

# Vulnerabilities &;
# Vulnerability Scanning


Ken Houghton

# Vulnerabilities & Vulnerability Scanning

***A white paper to help small to medium businesses understand the issues and some proposals to help resolve those issues.***

***Author Ken Houghton***
***July 2003***

Vulnerabilities and Vulnerability Scanning

## *Introduction*

This white paper will discuss the benefits and pitfalls of Vulnerability Scanning and will suggest an approach suitable for small and medium-sized businesses, as well as discussing the possibility of buying this as a service from a specialist agency. The intended audience is the systems administrators of these companies, who will typically have a good level of technical understanding, but limited security experience. This leads to the requirement to explain the subject in a manner that will ensure their interest and an understanding of the potential benefit to their business.

Ken Houghton
09/08/03



# Table of Contents

## *Vulnerability Scanning – what is it?*

Vulnerability Scanning is the art of using one computer to look for weaknesses in the security of another computer - so that you can find and fix the weaknesses in your systems before someone else finds that there is a security weakness and decides to break in. It's a bit like a shop keeper making sure all the doors and windows are closed and locked, the money is in the safe and the alarm is set, before closing up for the evening.

The task involves running a program (a vulnerability scanning application) on one machine and then connecting, via a network, to the machines that you wish to check. It sounds simple enough and if you are at all worried about IT security (and if you're not – maybe you should be) it sounds like a good thing to do. But, as a small or medium sized business, do you have the skills / resource / time to undertake such checks and is it really necessary. Also, what will you do with the output from such checks?

This paper will take you through some of arguments as to why you might consider vulnerability scanning to be a good thing to do, either by learning to do it yourself or by contracting others to do it for you. We will look at some of the common misconceptions that are prevalent in society today and will consider their impact on your security posture.

## *Security Vulnerabilities – what are they and where do they come from.*

When software vendors create a new application, they are often writing millions of lines of software code. These are the instructions that tell the computer exactly what to do. Although they are (generally speaking) developing their code to the current best practices, mistakes are made and errors slip through. Sometimes it is not even realised that the way the code has been written is erroneous. This fact only comes to light later when someone discovers a way of manipulating the code to do something other than its planned task.

When that software is released for public use, it is open to people using it in many (and often unplanned for) ways. This can lead to the discovery that by doing something totally unexpected with the software you get an unexpected result. Sometimes that result is not only unexpected but also unpleasant and by utilising these flaws in the software intruders or 'hackers' can gain access to your machines. The main aim of an intruder is get your system to give system level access to the intruder, preferably as if he were logged in as 'Root' or 'Administrator' (depending whether you are running Unix or Microsoft operating system).

## *System Security – is it really a concern?*

So - you are now using a computer for your business, maybe not in any large way, but you may have customer contact data, orders, invoices, artwork and letter heads, standard letters, brochures, pricing and so forth now on your systems. Maybe you have multiple PC's for your workers and they have been networked together. Perhaps you have decided that you need a presence on the Internet to give you a way to reach a world wide audience at a fraction of the cost of traditional advertising mediums, as well as providing an always available order taker, so you can receive orders even in your sleep.
However, perhaps you feel you are only a relatively small business, compared to the giants of industry, so do you really need to worry about security vulnerabilities and the undertaking of regular security health checks? No one you know of has ever been "hacked" (well they have never told you that they have) and you have nothing of real value on your web site; that is all on your office machines. So why would anyone want to bother you?

Well, there are lots of predators out on the Internet, just as there are in life in general and they have many different motivations. Some want to steal your secrets, some want to just take you off the air, some would like to make use of your computers for their own ends and some would just like to break in so that they can boast to their friends that they did so. In general though, the relative size and importance of your business is of little or no concern to the attacker, indeed in many cases they would not know.

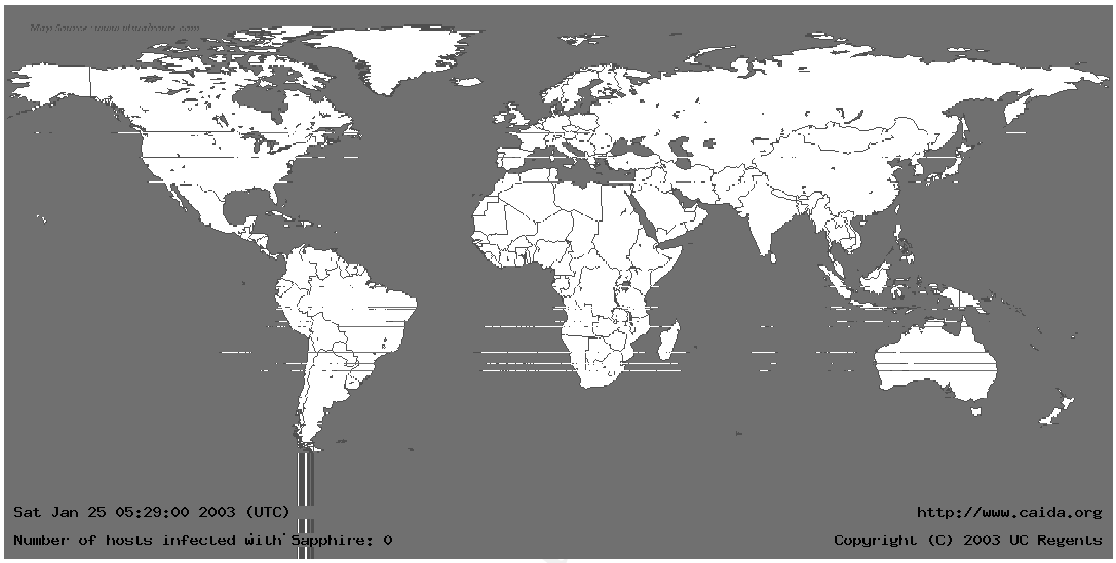
Furthermore there is a class of threat that doesn't even make a conscious decision to attack you (or anyone) specifically it is known as the Internet Worm.

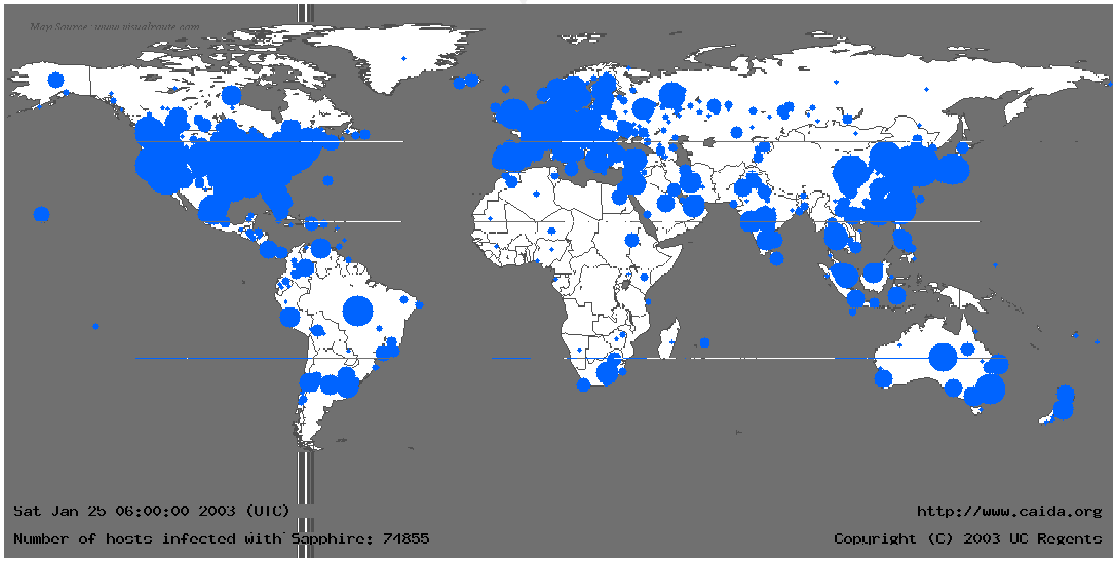
***Internet worms***, these have been around since November 1988, when Robert Morris, Junior, a graduate student in Computer Science at Cornell University (USA), wrote an experimental, self-replicating, self-propagating program (later dubbed a *worm* by the popular press) and injected it into the Internet. There are a number of accounts of this land mark event in the history of the Internet, you may find Katie Hafner and John Markoff,'s book "*Cyberpunk: Outlaws and hackers on the computer frontier",* published by Simon & Schuster, (1991) an informative read.
It is sufficient to say here that this small piece of code affected the majority of the Internet and brought large sections of to its knees.
A worm is a piece of code that effectively roams around a network, looking to exploit known vulnerabilities in machines. (In the case of the Morris worm it was weaknesses in the "sendmail" and "finger" daemons that were exploited).

Once a worm locates a vulnerable target, it usually has a mechanism to infect the host machine and then to start to use that machine to begin looking for other machines that are similarly vulnerable. In this way worms are able to spread across the entire Internet at frightening speed and with no regard as to the size and type of business they are infecting. The Code Red worm infected more than 250,000 systems in just 9 hours on 19 July 2001. However, The Sapphire (also known as SQLSlammer) Worm was the fastest computer worm

in history (to date). As it began spreading throughout the Internet, it doubled in size every 8.5 seconds. It infected more than 90 percent of vulnerable hosts within 10 minutes. At its peak, which came only 3 MINUTES after its release onto the Internet, it was conducting 55 Million scans per second in order to seek out further vulnerable hosts. Ref http://www.caida.org/outreach/papers/2003/sapphire/sapphire.html

The pictures below show the speed of spread of this worm – please note the times in the lower left corner.



## *I'm safe, I've got a Firewall!*

So, you have a firewall - right, so that's OK then, all your security worries are dealt with at the edge of your network? ……Well, err no actually, they're not! Firewalls are now part of most people's perimeter defences and they generally do a good job. That is always assuming that they have been set up properly and that they are regularly checked for misconfigurations. (Do you know who set up your firewall and do you know have faith in how skilled they were at that task?)

Firewalls work primarily by only allowing access through particular ports and sometimes from only certain machines[1].

Probably the most commonly known ports are port 80 (HTTP – this is the port most Web conversation takes place on) and port 443 (HTTPS – this is the port that the most popular form of encrypted web conversation takes place over.)

Now to enable your Web server to talk over the Internet, you are most probably going to need port 80 open to everyone. If you want to have encrypted conversations between you and your customers, then you will also need port 443 open (this is useful when say transferring sensitive information like credit card details across the Internet).

You may also want to use say port 23 (Telnet) to transfer data to and from your web server, however, you probably don't want to give access to this port to just anyone on the internet as this may be abused.
Most Firewalls are now more sophisticated than the simple allow / deny model discussed above. They have another capability which is to only to allow access from specific machines. Machines are identified by their IP address and so the Firewall is told which IP addresses are allowed to talk over any particular port. If your address is not known to the firewall then you cannot talk through that port.

Ok so now your customers are getting to your published data and the firewall is allowing you to get to the server to manage your content whilst keeping external people away from the other ports that you don't want them touching. So, as you thought, you are safe from harm?
Well, not exactly. Although you have restricted anyone on the Internet from accessing any of the thousands of ports available on your machines, you have had to leave some ports open in order to be connected to the network.

---

1 . [1] The principle of ports can be a little difficult to grasp, particularly when you find out that there over 64,000 (or even 128,000 if you take TCP and UDP ports but the principles the same) of them. To explain firewall ports, I like to think of an actual brick wall. With all the bricks in place nothing gets through – your safe, but also you are effectively off the air, disconnected from the network.
So to allow business to be conducted, you need to take out the odd brick to allow traffic through. Which ports you open up is dependant on what types of traffic you want to flow into and out of your organisation. The traffic that is allowed to flow is said to be associated with a port (or brick) number, and only that port. Now once you do that, you now give your customers a way in to your site (but you also give a way in to anybody else that wants in!). Depending on what that port allows access to, this may or may not be desirable. To ensure that the right bricks can be removed, each one is numbered and has a service associated with it. These services are agreed and published, http://www.iana.org/assignments/port-numbers so everyone knows (generally speaking) what each port does.

Now what happens if those predators have found ways of attacking your systems via the always open ports (like 80 & 443)?
Your firewall is not going to protect you as you have to keep those ports open to be a website and/or do business. Now once the attackers have figured out how to attack you on these ports (be they human or machine) they are automatically past the Firewall and are inside your network.

Now, as we are past your firewall, your security posture is dependant upon the strength of security of your applications.

## Application Security

There are many types of application that work with many different ports, however, to keep these examples reasonably simple we will continue to consider an application that is designed to talk on port 80 in order to serve web pages. However, as we have shown that our perimeter security (Firewalls) isn't a total security solution in itself, we now have to consider what else is included in our security posture and how that is built upon the security of our chosen application.

As we discussed earlier, applications and their underlying operating systems are very complex pieces of software. As soon as they are released to the market place, people can start investigating their properties and trying to find weaknesses in their construction. Once a weakness is discovered, it may be shared amongst groups of computer hackers and refined into a practical exploit. What happens to the information then is very dependant upon the hackers and their motivation for having done this work. Some may keep it to a very close group and use it for their own ends (commercial espionage, government information warfare are just two options that spring to mind – there are many more), some may publish the information widely, in the hope of fame and recognition in their skill at doing such things, others will just report it to the vendors in order that the software can be repaired or patched before the weakness is exploited.

## "But surely this is very hard to do?"

Certainly, the knowledge and skill required to discover a weakness is quite advanced and beyond the capabilities of most normal Internet users. However, there are sufficient people about with the right skills and tools to make this now a common occurrence. Once an exploit is discovered and published, exercising it is often trivial. Indeed, often when exploits are announced, the hackers that announce it also release a "toolkit" to make the attack process trivial to undertake. See this article http://www.landfield.com/isn/mail-archive/2001/Feb/0131.html on how "Script Kiddie" toolkits are becoming a menace.

A common example of such a vulnerability would be a buffer overflow (or overrun) exploit.

This is where a programme, for this example we'll say it's a web server, is discovered to be susceptible to having a URL entered which contains a huge amount of characters. In this case it's 4000 "A" characters. The use of such a large URL would be way outside the bounds of expected use of a web server; however, the firewall will pass the data to the application as it travels over port 80 which is open to all. The application software allows this entry, even though it is way outside of design parameters and passes it on unchecked to the programme. The programme has memory space allocated to receive the input of a URL, it is known as a buffer, which is just a section of memory. When the extra long URL is entered it fills up the memory space (buffer) allocated to the task, but when the buffer is full, instead of stopping, it continues to write into the adjacent memory space. This is the error in the software; the buffer is unchecked for length. Now just randomly overwriting a section of memory is likely to cause the programme to crash. This could be the intention of an unsophisticated attack and would result in a "denial of service" attack.

Therefore, it can be seen how trivial it can be to stop servers functioning on the Internet if they have unpatched vulnerabilities. By sending a single entry to a web server you may be able to stop it working completely (at least until the software is restarted).

However, by carefully crafting the URL a clever intruder can over write the adjacent memory space in such a way that the system doesn't just crash, but fails in a particular way. Namely that it will undertake a planned for action, usually an instruction that is appended to the end of the URL. Here is an example of such a URL

```
GET /iisadmpwd/..%c0%af../..%c0%af../..%c0%af../winnt/system32/cmd.exe?/c+dir+c:\
```

Ref http://www.theorygroup.com/Archive/Unisog/2002/msg00851.html

So, having done something as simple as enter a URL, our intruder has potentially gained Administrator privileges to your Web Server. Having gained such privilege he is now at liberty to exploit our web server in just about anyway he sees fit.
Clearly this is a highly undesirable state of affairs and one that any system administrator would like to know about and fix as quickly as possible.

The bad news is that this type of attack cannot be defended against by the use of Firewalls as the attack takes place via the always open ports (port 80 & 443). This type of vulnerability is one of biggest growth areas of reported security vulnerabilities.

So what is your best defence against such devastating events?

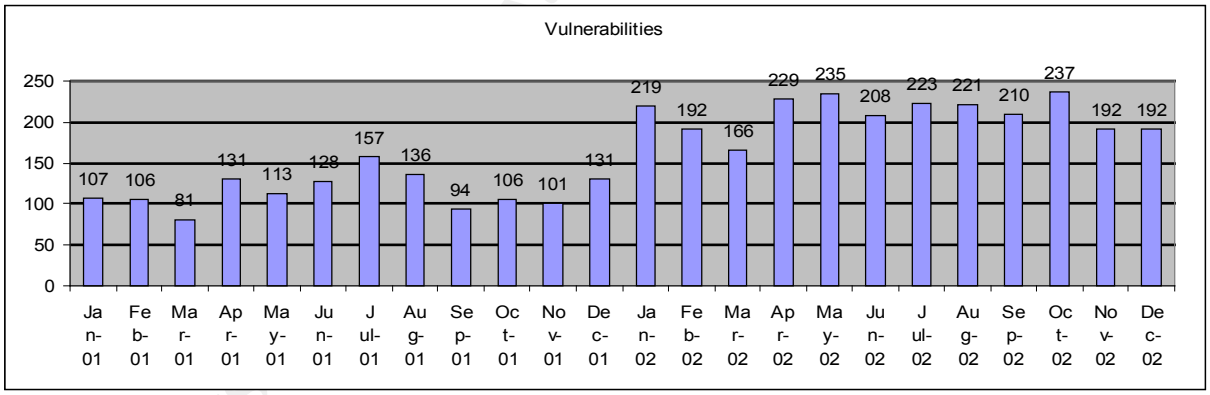
## *Knowing the state of your systems*

To determine what defence we require, we must first understand the weaknesses we are trying to defend against and where possible, remove those weaknesses.
So firstly we need a source that will tell us about all the known vulnerabilities and what patches are available to address these vulnerabilities. If we are using only one manufacturers software (say all Microsoft), then it may be sufficient to just monitor that manufacturers site regularly for updates / patches or sign up to a notification scheme that they may run. If, however, you have multiple suppliers, you are either going to need to monitor multiple vendors sites for patches or to sign up for a consolidated supply of information like that provided by Security Tracker (ref http://securitytracker.com/server/info?1245+learn/endusers.html).

### Sizing the problem

In 2002 leading antivirus company, Symantec, documented 2,524 vulnerabilities affecting more than 2,000 distinct products. This is 81.5% higher than the total documented for the whole of 2001.
Below is a graph of the total new vulnerabilities found per month from January 2001 to December 2002.

. Reference - http://www.securitystats.com/reports/Symantec-Internet_Security_Threat_Report_vIII.20030201.pdf




So as can be seen, there has been on average, approximately 150 security vulnerabilities recorded per month over the last 2 years and there is nothing to indicate that this trend will reduce in the future. This means that keeping your site patched and up to date is a considerable task; however, at this stage you still do not know fully the size of the task. Calculating an average time required to apply each patch is difficult, as they can vary significantly depending upon the type of patch, the number of servers being done at one time and the OS being worked upon. However, for the purposes of estimation I will assume that typically there will be a 1 : 1 relationship of fixes to vulnerabilities and I suggest you assume 2 hours per patch in order to cover obtaining the patch, managing change control, applying the patch and tidying up the paperwork afterwards. So in order to determine the amount of work you now have, you will need to know the number of patches that apply to your

particular servers and then multiply that by the number of servers and multiply by 2 (hours). (Time = Vulnerabilities x Servers x 2 hours)

It would be quite usual for this to be quite a large number of hour's worth of work, so what do you need to know next in order to prioritise this work?

Well it would be very useful to know which patches really apply to you and your servers, and how serious each one is as you don't want to have your scarce technical resources tied up applying patches that are not required. One way to do this is to have a central log of all severs, the software type and versions on them, along with all the configuration options, patch levels and other relevant settings listed.
This would involve taking each patch and trawling through the database looking for systems that do not have this patch. However, this is an extremely manual task and its success is reliant upon the discipline of everyone involved in managing the servers to keep the database up to date. With the volumes of changes that are typically made to servers and the volume of patches that are continually being released the chances that this method would be efficient and accurate are slim.

Alternatively you could try logging all the traffic on those ports and analysing the logs for traces of intrusion. However, the chances are that the volume of data you will generate (especially on Port 80) will swamp the resources available to analyse it and even if you do see something it will be after the event. So although you may know that you have been hacked, it would not necessarily help you in preventing the hack, although you would know what to fix to stop that hack happening again.

What is really required here is a way to automatically interrogate each server in order to look for evidence of missing patches. Then we need to prioritise the most urgent patches and the most vulnerable machines in order to ensure that our scarce technical resource is working as effectively as possible at securing our web presence.

## *Automated Tooling*

Having determined that we need some automated way of testing our servers, we need to identify what tools exist and what the limitations of these tools are. There are a number of different products out there that will do this to a greater or lesser degree. They break down into two families:

Those which have a client and a server or manager portion, where the client scans the machine from within and then the server recovers the results (like Symantec Enterprise Security Manager ref www.symantec.com), these are often referred to as Policy Compliance Tools.
Those tools which scan the server without a client and which try a whole catalogue of 'events' in an attempt to obtain an inappropriate response (like Nessus or Saint) are known as Vulnerability Scanners.
Reference www.nessus.org and
http://www.saintcorporation.com/products/saint_engine.html

Both of these types of tool rely on a "profile" against which the servers are checked.
Obviously the quality of the check will have a high dependency on the quality of the "profile", its accuracy and its coverage, but also will be dependant upon the operator to correctly setup the scans and to interpret the results.

The two different types of tool will potentially give us different views of our security status, dependant on where they are looking from and what they are looking for.

## Policy Compliance Tools

The Policy Compliance Management tools will look for the presence of known patches by comparison to a set of signatures. As they have a client which runs locally on the server they are able to inspect deep into the Operating System and check things like registry settings etc in order to determine what is and isn't present against a given profile. These types of tools have wider usages than just security patches, they are able to interrogate servers for a whole range of settings that relate to a local security policy - such as password strength and complexity for example.

However, what they cannot always tell you is whether or not the missing patch is **required.**

Take a recent vulnerability as a case in point. On the 17/03/2003 Microsoft released patch MS03-007, (ref http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms03-007.asp)
in order to address a vulnerability which had been found in the wild that had been used to gain access to an American military server. The service being compromised by a carefully crafted buffer overflow attack was a service called WebDav. (World Wide Web Distributed Authoring and Versioning (WebDAV) protocol, defined in RFC 2518) reference http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc2518.html

WebDav is a feature supported by Microsoft's IIS web server that allows a remote user access to be able to add & modify web content over the internet following appropriate authentication. However, by sending these servers a carefully crafted buffer overflow, it is possible to make the server crash or execute code of the hackers choice.

However, not all Windows servers will be web servers and not all that are web servers will use WebDAV.
Therefore if you ran a policy type scanner against your systems, it would probably alert you to the fact that you need to patch ALL your Windows servers, where as in reality you only need to patch those that have IIS and WebDAV enabled.

So as can be seen, although Policy Checking tools definitely have a place in the arsenal of security tooling, their effectiveness at spotting open vulnerabilities is limited.

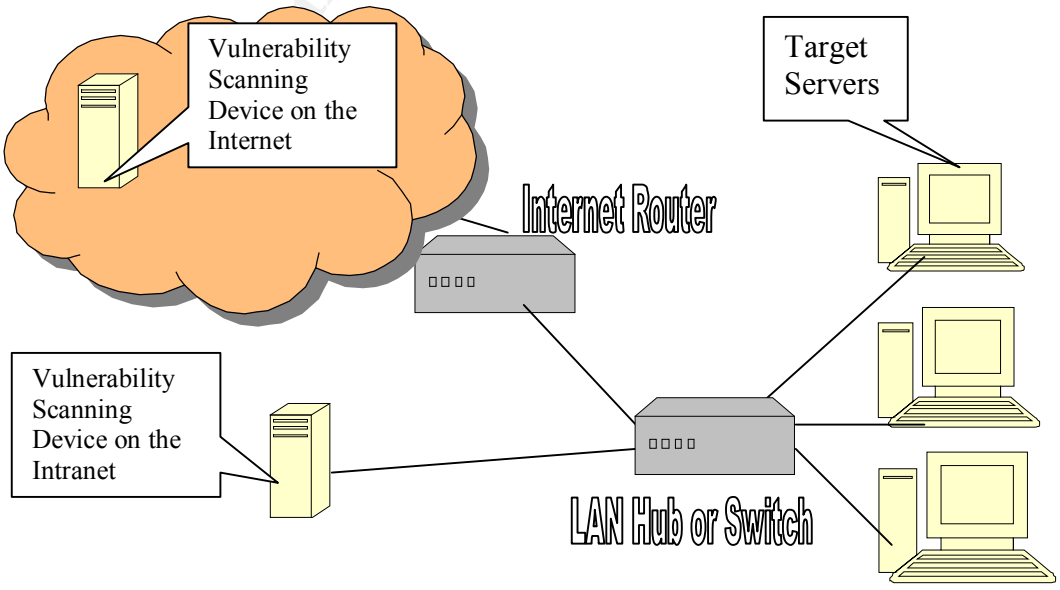
## Vulnerability Scanners

An alternative tool is the Vulnerability Scanning Device (VSD). This tool works without having any predefined knowledge of your systems, other than its IP address. There is no need to load or maintain any client software on the servers; everything is done over the network. With just this information the Vulnerability Scanner will attempt a whole suite of events to see what can be done with the target box.

Different versions of VSD will operate slightly differently, however, what we will look at here is the common steps that a VSD may take to analyse a group of servers that reside on a common subnet.

### *The scanning process*

The first thing that we require is a network path (or route) between our VSD and the target servers to be scanned. When checking Internet facing servers the best place to have your VSD is out on the Internet so that you get the same view of your server as the rest of the world, i.e. if you can see a vulnerability, then so can anyone else that cares to look. If you are looking at servers that are just within your organisation and have no Internet connection then the VSD will need to be located on an internal LAN that has a route to the boxes you wish to scan.



You may wish to consider scanning both externally and internally to your network. The reason for this is that many companies these days use private or non-routable addresses to address their Intranets. Therefore only those few machines that are allowed to have external IP addresses that would be visible from the Internet and could be scanned. By including a scanner locally you are able to check the security of your machines on the private networks as well as the public address space. Remember that not all attacks on your network have to emanate from outside your network. There are a number of ways that someone may gain access to your local network – maybe they are a contractor or service engineer with direct or remote access, for instance. What about if your employees add modems to a system for "support purposes" and someone else finds it? Or maybe it is just a disgruntled employee who has access to the network anyway (but not all of the systems on the network).

Having established that our VSD can route traffic to the network(s) you wish to check, you need to consider whether we want to target individual servers or check all servers on the sub net. If we target individual servers we are faced with the task of maintaining a list of those servers IP addresses. This can lead to errors, perhaps when a new devise is added due to expansion or the failure of an existing server. It could be quite easy to forget to add the new servers IP address into the scanning tables and as such that device will not be scanned. However, if we select to scan the whole subnet, then we will be sure of discovering all servers that exist on that network. This saves on administering the IP list and ensures that no-one can add a new device and "forget" to get it scanned. This also acts as a useful check for the administrators on what is actually connected to his network, rather than what he believes is connected.

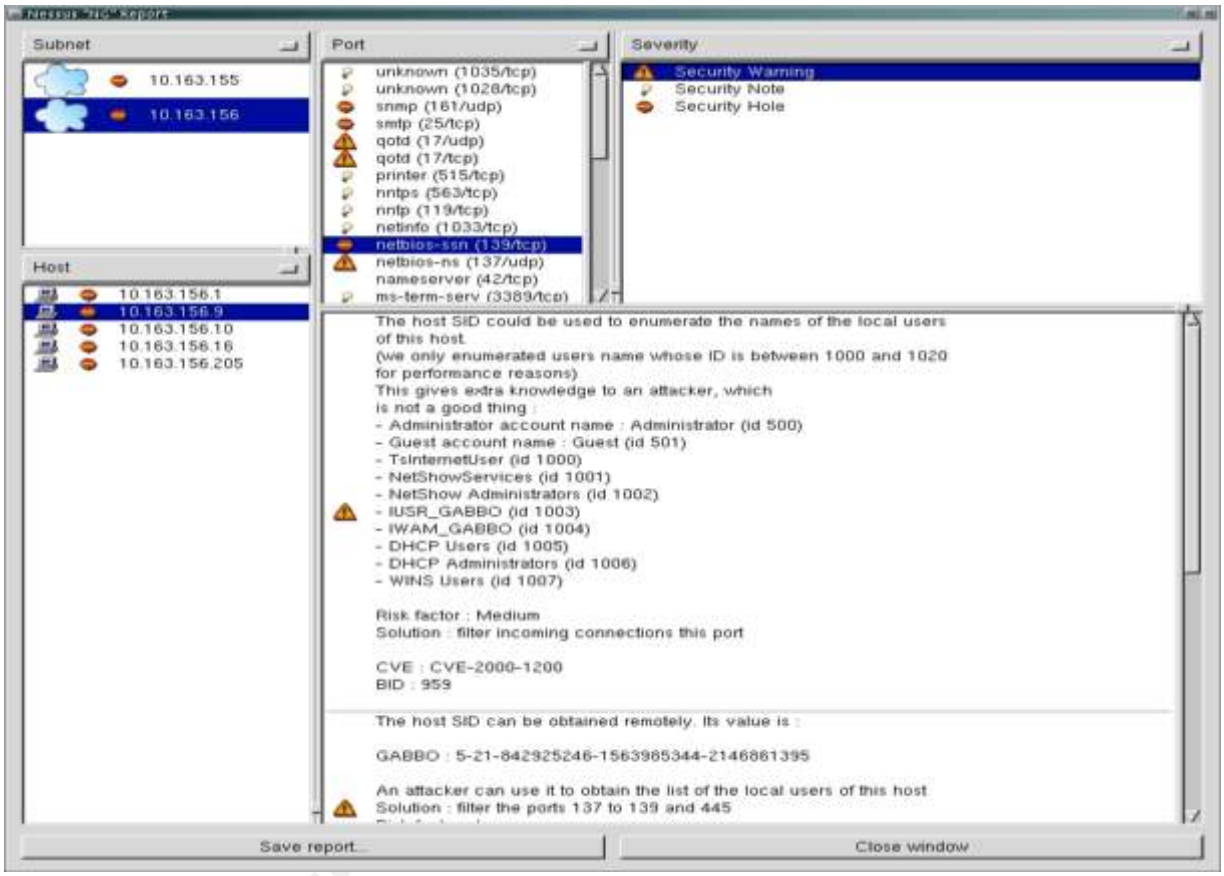
In either configuration, the normal first step is to do a limited port scan of each address in order to see whether a server exists on that address and is listening for connections. Having determined that something is listening on that port the scanner logs that IP address for later and moves on along either its pre determined list or sequentially through the subnet. This enables the scanner to obtain a quick listing of available servers and stops the scanner wasting time trying to run a whole suite of vulnerabilities against a machine which may be off line or non existent.

Once it has a list of the servers that are attached to the network the scanner goes back and undertakes a more complete scan of what ports are actively listening for connections. For the most comprehensive scans this scan may be all 64535 TCP and UDP ports, however, scans are often tailored to a list of the more commonly used ports in the interests of time to complete the scans. It is important to understand any limitations on coverage that you select as it may lead to ports going un-scanned and hence vulnerabilities going undiscovered. One way to cover this is to have a selection of well known ports listed in your scan profile and then to do say a further 5% of the remaining ports. In this way you will eventually scan every port on every machine.

Having established which ports are listening, the scanner now runs its suite of vulnerability checks against each server to see what access it can gain.

Once the scanner has run it will provide a raw output of events that it believes were noteworthy.
From this data the VSD will create a report and will categorise the results according to the script it was working from. So you should now get a list of vulnerabilities, ranked from high to low, based on the ease of exercising that vulnerability and the amount of access doing so would give you.

Here is a screen shot of a Nessus report.




## Nessus Report

Now you have the start of a priority listing for your patching exercise. Appendix A gives a high level Nessus report that categorizes the results from all hosts discovered on a given network. This indicates the percentage of High / Medium / Low vulnerabilities overall and gives you an idea of the worst servers and the most dangerous services. However, before jumping in and applying patches it is worth checking the accuracy of the data presented.

## Interpretation of Results

At this stage, we cannot say that these are definitely vulnerabilities, but they are certainly responses that are in need of investigation. The reason that we cannot declare all this output vulnerabilities straight away is because the VSD

only sees the response (or lack of response) from the server to a given event. From that response the VSD has to extrapolate a result.

For reference there is an example of a detailed Nessus Report in appendix B.

If the response received by the Scanner was within the expected parameters, then the test will either pass or fail. However, sometimes the server does something that is outside of what is expected and then the VSD has to make a judgement based on the information it has.

Let me see if I can clarify this a little with some examples.

Let us say that a particular event that has been run by the VSD is expecting to receive back a web page with Error 404 on, if the system is not vulnerable to this particular event, any other response from the server and the VSD will interpret this as a vulnerability. However, on running this event lets say that the server responds with an Error 200 page, not Error 404.
Now is this a sign that the server is actually vulnerable to this event or is it a sign that someone has written their HTML error pages to a different standard than that which the scanner was expecting? Clearly this matter will need to be investigated by the System Administrator that owns that server. If it turns out that this was as a result of a difference in error page coding then the event can either be suppressed from future reports or the scanning software could be updated to accept this as an expected response. If this turns out to be a real error, then the sys admin will need to apply the appropriate patches to the server.

Other cases are sometimes more obvious. For example an exploit is run that is say designed to elicit an illegal response from a Microsoft Windows server. However, when it is run against say a HP-UX box, the VSD gets a response that is not what is expected. The VSD will flag this as a vulnerability; however, the response is just the HP-UX box telling the VSD that it doesn't understand the request. Again it may be possible to tune the VSD to understand this response in future; however, it may prove difficult as you may need to find the responses for every different version of server and its operating system.

Finally, what does the VSD assume if the server just stops responding? Did the server terminate the session, as it believed it was an invalid entry? Or did the server just fail, indicating a possible denial of service or buffer overflow attack was successful?
The VSD has no way of knowing, so it will have to suspect the worse and flag it as vulnerability as in this example from the detailed Nessus report in Appendix B:-

**Warning found on port http (80/tcp)**

*This port was detected as being open by a port scanner but is now closed. This service might have been crashed by a port scanner or by some information gathering plug in.*

To really understand what the server did at that point, a system administrator

is going to need to look at the event logs and correlate the events together with the actions of the scanner. Only then will you be able to say for sure whether or not you discovered a security problem.

Once the report has been reviewed for false positives then you can consider the report as a priority list for patching. Along with the High / Med / Low category ratings, you should also consider reviewing the SANS / FBI Top 20 list of vulnerabilities (http://www.sans.org/top20/) to help you further prioritise as these are the most commonly exploited vulnerabilities in both Windows and Unix at any given time.

Just a word about false positives; they are not uncommon when working with VSD's and this is one of the reasons that many people shy away from their use. However, there is a conundrum to be considered, whether it is better to err on the side of false positives or false negatives. In other words it is better to have false positives (events reported and checked that are not true vulnerabilities) rather than false negatives (vulnerabilities in existence that are not picked up by the VSD, creating the illusion that the Server is not vulnerable).
Obviously in an ideal world you would have neither and all the data would be accurate. However, I hope I have shown by some of the examples that the variety of things that could be affecting the scanners ability to accurately judge an event is vast and therefore, it is almost certainly better to err on the side of caution.

## *Scanning as a Service*

Businesses have never been more reliant on computers and therefore on the ability of computers to keep their valuable data secure. As this paper demonstrates, keeping systems secure is a constantly moving target and businesses that do not pay regular attention to the state of their systems will sooner or later suffer some form of break in. The potential for that incident to be life threatening to the business is a direct correlation between the number of vulnerabilities that businesses systems are open to and the seriousness of those vulnerabilities. Consider for a moment what information your business has stored on computer and how devastating that could be if a competitor got hold of that information or if that information was denied to you.

However, like most other businesses you have a continual demand to keep costs low in order to maximise your operating profits, therefore expenditure on IT staff is limited and those staff need to cover many duties within the organisation. This can lead to a demand for generalist IT staff as opposed to specialists in say the area of Security.

Therefore, businesses should consider the option of buying in certain high value services, especially those that they are not staffed and trained to do, or that are not part of the daily work of the company.

Security Assessment and in particular Vulnerability Scanning is one of the services that should be considered for this.

In order to do Vulnerability scanning well, a company would need a number of specialist staff who have a strong grounding in IT Security. They will ideally need to maintain scanning devices on the open Internet and be sure that these devices are maintained to the tightest levels of security to prevent the scanners themselves being compromised. They need to understand the Scanning software and how to rapidly update the signature files to contend with the ever increasing list of vulnerabilities and they need to have the time to review the output of the scanners and the capability to analyse the reports and remove the false positives from the findings.

Once that is done, then the resultant data is useful to the system administrators who can then go and patch the systems.

This in itself is a major task and is the one that really benefits the business and therefore is the one which businesses should put a high level of focus.

Therefore, it can become a choice between staffing your own Security Assessment team and employing someone to do some specific security assessment work for you and provide you with a list of results that can help inform your action plan.

## *Conclusions*

Therefore, as can be seen, the whole issue of security vulnerabilities and patching is a complex and serious business. Hopefully this paper has shown why it is imperative to have a regular review of the status of your systems as compared to the latest vulnerabilities and why it is imperative that you keep your systems up to date with their patches.

I have shown how the Vulnerability Scanning Device gives you a unique view on your network and how that view is very similar to the view that anyone on the Internet (or your network if you are Intranet scanning) can obtain.

However, just understanding that you have security weaknesses is of no use unless you are able to do something about them. You need to have skilled resources that are able to interpret the results, remove the false positives and forward the resulting report to the systems administrators for action. Your system administrators must understand the importance of applying security patches and be capable of analysing the reported findings and comparing them to what is seen on the server at the time of the scan.

From a business perspective, you need to understand the risks that these vulnerabilities present you with as well as understanding the disruption that applying patches can entail. Remember, no business ever wants to take down their web presence for maintenance. However, if the alternative to a scheduled outage, that you control, is a total loss of service and data integrity, that will happen at a time that you have no control over (and inevitably will be at the worst possible time), you will no doubt conclude that the former is preferable to the latter.

Every time a vulnerability is discovered, it becomes somewhat of a race between the hackers and the system administrators as to whether or not the system administrators can patch the servers before the hackers (or their worms) break in. With the increase in worm activity over the last few years it becomes less a question of whether a vulnerability will be exercised, more a question of how soon will it be exercised and how much damage will result.

The inevitable fact is that without regular attention, your web presence will become vulnerable to attack. The decisions you need to consider is whether you have the ability to maintain your security posture with your current staff or whether you need to contract in help to assist you.

## *Glossary of Terms*

| | |
|---|---|
| Administrator | The highest level of authority in a Microsoft Windows system or a person that "administers" the system – see System Administrator |
| Deamons | Small programmes or utilities in Unix |
| Finger | A programme that announces details about a specific server |
| Firewall | A network system that controls ingress and egress of that network |
| Hacker | A malicious individual that attempts unauthorised access to computers |
| HP-UX | Hewlett Packard's variant of Unix |
| IP | Internet Protocol |
| IP Address | Internet Protocol Address usually shown as 4 sets of digits ie 10.123.234.235 |
| Modem | A device that allows computers to communicate over standard voice telephone lines. |
| Nessus | An open source programme that carries out Vulnerability Scanning |
| NSA | Network Services Auditor – an IBM programme that carries out Vulnerability Scanning |
| Ports | Elements of TCP or UDP |
| Root | The highest level of authority in a Unix system |
| Script Kiddie | A "hacker" that relies on automated tools and scripts developed by others in order to be able to launch attacks. Usually thought to be a young person with little skill in the area of computer hacking. |
| Sendmail | A programme for transmitting and receiving email |
| Saint | An open source programme that carries out Vulnerability Scanning |
| System Administrator | A person charged with the role of maintaining servers in a fit and proper state to carry out the role they were designed for. |
| Telnet | A program that allows the transfer of data between two systems |
| UDP | Universal Datagram Protocol |
| VSD | Vulnerability Scanning Device |
| Vulnerability | A weakness that can result in a machine being stopped, modified or taken over by an unknown 3rd party |
| Worm | A small piece of code that attacks specific vulnerabilities and then uses the access gained to replicate itself across the network |

## *References*


1. "*Cyberpunk: Outlaws and hackers on the computer frontier",* published by Simon & Schuster, (1991)
2. The spread of SQL Slammer - (http://www.caida.org/outreach/papers/2003/sapphire/sapphire.html)
3. Script Kiddie tools http://www.landfield.com/isn/mail-archive/2001/Feb/0131.html
4. Examples of Malicious URL's - http://www.theorygroup.com/Archive/Unisog/2002/msg00851.html
5. Example of a site offering Security Vulnerability notification service. http://securitytracker.com/server/info?1245+learn/endusers.html
6. Site that provides information on numbers of vulnerabilities released. http://www.securitystats.com/reports/Symantec-Internet_Security_Threat_Report_vIII.20030201.pdf
7. Example of a company providing policy compliance tools. www.symantec.com
8. Examples of companies providing Vulnerability Scanners. www.nessus.org & http://www.saintcorporation.com/products/saint_engine.html
9. Example of a patch released by Microsoft. http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/bulletin/ms03-007.asp)
10. Examples of Nessus reports http://www.nessus.org/report/10_163_155_6/index.html
11. SANS / FBI top 20 list of vulnerabilities http://www.sans.org/top20/
12. RFC 2518 http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc2518.html
13. Where to find port numbers defined http://www.iana.org/assignments/port-numbers

## *APPENDIX A*

### Network Overview Report from Nessus

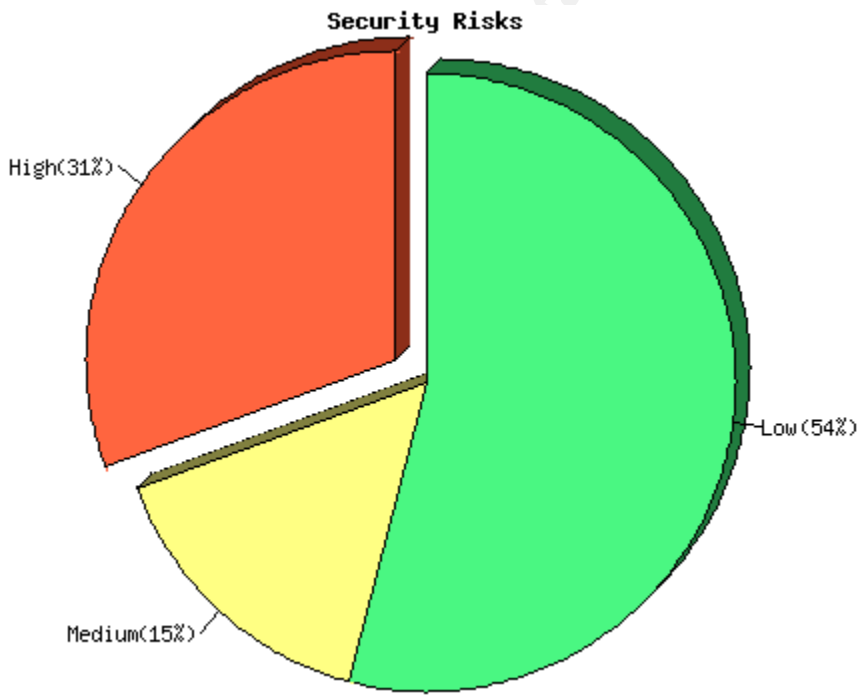
---

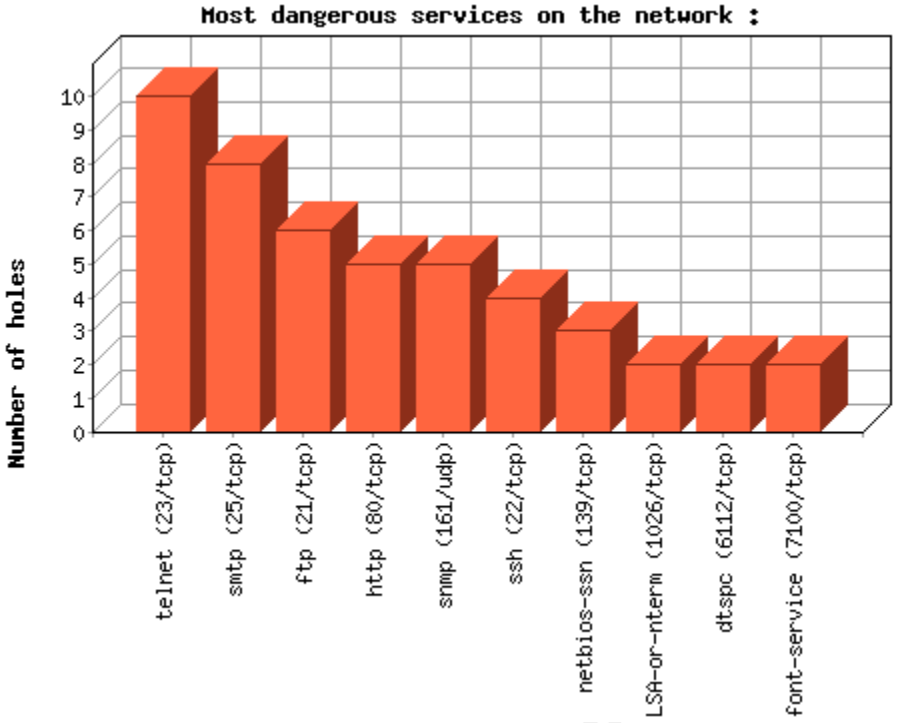
The Nessus Security Scanner was used to assess the security of 9 hosts

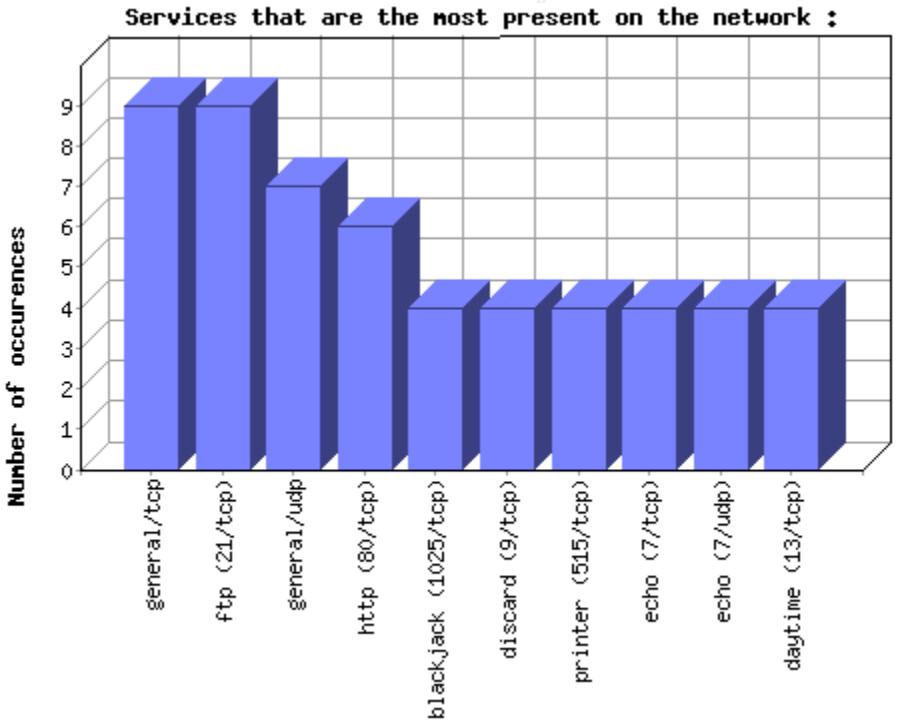
- **54 security holes have been found**
- **303 security warnings have been found**
- **113 security notes have been found**

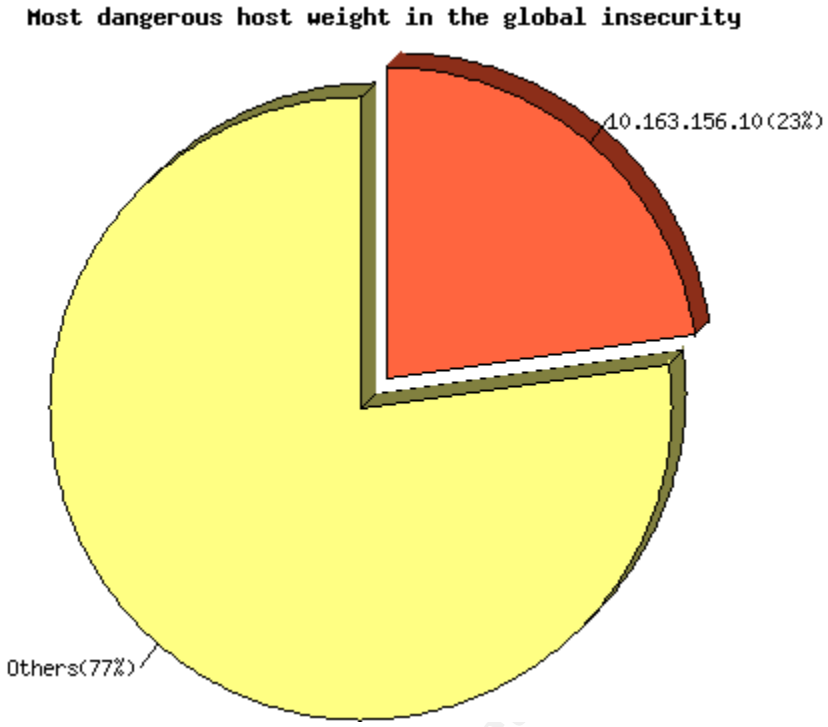
---

### Part I : Graphical Summary :








---

**Part II. Results, by host :**

| | |
|---|---|
| 10.163.156.10 | **(found 20 security holes)** |
| 10.163.156.9 | **(found 8 security holes)** |
| 10.163.155.4 | **(found 2 security holes)** |
| 10.163.155.3 | **(found 3 security holes)** |
| 10.163.155.2 | **(found 1 security hole)** |
| 10.163.156.1 | **(found 1 security hole)** |
| 10.163.155.6 | **(found 2 security holes)** |
| 10.163.156.205 | **(found 9 security holes)** |
| 10.163.156.16 | **(found 8 security holes)** |

---

*This file was generated by Nessus, the open-sourced security scanner.*

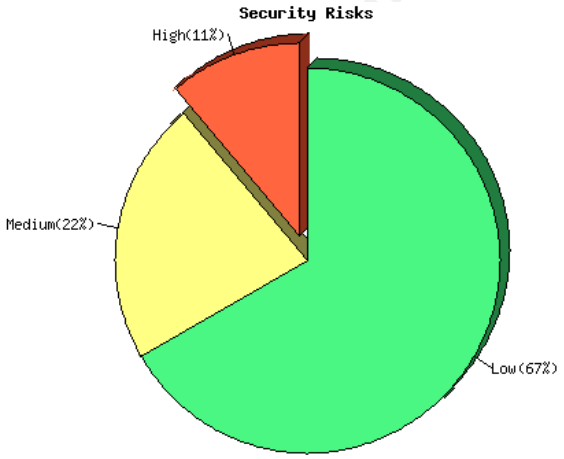
## *Appendix B*

By drilling down further Nessus will give you its analysis of what it found and some basic information as to why it believes this is an issue. I have abbreviated this report in the interests of clarity and brevity. If you wish to view the whole report go to http://www.nessus.org/report/10_163_155_6/index.html

**Detailed report of an individual machine generated by Nessus**

**Report for machine - 10.163.155.6**

Repartition of the level of the security problems:




---

[Next host : 10.163.156.205]

List of open ports :

- *http (80/tcp) (Security hole found)*
- *netbios-ssn (139/tcp) (Security hole found)*
- *loc-srv (135/tcp) (Security warnings found)*
- *blackjack (1025/tcp) (Security warnings found)*
- *unknown (1027/udp) (Security warnings found)*
- *ms-term-serv (3389/tcp) (Security warnings found)*
- *netbios-ns (137/udp) (Security notes found)*
- *general/udp (Security warnings found)*

- *general/tcp (Security warnings found)*
- *ftp (21/tcp) (Security warnings found)*
- *microsoft-ds (445/tcp)*

### Vulnerability found on port http (80/tcp)

It was possible to kill the web server by
sending an invalid request with a too long HTTP 1.1 header
(Accept-Encoding, Accept-Language, Accept-Range, Connection,
Expect, If-Match, If-None-Match, If-Range, If-Unmodified-Since,
Max-Forwards, TE, Host)

A cracker may exploit this vulnerability to make your web server
crash continually or even execute arbirtray code on your system.

Solution : upgrade your software or protect it with a filtering reverse
proxy
Risk factor : High

Nessus ID : 11129

[ back to the list of ports ]

### Warning found on port http (80/tcp)

This port was detected as being open by a port scanner but is now
closed.
This service might have been crashed by a port scanner or by some
information gathering plugin

Nessus ID : 10919

[ back to the list of ports ]

### Warning found on port http (80/tcp)

The remote web server type is :

squid/2.5.PRE13

Solution : We recommend that you configure (if possible) your web
server to return
a bogus Server header in order to not leak information.

Nessus ID : 10107

### Information found on port http (80/tcp)

The misconfigured proxy accepts requests coming
from anywhere. This allows attackers to gain some anonymity when browsing
some sensitive sites using your proxy, making the remote sites think that
the requests come from your network.

Solution: Reconfigure the remote proxy so that it only accepts requests coming
from inside your network.

Risk factor : Low/Medium

Nessus ID : 10195

[ back to the list of ports ]

**Information found on port http (80/tcp)**

Your webserver supports the TRACE and/or TRACK methods. It has been
shown that servers supporting this method are subject
to cross-site-scripting attacks, dubbed XST for
'Cross-Site-Tracing', when used in conjunction with
various weaknesses in browsers.

An attacker may use this flaw to trick your
legitimate web users to give him their
credentials.

Solution: Disable these methods.

If you are using Apache, add the following lines for each virtual
host in your configuration file :

```
RewriteEngine on
RewriteCond %{REQUEST_METHOD} ^(TRACE|TRACK)
RewriteRule .* - [F]
```

If you are using Microsoft IIS, use the URLScan tool to deny HTTP TRACE
requests or to permit only the methods needed to meet site requirements
and policy.

See http://www.whitehatsec.com/press_releases/WH-PR-20030120.pdf
http://archives.neohapsis.com/archives/vulnwatch/2003-q1/0035.html

Risk factor : Medium

Nessus ID : 11213

[ back to the list of ports ]

**Vulnerability found on port netbios-ssn (139/tcp)**

. It was possible to log into the remote host using a NULL session.
The concept of a NULL session is to provide a null username and
a null password, which grants the user the 'guest' access

To prevent null sessions, see MS KB Article Q143474 (NT 4.0) and
Q246261 (Windows 2000).
Note that this won't completely disable null sessions, but will
prevent them from connecting to IPC$
Please see http://msgs.securepoint.com/cgi-bin/get/nessus-0204/50/1.html

. All the smb tests will be done as "/" in domain WORKGROUP
CVE : CVE-2000-0222
BID : 990

Nessus ID : 10394

[ back to the list of ports ]

**Warning found on port netbios-ssn (139/tcp)**

The remote native lan manager is : Windows 2000 LAN Manager
The remote Operating System is : Windows 5.1
The remote SMB Domain Name is : WORKGROUP

Nessus ID : 10785

[ back to the list of ports ]

**Information found on port netbios-ssn (139/tcp)**

Here is the browse list of the remote host :

BENDER -
XP -

This is potentially dangerous as this may help the attack
of a potential hacker by giving him extra targets to check for

Solution : filter incoming traffic to this port
Risk factor : Low

Nessus ID : 10397

[ back to the list of ports ]

**Information found on port netbios-ssn (139/tcp)**

The host SID can be obtained remotely. Its value is :

XP : 5-21-583907252-2111687655-1957994488

An attacker can use it to obtain the list of the local users of this host
Solution : filter the ports 137 to 139 and 445
Risk factor : Low

CVE : CVE-2000-1200
BID : 959

Nessus ID : 10859

[ back to the list of ports ]

---

*This file was generated by Nessus, the open-sourced security scanner.*