an intrusion has occurred.
- All alarm systems have a weakness that allows them to be defeated or circumvented.
    - Accessible cabling that can be set to loopback for testing purposes, ability to silence the alarm sirens, motion detectors that only detect motion if it occurs at a certain rate or can even be blinded are all example of these types of weaknesses.
    - Understanding these weaknesses and compensating for them with different types of intrusion detection sensors for critical security or restricted zone areas will help reduce these weaknesses, however they can never be truly eliminated.

Closed circuit camera systems, or CCTV systems, are often considered an extension of or integral part of many alarm system today, because many of them are digitally-based solutions that record to a hard drive storage device. The cameras are often coupled with or have built in motion sensors so that they are only actively recording when the sensor detects some form of motion. However, these systems can be kept completely separated and on their own network to provide an additional monitoring solution that has to be defeated or circumvented separately from the alarm system.

By itself, a CCTV solution really only acts as a monitoring and recording systems which can track the movements of an intruder and record evidence of malicious activity. However, they do have some advantages:

- One person can monitor several locations at once
    - Multiple camera, multiple screen systems can allow a single individual to monitor several location at one time from a single, secure and even remote location.
- The information provided and recorded is visual
    - Visual information provided much more information than all other types of censors combined and allows the person monitoring the camera system to direct responding security or the authorities much more accurately to an intruder's location.
- There is a wide variety of camera solutions to choose from, meeting nearly all viewing needs
    - As discussed previously there are outdoor and indoor camera applications, but there are also cameras for:
        - low light or infrared applications
        - zooming or panning applications and
        - high definition applications

There only two real disadvantages of the CCTV system:

- Improper implementation of the camera or the wrong camera type for the job
    - Improper camera setup or using the wrong camera type will not provide you with the viewing capabilities needed for the effectively monitoring an area and as a result could provide the organization with a false sense of security.
- The Human Factor
    - There are really two things covered by the Human Factor
        - Most people who are monitoring the camera systems have this as only one function of their job duties. They do not sit there, staring at the camera monitors during their entire shift, and their attention is drawn away from the monitors to their other task or duties.
        - Most people have a short attention span, especially if they are doing something which is heavily monotonous. Watching unchanging, camera monitors can become exceedingly boring and in the absence of stimulation the person will find something else to direct their attention towards.

Finally, let's discuss the facility's environmental controls, including fire suppression and backup power systems. Often the reason that many IT departments began managing environmental controls is because they began managing large data centers. These data centers required specialized environmental controls that are designed differently from the buildings general heating, ventilation and cooling (HVAC) systems.

For many IT departments cooling, humidity control, prevention of static electricity and providing redundant power systems are critical to managing the data center. These environmental elements are monitored much more rigidly than they would be as part of the normal facility or building HVAC systems. In fact, for many data centers, large or small, this environmental control management is completely separated from the building HVAC systems.

Fire suppression solutions also became part of the IT management portfolio with the realization that water and expensive electronics such as main frames, servers, drive storage units (NAS and/or SANs), tape backup units, phone systems, etc., really don't mix. In many cases, alternative solutions to water-based sprinkler systems are installed. These fire suppression systems are generally chemical-based and are often contain fire suppressing powder or gas as seen in Figure 8.

Figure 8: Chemical-based Fire Suppression System

For most IT people, managing the monitoring requirements of the environmental systems is the easy part, after all, once they are properly configured most of these systems will alert that something is wrong long before it actually becomes a critical problem. While, figuring out what solution is actually needed for their facility or building is the hard part and can be a career in itself.

The same goes for nearly all aspects of the Physical Integrity portion of the Defense-in-Depth model. In the end, your job is really to protect the systems and ultimately the data on those systems. Knowledge, or more importantly, the use of other people's knowledge to provide expert input into designing and implementing the physical solutions needed for a given facility/building is critically important.

Every facility and building is different, every data center is different and every organization's critical infrastructure is different, as a result, every Physical Integrity solution design will be different. It would not be possible to cover all of the possibilities here and you will need to bring in the "experts" to consult and provide input into any area where the organization does not have adequate internal knowledge.

Perimeter Integrity

Perimeter integrity takes us back to the IT world we are more familiar with. For this area of security we are really going to focus on two areas of the perimeter; the router and the firewall. The primary reason we focus on the router is because this is often the first device an attacker encounters when they attempt to gain access to an organization.

Routers of the past often had limited security capabilities and even routing protocols that could be configured, this also limited their effectiveness in preventing an attack. However, todays routers and be configured with many kinds of Access Control Lists (aka ACLs) for basic security filtering and often include network services such as DHCP, BootP, CDP (for Cisco Routers), DNS and HTTP.

These added network services features make the router an excellent device for running small or remote offices, since domain controllers and other network services servers do not need to be installed. The problem is that all these added features also means a greater potential for increased security holes being opened through misconfiguration.

New services aside, routers have also suffered from some several inherent weaknesses including:

- weak or non-existent patch management,
- weak and/or crackable password security,
- long timeout or no timeout periods for administrative interface connections.

Traditionally, patches and upgrades for routers meant taking the router offline and updating the operating system manually. This is often a problem for IT staff who need to update routers that are located at remote or branch offices of an organization. They must literally travel to these offices to perform the update.

Even those that are running in an organization's main office will often suffer from the inability of the IT staff to take the router down to patch or upgrade the software. Unlike server operating systems, many organizations do not look at patching their routers on a regular basis. The thinking is, that once it is installed and working there is no need to change it and for most organizations the router is still one of the few configure it and forget it devices until it needs to be replaced.

This is a problem, since there are many updates that are or have been released which provide fixes for security weaknesses discovered in the operating system or protocols running on the router itself. Vulnerabilities in the operating systems, the routing protocols and the added network services are all areas which needs to be patched and updated on a regular basis, as seen at the following links:

http://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20041110-dhcp

http://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20130801-lsaospf

http://tools.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-20120328-mace

http://www.juniper.net/techpubs/software/erx/junose112/sw-rn-erx-1123/sw-rn-erx1123-body-11.html

http://www.cvedetails.com/cve/CVE-2012-3268/

Thankfully, today's newer routers, or at least the router vendors, are providing better solutions for patching the router software but there are still many organizations that are running older routers in remote offices. These devices, as seen by the short list of links above, can contain many vulnerabilities and it doesn't matter which vendor's router you buy.

In short, when it comes to these devices awareness of the potential vulnerabilities in the routers running in your organization is your biggest tools to ensuring that you limit the potential for an attacker to use your own equipment against you. Shutdown unnecessary features, protocols, network services and remember, if it is easy for you to search for these vulnerabilities, it is easy for the attacker to search for them as well.

While we are referring to the older routers which are still in use across the Internet world, let's take a moment to discuss the password weakness issue. Today, depending on the router vendor, there are reasonably solid solutions to securing your organization's routers with authenticated login solutions. But this is not always implemented, and using Cisco's configuration process as an example, many systems admins begin basic router configuration using the insecure "*enable password*" command

This made it easy for logging in and out of the router during the configuration and testing phase. However, when they migrated to using the "*enable secret*" command they used the same password again. Using the same password defeated the purpose of the stronger encryption provided by the "*enable secret*" command, since the "*enable password*" command used a weak password encryption which was easily crackable.

The advice here is that, these are very good ways to ensure secure, authenticated logins are implemented for your routers, even most of the older routers were capable of using TACACS and RADIUS servers for authentication purposes. Using these more secure authentication methods with help to prevent attackers from accessing the router and, again, using it against you.

Finally, the administrative interface timeout, a weakness caused mostly by accepting default settings and poor configuration management. For many older routers, the default timeout for the administrative ports was unlimited, while newer routers are generally limited to 10 minutes. These timers should, based on ability to access the device, be limited to 5 minutes or less.

If there is no need to be able to access the router through the administrative interfaces, then access should be eliminated in the configuration. For example, to turn off administrative access for the aux port on a cisco router, you would use the **no exec** command in the auxiliary line configuration mode.

Again, deciding on whether or not to allow access to the administrative interfaces and how long to set the timeouts should be based on the ability to access the device. If the device is located in a Restricted Area, where access is limited to IT staff only, then reducing the timeouts is often sufficient. If it is located in a remote site such as a branch office, where security of the device may be lax, then eliminating the ability to access the ports may be more appropriate.

However, your decision on how to configure these interfaces must be based on both your organization's security posture and the supportability of the routers. While it is best for security purposes to prevent any access to the devices physical administrative interfaces, it prevents local troubleshooting to the device in the event that you are unable to remotely connect to it.

From routers we move onto firewalls, often the second device encountered by the attacker. We have already covered the different categories of firewalls so in this section our focus will be more on the concepts of firewall rules and the placement of firewalls with respect to perimeter security.

Today's firewalls, such as those produced by Checkpoint, Cyberoam, SonicWALL, and Cisco often have many additional features beyond the basic firewall functions. While the units are still classified as firewalls they are more appropriately labeled as multi-function security devices, while a basic firewall has really two functions; allowing approved traffic to pass through it and preventing unapproved traffic from passing through it.

While this sounds like the same thing, allow approved vs. deny unapproved, it isn't. Approved traffic is traffic that is allowed into or out of the network protected by the firewall, generally based on IP Addresses, protocols or TCP/UDP ports. Approved IP Addresses, protocols, and TCP/UDP ports are configured on the firewall by the Administrator. It is also based on the firewall interface, e.g. internal interface vs. external interface.

In general this means that if the firewall is configured to allow only web traffic on TCP port 80 outbound, then no other traffic including HTTPS, FTP, etc. will be allowed to traverse the firewall from the internal interface to the external interface and the response, by default, is allowed back into the network. It means that no traffic is allowed to initiate a connection or traverse the firewall from the external interface to the internal interface.

It also means that if an HTTP request is allowed out on TCP port 80 then the responding web server is allowed to send its response back through the firewall on TCP port 80. However, if the website responds with some of its information on a different TCP port then this information is not allowed back in and is effectively blocked by the firewall.

This is where the "deny unapproved traffic" comes in. Most people think of unapproved traffic as traffic that is not allowed to traverse the firewall from outside to inside under any circumstance. On most firewalls, not defining the particular type of traffic, i.e. a TCP/UDP port or an IP Address range explicitly denies it.

This is because the last rule in any firewall ruleset is a default "catch-all" deny rule. If the traffic did not met any of the approved traffic rules, it is explicitly denied. However, using this default also denies legitimate traffic that has not been properly configured to pass through the firewall.

It is this valid, yet blocked traffic, which is the biggest headache for Firewall Administrators. For example, while the workstation that is requesting the webpage from the remote server is allowed to do so and the remote server is allowed to respond back, it can only do so on TCP port 80. As a result, legitimate communications with the website such as login information or site certificates

which are passed through TCP port 443 (HTTPS), or Kerberos Keys passed on TCP port 88 would not be allowed.

This traffic is valid and most likely required for full use of the website being accessed however, it is not allowed out and is therefore the response is not allowed back in either. As many Firewall Administrators are finding out today, the Bring Your Own Device (BYOD) policy being implemented by many organizations has caused endless headaches as each of these mobile devices use or require TCP/UDP ports for its many apps and/or services that they never intended to open on their firewalls.

For example, iPhones and iPads, use many of the common TCP/UDP ports for email, DNS, web surfing (HTTP/HTTPS) but also uses dozens of additional ports for syncing (iSync: TCP-3004), notifications (Apple Push Notification: TCP-5223), chatting/messaging (iChat: TCP-5269, 5297 and TCP/UDP-5298) and many other services.

While the Administrator is dealing with these services for the organization's Apple device users, the Blackberry device users require another set of ports that need to be opened for syncing (BB Sync Server: TCP-3200, TCP1433 and UDP-4185 to UDP-4499), notifications (BB Dispatcher: TCP-5096, 3200, 3101 and 1433, UDP-4071, 4185 to 4499), chatting/messaging (BB Messaging: TCP-5096, 1433 and UDP-4070, 4071, 4085 to 4499), etc.

From there we can add the ports used by the apps that run on Samsung tablets and phones, Microsoft's Surface, Google's Nexus, and many others. All of these mobile devices, or more importantly, the apps running on them use numerous TCP/UDP ports that must all be managed to allow access through the firewall.

Depending on the organization's policies, this is valid and legitimate traffic that may not be allowed through their firewalls because of improper or poor configuration definition of approved traffic, even if the device, the app running on the device and the user are all approved to do so. The "catch-all" explicit deny does not "deny unapproved traffic" it denies all traffic that is not configured as approved.

So what's the difference?

To deny unapproved traffic, means that the Administrator has spent the time to define and configure deny statements in the firewall configuration. Creating these rules also allows the Administrator to setup more appropriate alerting and logging of any traffic matching these "deny unapproved traffic" rules.

So why not just leave the approved traffic rules and let the "catch-all' deny rule drop all of the other traffic?

It is mostly because of how the firewall processes the rules. As the firewall processes the traffic, it compares the information in the packet to against each configured rule, in order, and if it finds a match it processes the traffic according to what the rule said to do with the traffic. If there is no

match, then the "catch-all" explicit deny rule is left to do its job deny all other traffic not configured as approved.

However, in the right circumstance, bad (unapproved) traffic may be allowed to pass through the firewall because it matched an approved rule and never hit the "catch-all" deny rule. For example, let's look at a very simplified scenario where we assume that an organization has a Web Server behind its firewall in the DMZ (if you are unfamiliar with this term we will explain the purpose of the DMZ later).

The Administrator allows traffic to this Web Server by configuring a rule which states that any inbound TCP port 80 traffic is to be redirected out the DMZ interface, and possibly to the specific IP Address of the Web Server. Once this traffic is sent out the DMZ interface the Web Server receives it, processes it and responds. The response travels back to the DMZ interface, through the firewall, to the system which originated the request.

Two weeks later, the website suffers a DDOS attack and is inaccessible to the organization's clients. After scanning the enormous log file, the Administrator determines that all of the attacking traffic came from an IP address range located in a country that the organization does not have any business relationships with.

To prevent future attacks, the Administrator must now create a "deny unapproved" traffic rule which is designed to prevent traffic from the IP Address range where the DDOS attack originated, while still allowing the approved traffic through. Really, the Administrator has a couple of options here:

- they can deny all TCP port 80 traffic from the identified attacking IP address range,
- they can deny all traffic from the identified attacking IP address range,
- they can deny all TCP port 80 traffic from the identified country the IP address range is assigned to,
- they can deny all traffic from the identified country the IP address range is assigned to.

The choice itself is up to the Administrator, however as a rule of thumb, the preference would be to implement the most restrictive posture first and then move towards a lesser restrictive posture, when and if, business needs dictate. In this case the Administrator, took the most restrictive posture and blocked all traffic coming from the country in which the attack originated, simply because the organization has no business relationships with organizations in the other country.

To define this rule, the Administrator verified the IP Address ranges that have been assigned to the country. There are various, documents and online websites which can provide this information for an example visit the website http://www.nirsoft.net/countryip/. Once these address ranges are defined in the rule, the Administrator must then place the "deny unapproved" traffic rule before the "allow TCP port 80" approved traffic rule.

Remember, the firewall processes the rules in order, so the deny unapproved traffic rule must be before the approved traffic rule. Any TCP port 80 traffic that comes from legitimate sources, i.e.

IP address ranges that are not defined in the deny rule, will not match the new deny rule and will then be processed by the next rule which is the "allow TCP port 80" traffic rule.

Again, if we do not implement "deny unapproved" traffic rules, sooner or later bad traffic posing as legitimate traffic will make it through the firewall and cause a problem. Once the Administrator takes the time to configure both "allow approved" and "deny unapproved" traffic rules, then the "catch-all" explicit deny rule (by default, the last rule) can properly do its job and filter out the remaining traffic that was not allowed or unwanted.

Another aspect of firewalls that must be taken into consideration is placement. The placement of firewalls is as important as proper configuration of the rules. In fact, where the firewall is placed helps to determine the rules that need to be configured.

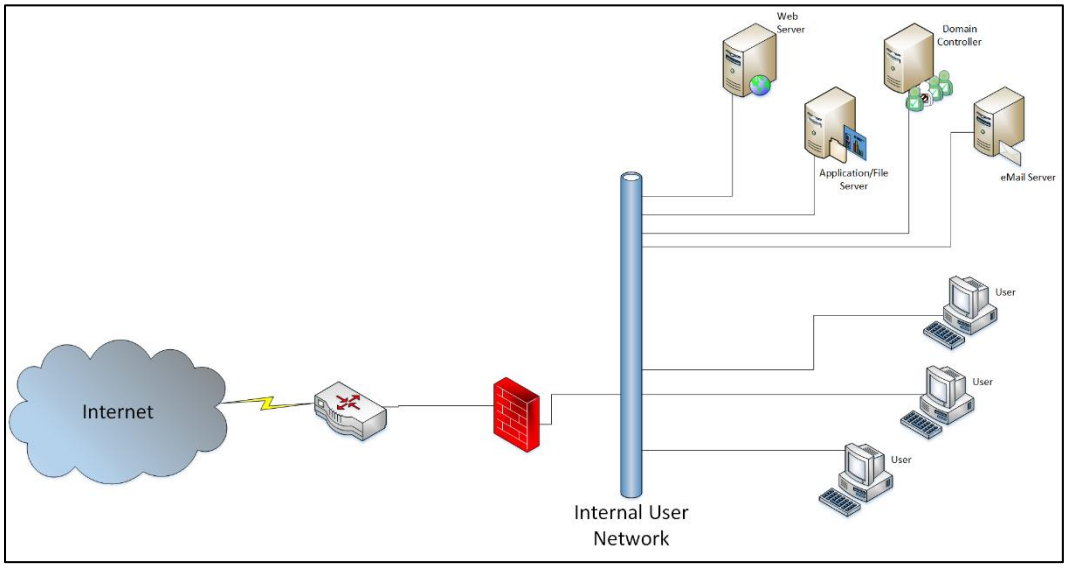


Figure 9: Small business firewall implementation

Small organizations may only use one firewall as a perimeter security device, as seen in Figure 9. However the concept of perimeter security as just the line between the internal network and the external network (or Internet) has been redefined within larger organizations which use multiple firewalls. Perimeter security in these organizations is the line between a less secure network and a more secure network.

With this redefinition in mind, we can look at three different locations that organizations implement firewalls to provide enhanced perimeter security. This is not to say that these are the only locations in which firewalls should be implemented, again anywhere an organization defines that there should be a separation between devices (less secure to more secure) a firewall is most likely the device that will be implemented.

Looking at Figure 10, we can see the traditional implementation of a firewall between the external router and the DMZ. The DMZ (aka Demilitarized Zone) is the network where publicly accessible devices are placed, i.e. nearly everyone who can access the internet can access these devices. These devices include web servers, email servers, ftp servers, fileshare mirrors, etc.

If an organization uses VPNs to allow remote sites or users to connect to internal resources, it is often allowed to pass directly through this firewall unimpeded and unmodified. This is because this firewall is not in a position where it can confirm a user's authentication credentials or certificates. This authentication is performed further into the network where the authentication credentials, certificates and even user access rights can be verified against the internal domain servers.

This firewall is our first real opportunity to stop an attack or block bad traffic. It is also our first opportunity to raise any alarms to the Administrators that something may be wrong. This is the firewall from our previous example regarding the DDOS attack and the need to better define our rules by denying unapproved traffic.

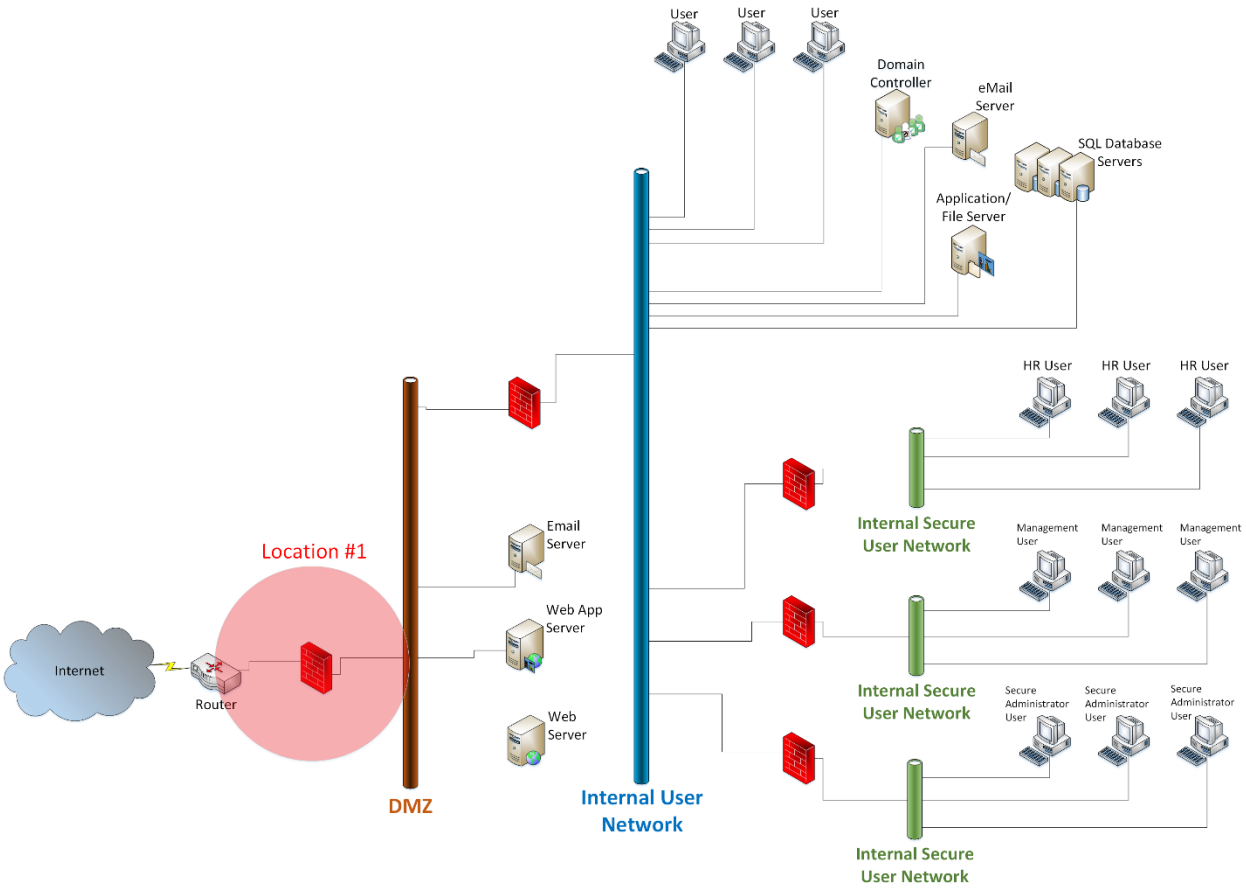


Figure 10: Recommended firewall location #1

The second location for firewall placement, seen in Figure 11, is between the DMZ and the corporate internal network. This is where the majority of the organizations user workstations and general business servers reside. Only traffic bound directly for devices inside should be allowed through this firewall, most of which is response traffic from user requests, such as HTTP traffic. DMZ servers are also allowed to talk to internal devices through this firewall under very strict rules and for many organizations, VPN traffic is often authenticated at this point.

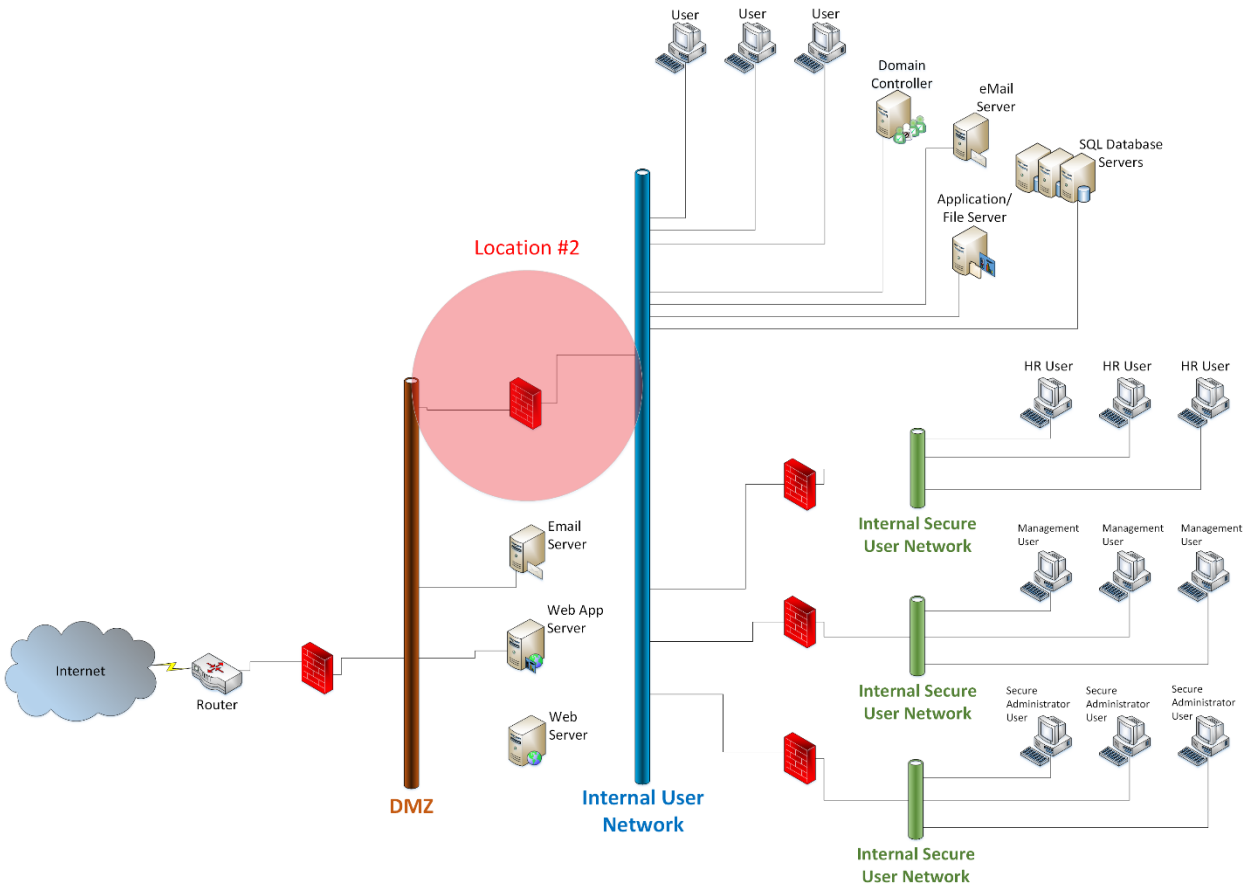


Figure 11: Recommended firewall location #2

For example, an active sales website would be allowed to talk to the internal SQL server to pull pricing information on products that the user adds to their online shopping cart. It may also be allowed to send completed orders to the SQL server for processing. However the firewall rules would require that the TCP ports, IP addresses, certificates and even data formats for this traffic meet strict rules to prevent someone who may have compromised the order entry server from using it to steal customer information through a modified SQL Injection attack.

Basically, the purpose of this firewall is to protect the internal network from external and DMZ launched attacks. As such, the rules on this firewall must be thought through and thoroughly tested before being implemented. Even if the servers in the DMZ have little need to read data from or write data to the internal servers because most of the traffic passing through this firewall is just web, email and the occasional file transfer.

However, if the services provided in the DMZ are transaction-based such as online order entry then the requirements to protect access to or modification of the data is of paramount importance. The Administrator, will need to have a much greater understanding of the traffic flows and the data contained within those traffic flows in order to determine what is and is not valid.

Another important factor to consider regarding this firewall is that is cannot be from the same manufacturer or the same model as the firewall discussed in Location #1. This is because, if both

firewalls were the same, then any security flaw or weakness found in one device would also exist in the other. I.e. if an attacker can breach the first firewall, using a known exploit, they can also breach the second firewall with the same exploit and gain access to the internal network.

Figure 12, identifies firewall location #3. This firewall is used to protect more secure internal network(s) from the general user network located behind the firewall in location #2. Its purpose is to protect against attacks that are launched from the internal user network.

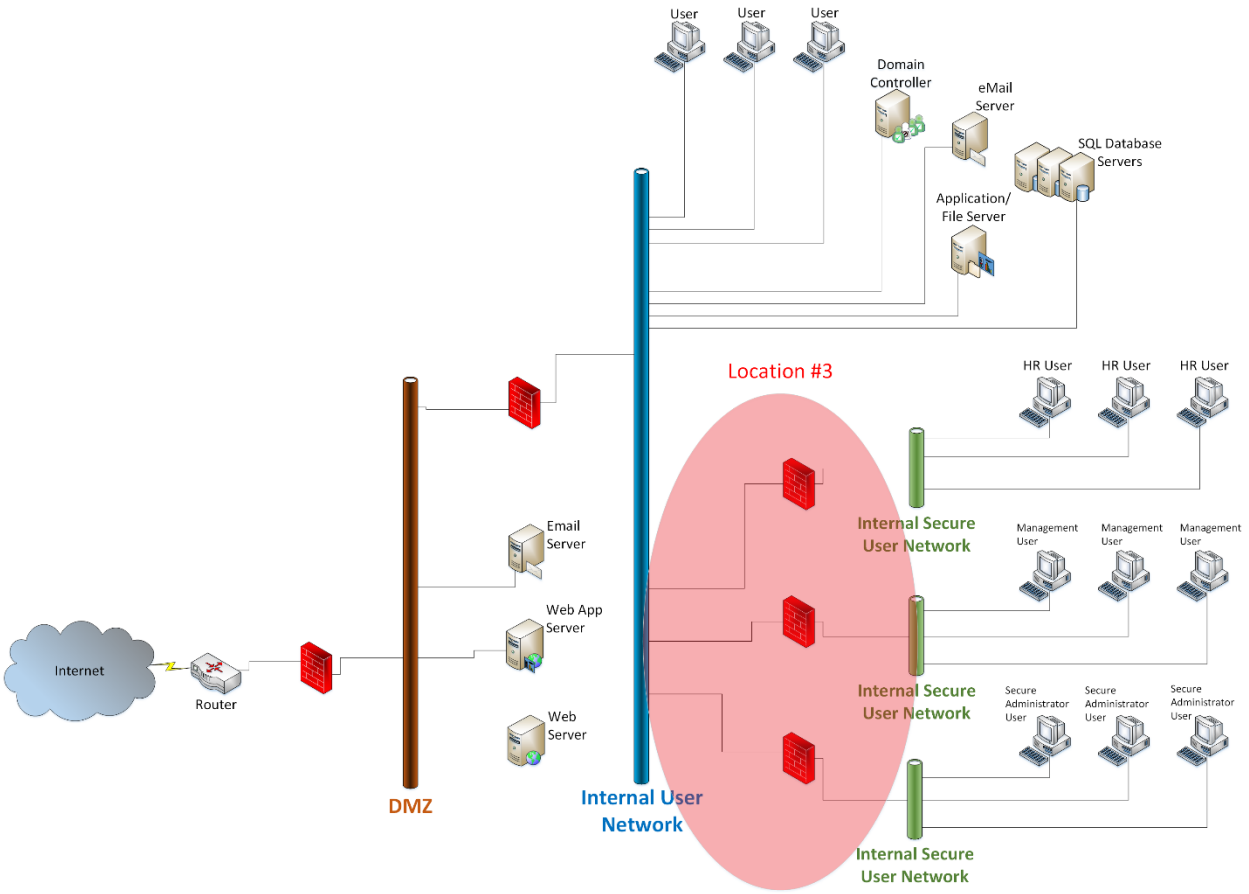


Figure 12: Recommended firewall location #3

For many organizations these internal perimeters are needed to ensure that confidential corporate information and processing servers have an increased level of protection. Like the firewall located in Location #2 this firewall should have its ruleset carefully thought out to allow users behind this firewall to authenticate to the domain controllers in the general user network. Alternatively, a secondary authentication server may be located in this network and allowed to replicate domain authentication information.

Also, data transfer between devices in these internal secure networks and the general user network should be carefully controlled through the firewall since, by design not all devices will need to communicate through the firewall. This ensures that only the devices, their communications protocols and defined data types should be allowed through.

The rules should also ensure that the firewall alerts the appropriate Administrators in the event that something abnormal attempts to communicate through the firewall. Again, the more alarms/alerts that are in place, the more likely that an attacker will become noticed and prevented from gaining access to any critical corporate data that may be of value to them

Network Integrity

While many people, and other Defense-in-Depth Models define firewalls as part of Network Integrity, this model looks at network integrity differently. This is because there are known ways to breach the perimeter (i.e. the firewalls) and trick these devices into letting bad traffic through. As a result, when it comes to Network Integrity this model looks at the internal network devices and the monitoring of the traffic on that network.

Network integrity also deals with VPN (Virtual Private Network) traffic since, the whole purpose of using VPNs is to give someone direct access to the internal network across unsecure public networks. For many organizations, VPNs are used to connect remote offices to the head office (site-to-site) or to allow individual remote or roaming users to connect to the head office (client-to-site).

The main difference between the two types is in how long the connection exist for. Site-to-site VPNs are often created between the firewalls located at the remote office and head office. These VPN tunnels are longstanding, meaning that once established they rarely are disconnected, allowing all the remote office users to communicate to the head office systems at the same time over a single link.

They are implemented to allow the organization to take advantage of low cost, high speed business internet links instead of the more expensive dedicated point-to-point service links such as Frame Relay and provisioned T1 or fractional T1 links. When properly implemented, these site-to-site VPNs should be as secure as the dedicated link.

The client-to-site VPNs are temporary link establishments. Remote/roaming users create these VPNs on demand use them as long as required and them disconnects them. Typically, the client device uses a VPN client application that connects to an internet facing VPN server at the head office. Organizations will often enforce a maximum inactivity time limit on these connections to ensure that the remote user has not forgotten about their connection and left an unattended computer connected to the corporate network.

Despite which type of VPN connection is established there are three (3) core functions that the VPN performs to ensure security of the link:

1. Data Encapsulation
    - The transmitted packet is encapsulated within another packet with different IP addressing
    - The encapsulating packet contains the VPN assigned source address of the client and destination address of the VPN gateway.
2. Encryption
    - Both ends of the VPN tunnel will negotiate the encryption keys and will encrypt all data that is passed through the VPN tunnel

3. Authentication
    - Depending on the implementation, authentication of the user or device will occur using digital certificates, and the tunnel protocol used determines the authentication protocol that is used.

As stated earlier, VPNs can be a cost effective and secure connectivity solution and the main problems that arise from VPN implementations are mostly human configuration issues dealing with remote/roaming client devices. These problems include:

- unattended remote/roaming client workstations that are connected with no timeout for inactivity, allowing anyone taking control of the device and access internal data
- automated installation and/or configuration scripts which allow the remote device to connect without the user entering in a User ID or password.

These two issues are created by the Administrators themselves, mainly to make it easier for remote staff, especially the less technical, to connect and stay connected. The problem is that making things easy for the users, generally makes it easy for the thief that steals the user's laptop. It also makes it a logistics nightmare when it comes time to change these automated scripts with embedded User IDs and passwords, since many of these remote/roaming staff are rarely in the main office where the devices are configured.

Resolving this problem is easy, simply do not allow automated scripts on remote users laptops, forcing them to enter their User ID and password. You must also force them to change their password regularly as you would any other user.

Next we need to look at the network devices themselves and how they are configured. In particular, we are going to discuss network switches, since this is where a significant security problem can lie. For most medium to large organizations, especially those who have implemented physical security, most network switches are not available for someone to gain physical access to.

Smaller organizations and remote offices often do not have these physical security measures in place and in many cases the network switches are located in a spare unlocked closet turned into a central location for IT purposes. These easy to access locations often allow the users in these offices to make changes to their port connections, wiring and even connect additional, non-company systems to the network without the need to get the IT staff to do it for them.

For the IT staff, the problems with this type of situation are numerous:

- staff changing office locations can mean that the port they connect to is not tracked on the monitoring systems,
- new systems can be and are added that are not
    - configured to organizational standards,
    - joined to the domain properly,
    - do not get properly patched and
    - may not have any anti-malware installed or configured to get updates
- vendors can be connected to the corporate network without proper security notifications and configurations to isolate their systems

This list can go on and on, simply put, the problem is that this situation has now breached two of the security measures you are trying to establish with a defined Defense-in-Depth process; Physical Integrity and Network Integrity. Yet the solution to this problem is often built into the network switches themselves.

VLANs, port security and even port shutdown solutions exist on many vendor switches today. VLANs provide a means to isolate network traffic on the same switch. Only devices on the same VLAN can talk to each other directly and traffic must be routed to other VLANs even if the two VLANs are on the same switch.

Many medium to large organizations will implement VLANs in there central office due to the large number of staff located there. The implementation, along with firewalls, helps to further isolate traffic to different related departments based on organizational structure. However, smaller organizations and remote offices fail to do so simply because there is an increase in configuration complexity and as a result an increase in the time it takes to make a change.

While I agree that the configuration complexity increases, so does the security level and the risk level to the organization is similarly reduced. For small organizations or remote offices, the general configuration of defining VLANs is relatively simple since they generally only need two VLANs; on for the general staff and one to separate the unused ports to an isolated network that only has internet access.

By doing this, staff cannot simply move offices because they want a window view instead of the central cubicle they were in before. Computers brought in or bought by the small organization or remote office staff cannot be just simply plugged in to the corporate network without being properly configured, correctly joined to the domain, and have approved anti-malware software installed.

Likewise, vendors or at least their sales people, cannot just be simply connected to the corporate network while visiting to give a presentation, once plugged into an unused port they are connected to the VLAN that is isolated to internet traffic only. This allows them to give their presentation, but prevents them or their computers from running scans on the corporate network or even worse spreading worms, bots, or other malware throughout the corporate network.

It’s a simple rule when dealing with rogue machines such as vendor systems that connect to your network:

- The Administrator, cannot rely on the security practices or lack thereof, of other people or organizations to guarantee the security of your systems.

As the Administrator, it is your responsibility to ensure your organization’s security posture is met and adhered to. The visiting vendor’s machine may not have had an anti-malware application installed, have had the anti-malware disabled, it may not have been updated in months. Simply, you cannot rely on them to ensure that their systems are as secure as your own.

Another option for improving the security of your network, through the switch configuration, is to turn on MAC address filtering at the switch port level. Every network device has a defined permanent MAC address, aka the "Burned in Address" or BIA, for each network port. Configuring only a specific workstation's MAC address to be allowed to talk on a specific switch port, prevents anyone other than that workstation from connecting to the network on that port.

If the user moves locations, their computer will not work without reconnecting the port cabling to allow the new location to be connected to the port that is defined for that user's workstation. New computers brought in or bought will also not be able to connect, until they are properly defined on the systems and configured properly.

While, this may seem like a lot of work, it does ensure that only the companies systems are using and or connected to the company's network resources. It also ensures that the organization's Administrators are aware of any changes being made, and what device is connected where. This minimal increase in configuration complexity, and time to setup the systems properly also saves much time in troubleshooting and repairing problems, such as a malware outbreak, that are caused by having such an open system design.

Finally, there are two other solutions that can be implemented to help improve network integrity, one configuration-based; turning off unused ports, the other implementation-based; unplugging unused ports. These solutions work together, since in many small organizations and remote sites, all the available user ports on the patch panels are often directly connected to the switch, even if the ports are unused.

This is done to reduce the amount of time it takes to get a user connected to the network when they move to a new office location or are hired and have a new PC setup for them to work from. Or it is done simply because the remote site staff simply does not understand the network wiring and therefore everything is connected so that no one has to touch the patch panel wiring to get a new or moving user connected.

The preferred solution is to have someone, trained by the IT staff, at the remote site perform the patching for network connectivity as needed and leave any unused switch ports disconnected. It's a simple solution, someone cannot plug into the network from an empty desk if there is no cable connecting the desk to the network switch.

However, in the event that there is no one in the office to do this work, the easiest thing for the Administrator to do is to turn off the unused network ports on the switch. Most managed switches allow the individual ports to be disabled, thus preventing anyone from being able to connect a computer even if the network ports are all physically wired up.

One additional network integrity issue that needs to be address is with wireless networks. Nearly all organizations use wireless network at some level. Like network switches, how they are configured and managed will improve or reduce your overall network integrity and there are several things that must be considered when selecting wireless access points.

Like other network devices such as switches and routers, the configuration abilities of wireless access points must be matched to the organizations overall security posture and policies. If a proper security posture and policies are defined it means that simply buying cheap off-the-shelf home-based wireless access points will not meet the security needs of most organizations, small or large.

Business class wireless access points, are often much more expensive but allow the Administrators to monitor, manage and configure the device to work with the organization's security posture. Most are configurable to work with VLANs or multiple VLANs, and can even be configured to isolate traffic by allowing the configuration of both a corporate wireless network and a guest wireless network that only has limited internet access and no corporate access.

The wireless guest network works similarly to a wired guest VLAN, in that visiting guests such as a vendor's sales team can come to the organizations office, be given temporary password to the guest wireless network and be allowed to use the internet, but prevents them from accessing corporate network systems.

The business class access points can often be configured to work with a centralized wireless management controller. These devices allow the Administrator to manage all of the organization's access points through one central management system and can help prevent the implementation and use of "rogue" access points on the network.

These rogue access points are often setup by users who do not realize that their installation creates a security hole that allows direct access to the corporate network. This is especially true in small offices or remote offices, where a mobile user brings their laptop with them wherever they go. To make it easier to connect in the office, they will often bring in a home-based access point an plug it into the network instead of their laptop.

This allows them to come and go as they please without physically connecting their laptop to the network. It also allows them to connect their personal cell phone, tablet or other mobile devices to the network for internet access without having to configure them to work with the corporate wireless network.

Since this wireless access point is usually the same model of device that they use for their home wireless network it is often not configured with strong encryption, passwords or other security measures, if it is changed from the default settings at all. It's a wonderful device, since they can connect the same as if they were at home.

As an Administrator, you quickly realize that the problem is these minimal or default configured devices also allow anyone to connect to the network as well, which completely bypasses every single security measure you can put in place. So it is of the utmost importance that you can locate and remove any of these rogue access points.

It should be important to note that each and every small organization or small remote site is different. The final solution that is implemented at any site, be it for a small organization, a

remote site or a large organization's main office is always going to be different and must be well thought out to ensure the best security approach with a minimal impact of the user's ability to perform their job functions.

For most Administrators this means that the systems they choose, and the configurations that they put in place must work well together. They must also be designed and configured to allow flexibility for the users in their ability to connect, while still giving the Administrator the monitoring, controlling and restriction capabilities that are needed to respond to security related issues quickly and efficiently.

System Integrity

System integrity is one of the tougher issues that must be dealt with by all Administrators. It deals with ensuring that the workstations, laptops and today's smart mobility solutions such as iPhones, Android-based phones, iPads and other tablets are meeting the organization's security policies.

The BYOD (Bring Your Own Device) policy of many organizations only serve to complicate the Administrator's ability to ensure system integrity, since the Administrator really cannot force the owner of the device to remove or install software to comply with security. About the only thing that the Administrator can do is limit the access to the network and systems that these personal devices have.

We will discuss ways to manage the BYOD or personal devices a little later, first let's discuss the devices that the Administrator can enforce security compliance on; the company owned devices. These devices will most likely be connected to the corporate network and running corporate software.

For most organizations, this will likely be a Microsoft Windows-based solution since, based on many market share reports, Microsoft operating systems run on 80%-90% of all desktops and servers all over the world. However, when it comes to securing the organization's desktops, servers, tablets and other mobile solutions it really doesn't matter what operating system is running, the process to implement security is similar for all of them.

To begin, we start even before the operating system and look at hardening the hardware of the device where we can. Why do we start here? The answer is simple, it is performed to prevent modification to the system configuration.

Many people don't realize how many computers are actually installed in the world. Nearly every system built in the last two decades has some form of computer processor built into it for control or monitoring purposes. Where these systems are connected to our organization's infrastructure is a potential security hole, a risk point that must be mitigated.

One area that is often overlooked are isolated, stand-alone systems. These are located in remote locations, monitoring and collecting data on things like gas flows, power systems, wind speeds, etc. In isolation, there is no one to around logging into these systems on a daily basis. They sit in

their location day after day passing data back to the master system and are afforded no more protection than the door to the building they are located in.

Because these systems are often data collectors, they have the ability to connect to the organizations network through some form of communications channel and automated process. Anyone who can get to their location can use them to eventually gain access to the organization's more internal infrastructure, after all many of these systems are "trusted" corporate devices.

Like the isolate, stand-alone system the company laptop which often travels with its users and all too often is stolen, lost or "misplaced", faces the same security issues. If there are any automated processes or applications installed or the user has passwords recorded in cached files the thief can use this device for direct access into the organization.

The hardware and the hardware configurations on these systems can be protected in most cases through the use of BIOS passwords. BIOS passwords can be used to prevent the altering of bootable devices and system interfaces, they can even be used to prevent the system from booting at all. These are important critical control points for any system since access to the system and alteration of the first two elements can give the person directly accessing the device the ability to gain information or determine configuration settings that they need to bypass security protocols.

Again, like all of the other solutions we have covered up to this point, we must consider if the solution is practical or not. While enabling a power on password may seem like a good idea, it is not always practical. Consider this, if we enabled a power on password for all of our devices, the PC monitoring and reporting gas flow rates on a remote pipeline 3 hours in the middle of nowhere is probably not in our best interests.

Every time the device loses power, we will need to travel to it to enter in a password so that it can finish booting. This will happen much more often than you think. The fact that the system is so isolated, should be an indication that it is not likely located in an area that has "good" quality power and by design would have to have a very dedicated attacker willing to travel to this location to perform an attack using this device.

It would likely be more efficient to do this with our laptop devices which so often go missing or are stolen. The isolated device (as well as many other corporate devices) would be better served by a BIOS Administrator Password that prevents the modification of the drive/boot options and interface options.

Preventing modification of the drive/boot options, prevents the alteration of the system from booting from anything other than the configured bootable device, most often the system's internal HDD. The reason we want to implement this is because there are many bootable tools available of the Internet that are designed to collect data, collect or alter passwords or other information from the system's internal HDD.

These applications are often be configured to run from external bootable USB sticks or CD/DVD drives. There are also many malware applications which can also be spread through these same

bootable methods. So, even though the Administrator that configured and installed the system took the time and care to remove the built in CD/DVD ROM or floppy disk, they neglected to lock the BIOS allowing the attacker to alter the boot sequence to an externally connected device.

These bootable USB drives can also be large enough, and the bootable tools running on it, small enough that much of the system's internal HDD can be copied onto the USB stick itself. Allowing the attacker to copy much of the system's information and leave before anyone can do anything about it.

Like preventing the boot order, preventing alteration of the system's interfaces also assists in providing increase security. While the Administrator can remove, the floppy and CD/DVD ROM devices, they cannot remove interfaces that are built onto the motherboard itself. So, while the Administrator is configuring the system to only boot from the internal HDD, they need to disable any unnecessary onboard interface as well.

This includes the USB, serial, parallel, mouse, keyboard, modem, Bluetooth or other interface that is not used by the system for normal operation. It also included the internal controllers that are not used as well, this includes VGA controllers, SATA or IDE controllers, redundant network controllers, etc.

The purpose here is to make it as difficult and time consuming as you can to prevent someone from gaining physical access to and altering the hardware configuration. While there are always ways to eventually reset the BIOS of any system, even those with redundant backup BIOS solutions, the difficulty of doing so often outweighs the value of the information that can be harvested from a single system itself.

Moving onto the operating systems. As we stated earlier, most organizations have a large Microsoft implementation with a few additional "other" operating systems included. For most Administrators, this means that the infrastructure and its security are designed around the Microsoft Active Directory solutions, including global, group and individual Active Directory security policies.

For its part Microsoft does generally release very good documentation and best practices guides for configuring these policies. However, the Administrator must fight the urge to blindly follow these documents and instead take the time to ensure that their implementation is designed around their specific implementation, with their specific software and hardware solutions carefully considered.

The reason that these documents are called "guides" is because they are meant to provide example scenarios on how or what to do based on certain criteria. Best practices guides change regularly, especially with each new major release of desktop and server operating system.

For example, as late as 2010 it was still considered best practice to prevent certain functions such as print and file sharing on MS Windows operation systems. However, with the advent of Windows 7 and the recent release of Windows 8, these features are heavily relied on by the operating system to perform some of the systems functions including accessing server file shares.

What was once used to only allow PC to PC ad-hoc file sharing or allow sharing a printer directly attached to one PC on the network, is now a critical part of the PC to server communications process and the system simply does not work properly without it.

Despite changes such as the one described above, there are still many common hardening tasks that should be performed. This is by no means a comprehensive list, but a standard set of practices that have withstood the test of time to date. These include:

- Passwords
  - Passwords must be regularly changed by both the users and the Administrators as per a corporate defined password change policy.
  - Where possible, passwords must contain enough complexity to prevent easy cracking. This means that they should be:
    - a minimum of 8 characters,
    - contain at least one uppercase Alpha character
    - contain at least one lowercase Alpha character
    - contain at least one numeric character
    - contain at least one symbol
  - Where possible, passwords should be stored in encrypted or non-reversible format.
- Anti-malware and Firewall
  - All corporate owned systems must have anti-malware and firewall software installed.
  - The anti-malware software should be configured to update on a daily basis by either connecting to the corporate update server or directly to the anti-malware vendor site.
  - The firewall application should be configured to verify and update its firewall rule base daily though a primary configuration server or through global/group policies, where applicable.
  - Users, by global policy, must be prevented from disabling the anti-malware and firewall applications.
- Software and/or Operating System Updates (including Service Packs)
  - All application and operating system updates, patches and service packs must be regularly scheduled during a standard maintenance window.
    - Any system that is not updated during the maintenance window period must, the next time they connect to the corporate infrastructure, be segregated to an isolation network segment until all updates have been completed.
  - All systems must be scanned regularly for software update compliance, any system to be found deficient;
    - must be denied access to or not be given full access to other corporate system resources or
    - must be segregated to an isolation network segment until software and/or operating system updates have been performed.
  - Users must be prevented from installing non-corporate or non-approved software onto any corporate device they use.

    - Users must be prevented from modifying software setting other than that required by the normal performance of their job.
- System Processes and Services
    - Administrators must ensure that there are no errant or unnecessary system processes or services running on the system.
        - All processes and services required for normal system operation should be set to "Automatic" startup.
        - All processes and services that are required for normal application operation should be set to "Manual" startup where applicable.
        - All other processes and services that are not required should be set to "Disabled".
- System Communications
    - Administrators must ensure that there are no unnecessary TCP/UDP ports open on the system, other than those required for normal operation of the system and its applications.
    - Administrators must ensure that non-required communications interfaces, including network ports, Wi-Fi cards, Bluetooth cards, and infrared ports are disabled unless required for normal performance of the user's work duties.
- User Login Access
    - All users must be assigned their own unique User ID and Password combination for logging into any system.
    - All Administrators must be assigned two accounts as follows:
        - Their own unique User ID and Password combination as per any standard user access, for the performance of non-administrative work related duties.
        - Their own unique Administrative ID and Password combination for the performance of administrative work related duties.
        - Administrators must ensure that the default Administrative account has been renamed and that the account's password is substantially stronger than the password policy requires.
        - Administrators must ensure that User ID and Password combinations are not locally cached on any non-mobile system.
- Remote Administration and Administrative Workstations
    - Administrators must ensure that remote administration capabilities are disabled or locked down on any system which it is not required for purposes of standard IT support.
    - Administrators must ensure that remote administration of system capabilities are restricted to administrative accounts only, on any system which requires this access for standard IT support.
    - Administrative workstations:
        - Can only be logged into by the Administrators with either:
            - their standard User ID and Password combination or
            - their Administrative ID and Password combination.
        - Must be locked down any time they are not being used.
        - Must have inactivity timeouts which are significantly shorter than defined in the standard inactivity timeout policy.

At the point of sounding redundant, it must be stated again that this is not a comprehensive list each organization's environment and infrastructure is different. The list of items here is only a start and even then some of the items may not be relevant in a particular environment. The organization's policies should be the biggest guide you have to help you determine what of this list should be used, what shouldn't and what needs to be added.

One final note on this area, while many of the items here may seem quite old or out of date, they are not. In fact, if you look through various security guides and best practices documents and manuals written for Microsoft Windows and Linux in the last decade, these items are repeatedly covered. While the terminology and acronyms may change, they are referring to the same items over and over again.

Application Integrity

Much of today's off-the-shelf software provides the Administrator with the ability to change setting with relation to security. Sure, applications like Adobe Acrobat Professional and MS Office allow you to password protect files and restrict printing rights, however there are many applications available on the market to break these protections and open the file up to anyone.

More recently, many applications, like MS Office 2013 are written to be partially cloud-based. Allowing the user to access their documents from anywhere in the world without having to keep them on personal hard drives or USB sticks. Again, the Administrator is afforded little ability to enforce security here.

Where the Administrator is most able to make application security count is with server-based multi-user applications such as MS Exchange, SQL (or other database) and Web-based application servers, where the user must authenticate to the application. In fact, if you were to perform an "application integrity best practices" or "application security best practices" through any Internet search engine, you will find links for securing email-based, web-based or database-based application security practices more than anything else.

As with all the other areas of our Defense-in-Depth model, much of these security practices truly depends the organization's environment and infrastructure design and the security protections implemented up to this point. With that in mind, we will look at application integrity in much the same way we looked at system integrity, with a general guideline of best practices, that must be added to or subtracted from based on your organization's particular environment.

While we could be specific about the type of email, web or database server here we will not be. The truth is that it doesn't really matter who produces the software, the security practices that are followed should be consistent. Even if the current generation of these applications is not affected by a security weakness that some of its competitors face at this time, doesn't mean it won't be in the next release or three releases from now.

Checking and re-checking is the only way to make sure that a security hole hasn't been introduced. That said, let's look at a standard guideline for application security in which the users must authenticate to the application:

- General Application Security
    - All users where possible, dependent on the application's design, must authenticate with a unique User ID and Password combination.
        - Where possible, application authentication should be integrated with domain authentication.
        - If it is not possible to integrate the application's authentication with domain authentication, the assigned User ID and Password combination must be unique for each user and the password meet or exceed the organization's password complexity policy.
    - All access rights assigned to a user or group of users must be assigned with minimal privilege and permissions.
    - Where possible, all application vendor security recommendations must be implemented.
    - All servers running the application must be properly hardened as per the system integrity policies for the organization.
    - User accounts that are no longer in use must be disabled and/or otherwise prevented from authenticating.
        - Where the user account cannot be removed for data integrity reasons, the account must have all application privileges and permissions removed prior to disabling or locking down the account.
        - Where the application is capable, inactivity timeouts must be implemented to disconnect idle user connections.
            - Inactivity timeouts must be in compliance with other system inactivity timeout policies defined by the organization.
    - All application servers will log ALL failed login attempts to an appropriate log reporting server.
- Email Application Security
    - All email received by email server must filter incoming mail to prevent spam email from being forwarded to internal, corporate users.
    - All email and all email attachments, internally or externally generated must be scanned for viral infections.
        - Email and email attachments which cannot be scanned for viruses must be blocked from being sent and quarantined for review by the Administrator who is to be alerted to the email's status by automated notification systems.
        - Only the Administrator can remove permanently approve and release an email from its quarantine status or permanently delete or otherwise remove an email from the server.
    - Only authenticated users can send and receive emails through the email server.
        - By default the server, as an end-point device for the organization, will not act as an SMTP relay for other email servers.
    - Encrypted email, except that approved by the organization for business purposes, must be blocked.

    - In the event that users are able to authenticate to the email server through a web-based interface, the strongest authentication method available to the organization, such as certificate or forms-based authentication.
- Database Application Security
    - All unused or default service accounts and guest accounts must be disabled where possible or renamed if required for proper operation of the server and/or server application.
    - All data inputs must be properly validated, subject to type and length checks and sanitized to prevent SQL injection attacks.
        - ALL failed data validation checks will be logged to an appropriate log reporting server.
    - All traffic between the database and the database front-end interface which contains non-financial and non-personal information must be performed over secure and if possible encrypted communications channels.
    - All financial and personal information that is transmitted between the database and the database front-end application must be performed over secure and encrypted communications channels.
    - All financial and personal information stored in a database's tables must be in encrypted form.
    - If the database does not require Internet access or does not have a web-based portal interface for user access, the firewall must be configured to block the appropriate database server ports.
- Web Application Security
    - If the Web Application server accesses data from a database server, all data inputs must be properly validated, subject to type and length checks and sanitized to prevent SQL injection attacks.
        - ALL failed data validation checks will be logged to an appropriate log reporting server.
    - All personal and financial information recorded by the web application will be stored in encrypted form.
    - All web applications which record data input, must use the secure HTTP protocol (HTTPS) between the client entering the data and the web supplication server itself.
    - Where possible, any web application server which is accessed only by internal corporate staff, must have integrated domain authentication for user authentication.
    - Remote administrative access to the web application server must be prevented from external access by the firewall.
        - Where possible, remote access to the web application server should be restricted to a minimal number of internal workstations that are under administrative control.

Is any of this sounding familiar? It should, since many of the items covered in this list have been covered repeatedly throughout this document.

This list works well whether or not these applications are off-the-shelf, vendor supplied or built in-house. However, building in-house applications such as these does afford the organization even more control over its application security implementation. This is especially true if, the organization has the vision needed to build security in from the beginning. Unfortunately most don't.

There are many training courses and books written regarding how to properly design application solutions with security built in from the beginning, which is important for organizations that do develop their own applications. However, even if the organization does not build its own applications, the Administrators who do attend these courses find that they gain a wealth of knowledge about what questions to ask when they are evaluating these types of applications for their organization and vendors supplying them.

Data Integrity

When many people think about data integrity the first thing that they think of is data backups. While this is one aspect of data integrity, these backups only provide the ability to recover your data if it is lost, accidentally or intentionally, through deletion and over writing.

There are typically three common problems with data backups that are often overlooked by Administrators that basically render data backups useless in protecting data. These are:

- many organizations do not even test their ability to recover their data from the backups,
- often the backup rotation cycle is so short that recovery of files is not possible,
- backups do not protect against data theft through copying of files.

These are three very important failings on the part of the Administrator(s) who are tasked with managing and protecting the organization's data. First, the testing of data recovery from backups is something that is covered repeatedly in best practice guidelines from every backup product manufacturer and every disaster recovery/business continuity guideline written in the last 20 years.

It is even covered under most IT Security Audit practices and guidelines, however for many organizations, regardless of size, it is often not performed because it is not considered important enough or the staff simply do not have time to perform the testing on a regular basis. The result; the organization needs to recover a file and when attempting to do so finds that their backups are corrupted, unreadable or not actually occurring.

Why did the Administrator not notice? Well the answer here is simple, they relied on the backup system, which wasn't working properly, to tell them there was a problem…, but it didn't…, and the Administrators are actually surprised!!!

While automated systems, like a backup system, are nice to have they cannot be just setup and ignored, with someone changing a tape or hard disk drive periodically. They must be more vigilantly monitored, reviewed and tested to ensure that they are working properly. It is a real simple rule; poor administrative backup practices will lead to eventual, irreversible data loss.

The second issue with backups is also related to poor administrative practices; the tape or drive rotation cycle. Here the irreversible data recovery is not a result of bad backups, it is the result of not retaining backups over a long enough period of time.

Depending on the organization's size, there may be a little to a lot of data written to the backups and many organizations will follow a standard backup rotation cycle for their media. There are several different rotation models to choose from, depending on how your backup solution is designed. For example, one model that many Administrators will be familiar with is the "Grand Father-Father-Son" model.

A simple example of this model (there are a couple of ways to implement it), would be to perform a full backup once a month – the "Grand Father". A weekly full backup – the "Father", is also taken and the incremental backups – the "Son", is performed daily. With this model, a minimum of one days changes to the data can be recovered and a maximum of one month, if we assume a single cycle.

To improve on this, we would need to implement this over a 12-month period, with 12 monthly "Grand Father" backups, 52 weekly "Father" backups and 365 daily "Son" backups. This would give us a recovery window of a minimum of one day, and a maximum of one year. However, many organizations do not implement a full year's backup rotation and instead opt for a much shorter rotation window of three (3) to six (6) months of rotation backup.

It is this shorter rotation window that causes the data loss. This is because there are often items of data that get changed long before anyone realizes the problem. A file gets accidently over-written, a file is accidentally deleted when a user cleans-up their folders, a user accidently over-writes a project folder with another one of the same name, etc.

Most often, these files and folders which are lost are quickly found because they are often related to a user's everyday work files. Therefore, when they open a file or folder with the wrong contents, they quickly realize there is a problem. However, if the problem has occurred with archived files, or with files that are related to a long running project and are only updated on an annual or semi-annual basis, the short three or six month rotation cycle may mean that the data loss is permanent.

Most, best practices guideline recommend a minimum of a one (1) year rotation cycle for backup retention periods. However, for organizations that have long running financial obligation and reporting periods or for government reporting and record retention, even a one year backup retention period may be too short.

Finally, data backups do nothing to protect the organization from having the data simply walk out the front door of the organization, be sent to someone else through an email, or from being downloaded by someone. The backups simply help you recover data that has been lost somehow, not copied.

This all too often, seems to be forgotten by the Administrators – just because the data is still there, doesn't mean someone didn't take it. Unlike a building, a house, a car, or any other

physical object an organization or individual may place value on which would take a long time to copy and may be difficult to conceal, data takes mere minutes or seconds to copy and CAN easily be concealed in one's pocket.

To protect against this type of "data loss", the organization and more importantly the Administrators must implement other data protection methods. The methods that are implemented must be based on the data itself, i.e. how the data is classified, who has access to it, how it is validated and whether or not it needs to be encrypted.

The biggest denominator of data security is of course, how the data is classified. In many cases, the classification of data within the organization drives the remaining factors of who is allowed to access it, etc., and there are various data classification schemes which can be implemented. Here we will discuss a simple data classification example, where data can be classified as:

- Restricted
    - Data which is considered confidential to the organization and where the unauthorized disclosure or alteration of the data can cause significant impact to the organization's business processes.
- Private
    - Data which is not confidential to the organization, however it has also not been defined as being available for release to the public. In other words, it is for internal day-to-day operational use of the business.
- Public
    - Data which has been explicitly released to the public, ideally through the organizations Corporate Communications or Public Relations department. It can be viewed and used by anyone both internal and external to the organization.

While this classification list can be viewed overly simplified, for small organizations it really isn't. "Restricted" data, such as patent documents, customer information and employee personal information, is data that is sensitive enough that releasing it to the public domain could seriously hinder business operations or cause the organization its competitive advantage.

While "Private" data is general information which will not hinder the organization's operations or affect its competitive advantage. This data can include; internal company policy, maintenance schedules, process data, etc. While the organization does not generally want this data in the public domain, it doesn't hurt the organization if it does get out.

"Public" data has been formally or explicitly released by the company to the public domain. As stated earlier, this data should always be released by an organization's Corporate Communications or Public Relations department. If the data has not been released for advertising or marketing purposes, then there should be a policy in place for external requests for information. In this way, the organization can ensure that any information being released has gone through the appropriate channels and does not contain information the organization does not want released.

As stated earlier, there are many different data classification schemes, tailored for many different types of organizations including; businesses, government, military, etc. Our example is simple yet effective for small organizations but only contains three levels of classification, other schemes may contain four and even five levels.

Once the organization has classified the data the next step is determining who has access to it. Access of electronic data is usually controlled through domain rights and policies, and different organizational departments may have different levels of access to the same data. For example, staff in the research and development department may have access to all of the organization's research data or only the portions that are related to the projects they work on.

Likewise, the HR Director and the organization's Executive Board may have access to all personnel files, including those of the Executive Board members, while lower level HR staff may only have access to general personnel records and files. The right to access information or not, must be carefully considered when setting up where and even how the data is stored and managed.

It must also be taken into consideration when a staff member changes position. This was alluded to earlier when we discussed removing rights that a staff member, promoted to a new role, no longer needs. Too much access can lead to accidental violations of the data, while too little access can hinder someone's ability to perform their job effectively.

Taken into consideration, the department that generated the data should be considered the "Owners" of the data. At which point, the senior staff in these departments, i.e. the managers and directors, should be the ones that decide who should and who should not be able to access the data. Where boundaries are crossed, for example a research manager who needs access to the project's financial data for purposes of managing the project's budget, should be granted access to the financial system and the project records, but would not need access to the all of the organization's financial data.

It is this line, of a user may have access to some but not all information that becomes hard to manage in many organizations, since the organization would have needed to consider this capability in its system design or selection criteria before implementing the system. Many organizations did not have the foresight to do think of this situation, prior to selecting their systems, especially those that have grown from small to large organizations while still using the same financial systems.

To solve this problem there are really two solutions, change financial application or assign someone from the finance department to be the liaison for the project's manager. This person would be a staff member of the finance department with access to the finance system and would provide the manager with budgetary reports as needed so that they can manage the projects spending.

While both solution's cost the organization money, either for the software replacement or for the extra body that might be needed to provide financial management reporting for the project to the project's manager, it is often the providing of the person that usually wins out. This is, for many

organizations, just simple short-term versus long-term math. Short-term, a single person can provide reporting for multiple projects to multiple project managers for a fraction of the cost of replacing the old financial systems with a new corporate ERP solution. Long-term, the organization will likely agree to an ERP replacement, when the limitations of the existing system simply become too costly to maintain.

Again, it is all about access to the data, how it is provided, the level of exposure the organization is will to accept and the level of trust it has in its staff. The project manager could have simply been given the access that they needed, even if it meant that the system gave them access to all project information. They could also be provided with a liaison who would provide them only with the information they needed.

It is the "trust" issue that is usually the reason that a data classification scheme was developed in the first place. Along with the data classification scheme are policies that determine who is allowed access to what, based often on job or position description, and how they can request access to other information in the organization. Oddly enough, the last level of the Defense-in-Depth model takes us directly back to the first level.

Once the access issues are sorted out, the organization is not finished with protecting its data. All, it has done at this point is define the types of data it has and who can access it, it hasn't actually provided protection to prevent bad data or bad alterations of the data. It also hasn't protected the data from being accessed by those who have managed to gain access and copy it somehow.

This is where data validation comes in. Data validation actually has several parts to it, the biggest part of which is understanding the data itself. Is the data moving or being modified on a regular basis? If so by who? Is it staying still, i.e. it has been archived? Is it available to be copied, downloaded, uploaded, etc.? Are data consistency checks being performed for data entry fields?

All of these questions need to be answered, more importantly, if the data is not being monitored and validated, how does the organization know who made the changes, when they were made and if they are correct. The answer is, it doesn't.

Data monitoring can provide the answer to the first two parts of the question above. By using data monitoring tools the organization can determine who made changes and when they were made because the information is recorded in log transaction files. However, this does not provide the organization with validation of the data.

To validate the data, the applications that access the data must have validation routines that verify that the data entered into the system is correct and passes validation routines to ensure that it is not erroneous or contains illegal values. This is not an easy task for many data items such as documents of similar type files in which the user can often write whatever they want, however for most organizations, this is not the type of data that causes them concern.

The data that they really want to protect is the data recorded in their financial systems, their sales and purchasing systems and the associated databases. This is where the company makes and

spends its money, where customer information is recorded and where loss or corruption of this information can cause them significant business disruptions.

To prevent this, access to the databases must be restricted only to the application that uses the database, and likewise the users must be forced to login to these systems properly. Preventing external connections to the databases, forcing transactional logging within the application or database itself and ensuring there are routines for validating the data entry will remove most of the risk associate with data validation.

I.e. if the only way to change the data is through the database application and if the database application is properly designed so that invalid data cannot be entered, then there should be very little bad data on the system. This doesn't mean that there won't be any bad data, spelling mistakes and transposition of numbers may still make their way into the records, but the bad data will be greatly reduced.

While the risk of having invalid or bad data is reduced, it really doesn't protect against someone from copying this valid data for use outside of the organization. This is what the hackers really came for, it contains your customer information; their addresses, phone numbers, credit card numbers or account numbers. All of which the hacker can use to forge fake orders, or sell to others who will use the information for identity theft or fraud.

This is where data encryption comes into play. For all data within the organization which contains personal information about a customer, be it an individual or another organization encryption must be used. This even includes personal information on the organization's staff in its HR records. It must be encrypted.

This doesn't mean that all of the data in a database needs to be encrypted, but it does mean that any table or field in a table that contains personal or financial information regarding an individual or external organization must be encrypted. The purpose is to prevent the use of the information for criminal or malicious purposes.

It basically renders the information copied from the tables as valueless since the individual copying the data cannot decrypt the data without the application and encryption key, they cannot gain any information that can be used to commit identity theft or fraud. They also cannot sell the data since it contains no information that a buyer can use to commit the same crimes.

The problem is that not enough organizations commit to this level of security. They complain that this slows down the processing of the data and committing the transactions, it also increases processing and memory requirements of the servers. While it does take time to perform the encryption and decryption, but not enough to prevent the functional use of the system.

As for the increase in processing and memory requirements on the servers, this too is at minimal cost when compared to the expense of reporting a data breach and notify customers of the potential loss of their personal and credit card information. The additional hardware expense also pales in comparison to lost business revenues from reduced of customer confidence, negative publicity for the organization and potential fines and lawsuits that the organization may face.

The Sony example provided earlier, provides an excellent example of the issues facing an organization in today's business environment where the government has made it a legal requirement to report a breach of their systems and the potential loss of data. The solution, encryption, has been around long enough and in fact is already designed into many systems, yet many organizations fail to use it or even to simply turn it on.

This is unacceptable on the part of the organization and the Administrators. In most cases, if the encryption process is turned on from the very beginning most users do not even notice if it takes a half second more to refresh the data in the application window. It becomes regular behavior of the system. They only notice when it is turned on after the system has been used for some period of time and all of a sudden, "Gee, the Administrator turned on record encryption and now the system is sooooo slow it's almost useless."

As the Administrator, ignore these comments and move on. Six months from when you turn it on, it will again be normal behavior for the system and most users will again not notice a difference. Truth is, it doesn't matter when, as the Administrator, you turn on encryption. It just matters that you turn it on.

References

White Papers and Web References

United States Government: Department of Homeland Security: Recommended Practice: Improving Industrial Control Systems Cybersecurity with Defense-In-Depth Strategies, (US Department of Homeland Security, October 2009), http://ics-cert.us-cert.gov/sites/default/files/recommended_practices/Defense_in_Depth_Oct09.pdf

Australian Government, Department of Defense: Information Security Manual (Australia Department of Defense, August 2011), http://www.dsd.gov.au/publications/Information_Security_Manual_2010.pdf

InstantSecurityPolicy.com: The IT Security Policy Guide: Why you need one, what it should cover, and how to implement it (InstantSecurityPolicy.com, 2010), http://www.instantsecuritypolicy.com/Introduction_To_Security_policies.pdf

RSA Security Inc,, A Guide To Security Policy: A Primer for Developing Effective Policy, (Pamela Bury, 2000), https://www.cccure.org/Documents/Security_Policy/pol_guidefinal.pdf

Global Information Security Certification (GIAC): The Basics of an IT Security Policy, (Jack G. Albright, March 25, 2002), http://www.giac.org/paper/gsec/1863/basics-security-policy/103278

System Administration, Networking and Security Institute (SANS): Security Policy: What it is and Why – The Basics, (Joel S. Bowden, February 18, 2003), http://www.sans.org/reading-room/whitepapers/policyissues/security-policy-basics-488?show=security-policy-basics-488&cat=policyissues

Symantec: Conducting a Security Audit: An Introductory Overview page, (Bill Hayes, May 26, 2003), http://www.symantec.com/connect/articles/conducting-security-audit-introductory-overview

ITSecurity.com: The Essential Guide to Security Audits page , (John Edwards, April 29, 2008), http://www.itsecurity.com/features/security-audit-essentials-042908/

Government of Canada: Treasury Board of Canada Secretariat: Operational Security Standard on Physical Security page, (Treasury Board of Canada Secretariat, posting date unknown), http://www.tbs-sct.gc.ca/pol/doc-eng.aspx?id=12329&section=text

Royal Canadian Mounted Police: G1-026 Guide to the Application of Physical Security Zones, (Royal Canadian Mounted Police, September 2005), http://www.rcmp-grc.gc.ca/physec-secmat/pubs/g1-026-eng.htm

Global Information Security Certification (GIAC): Firewalls, Perimeter Protection and VPN's Version 1.8, (Renett Chan, February 2003), http://www.giac.org/paper/gcfw/378/firewalls-perimeter-protection-vpns/104474

Cisco Systems Inc.: Deploying Firewalls Throughout Your Organization page, (Cisco Systems Inc., posting date unknown), http://www.cisco.com/en/US/prod/collateral/vpndevc/ps5708/ps5710/ps1018/prod_white_paper0900aecd8057f042.html

National Institute of Standards and Technology (NIST), Special Publication 800-41 Revision 1: Guidelines on Firewalls and Firewall Policy, (Karen Scarfone and Paul Hoffman, September 2009), http://csrc.nist.gov/publications/nistpubs/800-41-Rev1/sp800-41-rev1.pdf

Cisco Systems Inc.: How Virtual Private Networks Work page, (Cisco Systems Inc., October 13, 2008), http://www.cisco.com/en/US/tech/tk583/tk372/technologies_tech_note09186a0080094865.shtml

Gizmodo.com: VPNs: What they Do, How They Work, and Why You're Dumb for Not Using One page, (Andrew Tarantola, March 26, 2013), http://gizmodo.com/5990192/vpns-what-they-do-how-they-work-and-why-youre-dumb-for-not-using-one

Cisco Systems Inc.: Understanding and Configuring VLANs page, (Cisco Systems Inc., April 20, 2007), http://www.cisco.com/en/US/docs/switches/lan/catalyst4500/12.2/25ew/configuration/guide/vlans.html

Schweitzer Engineering Laboratories Inc.: Security Through VLAN Segmentation: Isolating and Securing Critical Assets Without Loss of Usability, (Garrett Leischner and Cody Tews, March 2007), http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.125.5869&rep=rep1&type=pdf

System Administration, Networking and Security Institute (SANS): Virtual LAN Security: weaknesses and countermeasures, (Steve A. Rouiller, publish date unknown), http://www.sans.org/reading-room/whitepapers/networkdevs/virtual-lan-security-weaknesses-countermeasures-1090?show=virtual-lan-security-weaknesses-countermeasures-1090&cat=networkdevs

National Institute of Standards and Technology (NIST), Special Publication 800-123: Guide to General Server Security, (Karen Scarfone, Wayne Jansen and Miles Tracy, July 2008), http://csrc.nist.gov/publications/nistpubs/800-123/SP800-123.pdf

United States National Security Agency: Guide to the Secure Configuration of Red Hat Enterprise Linux 5, Revision 4.1, (United States National Security Agency, February 28, 2011), http://www.nsa.gov/ia/_files/os/redhat/rhel5-guide-i731.pdf

United States National Security Agency: Central Security Service: Operating Systems page, (United States National Security Agency, Last Modified August 14, 2013), http://www.nsa.gov/ia/mitigation_guidance/security_configuration_guides/operating_systems.shtml

Microsoft Inc.: Application Security Best Practices at Microsoft, (Microsoft Inc., January 2003), http://www.google.ca/url?sa=t&rct=j&q=&esrc=s&frm=1&source=web&cd=2&cad=rja&ved=0CD4QFjAB&url=http%3A%2F%2Fdownload.microsoft.com%2Fdownload%2F0%2Fd%2F3%2F0d30736a-a537-480c-bfce-5c884a2fff6c%2FApplicationSecurity.ppt&ei=vUBFUrfuJKboiAKfrIGQCw&usg=AFQjCNHs1b-t9oU_FaPu1zmTaohwpa0s0g

Cenzic Inc.: Best Practices for Application Security: 10 Ways to Protect Your Cloud, Mobile and web Apps from Hacker Attacks, (Cenzic Inc., February 21, 2012), http://www.cenzic.com/downloads/app-security-papers/Cenzic-executive-brief-Top-10-App-Sec-Best-Practices.pdf

Oracle: Oracle Application Server Security Guide 10g Release 2 (10.1.2) page, (Oracle, 2005), http://docs.oracle.com/cd/B14099_19/core.1012/b13999/securitybestprac.htm

National Institute of Standards and Technology (NIST), FIPS PUB 199: Standards for Security Categorization of Federal Information and Information Systems, (NIST, February 2004), http://csrc.nist.gov/publications/fips/fips199/FIPS-PUB-199-final.pdf

Online Trust Alliance (OTA): 2012 Data Protection and Breach Readiness Guide, (Online Trust Alliance, January 24, 2012), http://www.otalliance.org/resources/incident/2012DataBreachGuide.pdf

Closing Comments

Since starting in the IT Industry in Edmonton in 1987, I have held the titles of Senior Network Analyst, Network Administrator, Server Administrator, Security Administrator, Infrastructure Designer/Planner, Application Developer/Programmer, Senior Systems Architect, DBA, Instructor, IT Supervisor and IT Manager. With all of this, I have seen the advancements in technology and its cyclical patterns as well.

I have seen, thin client and remote desktop become the "big thing", go back to desktops, and then return with again with virtual desktop solutions. I have also seen it evolve from the embedded OS device to the current "zero client" solutions using the PC over IP (PCoIP) protocol.

The same can be said for operating systems, with features that came, went, and came back again. These cycles will repeat themselves for the foreseeable future, with each iteration advancing on the previous.

The hot trend today is of course "Cloud-based services". William Shakespeare will have to forgive me, but "a rose by any other name…" is still a rose, and the "Internet by any other name…" is still the Internet. The fact that we have coined so many new acronyms (BAAS, SAAS, IAAS, VAAS, SDS, etc.) and been fed the "hype" of this solution in the last couple of years should not confuse the fact that we have simply moved the providing of services from a main office server room, across a traditional corporate network, to servers, physical or virtualized, sitting somewhere on the Internet.

The reality is that, the ISP or Internet infrastructure, improved to the point where the companies offering these services are able to secure high speed Internet links at a fraction of the cost that they could have done so ten years ago. To coincide with this, virtualization solutions have come into their own, meaning that they are now considered mature technologies, stable, reliable and feature rich.

As with the past, I am positive that ten years from now there will be a new "big thing" that will once again change how we define our corporate infrastructure. The organizations I have worked for in the past, have embraced the changes as they have come along and have also felt the growing pains associated with moving from an older technology to a newer one.

Even at the organization I currently work for, we have both traditional servers and cloud-based servers, traditional applications and thin client applications, virtual and non-virtual servers. However, the one thing that has remained the same is in the need to provide and overall security solution for the systems.

It doesn't matter where you put your solutions, in-house or in the Cloud. You must still go through the process of ensuring system integrity at all levels with some form of Defense-in-Depth model. Your model may change to suit your needs of your environment, however it will never go away as long as there are "hackers" who would try to access, modify or steal your data.

I have used the model presented here for many years, however as my organization has begun to move some of its systems to the Cloud, I modified it to meet the needs of those systems. For instance I do not need to put into place the Physical Integrity piece of the model, since the servers that are in the Cloud are hosted in California in a secure data center owned and operated by our Cloud provider.

While I didn't need one portion of the model, I did increase my vigilance in other areas of the model. Consider this, even though the Cloud provider has a secure location for their systems (and mine), it is also located in a major earthquake zone. With that in mind, I made the systems running in the Cloud redundant with the second server running in another Cloud-based data center on the east coast.

I also became very aware of some of the less considered risks of outsourcing to the Cloud. The primary being the fact that, while the Cloud Service Provider has gone through great extent to ensure security of their systems and yours as a result, your systems and data are still running on someone else's equipment, and they are no less a target than before.

Your servers and data are most likely several thousands of miles from you, and based on current Cloud provider designs, they (or their staff) will always have some way access to your systems as "root". After all, it is just a virtual server running on their infrastructure. If you are running virtual servers on your own infrastructure, do you not also set up tools that allow you to take over and access them if something goes wrong? How confident are you in their security practices?

In the end, it all comes down to implementing security for your systems wherever they are. You must learn to modify the Defense-in-Depth model to meet your organization's needs, its overall systems design and infrastructure. Ultimately, implementing security solutions to ensure data integrity is **YOUR** responsibility and you can never rely on someone else to do it for you.

That being said, the only way to become proficient in security practices, is to practice security. The labs provided here will help you at gain an understanding of some of the issues faced with setting up basic perimeter security monitoring (using Snort) learning to write rules and even in testing it using Metasploit.

Your will face a few trial and tribulations, as would any Security Analyst, in working through the labs. One of the biggest obstacles to overcome with security is that you can find great and terrible documentation to help you through this. The applications used; hping3, Snort and Metasploit all have some great documentation, however you will find that no matter how good the documentation is you will still have to figure some things out on your own.

The troubleshooting and resolution of the problems you will face in completing the labs will be minimal compared to those you will face throughout your career. Most of the problems you will face in the labs will be configuration related, but are relatively easy to resolve if you take the time to think them through.

One final comment, most organizations, do not approve of or allow tools like hping3 or Metasploit to be used or installed by their Security Administrators or Analysts, or if they are allowed it is only in an isolated test environment. They are valid testing tools, and to understand your own system and infrastructure weaknesses you must use some form of tool to help test and identify weaknesses that may have been overlooked.

Practice where you can practice. Isolated virtualized systems work best, never test on your organization's production systems and always remember, ***the only difference between a security testing tool and a hacking tool is whose hands it is in***.

Course Labs and Assignments

Assignment 1: Cryptography Review

1) Given the following information :

   p = 23
   q = 17
   e = (pick the 14th relative prime number)
   C = G-O-L-D

   Using the RSA formula of $C = P^e \ MOD \ (n)$, answer the questions and encrypt the message.

   a. What is the value of e ?
   b. What is the value of (n) ?
   c. What is the value of $\varphi(n)$?
   d. What is the encrypted message (numbers only)?

2) Given the following information

   p = 5
   q = 7
   e = 17
   C = 33-14-6-23

   Using the RSA formula of $P = C^d \ MOD \ (n)$, answer the questions and decrypt the message.

   a. What is the value of (n) ?
   b. What is the value of $\varphi(n)$?
   c. What is the value of d?
   d. What is the decrypted message?

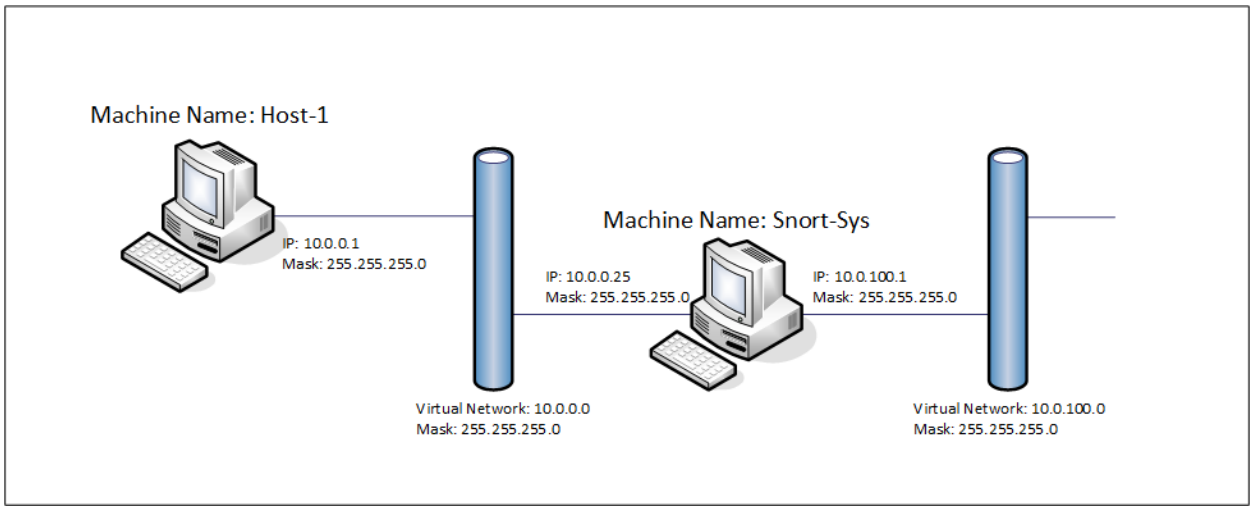
3) Describe how the 3DES encryption process works.

4) In 3DES, what function or purpose does decrypting with the wrong key perform?

5) Describe how the AES encryption process works.

Lab 1: Intrusion Detection – Writing Rules

Using VMWare Player or VMWare Workstation create the virtual machines depicted in the diagram.



Important Notes:

- Use Linux for all workstations in the Lab, the Linux OS preference is yours however be sure that you **do not** install the SELinux kernel version. The lab was tested using CentOS 6.
- Ensure that you have familiarized yourself with the latest versions of the Snort Setup Guide for your chosen Linux OS and Snort User's Manual, including the sections related to configuring and writing Snort rules. Both are available at the Snort website: http://www.snort.org/docs
- Ensure that you have registered the Snort website so that you may download the latest version of Snort Rules for Registered Users.

Host-1 Setup

- Host-1 is a basic Linux setup with the hping3 installed.
- hping3 can be downloaded from http://www.hping.org/
- The hping3 scripts are located in the Lab-files.zip file and may also be reviewed in Appendix A.
- The lab files should be copied to a **/hping_scripts** folder on the workstation if you place it in a different location you must edit the script files to change the location path.

Snort-Sys Setup

- Snort-Sys is a basic Linux setup with Snort installed as per the Snort Installation Guide for your system or using a package installer.
- The latest Snort Ruleset should be downloaded and installed.
- ***snort.conf*** must be edited to disable all Snort rules files except the new rules that will created in the ***experimental.rules*** file.

Once the virtual workstations are setup based on the lab diagram above, verify connectivity between the two workstations using ping. Once connectivity is verified, you can proceed with the lab.

You can also verify that Snort is running properly by running the attack-demo hping3 script on Host-1 and run Snort using the "-i 2 –dev" options to ensure you are seeing the icmp attack-demo ping packet text.

Writing Snort Rules

Using Snort, you are required to write rules that alert and log on the scenario attacks described on the next page. The new rules are to be written in the "***experimental.rules***" file and they must disable all other Snort rules to ensure that only the new rules are being alerted on and logged. This will require editing of both the "***snort.conf***" file and the "***experimental.rules***" file.

Note: Each of the different attacks requires a different alerting and logging message. This message is described under the attack descriptions. The student is to replace the "***student's name***" portion of the message with their own name to uniquely identify the alert as having been generated by their rules.

Attack #1:
The first attack is an unsolicited ICMP Echo Reply attack which is an attempt to start a botnet application on infected hosts. The data in the Echo Reply message is the phrase "Start-now".
The Snort rule must capture and alert on these Echo Reply message with the message "Attempted Botnet application launch – ***student's name***".

This attack scenario uses the ***attack-01*** and ***attack-01.data*** files. Run the ***attack01*** script using hping3 on Host-1when you are ready to test your new Snort rule.

Attack #2:
The second attack is an attempt to connect to a PC infected with the "Len Wins" Trojan. This attack is always launched against port 135 on the infected host and originates from port 445 on the attacking host. The packet that initiates the attack is always a TCP SYN packet with "!DRAGON!" in the data field. The Snort rule must capture and alert on these TCP packets with the message "Attempt to launch the DRAGON Wins Trojan – ***student's name***".

This attack scenario uses the ***attack-02*** and ***attack-02.data*** files. Run the ***attack02*** script using hping3 on Host-1 when you are ready to test your new Snort rule.

Attack #3:
The third and final attack is another Trojan attack called the "1-2-3-4" attack and it is launched through UDP. This attack can be launched against UDP ports 11111, 22222, 33333 or 44444.

The Snort rule must capture and alert on this UDP attack with the message "Attempt to launch the 1-2-3-4 Trojan – ***student's name***".

This attack scenario uses the ***attack-03a***, ***attack-03b***, ***attack-03c***, ***attack-03d*** and ***attack-03.data*** files. Run the ***attack03a***, ***attack03b***, ***attack03c*** and ***attack03d*** scripts using hping3 on Host-1 when you are ready to test your new Snort rule.

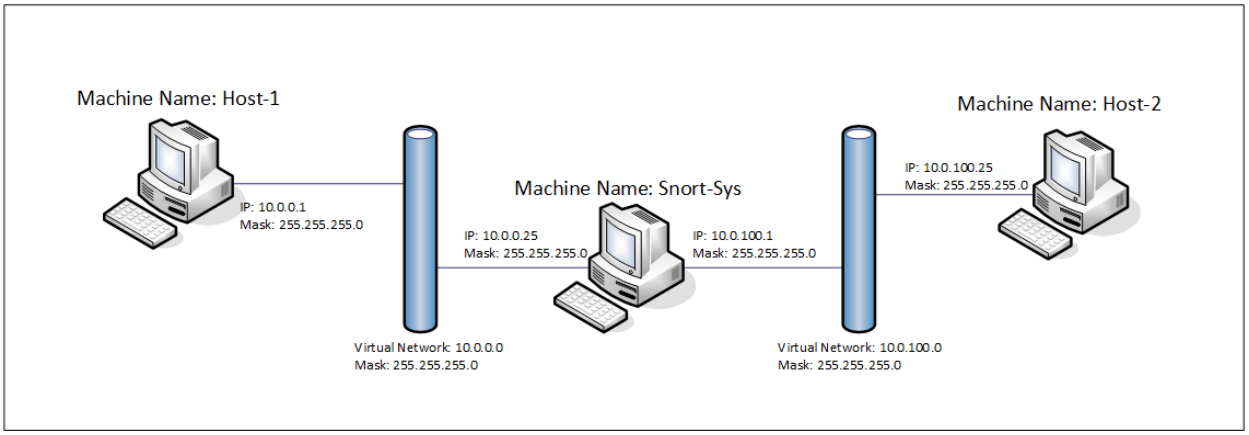
Lab Report Requirements

The following documentation is to be submitted for completion of this lab.

- An electronic copy of the ***experimental.rules*** file.
- An electronic copy of the ***snort.conf*** file.
- An electronic sample of each of the captured packets, alert messages and logged information with the individual message clearly highlighted.

Lab 2: Penetration Test

This lab is an advancement on Lab #1 with an additional virtual computer created as seen in the following diagram.



Important Notes:

- Read the entire lab before attempting to perform its steps. This will save you re-work later.
- Review the documentation on Metasploit, much of which can be found at https://community.rapid7.com/docs/DOC-2227

To complete this lab, the student's must successfully complete the following modifications:

Host-1 Setup

- Host-1 setup remains unchanged

Snort-Sys Setup

- Snort-Sys must be converted to an IPS.
  - There are numerous documents on the web which discuss how to do this, research them and follow their instructions.

Host-2 Setup

- Host-2 is a standard Linux install similar to Host-1 but with Metasploit installed.
- Metasploit can be found at the following website: http://www.metasploit.com/

Performing a Penetration Test

Once the lab setup is configured ensure that Host-1 can ping Host-2 and vise-versa. Again, you can use the Snort "-i 2 –dev" options to verify that the packets are going through the Snort-Sys computer. Ensure that the Snort rules are still disabled and as there are no firewalls in this lab, if the configurations are correct, Host-1 and Host-2 should be able to ping each other.

- **Part 1**
  - Without turning on the Snort rules, you must successfully exploit the victim computer (Host-1) using one of Metasploit's many remote exploits. Once you have access to the victim computer create a user account on the victim computer with root access. Ensure that you can RDP to the victim computer.

- **Part 2**
  - Turn on the Snort rule, setting Snort into prevention mode if your configuration does not do so by default.
  - Repeat the attack of Part 1 creating a new account with root access, analyzing any alerts and logs that are generated by Snort.
    - Was the attack successful?
      - If it was successful, what did Snort alert on and what did it not detect to stop the attack?
      - If the attack was not successful, what is Snort alerting on and what did it trigger on to stop the attack?

- **Part 3**
  - Turn the Snort rules off again.
  - Based on your analysis of Snort's alerts and log files, write a rule or rules in the ***experimental.rules*** file to prevent the attack.
  - Repeat the attack, again trying to create a new account with root access.

Lab Report Requirements

- **Part 1**
  - Identify the Metasploit remote exploit you used and summarize how the attack works and why it was successful.
  - Provide a screenshot of the victim computers RDP desktop.

- **Part 2**
  - Provide an electronic copy of the alerts and log reported by Snort.
  - Provide an explanation of why Snort was successful or not successful in stopping the attack.

- **Part 3**
  - Provide an electronic copy of the alerts and logs reported by Snort and a copy of your experimental.rules file.
  - Provide an explanation of what your rule does to identify and block the attack, be specific in identifying the triggers you used to identify the attack and why it made sense to use those triggers.

Appendix A: hping3 Scripts

Demo Attack

Script File (attack-demo file)

```
#!/bin/bash
#
#
# This file is being used to perform a standard ping against
# the Snort-Sys computer of Lab #1. The following options are used
#      1) -1         indicates the protocol is ICMP.
#      2) -d 100     indicates the data for the packet is 100
#                    bytes in length.
#      3) -E <path>  indicates the path to the file that contains
#                    the data for this packet.
#
#
# The cat commands at the beginning prints the contents of this
# file on the screen to visually ensure the correct file was
# launched.
#                                      --Leonard Rogers--
#
clear
#
cat /hping_scripts/attack-demo
#
#
cat /hping_scripts/attack-demo.data
#
hping 10.0.0.25 -1 -d 162 -E /hping_scripts/attack-demo.data
#
#
#
#
```

Data File (attack-demo.data file)

```
#
#
#
This is a standard PING packet created by Leonard Rogers for testing the Snort Intrusion
Detection System!!!
#
#
#
```

Attack #1

Script File (attack-01 file)

```
#!/bin/bash
#
#
# This file is being used to create an ICMP Echo Reply message
# against the Snort-Sys computer of Lab #1.
#
# The following options are used
#      1) -1         indicates the protocol is ICMP.
#      2) -d 100     indicates the data for the packet is 100
#                    bytes in length.
#      3) -E <path>  indicates the path to the file that contains
#                    the data for this packet.
#      4) -i 2       indicates the interval (in seconds) that the
#                    hping application waits before sending out
#                    the next packet.
#      5) --rand     indicates that the packets are to have a
#                    random source (or destination) IPv4 Address.
#
#
# The cat commands at the beginning prints the contents of this
# file  and the data file on the screen to visually ensure the
# correct file was launched.
#
#                                      --Leonard Rogers--
#
#
cat /hping_scripts/attack-01
#
#
cat /hping_scripts/attack-01.data
#
hping 10.0.0.25 -1 --icmptype 0 --icmpcode 0 -d 10 -E  hping_scripts/attack-01.data -i eth0 --
rand-source
```

Data File (attack-01.data file)

```
#
#
#
Start-now
#
#
#
```

Attack #2

Script File (attack-02 file)

```
#!/bin/bash
#
#
# This file is being used to create a malicious TCP SYN packet
# against the Snort-Sys computer of Lab #1.
#
# The following options are used
#      1) -s         indicates the TCP Source Port to be used.
#      2) --keep     indicates the TCP Source Port is not to
#                    change with each new packet.
#      3) -p         indicates the TCP Destination Port to be
#                    used.
#      4) -S         indicates that the SYN Flag is to be set in
#                    each packet that is sent out.
#      5) -i 2       indicates the interval (in seconds) that the
#                    hping application waits before sending out
#                    the next packet.
#      6) --rand     indicates that the packets are to have a
#                    random source (or destination) IPv4 Address.
#
#
# The cat commands at the beginning prints the contents of this
# file  and the data file on the screen to visually ensure the
# correct file was launched.
#
#                                      --Leonard Rogers--
#
#
cat /hping_scripts/attack-02
#
#
cat /hping_scripts/attack-02.data
#
hping2 10.0.0.25 -s 445 --keep -p 135 -S -d 6 -E /hping_scripts/attack-02.data -i 2 --rand-source
```

Data File (attack-02.data file)

```
#
#
#
!DRAGON!
#
#
#
```

Attack #3

Script File (attack-03a file)

```
#!/bin/bash
#
#
# This file is used to send an unsolicited ICMP Echo Reply packet to a
# against the Snort-Sys computer of Lab #1.
# The following options are used:
#
#      1) -2               indicates the protocol is UDP.
#      2) --destport 11111 indicates the destination UDP port on the
#                          target system.
#      3) -d 45            indicates the data for these packets is 45 bytes
#                          in length.
#      6) -E <path>        indicates the path to the file that contains
#                          the data for these packets.
#      7) -i 2             indicaes the time interval, in seconds, that
#                          the hping2 application will wait before sending
#                          out the next packet.
#      8) --rand-source    forces hping2 to randomly generate source IP
#                          addresses for these packets. This prevents the
#                          target system from actually being able to
#                          respond to the attacking system.
#
#
# The cat command at the beginning prints the contents of this file on the
# screen to visually ensure the correct file was launched.
#
#                                    -Leonard Rogers-
#
#
#
cat /hping_scripts/attack-03a
#
#
#
cat /hping_scripts/attack-03.data
#
#
#
#
hping 10.0.0.25 -2 --destport 11111 -d 46 -E /hping_scripts/attack-03.data -i 2 --rand-source
#
#
#
#
```

Script File (attack-03b file)

```
#!/bin/bash
#
#
# This file is used to send an unsolicited ICMP Echo Reply packet to a
# against the Snort-Sys computer of Lab #1.
# The following options are used:
#
#      1) -2               indicates the protocol is UDP.
#      2) --destport 22222 indicates the destination UDP port on the
#                          target system.
#      3) -d 45            indicates the data for these packets is 45 bytes
#                          in length.
#      6) -E <path>        indicates the path to the file that contains
#                          the data for these packets.
#      7) -i 2             indicaes the time interval, in seconds, that
#                          the hping2 application will wait before sending
#                          out the next packet.
#      8) --rand-source    forces hping2 to randomly generate source IP
#                          addresses for these packets. This prevents the
#                          target system from actually being able to
#                          respond to the attacking system.
#
#
# The cat command at the beginning prints the contents of this file on the
# screen to visually ensure the correct file was launched.
#
#                                    -Leonard Rogers-
#
cat /hping_scripts/attack-03a
#
#
#
cat /hping_scripts/attack-03.data
#
#
#
#
hping 10.0.0.25 -2 --destport 22222 -d 46 -E /hping_scripts/attack-03.data -i 2 --rand-source
#
#
#
#
```

Script File (attack-03c file)

```
#!/bin/bash
#
#
# This file is used to send an unsolicited ICMP Echo Reply packet to a
# against the Snort-Sys computer of Lab #1.
# The following options are used:
#
#      1) -2               indicates the protocol is UDP.
#      2) --destport 33333 indicates the destination UDP port on the
#                          target system.
#      3) -d 45            indicates the data for these packets is 45 bytes
#                          in length.
#      6) -E <path>        indicates the path to the file that contains
#                          the data for these packets.
#      7) -i 2             indicaes the time interval, in seconds, that
#                          the hping2 application will wait before sending
#                          out the next packet.
#      8) --rand-source    forces hping2 to randomly generate source IP
#                          addresses for these packets. This prevents the
#                          target system from actually being able to
#                          respond to the attacking system.
#
#
# The cat command at the beginning prints the contents of this file on the
# screen to visually ensure the correct file was launched.
#
#                                    -Leonard Rogers-
#
#
cat /hping_scripts/attack-03a
#
#
#
cat /hping_scripts/attack-03.data
#
#
#
#
hping 10.0.0.25 -2 --destport 33333 -d 46 -E /hping_scripts/attack-03.data -i 2 --rand-source
#
#
#
#
```

Script File (attack-03d file)

```
#!/bin/bash
#
#
# This file is used to send an unsolicited ICMP Echo Reply packet to a
# against the Snort-Sys computer of Lab #1.
# The following options are used:
#
#      1) -2               indicates the protocol is UDP.
#      2) --destport 44444 indicates the destination UDP port on the
#                          target system.
#      3) -d 45            indicates the data for these packets is 45 bytes
#                          in length.
#      6) -E <path>        indicates the path to the file that contains
#                          the data for these packets.
#      7) -i 2             indicaes the time interval, in seconds, that
#                          the hping2 application will wait before sending
#                          out the next packet.
#      8) --rand-source    forces hping2 to randomly generate source IP
#                          addresses for these packets. This prevents the
#                          target system from actually being able to
#                          respond to the attacking system.
#
#
# The cat command at the beginning prints the contents of this file on the
# screen to visually ensure the correct file was launched.
#
#                                    -Leonard Rogers-
#
#
cat /hping_scripts/attack-03a
#
#
#
cat /hping_scripts/attack-03.data
#
#
#
#
hping 10.0.0.25 -2 --destport 44444 -d 46 -E /hping_scripts/attack-03.data -i 2 --rand-source
#
#
#
#
```

Data File (attack-03.data file)

```
#
#
#
Snort Signature Attack #3
#
#
#
```