The Snort FAQ

The Snort Core Team

Suggestions for enhancements of this document are always welcome. Please email them to the Snort-Users mailing list.

Many people have contributed to this FAQ:

Marty Roesch	Fyodor Yarochkin	Dragos Ruiu	Jed Pickel
Max Vision	Michael Davis	Joe McAlerney	Joe Stewart
Erek Adams	Roman Danyliw	Christopher Cramer	Frank Knobbe
Phil Wood	Toby Kohlenberg	Ramin Alidousti	Jim Hankins
Dennis Hollingworth	Paul Howell	Stef Mit	Ofir Arkin
Jason Haar	Blake Frantz	Lars Norman S?ndergaard	Brent Erickson
Brian Caswell	Scot Wiedenfeld	Chris Green	Jeff Wirth
Edin Dizdarevic	Detmar Liesen	Don Ng	Matt Kettler
Joe Lyman	Jim Burwell	Jed Haile	Andrew Hutchinson
Jeff Nathan	Alberto Gonzalez	Jason Haar	Jeremy Hewlett

If you do not see your name on this list and you have contributed to the faq, please email bmc@snort.org.

Dragos Ruiu: This version of this guide has been brought to you by the kind generosity and sponsorship of Wiley and Sons publishers whose support let myself, and other snort developers Jeff Nathan and Jed Haile take the time to work on this document and other tutorials for Snort due out in our upcoming book. (route++)

Contents

1 Background

1.1 How do you pronounce the names of some of these guys who work on Snort?

For the record, 'Roesch' is pronounced like 'fresh' without the 'f.' Additionally, 'Ruiu' is pronounced like 'screw you' without the 'sc.' Jed's last name is like 'pick-el,' not 'pickle.'

1.2 Is Fyodor Yarochkin the same Fyodor who wrote nmap?

Nope. fyodor@insecure.org is the author of nmap, and he uses the same pseudonym as the other Snort Fyodor's real surname. Yeah, it messes up my mailbox too, but I think it's too late to change either of them.

1.3 Where do I get more help on Snort?

Check the website, http://www.snort.org/. Other good resources are available in the source distribution, including the Snort Users Manual and the USAGE file. There is also a excellent mailing list, snort-users. You can find info on how to signup at http://www.snort.org/lists.html. You can also join #snort on irc.freenode.net.

1.4 Where can I get more reading and courses about IDS?

All of the following offer courses on Intrusion Detection:

- SANS (http://www.sans.org)
- Usenix (http://www.usenix.org/event/)
- Networld/Interop (http://www.key3media.com/interop/)
- CanSecWest (http://www.cansecwest.com)

There are many good books on Intrusion Detection. Here are just a few:

Title	**Author(s)**	**Publisher**	**ISBN**
Snort: The Complete Guide to Intrusion Detection	Jeff Nathan, Dragos Ruiu, & Jed Haile	Wiley & Sons	0471455970
Intrusion Detection with Snort: Advanced IDS Techniques	Rafeeq Rehman	Prentice Hall	I0131407333
Snort Intrusion Detection	Ryan Russell	Syngress Media	1931836744
Snort Intrusion Detection	Jack Koziol	New Riders	157870281X
Network Intrusion Detection: An Analyst's Handbook	Stephen Northcutt	New Riders	0735708681
Intrusion Signatures and Analysis	Stephen Northcutt	New Riders	0735710635
TCP/IP Illustrated, Volume 1 The Protocols	W. Richard Stevens	Addison-Wesley	0201633469
Intrusion Detection	Rebecca G. Bace	MacMillan Technical Publishing	1578701856

1.5 Does Snort handle IP defragmentation?

Yes, use `preprocessor frag2`.

1.6 Does Snort perform TCP stream reassembly?

Yes, check out the stream4 preprocessor (see FAQ 3.18) that does stateful analysis session login, TCP reassembly and much, much more.

1.7 Does Snort perform stateful protocol analysis?

Yes. Stream4 does this as well. See (see FAQ 3.18).

1.8 I'm on a switched network, can I still use Snort?

Short version:

Being able to sniff on a switched network depends on what type of switch is being used. If the switch can mirror traffic, then set the switch to mirror all traffic to the Snort machine's port.

Extended version:

There are several ways of deploying NIDS in switched environments which all have their pros and cons. Which method applies to your needs depends on what kind of segments you want to monitor and on your budget. Here are the most common methods:

1. **Switch mirror:** If the switch can mirror traffic, then set the switch to mirror all traffic to the Snort machine's port.
   - Advantages:
     - Simple method, works with most decent switches.
   - Drawbacks:
     - If the switch is a fast Ethernet switch, you can mirror 100Mbit/s max. Since each switch port is capable of handling 100Mbit/s for each direction, the bandwidth per port sums up to 200Mbit/s, so the switch will not be able to mirror all packets at high network utilization.
     - Some switches suffer from performance degradation through port mirroring.
2. **Hub:** Insert a hub in line, so you can simply tap all traffic. Works fine for home networks, will lose data due to collisions at loads greater than 50%—so a 10Mbps hub should be fine for T1/E1, DSL or cablemodem. If you have a DS3 or greater, you should investigate taps.
   - Advantages:
     - Simple method
     - No impact on switch performance and no config changes
     - Low cost
   - Drawbacks:

  - Loss of full-duplex capabilities
  - Additional single point of failure
  - Collision loss at above 50% load levels

3. **Network taps:** Use network taps (e.g. Shomiti/Finisar [http://www.shomiti.com] and Netoptics [http://www.netoptics.com). You can find some rather good information in the papers by Jeff Nathan. You can find the papers at http://www.snort.org/docs/#deploy.
   - Advantages:
     - No impact on switch performance and no special configuration
     - Stealth—i.e., sending data back to the switch is disabled
     - No single point of failure, "fail-open" if the tap power fails
   - Drawbacks:
     - The datastream is split into TX and RX, so you need two NICs
     - The two datastreams have to be recombined, i.e. merged, if you don't want to lose the capability of doing stateful analysis. This can be done by using channel bonding. Information can be found at http://sourceforge.net/projects/bonding.
     - Cost
4. **Throw money at it:** Tap switch ports (using the forementioned network taps) but only tap all incoming packets (RX lines of the switch ports), connecting those tap ports to a dedicated gigabit switch, which is capable of mirroring up to ten RX taplines to one single dedicated gigabit port, which is connected to a gigabit IDS machine.
   - Advantages:
     - Maximum coverage (i.e. monitor all switchports)
     - No performance degradation or re-configuration of the switch
   - Drawbacks:
     - Mucho $$$

1.9 Is Snort vulnerable to IDS noise generators like "Stick" and "Snot"?

It is now possible to defeat these kinds of noise generators with the stream4 preprocessor (see (see FAQ 3.18)). Even without the stream4 preprocessor enabled, Snort will weather the alert storm without falling over or losing a lot of alerts due to its highly optimized nature. Using tools that generate huge amounts of alerts will warn a good analyst that someone is trying to sneak by their defenses.

1.10 Can Snort be evaded by the use of polymorphic mutators on shellcode?

Yes, and this could defeat some of the NOP sled detection signatures, but the ordinary exploit rules should not be affected by this kind of obfuscation. The fnord preprocessor attempts to detect polymorphic shellcode attempts.

1.11 Does Snort log the full packets when it generates alerts?

Yes, the packets should be in the directory that has the same IP address as the source host of the packet which generated the alert. If you are using binary logging, there will be a packet capture file (.pcap) in the logging directory instead.

2 Getting Started

2.1 Where do I find binary packages for BlueHat BSD-Linux-RT?

Repeat after me:

```
    wget http://www.snort.org/downloads/snort-stable.tgz
    tar zxvf snort-stable.tgz
    cd snort-stable
    ./configure
    make
    su
    make install
    mkdir /var/log/snort
    cd etc
    vi snort.conf
    snort -D -c snort.conf
    exit
```

...and if you want to use our binary package uninstaller :-):

```
    cd snort-stable; make uninstall
```

And if you must, you can find some binaries at http://www.snort.org/dl/binaries/. You can also find Snort in most BSD ports' trees.

2.2 How do I run Snort?

Run Snort in sniffer mode and make sure it can see the packets.

```
snort -dv{
```

Then run it with the HOME_NET set appropriately for the network you're defending in your rules file. A default rules file comes with the snort distribution and is called "snort.conf" You can run this basic ruleset with the following command line:

```
snort -A full -c snort.conf
```

If it's all set right, make sure the interface is in promiscuous mode by running the command from another window:

```
ifconfig -a
```

The output from ifconfig should show if the interface is in promiscuous mode. If it's not, there should be a way to set it manually.

Note that the default output mode (-A full) of Snort should not be used except in very controlled environments. It is the slowest way to run Snort and presents several hard to recover from problems with inode creation on filesystems.

For people doing real IDS work, use something like (-A fast -b) to combine fast alert mode with tcpdump binary log files or use the unified format coupled with barnyard.

2.3 Where are my log files located? What are they named?

The default location for logs is /var/log/snort. If snort is started with "-l <directory>", then the logs will be located in the directory specified.

In the past, running Snort in daemon mode (-D) produced a file named "snort.alert." For consistency's sake, this has been changed. Running Snort in both standard or daemon modes (-D) will produce a file named "alert."

Note the log file naming convention changed between 1.8 and 1.9. That funny alphanumeric soup at the end of the new names is a UNIX timestamp. This helps avoid file conflicts.

2.4 Why does Snort complain about /var/log/snort?

It requires this directory to log alerts to it. Try running the command:

```
    mkdir -p /var/log/snort
```

Make sure the logging directory is owned by the user Snort is running as.

2.5 Where's a good place to physically put a Snort sensor?

This is going to be heavily influenced by your organizations policy, and what you want to detect. One way of looking at it is determining if you want to place it inside or outside your firewall. Placing an IDS outside of your firewall will allow you monitor all attacks directed at your network, regardless of whether or not they are stopped at the firewall. This almost certainly means that the IDS will pick up on more events than an IDS inside the firewall, and hence more logs will be generated. Place an IDS inside your firewall if you are only interested in monitoring traffic that your firewall let pass. If resources permit, it may be best to place one IDS outside and one IDS inside of your firewall. This way you can watch for everything directed at your network, and anything that made it's way in.

ADDENDA AD NAUSEUM

Note: So this one still gets a lot of traffic even though it's in the FAQ. Erek Adams has noted this comprehensive and authoritative discussion of this perpetual discussion item—mildly edited, also see faq question about switches hubs and taps -dr

If your router/switch can do port mirroring, then just connecting a network IDS to it would be fine. Or else a hub could be another option. Most network IDSes can have a NIC that acts as a passive sniffer anyway.

As to where to place the sensor. I would go for both, one to monitor the external, one for the internal. I work in a distributor for security products, so over instrumentation is fun :) And in any case, if the traffic does not pass by the Sensor it will not get monitored. So some people deploy IDS on their internal segments too, I believe.

In "front" of the firewall(s):

Pro: Higher state of alert you know what attacks you are facing.

Con: Wall to Wall of data, boring? If your firewall has NAT turned on, tracking the sources originating from your internal network is difficult.

"Behind" the firewall(s):

Pro: Only what gets through the firewall gets monitored? Less load on the IDS analyst. You get to see what hosts are sending traffic to the internet.

Con: Less idea of the state of the environment, false sense of safety.

Where should IDS be placed relative to firewalls? Explore the pros and cons of placing IDS inside or outside firewall. What are the drawbacks of each?

- **MARCUS RANUM from NFR Security:** "I'd put mine inside. Why should I care if someone is attacking the outside of my firewall? I care only if they succeed, which my IDS on the inside would ideally detect. Placing the IDS on the outside is going to quickly lull the administrator into complacency. I used to have a highly instrumented firewall that alerted me whenever someone attacked it. Two weeks later I was deleting its alert messages without reading them. Another important factor arguing for putting it inside is that not all intrusions come from the outside or the firewall. An IDS on the inside might detect new network links appearing, or attackers that got in via another avenue such as a dial-in bank."

- **CURRY from IBM:** "The IDS should be placed where it will be able to see as much of the network traffic you're concerned about as possible. For example, if you're concerned about attacks from the Internet, it makes the most sense to put the IDS outside the firewall. the most sense to put the IDS outside the firewall. This gives it an "unobstructed" view of everything that's coming in. If you put the IDS inside the firewall, then you're not seeing all the traffic the bad guys are sending at you, and this may impact your ability to detect intrusions."

- **SUTTERFIELD from Wheel Group:** "IDS ideally plays an important role both inside and outside a firewall. Outside a firewall, IDS watches legitimate traffic going to public machines such as e-mail and Web servers. More importantly IDS outside a firewall will see traffic that would typically be blocked by a firewall and would remain undetected by an internal system. This is especially important in detecting network sweeping which can be a first indication of attack. External systems will also give you the benefit of monitoring those services that firewalls determine are legitimate. Putting an IDS inside the firewall offers the added benefit of being able to watch traffic internal to the protected network. This adds an important element of protection against insider threats. The major drawback of IDS inside a firewall is that it cannot see a good deal of important traffic coming from untrusted networks and may fail to alert on obvious signals of an impending attack."

- **CHRIS KLAUS from ISS:** "Outside the firewall is almost always a good idea—it protects the DMZ devices from attack and dedicates an additional processor to protecting the internal network. Just inside the firewall is also useful-it detects attempts to exploit the tunnels that exist through the firewall and provides an excellent source of data for how well your firewall is working. Throughout your intranet may be the best place for IDS deployment, however. Everyone agrees that attacks aren't the only things we're worried about-there's internal mischief, fraud, espionage, theft, and general network misuse. Intrusion detection systems are just as effective inside the network as outside, especially if they're unobtrusive and easy to deploy."
- **GENE SPAFFORD:** "The IDS must be inside any firewalls to be able to detect insider abuse and certain kinds of attacks through the firewall. IDS outside the firewall may be useful if you want to monitor attacks on the firewall, and to sample traffic that the firewall doesn't let through. However, a true IDS system is likely to be wasted there unless you have some follow-through on what you see."

  Bottom Line:

  DRAGOS RUIU: "Just pick a spot you're likely to look at the logs for. :-)"

2.6 Libpcap complains about permissions problems, what's going on?

You are not running Snort as root or your kernel is not configured correctly.

2.7 I've got RedHat and ....

Check your version of libpcap. If it's not >= 0.5, you should update.

2.8 Where do I get the latest version of libpcap?

You can find the most current version at:

http://www.tcpdump.org

You might also want to have a look at Phil Wood's patches to libpcap for Linux:

http://public.lanl.gov/cpw/

2.9 Where do I get the latest version of Winpcap?

http://winpcap.polito.it/

2.10 What version of Winpcap do I need?

It depends. If you only have one processor, you can use the most current version (3.x). If you have a SMP box, you'll have to use either an older version (< 2.3) or the 3.x version plus a patch from http://www.ntop.org/winpcap.html.

2.11 Why does building Snort complain about missing references?

You must configure libpcap with the –install-incl option. (On Red Hat, install the libpcap-devel rpm.)

2.12 Why does building snort fail with errors about yylex and lex_init?

You need the lex and yacc tools or their gnu equivalents flex and bison installed.

2.13 I want to build a Snort box. Will this <Insert list of hardware> handle <this much> traffic?

That depends. Lower the number of rules is a standard performance increase. Disable rules that you don't need or care about. There have been many discussions on 'tweaking performance' with lots of 'I handle XX mb with a ___ machine setup.' being said. Look at some of the discussions on the snort-users mailing lists.

Here is an oft quoted bit on the subject from Marty:

"Hardware/OS recommendations"

Ok, here are the guidelines and some parameters. Intrusion detection is turning into one of the most high performance production computing fields that is in wide deployment today. If you think about the requirements of a NIDS sensor and the constraints that they are required to operate within, you'll probably start to realize that it's not too hard to find the performance wall with a NIDS these days.

The things a NIDS needs are:

1. MIPS (Fast CPU)
2. RAM (More is *always* better)
3. I/O (Wide, fast busses and high performance NIC)
4. AODS (Acres Of Disk Space)

A NIDS also needs to be pretty quick internally at doing its job. Snort's seen better days in that regard (when 1.5 came out the architecture was a lot cleaner) but it's still considered to be one of the performance leaders available.

As for OS selection, use what you like. When we implement Data Acquisition Plugin's in Snort 2.0 this may become more of a factor, but for now I'm hearing about a lot of people seeing alot of success using Snort on Solaris, Linux, *BSD and Windows 2000. Personally, I develop Snort on FreeBSD and Sourcefire uses OpenBSD for our sensor appliance OS, but I've been hearing some good things about the RedHat Turbo Packet interface (which would require mods for Snort to use, not to mention my general objection to RedHat's breaking stuff all the time). (ed note: take a drink, see FAQ 7.2 -dr)

2.14 What are CIDR netmasks?

(Excerpt from url: http://public.pacbell.net/dedicated/cidr.html)

CIDR is a new addressing scheme for the Internet which allows for more i efficient allocation of IP addresses than the old Class A, B, and C address scheme.

CIDR Block	**Equivalent Class C**	**Addresses**
/27	1/8th of a Class C	32 hosts
/26	1/4th of a Class C	64 hosts
/25	1/2 of a Class C	128 hosts
/24	1 Class C	256 hosts
/23	2 Class C	512 hosts
/22	4 Class C	1,024 hosts
/21	8 Class C	2,048 hosts
/20	16 Class C	4,096 hosts
/19	32 Class C	8,192 hosts
/18	64 Class C	16,384 hosts
/17	128 Class C	32,768 hosts
/16	256 Class C	65,536 hosts
/15	512 Class C	131,072 hosts
/14	1,024 Class C	262,144 hosts
/13	2,048 Class C	524,288 hosts

For more detailed technical information on CIDR, check out the following RFCs:

- RFC 1517: Applicability Statement for the Implementation of CIDR
- RFC 1518: An Architecture for IP Address Allocation with CIDR
- RFC 1519: CIDR: An Address Assignment and Aggregation Strategy
- RFC 1520: Exchanging Routing Information Across Provider Boundaries in the CIDR Environment

RFCs are available at http://www.rfc-editor.org/rfcsearch.html

2.15 What is the use of the "-r" switch to read tcpdump files?

Used in conjunction with a Snort rules file, the tcpdump data can be analyzed for hostile content, port scans, or anything else Snort can be used to detect. Snort can also display the packets in a decoded format, which many people find is easier to read than native tcpdump output.

3 Configuring Snort

3.1 How do I setup snort on a 'stealth' interface?

In *BSD and Linux:

```
ifconfig eth1 up
```

Solaris:

```
ifconfig eth1 plumb
ifconfig eth1 up
```

For NT/W2K/XP users, try the following:

NOTE: You are at your own risk if you follow these instructions. Editing your registry is DANGEROUS and should be done with extreme caution. Follow these steps at your OWN risk.

1. Get your device's hex value. ('snort -W' works for this)
2. open Regedt32
3. Navigate to: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}
4. Select the network card you wish to setup as the monitoring interface (this will be the {XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX} value).
5. Set IPAddress:REG_MULTI_SZ: to null (Double click on the string, delete data in the Multi-String Editor, then click OK)
6. Set SubnetMask:REG_MULTI_SZ: to null (Double click on the string, delete data in the Multi-String Editor, then click OK)
7. Set DefaultGateway:REG_MULTI_SZ: to null (Double click on the string, delete data in the Multi-String Editor, then click OK)
8. Close the Registry Editor, your changes will be saved automatically.
9. In a command prompt, run 'ipconfig' to verify the interface does not have an IP bound to it.

If you do not recieve an IP address listing from the interface you modified, you are good to go. To run snort with the specified interface, use the -i flag such as 'snort -v -d -p -i1'

3.2 How do I setup a receive-only ethernet cable?

Use an ethernet tap, or build your own 'receive-only' ethernet cable. Anyway, here is the cable I use:

```
LAN         Sniffer
1 -----\   /-- 1
2 ---\ |   \-- 2
3 ---+-*------ 3
4 -  |       - 4
5 -  |       - 5
6 ---*-------- 6
7 -          - 7
8 -          - 8
```

Basically, 1 and 2 on the sniffer side are connected, 3 and 6 straight through to the LAN. 1 and 2 on the LAN side connect to 3 and 6 respectively. This fakes a link on both ends but only allows traffic from the LAN to the sniffer. It also causes the 'incoming' traffic to be sent back to the LAN, so this cable only works well on a hub. You can use it on a switch but you will get ...err... interesting results. Since the switch receives the packets back in on the port it sent them out, the MAC table gets confused and after a short while devices start to drop off the switch. Works like a charm on a hub though.

Another method which uses a capacitor and should work on 100mbs links:

http://www.geocities.com/samngms/sniffing_cable

And another:

The UTP Y-Cable specified by Joe Lyman:

A less noisy option: it involves a couple of cat 5 cables and a single speed hub. The idea is to use the rcv cables for the wire going to the sniffer box and use the xmit cables from another hub port. This will give you a link light and allow your sniffer to rcv only. Cannot xmit because the xmit cables are not connected. This has been successfully used on netgear single speed hubs. It wont work on dual speed hubs due to the negotiation of speed.

Pin outs. They are reversed in the picture in order to prevent lines from crossing, and I only included the pins used.

```
* []HUB PORT 1       HUB PORT 2

  -----    -----

  x x r r     r r x x

  6 3 2 1     1 2 3 6

  | | | |         | |

  | | | ----------- |

  | | -------------

  | |

  | |

  | |

  | |

  6 3 2 1

  r r x x

  ----
```

```
      SNIFFER

       x = xmit

       r = rcv
```

You could make it a single cable by adding a battery to simulate the voltage from the xmit cables on the nic, but batteries die.

It's not recommended to cut the transmit side, shunt it to ground (pin 2). Some OS's will disable the interface if PIN 1 does not indicate a completed circuit.

3.3 What are HOME_NET and EXTERNAL_NET?

HOME_NET and EXTERNAL_NET are standard variable names that all of the Snort.org rules use. HOME_NET refers to the network(s) that you want to protect, where EXTERNAL_NET is the network(s) that you think attacks would come from.

3.4 My network spans multiple subnets. How do I define HOME_NET?

Snort 1.7 supports IP lists. You can assign a number of addresses to a single variable. For example:

```
var HOME_NET [10.1.1.0/24,192.168.1.0/24]
```

NOTE: Not all preprocessors support IP lists at this time. Unless otherwise stated, assume that any preprocessor using an IP list variable will use the first value as the HOME_NET. The portscan preprocessor is an example. To catch all detectable portscans, pass 0.0.0.0/0 in as the first parameter.

```
preprocessor portscan: 0.0.0.0/0 5 3 portscan.log
```

Use the portscan-ignorehosts preprocessor to fine tune and ignore traffic from noisy, trusted machines.

3.5 How do I set EXTERNAL_NET?

Many people set EXTERNAL_NET to "any".

```
    var EXTERNAL_NET any
```

By setting it to "any" Snort will alert you on any traffic matching a rule coming into or leaving your network.

To cut down on the work that Snort has to do, many people set it to "not HOME_NET."

```
    var EXTERNAL_NET !$HOME_NET
```

This tells Snort to define EXTERNAL_NET as everything except HOME_NET. For most people this is the best thing to set it to.

3.6 How can I run Snort on multiple interfaces simultaneously?

LINUX: If you aren't running snort on linux 2.1.x/2.2.x kernel (with LPF available) the only way is to run multiple instances of snort, one instance per interface (with the -i option specifying the interface). However for linux 2.1.x/2.2.x and higher you can use libpcap library with S. Krahmer's patch which allows you to specify 'any' as interface name. In this case snort will be able to process traffic coming to all interfaces.

*BSD: Use the "bridge" interface to combine your nics into a logical interface (bridge0).

3.7 My IP address is assigned dynamically to my interface, can I use Snort with it?

Yes. With Snort 1.7 and later, <interface>_ADDRESS variable is available. The value of this variable will be always set to IP address/Netmask of the interface which you run snort at. if interface goes down and up again (and an IP address is reassigned) you will have to restart snort. For earlier versions of snort numerous scripts to achieve the same result are available.

3.8 I have one network card and two aliases, how can I force Snort to "listen" on both addresses?

If you're using at least version 1.7, you can specify an IP list like this:

```
var HOME_NET [ 192.168.10.0/24, 10.1.1.1/16 ]
```

If you're using something older (version 1.6.3-patch2 or whatever) you can re-specify the HOME_NET variable multiple times like this (for example):

```
var HOME_NET 10.1.1.0/24
include misc.rules
var HOME_NET 192.168.1.0/24
include misc.rules
```

3.9 How do I ignore traffic coming from a particular host or hosts?

There are two basic ways to ignore traffic from a host:

- Pass Rules
- BPF Filters

Details:

1. Pass Rules:
   - Advantages:

  - Gives you rule based control over the packets.
  - Puts all your changes into 'one place'-snort.conf.
- Disadvantages:
  - Reverses the Rule order, can cause some headaches in tracking down problems.
  - One poorly written pass rule can 'blind' your whole network.
  - The more specific the pass rule is, the more CPU snort needs to process it which may be important on loaded nets.
- Example:
  For example to ignore ALL ICMP traffic from host ¡foo¿ using a pass rule:

```
pass icmp <foo> any -> $HOME_NET any
```

2. BPF Filters:

- Advantages:
  - Drops the packet at the BPF interface, which saves on processing.
  - Speeds up Snort since it 'never sees' those packets.
- Disadvantages:
  - Poorly constructed filters can 'blind-side' you.
- Example:
  - To ignore all traffic from 192.168.0.1:
```
snort <commandline options> not host 192.168.0.1
```
  - To ignore all ICMP ECHO-REQUESTS (pings) and ICMP-ECHO REPLY's (ping reply) from host <foo>:
```
snort <options> ‘‘not ( (icmp[0] = 8 or icmp[0] = 0) and host <foo> )’’
```

3.10 How do I get Snort to log the packet payload as well as the header?

It depends on how your Snort configuration logs. If it logs in binary format, you'll have to process the binary log in order to get cleartext. You also might have "-A <foo>" on the command line. Command line options always take override the .conf file.

3.11 Why are there no subdirectories under /var/log/snort for IP addresses?

It depends on how your snort configuration logs. If it logs in binary format, you'll have to process the binary log in order to get cleartext.

3.12 How do you get Snort to ignore some traffic?

Snort can be made to ignore traffic in a number of different ways:

1. Specify bpf filters on the command line the tcpdump man page has a description of bpf filters.
2. Use a pass rule
3. The portscan preprocessor has it's own special exclusion list with the portscan-ignorehosts.rules file directive

3.13 Why does the portscan plugin log "stealth" packets even though the host is in the portscan-ignorehosts list?

These types of tcp packets are inherently suspicious, no matter where they are coming from. The portscan detector was built with the assumption that *stealth* packets should be reported, even from hosts which are not monitored for portscanning. An option to ignore "stealth" packets may be added in the future.

3.14 What the heck is a "Stealth scan"?

A Stealth scan can refer to more than one type of scan.

- **Half-Open or SYN scan:** Instead of completing the full TCP three-way-handshake a full connection is not made. A SYN packet is sent to the system and if a SYN/ACK packet is received it is assumed that the port on the system is active. In that case a RST/ACK will be sent which will determined the listening state the system is in. If a RST/ACK packet is received, it is assumed that the port on the system is not active.
- **FIN scan:** According to RFC 793 a system should send back an RST for all TCP ports closed when they receive a FIN packet for a specific port.
- **XMAS tree scan:** According to RFC 793 a system should send back an RST for all TCP ports closed when they receive a FIN/URG/PUSH packet for a specific port.
- **NULL scan:** According to RFC 793 a system should send back an RST for all TCP ports closed when they receive a packet without any specified IP flags for a specific port.
- **Slow scan:** Any of the above scans could be used as a slow scan. A slow scan is when the attacker sends packets at a **very** slow rate. Sometimes these scans can be conducted over hours, days, or weeks. The idea is since they are so slow, the victim's security measures won't "notice" the scan.

3.15 What the heck is a SYNFIN scan?

SYNFIN scans got their name because there are both the SYN and FIN flags set.

3.16 Which takes precedence, commandline or rule file ?

The command line always gets precedence over the rules file. If people want to try stuff out quickly without having to manually edit the rules file, they should be able to override many things from the command line.

3.17 How does rule ordering work?

For => 2.0:

Please see the documents on v2.0 at: http://www.snort.org/docs/#devel.

For <= 1.9.X:

Marty has answered this many times on the snort-users mailing list. Here is an excerpt from a post on Thu, 22 Feb 2001 00:31:53 -0500, titled *"Re: [Snort-users] order of evaluation of rules."*

Currently, the data structures that store Snort rule data are the RuleTreeNodes (RTN) and the OptTreeNodes (OTN). These data structs are stored in a two dimensinal linked list structure with the RTNs forming the top row of the "Array" and the OTNs forming the columns under the RTNs. Here's an ASCII illustration from the infamous "lisapaper":

```
 RTN                   RTN                    RTN
 --------------        --------------         -----
| Chain Header |      | Chain Header |      | Chai
|              |      |              |      |
| Src IP       |      | Src IP       |      | Src
| Dst IP       |----->| Dst IP       |----->| Dst   .....
| Src Port     |      | Src Port     |      | Src
| Dst Port     |      | Dst Port     |      | Dst
|              |      |              |      |
 --------------        --------------         -----
        |                     |
        |                     |
        |                     |
 OTN   \|/            OTN    \|/
 -------V------       --------V-------
| Chain Option  |    | Chain Option   |
|               |    |       :        |
| Content       |            :
| TCP Flags     |            :
| ICMP Data     |
| Payload Size  |
| etc.          |
|               |
 ---------------
        |
        |
        |
  OTN  \|/
 -------V------
| Chain Option |
|              |
| Content      |
| TCP Flags    |
| ICMP data    |
| Payload Size |
| etc.         |
|              |
 --------------
        |
        |
```

Rules with similar rule headers (i.e. all the CGI rules, the old stealth port scan detection rules, most of the rules that focus on any single service, etc) are grouped under a single RTN for the sake of efficiency and the applicable OTNs are hung below them. For instance, if you have three rules like this:

```
alert tcp any any -> $HOME 80 (content: "foo"; msg: "foo";)
alert tcp any any -> $HOME 80 (content: "bar"; msg: "bar";)
alert tcp any any -> $HOME 80 (content: "baz"; msg: "baz";)
```

They all get grouped under the same RTN and the OTNs are "hung" beneath them like this:

```
  RTN
 --------------------
|  SIP: any          |
|  SP: any           |
|  DIP: $HOME        |
|  DP: 80            |
 --------------------
          |
          |
  OTN    \|/
 ---------v----------
| content: foo       |
| msg: foo           |
 ---------------------
          |
          |
  OTN    \|/
 ---------v----------
| content: bar       |
| msg: bar           |
 ---------------------
          |
          |
  OTN    \|/
 ---------v----------
| content: baz       |
| msg: baz           |
 --------------------
```

This is an efficient way to do things because we only need to check the data in the RTN once with this method. There is actually another dimension to this array: the function pointer list. Each node in the "array" has a linked list of function pointers attached to it. The functions in this list are the tests that need to be done to determine whether the data in the current packet matches the current rule node's information. Having this function pointer list gives us great efficiency and flexibility: we don't need to perform tests for things the current rule doesn't contain (e.g., "any" ports/IPs, packet content on non-content rules, etc). It also allows us to analyze the packet with any function without having to make major modifications to the whole program (which was the case in versions prior to version 1.5).

There are a couple of implications of this architecture. For the sake of this discussion on rules ordering, the one we're interested in is that rule order is tricky to figure out. For instance:

```
alert tcp any any -> $HOME 80 (content: "foo"; msg: "foo";)
alert tcp any any -> $HOME 1:1024 (flags: S; msg: "example";)
alert tcp any any -> $HOME 80 (flags: S; msg: "Port 80 SYN!";)
alert tcp any any -> $HOME 80 (content: "baz"; msg: "baz";)
```

gets built like this:

```
        RTN                            RTN
 --------------------           --------------------
|  SIP: any          |         |  SIP: any          |
|  SP: any           |-------->|  SP: any           |
|  DIP: \$HOME        |        |  DIP: \$HOME        |
|  DP: 80            |         |  DP: 1-1024        |
 --------------------           --------------------
       |                              |
       |                              |
  OTN    \|/                            \|/
 ---------v----------           ---------v----------
| content: foo       |         | flags: S           |
| msg: foo           |         | msg: example       |
 --------------------           --------------------
       |
       |
  OTN    \|/
 ---------v----------
| flags: S           |
| msg: Port 80 SYN!  |
 --------------------
       |
       |
  OTN    \|/
 ---------v----------
| content: baz       |
| msg: baz           |
 --------------------
```

Note that all three of the port 80 rules will be checked before the "1:1024" rule due to the order in which the applicable RTN has been created. This is because the rules parser builds the first chain header for port 80 traffic and sticks it on the rules list, then on the next rule it sees that a new chain header is required, so it gets built and put in place. In this case you would intuitively expect to get the "example" message and never see the "Port 80 SYN!," but the opposite is true.

3.18 How do I configure stream4?

Stream4 is an entirely new preprocessor that preforms two functions:

- Stateful inspection of TCP sessions
- TCP stream reassembly

Marty implemented stream4 out of the desire to have more robust stream reassembly capabilities and the desire to defeat the latest "stateless attacks" that have been coming out against Snort (c.f. stick and snot). Stream4 is written with the intent to let Snort be able to handle performing stream reassembly for "enterprise class" users, people who need to track and reassemble more than 256 streams simultaneously. Marty optimized the code fairly extensively to be robust, stable, and fast. The testing and calculations I've performed lead me to be fairly confident that stream4 can provide full stream reassembly for several thousand simultaneous connections and stateful inspection for upwards of 64,000 simultaneous sessions.

Stream4 is a large and complex piece of code (almost 2000 lines) and there are a lot of options associated with its runtime configuration, so I'll go over them here.

```
preprocessor stream4: [noinspect], [keepstats], [timeout <seconds>], [memcap]
```

stream4_reassemble defaults:

```
Reassemble client: ACTIVE
Reassemble server: INACTIVE
Reassemble ports: 21 23 25 53 80 143 110 111 513
Reassembly alerts: ACTIVE
```

There is a new command line switch that is used in concert with the stream4 code, "-z". The -z switch can take one of two arguments: "est" and "all". The "all" argument is the default if you don't specify anything and tells Snort to alert normally. If the -z switch is specified with the "est" argument, Snort will only alert (for TCP traffic) on streams that have been established via a three way handshake or streams where cooperative bidirectional activity has been observed (i.e. where some traffic went one way and something other than a RST or FIN was seen going back to the originator). With "-z est" turned on, Snort completely ignores TCP-based stick/snot "attacks".

3.19 Where does one obtain new/modifed rules? How do you merge them in?

New rules can be downloaded via CVS (see FAQ 3.20) or, alternatively, may be found at www.snort.org. There is a mailing list dedicated to Snort rules, called snort-sigs hosted at Sourceforge.

There are some scripts/programs to help you with rule management:

- oinkmaster: A simple Perl script to update the ruleset for you.

  http://www.algonet.se/ nitzer/oinkmaster/
- IDS Policy Manager: A win32 application that updates the ruleset using a GUI, then uploads your rulesets via scp.

  http://www.activeworx.com/idspm
- snortpp: a program to merge multiple files into one master file sorted by SID.

  http://dragos.com/snortpp.tgz

There is also this script that might be useful:

```
* []#!/bin/sh
  ###########################################################################
  ####
  #
  # Das Skript zum Herunterladen und installieren neuer IDS-Signaturen.
  #
  ###########################################################################
  ####
  MAILTO="admin@mydomain.de"
  MACHINE="machine1"
  #set -x
  SIGS_URL1="http://www.snort.org/dl/signatures/snortrules-stable.tar.gz"
  MD5_URL1="http://www.snort.org/dl/signatures/snortrules-stable.tar.gz.md5"
  WGET="/usr/bin/wget"
  #WGET_PARAMS="-N"
  WGET_PARAMS="-t 3 -T 5 -N -a /etc/snort/snort.log -P /etc/snort"
  # Wget parameters:
  #
  # -t            : Retries (here 3)
  # -N            : Get the file only if newer
  # -a            : Append the log messages to the specified file
  # -P            : Save the file to the specified directory
  # -T            : Timeout
  ECHO="/bin/echo"
  TAR="/bin/tar"
  KILL="/bin/kill"
  PIDOF="/sbin/pidof"
  SNORT="/usr/local/bin/snort"
  SNORTUSER="snort"
```

```
SNORTGROUP="snort"
KILLSIG="SIGUSR1"
SERVICE="/sbin/service"

# Where is the Snort configuration dir:
RULESPATH="/etc/snort/snortrules"
SNORTCFGPATH="/etc/snort"
MD5SUM="/usr/bin/md5sum"
MD5SUM_PARAMS=""

# The list of sensor interfacec divided by blanks
IFACES="eth0"

###########################################################################
####
#                          F U N C T I O N S
#
###########################################################################
####
###########################################################################
####
#    Die Funktion, die Snort fuer alle def. Interfaces auf dem System startet
#
#
#
#     Um sie zu erweitern muss man zwei Dinge tun:
#
#     1. Die Parameterliste von Interfaces erweitern
#
# 2. Das Konfigurationsfile unter /etc/snort/snort.conf_ethX anlegen          #
#
#
###########################################################################
####
restartsnort() {

# Restarting Snort for all interfaces
for i in $IFACES; do
         "$ECHO" "Setting up Snort for interface "$i""
         $ECHO "Restarting Snort..."
         #/usr/bin/killall snort
         if [ -f /var/run/snort_"$i".pid ]
         then
                 PID=$("$PIDOF" "$SNORT")
                 if [ -z "$PID" ]
                 then
                         "$SERVICE" snort restart
                 else
```

```
                                #`cat /var/run/snort_"$i".pid`
                               "$ECHO" "Restarting Snort running with PID "$PID" and reloading the rules..
                               "$KILL" -s "$KILLSIG" "$PID"
                        fi
                else
                        "$ECHO" "No PID file for interface "$i" found under /var/
run"
                fi
                "$ECHO" "Starting Snort"
                "$SNORT" -a -b -c "$SNORTCFGPATH""/snort.conf_""$i" -I -D -v
-i $i -u "$SNORTUSER" -g "$SNORTGROUP"
                PID=`cat /var/run/snort_"$i".pid`
                "$ECHO" "Snort running now with PID "$PID""
done
}
###########################################################################
####
#     Die Funktion zum ueberpruefen, ob und wie Snort auf dem System laeuft
#
###########################################################################
####
checksnort() {
SNORTS=$("$PIDOF" "$SNORT" | wc -w | awk '{print $1}')
SNORT_PIDS=$(/usr/bin/find /var/run -name snort\_eth[0-9]\.pid -ls |
wc -l | awk '{print $1}')
"$ECHO" "Snort instances counted:  $SNORTS"
"$ECHO" "Snort PID files found:    $SNORT_PIDS"

# 1. Fall: Snort laeuft nicht oder PID-File nicht da:
if [ "$SNORTS" = "0" -o "$SNORT_PIDS" = "0" ]
then
         "$ECHO" "Snort seems to be down or no PID file there..."
         "$ECHO" "Restarting Snort for all Interfaces..."
         "$SERVICE" snort restart
fi
# 2. Fall: Anzahl der Instanzen ungleich der Anzahl der PID-Files
if [ "$SNORTS" -gt "$SNORT_PIDS" ]
then
         "$ECHO" "More Snort instances than found PID files..."
         "$ECHO" "Something is wrong outthere..."
         "$ECHO" "Stopping all Snort processes..."
#        /usr/bin/killall -9 snort
        "$SERVICE" snort stop
        "$ECHO" "Hold on... Restarting Snort now..."
        "$SERVICE" snort restart
fi

# 3. Fall: Anzahl der Instanzen stimmt mit der Anzahl der PID-files ueberein
```

```
}
############################################################################
####
############################################################################
####
getrules() {
# Get the rules, since we know that they are newer...
$WGET $WGET_PARAMS $SIGS_URL1
$WGET $WGET_PARAMS $MD5_URL1
"$ECHO" "Readout the checksum..."
# MD5-Summe auslesen
if [ -f /etc/snort/snortrules-stable.tar.gz.md5 ]
then
         MD5SUM1=‘grep MD5 \
                  /etc/snort/snortrules-stable.tar.gz.md5|awk
’{print $4}’‘
else
         "$ECHO" "Error! No MD5-file found"
         exit 1
fi
"$ECHO" "Generating our own checksum..."
# MD5-Summe bilden
if [ -f /etc/snort/snortrules-stable.tar.gz ]
then
    MD5SUM2=‘md5sum /etc/snort/snortrules-stable.tar.gz|awk ’{print $1}’‘
else
         "$ECHO" "Error! No rules file found"
         exit 1
fi
if [ "$MD5SUM1" = "$MD5SUM2" ]
then
         "$ECHO" "The MD5-Checksum fits!"
         "$ECHO" "$MD5SUM1"
         "$ECHO" "$MD5SUM2"
         "$ECHO" "$MD5SUM1" >> /etc/snort/snort.log
         "$ECHO" "$MD5SUM2" >> /etc/snort/snort.log
         "$ECHO" "Proceeding..."
#        /bin/sleep 1
else
         "$ECHO" "Error! Wrong checksum! Aborting!"
         "$ECHO" "Install rules manually!"
         "$ECHO" "$MD5SUM1" >> /etc/snort/snort.log
         "$ECHO" "$MD5SUM2" >> /etc/snort/snort.log
         exit 1
fi
# Extract the new rules
if [ -f "/etc/snort/snortrules-stable.tar.gz" ]
then
```

```
         "$ECHO" "Extracting Snort rules..."
         "$TAR" -xzvf /etc/snort/snortrules-stable.tar.gz -C /etc/snort
else
         "$ECHO" "Lost the file! Something is wrong!"
         "$ECHO" "Aborting!!"
         exit 1
fi
# Deleting old rules
# Existiert das Verzeichnis ueberhaupt?
if [ -d "$RULESPATH" ]
then
#        /bin/rm "$RULESPATH"/*.rules
         /bin/mv -f /etc/snort/rules/*.rules "$RULESPATH"
         /bin/cp -f /etc/snort/rules/classification.config "$SNORTCFGPATH"
else
         "$ECHO" "Missing rules-directory!"
         "$ECHO" "Aborting!"
         exit 1
fi

# Cleaning up...
/bin/rm -rf /etc/snort/rules
# Give everything to root
/bin/chown root:root ${RULESPATH}/*
}
###########################################################################
####
#                                   M A I N
#
###########################################################################
####
# Error handling first
FCHK=$(/usr/bin/wget -spider -N -t 3 -T 5 "$SIGS_URL1" -P /etc/snort 2>&1)
ERR_MSG=$("$ECHO" "$FCHK" | egrep -oi "failed error")
# Log the error message explicitly
"$ECHO" "$FCHK" >> /etc/snort/snort.log
# If there is a word "failed" or "error" we break..
if [ "$("$ECHO" "$FCHK"| grep -i "failed")" ] || \
   [ "$("$ECHO" "$FCHK"| grep -i "error")" ]
then
         "$ECHO" "Error getting the files. The server seems to be not available."
         "$ECHO" "Error message:"
         "$ECHO" "$FCHK"
         "$ECHO" "Aborting!"
         exit 0
fi

"$ECHO" "Checking/getting files..."
```

```
# First extract the wget message
FCHK=$(/usr/bin/wget -spider -N -t 3 -T 5 "$SIGS_URL1" \
                              -P /etc/snort 2>&1 | grep "not retrieving")
/bin/date >> /etc/snort/snort.log
"$ECHO" "Wget-output:"
"$ECHO" $FCHK
# Logging what we've done and when
"$ECHO" "$FCHK" >> /etc/snort/snort.log
if [ -z "$FCHK"  ]
then
         "$ECHO" "The files on the server seem to be newer."
         "$ECHO" "We will get them now..."
         getrules
         # Reload rules
         "$SERVICE" snort reload
#        restartsnort
else
#
        "$ECHO" "The signature files on the server are older or not newer."
         "$ECHO" "Doing nothing for now..."
         "$ECHO" "Checking if Snort is running...."
         checksnort
         exit 0
fi
# Send Email
"$ECHO" -e "`ls -lA "$RULESPATH"`\n\nSnort running with PID $("$PIDOF"\
           "$SNORT")" | mail -s "Reloaded Snort signatures on $MACHINE"\
           "$MAILTO"
###########################################################################
####
###########################################################################
####
exit 0
#EOF
```

3.20 How do you get the latest Snort via cvs?

Snort can be checked out through anonymous (pserver) CVS with the following instruction set. The module you wish to check out must be specified as the modulename. When prompted for a password for anonymous, simply press the Enter key.

```
cvs -d:pserver:anonymous@cvs.snort.org:/cvsroot login

cvs -z3 -d:pserver:anonymous@cvs.snort.org:/cvsroot co snort
```

Updates from within the module's directory do not need the -d parameter.

You will need to issue the command "sh ./autojunk.sh" before starting ./configure.

3.21 How do I use a remote syslog machine?

Add the syslog switch, -s, and put this statement syslog.conf:

```
auth.alert         @managmentserverIP
```

Look at your snort.conf file for more info on the facility and Priority settings.

Make sure you have syslogd on the management server configured to allow syslog over UDP. Under RedHat, you can do this by editing /etc/sysconfig/syslog and adding the following line:

```
SYSLOGD_OPTIONS="-r -m 0"
```

This will start syslogd with the mark interval set to 0 (turning it off) and set it to receive network connections.

Then restart syslog. "man syslogd" for more info. You might also want to investigate syslog-ng (http://www.balabit.hu/en/downloads/syslog-ng/).

Example invocation of snort:

```
/usr/local/bin/snort -c /etc/snort/snort.conf -I -A full -s 192.168.0.2:514
-i rl0
```

Note for Win32 users:

Frank Knobbe wrote a patch for Snort to allow you to use '-s <host>' on the command line under Windows without nullifying the snort.conf. In other words, Snort still uses all settings from snort.conf but in addition uses the host from '-s' to send syslog alerts to. You can find the patch at:

http://www.snort.org/dl/contrib/patches/win32syslog/

3.22 How do I build this ACID thing?

Read carefully through all the docs for each package. Getting ACID to work is a lot of work, since it depends on many packages. You need a working Apache, a working PHP, a working GD (and the many libraries GD depends on), the ADODB package, and Phplot. This is a lot of stuff to configure.

A typical sequence to get this all working on Solaris 8: Use some binary packages from a trusted Sun freeware site (sunfreeware.com). The most problems were with PHP and the GD library. GD itself needs a bunch of packages and libraries to work also. It needs the libpng stuff, the libjpeg stuff (if you want jpeg), etc, etc. Read through the readme for GD. So you either need to get these and compile them also, or get some binary packages. PHP is the most difficult thing to get compiled correctly. The PHP package needs to be compiled with lots of "-with" flags for GD to work properly, otherwise it gets lots of run-time unresolved reference errors. Just using a "with" for GD isn't sufficient. You also need to "with" each library which GD uses also, or PHP can't find the functions it needs. Here's the "configure" line you can use to get PHP working:

```
./configure --with-mysql --with-apxs=/usr/apache/bin/apxs --with-gd
--enable-sockets --with-jpeg-dir=/usr/local/lib --with-png-dir=/usr/local/
lib --with-zlib-dir=/usr/local/lib --with-xpm-dir=/usr/local/lib
```

These 'with' statements basically have the effect of the Makefile including -L and -R statements for each library so that both the compile and run time linkers can find all the functions needed to find in the Apache module environment. Apache doesn't seem to consult the LD_LIBRARY_PATH when running a module (or PHP doesn't, or there's some config item in the Apache conf files, but you can just use the "withs").

Basically, you need to work from the bottom up. So you need to obtain/compile any libraries that GD needs and install them, and any libraries/packages those packages need. Then once you get GD compiled properly and installed, compile PHP. Then make a PHP script that calls phpinfo() (this is referenced in the ACID install) and carefully examine the page produced. Once satisfied PHP is working, then the 'foundation' is ready for the other stuff. Install PHplot and run a few of the tests. If they succeed, then install ADODB and ACID, tweak the config files, and it should all work. (heh, heh)

Also make sure you read the ACID FAQ on the web site. There's some stuff not in the ACID install guide that should probably be there. Namely the fact that the PHP "register_globals" option must be turned on in the php.ini file (it's off in the default PHP configurations).

ACID FAQ: http://www.andrew.cmu.edu/ rdanyliw/snort/acid_faq.html

4 Rules and Alerts

4.1 Errors loading rules files

Some common ones:

- `ERROR telnet.rules:YYY => Port value missing in rule!`
- `ERROR telnet.rules:YYY => Bad port number: "(msg:"blah"`
- `ERROR telnet.rules:YYY => Couldn't resolve hostname blah`

What's going on?

"telnet.rules" is the file where the syntax error occurred, and "YYY" is the line number it occurred on. There are a couple of possibilities:

1. The rule is missing a port value, has an invalid port number, or a bad hostname - in which case the ruleset author/maintainer should be notified.
2. More often, the rule is just fine, but a variable in it was not declared. Open the rules file, look at the rule on the line number provided, and confirm that the variables it uses have been declared. You can read more about variables at http://www.snort.org/docs/writing_rules/chap2.html#tth_sEc2.1.2

4.2 Snort says "Rule IP addr ("1.1.1.1") didn't x-late, WTF?"

Get rid of the quotes around the IP address and try again.

4.3 Snort is behind a firewall (ipf/pf/ipchains/ipfilter) and awfully quiet...

Your firewall rules will also block traffic to the Snort processes.

Note: This does not apply if Snort is installed **on** the firewall box.

4.4 Does snort see packets filtered by IPTables/IPChains/IPF/PF?

Snort operates using libpcap. In general it sees everything the network adapter driver sees before the network stack munges it. Linux IPTables, Linux IPChains, BSD PF and IPF and other packet filters do not prevent snort from seeing a packet that is present on the network wire. Even if an inbound packet is denied by the packet filter Snort will still see and analyze the packet if it is listening to that interface. Snort/pcap sees whatever comes out of or goes into the network adapter.

Note however that Snort is affected to the extent that the stream of data on the network wire is affected. Thus Snort will not see outbound packets which were denied while being sent since they will never reach the network adapter.

Under OpenBSD you can snort just the PF rejects by using the /dev/pflogN interface.

4.5 I'm getting large amounts of <some alerts type>. What should I do? Where can I go to find out more about it?

Some rules are more prone to producing false positives than others. This often varies between networks. You first need to determine if it is indeed a false positive. Some rules are referenced with ID numbers. The following are some common identification systems, and where to go to find more information about a particular alert.

System	**Example**	**URL**
IDS	IDS182	http://www.whitehats.com/IDS/182
CVE	CVE-2000-0138	http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0138
Bugtraq	BugtraqID 1	http://www.securityfocus.com/vdb/bottom.html?vid=1
McAfee	Mcafee 10225	http://vil.nai.com/vil/dispVirus.asp?virus_k=10225
Nessus	Nessus 11073	http://cgi.nessus.org/plugins/dump.php3?id=11073

It may be necessary to examine the packet payload to determine if the alert is a false positive. The packet payload is logged using the -d option. If you determine the alerts are false positives, you may want to write pass rules for machines that are producing a large number of them. If the rule is producing an unmanageable amount of false positives from a number of different machines, you could pass on the rule for all traffic. This should be used as a last resort.

4.6 What about all these false alarms?

Most think that a pile of false positives is infinitely preferable. Then people can turn off what they don't want. The reverse, having a small rule set, can lure people into complacency thinking that Snort is doing "its thing" and there is nothing to worry about.

4.7 What are all these ICMP files in subdirectories under /var/log/snort?

Most of them are likely destination unreachable and port unreachables that were detected by snort when a communications session attempt fails.

4.8 Why does the program generate alerts on packets that have pass rules?

The default order that the rules are applied in is alerts first, then pass rules, then log rules. This ordering ensures that you don't write 50 great alert rules and then disable them all accidently with an errant pass rule. If you really want to change this order so that the pass rules are applied first, use the "-o" command line switch, or the "order" config directive.

One other thing to keep in mind is that the alert might be generated from a preprocessor. If that is the case, then no pass rule will help you minimize the false positives. You will need to use a BPF filter.

4.9 What are all these "ICMP destination unreachable" alerts?

ICMP is the acronym for Internet Control Message Protocol They are failed connections ICMP unreach packet carries first 64 bits(8bytes) or more of the original datagrami and the original IP header.

The ICMP Destination Unreachable (message type 3) is sent back to the originator when an IP packet could not be delivered to the destination address. The ICMP Code indicates why the packet could not be delivered. The original codes are:

- 0 - net unreachable
- 1 - host unreachable
- 2 - protocol unreachable
- 3 - port unreachable
- 4 - fragmentation needed and DF bit set
- 5 - source route failed

As far as why... "it all depends..."

ICMP Unreachable Error Messages are divided into two groups:

1. ICMP Unreachable Error Messages issued by routers (all 16 of them)
2. ICMP Unreachable Error Messages issued by a Host (only 2)

What are the only 2 issued by a host? ICMP Port Unreachable - the destination port on the targeted host is closed (a.k.a. not in a listening state). ICMP Protocol Unreachable - the protocol we were trying to use is not being used on the targeted host.

Both ICMP Type field and Code field indicates why the packets could not be delivered. Some snort ICMP alerts" are informational like the ICMP alerts found in icmp-info.rules. At this time there are no references or even classtypes associated with these rules.

Other rules are more likely to be associated with untoward activity. For example, in icmp.rules you will find:

```
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP ISS Pinger";
content:"|495353504e475251|";itype:8;depth:32; reference:arachnids,158;
classtype:attempted-recon; sid:465; rev:1;)
```

which has a reference where the importance might be determined by checking out the arachnids reference. The classtype may indicate more or less the relative importance of the event.

When a destination UDP port is closed on the targeted host, a.k.a. not in a listening state, the targeted host will issue an ICMP Port Unreachable error message back to the offending packets source IP address, given in the query. Some programs use these messages, like traceroute with *nix based machines. Windows based machines (tracert) will default to ICMP Echo requests...

For further information about this, see:

- IP - ftp://ftp.isi.edu/in-notes/rfc791.txt
- ICMP - ftp://ftp.isi.edu/in-notes/rfc792.txt
- TCP - ftp://ftp.isi.edu/in-notes/rfc793.txt
- UDP - ftp://ftp.isi.edu/in-notes/rfc768.txt

and

http://www.iana.org/assignments/icmp-parameters

Actually, putting this URL somewhere handy is a good idea:

http://www.iana.org/

There is also a good ICMP paper on http://www.sys-security.com/

4.10 Why do many Snort rules have the flags P (TCP PuSH) and A (TCP ACK) set?

One of the reasons it alerts on a PA flags is to minimize the false positive. You will only get an alert upon successful connections. If you want to see all the attempts, you either have to modify the signatures, add you own signatures or use your firewall logs to see if an attempt to specific a port occurred.

4.11 What are these IDS codes in the alert names?

IDS means "Intrusion Detection Signature" and identifies a known attack attempt. You can learn more about a specific IDS id at the arachNIDS search engine on http://www.whitehats.com/. The "references" keyword in rules can also be a good pointer for further research.

4.12 Snort says BACKDOOR SIGNATURE... does my machine have a Trojan?

If you are dumping the data part of the packet, review it. These rules are known to have high false rates as most of them are just based on numeric port numbers.

4.13 What about "CGI Null Byte attacks?"

It's a part of the http preprocessor. Basically, if the http decoding routine finds a %00 in an http request, it will alert with this message. Sometimes you may see false positives with sites that use cookies with urlencoded binary data, or if you're scanning port 443 and picking up SSLencrypted traffic . If you're logging alerted packets you can check the actual string that caused the alert. Also, the unicode alert is subject to the same false positives with cookies and SSL. Having the packet dumps is the only way to tell for sure if you have a real attack on your hands, but this is true for any content-based alert.

4.14 Why do certain alerts seem to have 'unknown' IPs in ACID?

The Snort database plug-in only logs packet information into the database when an alert is triggered by a rule (signature). Therefore, since alerts generated by pre-preprocessors such as portscan and mini-fragment have no corresponding rules, no packet information is logged beyond an entry indicating their occurance. As a consequence, ACID cannot display any packet-level (e.g. IP address) information for these alerts.

For these particular alerts, certain statistics may show zero unique IP addresses, list the IP address as 'unknown', and will not list any packet information when decoding the alert.

4.15 Can priorities be assigned to alerts using ACID?

The quick answer to this question is no. ACID is at the mercy of the underlying database, since Snort doesn't assign priorities, ACID does not have priorities. Nevertheless, there are several work-arounds:

- It is possible to enforce priorities of sort at the database level by writing alerts of different severity to separate databases. For example, critical alerts such as buffer overflows can be written to one database, while scan alerts can be written to another. Then load two different versions of ACID, each pointing to a different instance of the database.
- With manual intervention Alert Groups (AG) can be used to assign priority. Essentially, this strategy entails creating an AG for each severity level and manually moving the alerts as they arrive into the appropriate group.

4.16 What about 'SMB Name Wildcard' alerts?

Whitehats IDS177 http://dev.whitehats.com/cgi/test/new.pl/Show?_id=netbios-name-query specifies traffic coming from *outside* of your local network. Allowing netbios traffic over public networks is usually very insecure.

If the rule you are using also refers to ingres traffic only, then it would explain why you don't see a lot of false positives. For anyone reading that does see a lot of false postiives - if you change your rule to reflect the source address as being !$HOME (or whatever variable you use to represent your internal network), then you should see most of the false positives go away.

The value of this chack is that a default administrative share C$ ADMIN$ or some such has been accessed. This shouldn't happen in normal use - when people want to share files they should be implicitely defining the shares and ACL.

4.17 What the heck is a SYNFIN scan?

SYNFIN scans got their name because there are both the SYN and FIN flags set.

4.18 I am getting too many "IIS Unicode attack detected" and/or "CGI Null Byte attack detected" false positives. How can I turn this detection off?

These messages are produced by the http_decode preprocessor. If you wish to turn these checks off, add -unicode or -cginull to your http_decode preprocessor line respectively.

```
preprocessor http_decode: 80 8080 -unicode -cginull
```

Your own internal users normal surfing can trigger these alerts in the preprocessor. Netscape in particular has been known to trigger them.

Instead of disabling them,try a BPF filter to ignore your outbound http traffic such as:

```
snort -d -A fast -c snort.conf not (src net xxx.xxx and dst port 80)
```

This has worked very well for us over a period of 5-6 months and Snort is still very able to decode actual and dangerous cgi null and unicode attacks on our public web servers.

4.19 How do I test Snort alerts and logging?

Try a rule that will fire off all the time like:

```
alert tcp any any -> any any (msg:"TCP traffic";)
```

Also take a look at sneeze at http://snort.sourceforge.net/sneeze-1.0.tar Sneeze is a false positive generator that reads snort signatures and generates packets that will trigger the rules.

4.20 What is the difference between "Alerting" and "Logging"?

There are two primary output facilities in Snort, logging and alerting. The alerting facility exists to let you know that something interesting has happened. The logging facility exists to log full packet information to the output format (pcap, ascii, database, etc).

The “alert” action in Snort is hard coded to do two things when an event is detected by Snort, write an event to the alert facility and log as much as possible/desired to the output facility. The “log” action merely logs the current packet to the logging facility without generating an alert. This is done so you can log interesting things (telnet sessions, whatever) without having to generate an alert on every packet.

The database plugin is something of an anomaly because it doesn’t separate the two functionalities very much. The “log” option attaches the log facility and the “alert” option attaches it to the alert facility. What this means in practical terms is that if the db plugin is in alert mode, it will only receive output from alert rules, whereas if it’s in “log” mode it will receive output from both log and alert rules.

4.21 Are rule keywords ORed or ANDed together?

¿From Section 2.1 of the Snort Manual:

> All of the elements in that make up a rule must be true for the indicated rule action to be taken. When taken together, the elements can be considered to form a logical AND statement. At the same time, the various rules in a Snort rules library file can be considered to form a large logical OR statement.

4.22 Can Snort trigger a rule by MAC addresses?

Not exactly. Snort logs MAC addresses and other L2 info within the packets. The arpwatch pre-processor can watch for games with MAC address changes. But there is no facility for triggering Rules form the L2 information. The content search keywords and depth and offset begin from the L3 payload, though we haven’t tried playing with really big offsets yet :-).

4.23 How can I deactivate a rule?

Rules can be called from an included file in snort.conf, which tells Snort to follow the path to the rules file specified, and load it at initialization. Rules can also be included in snort.conf directly. If you want to deactivate a single rule within any list of rules, you can use one of these techniques:

1. Delete the rule and re-initialize Snort
2. Place a # in front of the rule, commenting it out, and re-initialize Snort
3. Write a pass rule with the same properties in local.rules (or wherever you prefer), and re-initialize Snort with the -o option.

4.24 How can I define an address to be anything except some hosts?

Use the ! operator. E.g.:

```
var EXTERNAL_NET !$HOME_NET
```

Note that the negation operator does not work inside a list so the following will NOT work:

```
var EXTERNAL_NET [!192.168.40.0/24,!10.14.0.0/16]
```

but this will work:

```
var EXTERNAL_NET ![192.168.40.0/24,10.14.0.0/16]
```

4.25 After I add new rules or comment out rules how do I make Snort reload?

Usually a kill -HUP will work just fine. But if you are running inside of a chroot setup, this will not work as expected (see FAQ 6.19). If you're running like inside of a chroot jail, your best bet would be to kill and restart the snort process instead.

4.26 Where do the distance and within keywords work from to modify content searches in rules?

The "distance" keyword gives you a relative offset from the end of the last match, so it basically acts as a wildcarding mechanism. You can also use the new "within" keyword to limit how deep into the packet from the end of the distance it'll search before it stops.

4.27 How can I specify a list of ports in a rule?

You can't yet. You can specify a range of ports between X and Y with the notation X:Y. See the users manual (http://www.snort.org/docs/writing_rules/chap2.html#tth_sEc2.2.4) for more info on port ranges.

4.28 How can I protect web servers running on ports other than 80?

It is possible... It's a kludge, but it can work. Since the newer rules use the $HTTP_PORTS variable, you simply reset it and re-run the rules for the other ports.

For example:

```
var HTTP_PORTS 80
include web.rules
var HTTP_PORTS 8080
include web.rules
```

4.29 How do I turn off "spp:possible EVASIVE RST detection" alerts?

You want to pass the "disable_evasion_alerts" argument to stream4 in snort.conf.

4.30 Is there a private SID number range so my rules don't conflict?

Yes. Private SIDs start at 1000000.

4.31 How long can address lists, variables, or rules be?

The Snort parser has an 8K limit on variables and rules **after** expansion. In practice, this is not a major limitation. :-)

4.32 What do the numbers (ie: [116:56:1]) in front of a Snort alert mean?

For this explanation, we'll use the following example:

```
[**] [116:56:1] (snort_decoder): T/TCP Detected [**]
```

The first number is the Generator ID, this tells the user what component of Snort generated this alert. For a list of GIDs, please read etc/generators in the Snort source. In this case, we know that this event came from the "decode" (116) component of Snort.

The second number is the Snort ID (sometimes referred to as Signature ID). For a list of preprocessor SIDs, please see etc/gen-msg.map. Rule-based SIDs are written directly into the rules with the "sid" option. In this case, "56" represents a T/TCP event.

The third number is the revision ID. This number is primarily used when writing signatures, as each rendition of the rule should increment this number with the "rev" option.

5 Getting Fancy

5.1 I hear people talking about "Barnyard". What's that?

Barnyard is a output system for Snort. Snort creates a special binary output format called "unified." Barnyard reads this file, and then resends the data to a database backend. Unlike the database output plugin, Barnyard is aware of a failure to send the alert to the database, and it stops sending alerts. It is also aware when the database can accept connections again and will start sending the alerts again.

5.2 How do I process those Snort logs into reports?

1. Barnyard 5.1 can be used to process unified output files into a number of formats, including output to a database for further analysis.
2. SnortSnarf, a tool for producing HTML out of snort alerts for navigating through these alerts.

   http://www.silicondefense.com/snortsnarf/
3. If you want to set up logging to a database you could try ACID. Some documentation describing the current ACID functionality includes:

http://www.cert.org/kb/acid/

4. You can manipulate the unified output files directly without a separate database and browse/correlate them with Cerebus:

   http://dragos.com/cerebus/

5. For GUI front ends with simple log browsing, look at:

   - HenWen (OSX)
     http://homepage.mac.com/nickzman
     http://home.attbi.com/ rickzman/software/HenWen1.0.sit.bin
   - IDS Center (Win32)
     http://www.packx.net/
   - Puresecure (UNIX and Win32) (Formerly known as Demarc.)
     http://www.demarc.com/downloads/puresecure/
   - SnortCenter (UNIX and Win32)
     http://users.pandora.be/larc/
   - IDS Policy Manager (Win32)
     http://www.activeworx.com/IDSPM/

5.3 How do I log to multiple databases or output plugins?

Feed the unified output files through Barnyard twice to separate databases, or...

You can build redundancy by using multiple output plugins. Here are some examples.

Multiple instantiations of the database plugin:

```
output log_database: mysql, dbname=snort host=localhost user=xyz
output log_database: mysql, dbname=snort host=remote.loghost.com user=xyz
```

Remote database and local tcpdump:

```
output log_database: mysql, dbname=snort host=remote.loghost.com user=xyz
output log_tcpdump: /var/log/snort.tcpdump
```

Then you can replay the tcpdump file through snort to recreate the database.

CAVEAT: Just playing back the log packets might not trigger some of the state dependent pre-processors.

5.4 How can I test Snort without having an Ethernet card or a connection to other computers?

You have to use routing between two dummy devices:

```
modprobe -a dummy # (The dummy device has to be build by the kernel)
ifconfig dummy0 192.168.0.1
ifconfig dummy0:0 192.168.0.2
telnet 192.168.0.3 12345
```

It's important that the second IP is on the same interface and not, e.g. dummy1 or dummy2 and that the IP you try to access is *not* one of those you put on the interfaces. Use snort's ability to hear in promiscious mode on an IP address range. (HOME_NET=192.168.0.0/16)

5.5 How to start Snort as a win32 service?

1. You must use complete paths for everything. This means EVERYTHING: Command line, configuration files, everything.

   Examples: All include statements must be full paths:

   WRONG: include scan-lib

   CORRECT: include C:\snort\scan-lib

   All command line options must be full paths:

   WRONG: snort.exe -l ./log

   CORRECT: snort.exe -l C:\snort\log

2. YOU MUST ALWAYS HAVE A LOGGING DIRECTORY SET VIA THE COMMAND LINE (-l switch). If you do not set a logging directory the service will not start and, on NT/Win2k, your bootup will hang for about 4 minutes.

3. Make sure that snort runs correctly from the command line, without yet worrying about any service related issues. Test that all of your desired command line parameters are causing snort to function as you expect, such as correctly generating logging and alert output. If you can't get this part to work, then you don't have much hope of snort miraculously starting to work as a service.

4. Once you have step (3) running correctly, modify the command line parameters you used in step (3) to include the additional parameters "/SERVICE /INSTALL." For example, if your command line in step (3) was:

   ```
          snort -i1 -lC:\( \backslash \)snort\( \backslash \)log -cC:\( \
           backslash \)snort\( \backslash \)snort.conf
   ```

   then you should change it to be:

   ```
           snort /SERVICE /INSTALL -i1 -lC:\( \backslash \)snort\( \backslash \)
           log -cC:\( \backslash \)snort\( \backslash \)snort.conf
   ```

   Verify that the command line parameters were received correctly by running the command 'snort /SERVICE /SHOW.'

5. Start the service by running the command:

   ```
           net start snortsvc
   ```

Note that versions 1.9 (build 228), 2.0 (build 50), or any versions newer than these, will add entries to the Win32 event Log if there is ever a problem starting the service. Stop the service by running the command:

```
net stop snortsvc
```

6. The service can be uninstalled by running the command:

```
snort /SERVICE /UNINSTALL
```

5.6 Is it possible with snort to add a ipfilter/ipfw rule to a firewall?

Yes, with additional software in the contrib directory. But this can be dangerous and is not recommended unless you know what you're doing.

- SnortSam http://www.snortsam.net
- You also might wat to look at inline-snort at: http://www.snort.org/dl/contrib/patches/snort-inline
- Guardian is available and is part of the contrib directory in the tarball distribution.

  Guardian is a perl script which uses Snort to detect attacks, and then uses IPchains to deny any further attacks. The Guardian webpage can be found at: http://www.chaotic.org/ astevens/Guardian/index.html or you can use the mirror, http://www.cyberwizards.com/ midnite/Guardian/index.html

But one caveat... running external binaries can also be a performance limiter and your should read the caution below...

CHRISTOPHER CRAMER wrote:

> I'm sure this has been mentioned before in similar discussions, but this feels like a _really_ bad idea. What if the bad guys realize what is going on and make use of your blocking method as a DoS attack. All one would have to do start sending a series of triggering packets with spoofed IP addresses.
>
> Since I am no longer interested in breaking into your site, but rather making your life hell, I don't worry about the resulting data getting back to me. All I have to do is start proceeding up a list of IP addresses that I think you should no longer be able to talk to. When you come in the next morning, you find that you can no longer access the world.
>
> Just my $0.02.

Danger Will Robinson: Conventional wisdom says that auto-blocking is inherently dangerous.

However, for those that like to live at the bleeding edge of tech (and the separate process scanning logs and processing firewall commands sounds like a good way to do this...):

Please remember to include an exclusion list and put on them important sites such as root servers, other important dns servers (yours, and important sites for your users), and in general any host you don't want to receive phone calls about being DoSed when they are spoofed - usually inconveniently like that first time you actually manage to get on vacation.... (i.e. imagine "Crisis: the CEO can't reach his favorite redlite.org game.... you have to fly back from the Carribean ASAP....")

5.7 What is the best way to use Snort to block attack traffic?

```
snort-inline > hogwash >> SnortSAM|Guardian >> flexresp
```

5.8 Snort complains about the "react" keyword...

Rerun configure with the –enable-flexresp option and rebuild/reinstall.

5.9 How do I get Snort to e-mail me alerts?

You can't. Such a process would slow Snort down too much to make it of any use. Instead, log to syslog and use swatch or logcheck to parse over the plaintext logfiles.

With the Logsurfer docs, this might get you on the road to doing something with Snort and Logsurfer:

- http://www.obfuscation.org/emf/logsurfer/snort.txt

JASON HAAR provided an example Swatch (3.1beta) config that emails alerts:

- http://www.theadamsfamily.net/ erek/snort/snort-swatch.conf.txt

Here are some docs on swatch:

- http://www.oit.ucsb.edu/ eta/swatch/
- http://www.stanford.edu/ atkins/swatch
- http://rr.sans.org/sysadmin/swatch.php
- http://www.enteract.com/ lspitz/swatch.html
- http://www.cert.org/security-improvement/implementations/i042.01.html

IDS Center (see FAQ 5) on Win32 will also mail alerts.

5.10 How do I log a specific type of traffic and send alerts to syslog?

An example addition to snort.conf:

```
ruletype redalert {
   type alert
   output alert_syslog: LOG_LOCAL2
    output database: alert, postgresql, user=user dbname=snort password=pwd
}
```

Go into your local.rules and make sure you have something like:

```
redalert tcp any any -> any any (msg:"REDRUM REDRUM"; content:"redalerttest")
```

Then just do a telnet and type 'redalerttest.' Presto, alerts to both.

5.11 Is it possible to have Snort call an external program when an alert is raised?

Calling another program from within your main IDS loop is generally a bad idea. Having your IDS block while waiting for <something> of dubious reliability and origin nevermind timing while the packets are piling up is inviting packet loss. Especially with the already oh-so-consistent "Gee I think I'll go away for a minute" rock steady even cpu slicing Windows gives you (that's sarcasm, sorry). Go with the second approach.... process invokation is expensive on Windows.

You want to keep that IDS task humming and munching packets as efficiently as possible with as few interruptions as possible, imho, and not be invoking the penalty of process invocation.... particularly on Windows where process invocation is much much heavier task than *nix.

Even in a secondary process... You'll probably find something that stays "awake" all the time will work out much more nicely than something that gets "woken up" on a per alert basis for the aforementioned reasons.

As a better alternative go check out swatch or logwatch. Also for those new to UNIX, logging alerts to syslog and then using "tail -f /var/log/messages" might be what you are looking for.

5.12 How can I use Snort to log HTTP URLs or SMTP traffic?

It can be done with Snort, but you might find it faster to use mailsnarf and urlsnarf from Dug Song's dsniff package. Dsniff is available from:

http://www.monkey.org/ dsong/dsniff/

You can get a win32 port of dsniff at:

http://www.datanerds.net/ mike/dsniff.html

5.13 How can I move data from the snort db to snort_archive db like ACID does?

Use the perl script snort_archdb.pl found in the contrib dir of the Snort distribution (snort_archdb-90a.tar.gz).

5.14 What are some resources that I can use to understand more about source addresses logged and where they are coming from?

- http://www.arin.org/
- http://www.caida.org/tools/utilities/netgeo/
- http://netgeo.caida.org/perl/netgeo.cgi

- http://standards.ieee.org/regauth/oui/oui.txt
- http://www.codito.de/manufactor_hash
- http://coffer.com/mac_find/
- http://www.idefense.com/Intell/CI022702.html
- http://www.idefense.com/excelfiles/All.zip

Also, try "dig."

5.15 How do I understand this traffic and do IDS alert analysis?

1. You'll need to understand some basics of IP, TCP, and UDP. Things like destination addresses, source addresses, common ports, what TCP SYN, FIN and RST mean, etc. The same kind of basic knowledge of the internet you need to successfully configure a multi-interface router applies here, although you don't need to know router syntax. Some useful online references:
   - A truly basic "intro to TCP/IP" http://pclt.cis.yale.edu/pclt/COMM/TCPIP.HTM
   - A reasonable looking TCP/IP FAQ: http://www.itprc.com/tcpipfaq/default.htm
   - A basics of firewalls, DMZ's, etc.
     http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Firewall-HOWTO.html
2. You'll need to understand some basics of how network attacks work. I'd recommend skimming over "Smashing the Stack for fun and profit" by Aleph one. A deep understanding isn't necessary, but a casual read of this will give you some helpful basics in understanding the kinds of things that happen in an attack, and give you a better understanding of what to look for.

   http://www.insecure.org/stf/smashstack.txt
3. A good guide on securing systems is helpful, something like this one:

   http://www.openna.com/products/books/sol/solus.php

   http://www.seifried.org/lasg/
4. You'll need to understand the basics of internet servers, ie: what DNS, HTTP, FTP, SMTP, etc. are for. Most of that should be covered in the various other references made here.
5. An excellent reference on "oddball" traffic patterns commonly seen at network borders, also very helpful:

   http://www.robertgraham.com/pubs/firewall-seen.html
6. Also take a look at the "Recommended Reading" section (see FAQ 1.4)

5.16 How can I examine logged packets in more detail?

If you are using unified logging, you can use Barnyard (see FAQ 5.1) or the unified log to pcap converter written by Dragos:

http://dragos.com/logtopcap.c

You can then get additional decoding of the packet contents by analyzing these pcap files with either:

- Tcpdump - http://www.tcpdump.org
- Ethereal - http://www.ethereal.com

6 Problems

6.1 I think I found a bug in Snort. Now what?

Get some more diagnostic information and post it to "snort-users" at http://www.sourceforge.net/lists/listinfo/snort-users.

To get diagnostic information, compile snort as either:

```
make clean; make CFLAGS=-ggdb
```

or

```
make clean; make "CFLAGS=-ggdb -DDEBUG"
```

trace coredump as:

```
gdb /path/to/snort /path/to/snort/core

gdb> where
gdb> bt
gdb> print $varname, varname, \$\$varname etc..
```

or if corefile isn't generated, Snort should be started as:

```
gdb snort

gdb> run snort\_args\_go\_here
```

Then, when it crashes:

```
  gdb> where
  gdb> bt
  gdb> print \$varname, varname, \$\$varname etc..
```

6.2 SMB alerts aren't working, what's wrong?

The SMB alerting output plugin was removed in Snort 2.1 due to security issues.

6.3 Snort says "Garbage Packet with Null Pointer discarded!" Huh?

This was an internal diagnostic message triggered by an old bug in early versions of the defragmentation preprocessor. Upgrade to to the latest version of Snort.

6.4 Snort says "Ran Out Of Space." Huh?

This is an internal diagnostic message when the defragmentation preprocessor runs into its 32MB hard allocation space limit. Tell Dragos about it <dr@kyx.net>.

6.5 My ACID db connection times-out when performing long operations (e.g. deleting a large number of alerts).

PHP has an internal variable set to limit the length an script can execute. It is used to prevent poorly written code from executing indefinitely. In order to modify the time-out value, examine the 'max_execution_time' variable found in the 'php.ini' configuration file.

6.6 Why does ACID keep changing my sensor number and how do I keep it consistent?

¿From the code in op_acid_db.c:

```
* []/* if sensor id ==
   0, then we attempt attempt to determine it dynamically */ if(data->
  sensor_id == 0)

  {

      data->sensor_id = AcidDbGetSensorId(data);

  }
```

And AcidDbGetSensorId does the following:

```
* []"SELECT sid FROM sensor WHERE hostname='%s' AND interface='%s' "

  "AND filter='%s' AND detail='%u' AND encoding='0'", pv.hostname,

  pv.interface, pv.filter, op_data->detail)
```

If it gets a sensor back, it uses that sensor_id, if not, it inserts the new sensor. So from the code, to keep it consistent, don't change the hostname / interface / filter and detail.

6.7 Why does snort report "Packet loss statistics are unavailable under Linux?"

The Linux IP stack doesn't report lost packet stats. This also has been recently fixed with the 2.4+ kernel in the new version of libpcap...upgrade kernels and libpcap and it should now work.

6.8 My /var/log/snort directory gets very large...

Try this script to archive the files:

```
* []#!/bin/sh

  #
  # Logfile rotation script for snort writen by jameso@elwood.net.
  #
  # This script is pretty basic. We start out by setting some vars.
  # Its job is tho rotate the days logfiles, e-mail you with what
  # it logged, keep one weeks worth of uncompressed logs, and also
  # keep compressed tgz files of all the logs. It is made to be run
  # at midnight everynight. This script expects you to have a base
  # dir that you keep all of your logs, rule sets etc in. You can
  # see what sub dirs it expects from looking at the var settings
  # below.
  #
  # Things to note in this script is that we run this script at 12
  # every night, so we want to set the dirdate var the day the script
  # runs minus a day so we label the files with the correct day. We
  # Then create a dir for the days logs, move the log files into
  # todays dir. As soon as that is done restart snort so we don't miss
  # anything. Then delete any logs that are uncompressed and over a
  # week old. Then compress out todays logs and archive them away, and
  # end up by mailling out the logs to you.
  #
  # Define where you have the base of your snort install
  snortbase=/usr/snort
  # Define other vars
  # logdir   - Where the logs are kept
  # oldlogs  - Where you want the archived .tgz logs kept
  # weeklogs - This is where you want to keep a weeks worth of log files uncompressed
  # dirdate  - Todays Date in Month - Day - Year format
  # olddirdate - Todays date in the same format as dirdate, minus a week
  logdir=$snortbase/log
  oldlogs=$snortbase/oldlogs
  weeklogs=$snortbase/weeklogs
```

```
# When I first wrote this script, I only ran it on BSD systems. That was a
# mistake, as BSD systems have a date command that apperently lets you walk the
# date back pretty easily. Well, some systems don't have this feature, so I had
# to change the way that dates are done in here. I left in the old way, because
# it is cleaner, and I added in a new way that should be portable. If anyone
# has any problems, just let me know and I will try to fix it.
#
# You have to change the system var to either bsd or other. Set it to bsd if
# your system supports the "-v" flag. If you are not sure, set it to other.
system=bsd
if [ $system = bsd ]
then
 dirdate=`date -v -1d "+%m-%d-%y"`
 olddirdate=`date -v -8d "+%m-%d-%y"`
elif [ $system = other ]
 month=`date "+%m"`
 yesterday=`expr \`date "+%d"\` - 1`
 eightday=`expr \`date "+%d"\` - 8`
 year=`date "+%y"`
 dirdate=$month-$yesterday-$year
 olddirdate=$month-$eightday-$year
fi

# Create the Dir for todays logs.
if [ ! -d $weeklogs/$dirdate ]
then
 mkdir $weeklogs/$dirdate
fi

# Move the log files into todays log dir. This is done with
# a for loop right now, because I am afriad that if alot is
# logged there may be to many items to move with a "mv *"
# type command. There may a better way to do this, but I don't
# know it yet.
for logitem in `ls $logdir` ; do
 mv $logdir/$logitem $weeklogs/$dirdate
done

# Kill and restart snort now that the log files are moved.

kill `cat /var/run/snort_fxp0.pid`

# Restart snort in the correct way for you

/usr/local/bin/snort -i fxp0 -d -D -h homeiprange/28 -l /usr/snort/log \
-c /usr/snort/etc/08292k.rules > /dev/null 2>&1

# Delete any uncompressed log files that over a week old.
```

```
if [ -d $weeklogs/$olddirdate ]
then
 rm -r $weeklogs/$olddirdate
fi

# Compress and save the log files to save for as long as you want.
# This is done in a sub-shell because we change dirs, and I don't want
# to do that within the shell that the script runs in.

(cd $weeklogs; tar zcvf $oldlogs/$dirdate.tgz $dirdate > /dev/null 2>&1)

# Mail out the log files for today.

cat $weeklogs/$dirdate/snort.alert | mail -s "Snort logs" you@domain.com
cat $weeklogs/$dirdate/snort_portscan.log |
 mail -s "Snort portscan logs" you@do
main.com
```

6.9 Why does the 'error deleting alert' message occur when attempting to delete an alert with ACID?

Most likely the DB user configure in ACID does not have sufficient privileges. In addition to those privileges granted to log the alerts into the database (INSERT, SELECT), DELETE is also required.

This permission related issue can be confirmed by manually inserting a row into the database, then trying to delete it.

1. Log into MySQL with the same credentials (i.e. username, password) as you use in ACID:

   ```
   mysql -u -p
   ```

2. Insert a test row into the event table:

   ```
   mysql> INSERT INTO event (sid, cid, signature, timestamp)
   VALUES (1,1000000, "test", "0");
   ```

   (this assumes that you don't already have a row with an event ID=1000000. If you do just choose another event id #)

3. Now delete this newly inserted row:

   ```
   mysql> DELETE FROM event WHERE sid=1 AND cid=10000000;
   ```

   If you were not able to delete, this confirms that this is a permission problem. Re-login to mysql as root, and issue a GRANT command (giving the DELETE permission) to the ACID DB user:

   ```
   GRANT DELETE on snort.* to acid@localhost
   ```

   (this assumes that my alert database is 'snort', username is 'acid', and logging from the 'localhost')

6.10 ACID appears to be broken in Lynx

This is a known issue. Lynx mangles some of the form arguments appended to the URL. It's resolution is being investigated, but use Netscape, Opera, or IE in the mean time.

6.11 I am getting 'snort [pid] uses obsolete (PF_INET, SOCK_PACKET)' warnings. What's wrong?

You are using an older libpcap version with recent linux kernel. There should be no problem with it as long as your kernel supports SOCK_PACKET socket type. To get rid off the warning message however, you'll have to upgrade to some recent version of libpcap (a copy from www.tcpdump.org is recommended).

6.12 On HPUX I get device lan0 open: recv_ack: promisc_phys: Invalid argument

It's because there's another program running using the DLPI service. The HP-UX implementation doesn't allow more than one libpcap program at a time to run, unlike Linux (from snort.c).

6.13 Snort is dying with a 'can not create file' error and I have plenty of diskspace. What's wrong?

You may run out of free inodes, which basically also means you can not create more files on the partition. The obvious solution is to rm some. ;-)

6.14 I am using Snort on Windows and receive an "OpenPcap() error upon startup: ERROR: OpenPcap() device open: Error opening adapter" message. What's wrong?

Either winpcap is not installed, or you are using an incompatible version. Try upgrading to the latest version (2.3 as of 01/17/03). It is available from http://netgroup-serv.polito.it/winpcap/. It might also be an issue with SMP machines (see FAQ 2.10).

6.15 Snort is not logging to my database

There are a number of problems that may be causing Snort to fail to log to a database. You should check these:

1. You did not set up the database plugin in your configuration file.
2. You are using an older database schema, and should update it by running the create scripts from the /contrib directory of the source tarball.

3. You are using a command line option that overrides what you have in your configuration file. This is most often -A or -s. NOTE: If you wish to log to syslog as well, specify so in your configuration file rather then the command line.
4. There is a problem with your database configuration itself. Make sure the user you specify has the correct permissions, or that the database is even up and running.

6.16 Portscans are not being logged to my database

You need to change the output facility to 'alert' rather then 'log'. The portscan preprocessor calls output plugins registered as 'alert' plugins rather then 'log'.

```
output database: alert, mysql, user=snort dbname=snort host=localhost
```

6.17 Snort is not logging to syslog

There are a number of problems that may be causing snort to fail to log to syslog. You should check these:

- You are using a command line option that overrides what you have in your configuration file. This is most often -A.
- It may be logging to the wrong place. Make sure syslog is configured correctly.

6.18 I am still getting bombarded with spp_portscan messages even though the IP that I am getting the portscan from is in my $DNS_SERVERs var

Try adding /32 netmasks to those addresses:

```
var DNS_SERVERS \[xxx.xx.0.3/32,xxx.xxx.0.2/32\]
```

And make sure the $DNS_SERVERS variable is on the portscan-ignorehosts line:

```
preprocessor portscan-ignorehosts: $DNS_SERVERS
```

6.19 Why does chrooted Snort die when I send it a SIGHUP?

It's a known problem with permissions. Workaround, restart snort instead.

But the short answer is this: Due to the way the execv(2) call works, it "Restarts" snort from scratch. This has the odd side effect of making HUPS to a chrooted snort become recursive. For example, chroot to /snort. It now sees /snort as / . Now HUP snort. Snort now expects to have /snort/snort as /. In other words, you have to re-create your directories for your jail inside it. 4 HUPS and you will be in /snort/snort/snort/snort.

6.20 My snort crashes, how do I restart it?

Try one of these two shell scripts or daemontools (refer to website to daemontools)

```
* []#!/bin/sh
  #snorthup: Snort Restarter and Crash Logger
  #(dr@kyx..net with help from kmaxwell@superpages.com)

  $conf = "snort.conf"
  for $IFACE in fxp0 fxp1
  do
      if [ -f /var/run/snort_$IFACE.pid ]; then
          if !  ps -p ‘cat /var/run/snort_$IFACE.pid‘ > /dev/null ; then
              /usr/bin/logger -p user.notice snorthup: removing bogus pidfile
              /usr/bin/
  logger -p user.notice snorthup: restarting absentee snort o
  n $IFACE with conf file $i
              rm -f /var/run/snort_$IFACE.pid
              /usr/local/bin/snort -D -c $conf -i $IFACE
          fi;
     else
         /usr/bin/
  logger -p user.notice snorthup: restarting snort on $IFACE with
  conf file $conf
         /usr/local/bin/snort -D -c $conf -i $IFACE
     fi
  done
```

Another version:

```
* []#!/bin/ksh
  # snortstartd: Snort (Re)Starter
  # Dom De Vitto (dom@devitto..com)
  # (original idea by dr@kyx..net & kmaxwell@superpages.com)
  #
  # Note: You’d better get CONF and INTERFACES right or
  # this script will just keep trying to start snort.
  # Path to echo, sed, test, ps, grep, logger, rm, and sleep.

  PATH=$PATH:/usr/bin:/usr/local/bin ; export PATH

  # Point this to your conf file:

  CONF="/usr/local/share/examples/snort/snort.conf"

  # Which interfaces should Snort run on, e.g.:

  INTERFACES="hme0 hme1"
```

```
# Wait this many seconds between checks:

CHECKEVERY=5

# Full path to Snort:

SNORTBINARY=/usr/local/bin/snort

while :; do
  for INT in $INTERFACES
  do
    GREPSTRING="‘echo $SNORTBINARY -N -D -c $CONF -i $INT|sed
’s?\/?\\\/?g’‘"
    PSCMDLINES=‘(ps augxww 2>/dev/null||ps -ef 2>/dev/null) | grep
"$GREPSTRING"|wc -l‘
    if [ $PSCMDLINES = 0 ]; then
      logger -p user.notice -t "$0" "Starting Snort on $INT."
      $SNORTBINARY -N -D -c $CONF -i $INT 2>&1 > /dev/null
    fi
  done
  sleep $CHECKEVERY
done
```

6.21 Why can't snort see one of the 10Mbps or 100Mbps traffic on my autoswitch hub?

Basically it's a function of the design and all autoswitching hubs will behave in this way. It's the result of just not being able to stuff all the 100 Mbps traffic into the 10Mbps CSMA/CD. One solution I use to the problem is these new cheapie four port switches... put all the 10Mbps on it's own hub/switch/whatever and then route that to the 100Mbps hub I use for monitoring but put a cheapie switch in between that works as an adapter basically mediating the 10 up to 100 and vice versa.

The bad thing about hubs that *don't* have this "feature," is that in order to support 10bt devices, they throttle the entire hub speed down to 10bt if there is one or more 10bt only devices hooked up to it. I have seen this behavior (and did the bandwidth tests to proove it) on old 3com office connect 10/100 hubs (newer ones do the 2 hubs with a switch thing.) So, the point of what I am saying is, since these old hubs have no switching capabilities, and they don't know which port the traffic is supposed to go to (no switch=no arp table), they have to throttle bandwidth.

None of the hubs and switches have any significant amount of storage on the ethernet chip sets, and therefore *any* non-layer-three box that has $100 -> 10$ capability can only handle small amounts of traffic before the chip set drops incoming packets on the floor. Guess one might call that throttled bandwidth, but at the expense of retransmission timeouts and retransmissions at the end nodes.

If the box has a backplane, multiple cards and some network management functions, there is a higher *probability* the manufacturer has some additional buffering going on to keep dropped packets from happening on at least small bursts of traffic.

In the most generic of terms, if a box supports 100 "full-duplex," then its a switch (regardless of what the manufacturer calls it). If it supports $100 -> 10$, there is 50-50 chance the box has some MAC address awareness. If a box only supports $10 -> 10$ or $100 -> 100$, there is a high probability it is not MAC address aware and therefore functions like a hub.

Many hubs have different back planes, i.e., one for 10 and one for 100.

¿From a definition standpoint, a hub segment whether it be 10 or 100 is a single broadcast/collision domain. You will not see ANY traffic between segements without a bridge or layer3 route function between them.

In a switched environment, typically each port is a separate collision domain but one big broadcast domain. VLANs can be created in some to separate into separate broadcast domains and some have built in layer 3 functionality which basically connects a router into the backplane so that it can route between vlans at wire speed.

Think of a switch as a bridge with many ports. (that's what it is). Some switches support port mirroring or span ports. When you want to "sniff" frames in a switched environment (beyond just broadcast/multicast traffic) you need to be able to "see" the unicast traffic (telnet,http for example). You set up a port to mirror traffic from the ports that have the devices your interested in to the port you have your analysis device plugged into. Without doing so, you don't see the unicast conversations because the traffic is getting "switched" accross the backplane so pc on port 1 talks to server on port 2 and no other ports get this traffic. If server on port 2 broadcasts or multicasts, the information is flooded out all ports. (multicast can be controlled on some switches so only those ports that have listening stations get the traffic. Not all switches have these capabilities.

An excellent book on the topic is Interconnections by Radia Perlman. (Bridges and Routers).

Additional caveat: if you deal with full duplex on a switched port, only a tap would save you - users have succesfully used Shomiti's ones on 100MB FD ports, and used two Snort instances, capturing traffic on both directions. Port mirroring didn't work in that case ...

6.22 Trying to install snort it says: "bad interpreter: No such file or directory"

Usually this error comes from editing files on Windows machines. Often it shows up on the ./configure step. The configure script should be looking for the /bin /sh shell as its interpreter. If /bin/sh doesn't exist then you'll get this error. Check that whatever comes after the #! on the first line of configure is actually there.

If the file has been edited on a Windows machine it can sometimes Add CR/LF (VM) characters on the end of each line, so #!/bin/sh becomes #!/bin/shVM and as the ctrl-v/ctrl-m characters are special, and hidden by default on most editors, it can create a really hard to find problem. To remove the extra CR characters that UNIXish machines don't like, simply use the dos2unix command:

```
* []dos2unix <infile> <outfile>
```

If your OS doesn't have dos2unix, then you can use:

```
* []cat <infile> | tr -d ''\r'' > <outfile>
```

6.23 I'm not seeing any interfaces listed under Win32.

The reason you're seeing nothing in the interface list is a WinPcap problem. In previous versions of WinPcap there is a 1K buffer, which overflows if you have many interfaces (i.e., 10+). This has been replaced with an 8K buffer in more recent versions of WinPcap. The current snort distribution should already be linking against the newer WinPcap libraries, which should resolve this problem. Try obtaining a more recent build of snort.

6.24 It's not working on Win32, how can I tell if my problem is Snort or WinPcap?

See if WinDump will work with WinPcap. This should help you isolate which component is being bogus.

6.25 I just downloaded a new ruleset and now Snort fails, complaining about the rules.

First, make sure you downloaded the right ruleset for your version of snort. Snort.org generally hosts a ruleset for the released version of Snort, as well as rules for the development branch and sometimes copies for older versions of snort. This is generally the case for "unknown keyword in rule" type errors.

If you have the rules that are correct for your version of snort be aware that the snort rules tarball contains a snort.conf file. From time to time the snort.conf included with the rules gets changed as new .rules files are added, and new variables are added to support a better ruleset. When downloading new rulesets you should always give the included snort.conf a quick look-over to see if new includes or vars have been added, or at least be aware you should consult it if things do not work as expected. This is generally the case if you get messages indicating that something is undefined in a rule.

6.26 How do I speed up ACID and MySQL?

(ACID FAQ B-10) Two MySQL optimizations for you to check from the ACID faq:

http://www.andrew.cmu.edu/ rdanyliw/snort/acid_faq.html

- Compact the tables

  After numerous delete operations, 'holes' will occur in the native files used to store the tables decreasing the speed of the all queries. The following shell script will examine all the MySQL tables and compact them.

  ```
  []for table in ‘echo show tables$|$mysql snort$|$tail +2‘
    do
       echo optimize table $table|mysql snort
    done
  ```

- Creating indexes

  Some of the required indexes are not created in initial MySQL creation script. The following indexes can be added to significantly improve performance:

```
tcphdr.tcp_sport

tcphdr.tcp_dport

acid_ag_alert.ag_sid + acid_ag_alert.ag_cid
```

MySQL can be fast - you just need to have the proper indexing set up. If you need a good MySQL reference, pick up a copy of Paul DuBois' book, which is currently the bible for MySQL. O'Reilly also recently released a reference by Monty and the MySQL AB team.

The way to check if the indices are already there are with the SHOW INDEX command. For instance, to check the tcphdr table, you would run:

```
+ []mysql> show index from tcphdr;

  +----+------+------+-------+-------+-----+-------+-----+----+-----+

  | Table  | Non_unique | Key_name  | Seq_in_index | Column_name |
  Collation  | Cardinality | Sub_part | Packed | Comment |

  +----+------+------+-------+-------+-----+-------+-----+----+-----+

  | tcphdr |          0 | PRIMARY   |            1 | sid         |
          A |        NULL |     NULL | NULL   |         |

  | tcphdr |          0 | PRIMARY   |            2 | cid         |
          A |     2543146 |     NULL | NULL   |         |

  | tcphdr |          1 | tcp_sport |            1 | tcp_sport   |
          A |        NULL |     NULL | NULL   |         |

  | tcphdr |          1 | tcp_dport |            1 | tcp_dport   |
          A |        NULL |     NULL | NULL   |         |

  | tcphdr |          1 | tcp_flags |            1 | tcp_flags   |
          A |        NULL |     NULL | NULL   |         |

  +----+------+------+-------+-------+-----+-------+-----+----+-----+

  5 rows in set (0.00 sec)
```

You can see that in this case, the tcphdr.tcp_sport index is in line 3, and the tcphdr.tcp_dport is in line 4.

If you need to create the index, you can run:

```
+ []CREATE INDEX idx_tcp_sport ON tcphdr(tcp_sport);
```

To create a compound index, you would do this:

```
+ []CREATE INDEX idx_cpd_sid_cid ON acid_ag_alert(ag_sid,ag_cid);
```

If you want to take a closer look at the table structures, you can use the DESCRIBE command, and pass it the table name:

```
+ []      mysql> DESCRIBE tcphdr;

        +------+-----------+---+---+-----+----+
        | Field     | Type                 | Null | Key | Default |
  Extra |
        +------+-----------+---+---+-----+----+
        | sid       | int(10) unsigned     |      | PRI | 0       |
        |
        | cid       | int(10) unsigned     |      | PRI | 0       |
        |
        | tcp_sport | smallint(5) unsigned |      | MUL | 0       |
        |
        | tcp_dport | smallint(5) unsigned |      | MUL | 0       |
        |
        | tcp_seq   | int(10) unsigned     | YES  |     | NULL    |
        |
        | tcp_ack   | int(10) unsigned     | YES  |     | NULL    |
        |
        | tcp_off   | tinyint(3) unsigned  | YES  |     | NULL    |
        |
        | tcp_res   | tinyint(3) unsigned  | YES  |     | NULL    |
        |
        | tcp_flags | tinyint(3) unsigned  |      | MUL | 0       |
        |
        | tcp_win   | smallint(5) unsigned | YES  |     | NULL    |
        |
        | tcp_csum  | smallint(5) unsigned | YES  |     | NULL    |
        |
        | tcp_urp   | smallint(5) unsigned | YES  |     | NULL    |
        |
        +------+-----------+---+---+-----+----+
        12 rows in set (0.02 sec)
```

6.27 Why am I seeing so many “SMTP RCPT TO overflow” alerts ?

That rule looks for a TCP frame going to your SMTP server which contains more than 800 bytes of data. Any email can easily set that off if pipelining is used. SMTP command pipelining allows several command lines lines to be sent as a single packet without waiting for an OK response. Any good high-volume mailserver will try to pipeline where possible, resulting in a single TCP frame containing a series of command lines, each of which is not very long, but in aggregate easily exceed the 800 byte threshold, particularly if there is a large recipient list.

For more info on pipelining:

http://www.faqs.org/rfcs/rfc1854.html

If your mailservers are not vulnerable to these overflows you can disable this rule and regain some peace...

6.28 I'm getting lots of *ICMP Ping Speedera*, is this bad?

Quite ordinary. Windows update uses speedera based DNS, among other things. Of course, if the speedera traffic is coming from a Dialup account (as there have been reports of) it's likely a hacker tool. ;-)

6.29 Why are my unified alert times off by +/- N hours?

Unified log and alert files are stored in UTC.

6.30 I try to start Snort and it gives an error like "ERROR: Unable to open rules file: /root/.snortrc or /root//root/.snortrc." What can I do to fix this?

When Snort starts, it looks at the command line and checks for "-c /some/path/ snort.conf." If thats not there, then it will look for the one of the following files:

- /etc/snort.conf
- ./snort.conf
- $HOMEDIR/snort.conf
- $HOMEDIR/.snortrc
- ./.snortrc

Make sure your .conf is in one of those locations and then Snort will be able to find it or use the -c parameter to tell Snort the full pathname to the snort.conf.

```
snort -c /usr/local/etc/snort.conf
```

7 Development

7.1 How do you put Snort in debug mode?

In Snort 1.9 or higher,

1. ./configure –enable-debug

2. Look up the section of Snort you'd like to debug (look at src/snort.h) and add up the contants.
   For example,

   ```
   #define DEBUG_PORTSCAN2       0x00080000  /* 524288 / (+ conv2 ) 589824 */
   ```

   To debug both just portscan2,

   ```
   export SNORT_DEBUG=524288
   ```

   To debug both portscan2 and conversation:

   ```
   export SNORT_DEBUG=589824
   ```

3. Run snort as normal. You will need to redirect output to a file to cope with the large amounts of debug output.

8 Miscellaneous

8.1 What's this about a Snort drinking game?

:-) Check it out for yourself: http://www.theadamsfamily.net/ erek/snort/drinking_game.txt