Detecting Intrusions with your Firewall Log and OsHids

Daniel B. Cid – cidd@nhlbi.nih.gov
http://www.ossec.net

Introduction

A lot of articles have been published about IDSs, categories of IDSs (Network-Based, host-Based), IDS signatures and etc; but, what I have noticed is that people always forget the basic methods of intrusion detection. They think that the only way to detect an attack is using some tools like Snort, Portsentry or any other commercial IDS (the commercial term). Actually, any device or software that is able to detect an attack (or system misuse) can be called an IDS. If you have a little shell script that looks for anomalies on your log files, you have an IDS (not very complete, but it is one).

In this article we are going to talk about one of the basics, but powerful, methods of Intrusion Detection: Firewall's Log analysis. Although a firewall generates a lot of log, being difficult to analyze it, you can use the OsHids tool to monitor your logs (generating an easy to view log in html with an PHP interface) and help you visualize any attempt to bypass your firewall policy.

Why do I want to use a firewall as an IDS?

- All traffic that enters on your network passes through the firewall (or are supposed to), which means that you can really know what is entering into your network.
- Almost all kind of firewalls produce excellent logs, you only need to configure and find a way to see it.
- Sometimes, the IDSs (real IDSs) are located after the firewall, so you really lose a lot of information.
- You can easy discover new worms or virus, only looking for increase attempts to access different blocked ports (like 444, for example).
- You can add some kind of "string match" to look for a specific problem, making easy to solve a specific problem.

Remember that analyzing your firewall's log is only an additional option to improve your ways to detect intrusions. You can't replace your IDSs with your firewall. First, because the firewall only watch for packets passing through it; and second, because it will not detect any attempt to attack your allowed traffic (like an attack to port 80 of your web server).

Firewall Policy

The scope of this article is not to help you to create your firewall policy or to configure your firewall, but to show an easy method to visualize its log. To accomplish that, I will use an example of a simple firewall policy to help you understand how it works.

Simple Firewall Policy:

Allowed traffic:
From anywhere to 192.168.1.30 ports TCP 80,22,443
From anywhere to 192.168.1.31 ports TCP 25
From anywhere to 192.168.1.32 ports UDP 53
From 192.168.1.0/24 to anywhere
Denied traffic:
Everything else

As you can see, it is a very simple policy. We allow any internal traffic to outside and only allow external (new) traffic to our Internet servers.

Do you already have a firewall policy? If don't, you must start with that. What kind of traffic is allowed to reach your internal systems? What kind of traffic your internal users can access? Your firewall policy will cover all this questions.

Setting up your firewall

With this policy in mind, we are going to build our firewall rules. On our example here we are going to use *Iptables* as our firewall. You can use any kind of firewall you want, but on the actual version of OsHids (0.2), it will only support *Iptables* to generate the html logs correctly.

On version 0.2, it will only analyze a log as being from Iptables if it finds the string "Iptables" on the log and on the log type it is specified as iptables too. If you have any other firewall and add the string iptables together with the log, it will be analyzed; but I'm quite sure that it will not be correctly logged, because OsHids is set up to the iptables format (DST=, DPT=, etc). If you want to help and send me some lines of your firewall log, I can add support to it.

Before we continue, I want to remember that everything that are not allowed, is considered *possible dangerous*. Think with me, if anyone is trying to access port TCP 110 of your web server from the Internet, and this traffic is not allowed, probably your IDS will not complain about; but your firewall will. It can be a reconnaissance attack or someone scanning your hosts and looking for a vulnerable system. It's always good to have a complete vision about what is happening on the network.

 # Simple Iptables rules.
 # Do not use it on your real network; it is only for this example.
 # Pay attention to the “log level” and the “log-prefix” entries.
 # You will need to configure your syslog.conf to send this kind of log to your
 # log server (or to the correct place).
 # The log will be on the level kernel.info
 # OsHids will only analyze this log if it finds the entry: “Iptables” (with the i uppercase)
#
 # Cleaning all existent entries
iptables -t nat -F
iptables -F

 # Allowing internal traffic to the internet (nat) – 1.1.1.1 is the external IP addr.
iptables -t nat -A POSTROUTING -o eth0 -j SNAT --to-source 1.1.1.1

 # http traffic redirection
iptables -t nat -A PREROUTING -p TCP -i eth0 --dport 80 -j DNAT --to-destination 192.168.1.30
iptables -A FORWARD -i eth0 -o eth1 -p tcp --dport 80 -d 192.168.1.30 -j ACCEPT

 # https traffic redirection
iptables -t nat -A PREROUTING -p TCP -i eth0 --dport 443 -j DNAT --to-destination 192.168.1.30
iptables -A FORWARD -i eth0 -o eth1 -p tcp --dport 443 -d 192.168.1.30 -j ACCEPT

 # other redirect traffic stripped (only need to change the IP and port
 # …

 # Allowing forward traffic
iptables -A FORWARD -i eth1 -s 192.168.1.0/24 -j ACCEPT
iptables -A FORWARD -i eth0 -d 192.168.1.0/24 -m state --state ESTABLISHED,RELATED -j ACCEPT

 # Dropping forward traffic
iptables -A FORWARD -m tcp -p tcp -j LOG --log-level info --log-prefix "Iptables TCP denied "
iptables -A FORWARD -m tcp -p tcp -j DROP
iptables -A FORWARD -m udp -p udp -j LOG --log-level info --log-prefix "Iptables UDP denied "
iptables -A FORWARD -m udp -p udp -j DROP
iptables -A FORWARD -j DROP

 # Allowing and denying some traffic to the firewall
iptables -A INPUT -m tcp -p tcp --dport 22 -j ACCEPT
iptables -A OUTPUT -j ACCEPT
iptables -A INPUT -m tcp -p tcp -j LOG --log-level info --log-prefix "Iptables TCP denied "
iptables -A INPUT -m udp -p udp -j LOG --log-level info --log-prefix "Iptables UDP denied "
iptables -A INPUT -m tcp -p tcp -j DROP
iptables -A INPUT -m udp -p udp -j DROP

Above you can see our simple iptables rules. Some parts were stripped to avoid unnecessary information, but the most important part is in bold (about the log level and the log prefix).

Setting up syslog.conf (syslogd)

Now we need to set up the syslog.conf. In this example I'm going to send all kern.info logs (from the firewall) to our log server. On it, I will set the syslog to send all kern.info information to the file */var/log/iptables.log* and configure the OsHids to analyze the logs.

On the firewall I added the following line:

kern.info *@logserver*

And on the log server:

kern.info */var/log/iptables.log*

Configuring OsHids

After all other steps are completed; we can go to the log server and configure the OsHids.
Its configuration is very simple and you don't need to compile anything (all code is in Perl). I will subdivide OsHids installation for easier understanding.

**You must have perl (OsHids) and apache+php (php engine) installed on the log server.*

1. Download OsHids:
    - *Shell# wget http://www.ossec.net/oshids/files/oshids-0.2.tar.gz*

2. Untar/Ungzip it:
    - tar –zxvf oshids-0.2.tar.gz
    - cd oshids-0.2/

3. Move the *example 7* configuration dir to the default *config* directory:
    - mv ./config ./examples/orig-config
    - mv examples/ex7 ./config

4. Open the *./config/oshids.conf* and choose your log dir.
    - $logg[0]="/var/log/iptables.log";
    - $rulesfile="./conf/oshids.rules";
    - $mylogdir="/var/www/html";

On step four, you need to make sure that you did set up the syslogd to send all *kern.info* logs to the /var/log/iptables.log (or change it to whatever file you chose) and that the directory */var/www/html* is accessible by Apache (everything will be inside */var/www/html/fw/).*

[root@logserver oshids-0.2]# cat ./conf/oshids.rules
#log everything with the orange color
iptables:log-html::orange:: IPtables log

5. Copy the php engine to the specified www dir:
    - shell#mkdir /var/www/html/fw
    - shell#mv ./conf/php/* /var/www/html/fw/

6. Modify your /var/www/html/fw/conf-web.php:
    - $logdir="./";
    - $name="Firewall Policy";

In our case, the log dir will be the current directory, because we configured before the OsHids default dir to *"$mylogdir="/var/www/html";*", which means that all html logs will be stored on *"/var/www/html/fw/Month-Day-Year".*

7. Start the OsHids:
    - [root@logserver oshids-0.2]# perl os-hids.pl

And we are done with this part!

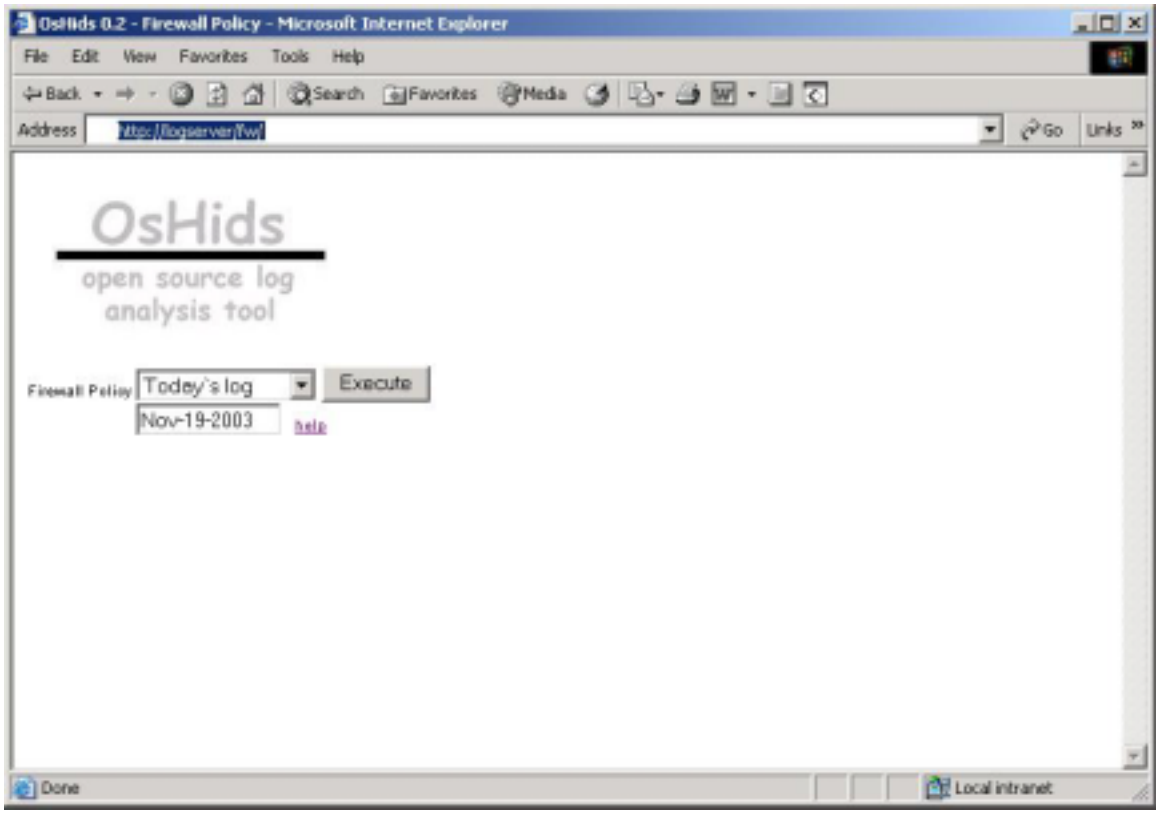
Monitoring your logs

With all the hard part done, we can move to the "easy" part (I'm kidding, monitor your logs will never be easy). You only need to point your browser to the IP address of your log server (or wherever you configured the OsHids) and to the directory that you choose. In our case, as we chose the directory */var/www/htdocs/fw/* (and the Apache Document Root was configured to /var/www/htdocs), we need to point our browser to http://logserver/fw/.

I want to remember you that our PHP engine don't have any kind of access control, so anyone will be able to look at it. To improve its security, at least follow these recommendations:

- *Use .htaccess to restrict any access to it.*
- *Configure a firewall or any other access control method to restrict access to the log server (only allow the administrator's IP address)*
- *Run it only on your private network (never let external machines access it)*
- *Chroot your Apache (will avoid problems)*
- *Enable all PHP security options (safe dir, etc)*

Look at our references for some texts about these topics.

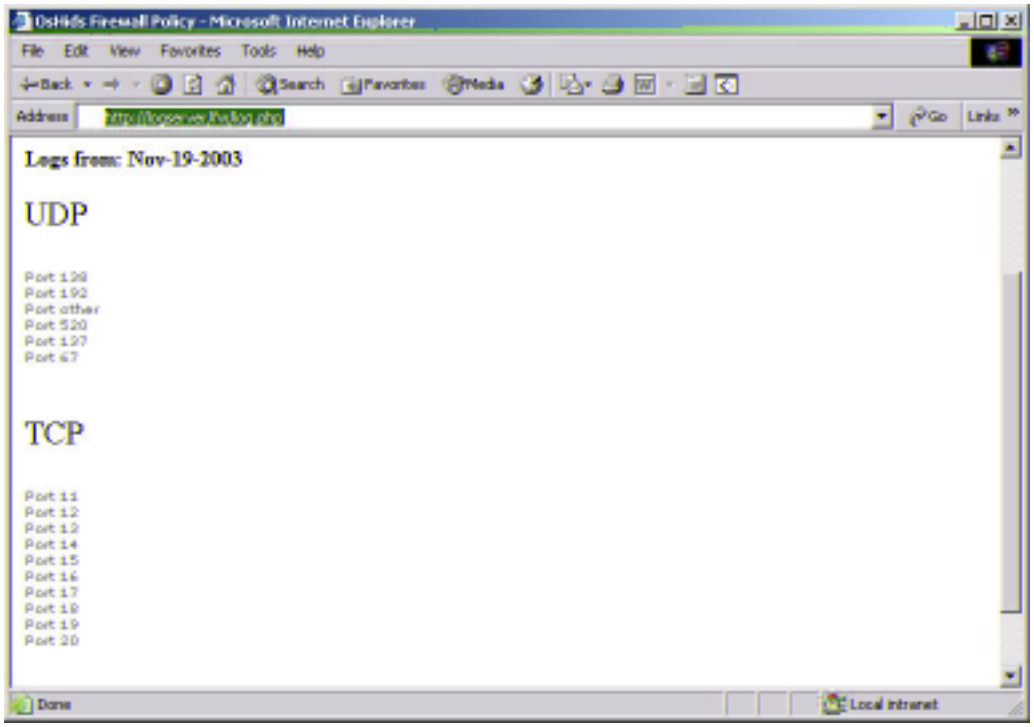


Pict 1 – Initial Page of the PHP engine

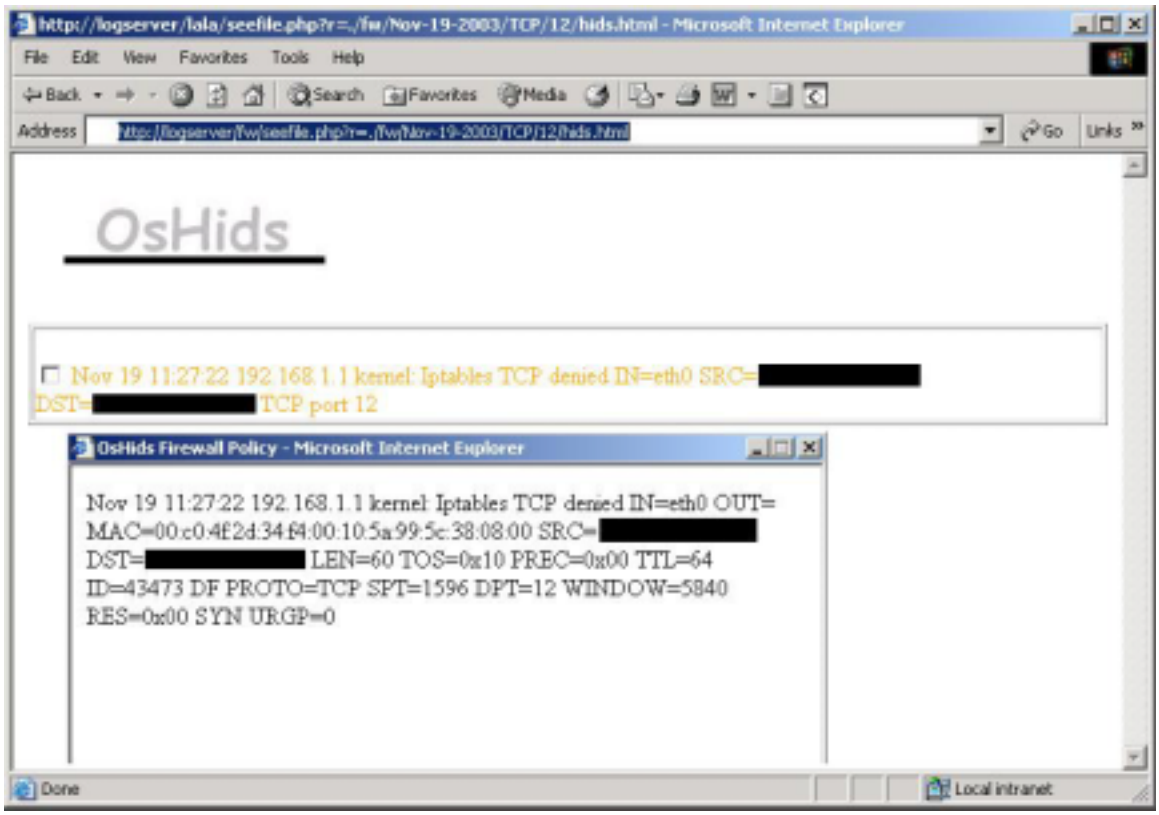
On the picture above (Pict 1) you can see the initial page of our PHP engine. You can choose to see “Today’s log”, “Yesterday’s log” or any other Day’s log you want (be sure to follow the correct format, as explained on the help page).

After you choose the day, a new page will appear with the logs divided by TCP and UDP (only for that day). On each one you can see if someone tried to access a denied port. Look at the picture bellow (Pict 2); it’s easy to see that someone probably did a port scan on you (port 10,11,12,13,14,15,16, etc logged). Another important thing to remember is that you can receive a lot of information from port UDP 67, 138, etc. You can tune your firewall to only log what is important. If you click on a specific port, you will see the page shown on *Pict 3*. There you can have more information about who tried to access this port, at what time, etc. If you need more information (TCP flags, MAC address, etc), you can click on the checkbox to view the complete packet information. Using this simple PHP engine (and OsHids) you can now monitor (on a daily basis) any attempt to bypass your firewall policy.

All ports above 1024 will be logged as “other”.



Pict 2 – Logs from a specific day



Pict 3 – Logs from a Specific Port with the box checked (for more info)

Summary

The initial purpose of this article was only to show how to configure OsHids to analyze Iptable's log, but I also added some other information about intrusion detection and Iptables (firewall policy) configuration to help the reader to understand the importance of log analysis and to make the article best understandable. Take a look at the references for more information.

References

http://www.ossec.net/oshids/ - OsHids Web site
http://sourceforge.net/projects/oshids/- OsHids on Sourceforge
http://www.loganalysis.org - Log Analysis Web site
http://www.securityfocus.com/infocus/1694 - Securing Apache: Step-by-Step
http://www.stearns.org/snort2iptables/ - Snort to Iptables
http://www.robertgraham.com/pubs/firewall-seen.html - FAQ: Firewall Forensics