Table of Contents 1
stefan-mititelu-gcia-version4_final.doc..2

“The Security & Network Toolkit comes to the rescue”

GIAC Certified Intrusion Analyst (GCIA)

Practical Assignment

Version 4.0 – December 2004

Stefan Mititelu

Table of Content

1. Executive briefing 3

2. First phase analysis: identification of network environment at the time of the capture 5

3. Detailed network analysis of three most critical detects 13

3.1 Detect #1 – attacker 10.10.10.186 → victim 172.20.201.198 17

3.2 Detect #2 – attacker 10.10.10.122 → victim 192.168.17.135 24

3.3 Detect #3 – attackers identified with "bleeding-edge" rules 27

4. Analysis process 30

REFERENCES 31

BIBLIOGRAPHY 32

APPENDICES 33

Appendix A – various data from the first section 33

Appendix B – various ntop screenshots 52

Appendix C – network diagram + sample traffic 63

Appendix D – preliminary detects analysis 64

Appendix E – critical detects analysis 90

Appendix F – open source and/or free tools, used and useful in the practice of network intrusion detection 111

1. Executive briefing

The following material is the result of an analysis having been ordered by my client, University of X. The material/information available to me consisted of a set of trace files, whose number and content will be discussed in this paper. The analysis will be directed toward trace files of **network traffic**, after the said traffic was long gone from the network from which it had been recorded. The choice of which [group of] trace files, and the number of them, to be utilized during this analysis, was left to me, but the client required an explanation of the selection thusly made.

Due to the lack of in-house expertise, a diagram of the network was not available from the client. It was – thus - imperative, for my correct interpretation and analysis of data, to attempt, first, to determine the connectivity between systems, based on the available information, from the University's logs/traces. Building a network diagram was the object of the first (major) step undertaken in this research paper, and consisted in the utilization of various tools usually associated with network management. The client has also requested a graphical depiction of the most important aspects of identified traffic and/or shape of the network. The choice of open source and/or free applications was an obvious one, due to the financial constraints of an educational environment such as the University of X.

A second stage of this document was determined by the next major requirement of my client: that of identifying potentially malicious traffic, actual attacks and – if proven as such – possibly successful penetration, followed – in the end – by a more comprehensive analysis and discussion of the three most dangerous/critical threats revealed in this University network environment, from the trace files utilized. Part of this section will also be comprised of expert advise in regards to options for preventing any further/possible compromise of internal systems, based on the threat models previously identified.

The last section – per client's request – contains an overview/summary of the tools and techniques having been utilized during this analysis, as well as some additional information from my past exposure to similar applications.

I need to clarify at this stage a couple of issues, which will greatly impact my approach toward the assignment, as well as being a justification on what I believe could be a better service delivered to my client.

Firstly – having known that the log files offered by my client (of which I have chosen a limited subset, for reasons to be mentioned later in the paper) have been analyzed probably many times before, by other analysts, before me – and in combination with the very specific requirement of identifying the three most important malicious/intrusive attacks/exploits – I realized that the **possibility of redundancy in the detects being discussed is very high** (i.e. I thought of being very hard for many people to come up with “other” most important attacks, from the same trace files, analyzed over and over again). Secondly – based on another requirement which was my client asking for **add-on value** (i.e. innovation or knowledge addition, rather then mere repetition) in the material I was going prepare and deliver – I realized that my “virtual client money” (i.e. SANS expectations) would be better spent if I could come up with something somehow different than other previous papers.

So – considering the two issues above – I have decided to spend a few months in gathering all the tools I have previously used in similar assignments, then compile/make/install/configure/make them working on the system I was going to use – then provide insights on how such tools could be used in various stages of analysis. I have spent a lot of time in preparing a part of the paper (the first section) introducing all these tools, thus slightly modifying the expected “weight” of the assignment toward tools description and usage, vs. results of specific detects in too much detail.

The client – University of X – will consider the assignment correct and complete, if all the objectives mentioned above have been achieved.

2. First phase analysis: identification of network environment at the time of the capture

After having browsed through the trace files made available to me, from the University of X, via http://isc.sans.org/logs/Raw, I have determined that the best approach in regards to analyzing and advising my client, in her best interest, would be to utilize the latest trace files available. I have – thus – decided to proceed further with the download of the 2003.12.15.tgz tarball, and to start my analysis from that point:

$ wget[1] *http://isc.sans.org/logs/Raw/2003.12.15.tgz*

After having expanded the tarball in the working directory of my choice, on a machine completely isolated from the network (a MacOSX 10.3.4 platform[2]):

$ tar -zxvf 2003.12.15.tgz

I have proceeded, next, in determining the type of files I was dealing with:

$ for file in 2003.12.15.?; do file $file; done*

which resulted in the following information, for every file analyzed (fourteen in total):

tcpdump capture file (big-endian) - version 2.4 (Ethernet, capture length 96)

The above length of capture (96 bytes), if alone, would have been indicative of a default snaplength, using ***tcpdump***, for an OS like MacOSX (tested on mine), or Windows' ***windump***, or – from the tcpdump docs - for SunOS (http://tcpdump.org/tcpdump_man.html). Because the files are also almost identical in overall length, this makes me think that another utility (other than tcpdump) was used to capture the network traffic, which utility was able to *properly split (i.e. at datagram boundaries)* the captures, while rotating files and keeping them around 3MB. Such an output **could** have been created by using the following command (and utilities found at http://www.ethereal.com):

$ sudo tethereal -i <interface> -a filesize:3000 -b 14 -s 96 -w 2003.12.15

Having noted the above, the next step in the initial phase process was to obtain

[1] http://www.gnu.org/software/wget/wget.html

[2] The choice of OS was based on personal experience with robustness, reliability and reduced potential of virus/worms contamination on this platform, vs. the most commonly encountered Win32 environments. Its underlying "BSD" kernel allowed me, also, to port my favorite Unix/Linux tools, as well as taking advantage of some Mac specific office applications, to be used in preparing this paper. Overall MacOSX proved to be an unbeatable analyst all-in-one system.

the whole “picture”, by merging together the capture files. This was easily accomplished using ***mergecap*** (part of the same ethereal “package” of utilities):

```
$  mergecap –w 2003.12.15.cap * (in the directory of files)
```

The resultant “merged” file (2003.12.15.cap) was further processed to identify the “shape” of the network(s), i.e. the connectivity between devices, based on the pattern of communications found in the capture. The followings specific issues, concerning my client’s network at the time of the capture, were revealed:

a. The sniffer utilized by my client **had direct visibility of (“was on”) the 10.10.10.0/24 network** – obvious by the fact that, at the time of capture, it was capable of identifying and recording different MAC addresses, belonging to (paired with) individual IP addresses, for the systems on the same 10.10.10.0/24 LAN, and one-to-many MAC-to-IP addresses, when those IP addresses belonged to networks "“behind” MAC " a router (whose MAC address was "hiding" those other networks “behind” a unique interface). This observation was critical for correctly laying out my client’s network diagram.

b. The trace files have most likely been produced “unfiltered” (i.e. traffic was NOT run through an NIDS equipped with rules, meant to be triggered on suspicious traffic) – obvious from the fact that traffic such as Cisco Discovery Protocol, Spanning Tree Protocol, Ethernet loopback (protocol 0x9000) has been recorded, also.

NOTE: it is worth mentioning at this stage that all **networks** that appear in my client’s trace files (thus internal to their organization) are from the RFC1918 range[3], thus me seeing **absolutely no reason for further “sanitizing” those addresses**. For completeness, though, I feel I should mention here possible tools, which could easily “anonymize” IP addresses, when used in consolidating data for analysis: ***ipsumdump***[4]***,*** or ***tcpurify***[5]

Throughout this paper I have used various tools and scripts (obtained from different sources on the ‘net, or available with my OS, which I have modified to suit my specific needs), to obtain all the necessary information for my analysis (and – in some cases - even confirm/validate those findings). Following is a brief summary of those scripts and tools, used at this first stage (understanding the network and connectivities):

- ***tcpslice***[6] - as used in this example – will provide immediate information regarding the times of trace (first and last packet recorded), as follows:

```
$tcpslice –r 2003.12.15.cap

Tue Nov 18 12:57:23 2003          Tue Nov 18 14:15:57 2003
```

[3] http://www.faqs.org/rfcs/rfc1918.html

[4] http://www.cs.ucla.edu/~kohler/ipsumdump/

[5] http://masaka.cs.ohiou.edu/~eblanton/tcpurify/

[6] http://www.tcpdump.org/other/tcpslice.tar.Z

- though very space consuming, ***tcpflow***[7] will prove very useful in properly rebuilding sessions data (not only summaries, as I will be doing later on, but rather all capture information, segregated by “conversations”), then the network diagram. For that the following command was processed:

```
[tcpflow-info-dir]$ tcpflow –r 2003.12.15.cap
```

- script to reveal the different MAC addresses, associated with different IP addresses, proving the placement of the sniffer used by the my client on the 10.10.10.0/24 network. Piping the output twice through sort, may prove useful later on, when considering other aspects of the analysis (such as how “talkative” various systems on this network were, at the time of the trace):

```
$  tcpdump -enqr 2003.12.15.cap "ip src net 10.10.10.0/24" |cut -d " " -f 2,9 |cut -d "." -f
1-4 |sort |uniq -c |sort -nr  |sed 's/^[ ^t]*//;s/ /       /g'>mac-and-ip-on-10-10-10-0-
network.txt[8]
```

where: -e = revealed the link layer info (MAC addresses)
-n = no name resolution
-q = “quiet” (less protocol info)
-r = read capture file

- script to reveal the rest of MAC addresses (similar to the above):

```
$  tcpdump -enqr 2003.12.15.cap "ip and not dst net 10.10.10.0/24" |cut -d " " -f 4,11 |
cut -d "." -f 1-4 |sort |uniq -c | sort -nr |sed 's/,//;s/[:]*$//;s/^[ ^t]*//;s/ /       /g'>mac-and-ip-
not-destined-to-10-10-10-0-network.txt[9]
```

It is obvious from the above file that some systems, belonging to specific networks, communicate from “behind” 00:50:56:40:00:6d, which is an address within the range assigned to VMWare, Inc. (now an EMC Corporation company) – see [2]. To further analyze this info, I will grep for all the other entries:

```
$  grep –v “00:50:56:40:00:6d” mac-and-ip-not-destined-to-10-10-10-0-network.txt
>other-mac-to-ip.txt
```

[7] http://www.circlemud.org/~jelson/software/tcpflow/

[8] Modified version of the script found at http://www.giac.org/practical/GCIA/Patrik_Sternudd_GCIA.pdf, to eliminate leading spaces and add TABs, for better alignment with title column - complete file output mac-and-ip-on-10-10-10-0-network.txt in Appendix A

[9] Modified version of script from http://www.giac.org/practical/GCIA/Patrik_Sternudd_GCIA.pdf, to get a clean output (no “,”s or trailing “:”s, removed leading spaces and introduced TABs for better alignment with title row) – partial file mac-and-ip-not-destined-to-10-10-10-0-network.txt in Appendix A

which are all multicast or broadcast type addresses (NetRange: 224.0.0.0 - 239.255.255.255 CIDR: 224.0.0.0/4), proving – once again – that the network that our sniffer resided on – at the time of the capture – was in fact 10.10.10.0/24, and all the communication with other networks took place via a “router” whose Ethernet interface “toward” our LAN was the MAC address of a VMWare-based system.

```
$  tcpdump -enqr 2003.12.15.cap 'ip' | cut -d " " -f 2,9 |cut -d "." -f 1-4 |sort |uniq |sed 's/ /
/g'  >source-mac.txt
```

and

```
$  tcpdump -enqr 2003.12.15.cap 'ip' | cut -d " " -f 4,11 |cut -d "." -f 1-4 |sed 's/,/     /g'
|sed 's/[:]*$//g'  |sort |uniq >dest-mac.txt
```

are the last scripts I needed to run, to give me a picture of what systems are “behind” what specific MAC address. The first file (source-mac.txt) is the one that reflects traffic originated from a specific MAC source (and the IP behind it, when applicable), while the second file (dest-mac.txt) contains destination MACs (and – possibly – IPs behind those), when attempting to contact hosts outside the local network. Both files are shown in **Appendix A** (I have removed duplicate MAC addresses from the first column)

- ***ntop***[10] for the most complete “consolidated” view of data having been extracted from the trace file (see **Appendix B**, ***ntop*** information, for some of the data gathered this way, which is then to be found in the network diagram). This tool was the one having given me the most information in one single place, from overall traffic, top talkers, hop distance between various networks/systems, to individual stations characteristics, or even OS identification (as it relies on ***p0f***[11] and ***ettercap***[12] signature support, for passive OS fingerprinting) – and absolutely everything via a, easy to use web interface. Command:

```
$  sudo ntop -u ntop -m 10.10.10.0/24 -n -c -q -L -f 2003.12.15.cap
```

where -m = allowed me to assign a specific network to be local – in the context of our configuration (my client’s sniffer being local to the 10.10.10.0/24 one – see previous comments). **Sample data**, such as the one in **Appendix B**, was obtained from having run ***ntop*** as described above, then connecting a browser to http://localhost:3000.

NOTE: Because I have found ***ntop*** such a useful tool, I have decided to utilize the stable version available at the time of this paper (**3.0**), as well as the CVS development one (**3.1**) – soon to be released. Having done so, I have found that the

[10] http://www.ntop.org
[11] http://lcamtuf.coredump.cx/p0f.shtml
[12] http://ettercap.sourceforge.net/

CVS one had better signatures for OS fingerprinting, while lacking in some areas of client type of traffic. I have decided to present both results, and mark them as appropriate. The data obtained from combining both versions was of critical importance to this paper. When not specified, the version in the pictures from **Appendix B** had identical info obtained from both versions of ***ntop***.

I have – then – used two different tools for obtaining the "conversations" list, not because one wouldn't have done the job, as much as to offer other analysts example of the choices we have in our field (and because the second tool, though I have not seen it mentioned in any other papers, was found to be much better at exactly this stage of analysis):

- ***tcptrace***[13] for the total number of unique conversations:

```
$  tcptrace -n -t 2003.12.15.cap |sed s/":"/" "/g |awk '{print $2 $4 $5}' |sort |uniq -c |sort
–r > conv.txt
```

where: - all "pipes" I've run tcptrace through have allowed me to obtain, in the end, the total number of conversations between any pair of IP addresses, without regard to the port or protocol. Why would I have done this? Because it gave me an idea of the magnitude of connections (not always successful, sometimes just attempts) between any two addresses, as well as who those pairs were.

- ***ipsumdump***[4] for the conversations by port and protocol;

```
$  ipsumdump -psSdD -r 2003.12.15.cap |sort +1n -n |uniq >conv_ipsumdump.txt
```

where: -p = include protocl in the dump ("T" = TCP, "U" = UDP, "I" = ICMP)
-s = include source address
-S = include UDP or TCP port
-d and –D = destination, respectively
-r = read pcap file

NOTE: See **Appendix A** for the files related to the above "conversations"

- ***p0f*** [9] for a confirmation of hops (previously obtained from ntop), and – mostly – for identification of Operating Systems of some of the hosts involved (with some data obtained from ntop). Here below is an example of having used p0f for number of hops identification:

```
$  sudo p0f –l -s 2003.12.15.cap |sed 's/>//g' |awk –F "-" '{print $1,$3}' |grep distance
|sed 's/:/ /g' |awk '{print $1"<-->"$3"=="$6}' |sed 's/,//' |sort |uniq
```

[13] http://jarok.cs.ohiou.edu/software/tcptrace/tcptrace.html

- ***tcpdstat***[14] for a "consolidated" view of all transactions statistics, and some other very useful information by protocol, start and end time, packet size distribution, bytes, bytes/packet, etc. (see **stats.txt** in **Appendix A**),

```
$  tcpdstat 2003.12.15.cap > stats.txt
```

- ***tcpick***[15] is yet another useful tool for gathering statistics of TCP conversations. Here below (and **Fig. 1** in **Appendix A**) is an example of having used this tool for identification of established sessions. The colorized output is a very nice feature, having allowed me to better visualize parts of the traffic, when using it for network traffic analysis:

```
$ tcpick –r 2003.12.15.cap –C | grep ESTABLISHED
```

- one of the final steps consisted in a very "rudimentary" script, meant to reveal the Cisco ports on the switch, which made their presence "known". In order to reveal the "talkative" ports on the switch (their MAC addresses) as well as the three types of traffic (loopback, spanning-tree advertisements and CDP), all in one single "shot", I have combined the MAC resolution capability of tethereal (***-Nm***), with some piping to leave the MAC addresses in, and the type of traffic (see file **Cisco-switch-traffic.txt** in **Appendix A**)

```
$  tethereal.exe -Nm -r 2003.12.15.cap |grep Cisco |cut -d " " -f 5-9 |sort |uniq >Cisco-switch-traffic.txt
```

- while one other similar command offered me the list of the ports on the Cisco switch which have been mirrored/spanned[16] to the one where our sniffer was plugged in (see file **Cisco-switch-spanned-ports.txt** in **Appendix A**):

```
$  tethereal.exe -V -r 2003.12.15.cap |grep "Port identifier" |sort |uniq >Cisco-switch-spanned-ports.txt
```

NOTE: I would like to stop here for a second and re-emphasize something in the statement above: the STP traffic captured does NOT imply that the above ports are the only ones active on that switch, but rather the fact they are – actually – the ones ACTIVE AND HAVING BEEN SPANNED/MIRRORED to the port where our sniffer was plugged in, allowing it (the sniffer) to see the traffic "flowing" through the switch. In our case these ports appear to be part of **vlan3** (seen in the packet capture), which may assume that the rest of them were in the default-delivered Cisco **vlan1.** The port IDs

[14] http://staff.washington.edu/dittrich/talks/core02/tools/tools.html
[15] http://tcpick.sourceforge.net/
[16] http://www.cisco.com/warp/public/473/41.html

are in the file mentioned above.

If I wanted to get the port numbers as Cisco has named/numbered them (e.g. FastEthernet0/14), I would use the following command:

```
$  sudo tcpdump -vvvr 2003.12.15.cap 'ether[20:2]=0x2000' |grep Port-ID |sort |uniq
>Cisco-named-ports.txt
```

in **Appendix A**. This numbering scheme will actually be used in the diagram.

For completeness reasons I have – then – obtained the first three bytes of the MAC address in a list with the corresponding vendor names, by processing the file ***oui.txt***[17] with the following script:

```
$ awk '$1 ~ /^[0-9a-f][0-9a-f]\-[0-9a-f][0-9a-f]\-[0-9a-f][0-9a-f]/ {print $1, $3}' oui.txt |sed
's/-/:/g' > mac-to-vendor.txt
```

then use the above file to process (replace) the first three bytes of the MAC address in the source-mac.txt with the vendor name, using the script:

```
$ awk 'FNR==NR{a[$1]=$2;next} {b=$0;k=substr($1,1,8);if(k in
a)b=a[k]substr($0,9);print b}' mac-to-vendor.txt source-mac.txt > mac-to-vendor-to-
ip.txt
```

which file is to be found in **Appendix A.**

Same type of information could have been revealed looking at the **Host Information** screen produced with ***ntop.***

NOTE: Very important to reveal here is the existence of “virtual” machines. Many analysts have correctly identified the existence of VMWare, but I do not remember having seen anybody mentioning the existence of another virtual environment – **Microsoft’s Virtual PC**. Having revealed Microsoft as one of the “MAC vendors” (see file above) led me to research a little bit more into the history of this specific MAC “family” of addresses (00:03:FF), to determine that it was actually first registered with Connectix! This company was the original author of Virtual PC (now a Microsoft product). So in our environment we are dealing with two different virtual environments: VMWare and VirtualPC!

In regards to the VMWare deployment, I have searched extensively the Usenet groups (see example[18] - I have “played with dates ranging from 1981 (where Google’s

[17] http://standards.ieee.org/regauth/oui/oui.txt

[18] http://groups.google.com/groups?hl=en&lr=&as_drrb=b&q=00%3A0c%3A29+group%3Avmware*&btnG=Search&as_mind=30&as_minm=11&as_miny=1981&as_maxd=30&as_maxm=3&as_maxy=2003

archives start) to the present day, then narrow down the interval until the first posting mentioning a MAC whose first three bytes were **00:0C:29**). This way I have determined that the virtual machines with MAC address **00:0C:29** appeared only with the release of VMWare version **4.0RC1** (did not exist before). On the other hand, the other VMWare system having shown up in our trace (**00:50:56**) have shown up with MAC addresses from this "pool" since 1999[19], which makes me think that we either have various workstations configurations, utilizing addresses from the same pool (this implying two different algorithms for creating MACs, on same platform, one older, and one newer), or **at least two versions of VMWare running in our environment (4.x, as well as a much earlier version)**. Add to that the existence of **another virtual environment (Microsoft's VirtualPC)**, and **we could start thinking** of the possibility of some sort of security lab / testing facility environment inside the client's network.

Finally – based on all the information obtained above – I have built a diagram as close as data at hand has allowed me, of my customer's real network, which I have presented in **Appendix C.** In the diagram I have depicted only a few systems, considered at this stage (prior to the real detects analysis) as somehow significant (as either position on the network, or traffic related to them). Among those systems I have included the virtual machines mentioned earlier in the paper, including some of their roles.

I have chosen to include in this diagram, for **some of the known TCP ports,** having shown up in the trace, the corresponding client or server standard service name **in between quotes** (e.g. **"**POP3**"** client, or **"**SSH**"** server, or **"**NetBIOS**"** name resolution), as there is no guarantee – at this stage of analysis - about the nature of such traffic (valid or attempted malicious), or the fact that those ports really represented known services. The way I have identified such characteristics, of some of the **sample systems and services** depicted in the diagram, was by looking in the data available so far, choosing some systems of interest, then refining the data via simple filtering in ***tethereal*** for a combination of SYN <-->SYN-ACK <--> ACK, or for further detailing, ***tcpick***. This first phase of network analysis was meant to provide some information about the ports opened on some systems, without necessarily implying their usage in a malicious or valid way. I have also depicted in the diagram some of the "conversations" having taken place (using data obtained above, and documented in **Appendix A**). These are not necessarily the most critical ones, and are not – by any means – a comprehensive list (e.g. UDP services are not there, at this stage), as the more comprehensive security aspect of such communications is to be analyzed in the next phase.

[19] http://groups.google.com/groups?q=+%2200+50+56%22+vmware&start=10&hl=en&lr=&scoring=d&as_drrb=b&as_mind=12&as_minm=5&as_miny=1981&as_maxd=1&as_maxm=12&as_maxy=1999&selm=zRkl2.6694%24465.1036%40news.rdc1.sdca.home.com&rnum=14

3. Detailed network analysis of three most critical detects

I will – first – summarize the approach used to identify the malicious traffic, and then address the three most critical aspects.

The **local** dates and times of the whole trace have already been revealed in the file **stats.txt** (**Appendix A**):
StartTime: Tue Nov 18 **12:57:23** 2003
EndTime: Tue Nov 18 **14:15:57** 2003

An interesting piece of information could be obtained by running the following command (where –tttt forces the **UTC time**)

```
$ tcpdump –tttt –n –r 2003.12.15.cap
```

which revealed the beginning and end of the trace in **UTC format**:
2003-11-18 **18:57:23**
2003-11-18 **20:15:57**

Based on the two pieces of information above, the difference is of 6 hrs, which narrows down a little the location of the University campus where the trace was taken.

The next major step in this analysis was the identification of malicious activity, via utilization of the security tools known to provide information related to intrusion detection, as well as capability of consolidating results, and – if possible – categorizing such findings. The group of tools capable of addressing these needs, that I have chosen, was comprised of: ***snort***[20]***,*** as the Network Intrusion Detection System (NIDS), two well-known data processing/correlation front ends for it: Analysis Console for Intrusion Databases (***ACID***[21]) and ***sguil***[22]. For these tools to work I have also had to install ***MySQL***[23] and ***Barnyard***[24]. At the time of this writing the stable version of ***snort*** was 2.1.3. The regular rules[25] used where the ones dated October 2004. In conjunction with those I have also utilized rules available that date, from the newly developed Bleeding Snort site[26]

For ***snort*** the main variables in the configuration files (*$HOME_NET, $EXTERNAL_NET,* etc.) were setup to *"any".* For the rest – except for specified

[20] http://www.snort.org
[21] http://acidlab.sourceforge.net
[22] http://sguil.sourceforge.net
[23] http://www.mysql.com
[24] http://sourceforge.net/projects/barnyard
[25] http://www.snort.org/dl/rules/snortrules-snapshot-2_1.tar.gz
[26] http://bleedingsnort.com/

otherwise (as in the following comments), I have used the default entries, as delivered with the ***snort*** tarball. I have – then – constructed four different configuration files:

- a group of two configuration files, used to determine if there is any difference between the capture file processing with the ***stream4*** preprocessor enabled, or not. Those files were named *snort-GIAC-no-stream4.conf* and *snort-GIAC-with-stream4.conf* (with the only difference between the two being the commented-out ***stream4*** option in the former, vs. the latter). The reason for doing this resides in the possibility of the trace file(s) having been the result of a previous run of snort, with binary logging (though I was already against this possibility, earlier, in a "semi-formal" way, when I mentioned the "slicing" of the trace at precise boundaries, not possible with ***snort***), in which case some alarms may be missed, during a second run[27].

I have – then run the following commands:

```
$ snort –d –c /etc/snort/snort-GIAC-no-stream4.conf –l snort-log-without preprocessor
–r 2003.12.15.cap
```

and

```
$ snort –d –c /etc/snort/snort-GIAC-with-stream4.conf –l snort-log-without preprocessor
–r 2003.12.15.cap
```

and obtained the results shown in **Appendix D**, first part. The results prove that the capture files that made 2003-12-15.cap were not previously processed with another ***snort*** run.

- a set of configuration files (one for each of the tools mentioned), specifically designed for usage with ***barnyard, Mysql, sguild, sguli_tk, sensor_agent.tcl*** and ***log_packets.sh.***

NOTE: It is worth noting here that the setup of all needed components could constitute the subject of an entire paper! The closest to proper setup that I have found was the set of instructions maintained by Richard Bejtlich[28], with some changes required on Itcl and Iwidgets. See **Appendix D**, also, for the **proper sequence** of running ***sguil*** and associated tools, which was critical in getting the desired results, when reading from the trace file.

- a set of configuration files specifically designed to run ***snort, Mysql*** and ***ACID*** – the whole set of tools setup being "loosely" based on a non-Mac platform set of instructions from [3].

The purpose of running ***snort*** under a ***sguil,*** then under ***ACID*** "environment" was to produce easy to use (and – with two sets of tools "going after" the same set of data – appropriate confirmation/justification of findings) information, to be able to choose the most critical detects. A very interesting paper found here [4] made a strong point

[27] http://cert.uni-stuttgart.de/archive/intrusions/2003/01/msg00018.html
[28] http://sguil.sourceforge.net/sguil_guide_latest.txt

about one of the above tools (***sguil***), as being extremely useful in identifying the answer to the "now what" question, when dealing with the results of a ***snort*** run. A screenshot of each tool has also been included in **Appendix D**, as samples for those having never seen these tools "in action".

Another two-step process (as simple as may appear, but incredibly useful) in the process of identifying the most malicious attack, was to run the trace file through - **first** – a Unix utility called ***strings***[29]***,*** for a high level overview of potentially "questionable" strings:

```
$strings 2003.12.15.cap | sort | uniq > strings_2003-12-15.txt
```

The above file (I found no reason to show it in this paper, as it represents an intermediary stage, full of redundant or uninteresting – in my opinion -stuff) is then opened and analyzed for strings of interest (really labor-intensive, manual process – but where an analyst experience would have a real "saying" – as the strings chosen are the ones which may reveal one's experience in protocol and operating systems behavior). Without assuming as having obtained a perfect output, I have processed this file and produced a more reduced, **almost** redundant-free one, containing strings I have personally found of being interesting, in the trace. I have – then – reran this file through ***sort*** and ***uniq*** and obtained the file **interesting-strings-2003-12-15.cap,** shown in **Appendix D.**

I have – then - run **some** of the strings, through an Internet search, to try and determine if they had security "history" implications. See examples of such findings in **Appendix D**. I have left out from this process the strings with obvious meaning - such as */bin/sh*, *root*, *nessus*, *iss, /etc/passwd, shadow, Virtual PC* (**NOTE:** good guess in the initial phase of this project – this string just came to support my initial assessment regarding the utilization of another virtual environment, besides VMWARE), etc., which I processed directly in the next step (***ngrep***).

The strings identified as mentioned above (either as obviously questionable, or via links to security issues, obtained through Internet searches), were run through ***ngrep***[30], which is a "network" version (capable of reading ***libpcap*** files) of the popular Unix program ***grep***. In this case I have used this tool to determine the systems having exchanged such "interesting/questionable strings", during their conversations – i.e. full "systems dialogues" (i.e. two stations sending traffic in both directions (requests-replies), implying access from potential intruder(s), to potential victim(s)). The format of the command is:

```
$ sudo ngrep –I 2003.12.15.cap –q –i 'regex-or-string-of-interest' <bpf-filter>
```

where: - q = "quiet" (no ### printing) mode
- i = case insensitive match

[29] http://linux.about.com/library/cmd/blcmdl1_strings.htm
[30] http://ngrep.sourceforge.net/

Examples of such findings are also presented in **Appendix D**.

So – after having utilized all the tools mentioned above, from the beginning - (***ntop, ACID, sguil, strings, ngrep***, etc.), and based on all the data, in various formats, obtained so far (with the examples presented in **Appendices A-D**), I have drawn the following conclusions:

- the systems on the network 10.10.10.0/24 are at the origin of a very intensive (and mostly noisy) scanning and penetration/exploitation/malicious traffic attempts (see summary of numbers and timings above, and in **Appendices)**, directed at the 172.x.x.x and 192.168.y.y networks. Considering this traffic takes place on an University premises, and without (apparently) any restrictions, it appears that the 10.10.10.0/24 network of machines is a (V)LAN specifically setup for some class or lab in Security/Penetration Testing. Other analysts (Dana Weber's [5], or Ian Eaton's [6]) seem to have reached similar conclusions, based on same, or parts of the same trace files group.
- in order to provide **the three most critical detects**[31] I have relied upon the consolidated data provided by ***sguil*** (which seems to be the best interface into the snort logs, for proper categorization), ***ACID***, and findings of the previous ***strings + ngrep*** processing
- I will NOT consider critical detects those based simply on sheer volume, but rather those (potentially) revealing actual (or very close to happen) intrusion. An example of the **not-to-be-considered** - though having made it to the top of reports, in volume, in tools like ***ACID*** - is the scan carried out by 10.10.10.3 – with a combined number of over 22,000 alarms for **scan-null**[32] and **scan-null-with-tiny-fragments**[33]**.**

  DISCLAIMER: the above statement is not meant to deny the possibility of a **"concerted" work of all the 10.10.10.0/24 "live" systems**, in the attempts against the other networks, with some "making noise" or conducting only reconnaissance scans, in cooperation with others – stealthier – meant to carry out the real attacks.

NOTE: As previously explained, I have chosen to - somehow - modify the "weight" of the paper, by moving the "core" of the analysis, from the details for the three mandated detects (which is the most common approach of all analysts' papers I have read so far) to the preparatory stage. I strongly believe this could be of use, especially under the conditions of the trace files chosen, as the processing of data for proper understanding of the environment could be considered at the same level of importance (if not more!), as the detects themselves. This will **leave me with less "space" for detects**, but with the conviction that the material up until now (tools and analysis) could be the add-on value my client was expecting.

[31] http://www.giac.org/GCIA_assign_40.php
[32] http://www.snort.org/snort-db/sid.html?sid=623
[33] http://www.bleedingsnort.com/bleeding-all.rules - look for "nmap -f –sN"

3.1 Detect #1 – attacker 10.10.10.186 → victim 172.20.201.198

Reason this detect was selected

The attack was chosen as one of the most critical ones, based on information from the following sources (see **Appendix E** for examples):

- potential malicious nature of strings found in the trace, via ***ngrep*** (note the usage of *–A <number of lines after>* and *–x* (dump hex and ASCII output) as new options, as more detailed information is desired at this stage, vs. the initial "run"). At this time another string worth investigating is **'^2..'** which is actually a regular expression considering **beginning of line**, with any **successfully completed** FTP status codes, for example[34]

- alerts generated in ***snort***, and consolidated in ***sguil*** at the higher category levels[35]

- alerts generated in ***snort*** and consolidated in ***ACID***, which database is then queried for the highest levels of alerts

Looking at the data obtained via the tools mentioned above (with some samples depicted in **Appendix E**) it appeared clear that the system with the IP address 10.10.10.186 was among those having depicted successful connections to – in majority – the "remote" system 172.20.201.198, in ways indicative of possible intrusion.

Which rule(s) generated the detect?

The "attack" (i.e. attacker's system against victim's one) – in itself – is not limited to a unique intrusion type, but rather the result of a longer process, involving possibly steps such as probing -> "testing" -> intrusion -> attempts for elevating privileges -> attempts to connect to other systems, from the victim, etc. Considering the requirements of my client, spelled out in the "statement of work"[36], at this stage I will have to **focus to one of the events/detects**, probably the most remarkable in the attack process, and describe it further.

So – first – let's isolate the traffic exchanged by these two systems:

```
$ tethereal -r 2003.12.15.cap –R 'ip.addr == 10.10.10.186 and ip.addr ==
172.20.201.198' –w 10.10.10.186-172.20.201.198.cap
```

We can see that the start of communication between the attacker and the victim, as captured in the trace files given by my client, is "into" an existing FTP session, about whose initial establishment we had no information. I have decided to process the capture file above – from this point on – using ***ethereal*** (the GUI

[34] http://help.globalscape.com/help/support/Error_Codes/FTP_Codes.htm#200
[35] http://sguil.sourceforge.net/index.php?page=incident_categories
[36] http://www.giac.org/GCIA_assign_40.php

equivalent of ***tethereal***), for ease of visualization. I have started “backwards”, from the end of the file, looking for “interesting” (i.e. possible detects) information, assuming that the attacker is most likely to finish communication with a victim, once having reached the purpose of the attack. I have saved in **Appendix E** the ethereal screen capture of a TCP sessions, from the end, backwards. As it can be easily seen from that capture, using this time a display filter in ***ethereal***: *(ip.addr eq 172.20.201.198 and ip.addr eq 10.10.10.186) and (tcp.port eq 21 and tcp.port eq 48313)* the attacker ends up with a root access, after which he is just “poking around” (listing files, then exiting “cleanly”?!?). This is enough of a good reason to look at any rules which may have been triggered by this malicious traffic. For this, I went back into ***ACID***, and ran a query isolating the pair of IP addresses and TCP ports above (see screenshots of the query and rules identified, in **Appendix E**).

The four rules triggered were:

http://www.snort.org/snort-db/sid.html?sid=553 ➔ POLICY FTP anonymous login attempt:
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"POLICY FTP anonymous login attempt"; flow:to_server,established; content:"USER"; nocase; pcre:"/^USER\s+(anonymous|ftp)/smi"; classtype:misc-activity; sid:553; rev:7;)
Summary: The event is generated when an attempt is made to log on to an FTP server with the username of "anonymous".
Impact: Information gathering or remote access. This activity may be a precursor to navigating through the accessible directories on the anonymous FTP server to do reconnaissance of the server. Alternately, this may be a precursor of attempting an exploit, such as a buffer overflow, that may permit remote access to the vulnerable FTP server.

---

http://www.bleedingsnort.com/bleeding.rules ➔ BLEEDING-EDGE FTP Serv-U Local Privilege Escalation Vulnerability:
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"BLEEDING-EDGE FTP Serv-U Local Privilege Escalation Vulnerability"; content:"site exec"; nocase; rawbytes; reference:url,www.securiteam.com/windowsntfocus/5YP0F1FDPO.html; classtype:misc-activity; flow:to_server,established; sid:2001210; rev:3;)
Summary: rule supposed to be triggered by a pattern specific to the exploit posted at: http://www.securiteam.com/windowsntfocus/5YP0F1FDPO.html

---

http://www.snort.org/snort-db/sid.html?sid=1971 ➔ FTP SITE EXEC format string attempt:
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP SITE EXEC format string attempt"; flow:to_server,established; content:"SITE"; nocase; content:"EXEC"; distance:0; nocase; pcre:"/^SITE\s+EXEC\s[^\n]?%[^\n]*?%/smi"; classtype:bad-unknown; sid:1971; rev:4;)*
Summary: Someone has attempted a format string attack that is successful against

the SITE EXEC command on vulnerable versions of WU-FTPD.
Impact: Severe; **remote root compromise** possible if user is running a version of **WU-FTPD prior to 2.6.2 as root**.

---

http://www.snort.org/snort-db/sid.html?sid=361 ➔ FTP SITE EXEC attempt:

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP SITE EXEC attempt"; flow:to_server,established; content:"SITE"; nocase; content:"EXEC"; distance:0; nocase; pcre:"/^SITE\s+EXEC/smi"; reference:arachnids,317; reference:bugtraq,2241; reference:cve,1999-0080; reference:cve,1999-0955; classtype:bad-unknown; sid:361; rev:15;)
```

Summary: This event is generated when a remote user executes the SITE EXEC command in a session with an internal FTP server. This may indicate an attempt to exploit a vulnerability in the SITE EXEC command in **wu-ftpd version 2.4.1**.
Impact: Arbitrary code execution, **leading to remote root compromise**. The attacker must have a valid, non-anonymous FTP account on the server to attempt this exploit.

Based on the information provided in the above links, we need to determine one more thing: the version of the FTP server. This could be easily achieved by ***ngrep***-ing for the string “wu-ftpd”, in the hope of finding the version number:

```
$ sudo ngrep -x -I 2003.12.15.cap -q -i 'wu-ftpd' host 10.10.10.186 and 172.20.201.198
input: 2003.12.15.cap

T 172.20.201.198:21 -> 10.10.10.186:48253 [AP]
77 75 2d 66 74 70 64 2d    32 2e 36 2e 30 0a          wu-ftpd-2.6.0.
```

All of the above rules, even though **appearing** as having detected the attack, have – in fact – **not revealed the actual intrusion**, but only precursors of it (malicious, nonetheless, but without direct result – more of a “probing” for various known ftp-related flaws). Lots of papers I read, about this, seem to have **incorrectly** related the actual intrusion, to rules triggered by unsuccessful attempts, as the ones above. In fact**, there is no exact rule from the set available by default in October-November-December 2004, having been TRIGGERED, which actually matched the exploit** (!!!), and that – I believe - is because **the rule which would have actually applied** – either being considered very specific, or forcibly expired a while ago – **was not included in the default ones** (should have been in the **ftp.rules**), **anymore**. That rule is[37]:

```
alert TCP $EXTERNAL_NET any -> $HOME_NET 21 (msg: "IDS287/ftp_ftp-wuftp260-venglin-linux"; flags: A+; content: "|31c031db 31c9b046 cd80 31c031db|"; classtype: system-attempt; reference: arachnids,287;)
```

As a confirmation of the above statement is the fact that the payload of last packet

[37] http://whitehats.com/cgi/arachNIDS/Show?_id=ids287&view=signatures

before the victim’s “defeat” contains the hex string mentioned in the above rule:

```
$ sudo ngrep -I 2003.12.15.cap -x -A 6 -q -X 31c031db31c9b046cd8031c031db host
172.20.201.198 and 10.10.10.186 and tcp port 48313
input: 2003.12.15.cap

T 10.10.10.186:48313 -> 172.20.201.198:21 [AP]
  31 c0 31 db 31 c9 b0 46    cd 80 31 c0 31 db 43 89    1.1.1..F..1.1.C.
  d9 41 b0 3f cd 80 eb 6b    5e 31 c0 31 c9 8d          .A.?...k^1.1..

T 172.20.201.198:21 -> 10.10.10.186:48313 [A]

T 10.10.10.186:48313 -> 172.20.201.198:21 [AP]
  69 64 3b 0a                                           id;.

T 172.20.201.198:21 -> 10.10.10.186:48313 [AP]
  75 69 64 3d 30 28 72 6f    6f 74 29 20 67 69 64 3d    uid=0(root) gid=
  30 28 72 6f 6f 74 29 20    67 72 6f 75 70 73 00 00    0(root) groups..
  09 04 00 00 6e 6f 62                                  ....nob

T 10.10.10.186:48313 -> 172.20.201.198:21 [AP]
  0a 6c 73 0a                                           .ls.

T 172.20.201.198:21 -> 10.10.10.186:48313 [AP]
  62 69 6e 0a 62 6f 6f 74    0a 64 65 76 0a 65 74 63    bin.boot.dev.etc
  0a 68 6f 6d 65 0a 6c 69    62 0a 6c 6f 73 74          .home.lib.lost

T 10.10.10.186:48313 -> 172.20.201.198:21 [A]
```

where: -X <expression> = match the expression in HEX

Bingo! Root shell, right after the attack!

Probability the address was spoofed

In my opinion: none. The attack requires full TCP connectivity (i.e. “sessions”), for the results to be analyzed by the attacker, then to further proceed with the intrusion.

Description of detect

The attack whose detect was isolated above (as stated earlier - part of a larger malicious process having been carried out by the system at 10.10.10.186, against the 172.20.201.198 one) was used to allow the attacker full root privileges of the victim’s machine, via flaws in the implementation of the FTP protocol, in the application **wu-ftpd**, for a very specific version of it (2.6.0). The exploit is delivered through a *SITE*

EXEC command*,* with “user’s input going directly into a format string for a *printf function”. Max Vision, of the “Whitehats fame”, provided more details in his analysis[38]. A working exploit is available “courtesy” of Venglin[39]

[38] http://www.whitehats.com/cgi/arachNIDS/Show?_id=ids287&view=research
[39] http://packetstormsecurity.org/0006-exploits/bobek.c

Attack mechanism

The mechanism of attack has been known for a while, and relates directly to the following "pre-requisites":

- wu-ftp version 2.6.0
- configuration allowed for **anonymous** access to the FTP server, or the default **ftp** user enabled (default for anonymous access, in wu-ftpd implementation)

Looking back into the "filtered" trace – using a display filter with ethereal, this time: *(ip.addr eq 172.20.201.198 and ip.addr eq 10.10.10.186) and (tcp.port eq 21 and tcp.port eq 48313)* the attack process is very obvious:

1. Login with a username of **ftp** and a password of **mozilla@**
2. All sorts of attempts for gaining access (which actually triggered the rules mentioned earlier), based on various *SITE EXEC* commands and payload delivered with those – none successful until the
3. "Coup de grace"[40] via the actual exploit, as mentioned earlier

The only real question left to be answered would be: "**Why?**" – and my answer: if assuming a security/pen-testing ab environment, where everything is to be tried – than the reason is simply to determine which of the many exploits attempted is to produce the most damage, and – probably – learn how to protect, because of that. If the reason would have been "more" malicious, we should have found in the trace more damaging actions, followed by the root shell access, and not only a listing of files available;

NOTE: as easily seen above, and in the followings, due to space constraints, as a direct result of much deeper analysis in the preparatory phase, I have decided to limit the "verbiage" at a minimum, and provide more directed input, and links where and when applicable.

Correlations

I could not find any correlation among the GCIA papers having considered the above capture file, and the detect, but I have found numerous discussions related to this specific exploit:

- a still-candidate (since 2000 ?!? – probably why ***snort*** is not distributing the previously mentioned rule with its own) CVE mention[41]
- a GCIH paper, having discussed the issue [7]
- and – at last, but not at least – a thorough discussion, and an opportunity for having seen full payload (i.e. not limited to 96 bytes, like our trace) of a similar trace, in a

[40] http://dictionary.reference.com/search?q=coup%20de%20grace

[41] http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2000-0574

Honeynet Challenge Scan of the Month #19[42] This is actually the place to read a lot of information about the exploit, in the write-ups from the Honeynet Project members.
- within the capture, itself, utilizing the same HEX string search (via ***ngrep***) as above, I have been able to find other "pairs" of attacker-victim, attempting similar exploit:
10.10.10.196 <--> 172.20.201.198
10.10.10.165 <--> 172.20.201.135
10.10.10.228 <--> 172.20.201.135
which comes to prove – yet again – that this whole 10.10.10.0/24 network of systems is "practicing" security intrusion (pen-testing?!?) against (in this specific case) the 172.20.201.0/24 network.

Evidence of active targeting

It is obvious – by now – that **for this specific detect/exploit**, the attacker was definitely targeting the victim's machine. Even in a more broader view (the attacker's actions throughout the time seen in the capture file) there were 151 individual rules triggered, having 10.10.10.186 as "attacker" (see **Appendix E** screenshot of samples of such), in majority targeting 172.20.201.198, thus implying a concentrated effort of our attacker almost exclusively in one direction → toward the identified victim.

Severity

Regardless of the purpose/existence of the attacker (lab environment, or production pen-testing machine), the fact that the victim is "weak", as far as the specific service (**wu-ftp**) is concerned has to be accounted for, and this is why I approached ranking of severity levels as follows:

Criticality = assumed to be related to the content, I have approached as such, carrying out the following steps:

$ sudo ngrep -I 2003.12.15.cap -x -q '' src host 172.20.201.198 | more*

→ to determine if any "visible", possibly critical strings traversed the network, outbound from the victim
Having "stumbled" across the string "important-proposal", I have then narrowed down further the search, doing:

$ sudo ngrep -q -x -A 4 -I 2003.12.15.cap important 'host 172.20.201.198'

which revealed the following – in my opinion – part of the conversation:

```
T 10.10.10.186:48253 -> 172.20.201.198:21 [AP]
  76 69 20 69 6d 70 6f 72    74 61 6e 74 2d 70 72 6f    vi important-pro
```

[42] http://project.honeynet.org/scans/scan19/

```
70 6f 73 61 6c 2e 74 78    74 0a                    posal.txt.
```

which implies attacker’s attempt (**capable of holding a shell on the victim’s machine**, as seen above) to modify a file possibly critical to my client’s business. I would – thus – associate to this attack a maximum criticality level (**5**)

Lethality: the attack leads to root shell, so the level is obvious **5**

System countermeasures: nothing appears to keep the intruder(s) out, from a host perspective, related to the attack being discussed.

NOTE: There is one very minor exception, in regards to what could be seen in the capture file, and that is the existence of an SSH install, on the victim’s machine (detected via the previous ***ngrep***-ing process, as well as ***tcpdump***-ing the capture with a filter of *‘src host 172.20.201.198 and src port 22’*) which may remotely imply some consideration toward security (vs. – for example – having installed telnet).

Because of the note above, I would be slightly more lenient in this case (i.e. “could have been worse”) – then – and associate this aspect with a level of **2**

Network countermeasures: absolutely none that I could see, with the exception of the sniffer itself, which is really not a countermeasure per-se ➔ level **1**

So – **SEVERITY FOR DETECT #1 = (5 + 5) – (1 + 0) = 9**

Defensive recommendations

Based on the analysis above, there are obvious things that may be attempted, in order to better protect the victim:

- at the network level: a firewall capable of stateful inspection, or even a router with appropriate Access Control Lists, preventing indiscriminate access to the victim’s network. Possible rules/ACLs may also be designed around time thresholds, to avoid massive scans, if the router/firewall solution chosen would support them.
- at the host level:
    * if the client **needs** to run an anonymous ftp server, around **wu-ftp** (perhaps because of advantages offered by this, such as: various configuration options, with tighter control of anonymous uploads, and chroot()-ed environment for guest users, into their home directories) – the I recommend latest version of such, and keeping up with the patches[43]
    * if the client needs to run **ftp**, but not necessarily **wu-ftp**, then I would recommend more secure solutions, such as **vsftpd[44],** as an open-source alternative
    * if my client needs file transfer, but is not limited to **ftp** solutions, I would

[43] http://www.wu-ftpd.org/
[44] http://vsftpd.beasts.org/

definitely recommend the utilization of **scp**, or **sftp,** under **ssh2,** possibly in its free "incarnation"[45]

[45] http://www.openssh.com/

3.2 Detect #2 – attacker 10.10.10.122 → victim 192.168.17.135

NOTE: out of lack of space, and due to similarity of mechanism used in processing data, I have moved majority of details into the **Appendix E.**

Reason this attack was selected

Similar to the first detect approach – I first looked at the information from ***strings*** and focused on some other interesting ones, among which: "RETR passwd", "RETR shadow", "PASV", etc. In conjunction with a quick search with ***ngrep*** I have determined that these strings had something in common: the attacker and victim mentioned above (see **Appendix E – Detect #2**). Confirmation of something malicious was obtained via ***snort*** alerts, as captured and depicted in the screenshot of ***ACID***.

Which rule(s) generated the detect?

The specific detect being analyzed triggered the following rules:
http://www.snort.org/snort-db/sid.html?sid=553 POLICY FTP anonymous login attempt
http://www.snort.org/snort-db/sid.html?sid=1992 FTP LIST directory traversal attempt
http://www.snort.org/snort-db/sid.html?sid=356 FTP passwd retrieval attempt
http://www.snort.org/snort-db/sid.html?sid=1928 FTP shadow retrieval attempt
(see details in **Appendix E)**

Probability the address was spoofed

None – the attacker requires TCP connectivity → sessions → "real" address

Description of attack

The attack whose detect was presented before is taking advantage – first - of an implementation flaw in some **ftp** server software, which allows even an anonymous account to "break" out of its home directory, and "move up the tree" structure. Once in a specific point in the directory structure, attempts for retrieval of the files usually containing usernames and passwords is the "normal" next step. None of these are difficult exploits, rather being somehow carefully crafted "normal" ftp sessions. The one thing worth mentioning is that we have proof of the **passwd** file being obtained, but looking inside the parts of it visible to us (

In our case – based on the response received at the time of connection establishment – the FTP server runs the Suse 7.2 distribution of Linux. This one was known to have come pre-packaged with **wu-ftpd** , known to have been prone to security issues, including the ones whose traffic triggered the ***snort*** rules above. rules above.

Attack mechanism

A picture is worth a thousand words: see **Appendix E** for ***ethereal*** screenshot of the entire session. The steps taken were very clear: anonymous login → directory traversal → retrieval of passwd file, attempting retrieval of the shadow one. I just assume that other steps, outside this network trace, may have consisted in the attacker running some brute-force or dictionary-based password cracking tools, against the file(s) downloaded.

Correlations

CVE: http://cve.mitre.org/cgi-bin/cvename.cgi?name=2002-1054
http://www.whitehats.com/info/ids213
GCIA papers: [6] and [8]

Evidence of active targeting

Attempts for retrieval of the password file are obviously directed-toward-victim-system actions. Looking at the ***snort*** alerts, querying for the attacker's address, we can see that the victim was in fact the only system he was "after", in a malicious, known way. To double check this I ran:

```
$ sudo tcpdump -nnnxr 2003.12.15.cap 'host 10.10.10.122 and !host 192.168.17.135'
```

just come to confirm that the attacker (besides doing some DNS queries for the old RedHat network updates) did contact its DNS server (10.10.10.2) in an attempt to do a reverse pointer lookup for the victim (easy to reveal from the traffic, with a simple:

```
$ sudo tcpdump -nnnxr 2003.12.15.cap 'host 10.10.10.122 and !host 192.168.17.135' |
grep –A 4 135.17.168.192
```

which results were shown in **Appendix E)**

Severity

Criticality = same approach as before: what does it "reside" on that server (192.168.17.135), and/or gets exchanged with others, besides the attacker? Answer provided via:

```
$ sudo ngrep -x -n -q -I 2003.12.15.cap '*' "host 192.168.17.135 and not host
10.10.10.122"
```

Sample results in **Appendix E** ➔ conclusion: services running, widely accessible for others = **ftp** and **smtp** ➔ level **4**

Severity = successful **passwd** file transferred ➔ dangerous aspect in itself. Diminishing the effects of this is the fact that looking at the part of the **passwd f**ile having been transferred (remember the only 96 bytes in the capture packets?!?), I can see: root:**x**:0:0:Super – the important thing to notice here is the "**x**", indicative of the fcat that the system is using a root:x:0:0:Super **shadow** file, to store the passwords, which file **has not been successfully obtained** by the attacker, at least per the part of trace I had access to. Conclusion = level **3**

System countermeasures = for the specific attack discussed – really none. So if the system was to contain any important files, a traversal of directory via **ftp,** having proven feasible, would have been damaging. The fact that shadow passwords are used makes me think that they deserve more than "0". Conclusion = **1**

Network measures = see previous detect (networks are the same). Conclusion = **1**

SEVERITY FOR DETECT #2 = (4 + 3) – (1 + 1) = 5

Defensive recommendations

The platform having been identified as SUSE 7.2 ➔ most likely version of ftp software is wu-ftp. If my assumption is correct, all the previous recommendations, from the detect #1, apply here, also. In addition to the above, SUSE specifically decided to switch from wu-ftp, due to a history of security problems, so – as of version 8.0, they have not provided this as the ftp version of choice[46] (switched to vsftpd). If the ftp version running on the victim system is not the one I assumed, then only the general recommendations previously mentioned at the previous detect, may apply.

[46] http://www.novell.com/linux/security/advisories/2003_032_wuftpd.html

3.3 Detect #3 – attacker 10.10.10.113, identified with "bleeding-edge" rules

Reason this detect was selected

I have further examined the trace file from all aspects, and found nothing having potentially resulted in an **intrusion, after an attack** (with the exception of the previously discussed **wu-ftp**, and the questionable status of some SSH sessions, due to the encrypted traffic not allowing proper determination of whether those were client-server "valid-approved" communication, or "malicious-intrusive" in nature). Keeping consistent with this overall paper's purpose ("breaking" a little with the tradition, and presenting something new for my client), I have decided, then, to discuss sample detects, with alerts from what are known to be called **"bleeding-edge" rules[47].** These are rules being created "on the fly", **potentially incomplete, or capable of creating false positives** (I know, I know, some[48] do not think there is such a thing), but extremely useful for "0-day exploits", newly released worms, specific policies, viruses, p2p and other malicious type of traffic, not having made it into the mainstream" ***snort rules.***

The ones I like, having been able to identify/reveal more detailed information about the recon type of traffic generated by, are the **bleeding-edge scan nmap** ones, of which I will discuss an example (see **Appendix E** for a screenshot of ***ACID*** and ***sguil***).

Which rule generated the detect?

alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"BLEEDING-EDGE SCAN NMAP -f -sN"; dsize:0; ack:0; fragbits:!M; flags:0,12; window:2048; reference:arachnids,162; classtype:attempted-recon; sid:2000544; rev:1;)[49]

NOTE: from personal experience and testing on various platforms (MacOSX, Linux, Windows), I have found the original rule somehow limited by the window:2048 option. I would suggest its removal.

The critical issue to be pointed out here, in regards to the specificity (and – as far as I consider it – accuracy) of this type of rule, compared to the "regular" ***snort*** rule triggered by a NULL SCAN, is in the fact that the **bleeding-edge** one has the capability of giving the analyst more visibility into attacker's actions. DANGER: possibility of false positives.

Probability the address was spoofed

Under usual circumstances, an intruder may try to spoof the address, or hide it

[47] http://bleedingsnort.com/
[48] http://taosecurity.blogspot.com/2004_12_01_taosecurity_archive.html
[49] http://www.bleedingsnort.com/staticpages/index.php?page=SigInfo&sid=2000544

among others (e.g. **nmap –D <decoy1 [,decoy2][,ME],…>**[50]), but in our specific case it certainly looks like the attackers have nothing to fear, and are as “open” and as noisy as possible – thus the probability of this address being spoofed being very low.

Description of the detect

The rule mentioned above could be triggered by a command similar to:
$sudo nmap –f –sN <target>

where: -sN = TCP-level scan, with all flags unset

- f = allows the fragment the TCP header in tiny packets (in- obviously – IP fragments), to avoid detection by some ill-equipped (i.e. not having the capability of re-assembling all IP fragments). This could have the potential to elude detection by some IDS, which is why it is good to know that such scans may occur, and provide appropriate solutions.

Attack mechanism

The attacker is attempting a recon via a known tool (**nmap**[51]) for the very simple purpose of information gathering about the end/targeted host (the potential “victim”), and with possible secondary effects of identifying any filtering systems that may be staying “in between” (e.g. firewall). This attack/recon is based on a mechanism of TCP/IP, requiring – in case of fragmentation – the re-assembly ONLY at the receiving end. Unless the usual packet filters / firewalls – in the “path” between potential attackers and protected systems, are capable of queuing, then processing fragments (including re-assembly, of course), such attacks could go unnoticed, and allow the intruders to gather information “stealthily”.

Our attacker seems to have attempted the exploit of the above, in combination with a NULL scan (-sN). This part – in itself – attempts to add to the stealthiness of the probe, by avoiding regular detection or filtering mechanism, set for dropping **tcp()connect** and **syn** packets. In addition to this – the non-valid TCP header created by a NULL scan (i.e. all flags set to 0), tends to trigger various responses, from various implementations of operating systems, allowing better OS fingerprinting, or ports being revealed as open for specific services, at the victim’s end.

Correlation

In regards to the specificity of the particular **nmap** options I mentioned above. I have tested myself the command line above, and traced + **snort**-ed the traffic, and was able to trigger identical alerts. Some papers ([8]) had an analysis about such scans, identified with the regular **snort** rules.

Evidence of active targeting

[50] http://www.insecure.org/nmap/data/nmap_manpage.html
[51] http://www.insecure.org/nmap/

See **Appendix E** for a query into the **ACID** database, having revealed 4487 such specific alerts (**bleeding-edge scan nmap –f –sN)**, targeted by the attacker **I** chose (10.10.10.113) toward just three systems (192.168.17.68, 192.168.17.129 and 192.168.17.135).

Evidence of further exploit attempts (perhaps of a result of scans) can – then – be easily identified, by using similar queries into **ACID** – see **Appendix E** for sample info (e.g. 10.10.10.113 targeting 192.168.17.135 for SNMP).

Severity

Criticality = for the three victims above, based on an analysis similar in concept to the one carried out at detect #1 and #2, i.e.:
192.168.17.135 – providing **ftp** and **smtp** services ➔ level **4**
192.168.17.68 – just SYN/ACK-ing some (known?!?) services to scans ➔ level **2**
192.168.17.129 – same as .68 ➔ level **2**
I believe criticality should not be averaged ➔ Conclusion = **4**

Lethality = such a scan should be categorized low ➔ **1**

System countermeasures = none that we could observe. At least the systems do not appear to "die", as some reports existed about the reaction to thus type of ***nmap*** acans ➔ level **2**

Network countermeasures = none (otherwise the scans should/would have stopped at the router/firewall) ➔ **0**

SEVERITY FOR DETECT #3 = (4 + 1) – (2 + 0) = 3

Defensive recommendations

As previously hinted: provide intermediary, protecting devices (e.g. firewalls) capable of handling this type of scans (with packet re-assembly);

Operating systems should be checked (and patched, if need be) against potential damaging effects of such scans (e.g. tiny fragments, etc.)

If justified – use host-based intrusion detection solutions[52], specifically designed to address such attacks.

[52] http://sourceforge.net/projects/sentrytools/

4. Analysis process

This section would probably be redundant, if I were to focus specifically on the process, as I have described in as much detail as allowed by space constraints, all the steps I have gone through, in the preparatory phase. I also believe that the initial analysis part is a much better place for process description, as tasks, theory and results “blend” together, and lead to the specifics of other sections.

I have chosen – instead – to use this section to summarize all the **open source or free** tools I have “formally” used to prepare this analysis, and add to those other tools I have tried, but whose results could not justify taking the space of mentioning them explicitly in the body of this paper. I will also include tools I have personally used in other jobs, which are or could be helpful in intrusion analysis. For ease of use, I have consolidated all this information in tabular format, in **Appendix F – tools.**

REFERENCES

1. Sternudd, Patrik. *"GCIA Practical Assignment version 3.4"* http://www.giac.org/practical/GCIA/Patrik_Sternudd_GCIA.pdf

2. IEEE. *"IEEE OUI and Company_id Assignments"* – searchable database http://standards.ieee.org/regauth/oui/index.shtml

3. Neugebaur, Frank. *"Intrusion Detection "Knowing when someone is knocking on your door"* http://www.linux-tip.net/cms/images/stories/documents/snort_acid_mdk92.pdf

4. Bejtlich, Richard. *"Why Sguil Is the Best Option for Network Security Monitoring Data"* http://www.informit.com/articles/article.asp?p=350390

5. Weber, Dana *"GIAC Certified Intrusion Analyst (GCIA) Practical Assignment Version 3.4"* http://www.giac.org/practical/GCIA/Dana_Webber_GCIA.pdf

6. Eaton, Ian *"GCIA Certified Intrusion Analyst (GCIA) Practical Assignment Paper version 1.0"* http://www.giac.org/practical/GCIA/Ian_Eaton_GCIA.pdf

7. Terell, Steve *"GIAC Certified Incident Handler – Practical Assignment version 3"* http://www.giac.org/practical/GCIH/Steve_Terrell_GCIH.pdf

8. Lalla, Gregory *"GCIA Intrusion Detection Practical v3.4"* http://www.giac.org/practical/GCIA/Gregory_Lalla_GCIA.pdf

BIBLIOGRAPHY

1. Northcutt, S. and Novak J. – *“Network Intrusion Detection” – third edition* New Riders Publishing 2003

2. Northcutt, S., Cooper, M., Fearnow M., and Frederick K. – *“Intrusion Signatures and Analysis”* New Riders Publishing 2001

3. Orebaugh, A. et al – *“Ethereal Packet Sniffing”* Syngress Publishing Inc. 2004

4. Beale J. & The Snort Development Team, Baker A., Caswell B., Poor M. – *“Snort 2.1 Intrusion Detection” – second edition* Syngress Publishing Inc. 2004

5. The Honeynet Project – *“Know your enemy – Learning about security threats” – second edition* Addison-Wesley 2004

6. Schiffman, Mike – *“Hacker’s Challenge”* Osborne/McGrawHill2001

7. Schiffman, Mike et al. – *“Hacker’s Challenge 2”* Osborne/McGrawHill 2003

8. Stevens, Richard W. – *“TCP/IP Illustrated, Volume 1 – The Protocols”* Addison-Wesley 1994

APPENDICES

Appendix A – various data from the first section

File **mac-and-ip-on-10-10-10-0-network.txt**

Nb of occurrences	MAC	IP
100547	00:03:47:8c:89:c2	10.10.10.165
58619	00:04:76:45:61:39	10.10.10.195
28247	00:01:02:79:91:ed	10.10.10.112
21709	00:06:5b:e6:f8:43	10.10.10.231
18256	00:0a:95:7c:24:00	10.10.10.113
17275	00:02:a5:b6:e2:e3	10.10.10.186
16936	00:0c:29:9e:ef:53	10.10.10.224
6015	00:08:74:05:b7:f8	10.10.10.147
5876	00:03:ff:df:95:84	10.10.10.228
5775	00:e0:98:a1:7f:da	10.10.10.174
5356	00:d0:59:c6:5e:14	10.10.10.141
4827	00:0b:db:9b:46:fe	10.10.10.164
4808	00:01:03:88:29:92	10.10.10.234
3902	00:a0:c9:ba:6d:85	10.10.10.196
2980	00:0c:29:39:6e:67	10.10.10.160
1550	00:0c:29:14:1e:63	10.10.10.142
1361	00:09:6b:02:e9:3d	10.10.10.212
1341	00:0a:95:d9:95:84	10.10.10.232
892	00:50:56:40:00:6d	10.10.10.1
802	00:e0:b8:3d:20:40	10.10.10.214
407	00:50:56:40:00:64	10.10.10.2
298	00:00:e2:94:b0:2a	10.10.10.226
194	00:06:5b:d8:bf:ed	10.10.10.122
181	00:0b:db:17:f4:c9	10.10.10.194
48	00:00:e2:92:ee:0f	10.10.10.222
31	08:00:46:79:f7:7c	10.10.10.230
16	00:0b:db:df:53:8d	10.10.10.123
10	00:08:74:07:31:ee	10.10.10.111
1	00:d0:59:c6:5e:14	10.10.10.144
1	00:00:39:f2:67:88	10.10.10.117

File **mac-and-ip-not-destined-to-10-10-10-0-network.txt**

Nb of occurrences	MAC	IP
59406	00:50:56:40:00:6d	172.20.201.198
49458	00:50:56:40:00:6d	172.20.11.80

```
20822	00:50:56:40:00:6d	172.20.201.2
20426	00:50:56:40:00:6d	172.20.201.1
20086	00:50:56:40:00:6d	192.168.17.68
19547	00:50:56:40:00:6d	172.20.201.135
14206	00:50:56:40:00:6d	172.20.11.2
13092	00:50:56:40:00:6d	192.168.17.129
12138	00:50:56:40:00:6d	192.168.17.66
10434	00:50:56:40:00:6d	172.20.11.52
9844	00:50:56:40:00:6d	192.168.17.135
8223	00:50:56:40:00:6d	172.20.11.3
5613	00:50:56:40:00:6d	192.168.22.207
5325	00:50:56:40:00:6d	192.168.17.67
5308	00:50:56:40:00:6d	192.168.17.1
4790	00:50:56:40:00:6d	172.20.201.3
4657	00:50:56:40:00:6d	192.168.17.65
4205	00:50:56:40:00:6d	172.20.11.1
2174	00:50:56:40:00:6d	172.22.201.1
1860	00:50:56:40:00:6d	172.20.201.0
1377	00:50:56:40:00:6d	172.22.201.2
1361	00:50:56:40:00:6d	172.20.11.255
1346	00:50:56:40:00:6d	172.20.11.0
1035	00:50:56:40:00:6d	172.11.11.80
726	00:50:56:40:00:6d	172.22.201.3
452	00:50:56:40:00:6d	192.168.17.9
318	00:50:56:40:00:6d	172.10.11.80
247	00:50:56:40:00:6d	149.134.30.62
201	00:50:56:40:00:6d	149.134.52.149
72	00:50:56:40:00:6d	172.20.11.254
72	00:50:56:40:00:6d	172.20.11.253
72	00:50:56:40:00:6d	172.20.11.252
72	00:50:56:40:00:6d	172.20.11.251
72	00:50:56:40:00:6d	172.20.11.250
72	00:50:56:40:00:6d	172.20.11.249
72	00:50:56:40:00:6d	172.20.11.248
72	00:50:56:40:00:6d	172.20.11.247
72	00:50:56:40:00:6d	172.20.11.246
72	00:50:56:40:00:6d	172.20.11.245
72	00:50:56:40:00:6d	172.20.11.244
72	00:50:56:40:00:6d	172.20.11.243
72	00:50:56:40:00:6d	172.20.11.242
72	00:50:56:40:00:6d	172.20.11.241
72	00:50:56:40:00:6d	172.20.11.240
72	00:50:56:40:00:6d	172.20.11.239
67	00:50:56:40:00:6d	172.20.11.238
57	ff:ff:ff:ff:ff:ff	255.255.255.255
31	00:50:56:40:00:6d	172.20.201.198
```

```
24      01:00:5e:37:96:d0      229.55.150.208
21      00:50:56:40:00:6d      192.168.17.68
18      ff:ff:ff:ff:ff:ff      169.254.255.255
17      00:50:56:40:00:6d      192.168.22.207
17      00:50:56:40:00:6d      192.168.17.67
17      00:50:56:40:00:6d      192.168.17.66
17      00:50:56:40:00:6d      192.168.17.1
16      01:00:5e:00:00:16      224.0.0.22
15      00:50:56:40:00:6d      198.41.0.5
14      00:50:56:40:00:6d      172.20.11.2
13      00:50:56:40:00:6d      172.20.201.135
12      00:50:56:40:00:6d      172.20.201.1
12      00:50:56:40:00:6d      10.3.200.84
10      00:50:56:40:00:6d      172.20.201.99
10      00:50:56:40:00:6d      172.20.201.98
... – not part of original file, but all the way down to the following lines …
… skipped for brevity …
10      00:50:56:40:00:6d      172.20.201.10
9      00:50:56:40:00:6d      192.168.17.62
8      00:50:56:40:00:6d      172.20.201.87
8      00:50:56:40:00:6d      172.20.201.26
8      00:50:56:40:00:6d      172.20.201.231
8      00:50:56:40:00:6d      172.20.201.122
7      00:50:56:40:00:6d      192.168.17.135
7      00:50:56:40:00:6d      12.162.170.196
6      00:50:56:40:00:6d      192.168.22.133
6      00:50:56:40:00:6d      192.168.17.64
6      00:50:56:40:00:6d      192.168.17.63
6      00:50:56:40:00:6d      192.168.17.61
6      00:50:56:40:00:6d      192.168.17.60
6      00:50:56:40:00:6d      192.168.17.59
6      00:50:56:40:00:6d      192.168.17.58
6      00:50:56:40:00:6d      192.168.17.57
6      00:50:56:40:00:6d      192.168.17.56
6      00:50:56:40:00:6d      192.168.17.55
6      00:50:56:40:00:6d      192.168.17.54
6      00:50:56:40:00:6d      192.168.17.53
6      00:50:56:40:00:6d      192.168.17.52
6      00:50:56:40:00:6d      192.168.17.51
6      00:50:56:40:00:6d      172.20.11.1
6      00:50:56:40:00:6d      127.0.0.1
5      00:50:56:40:00:6d      198.123.30.132
5      00:50:56:40:00:6d      192.168.22.99
5      00:50:56:40:00:6d      192.168.22.98
… skipped for brevity …
5      00:50:56:40:00:6d      192.168.22.100
```

```
5     00:50:56:40:00:6d     192.168.22.10
5     00:50:56:40:00:6d     192.168.22.1
5     00:50:56:40:00:6d     192.168.17.99
5     00:50:56:40:00:6d     192.168.17.98
5     00:50:56:40:00:6d     192.168.17.97
5     00:50:56:40:00:6d     192.168.17.96
… skipped for brevity …
5     00:50:56:40:00:6d     192.168.17.101
5     00:50:56:40:00:6d     192.168.17.100
5     00:50:56:40:00:6d     192.168.17.10
4     00:50:56:40:00:6d     172.20.201.2
4     00:50:56:40:00:6d     172.20.11.3
3     01:00:5e:7f:ff:fa    239.255.255.250
3     01:00:5e:7a:0a:8c     238.122.10.140
3     01:00:5e:00:00:06     224.0.0.6
3     01:00:5e:00:00:05     224.0.0.5
3     01:00:5e:00:00:02     224.0.0.2
3     00:50:56:40:00:6d     102.168.17.62
2     00:50:56:40:00:6d     192.168.17.2
2     00:50:56:40:00:6d     172.27.1.8
2     00:50:56:40:00:6d     172.22.201.99
2     00:50:56:40:00:6d     172.22.201.98
… skipped for brevity …
2     00:50:56:40:00:6d     172.22.201.101
2     00:50:56:40:00:6d     172.22.201.100
2     00:50:56:40:00:6d     172.22.201.10
2     00:50:56:40:00:6d     172.22.201.1
2     00:50:56:40:00:6d     172.20.11.80
2     00:50:56:40:00:6d     172.20.11.52
2     00:50:56:40:00:6d     172.20.11.201
2     00:50:56:40:00:6d     134.248.127.21
1     01:00:5e:7f:ff:fd    239.255.255.253
1     00:50:56:40:00:6d     192.168.17.76
1     00:50:56:40:00:6d     192.168.17.75
1     00:50:56:40:00:6d     192.168.17.74
1     00:50:56:40:00:6d     192.168.17.73
1     00:50:56:40:00:6d     192.168.17.72
1     00:50:56:40:00:6d     192.168.17.71
1     00:50:56:40:00:6d     192.168.17.70
1     00:50:56:40:00:6d     172.20.12.99
1     00:50:56:40:00:6d     172.20.12.98
1     00:50:56:40:00:6d     172.20.12.97
… skipped for brevity …
1     00:50:56:40:00:6d     172.20.12.101
1     00:50:56:40:00:6d     172.20.12.100
1     00:50:56:40:00:6d     172.20.12.10
```

```
1       00:50:56:40:00:6d       172.20.12.1
1       00:50:56:40:00:6d       172.20.11.99
1       00:50:56:40:00:6d       172.20.11.98
… skipped for brevity …
1       00:50:56:40:00:6d       172.20.11.101
1       00:50:56:40:00:6d       172.20.11.100
1       00:50:56:40:00:6d       172.20.11.10
1       00:50:56:40:00:6d       172.20.102.198
1       00:50:56:40:00:6d       172.11.11.80
1       00:50:56:40:00:6d       172.10.11.80
```

File **source-mac.txt** (manually removed MAC duplicates, to better reveal the multi-IPs associated with one MAC)

```
00:00:39:f2:67:88   10.10.10.117
00:00:e2:92:ee:0f   10.10.10.222
00:00:e2:94:b0:2a   10.10.10.226
00:01:02:79:91:ed   10.10.10.112
00:01:03:88:29:92   10.10.10.234
00:02:a5:b6:e2:e3   10.10.10.186
00:03:47:8c:89:c2   10.10.10.165
                    192.168.117.1
                    192.168.213.1
00:03:ff:df:95:84   10.10.10.228
00:04:76:45:61:39   10.10.10.195
00:06:5b:d8:bf:ed   10.10.10.122
00:06:5b:e6:f8:43   10.10.10.231
00:08:74:05:b7:f8   10.10.10.147
00:08:74:07:31:ee   0.0.0.0
                    10.10.10.111
                    172.16.8.189
00:09:6b:02:e9:3d   10.10.10.212
                    172.16.8.229
00:0a:95:7c:24:00   10.10.10.113
00:0a:95:d9:95:84   10.10.10.232
00:0b:db:17:f4:c9   0.0.0.0
                    10.10.10.194
00:0b:db:17:f4:c9   169.254.135.50
                    172.16.9.13
                    192.168.222.1
                    192.168.84.1
00:0b:db:9b:46:fe   10.10.10.164
00:0b:db:df:53:8d   10.10.10.123
00:0c:29:14:1e:63   0.0.0.0
                    10.10.10.142
```

```
00:0c:29:39:6e:67   0.0.0.0
                    10.10.10.160
00:0c:29:9e:ef:53   10.10.10.224
00:50:56:40:00:64   10.10.10.2
00:50:56:40:00:6d   10.10.10.1
                    10.30.30.2
                    172.20.11.1
                    172.20.11.2
                    172.20.11.3
                    172.20.11.52
                    172.20.11.80
                    172.20.201.1
                    172.20.201.135
                    172.20.201.198
                    172.20.201.2
                    192.168.17.129
                    192.168.17.135
                    192.168.17.2
                    192.168.17.65
                    192.168.17.66
                    192.168.17.68
                    192.168.22.207
00:a0:c9:ba:6d:85   10.10.10.196
00:d0:59:c6:5e:14   0.0.0.0
                    10.10.10.141
                    10.10.10.144
                    238.122.10.140
00:e0:98:a1:7f:da   10.10.10.174
00:e0:b8:3d:20:40   10.10.10.214
08:00:46:79:f7:7c   0.0.0.0
                    10.10.10.230
```

File **dest-mac.txt**

```
00:00:39:f2:67:88     10.10.10.117
00:00:e2:92:ee:0f     10.10.10.222
00:00:e2:94:b0:2a      10.10.10.226
00:01:02:79:91:ed      10.10.10.112
00:01:03:88:29:92      10.10.10.234
00:02:a5:b6:e2:e3      10.10.10.186
00:03:47:8c:89:c2      10.10.10.165
00:03:ff:df:95:84     10.10.10.228
00:04:76:45:61:39      10.10.10.195
00:06:5b:d8:bf:ed     10.10.10.122
00:06:5b:e6:f8:43     10.10.10.231
00:08:74:05:b7:f8     10.10.10.147
```

```
00:08:74:07:31:ee      10.10.10.111
00:09:6b:02:e9:3d      10.10.10.212
00:0a:95:7c:24:00      10.10.10.113
00:0a:95:d9:95:84      10.10.10.232
00:0b:db:17:f4:c9      10.10.10.194
00:0b:db:9b:46:fe      10.10.10.164
00:0b:db:df:53:8d      10.10.10.123
00:0c:29:14:1e:63      10.10.10.142
00:0c:29:39:6e:67      10.10.10.160
00:0c:29:9e:ef:53      10.10.10.224
00:50:56:40:00:64      10.10.10.2
00:50:56:40:00:6d      10.3.200.84
00:50:56:40:00:6d      102.168.17.62
00:50:56:40:00:6d      12.162.170.196
00:50:56:40:00:6d      127.0.0.1
00:50:56:40:00:6d      134.248.127.21
00:50:56:40:00:6d      149.134.30.62
00:50:56:40:00:6d      149.134.52.149
00:50:56:40:00:6d      172.10.11.80
00:50:56:40:00:6d      172.11.11.80
00:50:56:40:00:6d      172.20.102.198
00:50:56:40:00:6d      172.20.11.0
00:50:56:40:00:6d      172.20.11.1
… removed for brevity …
00:50:56:40:00:6d      172.20.11.99
00:50:56:40:00:6d      172.20.12.1
… removed for brevity …
00:50:56:40:00:6d      172.20.12.99
00:50:56:40:00:6d      172.20.201.0
00:50:56:40:00:6d      172.20.201.1
… removed for brevity …
00:50:56:40:00:6d      172.20.201.99
00:50:56:40:00:6d      172.22.201.1
… removed for brevity …
00:50:56:40:00:6d      172.22.201.99
00:50:56:40:00:6d      172.27.1.8
00:50:56:40:00:6d      192.168.17.1
… removed for brevity …
00:50:56:40:00:6d      192.168.17.99
00:50:56:40:00:6d      192.168.22.1
… removed for brevity …
00:50:56:40:00:6d      192.168.22.99
00:50:56:40:00:6d      198.123.30.132
00:50:56:40:00:6d      198.41.0.5
00:a0:c9:ba:6d:85      10.10.10.196
00:d0:59:c6:5e:14      10.10.10.141
```

```
00:e0:98:a1:7f:da      10.10.10.174
00:e0:b8:3d:20:40      10.10.10.214
01:00:5e:00:00:02      224.0.0.2
01:00:5e:00:00:05      224.0.0.5
01:00:5e:00:00:06      224.0.0.6
01:00:5e:00:00:16      224.0.0.22
01:00:5e:37:96:d0      229.55.150.208
01:00:5e:7a:0a:8c      238.122.10.140
01:00:5e:7f:ff:fa      239.255.255.250
01:00:5e:7f:ff:fd      239.255.255.253
08:00:46:79:f7:7c      10.10.10.230
ff:ff:ff:ff:ff:ff      10.10.10.255
ff:ff:ff:ff:ff:ff      169.254.255.255
ff:ff:ff:ff:ff:ff      255.255.255.255
```

File **conv.txt**

```
6625 10.10.10.224-172.20.201.2
6461 10.10.10.112-192.168.17.68
6453 10.10.10.112-192.168.17.66
6453 10.10.10.112-192.168.17.129
4790 10.10.10.224-172.20.201.3
4645 10.10.10.112-192.168.17.65
4132 10.10.10.112-192.168.17.135
3882 10.10.10.113-192.168.17.68
3518 10.10.10.113-192.168.17.129
3393 10.10.10.113-192.168.17.135
2593 10.10.10.196-172.20.11.3
2574 10.10.10.195-172.20.11.2
2519 10.10.10.174-172.20.11.3
2347 10.10.10.165-192.168.17.66
2311 10.10.10.165-192.168.17.68
2238 10.10.10.165-192.168.17.67
2237 10.10.10.165-192.168.22.207
2231 10.10.10.165-192.168.17.1
2146 10.10.10.164-172.22.201.1
1845 10.10.10.224-172.20.201.0
1745 10.10.10.165-172.20.201.198
1695 10.10.10.195-172.20.11.52
1630 10.10.10.165-172.20.201.135
1614 10.10.10.224-172.20.201.1
1602 10.10.10.165-172.20.201.1
1569 10.10.10.228-172.20.201.135
1569 10.10.10.228-172.20.201.1
15525 10.10.10.186-172.20.201.198
```

```
14863 10.10.10.195-172.20.11.80
1361 10.10.10.231-172.20.11.80
1361 10.10.10.231-172.20.11.52
1361 10.10.10.231-172.20.11.3
1361 10.10.10.231-172.20.11.255
1361 10.10.10.231-172.20.11.2
1361 10.10.10.231-172.20.11.1
1361 10.10.10.164-172.22.201.2
1346 10.10.10.141-172.20.11.0
1329 10.10.10.165-172.20.201.2
1315 10.10.10.141-172.20.11.2
1184 10.10.10.174-172.20.11.52
1163 10.10.10.174-172.20.11.80
 714 10.10.10.164-172.22.201.3
 532 10.10.10.141-192.168.17.68
 452 10.10.10.231-192.168.17.9
 348 10.10.10.195-172.11.11.80
 106 10.10.10.195-172.10.11.80
  72 10.10.10.212-172.20.11.254
  72 10.10.10.212-172.20.11.253
  72 10.10.10.212-172.20.11.252
  72 10.10.10.212-172.20.11.251
  72 10.10.10.212-172.20.11.250
  72 10.10.10.212-172.20.11.249
  72 10.10.10.212-172.20.11.248
  72 10.10.10.212-172.20.11.247
  72 10.10.10.212-172.20.11.246
  72 10.10.10.212-172.20.11.245
  72 10.10.10.212-172.20.11.244
  72 10.10.10.212-172.20.11.243
  72 10.10.10.212-172.20.11.242
  72 10.10.10.212-172.20.11.241
  72 10.10.10.212-172.20.11.240
  72 10.10.10.212-172.20.11.239
  67 10.10.10.212-172.20.11.238
  41 172.20.201.198-10.10.10.186
  23 172.20.11.80-10.10.10.195
  16 172.20.201.198-10.10.10.165
  16 10.10.10.122-192.168.17.135
  12 172.20.201.135-10.10.10.228
  10 192.168.17.135-10.10.10.212
  10 10.10.10.196-172.20.201.198
   8 172.20.201.1-10.10.10.228
   8 172.20.11.2-10.10.10.195
   7 10.10.10.234-192.168.17.68
   7 10.10.10.234-172.20.201.198
```

```
7 10.10.10.226-192.168.17.135
7 10.10.10.147-172.20.201.198
6 172.20.201.135-10.10.10.165
6 172.20.11.80-10.10.10.174
6 172.16.9.13-192.168.17.68
6 10.10.10.226-192.168.17.64
6 10.10.10.226-192.168.17.63
6 10.10.10.226-192.168.17.62
6 10.10.10.226-192.168.17.61
6 10.10.10.226-192.168.17.60
6 10.10.10.226-192.168.17.59
6 10.10.10.226-192.168.17.58
6 10.10.10.226-192.168.17.57
6 10.10.10.226-192.168.17.56
6 10.10.10.226-192.168.17.55
6 10.10.10.226-192.168.17.54
6 10.10.10.226-192.168.17.53
6 10.10.10.226-192.168.17.52
6 10.10.10.226-192.168.17.51
6 10.10.10.195-10.10.10.2
6 10.10.10.194-192.168.17.68
5 172.20.11.52-10.10.10.195
5 10.10.10.226-192.168.17.50
5 10.10.10.186-172.20.11.1
5 10.10.10.165-198.41.0.5
4 192.168.84.1-192.168.17.68
4 192.168.222.1-192.168.17.68
4 172.20.11.52-10.10.10.174
4 10.10.10.232-172.20.201.198
4 10.10.10.228-172.20.201.198
3 192.168.17.135-10.10.10.142
3 192.168.17.135-10.10.10.122
3 10.10.10.224-172.20.201.198
3 10.10.10.195-172.20.11.1
3 10.10.10.160-172.20.201.198
3 10.10.10.142-192.168.17.135
2 192.168.22.207-10.10.10.224
2 192.168.17.66-10.10.10.112
2 192.168.17.135-10.10.10.112
2 192.168.17.129-10.10.10.112
2 172.20.201.198-10.10.10.228
2 172.20.201.198-10.10.10.224
2 172.20.201.1-10.10.10.165
2 172.20.11.2-10.10.10.141
2 172.16.8.229-12.162.170.196
2 10.10.10.212-192.168.22.133
```

```
2 10.10.10.186-172.20.11.2
2 10.10.10.165-172.20.11.80
2 10.10.10.142-172.20.201.198
2 10.10.10.142-172.20.11.3
2 10.10.10.112-192.168.17.2
1 238.122.10.140-172.20.11.2
1 192.168.213.1-192.168.22.207
1 192.168.213.1-192.168.17.68
1 192.168.213.1-192.168.17.67
1 192.168.213.1-192.168.17.66
1 192.168.213.1-192.168.17.1
1 192.168.213.1-172.20.201.2
1 192.168.213.1-172.20.201.198
1 192.168.213.1-172.20.201.135
1 192.168.213.1-172.20.201.1
1 192.168.17.68-10.10.10.112
1 192.168.17.66-10.10.10.165
1 192.168.17.65-10.10.10.112
1 192.168.117.1-192.168.22.207
1 192.168.117.1-192.168.17.68
1 192.168.117.1-192.168.17.67
1 192.168.117.1-192.168.17.66
1 192.168.117.1-192.168.17.1
1 192.168.117.1-172.20.201.2
1 192.168.117.1-172.20.201.198
1 192.168.117.1-172.20.201.135
1 192.168.117.1-172.20.201.1
1 172.20.201.198-10.10.10.196
1 172.20.201.1-10.10.10.224
1 172.20.11.3-10.10.10.196
1 172.20.11.3-10.10.10.174
1 172.20.11.3-10.10.10.142
1 10.10.10.228-172.20.102.198
1 10.10.10.226-192.168.17.76
1 10.10.10.226-192.168.17.75
1 10.10.10.226-192.168.17.74
1 10.10.10.226-192.168.17.73
1 10.10.10.226-192.168.17.72
1 10.10.10.226-192.168.17.71
1 10.10.10.226-192.168.17.70
1 10.10.10.226-192.168.17.69
1 10.10.10.226-192.168.17.68
1 10.10.10.226-192.168.17.67
1 10.10.10.226-192.168.17.66
1 10.10.10.226-192.168.17.65
1 10.10.10.224-172.20.11.3
```

```
1 10.10.10.222-192.168.17.68
1 10.10.10.214-192.168.22.207
1 10.10.10.214-172.20.201.198
1 10.10.10.212-192.168.17.69
1 10.10.10.212-192.168.17.62
1 10.10.10.212-192.168.17.135
1 10.10.10.212-12.162.170.196
1 10.10.10.212-102.168.17.62
1 10.10.10.196-127.0.0.1
1 10.10.10.195-172.20.11.3
1 10.10.10.195-134.248.127.21
1 10.10.10.174-192.168.22.207
1 10.10.10.174-172.20.201.198
1 10.10.10.174-172.20.11.201
1 10.10.10.160-127.0.0.1
1 10.10.10.123-192.168.17.69
```

File **conv_ipsumdump.txt** (samples of “I” = ICMP, “T” = TCP and “U” = UDP recorded traffic. While having worked through it in various ways, I did not think it would have made sense to attach it to this paper in its 4200(!) pages entirety)

```
!creator "ipsumdump -psSdD -r 2003.12.15.cap"
!data ip_proto ip_src sport ip_dst dport
!host Stef.local
!IPSummaryDump 1.2

… samples produced by ipsumdump …
… ICMP …

I 10.10.10.1 - 10.10.10.141 -
I 10.10.10.1 - 10.10.10.147 -
I 10.10.10.1 - 10.10.10.164 -

… samples produced by ipsumdump …
… TCP …

T 10.10.10.112 32770 192.168.17.65 937
T 10.10.10.112 32771 192.168.17.65 6110
T 10.10.10.112 32772 192.168.17.65 965

… samples produced by ipsumdump …
… UDP …

U 192.168.117.1 137 172.20.201.1 137
U 192.168.117.1 137 172.20.201.135 137
```

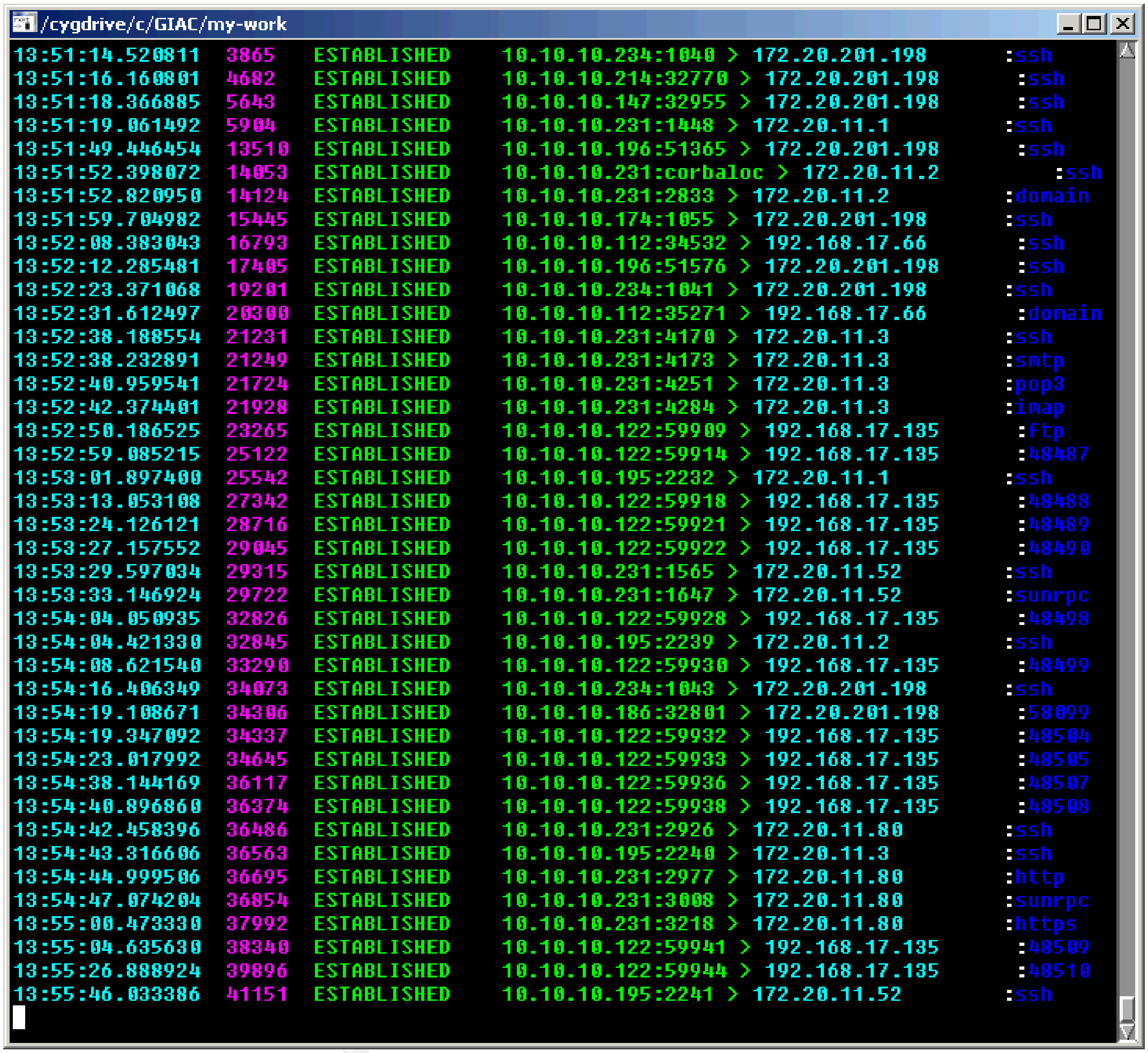
```
U 192.168.117.1 137 172.20.201.198 137

… etc …

T 238.122.10.140 42200 172.20.11.2 31097

… end of file.
```

File **stats.txt**

```
DumpFile:  2003.12.15.cap
FileSize: 37.25MB
Id: 200311181257
StartTime: Tue Nov 18 12:57:23 2003
EndTime:   Tue Nov 18 14:15:57 2003
TotalTime: 4714.02 seconds
TotalCapSize: 30.02MB  CapLen: 96 bytes
# of packets: 474024 (36.01MB)
AvgRate: 81.20Kbps  stddev:157.98K   PeakRate: 3.94Mbps

### IP flow (unique src/dst pair) Information ###
# of flows: 1827  (avg. 259.45 pkts/flow)
Top 10 big flow size (bytes/total in %):
  7.3%  7.0%  4.7%  4.6%  3.7%  3.3%  3.3%  3.2%  3.0%  2.5%

### IP address Information ###
# of IPv4 addresses: 1593
Top 10 bandwidth usage (bytes/total in %):
 35.9% 22.4% 19.7% 16.8%  7.5%  6.9%  6.8%  5.8%  5.3%  5.2%
### Packet Size Distribution (including MAC headers) ###
<<<<
 [   32-   63]:     340520
 [   64-  127]:     123747
 [  128-  255]:       3331
 [  256-  511]:       1577
 [  512- 1023]:       1548
 [ 1024- 2047]:       3301
>>>>


### Protocol Breakdown ###
<<<<
     protocol            packets                      bytes           bytes/pkt
```

```
--------------------------------------------------------------------
[0] total          474024 (100.00%)        37764203 (100.00%)     79.67
[1] ip            449144 ( 94.75%)        36052637 ( 95.47%)     80.27
[2]  tcp           372578 ( 78.60%)        30546920 ( 80.89%)     81.99
[3]   ftpdata          423 (  0.09%)           29007 (  0.08%)     68.57
[3]   ftp            4684 (  0.99%)          968594 (  2.56%)    206.79
[3]   ssh            43603 (  9.20%)         9015748 ( 23.87%)    206.77
[3]   telnet          1078 (  0.23%)           73898 (  0.20%)     68.55
[3]   smtp             916 (  0.19%)           64317 (  0.17%)     70.22
[3]   name              221 (  0.05%)           13916 (  0.04%)     62.97
[3]   dns             500 (  0.11%)           40037 (  0.11%)     80.07
[3]   http(s)         1896 (  0.40%)          117569 (  0.31%)     62.01
[3]   http(c)         2378 (  0.50%)          151413 (  0.40%)     63.67
[3]   kerb5            243 (  0.05%)           15464 (  0.04%)     63.64
[3]   pop3             337 (  0.07%)           21291 (  0.06%)     63.18
[3]   sunrpc           394 (  0.08%)           25952 (  0.07%)     65.87
[3]   ident           484 (  0.10%)           38058 (  0.10%)     78.63
[3]   nntp            310 (  0.07%)           19632 (  0.05%)     63.33
[3]   ntp            234 (  0.05%)          14946 (  0.04%)     63.87
[3]   epmap            266 (  0.06%)           16680 (  0.04%)     62.71
[3]   netb-ns          220 (  0.05%)           13894 (  0.04%)     63.15
[3]   netb-se          394 (  0.08%)           24654 (  0.07%)     62.57
[3]   imap             324 (  0.07%)           20485 (  0.05%)     63.23
[3]   bgp             252 (  0.05%)           16118 (  0.04%)     63.96
[3]   ldap            219 (  0.05%)           13670 (  0.04%)     62.42
[3]   https           1295 (  0.27%)           80078 (  0.21%)     61.84
[3]   ms-ds            225 (  0.05%)           14064 (  0.04%)     62.51
[3]   rlogin          351 (  0.07%)           22470 (  0.06%)     64.02
[3]   rtsp           171 (  0.04%)          10732 (  0.03%)     62.76
[3]   ldaps           209 (  0.04%)           13458 (  0.04%)     64.39
[3]   socks            636 (  0.13%)           39330 (  0.10%)     61.84
[3]   kasaa            92 (  0.02%)            5646 (  0.01%)     61.37
[3]   mssql-s          206 (  0.04%)           12922 (  0.03%)     62.73
[3]   squid           229 (  0.05%)           14268 (  0.04%)     62.31
[3]   ms-gc            135 (  0.03%)            8486 (  0.02%)     62.86
[3]   ms-gcs           146 (  0.03%)            9214 (  0.02%)     63.11
[3]   hotline            2 (  0.00%)            134 (  0.00%)     67.00
[3]   realaud           80 (  0.02%)            5144 (  0.01%)     64.30
[3]   icecast          155 (  0.03%)            9576 (  0.03%)     61.78
[3]   gnu6346          103 (  0.02%)            6542 (  0.02%)     63.51
[3]   gnu6347            2 (  0.00%)             134 (  0.00%)     67.00
[3]   gnu6348            2 (  0.00%)             134 (  0.00%)     67.00
[3]   gnu6349            2 (  0.00%)             134 (  0.00%)     67.00
[3]   gnu6350            2 (  0.00%)             134 (  0.00%)     67.00
[3]   gnu6355            2 (  0.00%)             134 (  0.00%)     67.00
[3]   irc6666          119 (  0.03%)            7584 (  0.02%)     63.73
```

```
[3]   irc6667          136 (  0.03%)            8582 (  0.02%)     63.10
[3]   irc6668           84 (  0.02%)            5412 (  0.01%)     64.43
[3]   irc6669            2 (  0.00%)             134 (  0.00%)     67.00
[3]   napster           89 (  0.02%)            5768 (  0.02%)     64.81
[3]   irc7000          154 (  0.03%)            9664 (  0.03%)     62.75
[3]   http-a          3817 (  0.81%)          233558 (  0.62%)     61.19
[3]   other         304756 ( 64.29%)        19308141 ( 51.13%)     63.36
[2]  udp             66543 ( 14.04%)         4750378 ( 12.58%)     71.39
[3]   name              35 (  0.01%)            2163 (  0.01%)     61.80
[3]   dns              759 (  0.16%)           83011 (  0.22%)    109.37
[3]   kerb5             44 (  0.01%)            3790 (  0.01%)     86.14
[3]   sunrpc           123 (  0.03%)            9986 (  0.03%)     81.19
[3]   ntp               63 (  0.01%)            4570 (  0.01%)     72.54
[3]   epmap             40 (  0.01%)            2508 (  0.01%)     62.70
[3]   netb-ns         1003 (  0.21%)           99595 (  0.26%)     99.30
[3]   netb-se          110 (  0.02%)           19827 (  0.05%)    180.25
[3]   ms-ds             40 (  0.01%)            2520 (  0.01%)     63.00
[3]   rip               83 (  0.02%)            5366 (  0.01%)     64.65
[3]   mcast             27 (  0.01%)            5901 (  0.02%)    218.56
[3]   halflif            9 (  0.00%)             612 (  0.00%)     68.00
[3]   unreal            12 (  0.00%)             792 (  0.00%)     66.00
[3]   quake              6 (  0.00%)             432 (  0.00%)     72.00
[3]   other          64189 ( 13.54%)         4509305 ( 11.94%)     70.25
[2]  icmp             9992 (  2.11%)          753479 (  2.00%)     75.41
[2]  igmp               29 (  0.01%)            1740 (  0.00%)     60.00
[2]  egp                 1 (  0.00%)              60 (  0.00%)     60.00
[2]  any-loca            1 (  0.00%)              60 (  0.00%)     60.00
>>>>
```



```
13:51:14.520811  3865   ESTABLISHED    10.10.10.234:1040 > 172.20.201.198       :ssh
13:51:16.160801  4682   ESTABLISHED    10.10.10.214:32770 > 172.20.201.198       :ssh
13:51:18.366885  5643   ESTABLISHED    10.10.10.147:32955 > 172.20.201.198       :ssh
13:51:19.061492  5904   ESTABLISHED    10.10.10.231:1448 > 172.20.11.1          :ssh
13:51:49.446454  13510  ESTABLISHED    10.10.10.196:51365 > 172.20.201.198       :ssh
13:51:52.398072  14053  ESTABLISHED    10.10.10.231:corbaloc > 172.20.11.2          :ssh
13:51:52.820950  14124  ESTABLISHED    10.10.10.231:2833 > 172.20.11.2          :domain
13:51:59.704982  15445  ESTABLISHED    10.10.10.174:1055 > 172.20.201.198       :ssh
13:52:08.383043  16793  ESTABLISHED    10.10.10.112:34532 > 192.168.17.66        :ssh
13:52:12.285481  17405  ESTABLISHED    10.10.10.196:51576 > 172.20.201.198       :ssh
13:52:23.371068  19201  ESTABLISHED    10.10.10.234:1041 > 172.20.201.198       :ssh
13:52:31.612497  20300  ESTABLISHED    10.10.10.112:35271 > 192.168.17.66        :domain
13:52:38.188554  21231  ESTABLISHED    10.10.10.231:4170 > 172.20.11.3          :ssh
13:52:38.232891  21249  ESTABLISHED    10.10.10.231:4173 > 172.20.11.3          :smtp
13:52:40.959541  21724  ESTABLISHED    10.10.10.231:4251 > 172.20.11.3          :pop3
13:52:42.374401  21928  ESTABLISHED    10.10.10.231:4284 > 172.20.11.3          :imap
13:52:50.186525  23265  ESTABLISHED    10.10.10.122:59909 > 192.168.17.135       :ftp
13:52:59.085215  25122  ESTABLISHED    10.10.10.122:59914 > 192.168.17.135       :48487
13:53:01.897400  25542  ESTABLISHED    10.10.10.195:2232 > 172.20.11.1          :ssh
13:53:13.053108  27342  ESTABLISHED    10.10.10.122:59918 > 192.168.17.135       :48488
13:53:24.126121  28716  ESTABLISHED    10.10.10.122:59921 > 192.168.17.135       :48489
13:53:27.157552  29045  ESTABLISHED    10.10.10.122:59922 > 192.168.17.135       :48490
13:53:29.597034  29315  ESTABLISHED    10.10.10.231:1565 > 172.20.11.52         :ssh
13:53:33.146924  29722  ESTABLISHED    10.10.10.231:1647 > 172.20.11.52         :sunrpc
13:54:04.050935  32826  ESTABLISHED    10.10.10.122:59928 > 192.168.17.135       :48498
13:54:04.421330  32845  ESTABLISHED    10.10.10.195:2239 > 172.20.11.2          :ssh
13:54:08.621540  33290  ESTABLISHED    10.10.10.122:59930 > 192.168.17.135       :48499
13:54:16.406349  34073  ESTABLISHED    10.10.10.234:1043 > 172.20.201.198       :ssh
13:54:19.108671  34306  ESTABLISHED    10.10.10.186:32801 > 172.20.201.198       :58099
13:54:19.347092  34337  ESTABLISHED    10.10.10.122:59932 > 192.168.17.135       :48504
13:54:23.017992  34645  ESTABLISHED    10.10.10.122:59933 > 192.168.17.135       :48505
13:54:38.144169  36117  ESTABLISHED    10.10.10.122:59936 > 192.168.17.135       :48507
13:54:40.896860  36374  ESTABLISHED    10.10.10.122:59938 > 192.168.17.135       :48508
13:54:42.458396  36486  ESTABLISHED    10.10.10.231:2926 > 172.20.11.80         :ssh
13:54:43.316606  36563  ESTABLISHED    10.10.10.195:2240 > 172.20.11.3          :ssh
13:54:44.999506  36695  ESTABLISHED    10.10.10.231:2977 > 172.20.11.80         :http
13:54:47.074204  36854  ESTABLISHED    10.10.10.231:3008 > 172.20.11.80         :sunrpc
13:55:00.473330  37992  ESTABLISHED    10.10.10.231:3218 > 172.20.11.80         :https
13:55:04.635630  38340  ESTABLISHED    10.10.10.122:59941 > 192.168.17.135       :48509
13:55:26.888924  39896  ESTABLISHED    10.10.10.122:59944 > 192.168.17.135       :48510
13:55:46.033386  41151  ESTABLISHED    10.10.10.195:2241 > 172.20.11.52         :ssh
```

Fig. 1 – sample output of established TCP sessions using *tcpick*

File **Cisco-switch-traffic.txt**

-> CDP/VTP
-> Cisco_17:04:ce LOOP Loopback
-> Cisco_17:04:cf LOOP Loopback
-> Cisco_17:04:d0 LOOP Loopback
-> Cisco_17:04:d2 LOOP Loopback
-> Cisco_17:04:d4 LOOP Loopback
-> Cisco_17:04:d5 LOOP Loopback
-> Cisco_17:04:d6 LOOP Loopback
-> Cisco_17:04:d8 LOOP Loopback

```
-> Spanning-tree-(for-bridges)_00 STP Conf. Root
0.066056 Cisco_17:04:ce -> Spanning-tree-(for-bridges)_00 STP
0.066737 Cisco_17:04:cf -> Spanning-tree-(for-bridges)_00 STP
0.067415 Cisco_17:04:d0 -> Spanning-tree-(for-bridges)_00 STP
0.068094 Cisco_17:04:d2 -> Spanning-tree-(for-bridges)_00 STP
0.068776 Cisco_17:04:d4 -> Spanning-tree-(for-bridges)_00 STP
0.069454 Cisco_17:04:d5 -> Spanning-tree-(for-bridges)_00 STP
0.070173 Cisco_17:04:d6 -> Spanning-tree-(for-bridges)_00 STP
0.070860 Cisco_17:04:d8 -> Spanning-tree-(for-bridges)_00 STP
CDP/VTP
Cisco_17:04:ce -> Cisco_17:04:ce LOOP Loopback
Cisco_17:04:ce -> Spanning-tree-(for-bridges)_00 STP Conf.
Cisco_17:04:ce LOOP Loopback
Cisco_17:04:cf -> Cisco_17:04:cf LOOP Loopback
Cisco_17:04:cf -> Spanning-tree-(for-bridges)_00 STP Conf.
Cisco_17:04:cf LOOP Loopback
Cisco_17:04:d0 -> Cisco_17:04:d0 LOOP Loopback
Cisco_17:04:d0 -> Spanning-tree-(for-bridges)_00 STP Conf.
Cisco_17:04:d0 LOOP Loopback
Cisco_17:04:d2 -> Cisco_17:04:d2 LOOP Loopback
Cisco_17:04:d2 -> Spanning-tree-(for-bridges)_00 STP Conf.
Cisco_17:04:d2 LOOP Loopback
Cisco_17:04:d4 -> Cisco_17:04:d4 LOOP Loopback
Cisco_17:04:d4 -> Spanning-tree-(for-bridges)_00 STP Conf.
Cisco_17:04:d4 LOOP Loopback
Cisco_17:04:d5 -> Cisco_17:04:d5 LOOP Loopback
Cisco_17:04:d5 -> Spanning-tree-(for-bridges)_00 STP Conf.
Cisco_17:04:d5 LOOP Loopback
Cisco_17:04:d6 -> Cisco_17:04:d6 LOOP Loopback
Cisco_17:04:d6 -> Spanning-tree-(for-bridges)_00 STP Conf.
Cisco_17:04:d6 LOOP Loopback
Cisco_17:04:d8 -> CDP/VTP
Cisco_17:04:d8 -> Cisco_17:04:d8 LOOP Loopback
Cisco_17:04:d8 -> Spanning-tree-(for-bridges)_00 STP Conf.
Cisco_17:04:d8 LOOP Loopback
Spanning-tree-(for-bridges)_00 STP Conf. Root =
```

File **Cisco-switch-spanned-ports.txt**

```
   Port identifier: 0x800e
   Port identifier: 0x800f
   Port identifier: 0x8010
   Port identifier: 0x8012
   Port identifier: 0x8014
   Port identifier: 0x8015
```

```
Port identifier: 0x8016
Port identifier: 0x8018
```

File **Cisco-named-ports.txt**

```
Port-ID (0x03), length: 16 bytes: 'FastEthernet0/14'
Port-ID (0x03), length: 16 bytes: 'FastEthernet0/15'
Port-ID (0x03), length: 16 bytes: 'FastEthernet0/16'
Port-ID (0x03), length: 16 bytes: 'FastEthernet0/18'
Port-ID (0x03), length: 16 bytes: 'FastEthernet0/20'
Port-ID (0x03), length: 16 bytes: 'FastEthernet0/21'
Port-ID (0x03), length: 16 bytes: 'FastEthernet0/22'
Port-ID (0x03), length: 16 bytes: 'FastEthernet0/24'
```

File **mac-to-vendor-to-ip.txt** (manually removed MAC duplicates, to better reveal the multi-IPs associated with one MAC)

```
TOSHIBA:f2:67:88        10.10.10.117
Acer:92:ee:0f                   10.10.10.222
Acer:94:b0:2a           10.10.10.226
3COM:79:91:ed           10.10.10.112
3COM:88:29:92           10.10.10.234
Compaq:b6:e2:e3         10.10.10.186
Intel:8c:89:c2                  10.10.10.165
                        192.168.117.1
                        192.168.213.1
Microsoft:df:95:84      10.10.10.228
3COM:45:61:39           10.10.10.195
Dell:d8:bf:ed           10.10.10.122
Dell:e6:f8:43           10.10.10.231
Dell:05:b7:f8           10.10.10.147
Dell:07:31:ee                   0.0.0.0
                        10.10.10.111
                        172.16.8.189
IBM:02:e9:3d                    10.10.10.212
                        172.16.8.229
Apple:7c:24:00          10.10.10.113
                        10.10.10.232
Dell:17:f4:c9           0.0.0.0
                        10.10.10.194
                        169.254.135.50
                        172.16.9.13
                        192.168.222.1
                        192.168.84.1
```

Dell:9b:46:fe	10.10.10.164
Dell:df:53:8d	10.10.10.123
VMWare,:14:1e:63	0.0.0.0
	10.10.10.142
VMWare,:39:6e:67	0.0.0.0
	10.10.10.160
VMWare,:9e:ef:53	10.10.10.224
VMWare,:40:00:64	10.10.10.2
VMWare,:40:00:6d	10.10.10.1
	10.30.30.2
	172.20.11.1
	172.20.11.2
	172.20.11.3
	172.20.11.52
	172.20.11.80
	172.20.201.1
	172.20.201.135
	172.20.201.198
	172.20.201.2
	192.168.17.129
	192.168.17.135
	192.168.17.2
	192.168.17.65
	192.168.17.66
	192.168.17.68
	192.168.22.207
Intel:ba:6d:85	10.10.10.196
AMBIT:c6:5e:14	0.0.0.0
	10.10.10.141
	10.10.10.144
	238.122.10.140
AboCom:a1:7f:da	10.10.10.174
AboCom:3d:20:40	10.10.10.214
SONY:79:f7:7c	0.0.0.0
	10.10.10.230

Appendix B – various ntop screenshots

Due to space constraints, and in an attempt to utilize the best screen capture vs. paper size ratio, I will summarize here the content and provide titles of all the following pictures. I will also provide only the **first** screen or **page** for every category/type of information, as the information for all the hosts found in the dump file would have taken hundreds of pages.

Fig1 – screen shot of host information, organized by **bandwidth** having been “consumed” during the trace

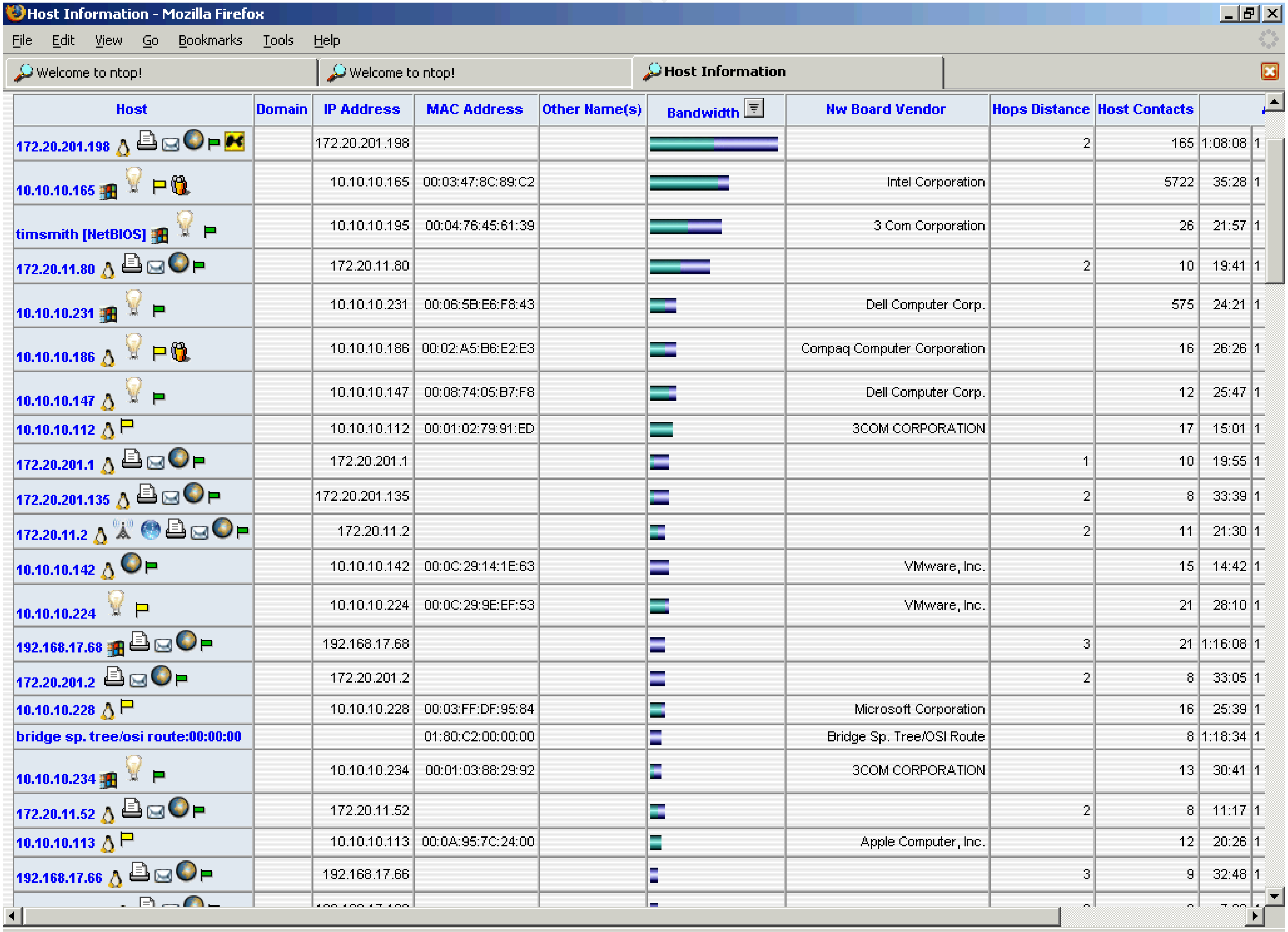
Fig2 – screen shot of host information, organized by **distance** (hops number), **from the 10.10.10.0/24 network**. This is derived from the TTL values from individual systems, and is used to identify entire networks distance (for the diagram), based on some of the hosts belonging to those networks.

Fig3 – screen shot of host information, organized by the **number of other hosts having been contacted**.

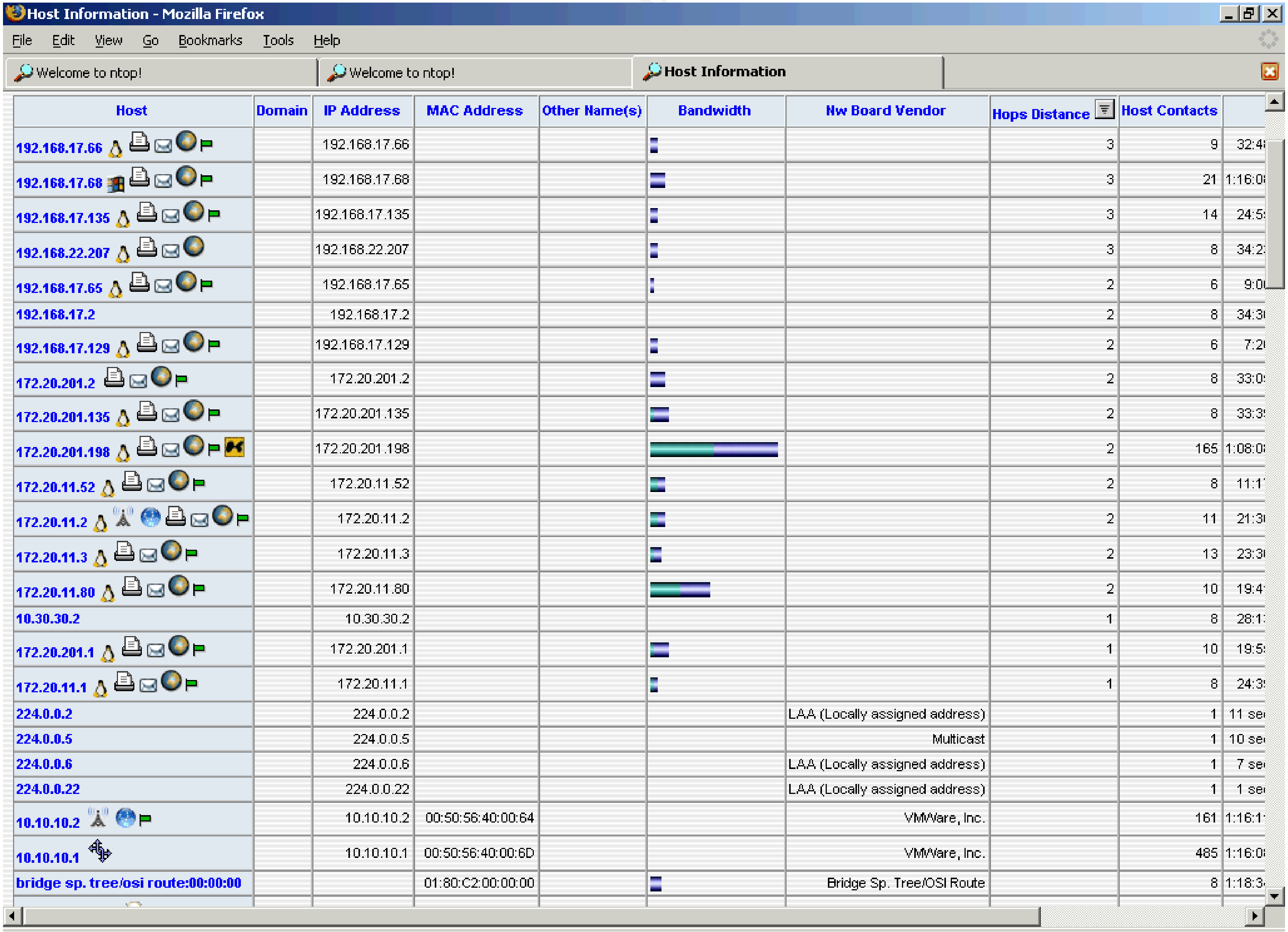
Fig4-ver.3.1CVS – 3 pages (this information is of outmost criticality, and I have decided to present it in its entirety, even if taking more than one page) - screen shots of **host fingerprinting information**, derived from using ***ntop*** version 3.1CVS (unstable at the time of this paper)

Fig5-ver.3.0 – same as above, but for ***ntop*** version 3.0 (stable at the time of this writing)

NOTE: As it can be seen from the screen shots for version 3.0 and 3.1CVS, the former had more comprehensive information regarding the client detects, while the latter was better at identifying the OS

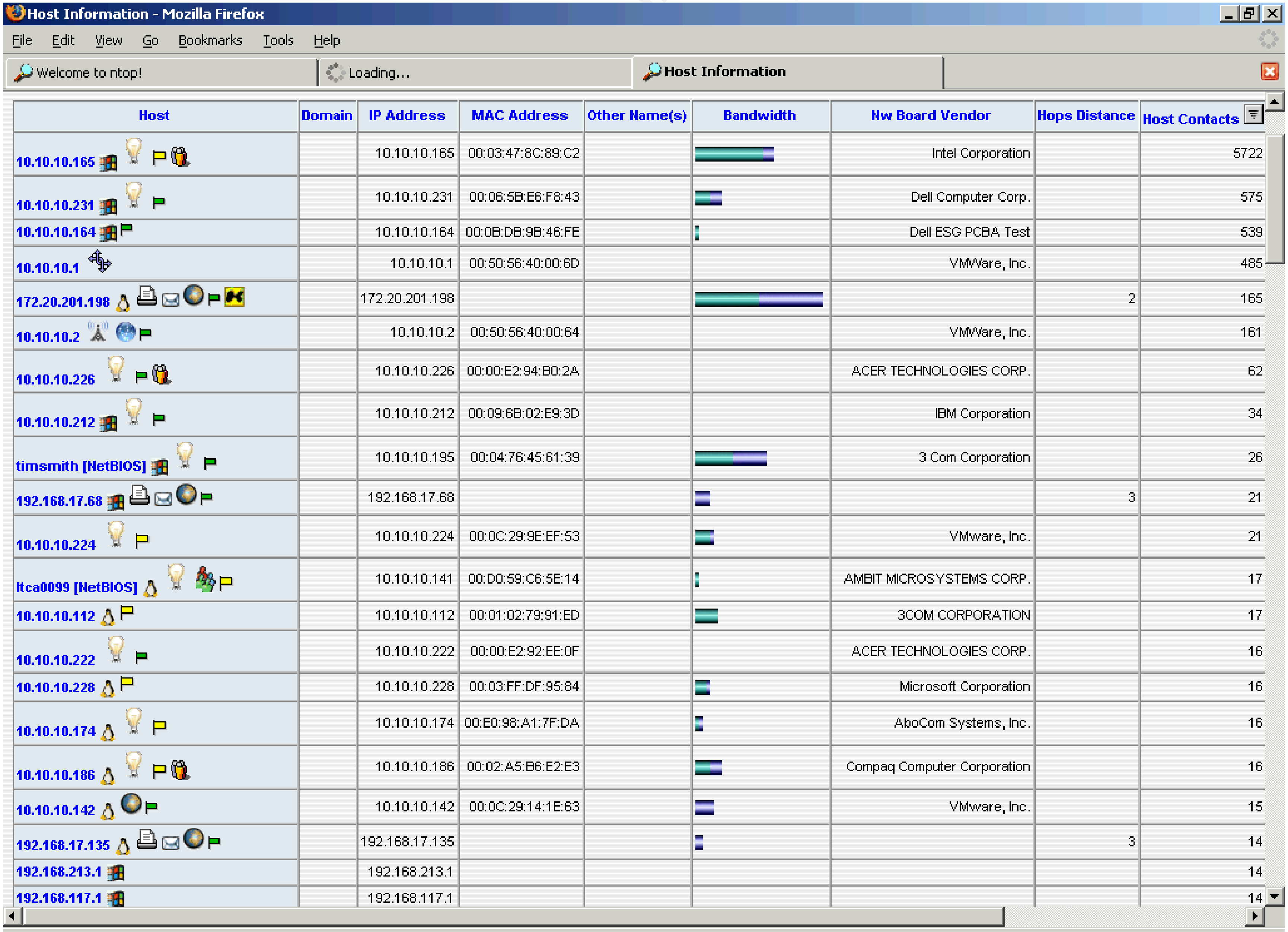


Host	Domain	IP Address	MAC Address	Other Name(s)	Bandwidth	Nw Board Vendor	Hops Distance	Host Contacts		
172.20.201.198		172.20.201.198					2	165	1:08:08	1
10.10.10.165		10.10.10.165	00:03:47:8C:89:C2			Intel Corporation		5722	35:28	1
timsmith [NetBIOS]		10.10.10.195	00:04:76:45:61:39			3 Com Corporation		26	21:57	1
172.20.11.80		172.20.11.80					2	10	19:41	1
10.10.10.231		10.10.10.231	00:06:5B:E6:F8:43			Dell Computer Corp.		575	24:21	1
10.10.10.186		10.10.10.186	00:02:A5:B6:E2:E3			Compaq Computer Corporation		16	26:26	1
10.10.10.147		10.10.10.147	00:08:74:05:B7:F8			Dell Computer Corp.		12	25:47	1
10.10.10.112		10.10.10.112	00:01:02:79:91:ED			3COM CORPORATION		17	15:01	1
172.20.201.1		172.20.201.1					1	10	19:55	1
172.20.201.135		172.20.201.135					2	8	33:39	1
172.20.11.2		172.20.11.2					2	11	21:30	1
10.10.10.142		10.10.10.142	00:0C:29:14:1E:63			VMware, Inc.		15	14:42	1
10.10.10.224		10.10.10.224	00:0C:29:9E:EF:53			VMware, Inc.		21	28:10	1
192.168.17.68		192.168.17.68					3	21	1:16:08	1
172.20.201.2		172.20.201.2					2	8	33:05	1
10.10.10.228		10.10.10.228	00:03:FF:DF:95:84			Microsoft Corporation		16	25:39	1
bridge sp. tree/osi route:00:00:00			01:80:C2:00:00:00			Bridge Sp. Tree/OSI Route		8	1:18:34	1
10.10.10.234		10.10.10.234	00:01:03:88:29:92			3COM CORPORATION		13	30:41	1
172.20.11.52		172.20.11.52					2	8	11:17	1
10.10.10.113		10.10.10.113	00:0A:95:7C:24:00			Apple Computer, Inc.		12	20:26	1
192.168.17.66		192.168.17.66					3	9	32:48	1

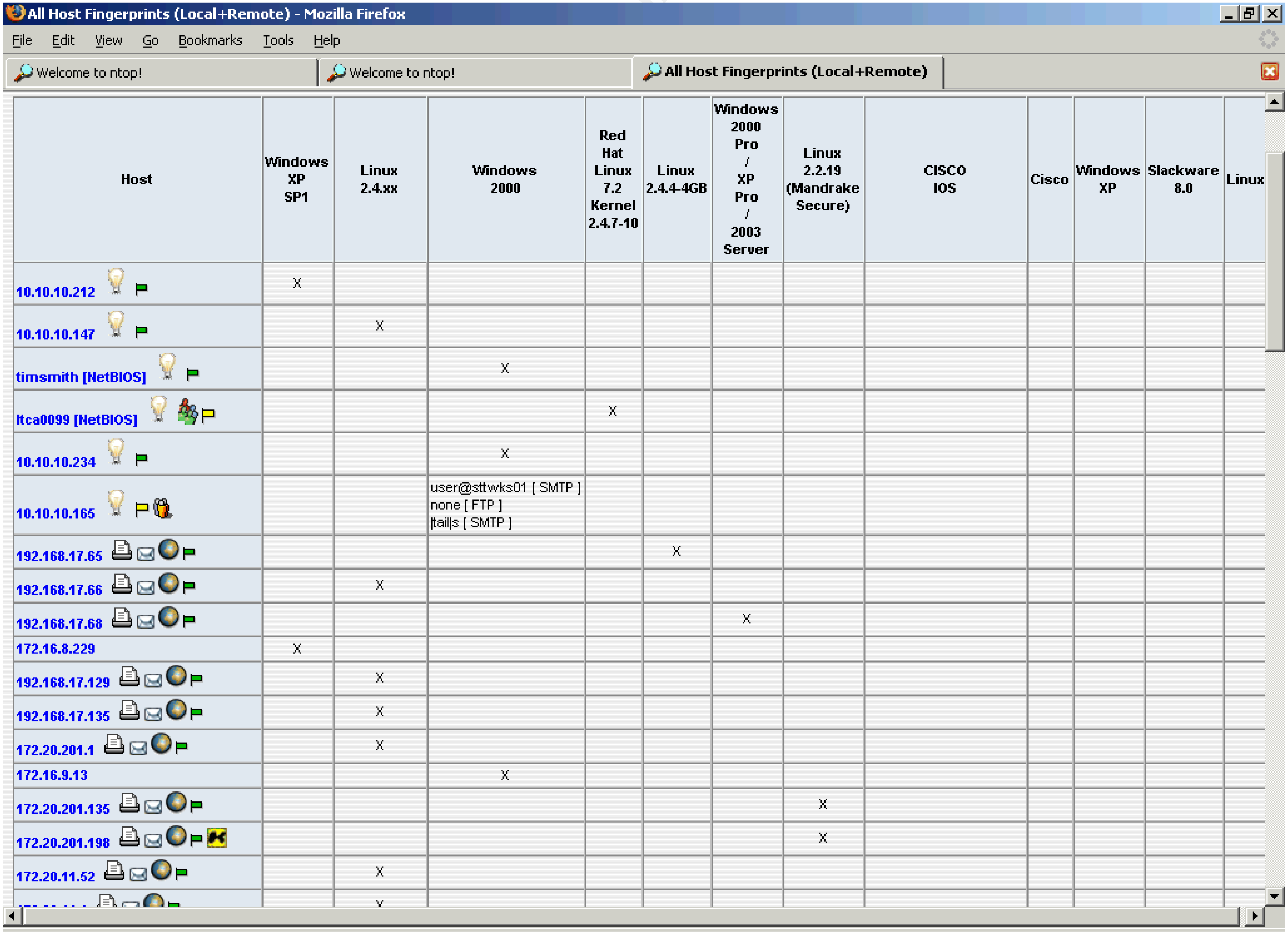


Host Information - Mozilla Firefox

Host	Domain	IP Address	MAC Address	Other Name(s)	Bandwidth	Nw Board Vendor	Hops Distance	Host Contacts	
192.168.17.66		192.168.17.66					3	9	32:4
192.168.17.68		192.168.17.68					3	21	1:16:0
192.168.17.135		192.168.17.135					3	14	24:5
192.168.22.207		192.168.22.207					3	8	34:2
192.168.17.65		192.168.17.65					2	6	9:0
192.168.17.2		192.168.17.2					2	8	34:3
192.168.17.129		192.168.17.129					2	6	7:2
172.20.201.2		172.20.201.2					2	8	33:0
172.20.201.135		172.20.201.135					2	8	33:3
172.20.201.198		172.20.201.198					2	165	1:08:0
172.20.11.52		172.20.11.52					2	8	11:1
172.20.11.2		172.20.11.2					2	11	21:3
172.20.11.3		172.20.11.3					2	13	23:3
172.20.11.80		172.20.11.80					2	10	19:4
10.30.30.2		10.30.30.2					1	8	28:1
172.20.201.1		172.20.201.1					1	10	19:5
172.20.11.1		172.20.11.1					1	8	24:3
224.0.0.2		224.0.0.2				LAA (Locally assigned address)		1	11 se
224.0.0.5		224.0.0.5				Multicast		1	10 se
224.0.0.6		224.0.0.6				LAA (Locally assigned address)		1	7 se
224.0.0.22		224.0.0.22				LAA (Locally assigned address)		1	1 se
10.10.10.2		10.10.10.2	00:50:56:40:00:64			VMWare, Inc.		161	1:16:1
10.10.10.1		10.10.10.1	00:50:56:40:00:6D			VMWare, Inc.		485	1:16:0
bridge sp. tree/osi route:00:00:00			01:80:C2:00:00:00			Bridge Sp. Tree/OSI Route		8	1:18:3

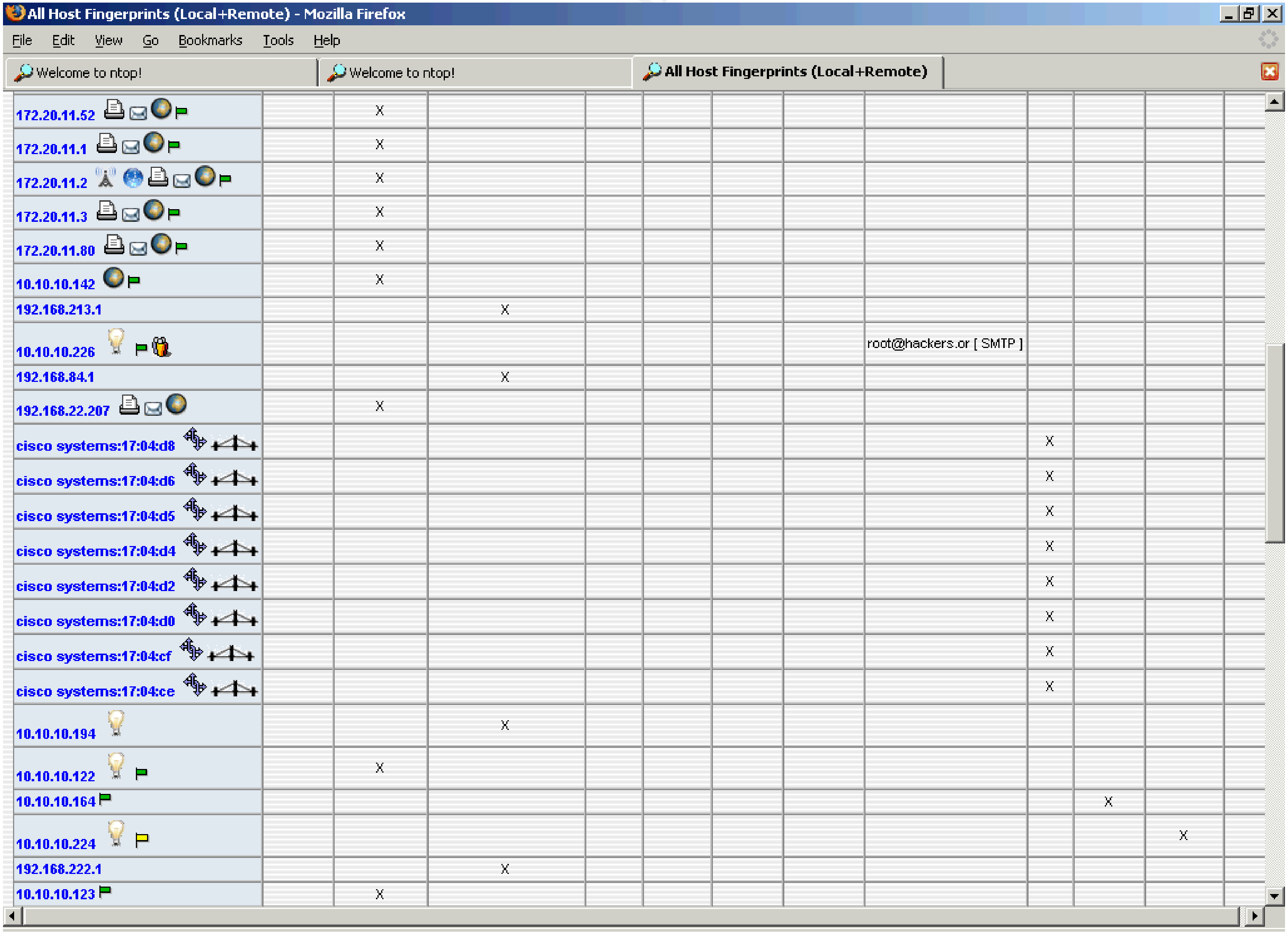


Host	Domain	IP Address	MAC Address	Other Name(s)	Bandwidth	Nw Board Vendor	Hops Distance	Host Contacts
10.10.10.165		10.10.10.165	00:03:47:8C:89:C2			Intel Corporation		5722
10.10.10.231		10.10.10.231	00:06:5B:E6:F8:43			Dell Computer Corp.		575
10.10.10.164		10.10.10.164	00:0B:DB:9B:46:FE			Dell ESG PCBA Test		539
10.10.10.1		10.10.10.1	00:50:56:40:00:6D			VMWare, Inc.		485
172.20.201.198		172.20.201.198					2	165
10.10.10.2		10.10.10.2	00:50:56:40:00:64			VMWare, Inc.		161
10.10.10.226		10.10.10.226	00:00:E2:94:B0:2A			ACER TECHNOLOGIES CORP.		62
10.10.10.212		10.10.10.212	00:09:6B:02:E9:3D			IBM Corporation		34
timsmith [NetBIOS]		10.10.10.195	00:04:76:45:61:39			3 Com Corporation		26
192.168.17.68		192.168.17.68					3	21
10.10.10.224		10.10.10.224	00:0C:29:9E:EF:53			VMware, Inc.		21
ltca0099 [NetBIOS]		10.10.10.141	00:D0:59:C6:5E:14			AMBIT MICROSYSTEMS CORP.		17
10.10.10.112		10.10.10.112	00:01:02:79:91:ED			3COM CORPORATION		17
10.10.10.222		10.10.10.222	00:00:E2:92:EE:0F			ACER TECHNOLOGIES CORP.		16
10.10.10.228		10.10.10.228	00:03:FF:DF:95:84			Microsoft Corporation		16
10.10.10.174		10.10.10.174	00:E0:98:A1:7F:DA			AboCom Systems, Inc.		16
10.10.10.186		10.10.10.186	00:02:A5:B6:E2:E3			Compaq Computer Corporation		16
10.10.10.142		10.10.10.142	00:0C:29:14:1E:63			VMware, Inc.		15
192.168.17.135		192.168.17.135					3	14
192.168.213.1		192.168.213.1						14
192.168.117.1		192.168.117.1						14

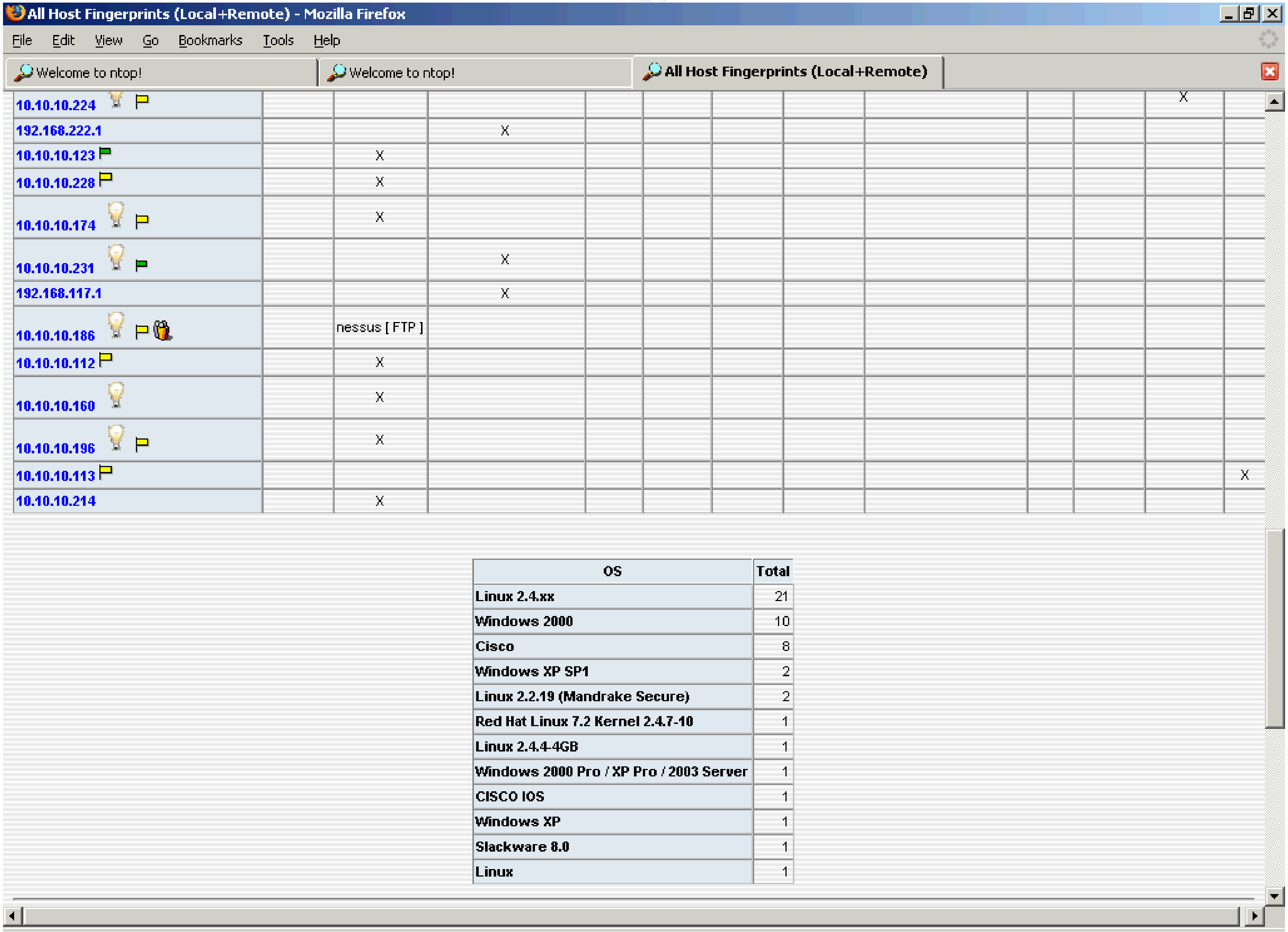


All Host Fingerprints (Local+Remote) - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Welcome to ntop!
Welcome to ntop!
All Host Fingerprints (Local+Remote)

Host	Windows XP SP1	Linux 2.4.xx	Windows 2000	Red Hat Linux 7.2 Kernel 2.4.7-10	Linux 2.4.4-4GB	Windows 2000 Pro / XP Pro / 2003 Server	Linux 2.2.19 (Mandrake Secure)	CISCO IOS	Cisco	Windows XP	Slackware 8.0	Linux
10.10.10.212	X											
10.10.10.147		X										
timsmith [NetBIOS]			X									
ltca0099 [NetBIOS]				X								
10.10.10.234			X									
10.10.10.165			user@sttwks01 [SMTP] none [FTP] |tail|s [SMTP]									
192.168.17.65					X							
192.168.17.66		X										
192.168.17.68						X						
172.16.8.229	X											
192.168.17.129		X										
192.168.17.135		X										
172.20.201.1		X										
172.16.9.13			X									
172.20.201.135							X					
172.20.201.198							X					
172.20.11.52		X										
[illegible]		X										

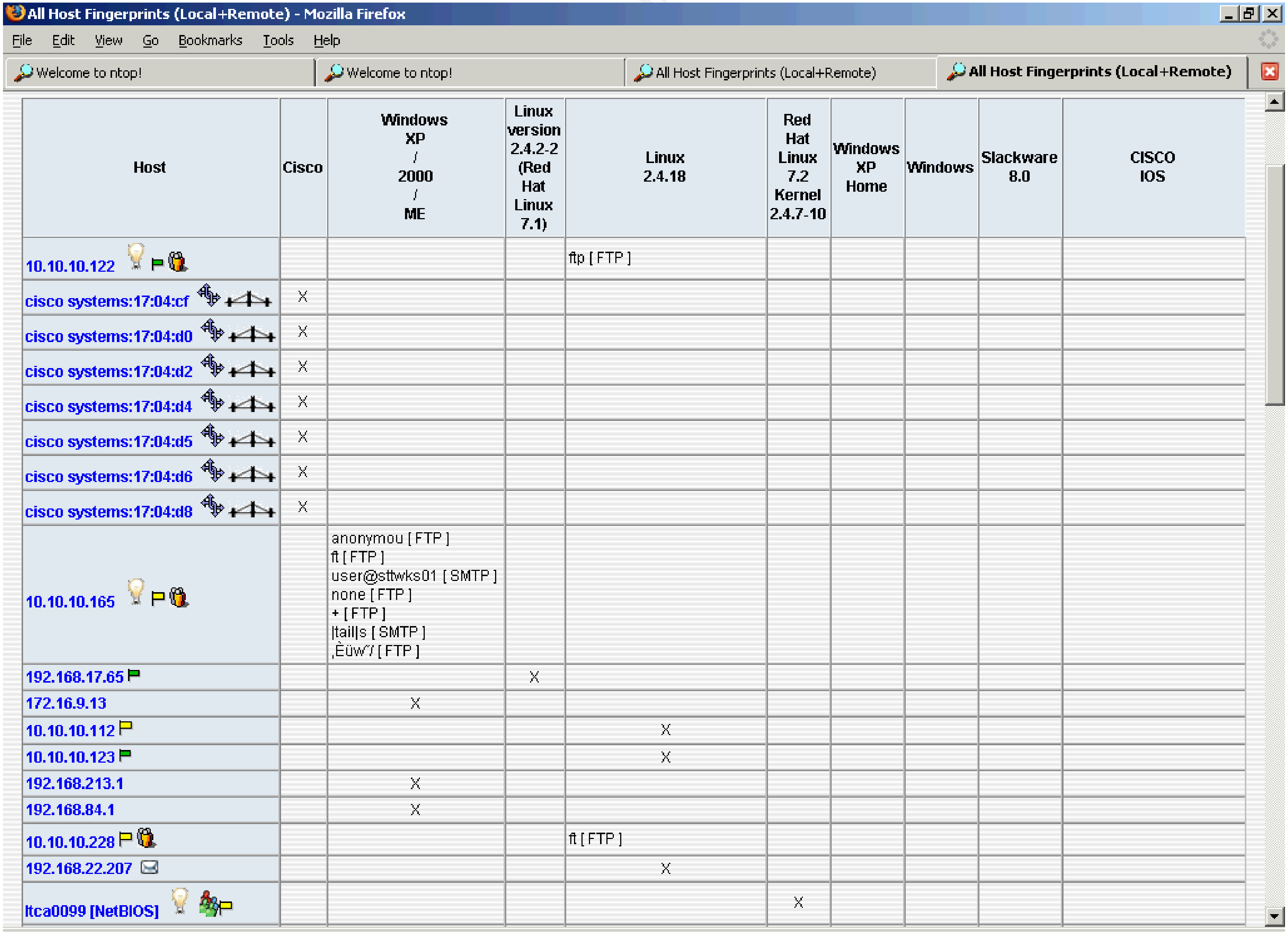

All Host Fingerprints (Local+Remote) - Mozilla Firefox
File
Edit
View
Go
Bookmarks
Tools
Help
Welcome to ntop!
Welcome to ntop!
All Host Fingerprints (Local+Remote)

172.20.11.52		X										
172.20.11.1		X										
172.20.11.2		X										
172.20.11.3		X										
172.20.11.80		X										
10.10.10.142		X										
192.168.213.1			X									
10.10.10.226								root@hackers.or [SMTP]				
192.168.84.1			X									
192.168.22.207		X										
cisco systems:17:04:d8									X			
cisco systems:17:04:d6									X			
cisco systems:17:04:d5									X			
cisco systems:17:04:d4									X			
cisco systems:17:04:d2									X			
cisco systems:17:04:d0									X			
cisco systems:17:04:cf									X			
cisco systems:17:04:ce									X			
10.10.10.194			X									
10.10.10.122		X										
10.10.10.164										X		
10.10.10.224											X	
192.168.222.1			X									
10.10.10.123		X										

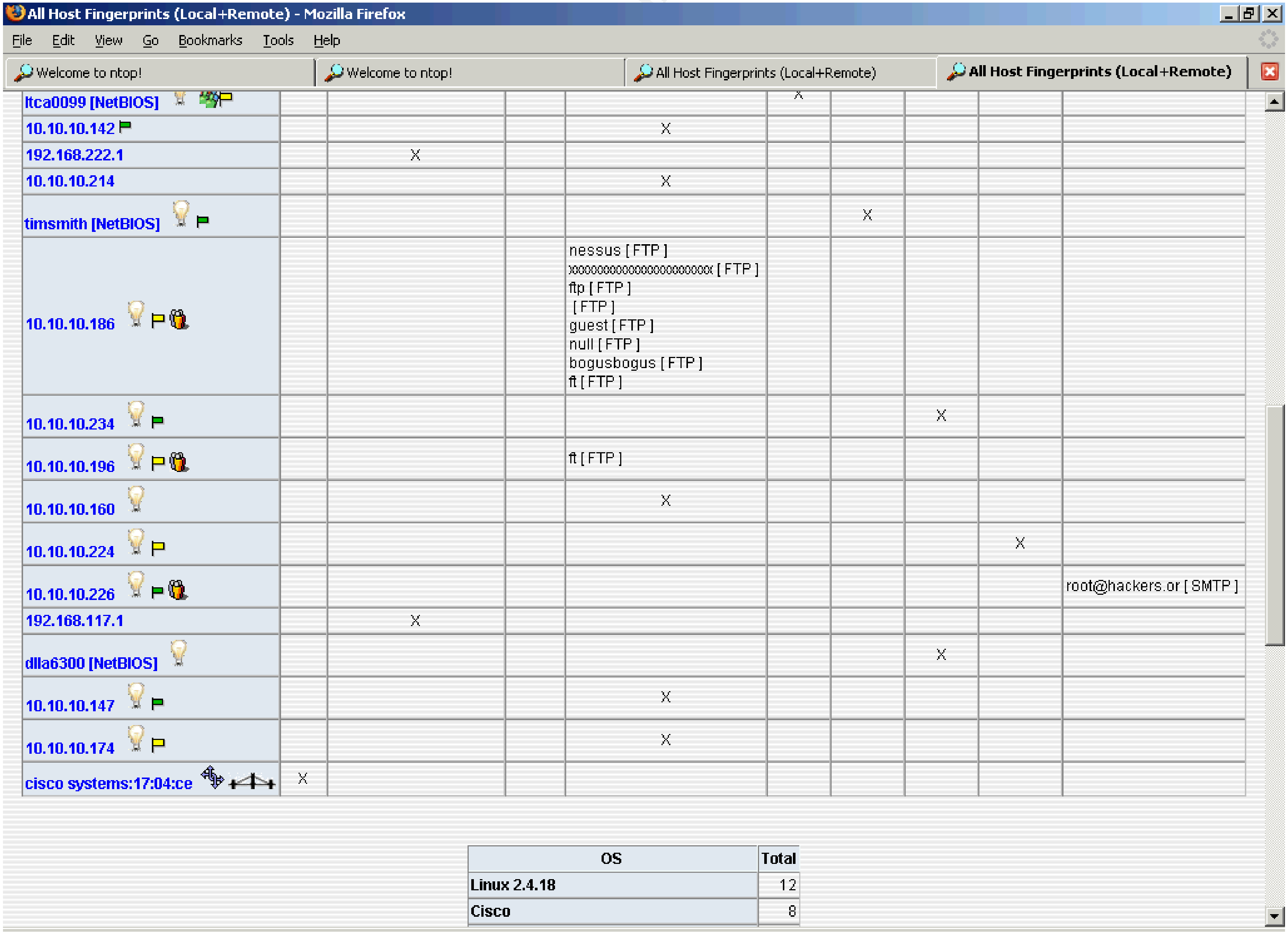

All Host Fingerprints (Local+Remote) - Mozilla Firefox
File
Edit
View
Go
Bookmarks
Tools
Help
Welcome to ntop!
Welcome to ntop!
All Host Fingerprints (Local+Remote)

10.10.10.224											X	
192.168.222.1			X									
10.10.10.123		X										
10.10.10.228		X										
10.10.10.174		X										
10.10.10.231			X									
192.168.117.1			X									
10.10.10.186		nessus [FTP]										
10.10.10.112		X										
10.10.10.160		X										
10.10.10.196		X										
10.10.10.113												X
10.10.10.214		X										

OS	Total
Linux 2.4.xx	21
Windows 2000	10
Cisco	8
Windows XP SP1	2
Linux 2.2.19 (Mandrake Secure)	2
Red Hat Linux 7.2 Kernel 2.4.7-10	1
Linux 2.4.4-4GB	1
Windows 2000 Pro / XP Pro / 2003 Server	1
CISCO IOS	1
Windows XP	1
Slackware 8.0	1
Linux	1

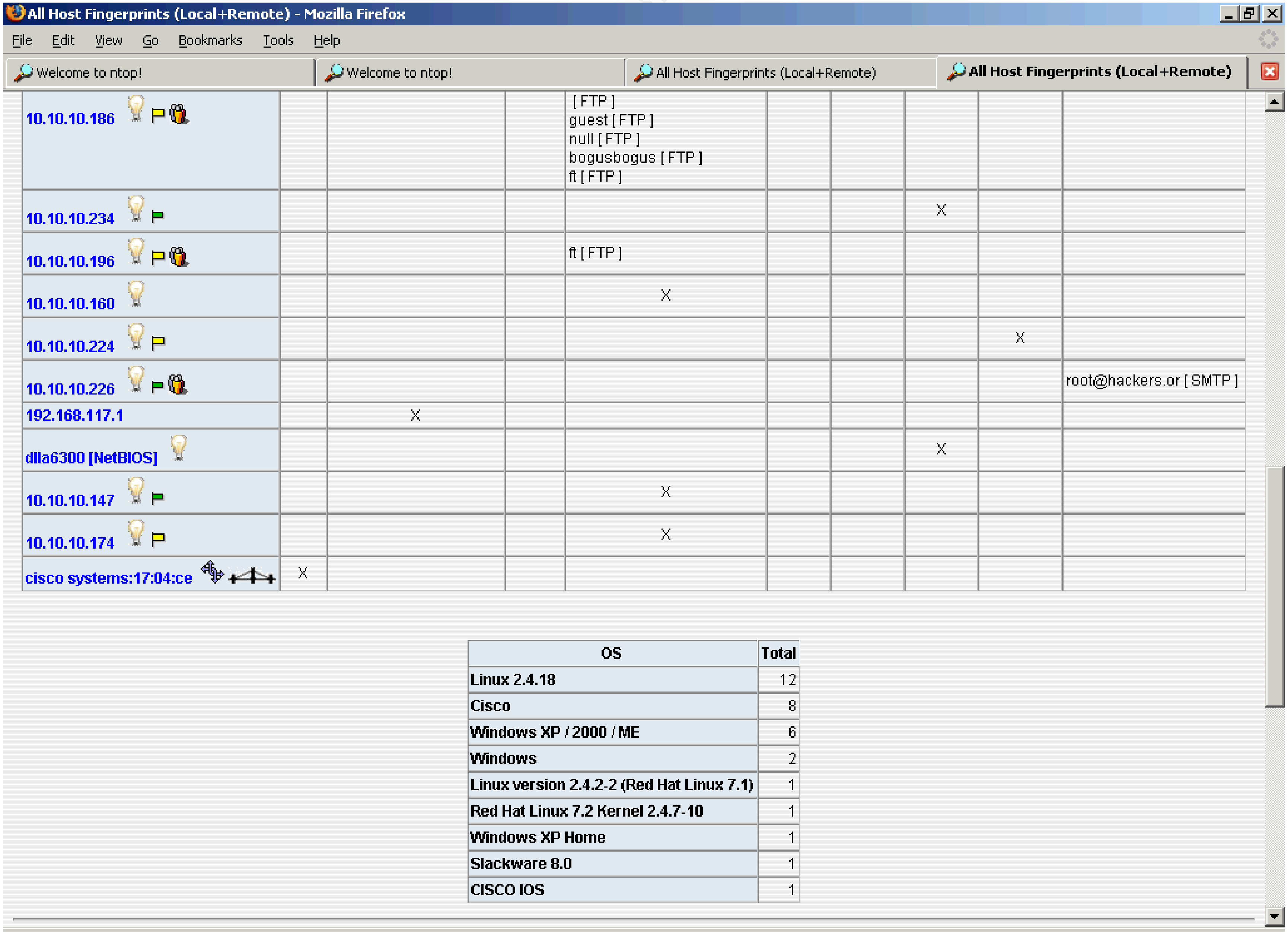

All Host Fingerprints (Local+Remote) - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Welcome to ntop!
Welcome to ntop!
All Host Fingerprints (Local+Remote)
All Host Fingerprints (Local+Remote)

Host	Cisco	Windows XP / 2000 / ME	Linux version 2.4.2-2 (Red Hat Linux 7.1)	Linux 2.4.18	Red Hat Linux 7.2 Kernel 2.4.7-10	Windows XP Home	Windows	Slackware 8.0	CISCO IOS
10.10.10.122				ftp [FTP]					
cisco systems:17:04:cf	X								
cisco systems:17:04:d0	X								
cisco systems:17:04:d2	X								
cisco systems:17:04:d4	X								
cisco systems:17:04:d5	X								
cisco systems:17:04:d6	X								
cisco systems:17:04:d8	X								
10.10.10.165		anonymou [FTP] ft [FTP] user@sttwks01 [SMTP] none [FTP] + [FTP] |tail|s [SMTP] ‚Èüw'/ [FTP]							
192.168.17.65			X						
172.16.9.13		X							
10.10.10.112				X					
10.10.10.123				X					
192.168.213.1		X							
192.168.84.1		X							
10.10.10.228				ft [FTP]					
192.168.22.207				X					
ltca0099 [NetBIOS]					X				


All Host Fingerprints (Local+Remote) - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Welcome to ntop!
Welcome to ntop!
All Host Fingerprints (Local+Remote)
All Host Fingerprints (Local+Remote)

ltca0099 [NetBIOS]					X				
10.10.10.142				X					
192.168.222.1		X							
10.10.10.214				X					
timsmith [NetBIOS]						X			
10.10.10.186				nessus [FTP] xxxxxxxxxxxxxxxxxxxxxxx [FTP] ftp [FTP] [FTP] guest [FTP] null [FTP] bogusbogus [FTP] ft [FTP]					
10.10.10.234							X		
10.10.10.196				ft [FTP]					
10.10.10.160				X					
10.10.10.224								X	
10.10.10.226									root@hackers.or [SMTP]
192.168.117.1		X							
dlla6300 [NetBIOS]							X		
10.10.10.147				X					
10.10.10.174				X					
cisco systems:17:04:ce	X								

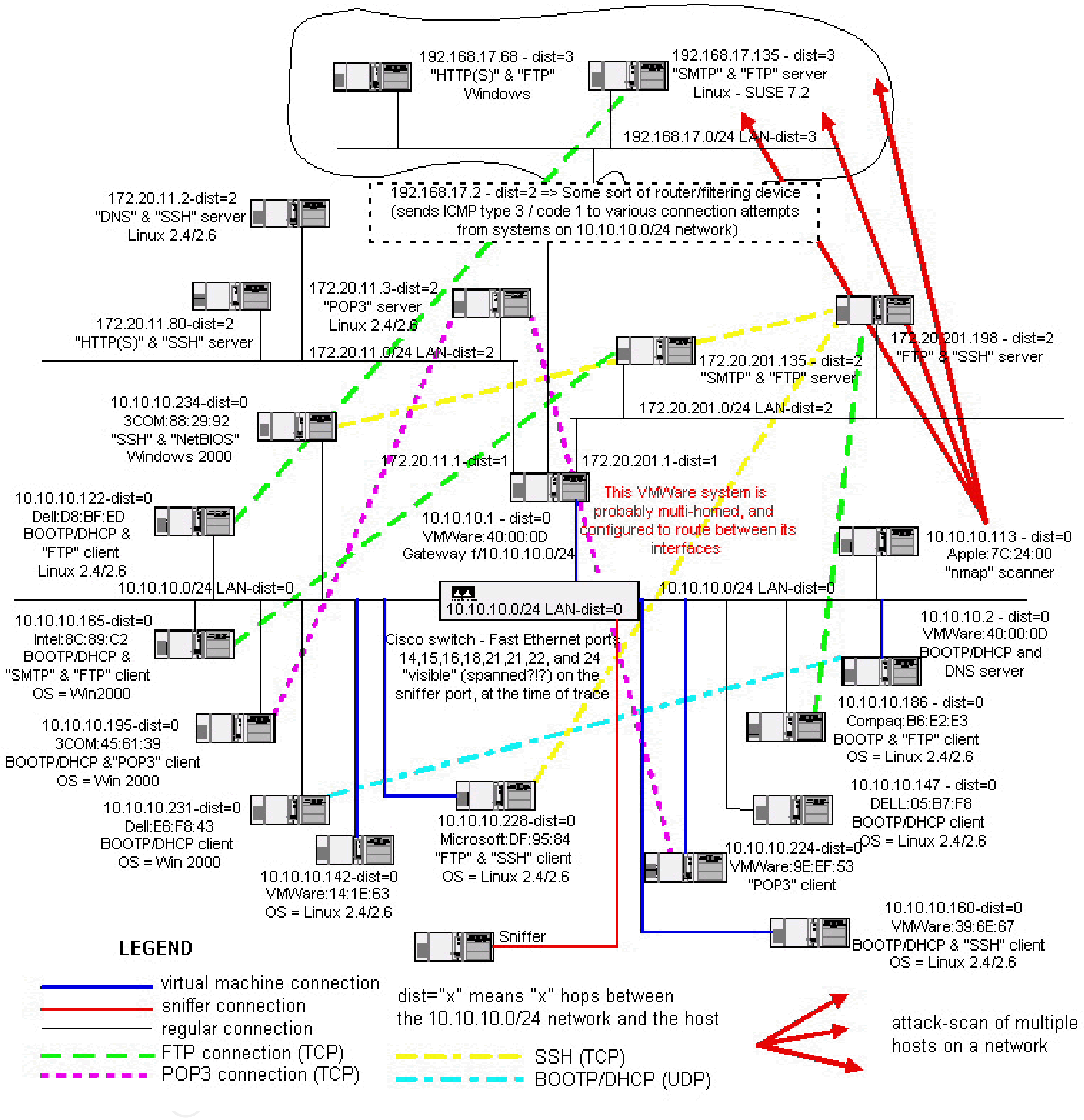
OS	Total
Linux 2.4.18	12
Cisco	8


All Host Fingerprints (Local+Remote) - Mozilla Firefox
File
Edit
View
Go
Bookmarks
Tools
Help
Welcome to ntop!
Welcome to ntop!
All Host Fingerprints (Local+Remote)
All Host Fingerprints (Local+Remote)

10.10.10.186				[FTP] guest [FTP] null [FTP] bogusbogus [FTP] ft [FTP]					
10.10.10.234							X		
10.10.10.196				ft [FTP]					
10.10.10.160				X					
10.10.10.224								X	
10.10.10.226									root@hackers.or [SMTP]
192.168.117.1		X							
dlla6300 [NetBIOS]							X		
10.10.10.147				X					
10.10.10.174				X					
cisco systems:17:04:ce	X								

OS	Total
Linux 2.4.18	12
Cisco	8
Windows XP / 2000 / ME	6
Windows	2
Linux version 2.4.2-2 (Red Hat Linux 7.1)	1
Red Hat Linux 7.2 Kernel 2.4.7-10	1
Windows XP Home	1
Slackware 8.0	1
CISCO IOS	1

Appendix C – network diagram + sample traffic



Appendix D – preliminary detects analysis

With stream4:

```
================================================================

Snort processed 475199 packets.
================================================================
Breakdown by protocol:
 TCP: 372578     (78.405%)
 UDP: 66543      (14.003%)
 ICMP: 9986      (2.101%)
 ARP: 1329       (0.280%)
 EAPOL: 0         (0.000%)
 IPv6: 0         (0.000%)
 IPX: 0          (0.000%)
 OTHER: 23582      (4.963%)
 DISCARD: 1181       (0.249%)
================================================================
Action Stats:
ALERTS: 41838
LOGGED: 41825
PASSED: 0
================================================================
```

Without stream4:

```
Snort processed 475199 packets.
================================================================
Breakdown by protocol:
 TCP: 372578     (78.405%)
 UDP: 66543      (14.003%)
 ICMP: 9986      (2.101%)
  ARP: 1329      (0.280%)
  EAPOL: 0         (0.000%)
  IPv6: 0         (0.000%)
  IPX: 0         (0.000%)
  OTHER: 23582      (4.963%)
 DISCARD: 1181       (0.249%)
================================================================
Action Stats:
ALERTS: 41684
LOGGED: 41763
PASSED: 0
================================================================
```

Sequence of running the *sguil* tools:

1. As user sguil:

```
[sguil@Stef sguil-server-directory]$ ./sguild -c sguild.conf -u sguild.users
```

2. As user root:

```
[root@Stef root]# snort -u sguil -g sguil
-c /usr/local/etc/snort/snort.conf -U -l /nsm/localhost -m 122 -A none
-r /home/scm/GIAC/my-work/2003.12.15.cap
```

3. As user sguil:

```
[sguil@Stef snort-config-directory]$ barnyard -c barnyard.conf -d /nsm/localhost -g
gen-msg.map -s sid-msg.map -f snort.log -w -waldo.file
```

4. As user sguil:

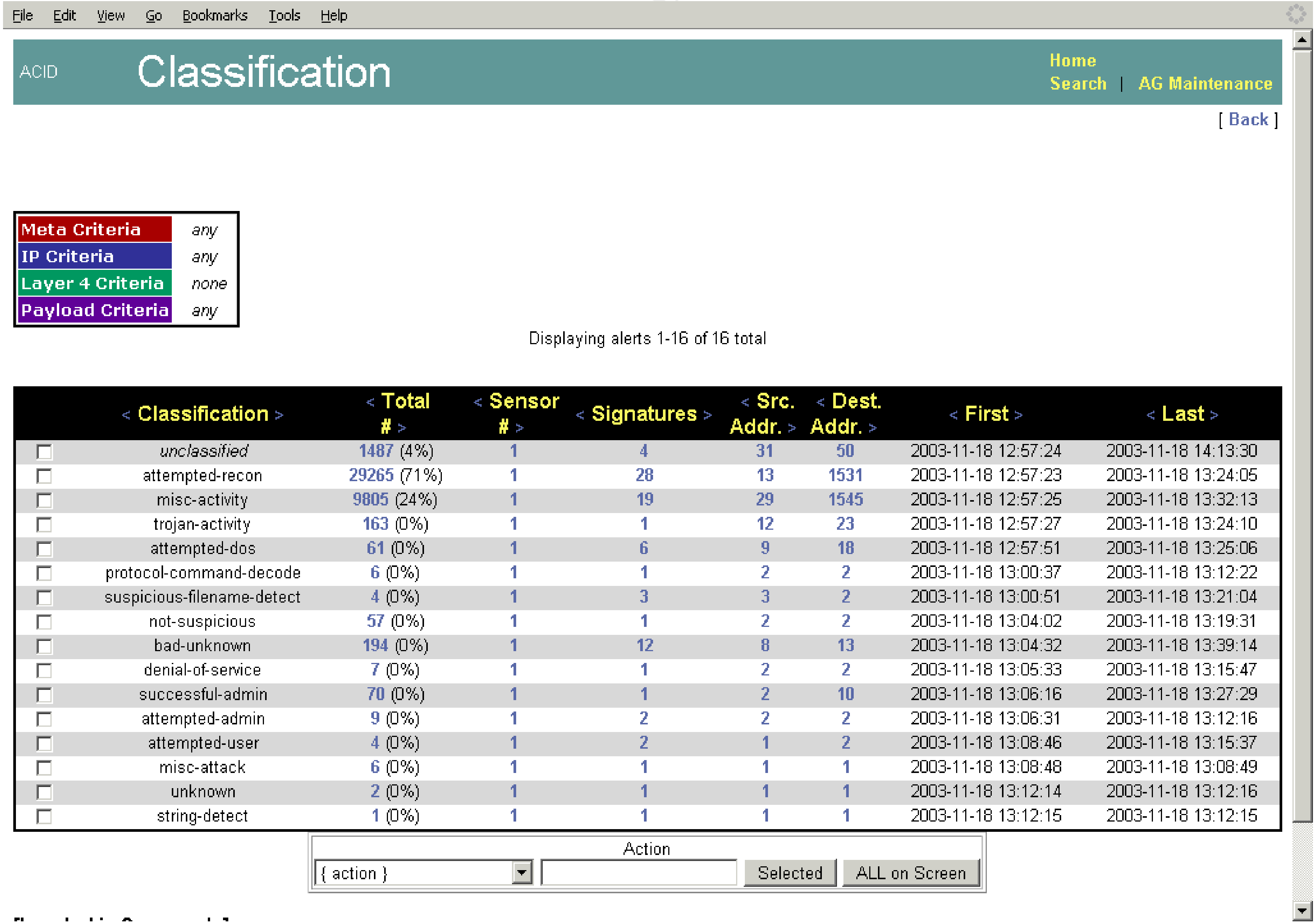
```
[sguil@Stef sguil-sensor-directory]$ ./sensor_agent.tcl
```

5. As user root:

```
[root@Stef sguil-sensor-directory]# ./log_packets.sh start
```

6. Finally - start the sguil client:

```
[sguil@Stef sguil-client-directory]$ ./sguil.tk
```



ACID

Classification

Home
Search | AG Maintenance

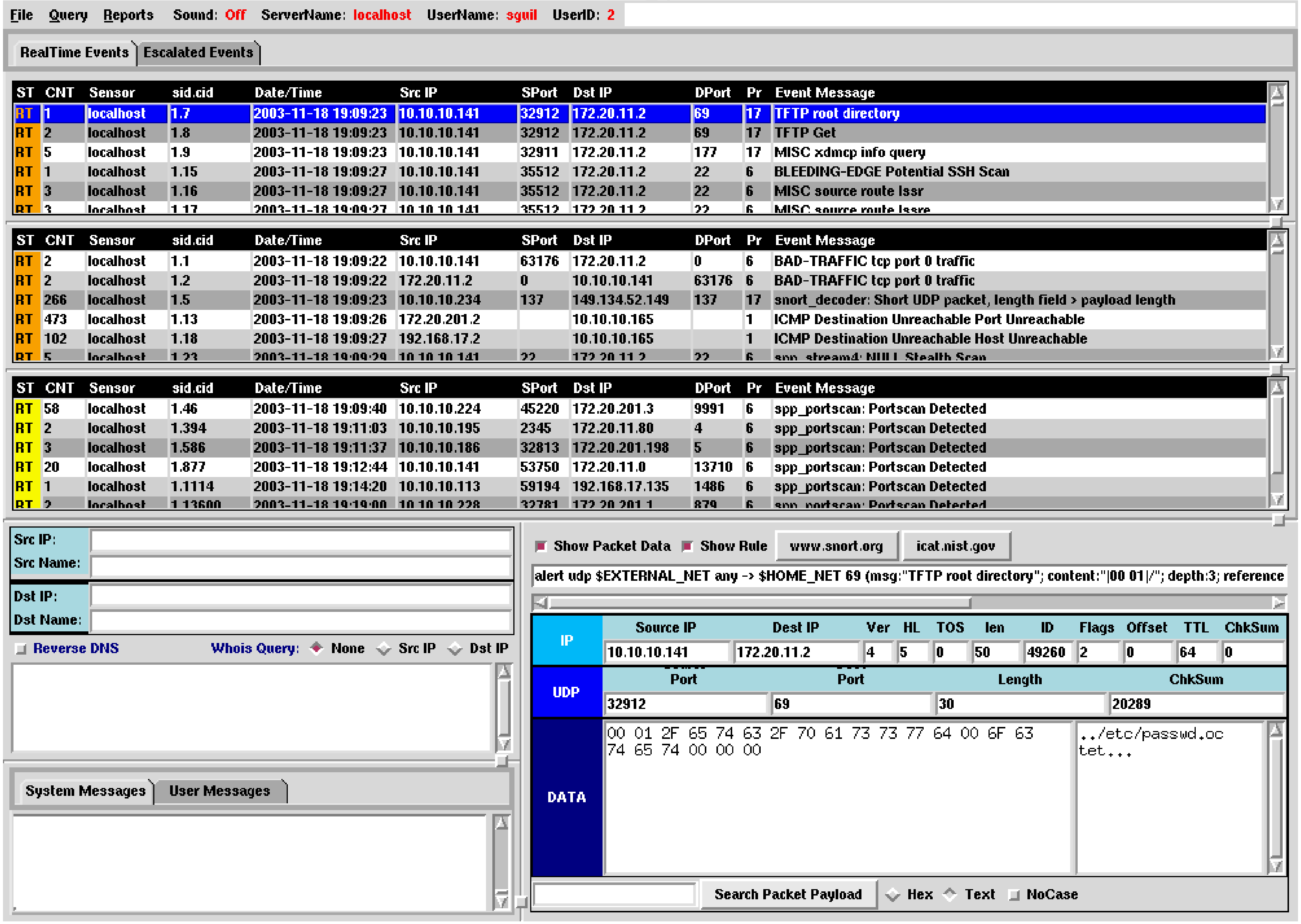
[Back]

Meta Criteria	*any*
IP Criteria	*any*
Layer 4 Criteria	*none*
Payload Criteria	*any*

Displaying alerts 1-16 of 16 total

	< Classification >	< Total # >	< Sensor # >	< Signatures >	< Src. Addr. >	< Dest. Addr. >	< First >	< Last >
□	*unclassified*	1487 (4%)	1	4	31	50	2003-11-18 12:57:24	2003-11-18 14:13:30
□	attempted-recon	29265 (71%)	1	28	13	1531	2003-11-18 12:57:23	2003-11-18 13:24:05
□	misc-activity	9805 (24%)	1	19	29	1545	2003-11-18 12:57:25	2003-11-18 13:32:13
□	trojan-activity	163 (0%)	1	1	12	23	2003-11-18 12:57:27	2003-11-18 13:24:10
□	attempted-dos	61 (0%)	1	6	9	18	2003-11-18 12:57:51	2003-11-18 13:25:06
□	protocol-command-decode	6 (0%)	1	1	2	2	2003-11-18 13:00:37	2003-11-18 13:12:22
□	suspicious-filename-detect	4 (0%)	1	3	3	2	2003-11-18 13:00:51	2003-11-18 13:21:04
□	not-suspicious	57 (0%)	1	1	2	2	2003-11-18 13:04:02	2003-11-18 13:19:31
□	bad-unknown	194 (0%)	1	12	8	13	2003-11-18 13:04:32	2003-11-18 13:39:14
□	denial-of-service	7 (0%)	1	1	2	2	2003-11-18 13:05:33	2003-11-18 13:15:47
□	successful-admin	70 (0%)	1	1	2	10	2003-11-18 13:06:16	2003-11-18 13:27:29
□	attempted-admin	9 (0%)	1	2	2	2	2003-11-18 13:06:31	2003-11-18 13:12:16
□	attempted-user	4 (0%)	1	2	1	2	2003-11-18 13:08:46	2003-11-18 13:15:37
□	misc-attack	6 (0%)	1	1	1	1	2003-11-18 13:08:48	2003-11-18 13:08:49
□	unknown	2 (0%)	1	1	1	1	2003-11-18 13:12:14	2003-11-18 13:12:16
□	string-detect	1 (0%)	1	1	1	1	2003-11-18 13:12:15	2003-11-18 13:12:15

Action



File Query Reports Sound: Off ServerName: localhost UserName: squil UserID: 2

RealTime Events Escalated Events

ST	CNT	Sensor	sid.cid	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	1	localhost	1.7	2003-11-18 19:09:23	10.10.10.141	32912	172.20.11.2	69	17	TFTP root directory
RT	2	localhost	1.8	2003-11-18 19:09:23	10.10.10.141	32912	172.20.11.2	69	17	TFTP Get
RT	5	localhost	1.9	2003-11-18 19:09:23	10.10.10.141	32911	172.20.11.2	177	17	MISC xdmcp info query
RT	1	localhost	1.15	2003-11-18 19:09:27	10.10.10.141	35512	172.20.11.2	22	6	BLEEDING-EDGE Potential SSH Scan
RT	3	localhost	1.16	2003-11-18 19:09:27	10.10.10.141	35512	172.20.11.2	22	6	MISC source route lssr
RT	3	localhost	1.17	2003-11-18 19:09:27	10.10.10.141	35512	172.20.11.2	22	6	MISC source route lssre

ST	CNT	Sensor	sid.cid	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	2	localhost	1.1	2003-11-18 19:09:22	10.10.10.141	63176	172.20.11.2	0	6	BAD-TRAFFIC tcp port 0 traffic
RT	2	localhost	1.2	2003-11-18 19:09:22	172.20.11.2	0	10.10.10.141	63176	6	BAD-TRAFFIC tcp port 0 traffic
RT	266	localhost	1.5	2003-11-18 19:09:23	10.10.10.234	137	149.134.52.149	137	17	snort_decoder: Short UDP packet, length field > payload length
RT	473	localhost	1.13	2003-11-18 19:09:26	172.20.201.2		10.10.10.165		1	ICMP Destination Unreachable Port Unreachable
RT	102	localhost	1.18	2003-11-18 19:09:27	192.168.17.2		10.10.10.165		1	ICMP Destination Unreachable Host Unreachable
RT	5	localhost	1.23	2003-11-18 19:09:29	10.10.10.141	22	172.20.11.2	22	6	spp_stream4: NULL Stealth Scan

ST	CNT	Sensor	sid.cid	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	58	localhost	1.46	2003-11-18 19:09:40	10.10.10.224	45220	172.20.201.3	9991	6	spp_portscan: Portscan Detected
RT	2	localhost	1.394	2003-11-18 19:11:03	10.10.10.195	2345	172.20.11.80	4	6	spp_portscan: Portscan Detected
RT	3	localhost	1.586	2003-11-18 19:11:37	10.10.10.186	32813	172.20.201.198	5	6	spp_portscan: Portscan Detected
RT	20	localhost	1.877	2003-11-18 19:12:44	10.10.10.141	53750	172.20.11.0	13710	6	spp_portscan: Portscan Detected
RT	1	localhost	1.1114	2003-11-18 19:14:20	10.10.10.113	59194	192.168.17.135	1486	6	spp_portscan: Portscan Detected
RT	2	localhost	1.13600	2003-11-18 19:19:00	10.10.10.228	32781	172.20.201.1	879	6	spp_portscan: Portscan Detected

Src IP:
Src Name:
Dst IP:
Dst Name:

□ Reverse DNS Whois Query: ◆ None ◇ Src IP ◇ Dst IP

System Messages User Messages

☒ Show Packet Data ☒ Show Rule www.snort.org icat.nist.gov

```
alert udp $EXTERNAL_NET any -> $HOME_NET 69 (msg:"TFTP root directory"; content:"|00 01|/"; depth:3; reference
```

IP	Source IP	Dest IP	Ver	HL	TOS	len	ID	Flags	Offset	TTL	ChkSum
	10.10.10.141	172.20.11.2	4	5	0	50	49260	2	0	64	0

UDP	Port	Port	Length	ChkSum
	32912	69	30	20289

DATA

```
00 01 2F 65 74 63 2F 70 61 73 73 77 64 00 6F 63     ../etc/passwd.oc
74 65 74 00 00 00                                   tet...
```

Search Packet Payload ◇ Hex ◇ Text □ NoCase

Results of the ***strings*** (and some additional processing) command **- interesting-strings-2003-12-15.txt** file

```
"Lt028#
%.f%.f%.f%
%.f%.f%.f%.f%.f%.f%.f%.f%
(Amanda 2.3 REQ HANDLE 000-65637373 SEQ 954568800
-2-2000-
-2-2000-2000000000000000000000000
-2-2200-207000000000
-g gateway              source-routing hop point[s],\n
-rw-r--r--  1 root  r
-rw-r--r--  1 root  rnl
../etc/passwd
../nessus
.conf
.rhosts: No such file or direc
.which: no nmap in (/usr/local/8p
/bin/login
/bin/ls: A*: No such file or directory
/bin/sh: /home: is a directory
/bin/sh: /sbin/login: No such file or dire
/bin/sh: adduser: command not
/bin/sh: cd: log: Not a direct
/bin/sh: cd: tcsh: Not a direc
/bin/sh: line 15: 17306 Segmen
/bin/uname -a
/etc
/etc/passwd
/sbin/login
/tmp/iss.routedappend
/usr/bin/id
/usr/bin/nc -e
/usr/bin/nc -h
/usr/bin/nc: option requires an argument -gn
10-10-10-111    attackers
10-10-10-117    attackers
10-10-10-122    attackers
10-10-10-160    attackers
10.10.10.196
113 , 1802 : USERID : OTHER :nobody
113 , 1855 : USERID : OTHER :nobody
150 Opening ASCII mode data co
150 Opening ASCII mode data connection for
150 Opening BINARY mode data ccl
150 Opening BINARY mode data connection fo
```

```
168.17.1
168.17.66
168.17.67
168.17.68
168.22.207
172.20.11.1: Connection refuse
172.20.201.1
172.20.201.135
172.20.201.198
172.20.201.2
192.168.17.1
192.168.17.66
192.168.17.67
192.168.17.68
192.168.22.207
20.201.1
20.201.135
20.201.198
20.201.2
200  (end of '%020d|%.f%.f|')
200  (end of '%p')
200  (end of '7
200  (end of '7
200  (end of '7 AAAAPsPsBAAAPs
200  (end of '7 AAAAPsPsBAAAPsyo
200  (end of '7 mmmmnnnn%.f%.f
200  (end of '7 mmmmnnnn%.f%.f%.f%.f%.f%.f
200  (end of '7 v
200  (end of 'sh -c /bin/id')
200 PORT command successful.
200 Type set to A.
200 Type set to I.
200-00000000000000000049|0-2|
200-7
200-7 AAAAPsPsBAAAPsPsCAAAPsPs
200-7 mmmmnnnn-2-2000-20000000000000000000
200-7 mmmmnnnn-2-2200-20700000
200-sh -c /bin/id
200-sh -c /usr/bin/id
201.1
201.135
201.198
201.2
213-status of A*:
214-2.0.0     Specifies the recip
214-2.0.0     Specifies the sende
```

```
214-2.0.0 MAIL FROM: <sender>
214-2.0.0 RCPT TO: <recipient>
214-The following SITE command
214-The following commands are
215 UNIX Type: L8
22.207
220 172-20-201-135.MSY-POP.ISP
220 172-20-201-135.MSY-POP.ISP.NET ESMTP
220 172-20-201-135.MSY-POP.ISP.NET FTP ser
220 lazy ESMTP Sendmail 8.11.0
220 lazy ESMTP Sendmail 8.11.0/8.11.0; Tue
220 lazy FTP server (Version w
220 lazy FTP server (Version wu-2.6.0(1)
220 mail.isp.net ESMTP
220 suse72all.target.labs.veri
220 suse72all.target.labs.veritect FTP ser
221 172-20-201-135.MSY-POP.ISP.NET closing
221 2.0.0 lazy closing connection
221 2.0.0 suse72all.target.lab
221 Goodbye.
221 You could at least say goo
221 You could at least say goodbye.
221-Thank you for using the FT
221-Total traffic for this ses
221-Total traffic for this session was 312
221-Total traffic for this session was 338
221-You have transferred 0 byt
221-You have transferred 0 bytes in 0 file
226 Transfer complete.
227 Entering Passive Mode (172
227 Entering Passive Mode (172,20,201,135,
227 Entering Passive Mode (172,20,201,198,
227 Entering Passive Mode (192
228-12-20-172
230 Guest login ok, access res
230 Guest login ok, access restrictions ap
250 172-20-201-135.MSY-POP.ISP.NET Hello i
250 2.0.0 Reset state
250 2.1.0 <user@sttwks01>... Sender ok
250 2.1.5 root <root@lazy>
250 <user@sttwks01>... Sender ok
250 CWD command successful.
250 Reset state
250 lazy Hello issCrootMprogP/bin/sh
250 root <root@172-20-201-135.MSY-POP.ISP.
250 suse72all.target.labs.veri
```

```
252 2.1.5 <17487703@ISS>
252 <17487703@ISS>
257 "/" is current directory.
331 Guest login ok, send your
331 Guest login ok, send your complete e-m
331 Guest login ok, type your
331 Guest login ok, type your name as pass
331 Password required for
331 Password required for bogu
331 Password required for gues
331 Password required for ness
331 Password required for none.
350 File exists, ready for destination nam
421 Timeout (900 seconds): closing control(s
44-12-20-172
451 4.1.8 jamesbond@attackers.
451 4.1.8 nobody@example.com..
451 4.1.8 root@attackers.org..
451 4.1.8 root@company.com...
451 4.1.8 root@hackers.org...
500 'GET nessus1062325010': co
500 'LIST a a a a a a a a a a a a a a a a
500 'SITE CHMOD': command not
500 'SITE INDEX': command not
500 'SITE LIST ../../../..': c
500 'SITE MINFO': command not
500 5.5.1 Command unrecognized
500 5.5.1 Command unrecognized"o
500 5.5.1 Command unrecognized: "DEBUG"
500 5.5.1 Command unrecognized: "WIZ"
500 Command unrecognized: "DEBUG"
500 Command unrecognized: "WIZ"
500 Illegal PORT Command
500 Illegal PORT rejected (add
500 Illegal PORT rejected (res
500 Nothing transferred yet
501 5.0.0 HELO requires domain address
501 5.5.2 Syntax error in para
501 HELO requires domain address
503 5.0.0 Need MAIL before RCP
530 Login incorrect.
530 Please login with USER and
530 Please login with USER and PASS.
550 %20..: No such file or dir
550 ../../../../../../nonexist
550 /incoming: No such file or
```

```
550 5.1.1 uudecode... User unknown
550 5.7.1 user%host@sttwks01... Relaying d
550 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
550 Nessus_test: Permission de
550 No files found.
550 iss.test: Permission denied on server.
550 nessus833740542: No such f
550 passwd: Not a directory.
550 pu: No such file or directory.
550 shadow: No such file or dill
550 user%host@sttwks01... Relaying denied
550 uudecode... User unknown
550 ~/A*: No such file or directory.
553 .nessus_test_2: Permission
553 5.1.3 :... List:; syntax illegal for r
553 5.1.7 |tail|sh... Invalid sender addre
553 :... List:; syntax illegal for recipie
553 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
553 Permission denied on serve
553 iss.test: Permission denied on server.
553 nessus_test: Permission de
553 |tail|sh... Invalid sender address
6:45   0:00 sendmail: accepti
: USERID : UNIX : i
;alskdjf;lkasjdfl;kasdjf
<?xml version = "1.0"?>
>/bin/sh: line 13: 17305 Segmen
?/bin/sh: useradd: command not
ABACF
ABACFPFPENFDE
ABACFPFPENFDECFCEPFHFDEFFPFPACAB
ANP
ANYCOM
ATHENA.MIT.EDU
Accept:
Accept: image/gihp
Accept: n
Access violation
Active Internet connections (s
Amanda 2.3 REQ HANDLE 000-65637373 SEQ 954568800
BIND
BISSPNGRQ; ISS Scanner v6.21.2001.320 Release key# "m
BIyes, it does.: No such file or
Blah blah blah..
Blah blah blah...
CISSPNGRQ; ISS Scanner v6.21.2001.320 Release key# 7m
```

```
        CKAAAAAAAAAAA
        CKAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        COCACACACACACACACACACACACACACAAA
        CORBA
        CPT TO: root@suseserver.compa
        CWD %20..
        CWD ..
        CWD ../../../../etc
        CWD /
        CWD /incoming
        CWD bin
        CWD dev
        CWD etc
        CWD lib
        CWD passwd
        CWD pu
        CWD pub
        CWD ~
        CWD ~nonexistinguser
        CWD ~root
        Cisco Inter
        Cisco router
        Ciss.n
        Ciss.net Root
        DDDDDDDDDDDDDDDDDDDDDDDDDD
        DDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD
DDDD
        DEBUG
        DELE AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
        DIR_COLORS
        DISSPNGRQ; ISS Scanner v6.21.2001.320 Release key# ?m
        DUSER anonymous
        Date:
        Date: Mon, 17 Nov 2003 04
        Desktop
        Desktop:        directory
        Direc
        Direct
        EBEEENEJEOEJFDFEFCEBFEEPFCCACAAD
        ECEFEIENDADBD
        ECEFEIENDADBDDDICACACACACACACA
        ERR Login failed.
        EXPN <root>
        EXPN decode
        EXPN uudecode
        FastEthernet0/14
        FastEthernet0/15
        FastEthernet0/16
        FastEthernet0/18
        FastEthernet0/20
        FastEthernet0/21
        FastEthernet0/22
        FastEthernet0/24
        FirstBogus
        GET / HTTP/1.0
        GET /cgi-bin/ HTTP/1.1
        GET /cgi-bin/. HTTP/1.1
        GET /etc/passwd HTTP/1.1
        GET /images/top-logo.jpg HTTP/1.1
        GET /index.php HTTP/1.1
        GET /index.php4 HTTP/1.1
        GET /users.html HTTP/1.1
        GET /users/jsmith/index.html HTTP/1.1
        GET nessus1062325010
        HEAD / HTTP/1.0
        HELO attackers.org
        HELO hackers.org
        HELO sttwks01
        HELP
        HELP MAIL
        HELP RCPT
        HTTP/1.1 200 OK
        HTTP/1.1 403 Forbidden
        HTTP/1.1 404 Not Found
        Harmless Nessus echo test
        Host '10-10-10-165.attackers.org' i
        Host '10-10-10-186.attalo
        Host:239.255.255.250:1900
        It
        KCDLC6107ZHKN11
KHDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDD
        LIST ../../../../../../../
        LIST /bin/
        LIST /etc/
        Linux 172-20-201-135.MSY-POP.I:q
        Linux lazy 2.2.16-22 #1 Tue Au
        Locate"
        Login     Name        Tty      Idle  Login
        Login incorrect
        Login timed out after 60 seconds
        Login: bin
        Login: daemon
```

```
Login: dml
Login: jsmith
Login: root
Login: sync
MAIL FROM: <user@sttwks01>
MAIL FROM: jamesbond@attackers
MAIL FROM: root@attackers.org
MAIL FROM: root@company.com
MAIL FROM: root@hackers.org
MAIL FROM: |tail|sh
MKD Nessus_test
MKD iss.test
NESSUS.ORG
OK Hello there.
OK Password required.
OK [CAPABILITY IMAP4rev1 UIDPLUS CHILDRE2l
PASS
PASS %.2048d
PASS -iss@iss.iss.iss
PASS -wwwuser@
PASS NULL
PASS XXXXXXXXXXXXXXXXXXXXXXXXX
PASS blah
PASS blah@blahcom
PASS jsmith@company.com
PASS linuxNIS
PASS mozilla@
PASS nessus@
PASS nessus@172-20-201-198.MSY
PASS nessus@nessus.org
PASS sadfpoi@
PASS scanner@test.net
PASS sdpofi@sdpdofi
PASS soogjksjka
PASS xforce@iss.net
PASV
PORT 10,10,10,165,3,255
PORT 10,10,10,212,18,31
PORT 10,10,10,212,18,32
PORT 10,10,10,212,18,33
PORT 10,10,10,212,18,34
PORT 10,10,10,212,18,35
PORT 10,10,10,212,18,36
PORT 10,10,10,212,18,37
PORT 10,10,10,212,18,38
PORT 10,10,10,212,18,39
PORT 172,20,11,3,0,144
PORT 172,20,11,3,13,129
PORT 172,20,11,3,23,162
PORT 172,20,201,199,0,21
Password:
Permission denied.
Pro
Product
Protocol major versions differ
Protocol mismatch.
RCPT TO: user%host@sttwks01
RCPT root@suseserver.company.c
RETR ../../../../../../nonexis
RETR passwd
RETR shadow
RMD XXXXXXXXXXXXXXXXXXXXXXXXXX
RMD iss.test
ROOT-S
ROOT-SER
ROOT-SERVERS
Red Hat Linux release
Red Hat Linux release 6.2 (Zoot)
Red Hat Linux release 7.0
Red Hat Linux release 7.0 (Guinnes
Red Hat Linux release 7.0 (Guinness)
Root <root@lo
SEARCH * HTTP/1.1
SERVI
SITE EXEC %020d|%.f%.f|
SITE EXEC %p
SITE EXEC 7
SITE EXEC 7 AAAAPsPsBAAAPsPsCA
SITE EXEC 7 mmmmnnnn%.f%.f%.f%
SITE EXEC 7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%
SITE bogus command
SITE checksum
SITE chmod
SITE chmod 777 libnss_files-2.
SITE exec /bin/sh -c /bin/id
SITE exec /bin/sh -c /usr/bin/
SITE exec vulnerable/ftp
SITE help
SITE index
SITE minfo
SOURCES
SRPMS
```

```
SSH-1.33-NessusSSH_1.0
SSH-1.5-NessusSSH_1.0
SSH-1.99-NessusSSH_1.0
SSH-1.99-OpenSSH_2.1.1
SSH-1.99-OpenSSH_3.4p1
SSH-1.99-OpenSSH_3.5p1
SSH-2.0-4.0.6 (build 430) SecureCRT
SSH-2.0-NessusSSH_1.0
SSH-2.0-OpenSSH_3.1p1
SSH-2.0-OpenSSH_3.4p1
SSH-2.0-OpenSSH_3.5p1
SSH-2.0-OpenSSH_3.6.1p2
SSH-9.9-NessusSSH_1.0
STAT {A*,A**,A***}A*/../A*/../A*/../A*
STAT ~/A*/../A*/../A*/../A*
STOR .nessus_test_2
STOR nessus_test
SWIFT
Scan by ISS
Secret C0de
Switch
This /bin/ps is not secure for
UID        PID  PPID  C STIME TTY
UMASK          GROUP
UMASK   IDLE    CHMOD   HEL
USER       PID %CPU %MEM   VSZ
USER    PORT    STOR    MSA
USER NULL
USER XXXXXXXXXXXXXXXXXXXXXXXXXX
USER anonymous
USER bogusbogus
USER ftp
USER guest
USER nessus
USER none
User
VRFY :
VRFY <17487703@ISS>
VRFY <root>
VT100/9600
Vim: Warning: Input is not fro
Vim: Warning: Output is not to
Virtual PC
X11R6
XNLST /../*/../
XXXX
```

```
XXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
\players\rules\status\pack
\players\rules\status\packets\
_tcp$a6c668c5-07c4-4228-8d8f-52efc3ym
`/bin/id`
`/usr/bin/id`
a a a a a a a a a a a a a a a a a a
a a a a a a a a a a a a a a a a a a a a a
aMa0LL
aaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
abcdefghijklmnopqrstuvwabcdefghi
access
adduser
admin
administrator
agent
agent_steal
alan
alex
alive
all private
allen
arch
archie
ash.static
bbbbbbbbbbbbbbbbbbbbbbbbbb
bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
bill
billy
bruce
cable-docsis
campbell
carl
carlos
cascade
cat "yes, it does." >>importan
cat .rhosts
cat /etc/passwd
cat imp*
cat impor*
cat passwd
cat shadow
cd ../work*
cd /bin
cd /etc
cd /home
cd /home/jsmith
cd /root
cd /usr
cd SOURCES
cd SRPMS
cd doc
cd httpd
cd log
cd redhat
cd src
cd star*
cd tcsh
cgi-bin
che:x:48:48:Apache:/var/www:/b
ching
chris
christopher
chun
cisco
colecorp-main
com     attackers
comCq
comN
comcomcom
community
company
compress
connect to somewhere:   nc [-options\n
d--x--x--x   2 root
d--x--x--x   2 root vm
darren
david
davis
debbie
default
demo
dennis
derek
diffie-hellman-g
douglas
download
drw
drwxr-xr-x   17 root
drwxr-xr-x   6 root     root    }m
drwxr-xr-x   6 root     root    ~m
drwxr-xr-x  2 root
drwxr-xr-x  2 root  r
drwxr-xr-x  2 root  r   q
drwxr-xr-x  2 root  rUl
drwxr-xr-x  2 root  ro
drwxr-xr-x  9 root  r
echo "yes, it does." >>importa
echo ISS logged in;echo PASSWORD F
echo ISS logged in;echo PASSWORD FILE
echo ISS logged in;echo PASSWORD FILE;;n
echo ISS logged in;echo PASSWORD FILE;Pn
etC<
expert
ffffffffffffffffffffffffffffffffffffffffffffffffffffff
finger: /etc/p
finger: 0: no such
finger: 1: no such
finger: access: no
finger: administra
finger: alan: no s
finger: alex: no s
finger: allen: no
finger: andrew: no
finger: ann: no su
finger: archie: no
finger: bill: no s
finger: billy: no
finger: bob: no su
finger: brad: no s
finger: brown: no
finger: bruce: no
finger: campbell:
finger: carl: no s
finger: carlos: no
finger: ching: no
finger: chris: no
finger: christophe
finger: chun: no s
finger: cliff: no
finger: craig: no
finger: dale: no s
finger: dan: no su
finger: darren: no
finger: david: no
```

```
finger: davis: no
finger: debbie: no
finger: demo: no s
finger: dennis: no
finger: derek: no
finger: don: no su
finger: donald: no
finger: douglas: n
finger: earl: no s
finger: ellis: no
finger: eric: no s
finger: expert: no
finger: francis: n
finger: fred: no s
finger: gary: no s
finger: gene: no s
finger: grady: no
finger: greg: no s
finger: guest: no
finger: hank: no s
finger: ingres: no
finger: irc: no su
finger: jack: no s
finger: jackson: n
finger: jacobs: no
finger: james: no
finger: jason: no
finger: jay: no su
finger: jeff: no s
finger: jill: no s
finger: jim: no su
finger: john: no s
finger: jones: no
finger: joseph: no
finger: julie: no
finger: kathy: no
finger: keith: no
finger: kent: no s
finger: kevin: no
finger: kim: no su
finger: kramer: no
finger: laura: no
finger: lee: no su
finger: lisa: no s
finger: lynn: no s
finger: mark: no s
```

```
finger: marshall:
finger: mary: no s
finger: matthew: n
finger: meyer: no
finger: michael: n
finger: mike: no s
finger: morris: no
finger: nick: no s
finger: norman: no
finger: oracle: no
finger: pam: no su
finger: pat: no su
finger: patrick: n
finger: paul: no s
finger: pete: no s
finger: phillip: n
finger: raymond: n
finger: rick: no s
finger: rita: no s
finger: rje: no su
finger: robert: no
finger: roger: no
finger: ron: no su
finger: ronald: no
finger: sam: no su
finger: scott: no
finger: sharon: no
finger: steve: no
finger: steven: no
finger: sue: no su
finger: susan: no
finger: system: no
finger: tami: no s
finger: terry: no
finger: tim: no su
finger: tom: no su
finger: tommy: no
finger: tony: no s
finger: vince: no
finger: walt: no s
finger: wayne: no
finger: welcome: n
finger: william: n
fingerd: Internal error
fingerd: forwarding not allowed
freekevin
```

```
froot
gesundheit!
getchallenge
help
hp_admin
html
httpd
important-proposal.tx
in-addr
include
inger: 0: no such user.
inger: 1: no such user.
inger: : no such user.
inger: ;cat: no such user.
inger: `/bin/id`: no such user$o
inger: `/usr/bin/id`: no such $o
inger: access: no such user.
inger: administrator: no such user.
inger: demo: no such user.
inger: expert: no such user.
inger: guest: no such user.
inger: ingres: no such user.
inger: irc: no such user.
inger: oracle: no such user.
inger: rje: no such user.
inger: system: no such user.
inger: welcome: no such user.
inger: |/bin/id: no such user.
jsmith    Joe Smith   pts/13
kerberos
kers.org)
latitude-d600
local
localdomain
localhost
log
login:
lost
lost+found
lpd: : Malformed from address
more shadow
ness
nessus
nessus A=B A=B A=B A=B A=B A=B
netascii
netstat -an
```

```
openview
opera
pHoiFQ6QNug7
packetstormsecurity
pass ncc1701
password
pine.con
pine.conf
players\rules\status\packets\
png l44adsl
png l44adsl
ps -aux
quit
redhat
rlogin 172.20.11.1
rlogind: Permission denied.
rmon
rmon_admin
root
root      4377   523  0 11
root      4711  0.0  0.8  314
root:$1$v1xcDeCt$o2UrR6PiM7qbQ
root:x:0:0:Super User:/root:/b q
root:x:0:0:root:/root:/bin/bas
secret
security
site help
site list
site list ../../../..
sounds imp
ssh-dss
ssh-rsa
ssh_config
ssh_host_dsa_k
ssh_host_dsa_key
startrek-stuff
strings - results:
sttwks01
su - jsmith
suseserver
tftp
tftp 10.10.10.196
total 104
total 1164
total 16
total 212
total 24
total 28
total 36
total 4
total 74
total 8
uid=0(root) gid=0(root) egid=5
uid=0(root) gid=0(root) egid=50(ftp) group
uid=0(root) gid=0(root) groups
uid=500(jsmith) gid=100(users)]p
uname -a
user jsmith
useradd
vi important-proposal.txt
weatherbug
whatever
which nmap
windows
windowsupdate
wisapidata
wu-ftpd-2.6.0
www     microsoft
www2
xfs:!!:11780:0:99999:7:::
ypserv.
ypserv.lp
{{{{{{{{{{{{{{{{{{{{{{{{{{
{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{{
```

Sample links, resulted from searching the Internet for strings found in the trace file:

Amanda: http://www.securityspace.com/smysecure/catid.html?viewsrc=1&id=10462

-g gateway   source-routing hop point[s],\n is indicative of Netcat: http://seclists.org/lists/fulldisclosure/2004/Mar/0300.html

/tmp/iss.routedappend: http://download.iss.net/manuals/unix_scanner53user.pdf

USERID : OTHER :nobody: information obtained via **successfully** telnet-ing to port 113 (identd) – as shown here: http://www.lsdp.net/~lotfree/doc/intrusions/testsintrusion.htm

200 (end of '%020d|%.f%.f|') – lots of information here: http://project.honeynet.org/scans/scan19/

sh -c /usr/bin/id and *sh -c /bin/id*
http://www.securityspace.com/smysecure/catid.html?viewsrc=1&id=10090

7 mmmmnnnn%.f%.f:
http://www.honeynet.org/scans/scan19/scan/som14/t/sol.html

iss.test: Permission denied on server.
http://www.secinf.net/misc/Maximum_Security/Maximum_Security__Chapter_9__Scanners_.html

uid=0(root) gid=0(root) egid=50(ftp) group
http://www.informit.com/articles/article.asp?p=350390&seqNum=6
(**NOTE:** what a nice "closure" to our discussions, so far – ***sguil*** choice inspired by an article which also discussed an exploit very likely to have affected my client, also, based on the strings identified in the traffic)

Sample results of having run various strings through *ngrep*:

```
$ sudo ngrep -I 2003.12.15.cap -q -i '%\.f%\.f%\.f%'
```

```
T 10.10.10.186:32802 -> 172.20.201.198:21 [AP]
  SITE EXEC 7 .....%.f%.f%.f%.f%

T 172.20.201.198:21 -> 10.10.10.186:32802 [AP]
  200  (end of '7 ....%.f%.f%.f%
```

➔ obvious communication in both directions, over TCP, with a **200** response from the server, implying "**success**". Further investigation will require a full analysis with other tools (***tcpdump, tethereal, snort*** in sniffer mode, ***tcpflow,*** etc.), and – where applicable - a BPF filter of the form: '(host 10.10.10.186 and 172.20.201.198) and (tcp port 21 and 32802)', then analyze the results, etc …

```
$ sudo ngrep -I 2003.12.15.cap -q -i '(Amanda 2.3 REQ HANDLE 000-65637373 SEQ 954568800'
```

```
U 10.10.10.141:32820 -> 172.20.11.2:10080
  Amanda 2.3 REQ HANDLE 000-65637373 SEQ 954568800.SERVI

U 10.10.10.141:32820 -> 172.20.11.2:10080
  Amanda 2.3 REQ HANDLE 000-65637373 SEQ 954568800.SERVI

U 10.10.10.141:32820 -> 172.20.11.2:10080
  Amanda 2.3 REQ HANDLE 000-65637373 SEQ 954568800.SERVI
```

➔ as opposed to previous example – unsuccessful client attempt over UDP (no response from the server) ➔ no need to follow-up on this, with my analysis

---

```
$ sudo ngrep -I 2003.12.15.cap -q -i 'source-routing hop point\[s\]'
```

```
T 172.20.201.198:21 -> 10.10.10.165:1062 [AP]
  ..-g gateway..source-routing hop point[s],
```

➔ the above leads me to try to see what the rest of the communication looked like, either continuing with ***ngrep***, by adding the "–B x" ("show the "x" line "B"efore the match), or just running ***tcpdump*** (or any equivalent tool) with the '(host 172.20.201.198 and 10.10.10.165)' BPF filter, etc …

```
$ sudo ngrep -I 2003.12.15.cap -q -i 'root'
```

```
U 10.10.10.2:53 -> 10.10.10.214:32768
  .&..........it.company.com................ .ns...root

T 10.10.10.226:34244 -> 192.168.17.135:25 [AP]
  RCPT root@suseserver.company.c....

U 10.10.10.2:53 -> 10.10.10.174:1026
  .1..........it.company.com................ .ns...root

T 10.10.10.226:34244 -> 192.168.17.135:25 [AP]
  MAIL TO root@suseserver.compan......n

U 10.10.10.2:53 -> 10.10.10.164:1910
  ............Cmh-dc.caa.local............{./.A.ROOT-S

T 10.10.10.226:34244 -> 192.168.17.135:25 [AP]
  TCPT TO: root@suseserver.compa......no

T 192.168.17.135:48487 -> 10.10.10.122:59914 [AP]
  total 28.drwxr-xr-x  2 root  r

U 10.10.10.2:53 -> 10.10.10.234:1042
  n...........www.symantec.com............../.A.ROOT-S

T 192.168.17.135:48488 -> 10.10.10.122:59918 [AP]
  root:x:0:0:Super User:/root:/b......no
```

```
T 192.168.17.135:48490 -> 10.10.10.122:59922 [AP]
  total 8..-rw-r--r--  1 root  r

T 192.168.17.135:48498 -> 10.10.10.122:59928 [AP]
  total 8..-rw-r--r--  1 root  r

T 192.168.17.135:48499 -> 10.10.10.122:59930 [AP]
  total 28..drwxr-xr-x  2 root

T 192.168.17.135:48504 -> 10.10.10.122:59932 [AP]
… and so on …
```

➔ here above we see that utilization of a more general string (“root”) revealed communication between more than one pair of systems, requiring further “narrowing down” of the traffic analysis

---

```
$ sudo ngrep -I 2003.12.15.cap -q -i 'CWD'

T 172.20.201.198:22 -> 10.10.10.228:32770 [AP]
  0TO...fWD.....f..a.\p.:<._'..d......nobo

T 172.20.201.198:22 -> 10.10.10.174:1055 [AP]
  ....b/.cs{YH.....:.......[)Wd/......nobo

T 10.10.10.122:59909 -> 192.168.17.135:21 [AP]
  CWD ../../../../etc..

T 192.168.17.135:21 -> 10.10.10.122:59909 [AP]
  250 CWD command successful...

T 10.10.10.122:59909 -> 192.168.17.135:21 [AP]
  RETR passwd..

T 10.10.10.122:59909 -> 192.168.17.135:21 [AP]
  CWD ....

T 192.168.17.135:21 -> 10.10.10.122:59909 [AP]
  250 CWD command successful...

T 10.10.10.186:32789 -> 172.20.201.198:21 [AP]
  PWD..

T 10.10.10.122:59909 -> 192.168.17.135:21 [AP]
  CWD usr..
```

```
T 192.168.17.135:21 -> 10.10.10.122:59909 [AP]
  250 CWD command successful...
```

… and more ..

➔ which implies successful execution of commands (FTP), between some pairs of systems. These (as for all the other above, and for the rest of strings having revealed some communication) will become the object of further analysis, on my part.

… and so on, until I exhausted all the other strings …

Appendix E – critical detects analysis

Detect number #1 – attacker 10.10.10.186 → victim 172.20.201.198

The followings are results from various tools, having revealed the malicious nature of traffic between the attacker, and the victim (see paper for what each tool is being used for):

```
$ sudo ngrep -A 4 –x -I ../2003.12.15.cap -q -i 'cat passwd'
input: ../2003.12.15.cap

T 10.10.10.186:48253 -> 172.20.201.198:21 [AP]
  63 61 74 20 70 61 73 73    77 64 0a                   cat passwd.

T 10.10.10.142:39470 -> 172.20.201.198:22 [A]

T 10.10.10.232:49162 -> 172.20.201.198:22 [A]

T 172.20.201.198:21 -> 10.10.10.186:48253 [AP]
  72 6f 6f 74 3a 78 3a 30    3a 30 3a 72 6f 6f 74 3a    root:x:0:0:root:
  2f 72 6f 6f 74 3a 2f 62    69 6e 2f 62 61 73          /root:/bin/bas

T 10.10.10.186:48253 -> 172.20.201.198:21 [A]
```

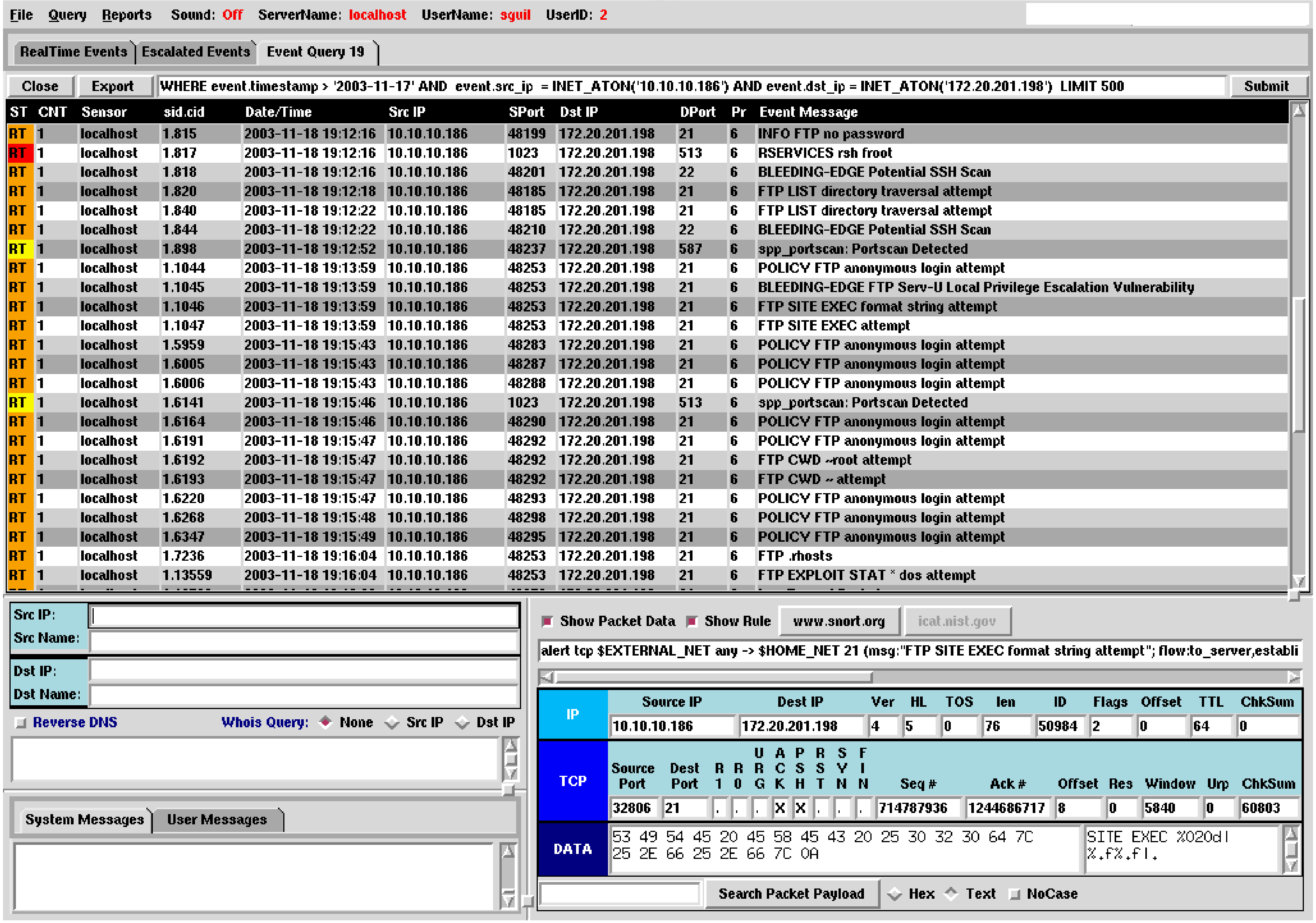
where: *-A 4* = show 4 lines after the match (which allowed me to reveal the results of various attempts)

-x = report hexadecimal and ASCII output

```
$ sudo ngrep -x -I ../2003.12.15.cap -q -i '^2..' host 10.10.10.186 and 172.20.201.198
input: ../2003.12.15.cap

T 172.20.201.198:21 -> 10.10.10.186:32789 [AP]
  32 31 34 2d 54 68 65 20    66 6f 6c 6c 6f 77 69 6e    214-The followin
  67 20 53 49 54 45 20 63    6f 6d 6d 61 6e 64          g SITE command

T 172.20.201.198:21 -> 10.10.10.186:32789 [AP]
  32 35 37 20 22 2f 22 20    69 73 20 63 75 72 72 65    257 "/" is curre
  6e 74 20 64 69 72 65 63    74 6f 72 79 2e 0d 00       nt directory...

T 172.20.201.198:21 -> 10.10.10.186:32789 [AP]
  32 35 30 20 43 57 44 20    63 6f 6d 6d 61 6e 64 20    250 CWD command
  73 75 63 63 65 73 73 66    75 6c 2e 0d 0a             successful...
```

```
T 172.20.201.198:21 -> 10.10.10.186:32789 [AP]
  32 32 37 20 45 6e 74 65    72 69 6e 67 20 50 61 73    227 Entering Pas
  73 69 76 65 20 4d 6f 64    65 20 28 31 37 32          sive Mode (172

T 172.20.201.198:21 -> 10.10.10.186:32789 [AP]
  32 32 36 20 54 72 61 6e    73 66 65 72 20 63 6f 6d    226 Transfer com
  70 6c 65 74 65 2e 0d 0a                               plete...

T 10.10.10.186:32789 -> 172.20.201.198:21 [AP]
  53 49 54 45 20 63 68 6d    6f 64 20 37 37 37 20 6c    SITE chmod 777 l
  69 62 6e 73 73 5f 66 69    6c 65 73 2d 32 2e 00 00    ibnss_files-2...
  09 04 00 00 6e 6f                                     ....no

T 172.20.201.198:21 -> 10.10.10.186:32789 [AP]
  32 32 31 2d 59 6f 75 20    68 61 76 65 20 74 72 61    221-You have tra
  6e 73 66 65 72 72 65 64    20 30 20 62 79 74          nsferred 0 byt

T 172.20.201.198:21 -> 10.10.10.186:32789 [AFP]
  32 32 31 2d 54 6f 74 61    6c 20 74 72 61 66 66 69    221-Total traffi
  63 20 66 6f 72 20 74 68    69 73 20 73 65 73          c for this ses

T 172.20.201.198:21 -> 10.10.10.186:32802 [AP]
  32 32 30 20 6c 61 7a 79    20 46 54 50 20 73 65 72    220 lazy FTP ser
  76 65 72 20 28 56 65 72    73 69 6f 6e 20 77          ver (Version w
………………………… and so on
```

Next pages depict sample alarms having been identified in the ***sguil*** and ***ACID*** interfaces, as result of queries I had to build, in order to reveal the items of interest. These are just samples, giving the reader an idea of the tools used and results obtained while having gone through the trace file, during my months-long analysis.



Sound: Off ServerName: localhost UserName: sguil UserID: 2

```
WHERE event.timestamp > '2003-11-17' AND  event.src_ip  = INET_ATON('10.10.10.186') AND event.dst_ip = INET_ATON('172.20.201.198')  LIMIT 500
```

ST	CNT	Sensor	sid.cid	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	1	localhost	1.815	2003-11-18 19:12:16	10.10.10.186	48199	172.20.201.198	21	6	INFO FTP no password
RT	1	localhost	1.817	2003-11-18 19:12:16	10.10.10.186	1023	172.20.201.198	513	6	RSERVICES rsh froot
RT	1	localhost	1.818	2003-11-18 19:12:16	10.10.10.186	48201	172.20.201.198	22	6	BLEEDING-EDGE Potential SSH Scan
RT	1	localhost	1.820	2003-11-18 19:12:18	10.10.10.186	48185	172.20.201.198	21	6	FTP LIST directory traversal attempt
RT	1	localhost	1.840	2003-11-18 19:12:22	10.10.10.186	48185	172.20.201.198	21	6	FTP LIST directory traversal attempt
RT	1	localhost	1.844	2003-11-18 19:12:22	10.10.10.186	48210	172.20.201.198	22	6	BLEEDING-EDGE Potential SSH Scan
RT	1	localhost	1.898	2003-11-18 19:12:52	10.10.10.186	48237	172.20.201.198	587	6	spp_portscan: Portscan Detected
RT	1	localhost	1.1044	2003-11-18 19:13:59	10.10.10.186	48253	172.20.201.198	21	6	POLICY FTP anonymous login attempt
RT	1	localhost	1.1045	2003-11-18 19:13:59	10.10.10.186	48253	172.20.201.198	21	6	BLEEDING-EDGE FTP Serv-U Local Privilege Escalation Vulnerability
RT	1	localhost	1.1046	2003-11-18 19:13:59	10.10.10.186	48253	172.20.201.198	21	6	FTP SITE EXEC format string attempt
RT	1	localhost	1.1047	2003-11-18 19:13:59	10.10.10.186	48253	172.20.201.198	21	6	FTP SITE EXEC attempt
RT	1	localhost	1.5959	2003-11-18 19:15:43	10.10.10.186	48283	172.20.201.198	21	6	POLICY FTP anonymous login attempt
RT	1	localhost	1.6005	2003-11-18 19:15:43	10.10.10.186	48287	172.20.201.198	21	6	POLICY FTP anonymous login attempt
RT	1	localhost	1.6006	2003-11-18 19:15:43	10.10.10.186	48288	172.20.201.198	21	6	POLICY FTP anonymous login attempt
RT	1	localhost	1.6141	2003-11-18 19:15:46	10.10.10.186	1023	172.20.201.198	513	6	spp_portscan: Portscan Detected
RT	1	localhost	1.6164	2003-11-18 19:15:46	10.10.10.186	48290	172.20.201.198	21	6	POLICY FTP anonymous login attempt
RT	1	localhost	1.6191	2003-11-18 19:15:47	10.10.10.186	48292	172.20.201.198	21	6	POLICY FTP anonymous login attempt
RT	1	localhost	1.6192	2003-11-18 19:15:47	10.10.10.186	48292	172.20.201.198	21	6	FTP CWD ~root attempt
RT	1	localhost	1.6193	2003-11-18 19:15:47	10.10.10.186	48292	172.20.201.198	21	6	FTP CWD ~ attempt
RT	1	localhost	1.6220	2003-11-18 19:15:47	10.10.10.186	48293	172.20.201.198	21	6	POLICY FTP anonymous login attempt
RT	1	localhost	1.6268	2003-11-18 19:15:48	10.10.10.186	48298	172.20.201.198	21	6	POLICY FTP anonymous login attempt
RT	1	localhost	1.6347	2003-11-18 19:15:49	10.10.10.186	48295	172.20.201.198	21	6	POLICY FTP anonymous login attempt
RT	1	localhost	1.7236	2003-11-18 19:16:04	10.10.10.186	48253	172.20.201.198	21	6	FTP .rhosts
RT	1	localhost	1.13559	2003-11-18 19:16:04	10.10.10.186	48253	172.20.201.198	21	6	FTP EXPLOIT STAT * dos attempt

Src IP:

Src Name:

Dst IP:

Dst Name:

□ Reverse DNS Whois Query: ● None ◇ Src IP ◇ Dst IP

☑ Show Packet Data ☑ Show Rule

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP SITE EXEC format string attempt"; flow:to_server,establi
```

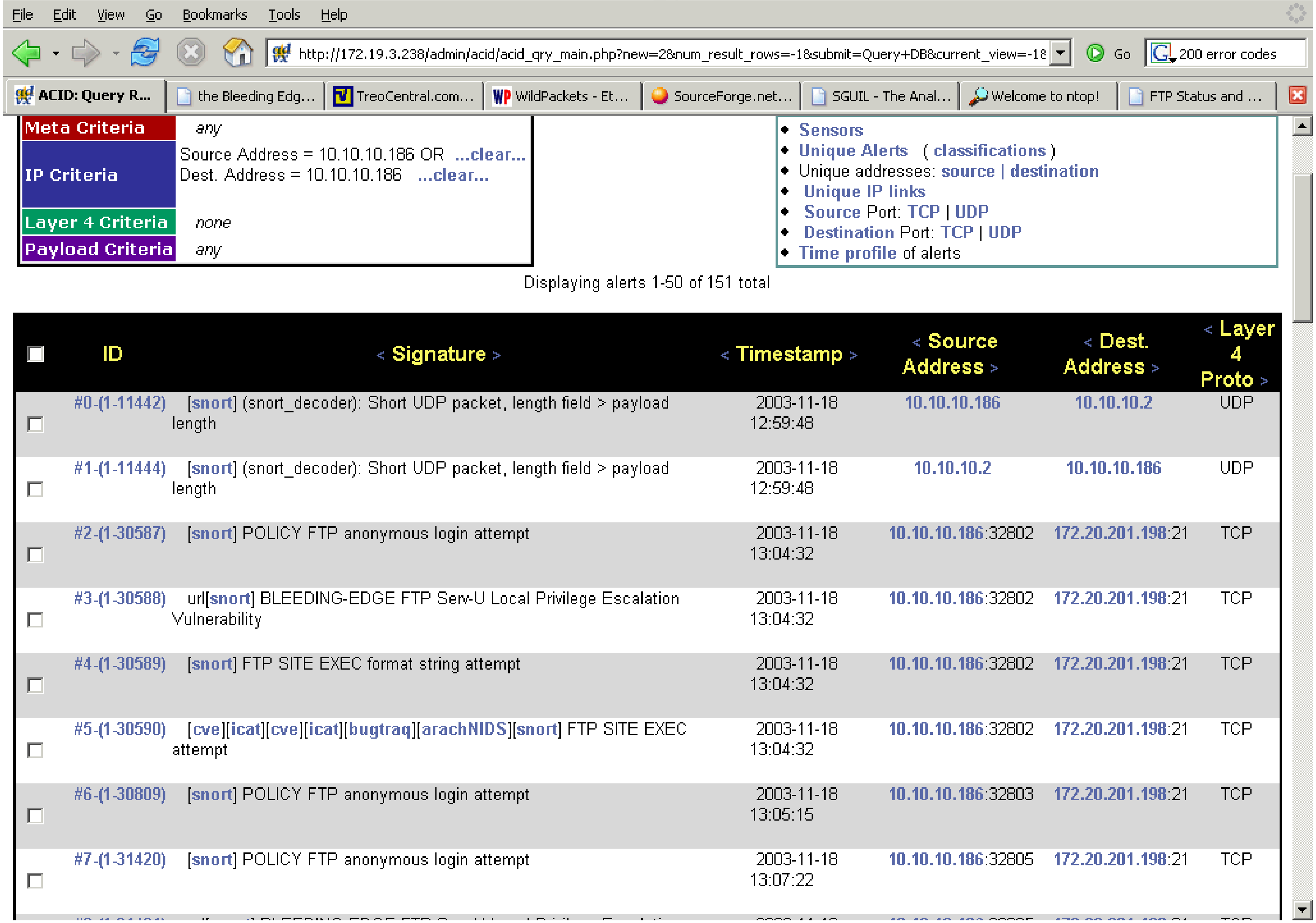
IP	Source IP	Dest IP	Ver	HL	TOS	len	ID	Flags	Offset	TTL	ChkSum
	10.10.10.186	172.20.201.198	4	5	0	76	50984	2	0	64	0

TCP	Source Port	Dest Port	R 1	R 0	URG	ACK	PSH	RST	SYN	FIN	Seq #	Ack #	Offset	Res	Window	Urp	ChkSum
	32806	21	.	.	.	X	X	.	.	.	714787936	1244686717	8	0	5840	0	60803

DATA

```
53 49 54 45 20 45 58 45 43 20 25 30 32 30 64 7C    SITE EXEC %020d|
25 2E 66 25 2E 66 7C 0A                            %.f%.f|.
```

◇ Hex ◇ Text □ NoCase



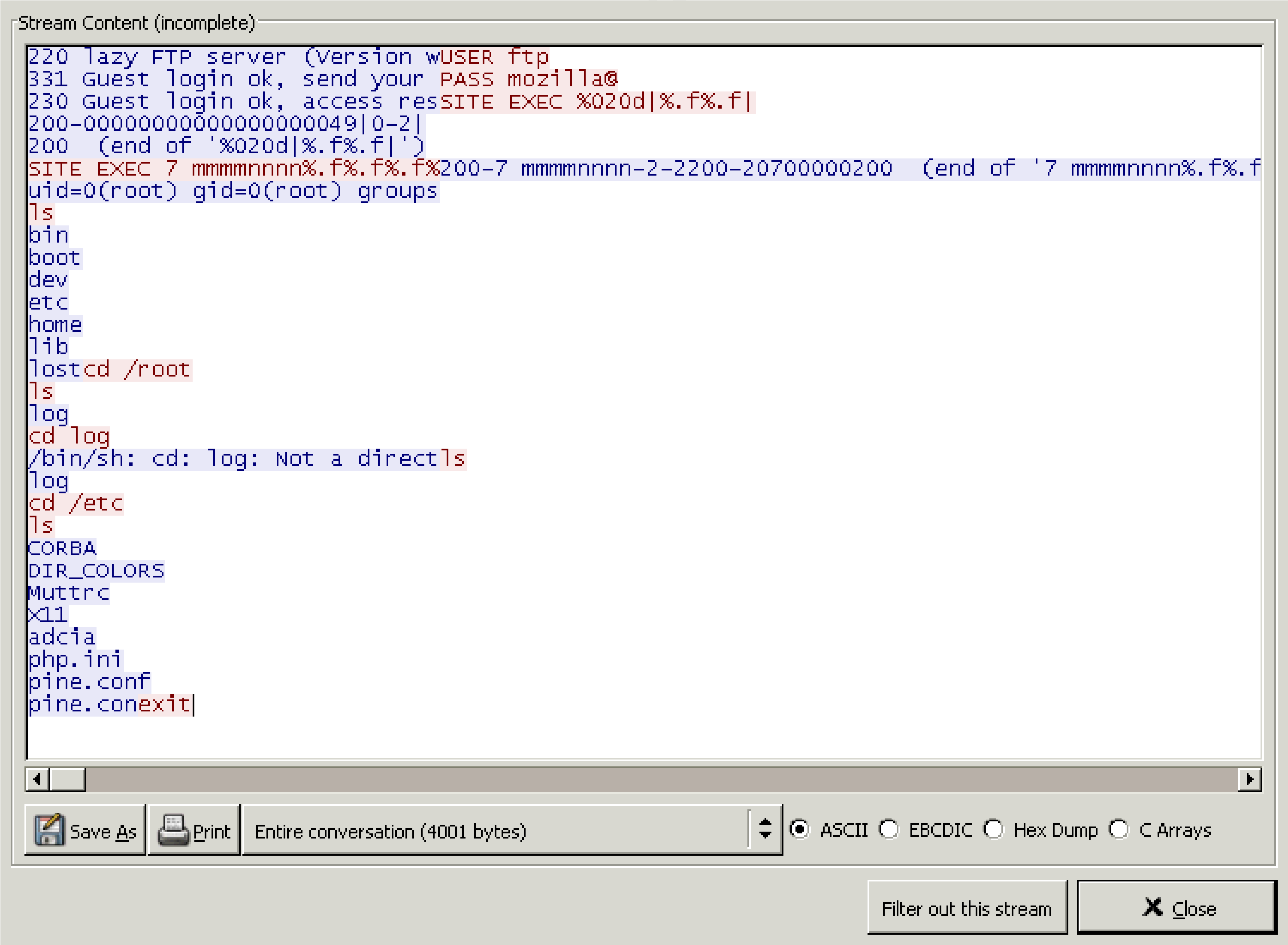
Meta Criteria	*any*
IP Criteria	Source Address = 10.10.10.186 OR ...clear... Dest. Address = 10.10.10.186 ...clear...
Layer 4 Criteria	*none*
Payload Criteria	*any*

- Sensors
- Unique Alerts (classifications)
- Unique addresses: source | destination
- Unique IP links
- Source Port: TCP | UDP
- Destination Port: TCP | UDP
- Time profile of alerts

Displaying alerts 1-50 of 151 total

☐	ID	< Signature >	< Timestamp >	< Source Address >	< Dest. Address >	< Layer 4 Proto >
☐	#0-(1-11442)	[snort] (snort_decoder): Short UDP packet, length field > payload length	2003-11-18 12:59:48	10.10.10.186	10.10.10.2	UDP
☐	#1-(1-11444)	[snort] (snort_decoder): Short UDP packet, length field > payload length	2003-11-18 12:59:48	10.10.10.2	10.10.10.186	UDP
☐	#2-(1-30587)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:04:32	10.10.10.186:32802	172.20.201.198:21	TCP
☐	#3-(1-30588)	url[snort] BLEEDING-EDGE FTP Serv-U Local Privilege Escalation Vulnerability	2003-11-18 13:04:32	10.10.10.186:32802	172.20.201.198:21	TCP
☐	#4-(1-30589)	[snort] FTP SITE EXEC format string attempt	2003-11-18 13:04:32	10.10.10.186:32802	172.20.201.198:21	TCP
☐	#5-(1-30590)	[cve][icat][cve][icat][bugtraq][arachNIDS][snort] FTP SITE EXEC attempt	2003-11-18 13:04:32	10.10.10.186:32802	172.20.201.198:21	TCP
☐	#6-(1-30809)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:05:15	10.10.10.186:32803	172.20.201.198:21	TCP
☐	#7-(1-31420)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:07:22	10.10.10.186:32805	172.20.201.198:21	TCP

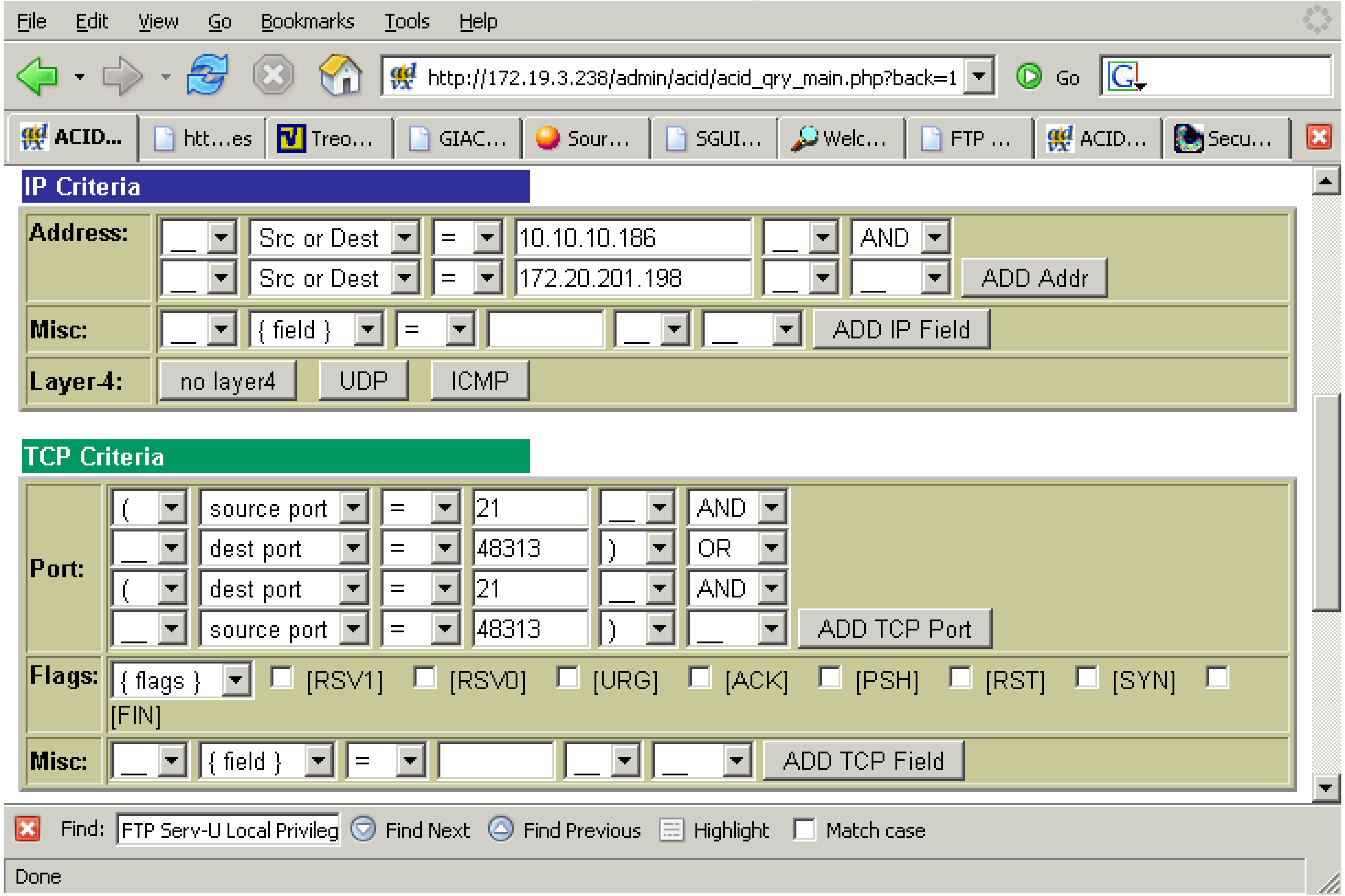
Ethereal "follow TCP stream" screenshot, from the last conversation captured in the trace file, between 10.10.10.186 and 172.20.201.198:



```
220 lazy FTP server (version wUSER ftp
331 Guest login ok, send your PASS mozilla@
230 Guest login ok, access resSITE EXEC %020d|%.f%.f|
200-00000000000000000049|0-2|
200  (end of '%020d|%.f%.f|')
SITE EXEC 7 mmmmnnnn%.f%.f%.f%200-7 mmmmnnnn-2-2200-20700000200  (end of '7 mmmmnnnn%.f%.f
uid=0(root) gid=0(root) groups
ls
bin
boot
dev
etc
home
lib
lostcd /root
ls
log
cd log
/bin/sh: cd: log: Not a directls
log
cd /etc
ls
CORBA
DIR_COLORS
Muttrc
X11
adcia
php.ini
pine.conf
pine.conexit
```

Save As
Print
Entire conversation (4001 bytes)
● ASCII ○ EBCDIC ○ Hex Dump ○ C Arrays
Filter out this stream
Close

Query constructed in ***ACID***, meant to isolate/reveal the rules triggered during the above mentioned communication:



http://172.19.3.238/admin/acid/acid_qry_main.php?back=1

IP Criteria

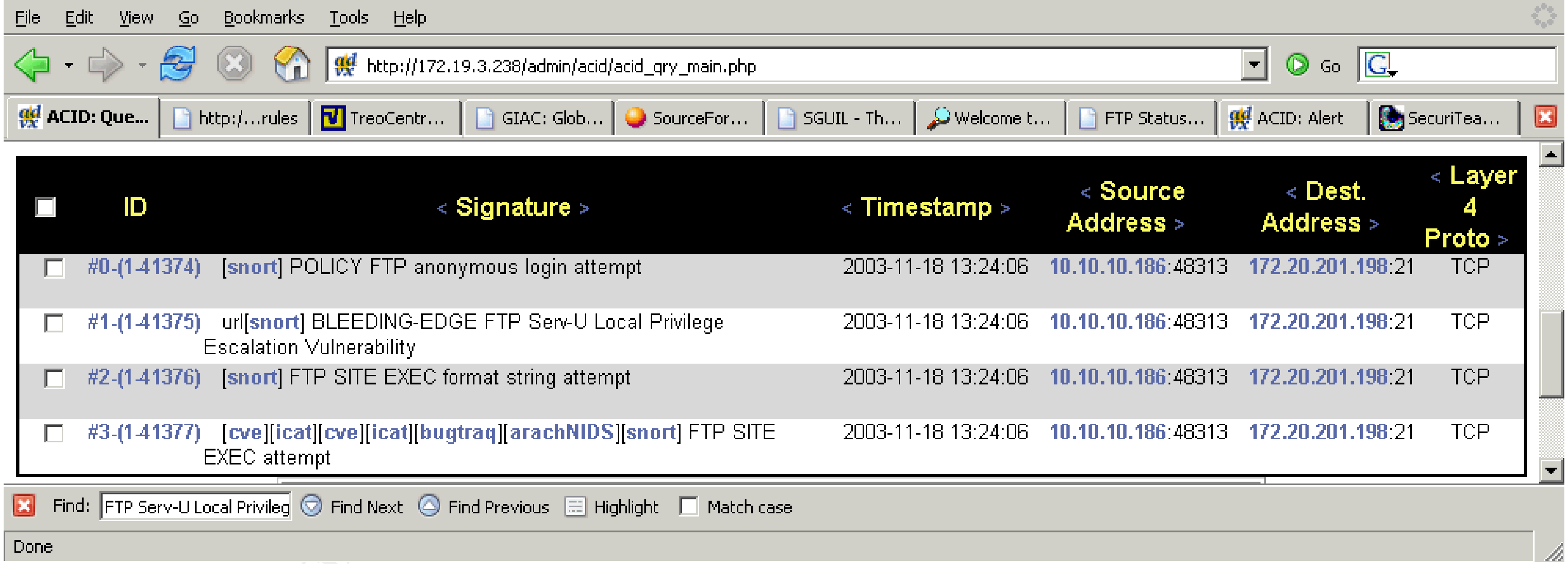
Address:	__	Src or Dest	=	10.10.10.186	__	AND	
	__	Src or Dest	=	172.20.201.198	__	__	ADD Addr
Misc:	__	{ field }	=		__	__	ADD IP Field
Layer-4:	no layer4	UDP	ICMP				

TCP Criteria

Port:	(	source port	=	21	__	AND	
	__	dest port	=	48313	)	OR	
	(	dest port	=	21	__	AND	
	__	source port	=	48313	)	__	ADD TCP Port
Flags:	{ flags }	□ [RSV1]	□ [RSV0]	□ [URG]	□ [ACK]	□ [PSH]	□ [RST] □ [SYN] □ [FIN]
Misc:	__	{ field }	=		__	__	ADD TCP Field

Find: FTP Serv-U Local Privileg

Which – in turn – reveals the following four rules:

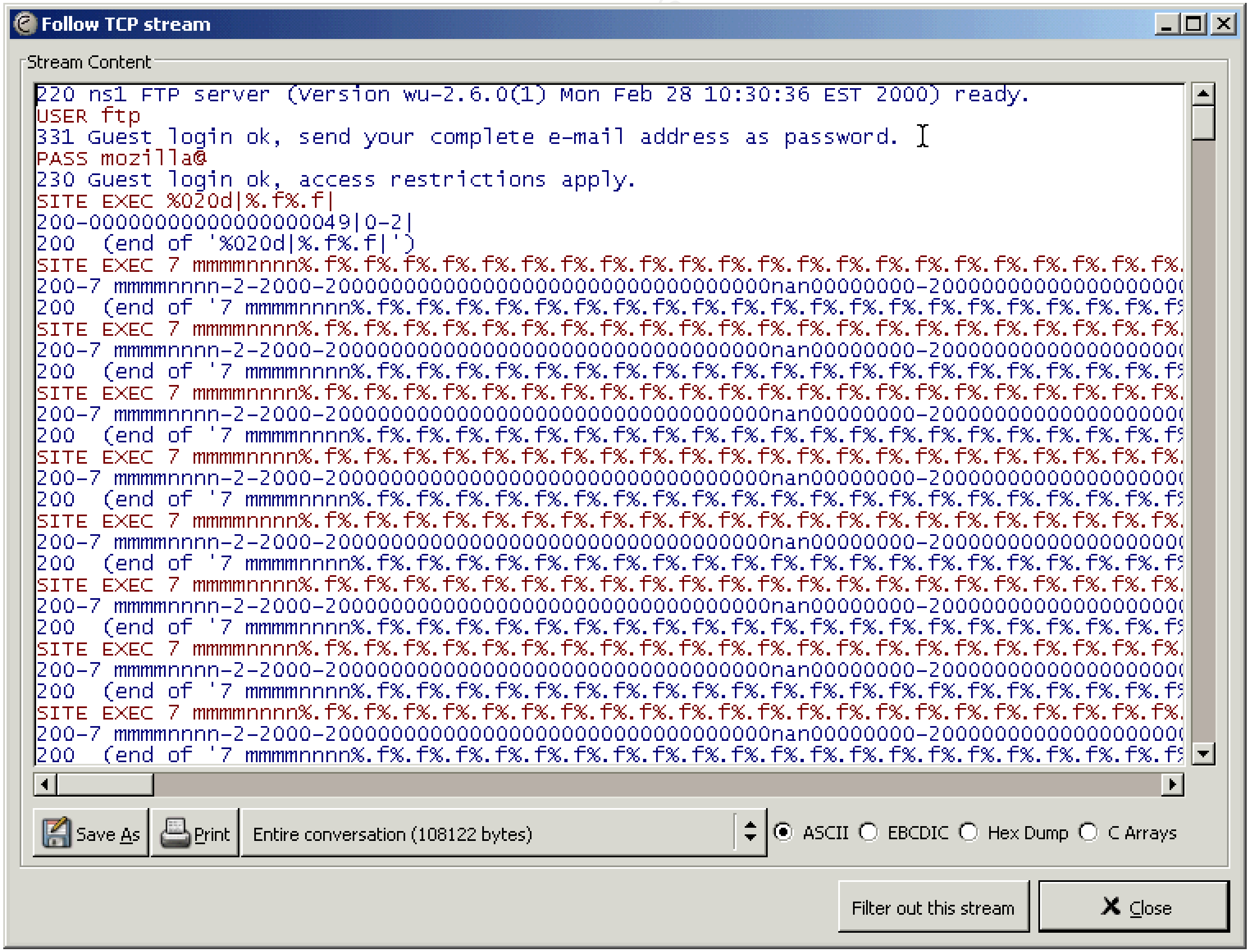


http://172.19.3.238/admin/acid/acid_qry_main.php

□	ID	< Signature >	< Timestamp >	< Source Address >	< Dest. Address >	< Layer 4 Proto >
□	#0-(1-41374)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:24:06	10.10.10.186:48313	172.20.201.198:21	TCP
□	#1-(1-41375)	url[snort] BLEEDING-EDGE FTP Serv-U Local Privilege Escalation Vulnerability	2003-11-18 13:24:06	10.10.10.186:48313	172.20.201.198:21	TCP
□	#2-(1-41376)	[snort] FTP SITE EXEC format string attempt	2003-11-18 13:24:06	10.10.10.186:48313	172.20.201.198:21	TCP
□	#3-(1-41377)	[cve][icat][cve][icat][bugtraq][arachNIDS][snort] FTP SITE EXEC attempt	2003-11-18 13:24:06	10.10.10.186:48313	172.20.201.198:21	TCP

Find: FTP Serv-U Local Privileg □ Match case

TCP flow capture, of a wu-ftpd attack, from the “Scan of the Month #19” - http://project.honeynet.org/scans/scan19/

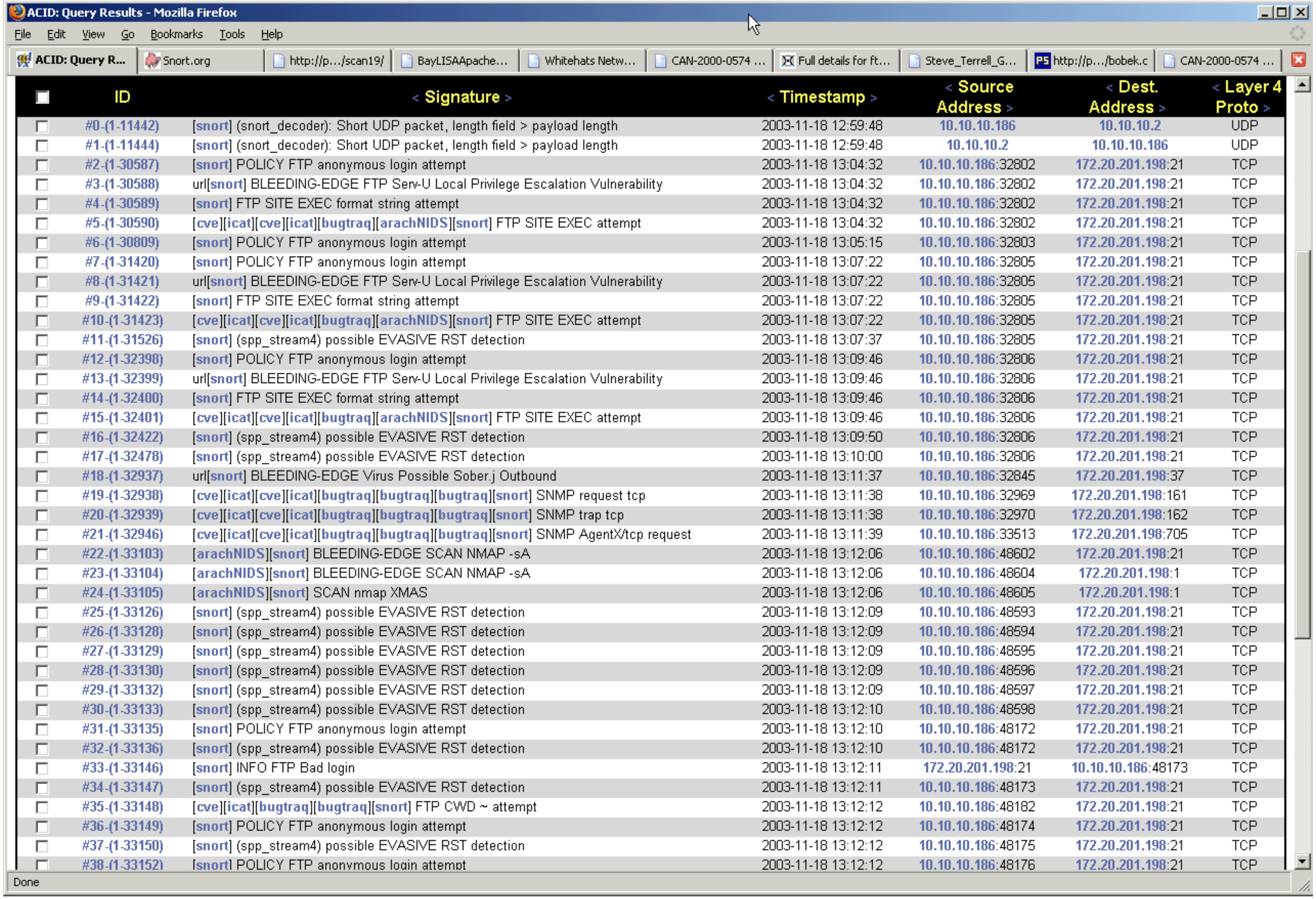


```
220 nsl FTP server (Version wu-2.6.0(1) Mon Feb 28 10:30:36 EST 2000) ready.
USER ftp
331 Guest login ok, send your complete e-mail address as password.
PASS mozilla@
230 Guest login ok, access restrictions apply.
SITE EXEC %020d|%.f%.f|
200-00000000000000000049|0-2|
200  (end of '%020d|%.f%.f|')
SITE EXEC 7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.
200-7 mmmmnnnn-2-2000-2000000000000000000000000000000000nan00000000-20000000000000000000
200  (end of '7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%
SITE EXEC 7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.
200-7 mmmmnnnn-2-2000-2000000000000000000000000000000000nan00000000-20000000000000000000
200  (end of '7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%
SITE EXEC 7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.
200-7 mmmmnnnn-2-2000-2000000000000000000000000000000000nan00000000-20000000000000000000
200  (end of '7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%
SITE EXEC 7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.
200-7 mmmmnnnn-2-2000-2000000000000000000000000000000000nan00000000-20000000000000000000
200  (end of '7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%
SITE EXEC 7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.
200-7 mmmmnnnn-2-2000-2000000000000000000000000000000000nan00000000-20000000000000000000
200  (end of '7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%
SITE EXEC 7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.
200-7 mmmmnnnn-2-2000-2000000000000000000000000000000000nan00000000-20000000000000000000
200  (end of '7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%
SITE EXEC 7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.
200-7 mmmmnnnn-2-2000-2000000000000000000000000000000000nan00000000-20000000000000000000
200  (end of '7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%
SITE EXEC 7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.
200-7 mmmmnnnn-2-2000-2000000000000000000000000000000000nan00000000-20000000000000000000
200  (end of '7 mmmmnnnn%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%.f%
```

Entire conversation (108122 bytes)

● ASCII ○ EBCDIC ○ Hex Dump ○ C Arrays

Detect #1 – screenshot of sample rules triggered by 10.10.10.186 (total of 151)

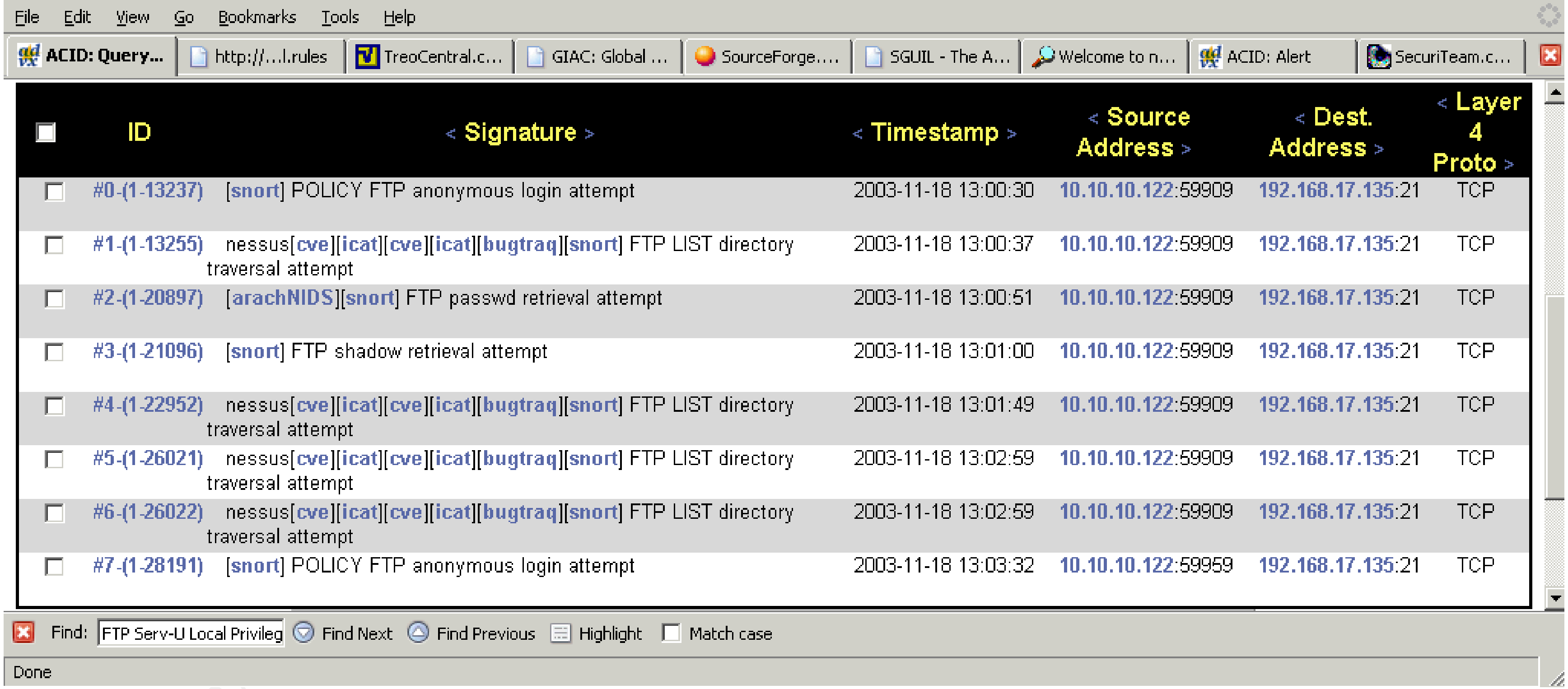


□	ID	< Signature >	< Timestamp >	< Source Address >	< Dest. Address >	< Layer 4 Proto >
□	#0-(1-11442)	[snort] (snort_decoder): Short UDP packet, length field > payload length	2003-11-18 12:59:48	10.10.10.186	10.10.10.2	UDP
□	#1-(1-11444)	[snort] (snort_decoder): Short UDP packet, length field > payload length	2003-11-18 12:59:48	10.10.10.2	10.10.10.186	UDP
□	#2-(1-30587)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:04:32	10.10.10.186:32802	172.20.201.198:21	TCP
□	#3-(1-30588)	url[snort] BLEEDING-EDGE FTP Serv-U Local Privilege Escalation Vulnerability	2003-11-18 13:04:32	10.10.10.186:32802	172.20.201.198:21	TCP
□	#4-(1-30589)	[snort] FTP SITE EXEC format string attempt	2003-11-18 13:04:32	10.10.10.186:32802	172.20.201.198:21	TCP
□	#5-(1-30590)	[cve][icat][cve][icat][bugtraq][arachNIDS][snort] FTP SITE EXEC attempt	2003-11-18 13:04:32	10.10.10.186:32802	172.20.201.198:21	TCP
□	#6-(1-30809)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:05:15	10.10.10.186:32803	172.20.201.198:21	TCP
□	#7-(1-31420)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:07:22	10.10.10.186:32805	172.20.201.198:21	TCP
□	#8-(1-31421)	url[snort] BLEEDING-EDGE FTP Serv-U Local Privilege Escalation Vulnerability	2003-11-18 13:07:22	10.10.10.186:32805	172.20.201.198:21	TCP
□	#9-(1-31422)	[snort] FTP SITE EXEC format string attempt	2003-11-18 13:07:22	10.10.10.186:32805	172.20.201.198:21	TCP
□	#10-(1-31423)	[cve][icat][cve][icat][bugtraq][arachNIDS][snort] FTP SITE EXEC attempt	2003-11-18 13:07:22	10.10.10.186:32805	172.20.201.198:21	TCP
□	#11-(1-31526)	[snort] (spp_stream4) possible EVASIVE RST detection	2003-11-18 13:07:37	10.10.10.186:32805	172.20.201.198:21	TCP
□	#12-(1-32398)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:09:46	10.10.10.186:32806	172.20.201.198:21	TCP
□	#13-(1-32399)	url[snort] BLEEDING-EDGE FTP Serv-U Local Privilege Escalation Vulnerability	2003-11-18 13:09:46	10.10.10.186:32806	172.20.201.198:21	TCP
□	#14-(1-32400)	[snort] FTP SITE EXEC format string attempt	2003-11-18 13:09:46	10.10.10.186:32806	172.20.201.198:21	TCP
□	#15-(1-32401)	[cve][icat][cve][icat][bugtraq][arachNIDS][snort] FTP SITE EXEC attempt	2003-11-18 13:09:46	10.10.10.186:32806	172.20.201.198:21	TCP
□	#16-(1-32422)	[snort] (spp_stream4) possible EVASIVE RST detection	2003-11-18 13:09:50	10.10.10.186:32806	172.20.201.198:21	TCP
□	#17-(1-32478)	[snort] (spp_stream4) possible EVASIVE RST detection	2003-11-18 13:10:00	10.10.10.186:32806	172.20.201.198:21	TCP
□	#18-(1-32937)	url[snort] BLEEDING-EDGE Virus Possible Sober.j Outbound	2003-11-18 13:11:37	10.10.10.186:32845	172.20.201.198:37	TCP
□	#19-(1-32938)	[cve][icat][cve][icat][bugtraq][bugtraq][bugtraq][snort] SNMP request tcp	2003-11-18 13:11:38	10.10.10.186:32969	172.20.201.198:161	TCP
□	#20-(1-32939)	[cve][icat][cve][icat][bugtraq][bugtraq][bugtraq][snort] SNMP trap tcp	2003-11-18 13:11:38	10.10.10.186:32970	172.20.201.198:162	TCP
□	#21-(1-32946)	[cve][icat][cve][icat][bugtraq][bugtraq][bugtraq][snort] SNMP AgentX/tcp request	2003-11-18 13:11:39	10.10.10.186:33513	172.20.201.198:705	TCP
□	#22-(1-33103)	[arachNIDS][snort] BLEEDING-EDGE SCAN NMAP -sA	2003-11-18 13:12:06	10.10.10.186:48602	172.20.201.198:21	TCP
□	#23-(1-33104)	[arachNIDS][snort] BLEEDING-EDGE SCAN NMAP -sA	2003-11-18 13:12:06	10.10.10.186:48604	172.20.201.198:1	TCP
□	#24-(1-33105)	[arachNIDS][snort] SCAN nmap XMAS	2003-11-18 13:12:06	10.10.10.186:48605	172.20.201.198:1	TCP
□	#25-(1-33126)	[snort] (spp_stream4) possible EVASIVE RST detection	2003-11-18 13:12:09	10.10.10.186:48593	172.20.201.198:21	TCP
□	#26-(1-33128)	[snort] (spp_stream4) possible EVASIVE RST detection	2003-11-18 13:12:09	10.10.10.186:48594	172.20.201.198:21	TCP
□	#27-(1-33129)	[snort] (spp_stream4) possible EVASIVE RST detection	2003-11-18 13:12:09	10.10.10.186:48595	172.20.201.198:21	TCP
□	#28-(1-33130)	[snort] (spp_stream4) possible EVASIVE RST detection	2003-11-18 13:12:09	10.10.10.186:48596	172.20.201.198:21	TCP
□	#29-(1-33132)	[snort] (spp_stream4) possible EVASIVE RST detection	2003-11-18 13:12:09	10.10.10.186:48597	172.20.201.198:21	TCP
□	#30-(1-33133)	[snort] (spp_stream4) possible EVASIVE RST detection	2003-11-18 13:12:10	10.10.10.186:48598	172.20.201.198:21	TCP
□	#31-(1-33135)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:12:10	10.10.10.186:48172	172.20.201.198:21	TCP
□	#32-(1-33136)	[snort] (spp_stream4) possible EVASIVE RST detection	2003-11-18 13:12:10	10.10.10.186:48172	172.20.201.198:21	TCP
□	#33-(1-33146)	[snort] INFO FTP Bad login	2003-11-18 13:12:11	172.20.201.198:21	10.10.10.186:48173	TCP
□	#34-(1-33147)	[snort] (spp_stream4) possible EVASIVE RST detection	2003-11-18 13:12:11	10.10.10.186:48173	172.20.201.198:21	TCP
□	#35-(1-33148)	[cve][icat][bugtraq][bugtraq][snort] FTP CWD ~ attempt	2003-11-18 13:12:12	10.10.10.186:48182	172.20.201.198:21	TCP
□	#36-(1-33149)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:12:12	10.10.10.186:48174	172.20.201.198:21	TCP
□	#37-(1-33150)	[snort] (spp_stream4) possible EVASIVE RST detection	2003-11-18 13:12:12	10.10.10.186:48175	172.20.201.198:21	TCP
□	#38-(1-33152)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:12:12	10.10.10.186:48176	172.20.201.198:21	TCP

Detect number #2 – attacker 10.10.10.122 → victim 192.168.17.135

```
$ sudo ngrep -x -q -I 2003.12.15.cap 'RETR passwd'
input: 2003.12.15.cap

T 10.10.10.122:59909 -> 192.168.17.135:21 [AP]
  52 45 54 52 20 70 61 73    73 77 64 0d 0a           RETR passwd..

T 10.10.10.212:4638 -> 192.168.17.135:21 [AP]
  52 45 54 52 20 70 61 73    73 77 64 0d 0a           RETR passwd..

$ sudo ngrep -x -q -I 2003.12.15.cap 'RETR shadow'
input: 2003.12.15.cap

T 10.10.10.122:59909 -> 192.168.17.135:21 [AP]
  52 45 54 52 20 73 68 61    64 6f 77 0d 0a           RETR shadow..

$ sudo ngrep -x -q -I 2003.12.15.cap 'PASV'
input: 2003.12.15.cap

T 10.10.10.122:59909 -> 192.168.17.135:21 [AP]
  50 41 53 56 0d 0a                                  PASV..

T 10.10.10.122:59909 -> 192.168.17.135:21 [AP]
  50 41 53 56 0d 0a                                  PASV..

T 10.10.10.122:59909 -> 192.168.17.135:21 [AP]
  50 41 53 56 0d 0a                                  PASV..
… and so on
```

Rules triggered by the detect #2 – ***ACID*** screenshot:



□	ID	< Signature >	< Timestamp >	< Source Address >	< Dest. Address >	< Layer 4 Proto >
□	#0-(1-13237)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:00:30	10.10.10.122:59909	192.168.17.135:21	TCP
□	#1-(1-13255)	nessus[cve][icat][cve][icat][bugtraq][snort] FTP LIST directory traversal attempt	2003-11-18 13:00:37	10.10.10.122:59909	192.168.17.135:21	TCP
□	#2-(1-20897)	[arachNIDS][snort] FTP passwd retrieval attempt	2003-11-18 13:00:51	10.10.10.122:59909	192.168.17.135:21	TCP
□	#3-(1-21096)	[snort] FTP shadow retrieval attempt	2003-11-18 13:01:00	10.10.10.122:59909	192.168.17.135:21	TCP
□	#4-(1-22952)	nessus[cve][icat][cve][icat][bugtraq][snort] FTP LIST directory traversal attempt	2003-11-18 13:01:49	10.10.10.122:59909	192.168.17.135:21	TCP
□	#5-(1-26021)	nessus[cve][icat][cve][icat][bugtraq][snort] FTP LIST directory traversal attempt	2003-11-18 13:02:59	10.10.10.122:59909	192.168.17.135:21	TCP
□	#6-(1-26022)	nessus[cve][icat][cve][icat][bugtraq][snort] FTP LIST directory traversal attempt	2003-11-18 13:02:59	10.10.10.122:59909	192.168.17.135:21	TCP
□	#7-(1-28191)	[snort] POLICY FTP anonymous login attempt	2003-11-18 13:03:32	10.10.10.122:59959	192.168.17.135:21	TCP

Find: FTP Serv-U Local Privileg

Rules triggered by detect #2:

http://www.snort.org/snort-db/sid.html?sid=553 ➔ *POLICY FTP anonymous login attempt*
Rule: alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"POLICY FTP anonymous login attempt"; flow:to_server,established; content:"USER"; nocase; pcre:"/^USER\s+(anonymous|ftp)/smi"; classtype:misc-activity; sid:553; rev:7;) – **plain English:** match regular expression (**pcre**), regardless of the case (upper/lower case letters- **nocase**), of the word **user**, in the beginning of a line (**^**), with one or more spaces following this word (**\s+**), followed by either the word **anonymous**, or (**|**) the word **ftp** (both known accounts used in anonymous ftp). This has to appear in the

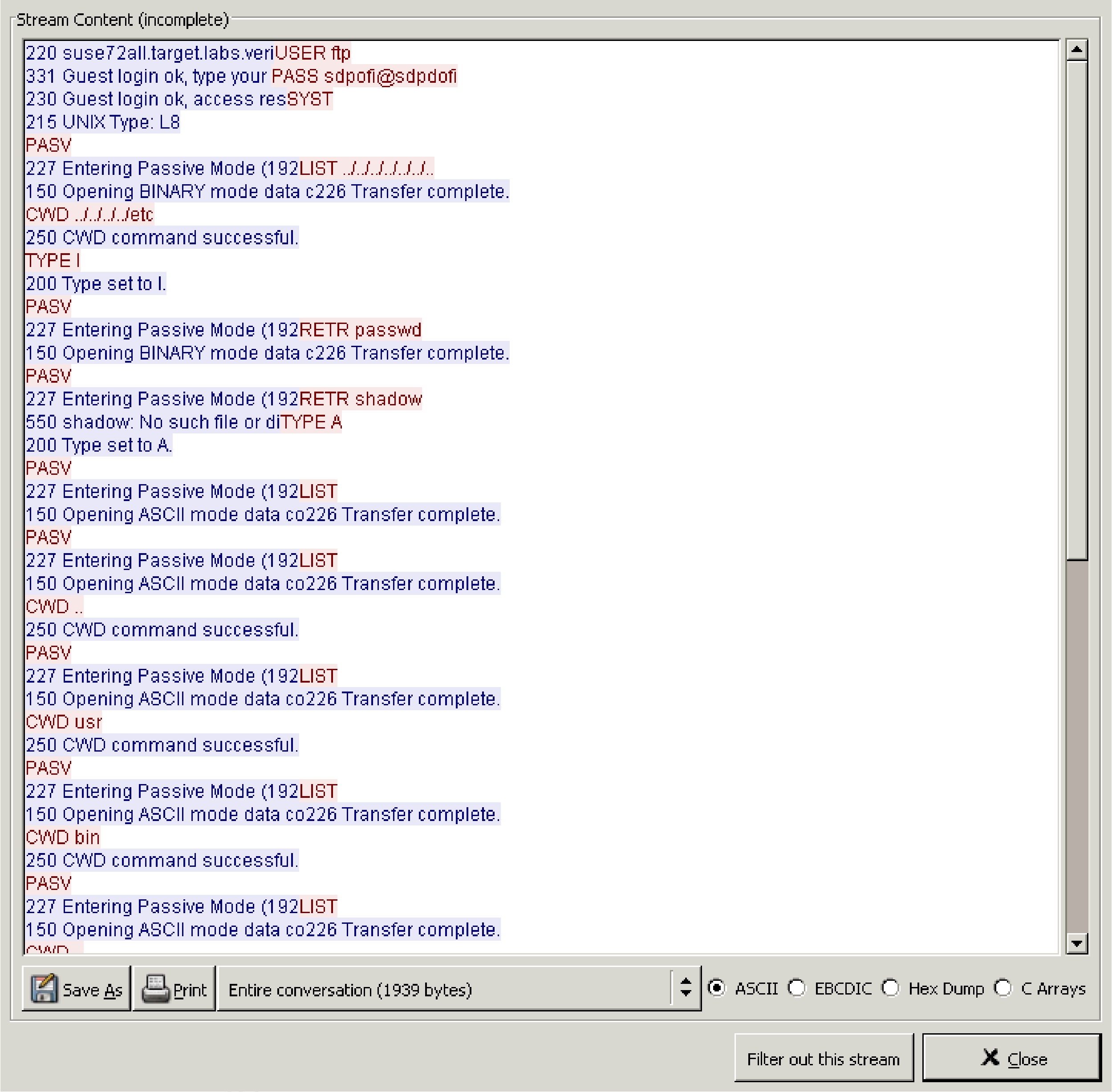
traffic from client from to server, in an established sessions, with the server listening on TCP port 21 (standard **ftp).**

http://www.snort.org/snort-db/sid.html?sid=1992 ➔ *FTP LIST directory traversal attempt* **Rule**: alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP LIST directory traversal attempt"; flow:to_server,established; content:"LIST"; nocase; content:".."; distance:1; content:".."; distance:1; reference:bugtraq,2618; reference:cve,2001-0680; reference:cve,2002-1054; reference:nessus,11112; classtype:protocol-command-decode; sid:1992; rev:8;) **– in plain English**: request from a client, to a server listening on TCP port 21 (standard **ftp),** in an established session, for a string in the format **LIST .. ..** (two pairs of two dots, with one character length "space" interval from the word LIST, and from each other – the space could be anything) ➔ matches "**LIST ../../**", for example.

http://www.snort.org/snort-db/sid.html?sid=356 ➔ *FTP passwd retrieval attempt* **Rule**: alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP passwd retrieval attempt"; flow:to_server,established; content:"RETR"; nocase; content:"passwd"; reference:arachnids,213; classtype:suspicious-filename-detect; sid:356; rev:5;) – **in plain English:** same environment as before (client from $EXTERNAL_NET, in a TCP session established to server from $HOME_NET, with the latter communicating on TCP port 21 (**ftp**)), with the case-sensitive **RETR** word, followed by the word **passwd**, regardless of its case.

http://www.snort.org/snort-db/sid.html?sid=1928 ➔ *FTP shadow retrieval attempt* **Rule**: alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"FTP shadow retrieval attempt"; flow:to_server,established; content:"RETR"; nocase; content:"shadow"; classtype:suspicious-filename-detect; sid:1928; rev:3;) – **in plain English:** same as above, with the replacement of **passwd** with **shadow**. For these last two rules, these are names of files which – under Unix – are possible storage places for user/account names and passwords.

Ethereal screenshots (with some overlap, for proper identification of the whole “flow/exchange” of information) of the **ftp** session from **detect #2:**

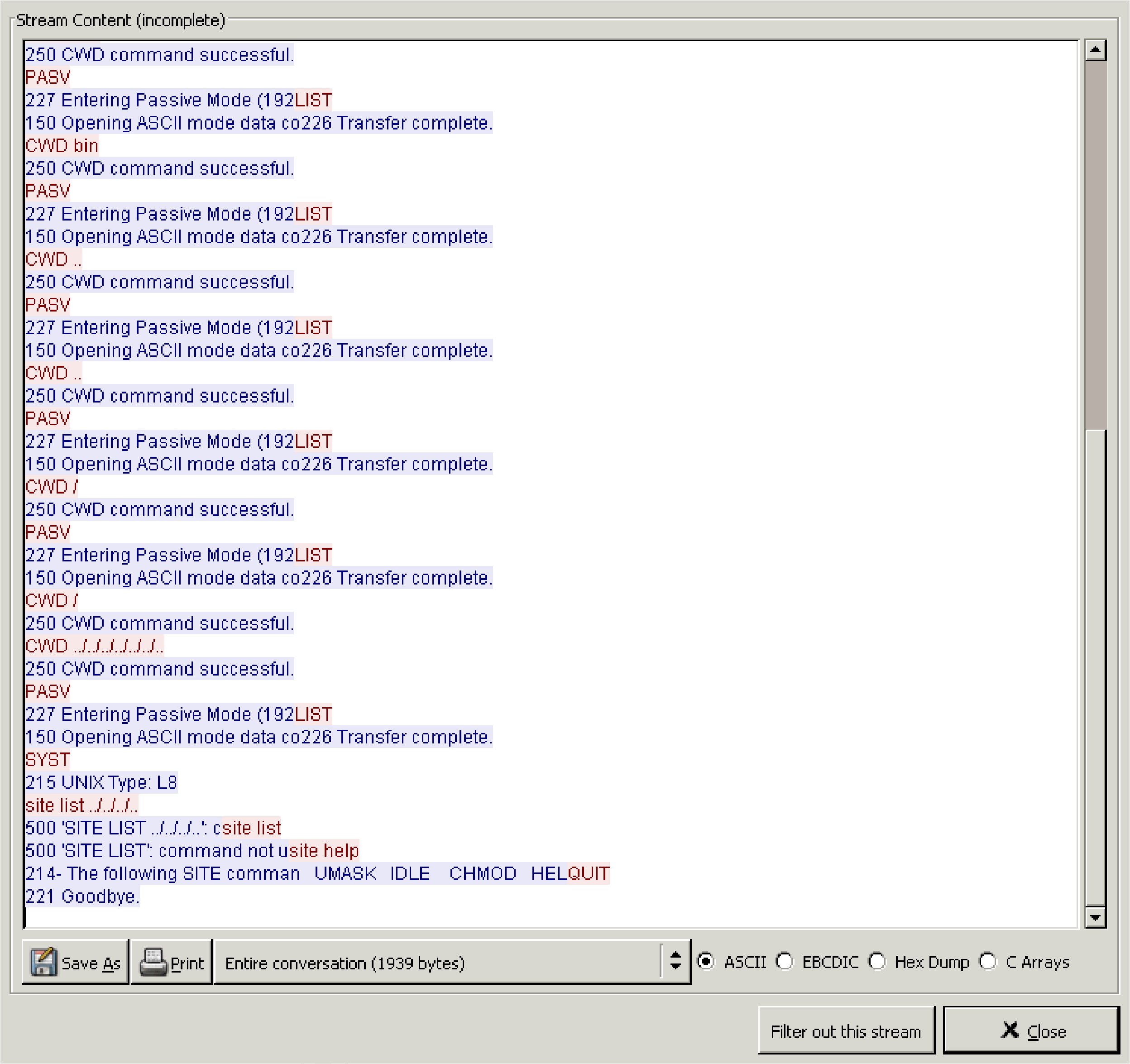


Stream Content (incomplete)

```
220 suse72all.target.labs.veriUSER ftp
331 Guest login ok, type your PASS sdpofi@sdpdofi
230 Guest login ok, access resSYST
215 UNIX Type: L8
PASV
227 Entering Passive Mode (192LIST ../../../../../..
150 Opening BINARY mode data c226 Transfer complete.
CWD ../../../../etc
250 CWD command successful.
TYPE I
200 Type set to I.
PASV
227 Entering Passive Mode (192RETR passwd
150 Opening BINARY mode data c226 Transfer complete.
PASV
227 Entering Passive Mode (192RETR shadow
550 shadow: No such file or diTYPE A
200 Type set to A.
PASV
227 Entering Passive Mode (192LIST
150 Opening ASCII mode data co226 Transfer complete.
PASV
227 Entering Passive Mode (192LIST
150 Opening ASCII mode data co226 Transfer complete.
CWD ..
250 CWD command successful.
PASV
227 Entering Passive Mode (192LIST
150 Opening ASCII mode data co226 Transfer complete.
CWD usr
250 CWD command successful.
PASV
227 Entering Passive Mode (192LIST
150 Opening ASCII mode data co226 Transfer complete.
CWD bin
250 CWD command successful.
PASV
227 Entering Passive Mode (192LIST
150 Opening ASCII mode data co226 Transfer complete.
CWD ..
```

Entire conversation (1939 bytes)

● ASCII ○ EBCDIC ○ Hex Dump ○ C Arrays



Stream Content (incomplete)
Save As
Print
Entire conversation (1939 bytes)
● ASCII ○ EBCDIC ○ Hex Dump ○ C Arrays
Filter out this stream
Close

```
250 CWD command successful.
PASV
227 Entering Passive Mode (192LIST
150 Opening ASCII mode data co226 Transfer complete.
CWD bin
250 CWD command successful.
PASV
227 Entering Passive Mode (192LIST
150 Opening ASCII mode data co226 Transfer complete.
CWD ..
250 CWD command successful.
PASV
227 Entering Passive Mode (192LIST
150 Opening ASCII mode data co226 Transfer complete.
CWD ..
250 CWD command successful.
PASV
227 Entering Passive Mode (192LIST
150 Opening ASCII mode data co226 Transfer complete.
CWD /
250 CWD command successful.
PASV
227 Entering Passive Mode (192LIST
150 Opening ASCII mode data co226 Transfer complete.
CWD /
250 CWD command successful.
CWD ../../../../../../..
250 CWD command successful.
PASV
227 Entering Passive Mode (192LIST
150 Opening ASCII mode data co226 Transfer complete.
SYST
215 UNIX Type: L8
site list ../../../..
500 'SITE LIST ../../../..': csite list
500 'SITE LIST': command not usite help
214- The following SITE comman   UMASK   IDLE    CHMOD   HELQUIT
221 Goodbye.
```

Running this:

```
$ sudo tcpdump -nnnxr 2003.12.15.cap 'host 10.10.10.122 and !host 192.168.17.135' |
grep –A 4 135.17.168.192
```

gave me this:

```
45149 174.888803 10.10.10.122 -> 10.10.10.2   DNS Standard query PTR
135.17.168.192.in-addr.arpa

0000  00 50 56 40 00 64 00 06 5b d8 bf ed 08 00 45 00   .PV@.d..[.....E.
0010  00 49 9d 4b 40 00 40 11 74 c9 0a 0a 0a 7a 0a 0a   .I.K@.@.t....z..
0020  0a 02 80 e4 00 35 00 35 59 73 73 46 01 00 00 01   .....5.5YssF....
```

```
<snipped similar>
```

Who is the victim communicating with, besides the attacker, and “what about”?

$ sudo ngrep -x -n -q -I 2003.12.15.cap '' "host 192.168.17.135 and not host 10.10.10.122*"

- **SMTP traffic**

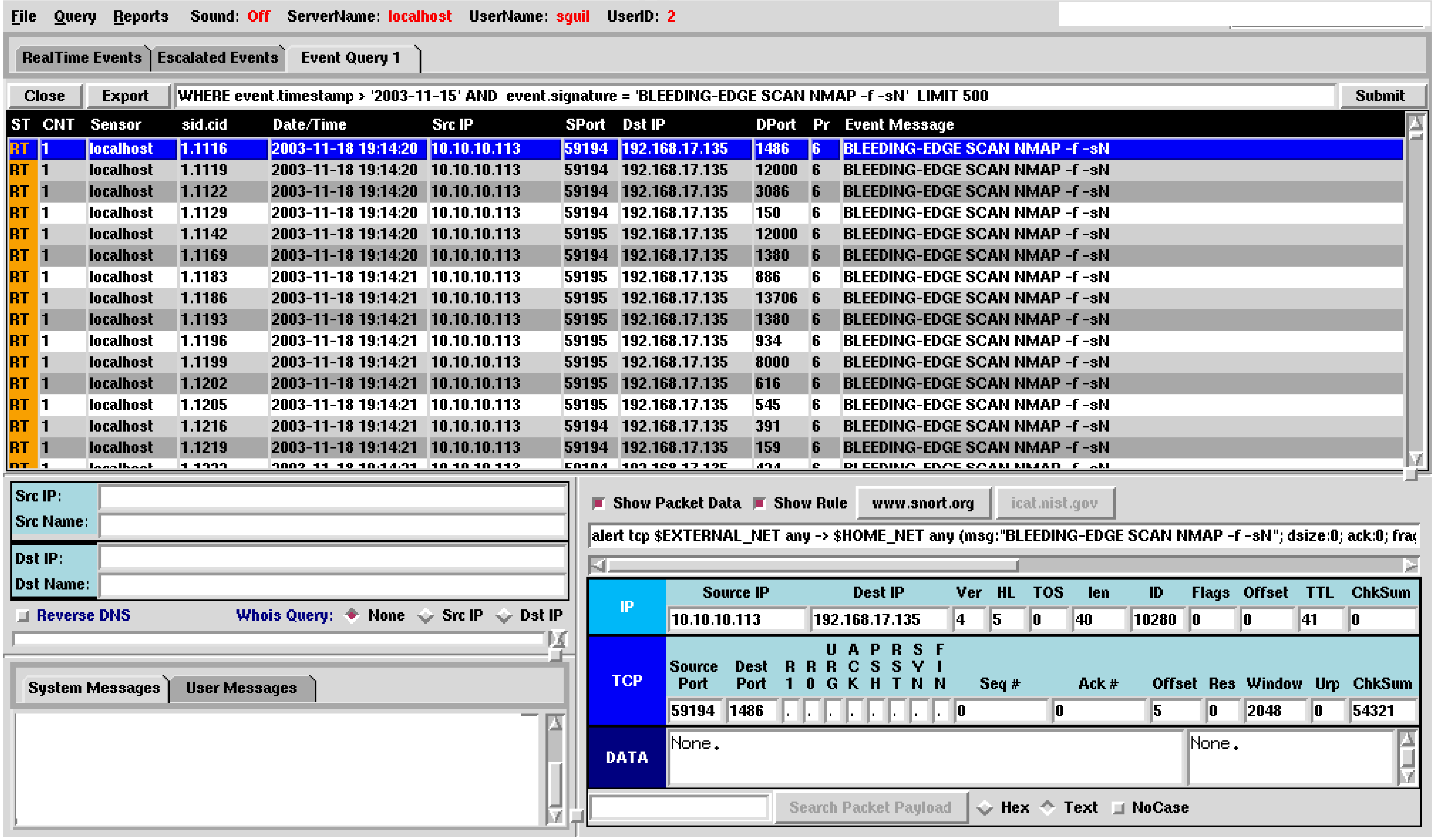
```
T 10.10.10.226:34344 -> 192.168.17.135:25 [AP]
  0d 0a                                             ..
######################################################################
#
<and the same pattern for more …>

<snip>

T 192.168.17.135:25 -> 10.10.10.226:34344 [AP]
  32 31 34 2d 32 2e 30 2e    30 20 52 43 50 54 20 54    214-2.0.0 RCPT T
  4f 3a 20 3c 72 65 63 69    70 69 65 6e 74 3e          O: <recipient>
######################################################################
#<and the same pattern for more…>

T 10.10.10.226:34344 -> 192.168.17.135:25 [AP]
  4d 41 49 4c 20 46 52 4f    4d 3a 20 72 6f 6f 74 40    MAIL FROM: root@
  61 74 74 61 63 6b 65 72    73 2e 6f 72 67 0d 00       attackers.org..
######################################################################
<and the same pattern for more …>
```

- **FTP traffic** – similar to the one already seen between the attacker and the victim, this time with other systems, with the same victim (ending up in the same “upward directory traversal already discussed)

```
T 192.168.17.135:21 -> 10.10.10.212:4638 [AP]
  32 32 30 20 73 75 73 65    37 32 61 6c 6c 2e 74 61    220 suse72all.ta
  72 67 65 74 2e 6c 61 62    73 2e 76 65 72 69 74 65    rget.labs.verite
  63 74 20 46 54 50 20 73    65 72                      ct FTP ser

T 10.10.10.212:4638 -> 192.168.17.135:21 [AP]
  55 53 45 52 20 61 6e 6f    6e 79 6d 6f 75 73 0d 0a    USER anonymous..

T 192.168.17.135:21 -> 10.10.10.212:4638 [AP]
  33 33 31 20 47 75 65 73    74 20 6c 6f 67 69 6e 20    331 Guest login
  6f 6b 2c 20 74 79 70 65    20 79 6f 75 72 20 6e 61    ok, type your na
  6d 65 20 61 73 20 70 61    73 73 00 00 09 04 00 00    me as pass......
  6e                                                    n
```

```
T 10.10.10.212:4638 -> 192.168.17.135:21 [AP]
  50 41 53 53 20 62 6c 61    68 40 62 6c 61 68 63 6f    PASS blah@blahco
  6d 0d 0a                                              m..

T 192.168.17.135:21 -> 10.10.10.212:4638 [AP]
  32 33 30 20 47 75 65 73    74 20 6c 6f 67 69 6e 20    230 Guest login
  6f 6b 2c 20 61 63 63 65    73 73 20 72 65 73 74 72    ok, access restr
  69 63 74 69 6f 6e 73 20    61 70 00 00 09 04 00 00    ictions ap......

<and so on …>
<snip>

T 192.168.17.135:21 -> 10.10.10.212:4638 [AP]
  31 35 30 20 4f 70 65 6e    69 6e 67 20 42 49 4e 41    150 Opening BINA
  52 59 20 6d 6f 64 65 20    64 61 74 61 20 63 6f 6e    RY mode data con
  6e 65 63 74 69 6f 6e 20    66 6f                      nection fo

T 192.168.17.135:20 -> 10.10.10.212:4647 [AFP]
  72 6f 6f 74 3a 78 3a 30    3a 30 3a 53 75 70 65 72    root:x:0:0:Super
  20 55 73 65 72 3a 2f 72    6f 6f 74 3a 2f 62 00 00    User:/root:/b..
  09 04 00 00 6e 6f                                     ....no

T 192.168.17.135:21 -> 10.10.10.212:4638 [AP]
  32 32 36 20 54 72 61 6e    73 66 65 72 20 63 6f 6d    226 Transfer com
  70 6c 65 74 65 2e 0d 0a                               plete...
```

Detect #3 – attacker 10.10.10.113 and sample bleeding-edge rule
sguil screenshot



Sound: Off ServerName: localhost UserName: sguil UserID: 2

WHERE event.timestamp > '2003-11-15' AND event.signature = 'BLEEDING-EDGE SCAN NMAP -f -sN' LIMIT 500

ST	CNT	Sensor	sid.cid	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
RT	1	localhost	1.1116	2003-11-18 19:14:20	10.10.10.113	59194	192.168.17.135	1486	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1119	2003-11-18 19:14:20	10.10.10.113	59194	192.168.17.135	12000	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1122	2003-11-18 19:14:20	10.10.10.113	59194	192.168.17.135	3086	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1129	2003-11-18 19:14:20	10.10.10.113	59194	192.168.17.135	150	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1142	2003-11-18 19:14:20	10.10.10.113	59195	192.168.17.135	12000	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1169	2003-11-18 19:14:20	10.10.10.113	59194	192.168.17.135	1380	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1183	2003-11-18 19:14:21	10.10.10.113	59195	192.168.17.135	886	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1186	2003-11-18 19:14:21	10.10.10.113	59195	192.168.17.135	13706	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1193	2003-11-18 19:14:21	10.10.10.113	59195	192.168.17.135	1380	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1196	2003-11-18 19:14:21	10.10.10.113	59195	192.168.17.135	934	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1199	2003-11-18 19:14:21	10.10.10.113	59195	192.168.17.135	8000	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1202	2003-11-18 19:14:21	10.10.10.113	59195	192.168.17.135	616	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1205	2003-11-18 19:14:21	10.10.10.113	59195	192.168.17.135	545	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1216	2003-11-18 19:14:21	10.10.10.113	59194	192.168.17.135	391	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1219	2003-11-18 19:14:21	10.10.10.113	59194	192.168.17.135	159	6	BLEEDING-EDGE SCAN NMAP -f -sN
RT	1	localhost	1.1222	2003-11-18 19:14:21	10.10.10.113	59194	192.168.17.135	424	6	BLEEDING-EDGE SCAN NMAP -f -sN

Src IP:
Src Name:
Dst IP:
Dst Name:

□ Reverse DNS Whois Query: ● None ○ Src IP ○ Dst IP

☒ Show Packet Data ☒ Show Rule

```
alert tcp $EXTERNAL_NET any -> $HOME_NET any (msg:"BLEEDING-EDGE SCAN NMAP -f -sN"; dsize:0; ack:0; frag
```

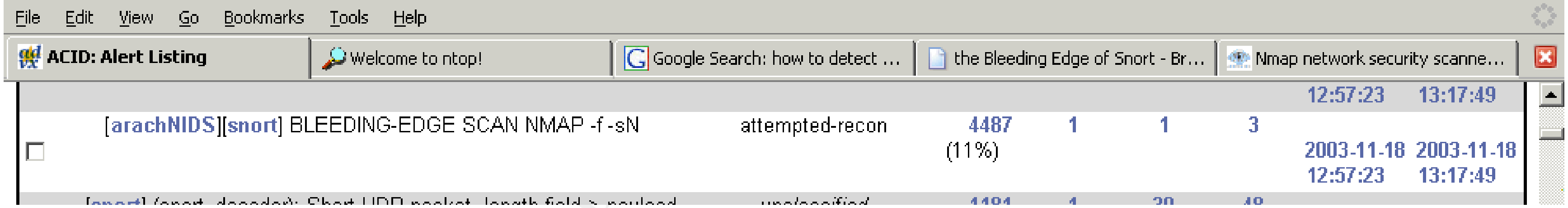
IP	Source IP	Dest IP	Ver	HL	TOS	len	ID	Flags	Offset	TTL	ChkSum
	10.10.10.113	192.168.17.135	4	5	0	40	10280	0	0	41	0

TCP	Source Port	Dest Port	R 1	R 0	U R G	A C K	P S H	R S T	S Y N	F I N	Seq #	Ack #	Offset	Res	Window	Urp	ChkSum
	59194	1486	.	.	.	.	.	.	.	.	0	0	5	0	2048	0	54321

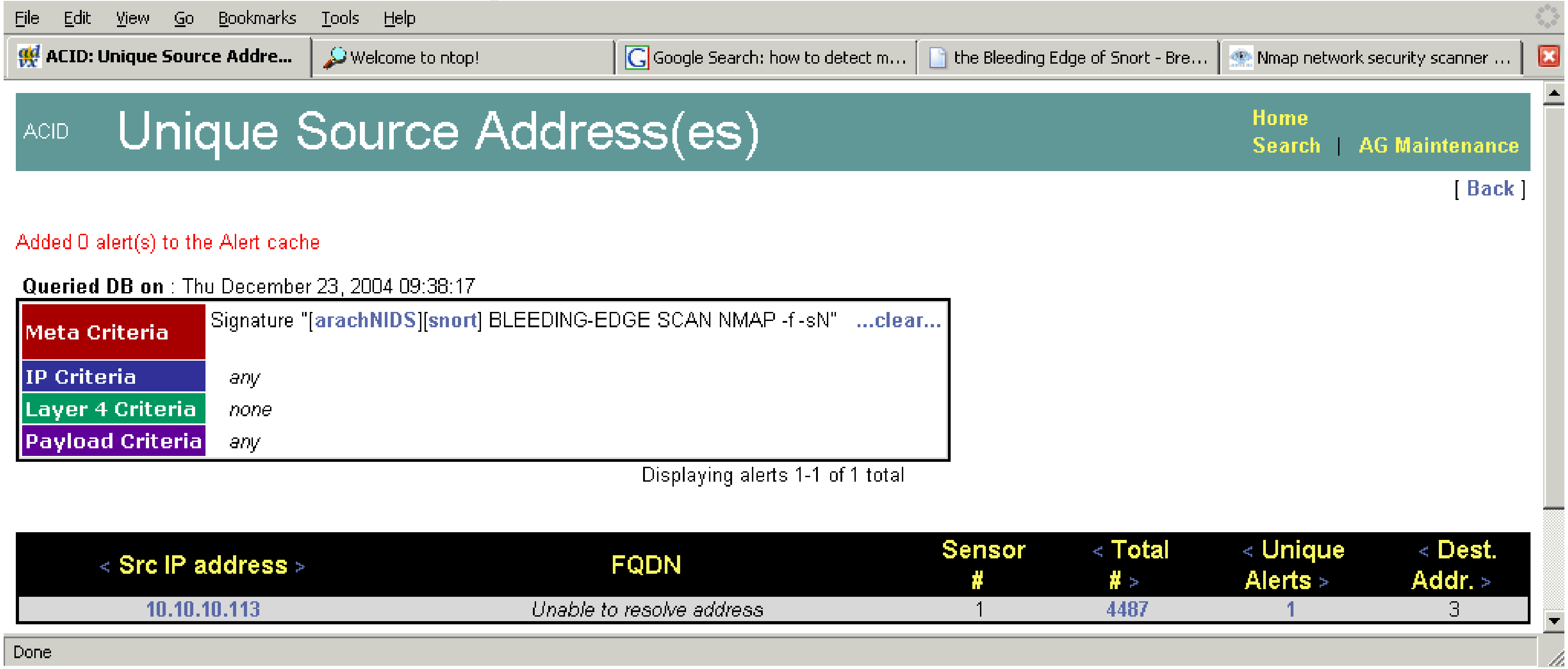
DATA None. None.

○ Hex ○ Text □ NoCase

ACID screenshot of the rule:


File Edit View Go Bookmarks Tools Help
ACID: Alert Listing
Welcome to ntop!
Google Search: how to detect ...
the Bleeding Edge of Snort - Br...
Nmap network security scanne...

```
                                                                                    12:57:23   13:17:49
☐  [arachNIDS][snort] BLEEDING-EDGE SCAN NMAP -f -sN   attempted-recon   4487   1   1   3
                                                                  (11%)          2003-11-18 2003-11-18
                                                                                  12:57:23   13:17:49
   [snort] (snort_decoder): Short UDP packet, length field > payload   unclassified   1181   1   20   48
```


File Edit View Go Bookmarks Tools Help
ACID: Unique Source Addre...
Welcome to ntop!
Google Search: how to detect m...
the Bleeding Edge of Snort - Bre...
Nmap network security scanner ...
ACID
Home
Search | AG Maintenance
[Back]
Done

Unique Source Address(es)

Added 0 alert(s) to the Alert cache

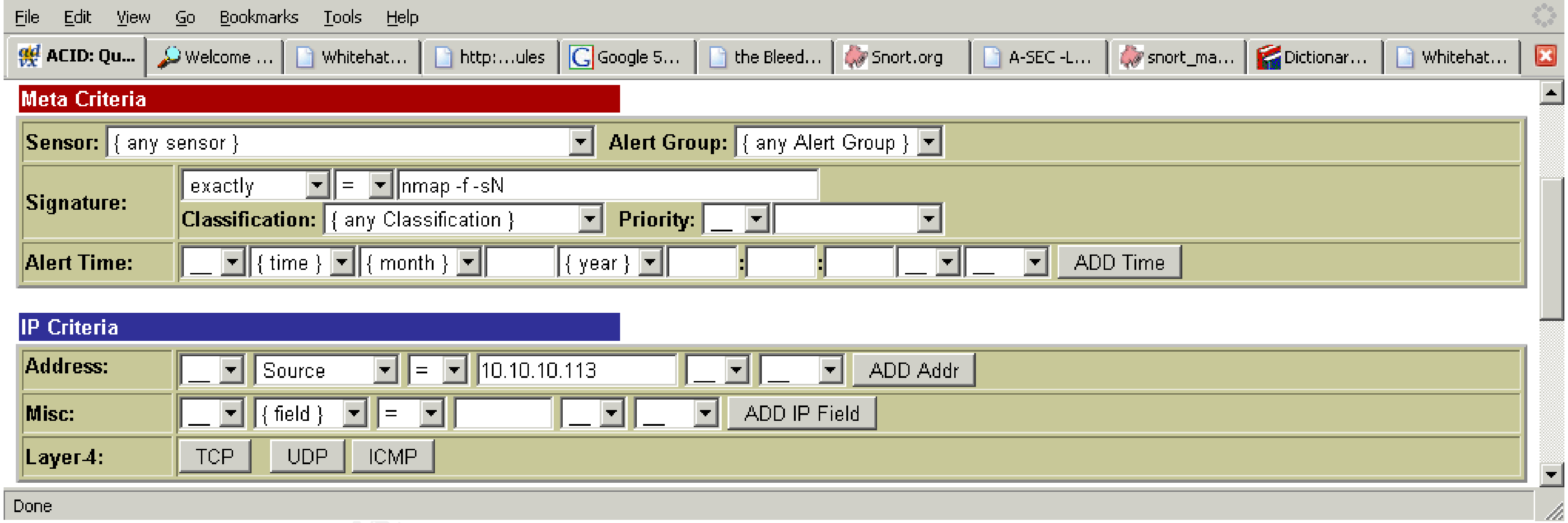
Queried DB on : Thu December 23, 2004 09:38:17

Meta Criteria	Signature "[arachNIDS][snort] BLEEDING-EDGE SCAN NMAP -f -sN" ...clear...
IP Criteria	*any*
Layer 4 Criteria	*none*
Payload Criteria	*any*

Displaying alerts 1-1 of 1 total

< Src IP address >	FQDN	Sensor #	< Total # >	< Unique Alerts >	< Dest. Addr. >
10.10.10.113	*Unable to resolve address*	1	4487	1	3

ACID query for attacker 10.10.10.113 and rule containing ***nmap –f –sN***



Meta Criteria

Sensor: { any sensor } **Alert Group:** { any Alert Group }

Signature: exactly = nmap -f -sN
Classification: { any Classification } **Priority:** __

Alert Time: __ { time } { month } { year } : : __ __ ADD Time

IP Criteria

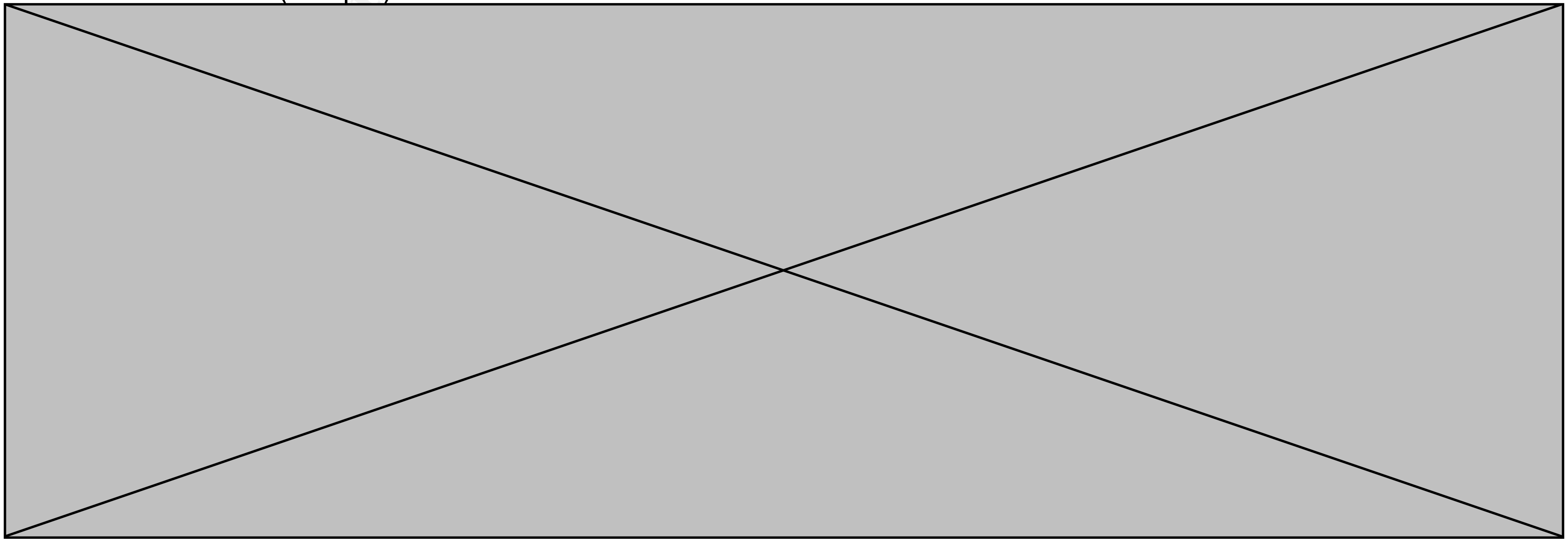
Address: __ Source = 10.10.10.113 __ __ ADD Addr

Misc: __ { field } = __ __ ADD IP Field

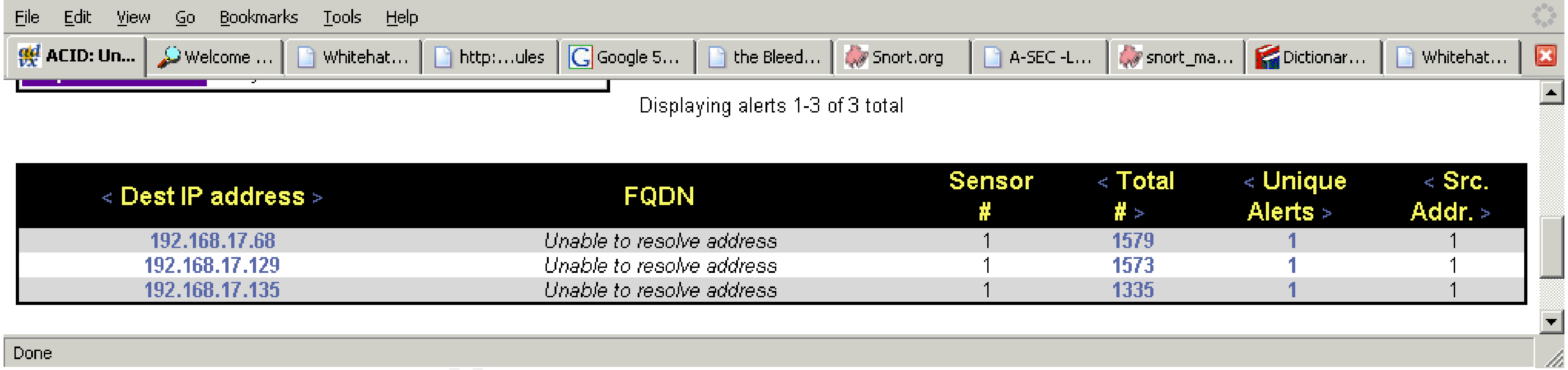
Layer-4: TCP UDP ICMP

Done

Results of the above (sample) – for a total of 4487:



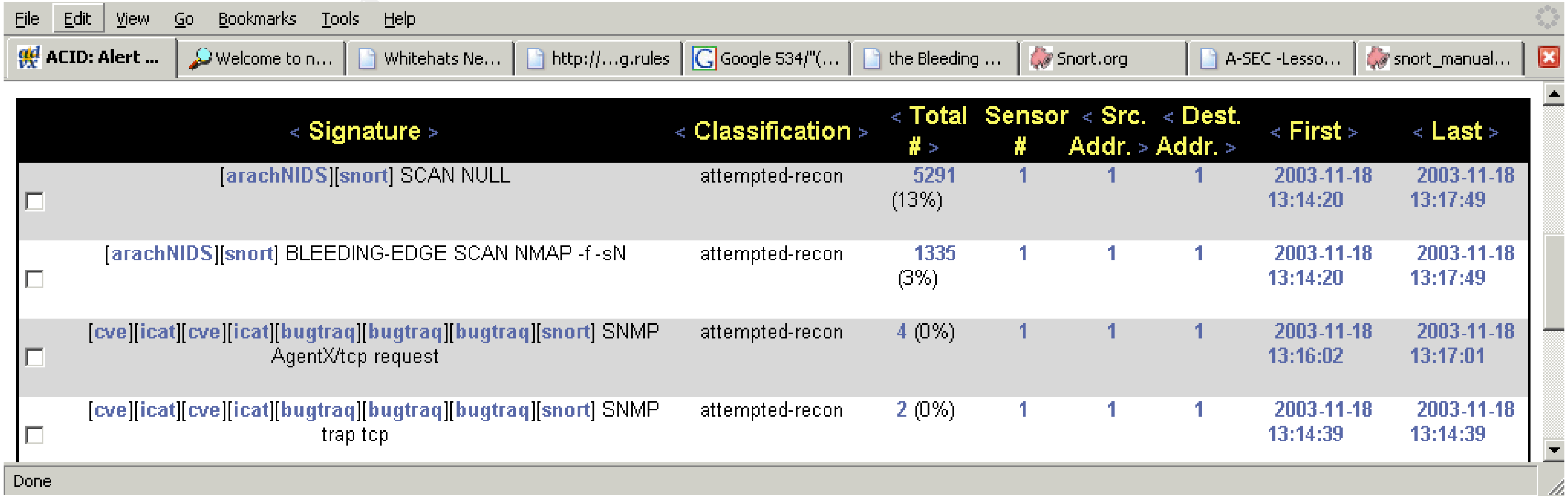
… then, processing the above results even further, for destination (victims) identification, we get three systems:


File Edit View Go Bookmarks Tools Help
ACID: Un... | Welcome ... | Whitehat... | http:...ules | Google 5... | the Bleed... | Snort.org | A-SEC -L... | snort_ma... | Dictionar... | Whitehat...
Done

Displaying alerts 1-3 of 3 total

< Dest IP address >	FQDN	Sensor #	< Total # >	< Unique Alerts >	< Src. Addr. >
192.168.17.68	*Unable to resolve address*	1	1579	1	1
192.168.17.129	*Unable to resolve address*	1	1573	1	1
192.168.17.135	*Unable to resolve address*	1	1335	1	1

New query into ***ACID***, narrowing down previous attacker and one of the above victims, reveals the targeted attack nature even further:


File Edit View Go Bookmarks Tools Help
ACID: Alert ... | Welcome to n... | Whitehats Ne... | http://...g.rules | Google 534/"(... | the Bleeding ... | Snort.org | A-SEC -Lesso... | snort_manual...
Done

	< Signature >	< Classification >	< Total # >	Sensor #	< Src. Addr. >	< Dest. Addr. >	< First >	< Last >
□	[arachNIDS][snort] SCAN NULL	attempted-recon	5291 (13%)	1	1	1	2003-11-18 13:14:20	2003-11-18 13:17:49
□	[arachNIDS][snort] BLEEDING-EDGE SCAN NMAP -f -sN	attempted-recon	1335 (3%)	1	1	1	2003-11-18 13:14:20	2003-11-18 13:17:49
□	[cve][icat][cve][icat][bugtraq][bugtraq][bugtraq][snort] SNMP AgentX/tcp request	attempted-recon	4 (0%)	1	1	1	2003-11-18 13:16:02	2003-11-18 13:17:01
□	[cve][icat][cve][icat][bugtraq][bugtraq][bugtraq][snort] SNMP trap tcp	attempted-recon	2 (0%)	1	1	1	2003-11-18 13:14:39	2003-11-18 13:14:39

Appendix F – open source and/or free tools, used and useful in the practice of network intrusion detection

Tool	Links	Comments
Tools I have used during the project, and mentioned in the paper		
nmap	*www.insecure.org/nmap/index.html*	the scanner of scanners – a tool with potential of auditing, as well as overall network “exploration” – not used as such during an analysis like the one in this paper, but definitely found to have been used by our “attackers”
ntop	*www.ntop.org*	a very powerful network tool – works by capturing live traffic, or accepting netflows, or by reading pcap files; capable of identifying hosts, communications, bandwidth consumed, protocols, trending, etc.
snort	*www.snort.org*	open source sniffer, as well as (mostly) IDS – highly potent, customizable; capable of capturing live data, or reading capture files, producing alerts based on customizable rules; high participation from a powerful users community
(t)ethereal	*www.ethereal.com*	set of tools (tethereal, ethereal = GUI) for network traffic sniffing and processing – extremely good support (dissectors) for a huge amount of protocols (data link, network, transport and application)
tcpdump	*www.tcpdump.org*	cross-platform (almost – Windows has windump) sniffer – the oldest “in the block” – still very powerful
ACID	*acidlab.sourceforge.net*	PHP-based interface into database of security events created in snort NOTE: at the time of this writing BASE (base.secureideas.net) seems to have “spawned” out of ACID, as an alternative
barnyard	*www.snort.org/dl/barnyard*	detached, fast processing engine to be used in conjunction with snort, to relieve the latter of the output-processing intensive tasks
sguil	*sguil.sourceforge.net*	GUI (X-based) capable of consolidating the information obtained in the snort alerts, via a backend MySql database, and barnyard
editcap; mergecap	*www.ethereal.com*	two very useful utilities, usually “bundled” with ethereal, capable of slicing or merging various format capture files, for later processing

ipsumdump	www.cs.ucla.edu/~kohler/ipsumdump	program capable of summarizing TCP/IP capture files, producing ASCII-based output and statistics; customizable via output options
tcpurify	masaka.cs.ohiou.edu/~eblanton/tcpurify	privacy-oriented, “sanitizer” sniffer; good for changing/randomizing data, if traces are to be made public, or shared with third parties NOTE: mentioned, but not necessary in my paper
tcpslice	tcpdump.org/other/tcpslice.tar.Z	as the name implies, this tool is capable of “slicing” the capture data, on various ways
tcpflow	www.circlemud.org/~jelson/software/tcpflow	sniffer-like program capable of reconstructing TCP session data, from either live capture, or reading from capture file
tcptrace	jarok.cs.ohiou.edu/software/tcptrace/tcptrace.html	tool to consolidate various pieces of information concerning TCP sessions/connections, obtained by reading capture files
p0f	lcamtuf.coredump.cx/p0f.shtml	passive operating system fingerprinting tool – very useful when processing existing capture files, as well as while “listening” on live networks
tcpdstat	www.csl.sony.co.jp/person/kjc/kjc/papers/freenix2000/node14.html	statistic at your fingertip – with this capture files processing tool
tcpick	tcpick.sourceforge.net	textmode libpcap-based sniffer, capable of tracking, reordering and reassembling tcp streamd; I personally like its colorized output, which makes items of interest stand out easily – good for analysis
ngrep	ngrep.sourceforge.net	a network capture processing tools, pcap-aware, equivalent to the UNIX grep utility (finds regex inside data payloads) – very powerful – a must for a security analyst, to be used in conjunction with strings
Other tools I have used in the past, that I recommend for network and security processing		
argus + ra	www.qosient.com/argus	processing a capture file through the argus server component (which produces a proprietary formatted data/records) offers then an incredible wealth of options for investigating such data with the ra client
netwox/ag/ib	www.laurentconstantin.com/en	an incredible set of tools (library + CLI + GUI) – a “netcat”
etherape	etherape.sourceforge.net	graphical sniffer – capable of capturing from live network, or reading from pcap files, then building graphs based on the communication identified between systems

tcpshow	www.software-facilities.com/net-software/tcpshow.php	yet another packet capture display utility – capable of reading capture files in pcap format
tcpreplay	tcpreplay.sourceforge.net	amazing tool for firewall or IDS testing – capable (as the name implies) to take an existing capture file (produces with a pcap-compatible sniffer), and “shoot it” back into either a live network, or – if properly configured – a virtual interface
tcpillust	www.csl.sony.co.jp/person/nishida/tcpillust.html	though somehow obscure, I like this tool for the simple fact that it processes tcp connections, then depicts them in a manner similar to commercial tools (e.g. Network Instrument’s Observer, OPNET, etc.)
congraph	www.soronlin.nildram.co.uk/ethereal.html	excellent tool (script) for building connected graphs of systems, from pcap capture files – I have used it extensively in many projects, of smaller scale (graphs tend to become unmanageable if the trace file contains large amounts of systems)