In this report, I will examine various security threats, as well as methods through which these threats can be protected against. I will cover four threats in detail, these being;

- Denial of Service
- Password Crackers
- Trojans / Worms / Viruses
- Internal / External Threats

Denial of Service

Essentially, a denial of service attack, or DoS, is when a hacker attempts to make a system unusable by flooding the target with packets and communication requests. By doing this, the victim becomes saturated, is unable to handle the unusually high volume of traffic, and becomes unstable. There are a variety of ways in which an attacker may which to deploy a DoS against a target, each with varying outcomes. Some DoS will reduce the performance of its target, others may result in the victim coming to a complete standstill. Most modern DoS attacks are targeted at web servers. As already mentioned, there are a variety of differing types of DoS. I will examine a number of these, namely; Buffer overflow attacks, SYN attacks and DDoS.

An attack using buffer overflow is a very simple concept. Attackers simply try to flood a system with more traffic than that system's buffer allows for. They do this by using traffic that the system does not flag as unusual. When a buffer attempts to store more data than it was intended to hold, the surplus will look to other buffers for space. This causes the data in the new buffer to become corrupted and lost, before the data looks for further adjacent buffers, spilling out across the system, eventually causing widespread data corruption.

A well-known example of this type of attack occurred in 2000 when it was discovered that Microsoft's Outlook software contained a programming error within the message header mechanisms[1]. This made it possible for attackers to target victims by simply sending them an eMail containing enough data to exploit the fault in Outlook's headers. Defending against this attack was not possible through the usual means of eMail virus protection, as users simply had to receive the eMail for the attack to come into effect.

SYN attacks take advantage of the TCP three-way handshake. Attackers deploy a DoS using a SYN attack by, firstly, sending multiple SYN packets to the target. As any system would treat these incoming SYN packets, the target machine will send a SYN-ACK back to the attacker. However, the attacker will not respond to this phase of the handshake, often because the IP is spoofed, so the target will attempt to resend the SYN-ACK a number of times. If this is done repeatedly, the resource allocation of the victim becomes so considerable that performance is soon adversely affected.

This method of attacking, also known as flooding, poses various other types of threat. For example, an attacker could use such a method to flood a switch. When flood with enough packets, a switch essentially becomes a hub and sends all packets that it receives out to everyone. Switches generate a list known as a Content Addressable Memory, or CAM, Table. By referring to this database, the switch is able to identify a specific MAC address with a particular port on the network. So, when data is being transmitted for a certain computer, it is aware of which port the packets are to be transmitted to. By flooding a switch, the CAM table will be disregarded due to the limitation in resources, and it will enter *failopen* mode, essentially causing the switch to behave like a hub, and broadcast all packets to all ports. It is then possible for the attacker to easily sniff these, compromising security and privacy on a network.

I previously mentioned spoofing an IP. There are a number of methods through which an attacker can achieve this. One of the ways which I myself find most interesting is a Man in the Middle Attack, or a MITM. Essentially, what an MITM attack does is make its victims think that they are communicating directly with each other, when in reality, they are doing so through the "middle" PC. In other words, the *Good PC* and the *Switch* think that they are communicating directly, when they in fact doing so through the *Bad PC*. To achieve this, *Bad PC* will use a tool called arpsoof. arpspoof that is distributed as part of the dsniff package. Firstly, *Bad PC* will enable IP forwarding on his host. Failing to do this will in fact cause *Good PC*, the victim in this case, to lose connectivity. In the first instance, *Bad PC* will tell *Good PC* that it is the *Switch*, and in the second instance, it will tell the *Switch* that it is *Good PC*, utilising ARP replies. Once completed, all *Bad PC* has to do to is monitor the traffic as it so pleases, both *good* systems now thinking that it is the other *good* system.

DDoS, Distributed Denial of Service, attacks are also quite common, and when deployed correctly, can be detrimental to the target system. Deploying a DDoS requires a hacker to first compromise a number of other machines. Once they have done this, they then point all of these at the target system, so that the victim is attacked not from one, but from multiple systems. There have been cases where targets have fallen victim to DDoS from hundreds, even thousands, of compromised systems. DDoS attacks are the most detrimental of the DoS forms as there are a number of victims. Not only is the target system affected, but the compromised machines used in the attack have been infected as well.

There have been a number of high profile large scale DDoS throughout the past decade. In 2002, a DDoS attack almost crippled nine of the 13 servers that manage global internet traffic[2]. In 2004, Microsoft.com was attacked by the MyDoom.B virus, a variant of the of MyDoom DDoS that originated in Russia earlier that year[3]. The attack was not a success, mainly because the DDoS did not have enough compromised systems with which to attack a service as competent as Microsoft's.

So, how does one protect against DoS attacks? There are, like most threats, a number of ways to combat intrusion. As should be the case by default, routers, firewalls and all systems should be deployed and configured properly, with the latest patches and system updates frequently applied.

Then there are also more advanced methods of protection. One such method is the implementation of NBA, Network Behaviour Analysis, on a system. NBA systems are designed to detect any unusual behaviour on a network; for example, unusual traffic flow. Amongst their functionality, they boast the ability to monitor bandwidth and protocol use. This is an extremely valuable feature when trying to monitor for DoS attacks. Through the use of NBA systems, an administrator can detect any unusual traffic volumes and take appropriate action.

More complex intrusion detection systems can also be put in place on a network. Network-based intrusion detection is responsible for the monitoring of traffic between different network segments and devices. It is often one of the most crucial forms of intrusion detection and can be of huge benefit when protecting a system against DoS attacks. Network-based intrusion detection functions through the use of sensor deployment. Sensors can be deployed in two ways. They can be deployed as an inline sensor, or alternatively, as a passive sensor. Inline deployment means that all traffic on a network has to pass through the actual sensor, whereas with passive, the sensor just monitors a copy of the actual traffic that is travelling on the medium.

The best location for inline deployment is usually at network segregation points, where other security devices, like a firewall, would be placed. This is because their primary concern is monitoring traffic passing between differing networks and network segments. A very common way in which inline sensors are deployed is by integrating them into one's firewall. By doing this, no further hardware is required, and the intrusion detections software can function from the hardware device that is already in place on the network. If being placed on the network as a separate device, it would be best advised if the firewall was in front on the inline IDS. Doing this would ensure that the IDS has only to be concerned with detecting any intrusions that manage to get beyond the firewall. In turn, they will have less traffic to process, reducing their workload. Passive deployment monitors a copy if the network traffic, rather than the actual traffic itself.

The aforementioned IP spoofing can be minimised by filtering incoming traffic, and minimising open ports. This can be done by removing any unnecessary programs or services that use TCP or UDP. In some cases, it may even be worth blocking all incoming ICMP traffic, to prevent against attacks based on flooding, unless there is any specific reason as to why such inbound packets types need to be allowed.

Password Crackers

There are a variety of password crackers freely and easily obtainable today. While many of these programmes are labelled as password "recovery" tools, their use, when malicious, can pose a very serious threat to a system security and an individual's privacy and data protection. Password crackers function off of a variety of methods. Dictionary attacks are used to check the password against all the entrants of an electronic dictionary. This method can be effective if the user has not taken the precaution of selecting a non-dictionary word as their password. Users that do select non-dictionary words and create alphanumeric passwords need not worry about this method, as it only works when the password matches against a dictionary word.

Hybrid attacks attempt to overcome the challenge set down by alphanumeric passwords. Hybrid attacks function off of the same principal as dictionary attacks, except that they also allow for numeric sequences ad the end of the password's alphabetical string, as is usually the case in passwords. Once again, hybrid attacks are dependant on the user having selected a password whose alphabetical string matches a word contained in an electronic dictionary. Long and complex numeric strings, particularly contained within the word, often replacing letters (1 for I etc) can also add complexity, further decreasing the chance of the cracking tool's success.

Brute force attacks look to overcome the challenge presented to hackers by password complexity. Attacking a system using a brute force attack can overcome this, however, it can often take a considerable length of time depending on password complexity. Brute force attacks simply look to identify each key individually, which is why it takes such a length of time, as there is a high volume of possibility. The hardware involved can also be expensive at times.

There are a variety of password crackers available for a host of uses. There are password crackers available for all platforms, as well as network crackers. Some of the more popular password crackers out there include L0phtcrack, THC Hydra and Ophcrack. Popular network password crackers include Aircrack, a WEP/WPA security cracking tool that can recover keys through the collection of packets.

To secure against the threat of password crackers, the first thing that a user should do is create a complex alphanumeric password that is not a common word. For example, C0rca1gh18 would be a good password to use. Passwords should never be written down or revealed to anyone, nor should they be transmit electronically through any means. There is no reason why one should want to reveal their password to anyone.

Defending against wireless network password crackers has become easier with the emergence of WPA2. Also, a lot of the newer hardware supports access through MAC address filtering. By implementing this form of security, a wireless network does not require a password, with access restricted to only those hardware devices predetermined by the network administrator.

Trojans / Worms / Viruses

Though many people believe that Trojans, worms and viruses are the same, there are differences between the three. They are all malicious, and do all pose a threat to a system's integrity, but they are not identical in makeup by any means.

A computer virus behaves in the same way that a human virus would. Once the body, in this case the computer system, has been infected, it spreads. Viruses spread throughout the system, continually compromising and corrupting, before often moving onto new systems through eMail, file sharing, further spreading, now across multiple systems. Viruses can find many point of entries to a system. Many of them enter a computer through an executable file. Once a virus has compromised a system, the effects of that virus can differ greatly dependant on what it was designed to do. Some viruses simply wipe out a system, others will look to attack a system's hardware directly, or simply antagonise a user in some way.

Worms and viruses have a subtle difference. Viruses infect a system and spread with the aid of human action. A virus infects a system through the user performing some action, for example, like foolishly executing a suspicious file. For this virus to spread, particularly to other systems, other users must perform the same foolish action. Essentially, they are dependant on human error to thrive. Worms on the other hand, though often not as detrimental to a system than a full blown virus, are self contained and able to replicate without the need for file execution and human actions. Some worms have been known to even access eMail address books and send out copies of itself. So while worms aren't always as seriously a breach as viruses, they are a greater threat as even the cautious user is open to their infection. Most worms are intended to compromise a system's performance through the process of replication to the point that a hardware's resources are being utilised by the infection, rather than by the tasks for which it was intended.

A Trojan gets its name from the old legend of the Wooden Horse of Troy. This is because their attacks are based on the same principle; the threat arrives at the intended target disguised as something that it isn't. For example, an eMail might contain a file that looks likes something that the user might deem useful or interesting in some way. A file on a website may appear to be some kind of free software, while in truth it will contain some form of malicious code. Trojans usually do one of two things, or both, after having successfully compromised a system. They will either behave like any other type of malicious code would and damage the system's data, or alternatively they might create a backdoor. A backdoor allows a malicious user access to a system, so that they can either damage the computer themselves, or access the personal data and compromise the privacy of the victim.

Some examples of well-known infections of these sort include the aforementioned MyDoom worm, and the ILOVEYOU virus, spread in 2004. The ILOVEYOU virus is estimated to have cost approximately $10 billion in damage[4]. Other large scale compromises to Windows platforms came in the form of the Lovesan virus, also known as the Blaster Worm, and the Vundo Trojan, which affected both a system's performance and loaded a computer with both adware and spyware.

Successful defence against these threats is dependant on a combination of common sense and appropriate security measures. Users should not execute anything that they deem as being remotely suspicious. They should also ensure that anyone using their system is either aware of such threats, or operating in an account that denies them the privilege to execute such a threat. Systems should also have an anti-virus software that has regularly updated virus definitions. A firewall is also crucial, as is a spam filter. There are a variety of packages available for this type of security, both free and commercially available. Two of the most popular proprietary licensed products available are McAfee and Norton Anti-Virus, while AVG offer a very well respected free version of their suite to personal users.

Internal / External Threats

Systems need to be secured from threats originated from both internal and external entities. Malicious intent can come from a hacker at the other side of the globe, or from a disgruntled employee in the very same office. The threats that are posed internally are as numerous, and as potentially detrimental, as those from external sources. I will now examine a number of these threats.

The first of these is abuse of privilege. There are various scenarios in which system administrator must keep watch for the abuse of privileges. The most obvious of these would be an unauthorised user gaining root or administrative access. There are numerous examples of this, one such incidence being the kadmin bug found several years ago in the Kerberos 5, an authentication and privacy package, administration service[5]. The exploit allowed malicious users to gain unauthorised root access via the kadmin daemon. Such privileges can also be often granted inadvertently, either by human or technical error. The majority of technical issues that caused such have been mostly filtered out in the newer operating systems, but one must always allow for the element of human error. An administrator could easily apply to wrong privileges to a user, particularly in a large corporation with very high account volumes. A user who received such privileges inadvertently may decide to use such maliciously. Administrators may also grant elevated privileges intentionally, often in the case of contractors.

Other internal threats come from old and back door accounts. As is often the case, once an employee leaves a company their account stays enabled for some time. A lot of administrators will have a policy that ensures such dormant accounts are closed down the minute the employee leaves, but some accounts often slip through the cracks, and some administrators simply don't bother with such procedure. This account is now a potential threat, and in the case of it remaining enabled, could be used maliciously by either the former employee, or other persons still working within the company. More critical perhaps than the issue of old accounts, is that of back door accounts. Ideally, every account created on a system will be documented and be recorded somewhere. The creation of such will follow some form of formal procedure. However, administrators are perfectly capable of just creating accounts, and granting to these accounts any level of rights that they deem fit. The creation of such unregistered accounts are known to only one person - the administrator who created them. Should that administrator decide, for whatever reason, to misuse this account, a considerable level of damage could be achieved.

Data modification and configuration changes also pose an internal security threat. It is critical for any system to monitor for the modification of sensitive data. In the event of any malicious alterations, the guilty party can be identified and the modification rectified. Issues with data modification also pertain to record theft, which could lead to company and personal privacy issues. Any unauthorised configuration changes to a system could prove critical to its functioning.

Protecting against internal threats involves the implementation of proper policy and technology, including the aforementioned intrusion detection sensor deployment, that ensures an administrator can monitor for such activity, and in the event of any occurring, identify the culprit, and more important, the change they have made or the threat they enacted.

External threats differ to internal threats in the sense that they do not have the same level of access, and thus are forced to attack, in most cases, through the system's network. These threats can come in the form of many of the aforementioned attacks. Hackers, external threats, will try to access the system, issue denial of service attacks etc. Some of the other threats from external entities include unauthorised access, packet flooding and bandwidth theft.

References

[1] http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci549024,00.html

[2] http://www.smh.com.au/articles/2002/10/23/1034561535264.html

[3] http://en.wikipedia.org/wiki/MyDoom

[4] http://en.wikipedia.org/wiki/Timeline_of_notable_computer_viruses_and_worms

[5] *Kerberos 5 unauthorised root access to KDC host vulnerability* from LWN.net. Viewed Online at http://lwn.net/Articles/7612/