Investigating the performance of Endace DAG monitoring hardware and Intel NICs in the context of Lawful Interception

Amiel A. Heyde
Centre for Advanced Internet Architectures, Technical Report 080222A
Swinburne University of Technology
Melbourne, Australia
amiel@swin.edu.au

***Abstract*—SPLIT is a simple prototype Lawful Interception software system which is used to evaluate the viability of using both standard network cards and specialised monitoring hardware for the purpose. Lawful Interception requires reliable packet capture, filtering and protocol analysis on high bandwidth links. While standard NICs can take on these various roles, their performance is only adequate in certain situations. Further optimisations to SPLIT are required to reveal exact limits of the NICs, but it is clear that Endace DAG 3.7GF high performance monitoring cards improve performance and reliability.**

I. Introduction

With the ever increasing amount of communication being carried out over the Internet, it is becoming essential for service providers of all types to be able to lawfully intercept transmissions. [1] While in the past it was sufficient to intercept at the application layer, (with applications such as email messages and instant messaging), it is now important that law enforcement agencies are able to access data at the packet level. This is due to the exploding number of various protocols and services for which the Internet is the transport mechanism. Increasing amounts of peer-to-peer traffic such as VoIP and Bittorrent have made interception at the server inadequate.

While it has long been a requirement for telecommunications providers to be able to intercept voice calls, many contries now have laws that mandate lawful intercept capability for Internet services. An example is the Communications Assistance for Law Enforcement Act (CALEA) [2] in the USA.

There are various providers of Lawful Interception (LI) products, however they tend to be expensive proprietary offerings. In this report we look at the possibility of constructing a LI system based on standard network interfaces and/or high performance hardware from Endace.

A. Endace Hardware

Endace [3] produce a wide array of high performance network monitoring hardware. As well as Gigabit Ethernet, their range includes PDH/TDM, 10 Gigabit Ethernet and SONET/SDH products. The DAG 3.7GF is a 2 Port Gigabit Ethernet monitoring card that offloads the Interrupt handling, exact timestamping and filtering to its on board FPGA so as to minimise use of host resources. These features appear to be beneficial for Lawful Interception purposes. In this report we will compare the performance of the DAG 3.7GF with an Intel Gigabit NIC when using the Service Providers Interception Tool (SPLIT) [4] to intercept traffic.

B. Service Providers Lawful Interception Tool

SPLIT [4] is a prototype LI software program that monitors users' RADIUS traffic to establish identity links to IP addresses in order to capture a user's traffic. This software will be used as it provides a simple example of a real world LI system.

SPLIT monitors RADIUS packets inspecting all 'User-Name' attributes in ACCESS-REQUEST messages looking for users that have been configured to be intercepted. Once a user is found, the ID of the ACCESS-REQUEST is stored in memory. ACCESS-ACCEPT messages are then inspected until an ID matching the stored ID is found. When found, the IP address of the user is extracted from the 'Framed-IP-Address' attribute of the ACCESS-ACCEPT message. A file is then created for the user and traffic is then logged from that IP address.

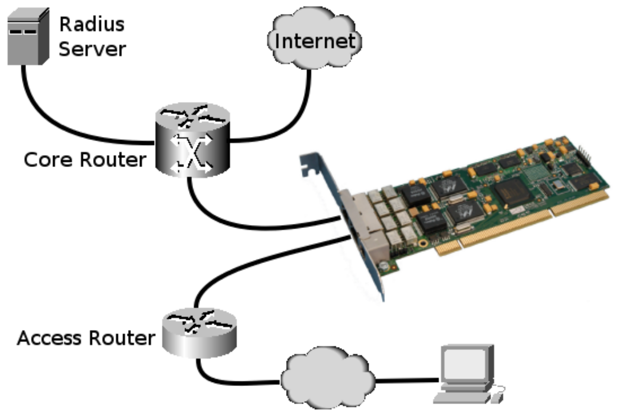

Fig. 1. Endace DAG 3.7GF sits in line passing through all traffic. Traffic capture and identity mapping occur on the same device.

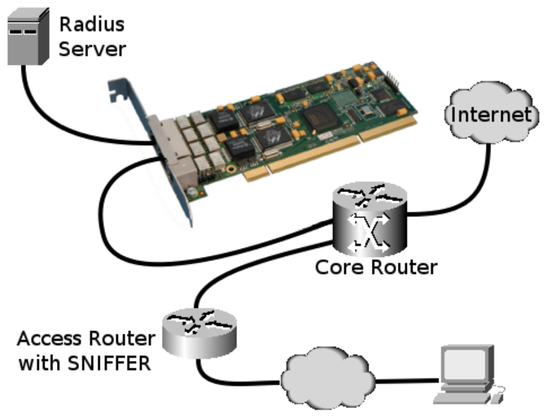

Fig. 2. Endace DAG 3.7GF used solely for linking identity to IP Address

II. Possible Architectures for an LI system

There are a few possible uses for the Endace DAG 3.7GF in a Lawful Interception system. This report will consider three uses.

The simplest application for LI purposes is shown in Figure 1. The DAG 3.7GF is being used in line to both link identity to IP address and capture the relevant IP traffic. This would be quite suitable for a small ISP with only one POP (Point of Presence).

The DAG 3.7GF could also be used for intercepting the radius traffic only (placed close to the radius server). In this configuration, the Endace card would be used purely for linking identity with IP address. Figure 2 shows this configuration. A large ISP with many points of presence could benefit from this model. A single identity linking unit would be configured close to the

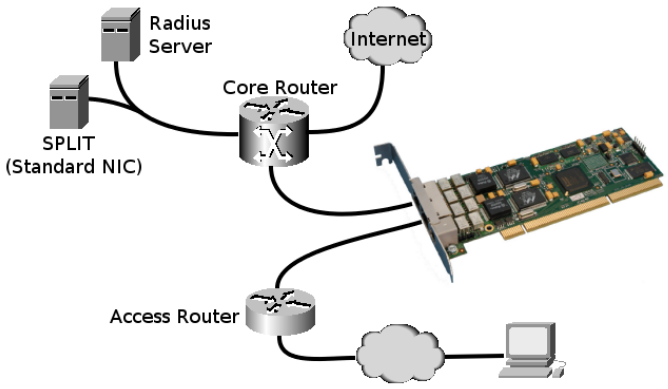

Fig. 3. Endace DAG card used solely for traffic filter and capture (Sniffing)

RADIUS server and multiple slave traffic sniffers could be placed at all the POPs throughout the ISP's network.

The third option would be to use a standard NIC for the identity linking and triggering the machine with the DAG 3.7GF to filter and capture data at the edge of the network.

III. Tools & Testbed

A. Testbed

The test server for traffic capture was a Compaq DL380. It contained dual Pentium III 1.266GHz Processors, 2Gig of RAM and Raid 1 for the operating system and 4 Disk RAID 5 storage for data. This server was used rather than a newer machine because it contained a PCI-X slot. (The DAG 3.7GF needs at least a 3.3volt PCI 2.1 Compliant slot which wasn't in any other machines currently available).

The host used for traffic generation was a standard HP/Compaq desktop machine with a 2.66GHz Pentium IV processor and 1GB RAM. A Separate 80Gig drive was used to store data.

Intel Gigabit NICs were used on both the traffic generator and traffic capture host due to their high performance in comparison to many other brands. FreeBSD was installed on both machines. The control network was connected via on board LAN. The Endace card features a hardware relay so it can be engaged in the circuit or a electrical transparent pass through to effectively remove it. Figure4 shows the hardware configuration.

B. Installation & Configuration of the Endace DAG 3.7GF

The drivers for the DAG 3.7GF come in source code form and need to be compiled according to the instruc-

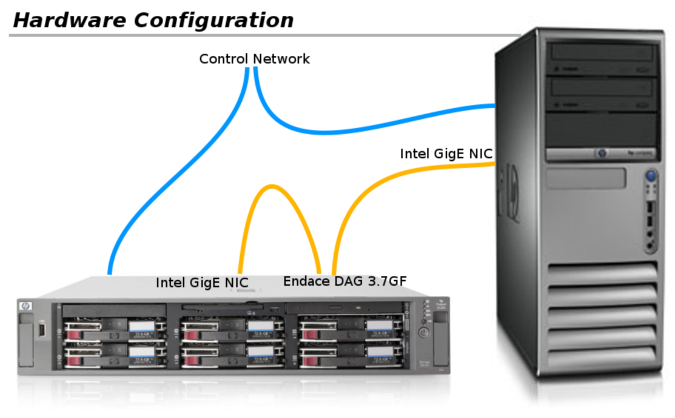

Fig. 4. Traffic generation host and test server containing the Endace DAG 3.7GF and Intel GigE NIC

tion in the User Guide [5]. A few small modifications were made to get the drivers to compile on FreeBSD 7. The DAG 3.7GF was configured as instructed in the User Guide with 128Mb of system memory available.

IV. Test Data

A. Radius Authentication Data Set

The FreeRADIUS [6] server was configured with thirty two thousand users stored in a flat file. A sample user entry is shown below.

```
 Client11 User-Password == Password,
   Pool-Name := main_pool
   Framed-Protocol = PPP,
   Framed-Compression =
                Van-Jacobson-TCP-IP,
   Service-Type = Framed-User,
   Framed-Routing = Broadcast-Listen,
   Framed-Filter-Id = std.ppp,
   Framed-MTU = 1500,
   Framed-IP-Address = 129.241.8.142
```

The radclient utility (part of the FreeRADIUS distribution) generated RADIUS requests. It was used as shown below, where $i is the variable containing a user number.

```
echo "User-Name=User$i,
   Password=Password$i,
   Framed-Protocol=PPP,NAS-Port=$i"
   | radclient 10.0.0.1:1812
   auth mylittlesecret
```

A simple BASH [7] script was used to continually authenticate random users at approximately twenty requests per second. Wireshark [8] was then used to create a PCAP [9] trace of 6000 radius Access-Request/Access-Accept pairs of length 89 and 99 bytes respectively (including IP headers).

B. IP Dataset

This dataset refers to the collection of samples of constant bandwidth traffic with a varying percentage of intercept target traffic. Each sample contained a constant TCP stream from two hosts. The hosts were sending approximately 1500 bytes packets and receiving 60 byte ACK's.

Traffic was generated with the Distributed Internet Traffic Generator (D-ITG) [10] and recorded with tcpdump. Samples lines from a configuration file for D-ITG creating 3% target traffic are shown below..

```
-a 192.168.86.1 -rp 10003 -sp 5603
   -t 60000 -c 1400 -C 485 -T TCP

-a 192.168.86.2 -rp 10003 -sp 5604
   -t 60000 -c 1400 -C 15 -T TCP
```

In the lines above the -c option set the payload size so that it was approximately 1500 bytes after headers were added, and the -C option set the constant packet rate to 485 and 15 PPS respectively. The -t option set the generation to 60000 milliseconds (1 minute) for a total sample length of 30000 packets.

Samples were created with target traffic percentages set at 0.8, 1.6, 2.4, 3.2, and 4.0 respectively. Upon analysis of the samples after creation, actual percentages were 0.93, 1.86, 2.82, 3.71 and 4.62.

C. Internet Traffic Data Set

In order to get an idea of real world performance, a trace from a real Internet link was used as test data. A 15 minute 112Mbps average trace from a Pacific backbone link was obtained from the MAWI Working Group Traffic Archive [11]. It was not immediately useful for testing as the packets had been truncated to 96bytes and there was no radius traffic in the trace which could be used to link identity with IP address.

TCP Rewrite [12] was used with the –fixlen=pad option in order to reconstruct the packets back to their full length to restore the statistical properties of the trace.

A small python script splitcap.py [13] was used to split the truncated trace into 64Meg pieces for analysis. Wireshark was then used to get IP flow statistics on the dump. Therefore IP addresses were extracted which would be suitable for matching (with a radius user). From here a radius configuration was created linking certain users to the IP addresses found in the previous step.

The timestamps from the original trace were compared with those of the newly generated RADIUS traffic and the difference was compensated for with the editcap tool (part of wireshark) which was used to rewrite the timestamps. Mergecap was then used to put all the data into one time aligned PCAP stream.

V. Performance Tuning for the Intel GigE NIC

Since the Intel NIC was losing packets at relatively low rates, various ways to increase performance were considered. It turns out that the BPF kernel subsystem (which PCAP uses when running on FreeBSD) has a variable buffer size which can be controlled by the FreeBSD sysctl command. It has been shown that in general increasing these buffers will decrease packet loss. [14] [15] The standard values were as follows:

```
net.bpf.bufsize: 40964
net.bpf.maxbufsize: 524288
net.bpf.maxinsns: 512
```

In the hope that a larger buffer size would reduce the packet loss, the following commands were used to increase the buffer size to 256KB for the RADIUS inspection test.

```
sysctl -w net.bpf.bufsize=262104
sysctl -w net.bpf.maxbufsize=10485760
```

With the high number of small packets (testing radius traffic only) the increased buffer size slightly degraded performance. This was puzzling however it has been suggested that for best performance, the PCAP buffer size should not exceed 80% of the CPU cache [16]. Since old hardware was being used with only 256KB cache, this seems a plausible explanation.

For later IP tests 512KB PCAP buffer size improved performance, however not all tests have yet been fully repeated with a larger buffer. Casual observation suggests that enlarging these buffers will increase the performance of the Intel NIC, but will not alter the trends that are presented here.

From preliminary testing with varied buffer sizes, it seems that when there is no chance of the buffer overflowing (with small radius packets) having a buffer larger than the system cache does degrade performance slightly, but if a small buffer cannot be used without packet loss (when capturing general IP traffic with large packet sizes), a much larger buffer gives better results.

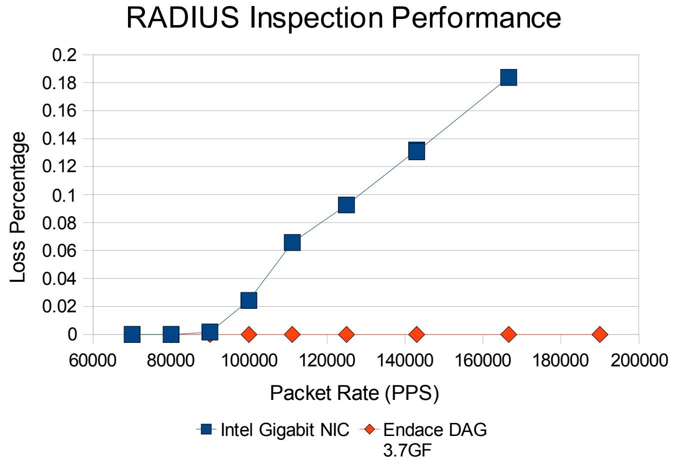

Fig. 5. Standard Intel GigE NIC struggles over about 80k PPS

VI. Performance of RADIUS traffic inspection

For the first round of tests, the RADIUS Dataset mentioned earlier was replayed to the test server at varying rates. The SPLIT software on the test server was configured so that all Radius packets were inspected but none triggered IP capture (No username matched with those that were to be intercepted) in order to ascertain the raw rate at which RADIUS packets could be monitored. This test is applicable to a situation where SPLIT is placed near the RADIUS server for linking identity and remotely triggered sniffers are used to record user traffic. Figure 5 shows the results obtained.

The Endace card outperformed the standard NIC in this configuration, not dropping even one packet at any of the tested rates. In comparison, the NIC performance degraded after a certain threshold. However, it is worth noting that the NIC may be able to be tuned to improve its performance. In this application though, this may not pose a significant problem since it is highly unlikely that a packet rate of more than 80k would need to be processed. A 80k PPS throughput equates to 40k radius requests per second. In reality RADIUS server performance peaks at around 2k requests per second [17], indicating that a standard NIC is more than adequate for the task of inspecting RADIUS packets.

VII. Performance of Identity Linking and Capture Provisioning

For the second round of tests SPLIT was configured to match set percentages of users and was again tested

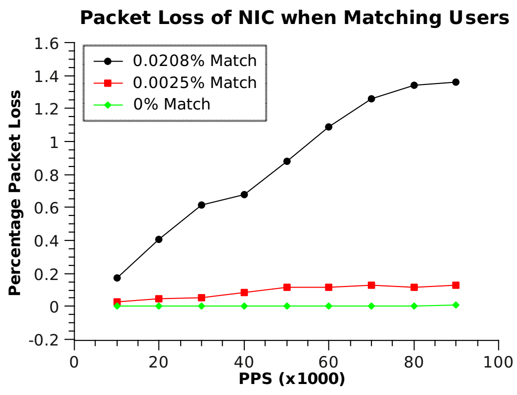

Fig. 6. Packets are dropped even at low rates when using the NIC.

at varying speeds. Figure 6 shows the results. The DAG 3.7GF did not drop any packets. Despite the relatively low rate of 10000 PPS, when using the NIC almost 0.2% of packets are lost when matching 0.021% of users (2.1 users per second). A larger PCAP buffer will likely increase performance. In situations where only one or two users are provisioned for interception at any one time, this would not be a problem because there would not be new users constantly being provisioned, however, the fact that this is happening highlights a lack of scalability in the design of the software. We have seen in the previous test that the NIC can easily capture at this rate without dropping packets and neither the processors or I/O subsystems were showing signs of stress. Currently there is a fair amount of overhead when a match is found. In the same thread that receives captured packets the following tasks happen when packets are found:

- Find user entry in memory
- Add IP address to entry
- Open new file for dumping packets
- Regenerate PCAP filter string
- Apply new filter string to PCAP

As these performance tests show, these tasks should be separated from the main thread which is processing packets so initialising a new user does not pose a threat to the reliability of packet capture.

VIII. Performance of IP Capture

For these tests the IP data set mentioned previously was used. As in the other tests, the DAG card dropped no packets at all. The performance of the NIC is shown in figure 7. It can be seen that as long as the percentage of target traffic is under 1% the NIC is quite adequate for IP capture to around the 100000 PPS second barrier. With these packet sizes, this equates to over 500Mbit/sec. If

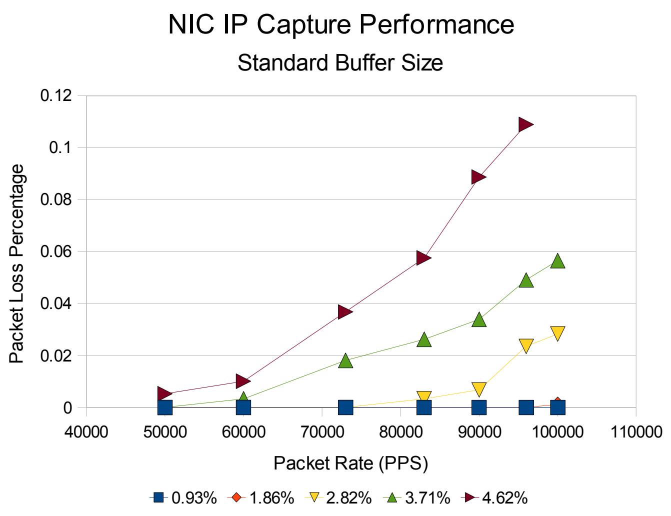

Fig. 7. Packet loss of NIC at various target traffic percentages

target traffic is 2.5% of the overall traffic, the NIC can still reliably capture at 70000 PPS. (over 400Mbit/sec) This suggests that using a NIC for Lawful Interception is definitely a possibility on lower bandwidth links. With an increased buffer size, preliminary testing indicates the NIC will perform even better.

IX. Performance of Internet Traffic Capture

Preliminary testing was undertaking using the Internet traffic dataset. Ten hosts were selected as targest for capture. Collectively these hosts' traffic totalled 2.5 gigabytes of the 14 gigabyte dataset. During testing, the Endace card captured all the target traffic successfully whereas when using the NIC, 1.8% of the data was lost. While results confirmed the general trends shown from other tests, the burstiness of this traffic needs to be investigated before any definitive conclusions can be made.

X. Comments

In all scenarios tested here, the Endace DAG performed excellently suggesting that it would do a fine job in any of the three architectures presented. In lower bandwidth environments, especially where the number of active intercepts are limited, a NIC based solution may be satisfactory. The advantages of the DAG card, namely the on board processing, would be most evident in larger environments where multiple analysis tasks are to be carried out on the same hardware. The NIC is adequate for IP capture at low rates and raw packet capture performance indicates that user matching should also have adequate performance once SPLIT has been optimised with a few more threads.

XI. Conclusion

In these experiments the Endace DAG 3.7GF far outperformed the standard Intel NIC in all tests. Nevertheless, SPLIT still requires significant optimisations before we can confidently specify exact performance limits of the hardware. In addition, further investigation of operating system tuning is required before optimal performance will be obtained from the Intel network card. Despite this, the Endace DAG 3.7GF can be recommended for LI purposes in high bandwidth environments where utmost reliability is required or where there is a desire to perform other forms of network monitoring and analysis on the same machine. An Intel NIC is appropriate for LI purposes in lower bandwidth environments, but further work is required to establish its exact limits.

References

[1] P. Branch, "Lawful Interception of the Internet," CAIA, Tech. Rep. 030306A, March 2003. [Online]. Available: http://caia.swin.edu.au/reports/030606A/CAIA-TR-030606A.pdf

[2] Federal Communications Commission, "Communications Assistance for Law Enforcement Act (CALEA)."

[3] "Endace." [Online]. Available: http://www.endace.com

[4] Adres Rojas, "Service Providers Lawful Interception Tool." [Online]. Available: http://caia.swin.edu.au/reports/040220B/

[5] Endace, "Endace DAG 3.7G User Guide."

[6] "FreeRADIUS radius server." [Online]. Available: http://www.freeradius.org

[7] "GNU BASH (Bourne Again Shell)." [Online]. Available: http://www.gnu.org/software/bash/

[8] "Wireshark Network Protocol Analyser." [Online]. Available: http://www.wireshark.org

[9] "PCAP packet capture library." [Online]. Available: http://www.tcpdump.org/pcap3_man.html

[10] Dipartimento di Informatica e Sistemistica, Universita' degli Studi di Napoli "Federico II" (Italy)v, "Distributed Internet Traffic Generator." [Online]. Available: http://www.grid.unina.it/software/ITG/documentation.php

[11] MAWI Working Group Traffic Archive, "Samplepoint F Traffic Trace - Tue Jan 9 19:15:01 2007 - 15min," January 2007. [Online]. Available: http://tracer.csl.sony.co.jp/mawi/samplepoint-F/20070109/200701091915.%html

[12] Aaron Turner, "TCP Rewrite Online Manual." [Online]. Available: http://tcpreplay.synfin.net/trac/wiki/tcprewrite#tcprewrite

[13] Nelson Minar, "Splitpcap: Simple utility for splitting up Pcap Files." [Online]. Available: http://www.somebits.com/weblog/tech/splitpcap.html

[14] Intelligent Networks, Technical University of Berlin, "High Performance Traffic Capture," 2006. [Online]. Available: http://www.net.t-labs.tu-berlin.de/research/hppc/

[15] J. But, J. Bussiere, "Improving NetSniff Capture Performance on FreeBSD by Increasing the PCAP Capture Buffer Size," October 2005. [Online]. Available: http://caia.swin.edu.au/reports/051027A/CAIA-TR-051027A.pdf

[16] R. B. Riddick, "Proper usage of kernel variables," July 2006. [Online]. Available: http://lists.freebsd.org/mailman/htdig/freebsd-performance/2006-July/0%02134.html

[17] Frank Robinson, "Review: Enterprise RADIUS Servers," June 2004. [Online]. Available: http://www.networkcomputing.com/channels/storageandservers/showArticle%.jhtml?articleId=21402172&pgno=8