Using Zonelabs Integrity Server and IPSec Policy to provide a Defense in depth Solution.

Peter L. Franchi. 9/17/2004

Giac 1.21.0 Practical
Gsec Version 1.4c Option 2

Table of Contents

Abstract

Within the environment of an educational establishment, there exist threats to the security infrastructure of the various departments.

These threats are from many possible sources and, as in any organization, are either deliberate or unintentional.

In our environment, the most inconvenient of these are the “Zombie” systems whose owners are often blissfully unaware of the duality of their machines. Activity of a repetitive nature makes these machines fairly easy to track down and disconnect from the network until the problem is resolved.

The more serious threats are the deliberate attacks from the outside world as well as those from within the boundaries of the organization.
The major problem facing many organizations is that no matter how good the perimeter defenses, the weakest point tends to be the endpoints; In this case, the workstations within the organization.
Therefore defense to protect from within the organization is just as important as defense from external attacks.

This paper is an attempt to detail the steps we utilized to add another layer of defense to our specific department and to provide a little information as to the types of firewalls that are available.

Types of Firewall

Firewalls can be classified by looking at the layer in the OSI stack in which they operate.

An OSI stack model is described at the link below

http://en.wikipedia.org/wiki/OSI_model

Packet Filters

Many filters operate at the network layer of the OSI Stack which is responsible for

- Routing
- Flow control
- Congestion control
- Network management
- Connections

These filters are referred to as Packet Filters and inspect the source and destination packet addresses, Protocol (UDP or TCP) and IP port numbers. This type of filter, though fast and fairly easily implemented in an existing infrastructure, can be fooled. Packet filters rely on destination ports. Certain protocols are expected to run on a default port but there is no reason why you can't specify another port. Opening up a port exposes the network to other traffic which can tunnel though this port. Packet filters are stateless and using the rules applied, decide to allow or block the packet. Decisions are made on the information in the headers with no regard to the content, either malevolent or beneficial.
For more information see: - IPSec for Windows 2000

http://www.microsoft.com/windows2000/technologies/communications/ipsec/default.asp

Stateful Packet Inspection

Stateful packet inspection can be considered as the next step up from packet filters. It too operates at the Network Layer. Not only does the filter examine headers of the packet, but also the contents in an effort to glean more information about the packet than just its source and destination. With Stateful inspection, the firewall takes a look at the content of the packet to see if it is HTTP. In addition, Stateful inspection firewalls also close off ports until connection to the specific port is requested. This allows an added layer of protection from the threat of port scanning

Proxy Firewall.

Proxy Firewalls operate at the Application layer of the OSI Stack, and are slow in nature but tend to provide the best security. Their major problem occurs when the proxy isn't available for a new protocol. They unpack each layer of each packet on the one interface examine it and rebuild it on the other interface. This, of course, adds to the overhead and slows delivery down. There is no actual direct connection between the two networks.

Application Firewall.

Application Firewalls, being stateful have the advantage to watch ports that have to remain open by monitoring the application using those ports.
For example, Zone Alarm Integrity Server can be deployed to, not only follow existing IPSec type rules to examine incoming packet sources and destinations but also to use a set of rules for actual applications . This makes the firewall very versatile for many possible products as it reports on applications that are used, giving the administrator choice to open the required ports or leave them closed. It is also possible to configure Zone Alarm integrity to prevent access to the network unless a certain level of patches and a recent level of virus protection have been applied to the new system. Zone Alarm can be configured to redirect the new system to a secure server to download the necessary updates and patches to prevent malware type applications from transmitting harmful data into the network. Zone Alarm Integrity client can be configured to protect the endpoint using a Stateful firewall that makes the system invisible to the outside world and allows client initiated traffic. Zone Alarm also contains an instant messenger control to encrypt IM traffic and block attempted intrusions, and finally, a filter that blocks potentially unsafe mail attachments.

The existing infrastructure

The original set of IPSec based firewall rules were not granular enough for precise firewall configuration , so our current departmental System Administrator (Mr. Siegfried "Ziggy" Hill) examined those rules and took a much less broad approach and the current premise was born resulting in a well planned single layer **IPSec** based firewall made up of many rules, but all based on the "***Deny All, allow by exception***" premise.

The overall IPSec policy is pretty comprehensive and encompasses several customizable modules, mainly because it has been built up to suit many possible departments.

The structure is made of a common set of rules to service the basic requirements of all the systems within the department. These rules allow ICMP within specific subnets, a DNS look up to the campus DNS servers, Kerberos Authentication within the required subnets, NetBIOS to specific subnets and of course, a "catch all" to clean up and drop all RAW, UDP, TCP and ICMP traffic other than those specified. This last rule is the "***Deny all***" part of the policy.

Click on the Firewall ports and protocols link off this link: -
http://filebox.vt.edu/users/zighill/rulesets/ to see a summary of the ports used.

The IPSecpol has been built up using groups of defined rules which permit access to specific machines and servers that exist within the campus. These various groups of rules have been modified over time and amalgamated into the current department policy,

The decision to install another layer of security was based on the possibility of the IPSec being circumvented by unmonitored use of open ports because IPsec is not stateful.

We looked at the possibilities open to us and because of the restrictions imposed by the various university authorities, we chose a software solution over a hardware solution. Of these, two were considered: Zone Alarm and Black Ice. At the time of decision, Black Ice had just been subject to a breach of security with the "witty" worm and we had just signed a new licensing agreement with Zonelabs.

Zone Alarm Integrity Server Installation

To enable Integrity server and clients to be used in conjunction with the existing IPSECPOL, various ports had to be opened , if they weren't already. Integrity uses ports 80 for http, 8443 for https and 5054 for the connections to the clients.
IIS service, if installed, needs to be disabled on the Windows 2000 server which integrity is being installed on.

Incorporating the IPSec rules into Zone Alarm is done in the Classic Rule set section of the Administrator Console and using our existing IPSec rules, are what I based the Classic rules on.

Many of the rules simply "plugged in" but one decision had to be made as to how to name the rule and the format to be used within Integrity.

After installing Microsoft SQL 2000 server which integrity requires for its databases, the 1st decision was which direction things flowed...
IPSec makes it pretty clear as to the "Source" and "Destinations" of data flow but within Integrity, a little more care is needed. I opted for the more conventional Left to Right as From and To respectively within Integrity

Policy Creation.

To create a Policy, Integrity has 3 levels that make up a "Rule".

Protocols

First is the Protocol section that determines the actual protocol being used and its source and destination ports. There are several pre-set port settings for HTTP, FTP, IGMP, ICMP etc but you have the option to specify your own port or port range or indeed, group of ports. You also can specify TCP, UDP or, if a rule requires a port on both protocols, then that can be specified too.

Sources

Next comes the Sources section in which you can create your own organizations' references to say, the Exchange server, DNS server, wins server and any other application specific servers.
However, at this point, along with the above protocol section, you have to be aware that for a single Integrity source, you may need a client source and destination, server source and destination as well as client and server ports.
So. if the Client opens one port to connect to a server, don't expect it to necessarily listen on the same port for the server's response!

Rules

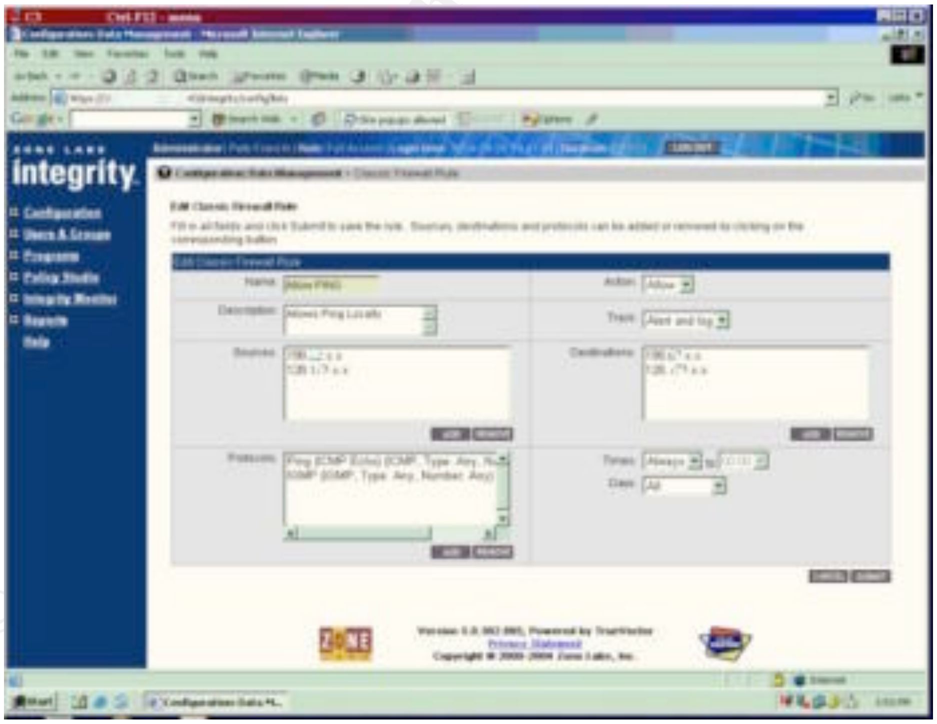
Finally comes the actual Rule section where the Integrity source and protocols are linked together.These rules can then be manipulated with the various other rules to determine it's order or preference. The final rule is the "Cleanup" rule which blocks all UDP/TCP traffic from and to all sources. All rules can be enabled and disabled from the main console, which is really useful when trying to diagnose any problems that may pop up.

When converting an IPSec rule, you have to look closely at its structure before entering it in Zone Alarm. Take, for example, the option to allow ping within certain subnets only within an organization..
The IPSec rule it written something like :-

Ipsecpol –w REG –p "Firewall Policy name " –r "ping_subnet" –f 128.x.*:+0: : ICMP –n PASS

Simplified , -w means write to the location (REG) Registry, with a –p (policy name of Firewall policy name) –r (Rule name of Ping _subnet) using the –f (filter list) pinging the 128.x.* subnet from +0 (the local machine) : : (on any port) ICMP (using the ICMP protocol) –n (using the negotiation policy list defaults) PASS (allow though the filter)

The same rule in Zone alarm, is somewhat more easy to understand when setting it up.



Edit Classic Firewall Rule

Name: Allow PING
Action: Allow
Description: Allows Ping Locally
Track: Alert and log
Sources: [illegible]
Destinations: [illegible]
Protocols: Ping (ICMP Echo) (ICMP, Type: Any, N
ICMP (ICMP, Type: Any, Number: Any)
Times: Always
Days: All

Fig 1: Allow Ping, Integrity Classic Rule.

The preceding picture starts off with the rule name, and whether its it allowed.
The next line has a description block for a more verbose indication on the scope of the rule followed by how the rule is logged. The source and destination blocks are on the next line together with the time of access (or denial) that the rule is in force.

The Same rule is shown below in XML format which can be exported from Integrity Server.

Fig 2: Allow Ping, Classic Rule. (x added in ip strings to protect actual information)

```
<rule name="Allow PING">
  <execute action="accept" log="logdb" alert="client" />
- <source>
  <ipsubnet address="198.x.0.0" mask="255.255.0.0" operation="maskeq" />
    </source>
- <destination>
  <ipsubnet address="198.x.0.0" mask="255.255.0.0" operation="maskeq" />
    </destination>
- <protocols>
  <icmpprotocol type="Any" description="Ping (ICMP Echo)" />
  <igmpprotocol type="Any" description="IGMP" />
    </protocols>
- <times>
  <daytimerange day1="all" />
    </times>
```

As can be seen, the rule structure is fairly self explanatory and pretty straight forward. Where it becomes more complex is when you have to have both a client and server sources within a rule. Careful consideration is required as to the level of granularity required for your rules. For example, you can include the Server and Client requirements all in one specific rule, however this means that in order to diagnose a problem, you can only affect both the client and server and not pinpoint down what exactly was the root cause of the problem.
Making the rules more granular obviously involves more initial work in setting up the policy from day one but it does have the advantage of being able to track down any potential problems with more accuracy.

Program rules over Classic Rules.

One specific problem we ran into was the fact that the “Classic firewall rules” seem to take priority over the “Program rules”
This was manifested with the addition of Retrospect tape back up software from Dantz and also Diskeeper Administrator console. With Retrospect and Diskeeper’s programs given full permission to the subnets we required in the Program rules section of integrity, they did not function correctly.

Retrospect uses port 497 TCP & UDP to communicate from the server to the client and from the client to the server, so in keeping with our choice for a fairly granular rule set, we have two rules as can be seen in the following XML format. One rule for the server communication, the other for the client return communication.

Within the organization there are two Retrospect backup servers, hence the duplicate IP address entries in each rule.
As can be seen, this rule, as are many others, is active 24/7 to permit overnight backups to function.

Fig 3: Retrospect Client and Server Classic rule in xml format.

```
<rule name="Retrospect Svr">
   <execute action="accept" />
 - <source>
   <ipaddress address="198.xx.xxx.xx" operation="eq" />
   <ipaddress address="198.xx.xxx.xx" operation="eq" />
      </source>
   <destination />
 - <protocols>
   <tcpudpprotocol protocol="IP_TCP_UDP" srcport="any" dstport="497"
      description="Retrospect Server" />
      </protocols>
 - <times>
   <daytimerange day1="all" />
      </times>
      </rule>
 - <rule name="Retro Client">
   <execute action="accept" />
   <source />
 - <destination>
   <ipaddress address="198.xx.xxx.xx" operation="eq" />
   <ipaddress address="198.xx.xxx.xx" operation="eq" />
      </destination>
 - <protocols>
   <tcpudpprotocol protocol="IP_TCP_UDP" srcport="497" dstport="any"
      description="Retrospect Client" />
      </protocols>
 - <times>
   <daytimerange day1="all" />
      </times>
      </rule>
```

Diskeeper was slightly different in its operation and there was an undocumented port required which prevented it from running correctly at first. Diskeeper Administrative console has the ability to push a client install onto a computer that's listed in the Active directory as long as an account with domain administrative rights is used. It also allows the Administrator to remotely operate the client and also to set a schedule for the client to operate. It uses RPC ports to permit these remote control sessions and uses DCOM for the push installations of the clients. What the documentation failed to inform us was that port 4201 TCP was also used. Fortunately with a quick use of netstat –a -n, we were able to see the syn_sent on that port.

The Program section of the Policy within integrity server, in our case, is little used as the Classic firewall rules take precedence. Basically, if the Classic firewall rules shut down all ports except those allowed by individual rules, then no matter what you tell the program section to permit, the non standard ports remain blocked. This process is a little inconvenient in some ways but relatively secure. This lock down should be in the final rule of the Classic firewall rules section and looks like the following

Fig 4: The Cleanup classic rule

```
- <rule name="Cleanup">
  <execute action="drop" log="logdb" alert="client" />
  <source />
  <destination />
- <protocols>
  <ipprotocol protocol="IP_EVERY" description="All Traffic" />
    </protocols>
- <times>
  <daytimerange day1="all" />
    </times>
    </rule>
```

In essence this final rule blocks all traffic from all protocols on every port and must appear at the bottom of the Integrity Policy list and not at the top.
The primary usage of the Program rules function in this specific installation would be to block any unauthorized programs from using a port other than those expected of it.
Example: Because port 497 is used and open for one application, for another application to use the same port. If the secondary application was a Trojan or virus, then with 497 open and no other rules in place, the firewall is potentially useless. This is one of the downfalls of using a Packet filter firewall only. If the Administrator adopts the same Deny All policy in the program section to deny every application except those listed, then that will successfully allow the firewall to function at the application layer of the OSI stack.

Here is an example of allowing Retrospect's client, access to the trusted network.
Fig 5:

```
<program path="retroclient.exe" action="add" checksum="897e2992-0c35848b-2d345f28-78beb70d" skimpChecksum="ec8644b3-df9fd740-17be07b4-4cee573c" allowTrusted="allow" allowInternet="Disallow" allowTrustedServer="allow" allowInternetServer="Disallow" passLock="yes" alertOnBlock="yes" pathNameOnly="no" moduleCheck="yes" privacy="no" SendMailPermission="allow" omp="false" />
```

Here the program has a checksum applied in order to confirm its authenticity and make sure another malevolent program is not masquerading as the genuine program.

Retrospect's central component uses a multicast addresses for its PITON name service to identify clients by name rather than IP address. Unless Retrospect was backing up a system over the internet, there is no reason for it to talk on 497 TCP to the internet. Setting the "allow internet server" option to disallow will prevent any other masquerading program from accessing the internet for example, Kazaa.

Preventing P2P

Kazaa is a Peer to peer file sharing utility that, if it cant use it own ports, will connect to other systems using port 80.
Closing port 80, is of course, not practicable as most web browsing uses this port.

In an effort to prevent utilities such as Kazaa the following can be used.

Fig 6:

```
program path="Kazaa.kpp" action="add" checksum="d9efb1d7-ed0676c7-5d545dc2-cae94bf0" skimpChecksum="b91279f5-2b8ed709-1638efcf-c7f95961" allowTrusted="Disallow" allowInternet="Disallow" allowTrustedServer="disallow" allowInternetServer="disallow" passLock="yes" alertOnBlock="yes" pathNameOnly="no" moduleCheck="yes" privacy="no" SendMailPermission="allow" omp="false" />
```

Here, Kazaa is blocked to the trusted as well as the internet from accessing servers or indeed acting as a server.

Instant Messengers

Due to the possible infiltration of viruses and Trojans via the various Instant messenger programs and file transfers, it is possible to allow the actual messaging, but block file transfers. The IPSec rules which were transferred over to the Classic Firewall rules section will permit port 80 based IM programs such as AIM anywhere and MSN messenger and their new web based utility but block all others. However, there is a section within Integrity server that will block the major IM programs functionality which we will be implementing as requirements dictate.

Client Deployment

The wish was to be able to remotely install Flex or Agent (see below) but we ran into a problem. We can remotely push an MSI but it did not contain the necessary XML configuration file and while it installed on the workstations successfully, the clients would not check to see if the server existed. If the client was FLEX, this wasn't too much of problem but more of an inconvenience because the end user had to teach the client to allow or block the various programs that required access to the outside world that were not in the standard programs list. If the client was AGENT, then the machine was rendered unable to connect to the outside world as the default policy is to deny every thing.

Our workaround was a little unusual in that when configuring Integrity server to provide a link to its sandboxed area for admins to download the appropriate package, Integrity creates a package which includes a correct config.xml file appropriate to either Flex or Agent. To correct the un-configured workstations, we had to unpack the correctly configured package on the server, extract the xml file and take it either on floppy or USB ram stick and manually configure the client with the following command line. ICLIENT.EXE –CONFIG (full path to config.xml or .ini in version 4.5.

Client Configuration.

There are 3 possible options that Zonelabs provide to Desktop, Flex and Agent"

Desktop:

This product is a stand alone product that can be installed on either a stand alone system or a networked system. This product will self configure for the standard Microsoft and other products that it knows about but will pop up with both outgoing and incoming requests for connectivity. This option is possibly the best option for the more security conscious user who needs to use a range of products that use a changing set of ports and protocols.
However, it is possible to permit what appears to be an innocuous request for access and open up a hole in the firewall.

Flex:

This client is useful for the security conscious laptop user. While out in the field, the Firewall acts like the stand alone product and asks for permission to allow or deny connections as required. Once the system is rejoined to the enterprise network, it detects the presence of an integrity server and changes its policy settings to use those of the server in favor of the local system policy. It also gives the user a degree of configuration with regard to cookies and privacy.

Agent:

This client is totally dependent on the server for its rules. Until it connects to the server, its in cautious mode and in essence, blocks all access to the outside

world. Once connected to the server, the policy is transferred and even when disconnected, the last received policy remains in effect.
We have chosen to use Agent on all but a select few workstations.

Zonealarm and the IP Stack

One interesting point we noticed with either the desktop or the flex agent installed was, when running in stand alone mode, if an unauthorized inbound attempt was made, and despite the port being blocked in IPSec, Zonealarm still popped up a warning about the attempt. This of course, peaked our interest and we initially started to check the IPSec firewall was actually running and that group policy was functioning.

Further research into this turned up the following statement made by Frederick Felman, VP or marketing for Zone Labs

> " Border firewalls of the kind found on many routers, furthermore, do not guard well against outgoing data because in order to have them access the Internet at all, "they have to have holes poked in them, such as opening port 80 for the browser." By comparison, "ZoneAlarm and Integrity sit on the network in the application layer and then in two places in the kernel, one just above the stack and one just below it." This is where the firewall lives. These positions help it determine the origin of the activity and the applications or ports through which the activity is occurring."

(From an article in processor.com)
http://www.processor.com/editorial/article.asp?article=articles%2Fp2528%2F09p28%2F09p28.asp&guid=&searchType=&WordList=&bJumpto=True

Caveat

Unfortunately, Integrity doesn't support Linux or UNIX or the Apple products, but within a Windows oriented environment, Integrity Server/Client is probably one of the most powerful, remotely-administrable products currently available.

After the Fact

Having installed server and client, it is just a case of deciding if the Base policy and ruleset is general enough for all of our users. What we expect to find over time is that certain applications will initially "break" and will require some investigation. The Base policy can be assigned on a machine by machine basis or on an IP address basis, making it possible to have slightly different policies for different groups of users within our departments.
As with any new piece of security software, there will be a learning curve associated with it and this will take a little time to sort out but with threats to data security and machine security increasing daily, the more levels of defense we can add, the more secure we will be.

References:-

From Winkipedia, the free encyclopedia, Alistair McMillan et al.
An OSI stack model Adapted from Federal Standard 1037C and previous Wikipedia content. Sept14 2004
http://en.wikipedia.org/wiki/OSI_model

Microsoft.com IPSec for Windows 2000 (Microsoft)
URL:
http://www.microsoft.com/windows2000/technologies/communications/ipsec/default.asp

Neil Randall. Zonelabs: Keep the Bad guys out. *The Processor, July 11, 2003 • Vol.25 Issue 28*
URL:
http://www.processor.com/editorial/article.asp?article=articles%2Fp2528%2F09p28%2F09p28.asp&guid=&searchType=&WordList=&bJumpto=True

Siegfried Hill, **Public Summary of Firewall ports and protocols Summary**
Version 1.208 Oct 2003
URL:
http://filebox.vt.edu/users/zighill/rulesets/Firewall%20Ports%20and%20Protocols%20Summary.pdf

Michael Howard, Defense in depth, Winnetmag, Feb 2001 URL:
http://www.winnetmag.com/WindowsSecurity/Article/ArticleID/16526/16526.html