A Methodology to Evaluate Rate-Based Intrusion Prevention System against Distributed Denial-of-Service (DDoS)

Prof Bill Buchanan, Flavien Flandrin, Richard Macfarlane, Dr Jamie Graves, Edinburgh Napier University, 10 Colinton Road, Edinburgh, EH10 5DT

Abstract

This paper defines a methodology for the evaluation of a Rate-based Intrusion Prevention System (IPS) for a Distributed Denial of Service (DDoS) threat. This evaluation system uses realistic background traffic along with attacking traffic, with four different DDoS attacks. The evaluation metrics are defined using Snort for: rate of packet loss; time to respond; available bandwidth; latency; reliability; CPU loading; and memory usage. The results show that system is effective in handling a low-throughput DDoS attack, but when a rate of 6,000 pps of malicious traffic is reached, Snort starts to drop malicious and legitimate packets, in at the same rate of loss. It also shows that the IPS operates well up to traffic throughputs up to 1Mbps.

1 Introduction

Intrusion Detection Systems (IDSs) are tools which interpret network traffic and/or host activity, and are often used to protect the network and hosts from malicious activity which is not detected by other security systems. Often they are used to detect reconnaissance attempts on a network by detecting port scan, or for any other method of scanning on the network (Mell & Scarfone, 2007). With IDSs it is often not possible to mitigate threats in real-time, thus, the need for Intrusion Prevention Systems (IPSs). Instead of IDSs, which can only record alerts, IPSs can drop or rewrite a packet, so it does not affect the network. They are thus more powerful against DoS attack than IDSs (Rowan, 2007). The main types of IPS are: content-based, which focuses on the information inside the packets (Pasquinucci, 2007); protocol-based (Lemonnier, 2001), which focuses on packets that do not respect the protocol standard; and rate-based, which analyses the amount of traffic and responds when the threshold of a particular parameter is overcome (Rowan, 2007).

One of the most difficult attacks to defend against is Denial of Service (DoS) which aims to disrupt legitimate activity by consuming computation and network resources. The success of this type of attack often corresponds to the time that the attacked is performed (Tanase, 2003), and when many machines are used to perform a DoS, there is a Distributed Denial of Service (DDoS) (Mirkovic, Dietrich, Dittrich, & Reiher, 2005) attack. The main aim of this paper is to produce a methodology that will permit the investigation of the performance of rate-based Intrusion Prevention System (IPS), for a range of network conditions. This will permit the

evaluation and comparison of different rate-based IPS, as they can be tested under the same conditions. The IPS used in this paper is the open-source tool Snort Version 2.8.5 which possesses a rate-based capability since this version (Snort Team, 2009).

2 Related Work

There is an increasing usage of IPSs by organisation, but very few research papers exist on the testing infrastructures (Chen, Delis, & Wei, 2008) (Shafi, Akram, Hayat, & Sohail, 2010) (Strasburg, Stakhanova, Basu, & Wong, 2008) of such system, while IDSs are fairly well covered. Related IDS literature covers the definition of metrics (Fink, O'Donoghue, Chappell, & Turner, 2002) (Ranum, 2001) (Sommers, Yegneswaran, & Barford, 2005) (Papadogiannakis, Polychronakis, & Markatos, 2010), automated evaluation (Lo, 2009), the analysis of their responses (Alessandri, 2004) (Càrdenas, Baras, & Seamon, 2006) (McHugh, 2000), and the analysis of the behaviour of such device used when used with load balancing (Le, Boutaba, & Al-Shaer, 2008) (Xu, Fu, Hu, Gao, & Su, 2008). IDS are often evaluated inside a test bed, where two different types of background traffic are used. The two types of traffic are:

- **Background traffic**. This can be generated from traffic generation tools, or is generated taken from real environments, which has the advantage of evaluating the IDS in a real environment (Sommers, Yegneswaran, & Barford, Toward comprehensive traffic generation for online ids evaluation, 2005) (Sommers, 2004(b)) (Lippmann, et al., 2000).
- **Attacking traffic**. This can be generated by tools or from captured traffic (Sommers, Yegneswaran, & Barford, 2004(a)).

Unfortunately, the results produced cannot be compared as they all use different methodologies for their evaluation (Maxion, 1998) (Lippmann, Haines, Fried, Korba, & Das, 2000) (Sommers, Yegneswaran, & Barford, 2005), (Lo, 2009), (Fink, O'Donoghue, Chappell, & Turner, 2002).

For DDoS, Mirkovic et al (2004), produced a taxonomy to classify it by the means used to prepare and perform the attack, along with the characteristics of the attack, and its effect on the target. This taxonomy offers a good understanding of the important methods of possible DDoS attacks. Many of these attacks use fairly simple attack tools, which are easy to defend against, but it is now possible that more complex attacks are possible (Mirkovic, Dietrich, Dittrich, & Reiher, 2005). Different protection mechanisms also exist to defend a network against DDoS, such has being oriented to application layer protection (Yu, Li, Chen, & Chen, 2007) (Yu, Fang, Lu, & Li, 2009) (Srivatsa, Iyengar, Yin, & Liu, 2006) or protocol-based protection (Schuba, Krsul, Kuhn, Spafford, Sundaram, & Zamb, 1997) (Mutaf, 2002). Keshariya and Foukia (2010) produced a toxonomy to classify the different defense mechanism which could be found.

Different tools have been produced by researchers to generate typical types of malicious traffic flows, such as, Cleo (Mirkovic J. , 2003) or MACE (Sommers, Yegneswaran, & Barford, 2004(a)), although they are not publicly available because of the misuse which could be performed with

them. Other tools are publically available which can perform DDoS such as the Packit tool (Bounds, 2003) or Hping3 (Sanfilippo, 2006).

For IPS evaluation methodologies, Chen, Deli, and Wei (2008) have developed an evaluation methodology, but it is not design to evaluate the device against DDoS. It only uses one machine to generate the background traffic, the attacking traffic and to simulate the target. Unfortunately the network traces used for the background are not publically available, so this methodology could not be implemented to perform evaluation of other IPS. Even if IPSs are more used, and DDoS attacks are a common threat, we are not aware of any methodology that permits to evaluate IPSs against such threats.

3 Design/Methodology

The IPS will be tested with DDoS, which has a large number of different profiles. Thus, the evaluation of the IPS will be done by using only four different attack profiles. Using only particular attack would thus permit an evaluation of against other IPS, using the same methodology. Besides reproducing the same type of attack, it is also important to design a test bed that will support any IPS type. With this, the IPS system tested is defined as the Device-Under-Test (DUT), and the methodology uses a black box testing approach, in an offline environment. Thus, a way to generate background traffic should be given, and, as the IPSs are working in-line, the background traffic should cross the device from both sides**.** Figure 1 thus shows a schematic of the test-bed, while Figure 2 is a schematic of the tested device.

3.1 Attack and Background Component Design

The tool Hping3 (Parker, 2003) can be used within a framework and has a scripting capacity over the TCP/IP stack, and allows the creation of nearly any kind of IP packet. This can be seen as a *scriptable TCP/IP stack* (Sanfilippo, 2006), and can be used to generate, in real time, crafted packets that will match the behaviour of a DDoS profile. Four different attacks will be used: TCP-based; UDP-based, ICMP-based; and a mixture of the three (**Table 1**). The TCP-based and UDP-based one will be against an FTP server, while the ICMP-based will be used against the host accessing an FTP server. By using three different protocols for the evaluation, it will be possible to view if the protocols have an impact on the capacity to handle attacks on the DUT.

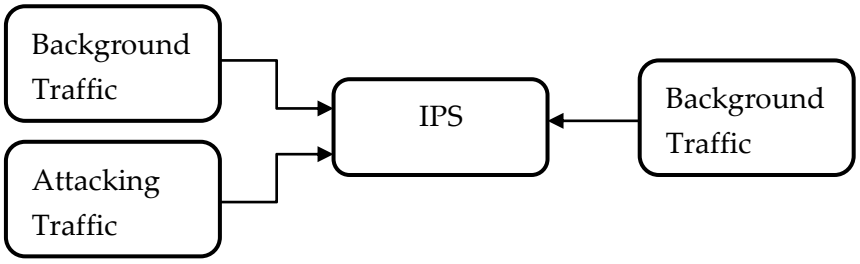

Figure 1: Schematic Implementation

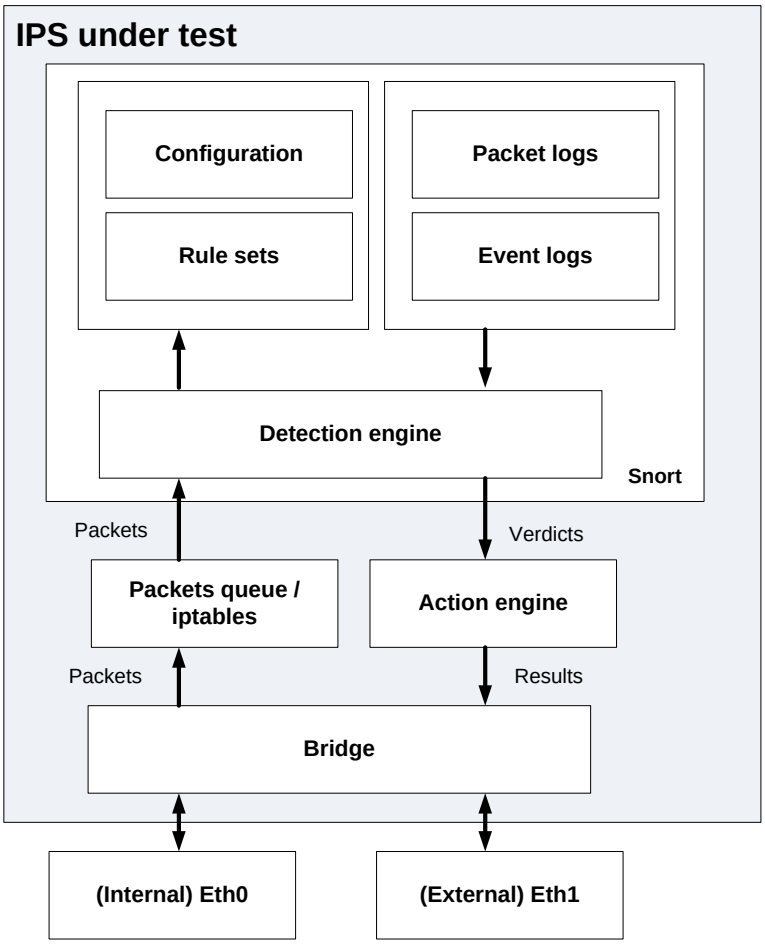

Figure 2: Schematic Implementation of the DUT

Generating realistic background traffic is fairly complex, as the tool must have the capacity to replay the same traffic previously generated, so each evaluation of the DUT has the same test environment. The utilisation of Harpoon is not possible because, although it generates realistic traffic it cannot guarantee that it will generate the same traffic again. Furthermore, Corsini (2009) has shown that Harpoon does not generate realistic enough traffic. This is why Tcpreplay (Turner, 2003) has been in this framework, as it offers the possibility to repeat data set with different options (such as for network speed).

Table 1: DDoS used overview

Distributed Denial of Service	Description
TCP-based	TCP packets with the port 21 for destination and t SYN flag on. The IP source will be random spoofed.
UDP-based	UDP packets with the port 21 for destination. The source will be randomly spoofed.
ICMP-based	ICMP echo request packets. The IP source will randomly spoofed.
Mixed traffic	A mix of the three previous DDoS

Within the IPS evaluation system, it would be best to use the same traffic that will actually be used in real-life, thus, one solution is to create an environment to generate traffic and capture it. The other solution is to use already existing data set, such as for:

- DARPA data-set (McHugh, 2000). These data sets have been widely used.
- Bro. This is an enterprise trace that can be found on the Bro web site (Allman et al, 2004). Bro (Paxson, 2003) is an open-source IDS which aims to handle high-speeds traffic (Gbps).

Using public available data sets permits the comparison of the results of an IPS evaluation using the same data set. The data set from Bro's website has been discarded as very little work has been done on it within a research context. These data sets also have many drawbacks (Mahoney & Chan, 2003) Even if the testing methodology depends of these data sets; however, the methodology is opened to accept other data set that proves to be better than DARPA's data sets.

3.2 Evaluation Metrics Design

The metrics used for the evaluation of an IPS should be similar to the metrics used in evaluation of IDS, as they have common points with IPS. These metrics include CPU loading and memory utilization (Sommers, Yegneswaran, & Barford, 2005), along with the available bandwidth, latency, and the time to respond to a threat. The rate of packets loss should also be monitored (NSS Labs, 2006). These metrics could be classified in three different categories:

- **Response Metrics**. This contains the packets lost by source and packets lost by destination. The destination mode for the rate filter tells the IPS to keep in-memory the each destination IP address for a rate alert that is watched by a filter. Snort does not have the capacity to discern the malicious traffic from the legitimate traffic when the filter is triggered. All packets that are detection to the watched destination will be dropped. The obvious problem is that legitimate user will not have access to this particular destination. The other mode for the rate filter is source mode, where Snort keeps in-memory each source IP address for a rate alert, which are watched by a filter. When the rate is reach, the filter is triggered. This will permit the protection of the network while still accepting legitimate users access the network. The main drawback of this mode is that if a large number of machines source the attacking traffic, the filter might never be triggered.
- **Input Metrics**. This contains available bandwidth, latency**,** time to respond metrics, and reliability.
- **Resources Metrics**. This includes the CPU and memory usage, and Table 2 summarises these metrics. The two packet lost metrics permit the determination of the capacity of the DUT to not block legitimate traffic (false positives) and to not miss malicious packets (false negatives). The monitoring of the CPU and memory utilisation with the number of packet lost permits the analysis of whether there is a relationship between the two. Latency and available bandwidth permits an evaluation of the capacity of the IPS to cope with different amounts of traffic.

Table 2: Evaluation Metrics

Response Metrics	Description
Packets lost by source	Rate legitimate packets lost
Packets lost by destination	Overall packets lost
Impact Metrics	
Available Bandwidth	Rate of maximum traffic that could pass through the DUT
Latency	Time spent for a packet to cross the DUT
Data Throughput Since Intervention (DTSI)	Number of packets before the DUT responds to the threat
Reliability	Time without a system error
Resources Metrics	
CPU Load	Percentage of CPU load
Memory Load	Percentage of memory load

4 Implementation/Experimentation

The test bed is composed of an FTP server, which is inside the network while the attacking machines are outside the network. Table 3 contains the description of the operating systems used, and the resources of the different devices. TippingPoint (2008) have demonstrated that it is important to be able to send traffic which covers 80% of the maximum throughput of the IPS. As a test, Wireshark (Wireshark Fundation, 2006) is used to monitor the incoming and outgoing packet that crossed the device, and the results have shown that Snort starts to drop packets when the throughput is over 1Mbps. Therefore, to respect the ratio of 80% of legitimate traffic, the background traffic should cross the device at a speed of 819kbps.

Table 3: Specifications of Machines

Machine	Operating System	CPU	Memory
IPS	Ubuntu 9.10	P4, 3.20 GHz	512 Mbytes
Tcpreplay Machine	Ubuntu 9.10	P4, 3.20 GHz	512 Mbytes
Attacking machines	Ubuntu 9.10	P4, 3.20 GHz	512 Mbytes
Client machines	Ubuntu 9.10	P4, 3.20 GHz	512 Mbytes
Server machines	Ubuntu 9.10	P4, 3.20 GHz	512 Mbytes

4.1 Snort Implementation

Snort is commonly used as an IDS, and in-line mode it is built with the flag *--enable-inline*, which will analyse packets stored in the queue of iptables. All IP packets which cross the devices are analysed using:

```
iptables –A OUTPUT –j QUEUE; iptables –A FORWARD –j QUEUE
```

These are the only rules for iptables, where the flag *-Q* has to be added to tell Snort to read packets from the queue. The set of rules used are from the Vulnerability Research Team (VRT Team, 2009) and the attacks that are being carried out to evaluate the IPS use packets, which are not malicious. For the ICMP-based attack, Rule 469 of the rule set match is used, and for TCP-based and UDP-based ones, the crafted rules are:

```
# TCP rule that detect TCP packet with the SYN flag on in
# destination of an FTP server.
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (flags: S; msg:"FTP - TCP SYN
   FLAG"; classtype:attempted-dos; sid:100001; rev:1;)

# UDP rule that detect UDP packet in destination of an FTP
# server.
alert udp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP - UDP";
   classtype:attempted-dos; sid:100002; rev:1;)
```

These rules will generate a lot of noise, which could have an impact on the final results. Since Version 2.8.5, rate-based filtering has been implemented in Snort, and the filter is defined with the *rate_filter* parameter. The following are then used to set the rate filter for ICMP-based, TCP-based and UDP-based attacks:

```
rate_filter \
      gen_id 1, sig_id 469, \
      track by_dst, \
      count 15, seconds 2, \
      new_action drop, timeout 30
rate_filter \
      gen_id 1, sig_id 100001, \
      track by_dst, \
      count 30, seconds 5, \
      new_action drop, timeout 30
rate_filter \
      gen_id 1, sig_id 100002, \
      track by_dst, \
      count 10, seconds 2, \
      new_action drop, timeout 30
```

4.2 Background and Attacking Traffic Generation

When using Hping3 the IP destination, protocol used, and interval between each packet is specified (where the interval is measured in microseconds). The generation commands then become:

ICMP-based: This attack sends ICMP echo-request (ping) to the target. The flag "-1" specifies that the protocol is ICMP. By default, Hping3 will send ICMP echo-request packets:

```
# hping3 -1 10.0.3.10 --rand-source –i u1000
```

TCP-based: This attack sends TCP SYN packets to the target, using the flag "-S", and the TCP port is specified using the "-p" flag. The source port is chosen randomly by the program (and will be above 1024) and will be incremented of one for each packet.

```
# hping3 -S 10.0.3.10 -p 21 --rand-source –i u1000
```

UDP-based: This attack sends UDP packet, using the "-2" flag. As for the previous attack the UDP port is specified with "-p", and the same procedure will be applied to define the source port:

```
# hping3 -2 10.0.3.10 -p 21 --rand-source –i u1000
```

Mix-based: This is a blend of the three previous attacks. The repartition between the attacking devices should be one third for each protocol so none of them are privileged.

4.3 Metrics Implementation

The following defines the implementation of the metrics:

- **Packets lost by destination:** A sniffer records the traffic that passes through the IPS from each side, which is then compared with the attack-free results. This permits the extract of the percentages of legitimate traffic that have been dropped for each rate of attacking traffic. It is expected that when the throughput of the attacking traffic increases, the number of packets lost increases too.
- **Packets lost by source:** For this metrics, a modification should be done on the rate filter, where the *by_dst* attribute is changed *by_src*. When the rate of malicious traffic fires the rate filter, it starts to drop all packets destined for the target. The filter will not recognise legitimate traffic from malicious traffic, as the IPS does not have the capacity to do it. In this case, measuring effectiveness is pointless. The *by_src* attribute tells Snort to keep a trace of each source IP address that passes through it. When the filter is fired, it starts to drop packets regardless from the destination. This filter could be used to prevent an attacker from attacking from multiple sources from inside the network, in order to try stress an edge router. This experiment will be the only one using this filter. The *--rand-source* flag should not be used with Hping3 for this experiment, as the source IP address is random, and the rate filters might never trigger, so the results will not be relevant. To evaluate the rate of dropped packets Hping3 will be used to send 60 packets (with the flag *-c*) using the same pattern than the attacking traffic with an interval of one second between each packet. The metric used will be the percentage of packets lost displayed by Hping3.
- **Available Bandwidth:** The tool Netperf 2.4.5 (Jones, 2000) was used to calculate the available bandwidth that the DUT could offer under attack. A server was set-up inside the network while the client was set-up outside the network (where inside represents the network which is protected by the IPS and outside represent the network where the attacking are set-up) .This metric permits the determination of the capacity of the DUT to keep a good throughput while handling malicious traffic. The hypothesis is that the available bandwidth will reduce, as the rate of malicious traffic will increase. To calculate the throughput with Netperf the TCP_STREAM test was used. This test involves the transfer of data between the client and the server [the Netperf client and Netperf server as describe earlier]; the time of transfer will be used to calculate the current throughput (Jones, 2007). The following line is the template used to launch Netperf client:

```
# netperf -H <IP address of the Netperf server>
```

- **Latency:** The experiment measure the RTT (Round Trip Time), which gives a measure of latency. The tool used to calculate the latency is Ping (Muuss, 1983), which sends 60 packets across the IPS and determines the average time to come back for each packet. It is expected

that the latency will increase when the rate of attacking traffic is increased. The following example shows how Ping is used:

```
# ping 192.168.1.100 -c 60
```

- **Data Throughput since Intervention (DTsI):** The different rate filters have attributes which specify the number of events that should pass in a particular time, before firing up the filter. The aim of this evaluation is to evaluate if the rate of malicious traffic has an impact on the number of packets that match an attack to fire up the rate filter. The number of packets that pass the IPS before the filter takes action is monitored with Wireshark (Combs, 2006). When the number of malicious traffic increases, it is expected that more packets will pass through the device before the rate filter is triggered.
- **Reliability**: This metric is evaluated by running the system over a long time period. The time chosen for the initial set of experiments is seven hours. During this time, the background traffic will be replayed with malicious traffic at a reasonable rate. This will permit the evaluation if DUT can handle DDoS over a relatively long time period.
- **CPU Load:** This represents the percentage of CPU used by the machine hosting the IPS. This metric will evaluate the impact of attacking traffic on the physical resources of the system. The percentage of CPU load is calculated using the *uptime* UNIX tool.
- **Memory Utilization:** This is the same experiment as the CPU load, using the *free* UNIX tool. For this experiment and the CPU load experiment, it is expected that the resources will increase as the rate of malicious traffic increases.

5 Results

Each experiment has been carried out as described in Section 3, including the part where the background traffic and attacking traffic are sent. After each experiment, the IPS was stopped and all data recorded for further analyse later on. In addition, the machine hosting the IPS was rebooted between each experiment to ensure that the environment is the same between each of them. Figure 3 is the representation of CPU load in relation with different attacking speed traffic, and Figure 4 is the results of memory load. Data for this experiment have been taken at the same time that the CPU load results. Figure 5 shows available bandwidth results of the experiment and latency results are represented in Figure 6. Figures 7 and Figure 8 represent respectively the rate filter by destination experiment and the rate filter by source experiment. Results where the attacking traffic is 0pps represent results with only the background traffic crossing the IPS.

5.1 CPU and Memory Load

As expected the CPU load depends of the traffic rate that the DUT should process. The ICMP protocol is the one who need less resources because it stayed at 40-39% of CPU usage while UDP consume until 66% of CPU resources. ICMP echo-packets sent are small; therefore, less processing time is needed for the IPS to take decisions about these packets. For this protocol

when the rate of 6 000pps is reached the CPU load stay stable (around 40%). For the TCP protocol the CPU load decrease when the rate of attacking traffic is 6 000 pps from 58% CPU load to 48%. In another end the UDP protocol, never stop its need of CPU load. The TCP and ICMP results highlight that an optimisation might be done by Snort during the processing of these packets. The mix protocols experiment follow the UDP experiment that comfort the idea as the UDP protocol need more resources than the others do.

Even if the results show an augmentation of memory load as expected, the results are not significant enough because of the augmentation is in the order of 20Mbytes for each protocols. Moreover, the results for the UDP protocol are strange. This might be because the experiment is run under a Linux machine and it is not possible to know how the system will handle its memory.

5.2 Available Bandwidth

This experiment has been carried out when the DUT was not in the system and without background traffic to measure the impact of the test-bed on the results. When the IPS is not set-up in the network the available bandwidth is 8.05Mbps. When the IPS is set-up in the test-bed the available bandwidth is 7.86Mbps. We could conclude that the test-ted does not bias the results for this experiment because results are about the same. When only background traffic is crossing the DUT, the available bandwidth is 2.47 Mbps.

The results obtained after the experiments with attacking traffic correspond to the expected results. All protocols have the same results with slight differences. The UDP protocol has the poorest result of the experiment. When attacking traffic reach a rate of 15 000 pps the available bandwidth is near zero and at zero when the number of packets per second is of 25 000pps. Then rate of malicious traffic reach 6 000 pps the available bandwidth drop of half is capacity when only background traffic was crossing the IPS.

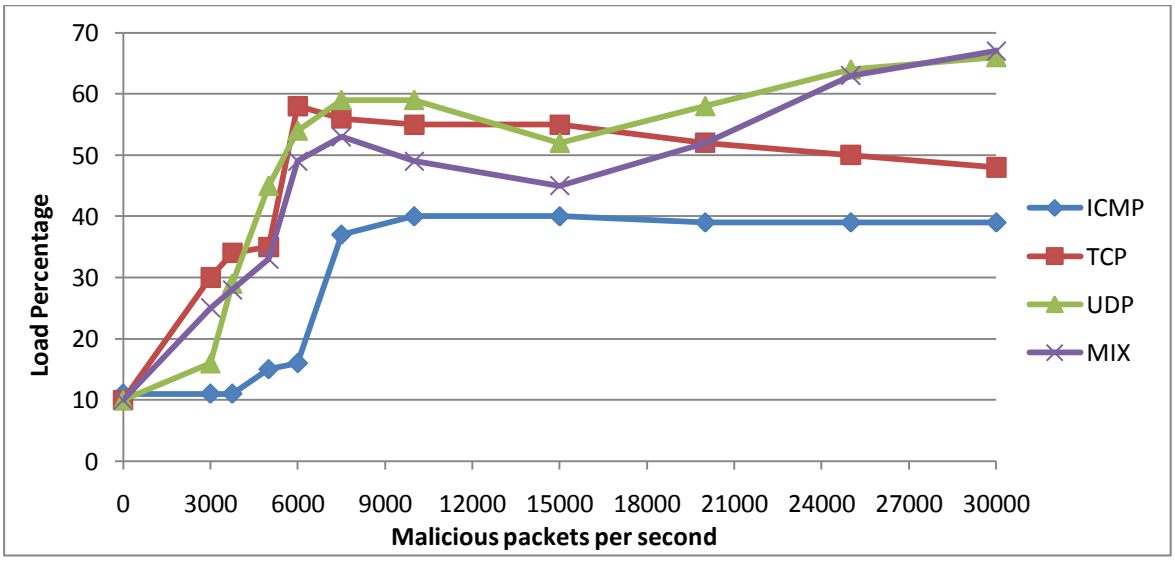

Figure 3: CPU Load Results

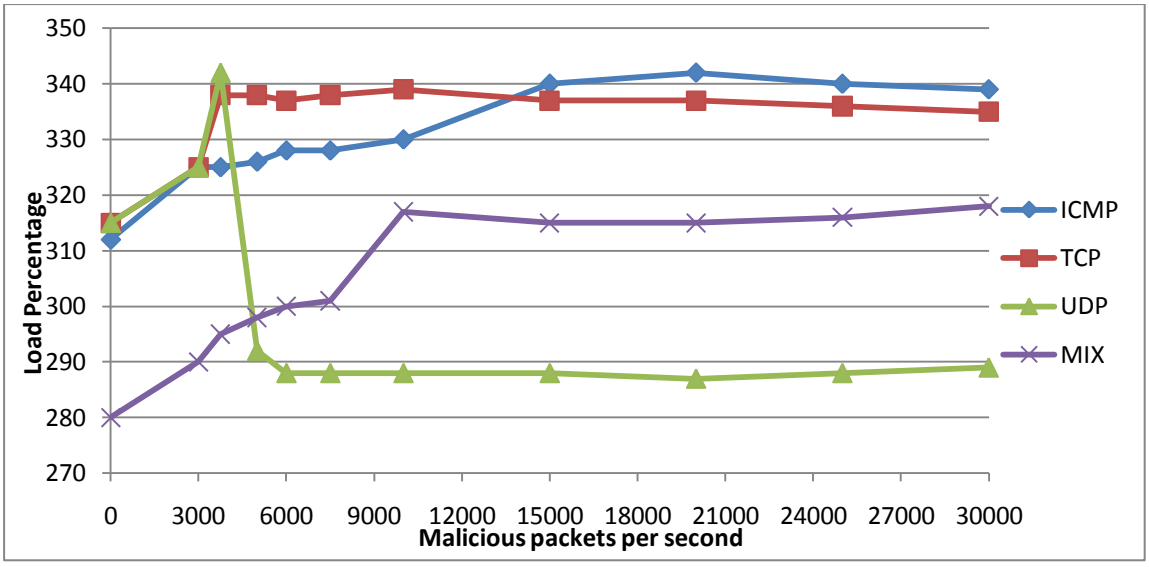

Figure 4: Memory Load Results

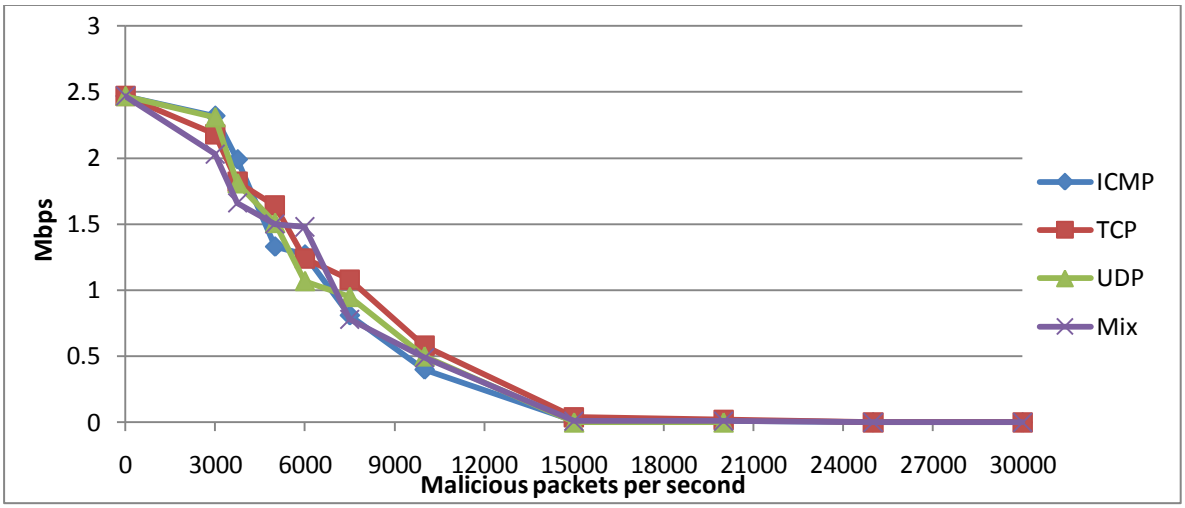

Figure 5: Available Bandwidth Results

5.3 Latency

The results for this experiment are in correlation of the hypothesis results. The latency grows as background traffic is growing. As for the available bandwidth, the experiment has been conducted without the IPS and without background traffic to evaluate the impact of the test-bed on the results. Without the IPS, the latency is of 0.4105 ms and with the IPS the results is 0.7435 ms. We can conclude that the test-bed does not bias the results because of the small impact of it on the experiment. The latency is of 2.219ms when only the background traffic is crossing the DUT. The latency is then multiply by two between each measure. When the rate of 6,000 pps is reached the latency for all protocol is in the area of 23 ms and when the attacking traffic display a rate of 7,500 pps the latency is around 71 ms for all protocols. After this augmentation, the latency stays stable for all protocols. This highlights that Snort might process to some optimisation when the latency is too important. The DUT might stop to use some feature to concentrate on all its resources on the routing of packets.

5.4 Rate Filter by Destination

The results of this experiment are in correlation with the expected results. The number of dropped packets grows, as the number of malicious packets is more important. The number of dropped packets are quickly increasing when the ratio of 5,000 attacking packets per second is reached. After 15,000 pps, the number of dropped traffic represent 50% of the traffic. For this experiment, the TCP protocols have the poorest results. This might be explained by the time required to analyse a TCP packets, because Snort shall analyse these packets along with the other packets of the same TCP session.

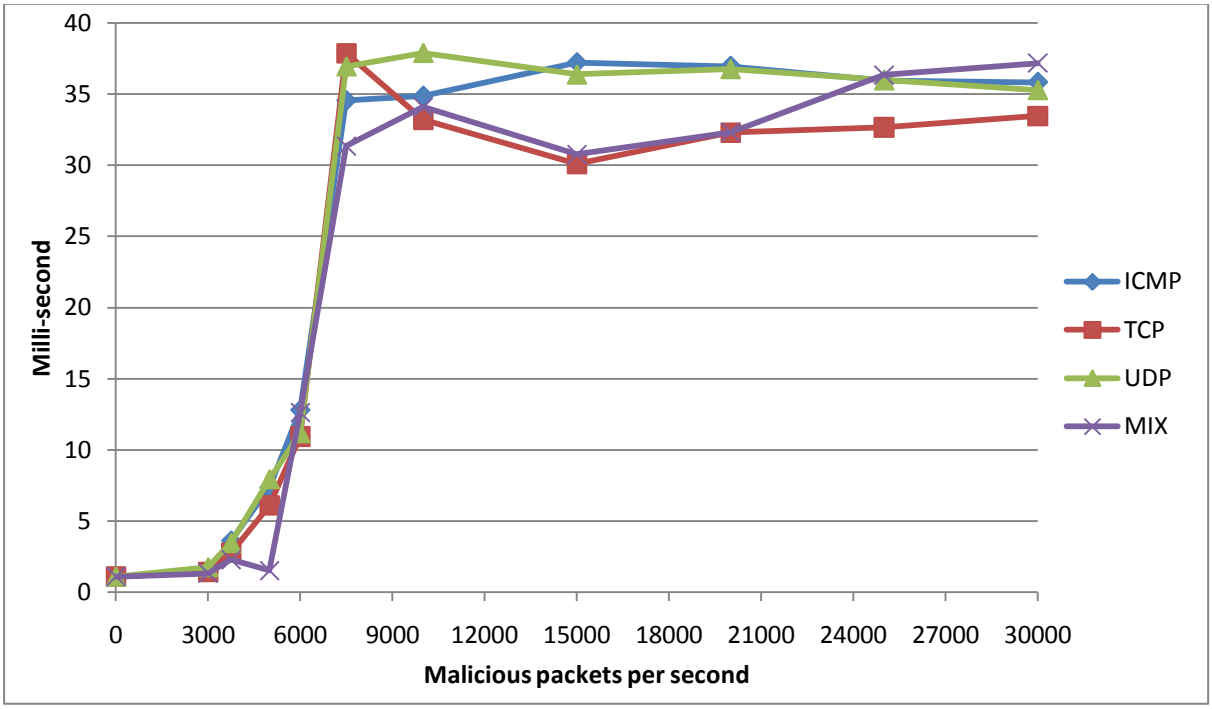

Figure 6: Latency Results

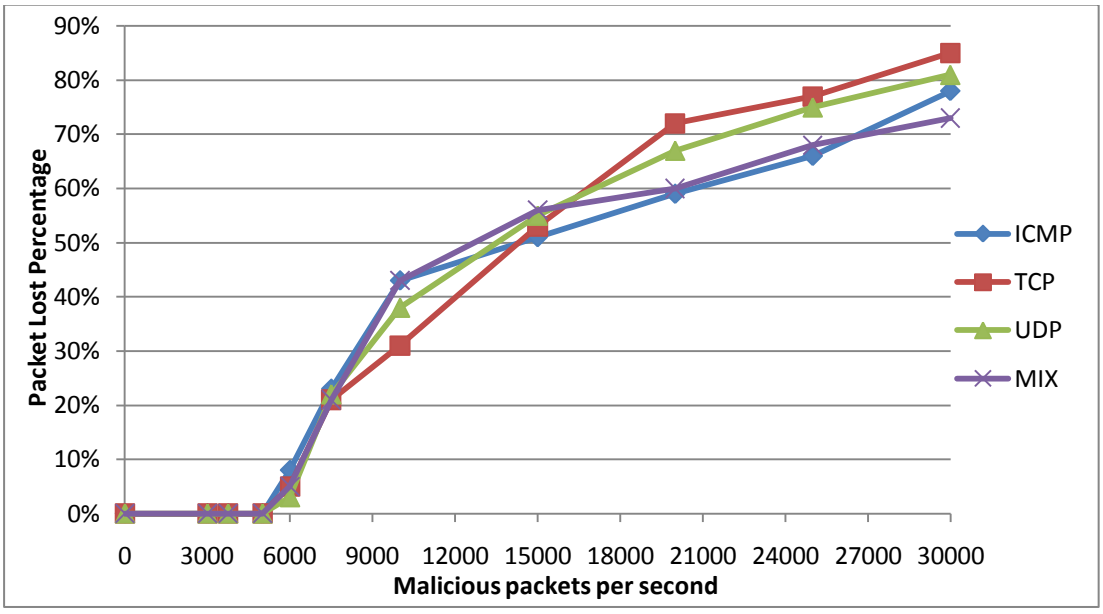

Figure 7: Rate Filter by Destination Results

5.5 Rate Filter by Source

The results of this experiment are confirming the expected results. The number of legitimate packets dropped is in correlation with the number of attacking packets. For all protocols, first packets are drop when the rate of malicious traffic is of 5000 pps. When this rate is multiply by three, more than 50% of legitimate traffic is dropped. For each protocol the rate of dropped traffic is low until 6,000 pps (around 5%) and percentage of dropped traffic is multiply by five with the rate of attacking traffic is 7 500pps. As for the previous experiment, the three protocols have slightly the same results. UDP and TCP have the poorest results as previously. The best results are for ICMP, this might be explain by the size of such packets, the system need less resources to process them.

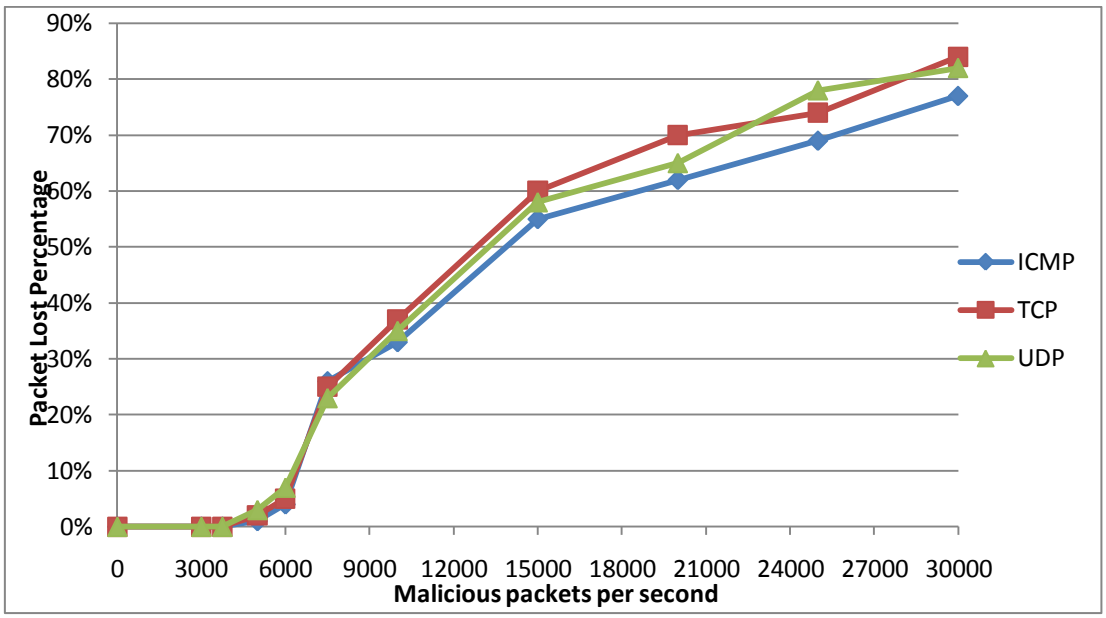

Figure 8: Rate Filter by Source Results

6 Analysis

The evaluation of the methodology shows that it is possible to generate background traffic along with attacking traffic. The maximum throughput of 1Mbps ensures no loss of data. It is poorest than the results found by Lo (2009). The results for his experimentation found that Snort could handle traffic of 60Mbps without dropping any packets in a virtual environment while Snort was running as an IDS (Version 2.7.0). This demonstrates the huge difference between the Snort capacities to handle traffic while running as an IPS than when it is configured as an IDS.

The results for the CPU load experiment show clearly the impact of the attacking traffic on the consumption of system resources. Snort start to drop packets when the CPU load is 16% for the ICMP protocol, 58% for the TCP protocol, 54% for the UDP protocol and, 49% when this is a mix of the three protocols. When the percentage of packets lost grows the CPU load of the ICMP protocol become stable at 39%, for the TCP protocol it decrease until 48%, only the UDP and Mix protocol continue to use more resources until respectively 66% and 67%. When Snort is

running as an IDS it starts to drop packets when 40% of the CPU is used and when 70% is reached a high number of packets are lost (Lo, 2009). We could conclude that the reason of why Snort start to drop packets is not because of the lack of resources but because of the design of Snort.

Regarding the two rate filters (by source and by destination) experiments they report that Snort begins to drop packets when attacking traffic reach 6 000 pps. When the rate filter is in destination mode 8% of packets are drop if the attack is using the ICMP protocol, 5% for TCP protocol, 3% for the UDP protocol and 5% when the three protocols are used. However, when the attacking traffic reach the rate of 30 000 pps 78% of packets lost for the ICMP protocol, 85% for the TCP protocol, 81% for the UDP protocol and 73% for the mix of the three protocols. If the rate filter is watching source IP addresses, results are the same with some slight differences. When the rate of malicious packet is of 6 000 pps, the rates for ICMP, TCP and UDP protocols are respectively of 4%, 5%, 7%. For the rate of 30 000 pps the results are respectively of 77%, 84%, 82%. It is possible to conclude that the break point of Snort happened with an attacking traffic of 6 000 pps while 80% of the maximum throughput of Snort is composed of legitimate traffic. Salah and Kahtani show that Snort (Version 2.8.1) running as an IDS has got a breaking point when the malicious traffic reach a rate of 40 Kpps.

7 Conclusion

The produced results show that Snort possesses effective mechanism to handle Distributed Denial of Service. It also highlighted the main weakness of Snort, which is the rate of packet lost. When the system reaches 16%, 58% or 54% of CPU load depending of the protocols used for the DDoS Snort starts dropping more and more packets. These loads are in correlation which the attacking traffic speed of 6 000 pps which is the breaking point of the IPS. When this rate is reached Snort could not ensure anymore that the legitimate user will still have access to the services of the trusted network neither that no malicious packets will reach its target. The system also offers a good reliability, and can handle an attack of seven hours without problems. It is important to note that during this experiment no other information has been taken so it might be possible that the number of packets dropped increase with the time of use.

After the analysis of these results it is possible to conclude that Snort could be use to protect a network where the traffic will never be grater that 1Mbps. However, this network should also not be a possible target to DDoS that will use attacking rates upper than 6 000pps. When this reached is overcome legitimates users might have to endure a denial of service, which was the aim of the attackers.

These results prove that the methodology could be used to evaluate IPS. However, different drawback should be pointed out of this methodology. The first is that the DARPA data set is used even if it has been demonstrated that is was not realistic enough traffic. Nevertheless, the methodology is open to receive any other data set, which could offer better results. The other

main drawback is the few numbers of DDoS that the device is test against, DDoS that are more complex have to be design and implemented. Finally, the methodology has strength like the use of metrics, which could be taken on any IPS. Furthermore, the methodology has been developed in a way that it could accept easily new metrics or even new type of threats.

References

Alessandri, D. (2004). Attack-Class-Based Analysis of Intrusion Detection Systems. *Ph.D. Thesis* .

Bounds, D. (2003). *Packit*. Retrieved May 6, 2010, from Sourceforge: http://sourceforge.net/projects/packit/

Càrdenas, A. A., Baras, J. S., & Seamon, K. (2006). A Framework for the Evaluation of Intrusion Detection Systems. *Proceedings of the 2006 IEEE Symposium on Security and Privacy (S&P'06)* , 77-92.

Chen, Z., Delis, A., & Wei, P. (2008). A Pragmatic Methodology for Testing Intrusion Prevention Systems. *The Computer Journal , 52* (4), 429-460.

Combs, G. (2006). *Wireshark*. Retrieved March 30, 2010, from Wireshark: http://www.wireshark.org/

Corsini, J. (2009). *Analysis and Evaluation of Network Intrusion Detection Methods to Uncover Data Theft, Master's thesis.* Edinburgh: Edinburgh Napier University.

Fink, G., O'Donoghue, K. F., Chappell, B. L., & Turner, T. G. (2002). A Metrics-Based Approach to Intrusion Detection System Evaluation for Distributed Real-Time Systems. *IPDPS '02: Proceedings of the 16th International Parallel and Distributed Processing Symposium* , 6-9.

Jones, R. (2007). *Care and Feeding of Netperf 2.4.X*. Retrieved March 30, 2010, from Netperf: http://www.netperf.org/svn/netperf2/tags/netperf-2.4.5/doc/netperf.html#Top

Jones, R. (2000). *Welcome to the Netperf Homepage*. Retrieved March 30, 2010, from Netperf: http://www.netperf.org/netperf/

Keshariya, A., & Foukia, N. (2010). DDoS Defense Mechanisms: A New Taxonomy. *4th International Workshop, DPM 2009 and Second International Workshop* (pp. 222-236). St. Malo: Springer Berlin.

Le, A., Boutaba, R., & Al-Shaer, E. (2008). Correlation-Based Load Balancing for Network Intrusion Detection and Prevention Systems. *Proceedings of the 4th international conference on Security and privacy in communication netowrks* (pp. 22-25). New York: ACM.

Lemonnier, E. (2001, June 28). Retrieved October 10, 2009, from Lemonier.se: http://lemonnier.se/erwan/docs/protocol_anomaly_detection.pdf

Lippmann, R. P., Fried, D. J., Graf, I., Haines, J. W., Kendall, K. R., McClung, D., et al. (2000). Evaluating Intrusion Detection Systems: The 1998 DARPA Off-line Intrusion Detection Evaluation. *Proceedings DARPA Information Survivability Conference and Exposition (DISCEX)* (pp. 12-26). Los Alamitos: IEEE Computer Society Press.

Lippmann, R., Haines, J. W., Fried, F. D., Korba, J., & Das, K. (2000). The 1999 DARPA off-line intrusion detection evaluation. *Computer Networks , 34* (4), 579-595.

Lo, O. (2009). Framework for Evaluation of Network-Based Intrusion Detection System. *Bachelor of Engineering with Honours in Computer Networks and Distributed Systems* .

Mahoney, M. V., & Chan, P. K. (2003). An Analysis of the 1999 DARPA/Lincoln Laboratory Evaluation Data for Network Anomaly Detection. *Proc. 6th Intl. Symp. on Recent Advances in Intrusion Detection* , 220-237.

Maxion, R. (1998). Measuring Intrusion Detection Systems. *The First Int. Workshop on Recent Advances in Intrusion Detection* , 1-44.

McHugh, J. (2000). Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory. *ACM Transactions on Information and System Security , 3* (4), 262-294.

Mell, P., & Scarfone, K. (2007). *NIST Special publication on intrusion detection systems.* Gaithersburg: National Institute of Standards and Technology.

Mirkovic, J. (2003). *D-WARD: Source-End Defense Against Distributed Denial-of-Service Attacks.* Los Angeles: University of California.

Mirkovic, J., & Reiher, P. (2004). A Taxonomy of DDoS Attack and DDos defense mechanisms. *Computer Communication Review , 34* (2), 39-55.

Mirkovic, J., Dietrich, S., Dittrich, D., & Reiher, P. (2005). *Internet Denial of Service: Attack and Defence Mechanisms.* Upper Saddle River: Pearson Education.

Mutaf, P. (2002). Defending against a Denial-of-Service Attack on TCP. *RAID International Symposium on Recent Advances in Intrusion Detection.*

Muuss, M. (1983). *The Story of the PING Program*. Retrieved March 30, 2010, from The Research Interests of MIKE MUUSS: http://ftp.arl.mil/~mike/ping.html

NSS Labs. (2006). *Network IPS Testing Procedure (V4.0).* Retrieved February 11, 2010, from NSS Lab: http://nsslabs.com/certification/ips/nss-nips-v40-testproc.pdf

Papadogiannakis, A., Polychronakis, M., & Markatos, E. P. (2010). Improving the accuracy of network intrusion detection systems under load using selective packet discarding. *Proceedings of the Third European Workshop on System Security* (pp. 15-21). Paris: ACM.

Parker, D. (2003). *Hping.* Retrieved February 12, 2010, from Hping: http://gd.tuwien.ac.at/www.hping.org/hping_conv.pdf

Pasquinucci, A. (2007). From network security to content filtering. *Computer Fraud & Security* , 14-17.

Paxson, V. (2003). *Bro Intrusion Detection System*. Retrieved February 26, 2010, from Bro-IDS.org: http://bro-ids.org/

Ranum, M. (2001). *Experiences benchmarking intrusion detection systems.* Retrieved February 08, 2010, from NFR Security White Paper: http://www.bandwidthco.com/whitepapers/compforensics/ids/Benchmarking%20IDS.pdf

Rowan, T. (2007). Intrusion prevention systems: superior security. *Network Security* , 11-15.

Sanfilippo, S. (2006). *hping*. Retrieved November 03, 2009, from hping: http://www.hping.org/documentation.php

Schuba, C. L., Krsul, I. V., Kuhn, M. G., Spafford, E. H., Sundaram, A., & Zamb, D. (1997). Analysis of a Denial of Service Attack on TCP. *Proceedings of the 1997 IEEE Symposium on Security and Privacy* (pp. 208-224). Washington: IEEE Computer Society.

Shafi, M. I., Akram, M., Hayat, S., & Sohail, I. (2010). Effectiveness of Intrusion Prevention Systems (IPS) in Fast Networks. *Journal of Comuting , 2* (6), 78-84.

Snort Team. (2009, October 22). *Snort Official Documentation.* Retrieved February 28, 2010, from Snort: http://www.snort.org/assets/125/snort_manual-2_8_5_1.pdf

Sommers, J. (2004(b)). *Harpoon: A Flow-level Traffic Generator.* Retrieved February 12, 2010, from The University of Wisconsin: http://pages.cs.wisc.edu/~jsommers/harpoon/

Sommers, J., Yegneswaran, V., & Barford, P. (2004(a)). A Framework for Malicious Workload Generation. *The 4th ACM SIGCOMM conference on Internet measurement* , 82-87.

Sommers, J., Yegneswaran, V., & Barford, P. (2005). *Toward comprehensive traffic generation for online ids evaluation.* University of Wisconsin: Tech Rep.

Srivatsa, M., Iyengar, A., Yin, J., & Liu, L. (2006). A Middleware System for Protecting Against Application Level Denial of Service Attacks. *Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware* (pp. 260-280). New York: Springer-Verlag.

Strasburg, C., Stakhanova, N., Basu, S., & Wong, J. S. (2008). *The Methodology for Evaluating Response Cost for Intrusion Response Systems.* Ames: Iowa State University.

Tanase, M. (2003, January 07). *Closing the Floodgates: DDoS Mitigation Techniques*. Retrieved October 12, 2009, from SecurityFocus: http://www.securityfocus.com/infocus/1655

TippingPoint. (2008, August 29). *The Fundamentals of Intrusion Prevention System Testing.* Retrieved November 02, 2009, from PreferredTechnology: http://www.preferredtechnology.com/support/whitepapers/download/IPSTesting.pdf

Turner, A. (2003). *Tcpreplay*. Retrieved November 02, 2009, from Tcpreplay: http://tcpreplay.synfin.net/trac/

VRT Team. (2009, October 23). *Download Snort Rules*. Retrieved October 30, 2009, from Snort: http://www.snort.org/snort-rules/?#rules

Wireshark Fundation. (2006). *Wireshark*. Retrieved April 18, 2010, from Wireshark: http://www.wireshark.org/

Xu, S.-X., Fu, X.-F., Hu, J.-X., Gao, B.-Q., & Su, L. (2008). Application of Load Balancing in Intrusion Detection System Based on Multi-Agent. *Computer Engineering , 34* (21), 184-186.

Yu, J., Fang, C., Lu, L., & Li, Z. (2009). A Lightweight Mechanism to Mitigate Application Layer DDoS Attacks. *The 4th International ICST Conference on Scalable Information Systems.* Hong Kong.

Yu, J., Li, Z., Chen, H., & Chen, X. (2007). A Detection and Offense Mechanism to Defend Against Application Layer DDoS Attacks. *Proceedings of the Third International Conference on Networking and Services* (pp. 54-61). IEEE Computer Society: Washington.