

A Novice's Guide to Using Multiple Layers of Snort to Defend the Home Network

James McQuaid

Note: This article refers to the use of Honeywall Roo 1.0 (roo-1.0.hw-139.iso). An update will be posted soon.

At my home, I utilize a SANS-style layered defense consisting of three concentric ring perimeter firewalls.

In the outer ring, a firewall appliance is employed to filter out some inbound attacks and provide an initial layer of stateful packet inspection. I'm using a Netgear FVS-338, which is a Linux firewall (the source can be downloaded at Netgear's site). Outbound blocking is achieved by defining custom services which you set to Always Deny. To protect the FVS-338, you should set it block all ICMP on the inbound interface as well as, many service ports. It will block unrequested traffic on the inbound interface by default, but if a client machine becomes infected, malicious traffic will be solicited. Limit the Netgear's LAN IP range to the single IP address utilized by Smoothwall (unless you are using Honey Pots). We previously used the FVS-318 in the same configuration, and found that it should be set to reboot itself daily. The FVS-338 provides six to nine times the bandwidth, and allows you to configure a second LAN address. You may be able to use the CISCO PIX on the outer ring depending upon RAM and installed licensing. Our cable modem's power cord is plugged into an inexpensive, analog timer (commonly used for electric lamps). The timer turns off the cable modem during the night, effectively reducing our attack window by 30%.

The center ring of the perimeter uses Snort Inline in bridging mode (the two bridging NICs do not have IP addresses). The Honeywall ISO (<http://www.honeynet.org/tools/cdrom/roo/download.html>) will work nicely here, but if you have good UNIX or Linux skills, you will be rewarded with far greater flexibility in your configuration by compiling your own Snort Inline. In Honeywall you are required to use the web interface to upload rulesets, and some rules can't be used due to resultant parser errors in the web interface. The center ring is where the Bleeding rulesets (<http://www.bleedingthreats.net>) provide your home network with a substantive defensive capability against unknown threats. Many of the Bleeding rules can be set to drop packets in inline mode.

[image unavailable in .txt]

Honeywall's Snort Inline rules management

Honeywall does however provide several distinct advantageous features. In Honeywall, Snort rules are treated separately from Snort Inline rules. Snort Inline rules drop packets while Snort rules flag and log packets. p0f fingerprints the attacker's operating system. Argus monitors flow, and Sebek captures process information. The honeypot affords some granularity in its configuration. To avoid potential issues with your ISP, I suggest that you set the honeypots to use the outer ring's DNS proxy rather than those of the ISP's DNS servers.

[image unavailable in .txt]

In Honeywall, Snort flagged packets are captured for analysis

The Honeywall sample Snort Inline config file can be downloaded at docs.bleedingthreats.net. You will note that we are inspecting 100% of the packets (http_inspect_server's flow_depth is set to 0) and the inspection ports include all of the instant messenger client ports. Instant messaging clients have become mini-browsers, and we treat them as such. In Smoothwall, we had difficulty with 100% inspection, and are only inspecting the first 1460 bytes in each packet (however, everything inbound has already been completely inspected).

Instant messaging clients have become mini-browsers, and we treat them as such.

We are able to access Honeywall's web management interface using a 3rd NIC. This box should have a considerable amount of RAM: I'm using a dual Pentium III 1 GHz processor machine with 2 GB of RAM. It can use up that amount of RAM after a couple weeks, and you should reboot it weekly. To further reduce attack surface, you can write custom Snort rules to drop inbound traffic to specific destination port ranges. During the initial configuration, Honeywall asks if you wish to limit allowed ports out. The ports allowed out are determined by inner LAN client software requirements (these ports should match those unblocked in the outer ring). The /etc/blacklist.txt file is used to block hostile IP addresses. We have just over 4,000 IP ranges blocked (having accumulated addresses from the SANS ISC Top 10 Sources list, shadowserver.org, observations at work, home logs, etc.). With this many ranges dropped, you will want to sort your blacklist.

One of the great disadvantages of using Snort Inline in the center ring is that you do not get as many packet captures as when you run it on the outer ring. This may degrade your ability to detect reconnaissance against your network. Honeywall has excellent packet capture logging, making it a nice tool for those desiring to write Bleeding signatures for new threats. If you deploy Honeywall on the outer ring, in order to maintain the rich experience on the client machines (i.e. Gmail and Yahoo mail), you will need to either 1) open up many more allowed ports out, (undesirable) or 2) manually configure port translation (which can require some time).

The inner ring of our layered home perimeter defense uses Smoothwall Express 2.0 with Fixes 1-9 (<http://www.smoothwall.org>) and several Smoothwall Mods (<http://sourceforge.net/projects/smoothimods/>). The hardware tested is a Pentium 2.4 GHz with 1 MB of RAM. Snort does not drop packets here, it places alerts in a web accessible log file. Smoothwall will utilize all of the Bleeding Snort rulesets with the exception of the bleeding-botcc-BLOCK.rules and the bleeding-drop-BLOCK.rules, which require SnortSam. Smoothwall functions as an internal DNS server, so you can also utilize Bleeding's BlackHoleDNS project (http://doc.bleedingthreats.net/bin/view/Main/AllProjects#BlackHoleDNS_for_Spyware) with it. We have a short list of TLDs blocked along with 54,000 hostile domains in blackhole.conf. I wrote a perl script which removes duplicates and sorts the domains alphabetically. If you participate in the Spyware Listening Post project (http://doc.bleedingthreats.net/bin/view/Main/AllProjects#Spyware_Listening_Post), you can have your blackhole.conf file updated automatically. In Smoothwall's Squid ACL files, you should set safe ports equal to those configured in Honeywall's allowed ports out list. Limit Smoothwall's client-facing IP range to the number of client machines and wireless devices you have. The Bleeding rulesets provide early warning of an infected client machine (frequently before the anti-virus vendors have developed the relevant signatures). You can run Snort on Smoothwall in an ultra-sensitive configuration on the inner ring because the Netgear appliance and Honeywall will have already dropped the inbound bad packets. Although you will see Google's web bugs (1 pixel gif files) and other traffic in the IDS logs, Smoothwall's firewall logs should usually be empty. A word of caution, if either the outer or center ring fails, you will need to quickly reconfigure Snort, or Smoothwall will very rapidly exhaust its available memory and be subject to attack. Configure Smoothwall's client machine-facing NIC with a different subnet than that facing the Netgear.

Networking tip: place switches between Honeywall and the inner and outer rings

Behind the inner ring the demographics of the machine population are half Windows and half Linux (Ubuntu and SE Fedora). All of them are limited to resolving DNS to the Smoothwall DNS server. The XP machines have multiple security apps installed to compensate for the OS. These include Avira Antivir PersonalEdition Classic, for protection against zero day viruses, and avast! Professional antivirus both are installed on each XP. Most avast users don't properly use the software's web scanning feature: this requires that browser proxies be set to 127.0.0.1 on port 12080. F-Prot antivirus and the CoreForce firewall (a port of OpenBSD's PF firewall) are installed on a Windows 2000 server. We have not been able to run Antivir and F-Prot on the same box. Sunbelt's Counterspy and Lavasoft's Ad-Aware Plus provide the XP machines a measure of realtime protection, and allow scheduled anti-spyware scanning. ZoneLabs' IMSecure Pro allows somewhat safer instant messaging configurations; it allows content type filtering and prevents address, telephone, and other personal data outbound via IM (parents should consider installing it). Sunbelt's Kerio Personal Firewall (driver 4.3.142) provides Snort on the XP desktops. Kerio Personal Firewall will accept the Bleeding rulesets (in the rlk files in C:\Program Files\Sunbelt Software\Personal Firewall\Config\IDSRules). Kerio drops offending packets when a Bleeding rule is tripped. Web pages will render even as packets are dropped. Use Kerio's packet filtering to prevent client machines from direct communication with the IP address of the device in the outer ring. You will need client machines with 64-bit processors and a minimum of 1 GB of RAM to run all of this software on Windows XP. Each security software package is scheduled to run a deep scan at 24 hour intervals. These scans can take hours, must not overlap, or occur when the machines are in use. The Linux machines require less RAM and perform well with 32-bit processors.

At our home, we've been able to substantially reduce malware infections and intrusions using this multiple layers of snort topology. My older teens can more safely instant message, watch streaming video, play games online and so forth; in contrast, Microsoft's ISA Server does not permit this much functionality. A few years ago I was using a double NIC Microsoft Small Business Server 2003 (with RRAS) in tandem with a perimeter firewall appliance, it was impossible to keep worms from traversing SBS's multitude of shares. SBS also had compatibility problems with both Sunbelt's Kerio Personal Firewall and Agnitum's Outpost firewall. I had read about Bleeding's BlackHoleDNS project, and when I couldn't get it to function in SBS, I reloaded that box with Smoothwall. This is not a set and forget network, you must regularly review the logs from all three rings as well as, your client machines, and then take appropriate action.

Your ability to use Smoothwall in an ultra-sensitized configuration will be affected by your network topology as well as, bandwidth usage. If you are configuring snort for use in a home network or small office, you can operate with higher sensitivity than you could in a production environment or on the perimeter of a large organization. In a larger organization, you will benefit from extensive segmentation of your network using snort inline, and by employing Bleeding's BlackHoleDNS project. Depending upon your domain's requirements, you may be able to use Smoothwall as well. Whether in a large or small environment, there are several snort.conf preprocessor settings that you will need to fine tune.

Because its primary purpose is early warning of a problem on the LAN, you want Snort on Smoothwall (in the inner ring) to use as much of the available RAM as possible without risking memory exhaustion, performance problems or instability.

Snort Pre-Processor Configuration:

Below elements of the default configuration are contrasted with that we're using in Smoothwall. The Honeywall sample Snort Inline config file can be downloaded at docs.bleedingthreats.net.

Preprocessor: flow

Purpose: the Flow tracking module is meant to start unifying the state keeping mechanisms of Snort.

Our Smoothwall config: preprocessor flow: memcap 100663296, rows 8198, stats_interval 0 hash 2

Default config: preprocessor flow: stats_interval 0 hash 2

memcap: the number of bytes to allocate

Smoothwall Caveats: the number of rows in the hash table can be increased to enhance performance; increases will require additional RAM.

Preprocessor: Frag 3

Purpose: the frag3 preprocessor is a target-based IP defragmentation module.

Frag 3 Global Configuration:

Our Smoothwall config: preprocessor frag3_global: memcap 67108864, max_frag 131072

Default config: preprocessor frag3_global: prealloc_nodes 8192

Frag 3 Engine Configuration:

Our Smoothwall config: `preprocessor frag3_engine: policy linux detect_anomalies bind_to (outer perimeter CIDR)`

Our Smoothwall config: `preprocessor frag3_engine: policy first detect_anomalies bind_to (inner perimeter CIDR)`

Our Smoothwall config: `preprocessor frag3_engine: policy last detect_anomalies`

Default config: `preprocessor frag3_engine: policy first detect_anomalies`

`memcap`: memory cap for self-preservation (default is 4 MB).

`max_fragments`: maximum simultaneous fragments to track (default is 8192).

Smoothwall caveats: Be certain to check `/var/smoothwall/messages/` to confirm the actual configuration after restarting Snort. Smoothwall will ignore your `snort.conf` file in certain circumstances.

Preprocessor: Stream 4

Purpose: Stream4 provides TCP stream reassembly and stateful analysis capabilities.

Our Smoothwall config: `preprocessor stream4: detect_scans, detect_state_problems, disable_evasion_alerts, state_protection, memcap 33608864`

Default config: `preprocessor stream4: detect_scans, disable_evasion_alerts, memcap 8388608`

Stream4 Options:

preprocessor stream4 default settings:

session timeout (timeout) 30 seconds

session memory cap (memcap) 8388608 bytes

stateful inspection (noinspect) active (noinspect disabled)
stream stats (keepstats) inactive
state problem alerts (detect_state_problems) inactive (detect_state_problems disabled)
evasion alerts (disable_evasion_alerts) inactive (disable_evasion_alerts enabled)
asynchronous link (asynchronous_link) inactive
log flushed streams (log_flushed_streams) inactive
max TCP sessions (max_sessions) 8192
session cache purge (cache_clean_sessions) 5
self preservation threshold (self_preservation_threshold) 50 sessions/sec
self preservation period (self_preservation_period) 90 seconds
suspend threshold (suspend_threshold) 200 sessions/sec
suspend period (suspend_period) 30 seconds
state protection (state_protection) inactive
server inspect limit (server_inspect_limit) -1 (inactive)
UDP session tracking (enable_udp_sessions) inactive
max UDP sessions (max_udp_sessions) 8192

stream4_reassemble Configuration:

Our Smoothwall config: preprocessor stream4_reassemble: both, favor_new, ports: all,
emergency_ports 21 23 25 42 53 80 110 111 135 136 137 139 143 222 445 513 1433 1521 3306

Default config: preprocessor stream4_reassemble: client

stream4_reassemble Options:

preprocessor stream4_reassemble default settings:

reassemble client (clientonly) active
reassemble server (serveronly) inactive

reassemble both (both) inactive

reassemble ports (ports) 21 23 25 42 53 80 110 111 135 136 137 139 143 445 513 1433 1521 3306

emergency reassemble ports (ports) 21 23 25 42 53 80 110 111 135 136 137 139 143 445 513 1433 1521 3306

reassemble alerts (noalerts) active (noalerts disabled)

favor old packet (favor_old) active

favor new packet (favor_new) inactive

flush on alert (flush_on_alert) inactive

overlap limit (overlap_limit) -1 (inactive)

large packet performance (large_packet_performance) inactive

Smoothwall caveats: If you configure stream 4 to be overly sensitive, you can seriously flood your Snort logs. If you under configure it, you won't see events that you might want to be aware of. Over time, you will want to experiment with most of the configuration options.

Preprocessor: sfportscan

Purpose: The sfPortscan module, developed by Sourcefire, is designed to detect the first phase in a network attack

```
Smoothwall config:  preprocessor sfportscan: proto { all } \
                    scan_type { all } \
                    memcap { 67108864 } \
                    sense_level { high } watch_ip { xxx.xxx.x.x, xxx.xxx.x.x }
```

Default config: preprocessor sfportscan: proto { all } \

scan_type { all } \

sense_level { low }

sense_level available options:

Low: Low alerts are only generated on error packets sent from the target host, and because of the nature of error responses, this setting should see very few false positives. However, this setting will never trigger a Filtered Scan alert because of a lack of error responses. This setting is based on a static time window of 60 seconds, after which this window is reset.

Medium: Medium alerts track connection counts, and so will generate filtered scan alerts. This setting may false positive on active hosts (NATs, proxies, DNS caches, etc), so the user may need to deploy the use of Ignore directives to properly tune this directive.

High: High alerts continuously track hosts on a network using a time window to evaluate portscan statistics for that host. A "High" setting will catch some slow scans because of the continuous monitoring, but is very sensitive to active hosts. This will require you to tune sfPortscan.

watch_ip:

Defines which IPs, networks, and specific ports on those hosts to watch. The list is a comma separated list of IP addresses and/or an IP address using CIDR notation. Optionally, ports are specified after the IP address/CIDR using a colon and can be either a single port or a range denoted by a dash. IPs or networks not falling into this range are ignored if this option is used.

Smoothwall caveats: Filtered port scan alert types are more likely than other alert types to be false positives. Yahoo and Google will generate false positives; take notice of other alerts. Generally, the alerts from sfportscan will require that you regularly do lookups of IP addresses which appear in your logs.

Preprocessor: `preprocessor http_inspect`

Smoothwall default: `preprocessor http_inspect: global iis_unicode_map
/var/smoothwall/snort/unicode.map 1252`

Our Smoothwall: `preprocessor http_inspect: global detect_anomalous_servers iis_unicode_map
/var/smoothwall/snort/unicode.map 1252`

`http_inspect_server` Options:

Smoothwall default: `preprocessor http_inspect_server: server default profile all ports { 80 }`

Our Smoothwall: `preprocessor http_inspect_server: server $HOME_NET profile all ports { 80 1863 3128
5050 5061 5190 5191 5192 5193 5222 5223 6891 8080 8180 13324 13325 32771 56885 }
oversize_dir_length 300 flow_depth 1460`

Our Smoothwall: `preprocessor http_inspect_server: server default profile all ports { 80 1863 3128 5050
5061 5190 5191 5192 5193 5222 5223 6891 8080 8180 13324 13325 32771 56885 } oversize_dir_length
300 flow_depth 1460`

Option: `oversize_dir_length`

This option takes a non-zero positive integer as an argument. The argument specifies the max char directory length for URL directory. If a URL directory is larger than this integer, an alert is generated.

Option: `ports`

This is how the user configures which ports to decode on the HTTP server. Encrypted traffic (SSL) cannot be decoded, so adding port 443 will only yield encoding false positives.

Option: flow_depth

Purpose: this specifies the amount of server response payload to inspect. Specifying a flow_depth of 300 (the default) increases throughput speed because it ignores a large part of the network traffic (HTTP server response payloads). The HTTP headers of friendly packets are usually under 300 bytes in length.

The flow_depth value can be set from -1 to 1460. A value of -1 causes Snort to ignore all server side traffic for ports defined in ports. A value of 0 causes Snort to inspect all HTTP server payloads defined in ports (this will slow down throughput speed). Values above 0 tell Snort the number of bytes to inspect in the first packet of the server response.