Overhead Issues for Local Access Points in IPsec enabled VPNs

John Ronan, Paul Malone, Mícheál Ó Foghlú
Telecommunications Software Systems Group (TSSG)
Waterford Institute of Technology, Ireland.
Email: {jronan, pmalone, mofoghlu}@tssg.org

Abstract

Virtual Private Networks (VPNs) use the Internet or other network service as a backbone to provide a secure connection across a potentially hostile WAN. Such security guarantees provide the motivation for VPN deployment. This security does, however, come at a performance cost brought about by the increased processing overhead. This paper presents an investigation into these overheads. In particular, this investigation will consider the server side overhead for VPN deployments and seek to establish a relationship between this overhead and the number of clients being serviced.

***Index Terms*—Communication System Security, Cryptography.**

1. Introduction

The results of this work comes from several different VPN scenarios which have been tested, measured and analysed. The tests were performed on IPv4 and IPv6 networks and results were collected for several client enumerations in both IPv4 and IPv6 control scenarios in addition to the IPv6 enciphered scenarios. Two different Linux kernel versions were used, firstly a *vanilla* or stock kernel[1] and secondly the USAGI[2] kernel which replaces the entire IPv6 stack with its own, more standards conformant version.

The results will demonstrate a relationship between the number of clients connected to a Local Access Point [3] (LAP – essentially a 'smart' router which sits between the access network and the core network) and the load placed upon the LAP. From this it should be possible to define limits to the number of clients that each LAP is capable of servicing while guaranteeing QoS requirements.

This work has evolved from an investigation within the TORRENT IST [4] project where it was deemed desirable to offer a service which consisted of a secure communications channel between a Residential Gateway (RG) and a LAP. In time, this led to an evaluation of the performance implications of using IP security (IPsec)[5] to achieve this goal. This, in turn, brought about a more detailed investigation as it became apparent that there were scalability issues involved. The results of this work will feed directly into the decision making process of the Agent Based SRM (Service Resource Management) system in TORRENT.

2. Context

2.1 TORRENT Overview

Among the expected outputs of the IST supported TORRENT project is a testbed providing for residential, multi-service access networks. This testbed (Figure 1) will allow the project to demonstrate the benefit of intelligent control, both for the customer and for the network operators and service providers.

An important additional need is to optimise the bandwidth utilisation in existing access and core networks, while at the same time meeting a user's requirements in an optimal

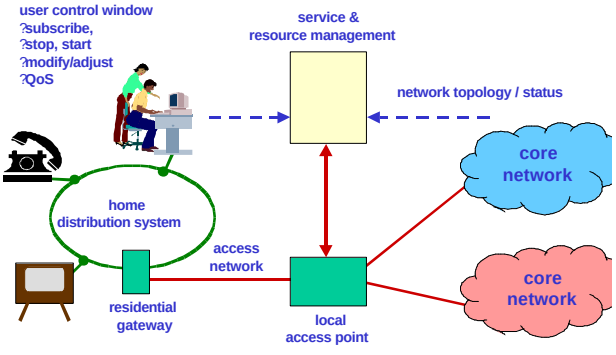

Figure 1 - TORRENT Architecture

manner. These requirements include Quality of Service (QoS), security, cost, and availability.

2.2 Motivation for IPsec deployment

It was proposed early in the TORRENT project lifecycle to integrate IPv6 as the transport protocol and IPsec as a service for securing the data between the RG and the LAP. It was understood that there would be performance implications arising from this, which were not quantified at the time. The

work presented here is a result of a desire within TORRENT to gain an understanding of these performance implications.

3. IPsec Protocol Suite

IPsec is the security architecture for the Internet Protocol (IP). This protocol is applicable to both IPv4 and IPv6. The architecture is defined in [5] and addresses the following 4 elements:

A. *Security Protocols*: Authentication Header (AH) [6] and Encapsulating Security Payload (ESP)[7].

B. *Security Associations*: Definition, management and processing.[8]

C. *Key Management*: The Internet Key Exchange (IKE) [8],[9],[10],[11].

D. *Algorithms*: Requirements of the authentication and encryption algorithms.

3.1 Security Protocols

Traffic Security is provided by two security protocols:

- The *Authentication Header* protocol [6] provides connectionless integrity and data origin authentication. There is also an optional anti-replay service available.

- The *Encapsulating Security Payload* protocol [7] potentially provides two types of security service. The first being confidentiality via encryption and limited traffic flow confidentiality. The second type is connectionless integrity, data origin authentication and an anti-replay service.

Either of these protocols can be applied alone or in combination, thus providing the desired level of security. The IPsec security protocols are represented by headers that appear before the IP header in the IP packet.

3.2 Security Associations

The security protocol headers do not contain information pertaining to the cryptographic algorithms and the associated parameters. These representations are achieved through the transmission of a *Special Parameter Index* (SPI). This index combined with the destination IP addresses and the type of protocol header (AH or ESP) determines the parameters of the IPsec processing.

These parameters of a unidirectional security service are represented by a *Security Association* (SA). There are two types of SAs:

- *Transport Mode SA:* This is a security association between two hosts, generally used to secure the traffic of the upper layer protocols.

- *Tunnel Mode SA:* This is a security association in an IP-in-IP tunnel, generally used in connecting to security gateways.

3.3 Key Management

IPsec mandates support for two separate methods of cryptographic key and SA management: manual and automatic.

- *Manual Key Management:* This is the simplest form of key management and involves each IPsec connection to be configured manually on both hosts. While this is suitable in small static situations, it is unsuitable in larger deployment scenarios due to scalability problems.

- *Automatic Key and SA Management:* Larger deployment scenarios call for an Internet-standard, scalable and automated SA and key management protocol. This is provided by *Internet Key Exchange* (IKE). IKE is required to allow for use of anti-replay features of AH and ESP and to facilitate on-demand creation of SAs.

3.4 Algorithms

The IPsec protocol suite does not define the authentication and encryption algorithms used in implementations. These are defined in individual RFCs per algorithm. Algorithms used in these tests were:

DES [12]
AES [13]
HMAC-MD5 [14]

4. The WIT IPv6 IPsec Testbed

To perform the tests required to examine the performance of the various IPsec scenarios, a testbed was set up. All hosts were interconnected using a Cisco 2924 Ethernet switch using their own isolated VLAN.

A logical view of this testbed configuration is shown in Figure 2. This view shows all six test machines configured with IPv4 and IPv6 addresses. From a physical viewpoint, the testbed has at its core a Flextel WebVision 4012 (identical to the TORRENT LAP), which provides 12 multipurpose slots each of which can take processor cards or I/O carrier cards. Four of processor blades acted as hosts for the testbed. Each of these processor blades was equipped with dual Pentium III 850Mhz processors, 512 MB RAM and an Intel Ethernet Pro 100 network card integrated onto the motherboard. The other two hosts used in the testbed consist of two Dell PIII 500Mhz desktop machines with 100Mbit 3Com 3c905 Network Cards.

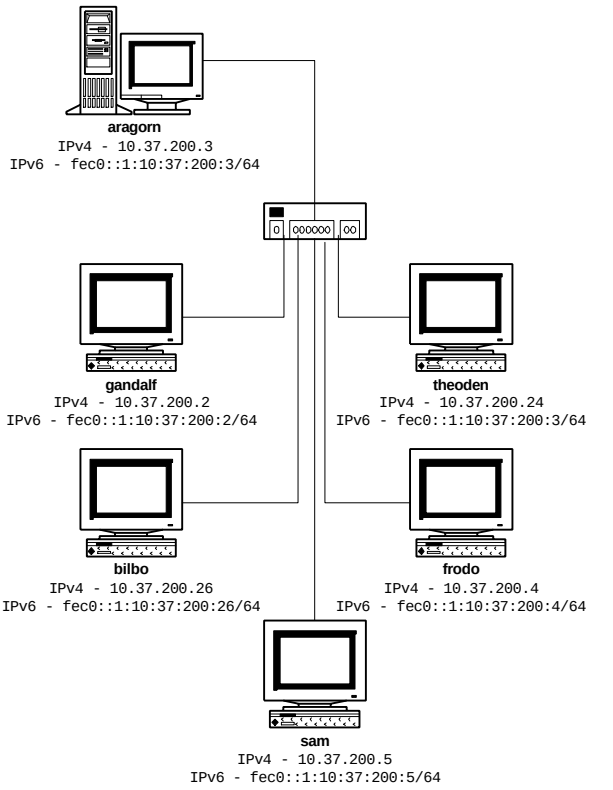

Figure 2 - Testbed configuration

4.1 Software

Each host was configured with the following software:

Operating System:
SuSE Linux 8.0 [15]

Kernel Version:
'Vanilla' Linux Kernel 2.4.19 [1]
USAGI Linux Kernel 2.4.19 [2]

In order to reduce the variables, we chose the 2.4.19 kernel as it was a relatively recent kernel and the USAGI stable Release 4, dated October 7 2002, which is based on this kernel.

Network performance Benchmarking:
Netperf version 2.2.pl2 [16]

Having searched for tools to do throughput testing that included metrics for CPU utilisation and also had IPv6 support. Netperf seemed to suit our needs after it had been patched with the KAME IPv6 patch [17].

IPsec Software:
The USAGI kernel and all its supporting utilities were compiled and installed as per the USAGI documentation. Pluto, the IKE daemon had to be patched [18] to allow for automatic usage of the AES algorithm as manual keying proved problematic.

5. The Tests and the Test Scenarios

Performance tests were organised as follows:
Host `aragorn` acted as the netperf server. This was invoked using the following command:

```
aragorn:#netserver -6
```

Where the –6 option enables IPv6 performance testing.

Scripts were written which ran Netperf User Datagram Protocol (UDP)[19] and Transmission Control Protocol (TCP) [20] stream tests. Each test was 4 minutes in length and was performed 3 times. The content of these scripts is shown in Listing 1 and Listing 2.

```
#!/bin/sh
#TCP Stream test
time=240

./netperf -H aragorn.tssg.org -t TCP_STREAM -C -c -l $time
./netperf -H aragorn.tssg.org -t TCP_STREAM -C -c -l $time
./netperf -H aragorn.tssg.org -t TCP_STREAM -C -c -l $time
```

Listing 1: Netperf TCP Stream test script

```
#!/bin/sh
#TCP Stream test
time=240

./netperf -H aragorn.tssg.org -t UDP_STREAM -C -c -l $time
./netperf -H aragorn.tssg.org -t UDP_STREAM -C -c -l $time
./netperf -H aragorn.tssg.org -t UDP_STREAM -C -c -l $time
```

Listing 2: Netperf UDP Stream test script

The purpose was to establish relationships between the performance overhead in the server (`aragorn`) and the number of clients being served. With this in mind, the above tests were first run sequentially on 1 client (`gandalf`), then on 2, 3, 4 and finally, all 5 clients.

The test start times were set up on each client using the standard unix job scheduler *cron*. All hosts times were synchronised to a local time-server using the *netdate* utility.

This test set was repeated for each of the following scenarios:

5.1.1 Control Scenario: IPv4

Protocol	IPv4
IPsec	No
Kernel	Vanilla 2.4.19 USAGI 2.4.19
Bandwidth Limited	None

This scenario (IPv4 tests with no IPsec VPN deployed) was

used as a guide to throughput and overhead figures.

5.1.2 Control Scenario: IPv6

Protocol	IPv6
IPsec	No
Kernel	Vanilla 2.4.19 USAGI 2.4.19
Bandwidth Limited	None

This scenario was used as a guide to throughput and overhead for IPv6 with no IPsec VPN deployed.

5.1.3 IPsec Scenario 1: IPv6

Protocol		IPv6
IPsec	*SA*	Transport Mode
	Auth	HMAC-MD5
	Enc	3des-cbc
Kernel		USAGI 2.4.19
Bandwidth Limited		None

This scenario provided results for throughput and overhead for IPv6 tests with IPsec VPNs deployed using the 3des-cbc algorithm for encryption.

5.1.4 IPsec Scenario 2:IPv6

Protocol		IPv6
IPsec	*SA*	Transport Mode
	Auth	HMAC-MD5
	Enc	aes-cbc
Kernel		USAGI 2.4.19
Bandwidth Limited		None

This scenario provided results for throughput and overhead for IPv6 tests with IPsec VPNs deployed using the AES algorithm for encryption.

6. Results

Notes on the results:

The results for IPv4 and IPv6 throughput with no encryption for both *vanilla* and USAGI kernels were extremely close (less than 1%), so for clarity they will not be shown here.

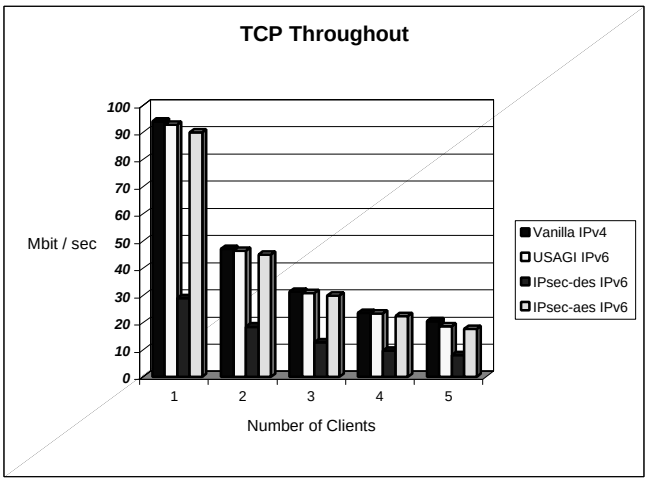

Figure 3 - TCP Throughput

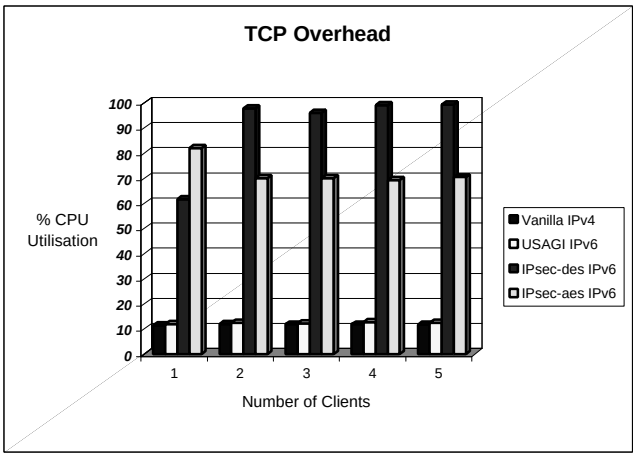

Figure 4 - TCP Overhead

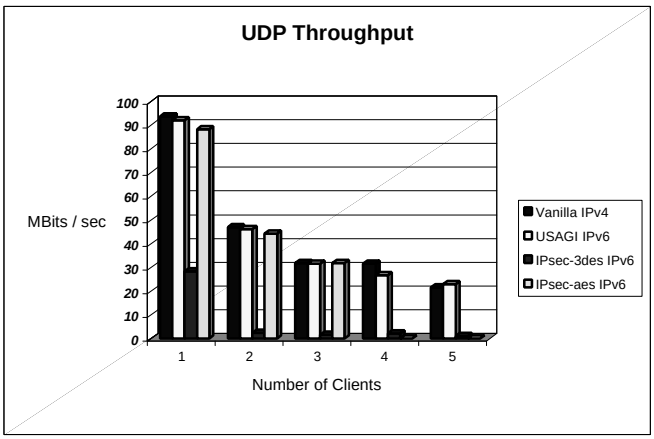

Figure 5 – UDP Throughput

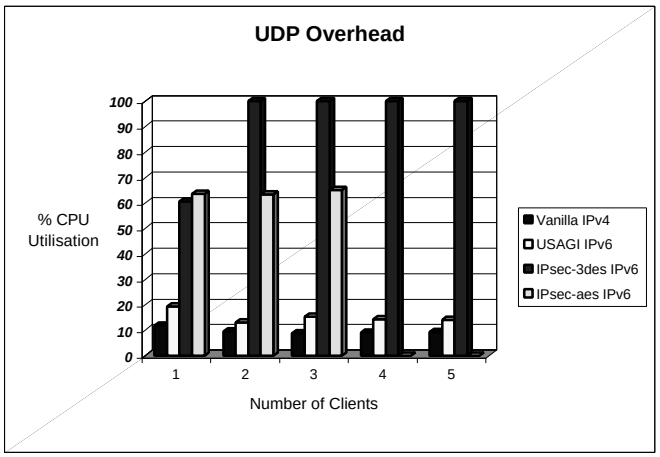

Figure 6 – UDP Overhead

Firstly, before analysing the results, it should be noted that results could not be obtained for 4 and 5 client connections using UDP. These tests were attempted several times and their failure is currently being investigated.

Looking at Figure 3 it can be seen that enciphering the link with AES and using HMAC-MD5 for authentication does not reduce the throughput of the clients appreciably. AES is markedly superior to DES in this case.

Looking at Figure 4 it can be seen that the load induced by the AES algorithm seems to maintain a relatively constant level of 60%, except in the case of a single client. This would seem to indicate that the bottleneck, in this case, is the network card and not the processor.

The results are similar for AES with UDP traffic, but it can be seen from Figure 5 (at least up to and including 3 clients) that DES throughput falls dramatically once more than one client is involved, which indicates that the server is being overworked. This is borne out by Figure 6, which shows the DES CPU utilisation approaching 100% when more than one client is involved.

7. Conclusion

After applying the above tests the conclusions can be drawn that the AES algorithm performs more efficiently than its predecessor, DES, on similar hardware. Hence, IPsec could be deployed as an encryption and authentication service in the TORRENT architecture, without hitting any significant performance bottlenecks, if the algorithms deployed are AES for encryption and HMAC-MD5 for authentication.

References

[1] The Linux Kernel Archives, Available: http://www.kernel.org
[2] USAGI UniverSAl PlayGround for Ipv6) Kernel, Linux IPv6 Development Project. Available: http://www.ipv6.org
[3] E. Scharf, P. Hamer, K. Smparounis, W. Payer, J. Ronan, M. Crotty, “An Intelligent Integrated Approach to Multi-service Residential Access Networks”, Journal of the Communications Network, July-September 2002
[4] TORRENT (Technology for a Realistic End User Access Network Test-bed), IST-2000-25187. http://www.torrent-innovations.org
[5] S. Kent and R. Atkinson, “Security Architecture for the Internet Protocol,” Internet Engineering Task Force, RFC 2401, November 1998.
[6] S. Kent and R. Atkinson, “IP Authentication Header,” Internet Engineering Task Force, RFC 2402, November 1998.
[7] S. Kent and R. Atkinson, “IP Encapsulation Security Payload,” Internet Engineering Task Force, RFC 2406, November 1998.
[8] D. Maughan, M. Schertler, M. Schneider, and J. Turner, "Internet Security Association and Key Management Protocol (ISAKMP)", Internet Engineering Task Force, RFC 2408, November 1998.
[9] H. Orman, "The OAKLEY Key Determination Protocol", Internet Engineering Task Force, RFC 2412, November 1998.
[10] D. Piper, "The Internet IP Security Domain of Interpretation for ISAKMP", Internet Engineering Task Force, RFC 2407, November 1998.
[11] D. Harkins, and D. Carrel, "The Internet Key Exchange (IKE)", Internet Engineering Task Force, RFC 2409, November 1998.
[12] C. Madson and N. Doraswamy, “The ESP DES-CBC Cipher Algorithm with Explicit IV”, Internet Engineering Task Force, RFC 2405, November 1998.
[13] S. Frankel, S. Kelly and R. Glenn, “The AES Cipher Algorithm and Its Use With IPsec”, Internet Engineering Task Force, Internet draft, December 2002.
[14] C. Madson and R Glenn, “The Use of HMAC-MD5 within ESP and AH”, Internet Engineering Task Force, RFC 2403, November 1998.
[15] SuSE, Linux Distribution, http://www.suse.com
[16] Netperf, A Network Performance Benchmark, Available: ftp://ftp.cup.hp.com/dist/networking/benchmarks/netperf/netperf-2.2pl2.tar.gz
[17] The KAME Project, http://www.kame.net/
[18] USAGI IPv6 IPsec AES enhancement patch, Available: http://www002.upp.so-net.ne.jp/h-yamamo/ipv6/usagi/ipsec.html
[19] J. Postel, “User Datagram Protocol”, Internet Engineering Task Force, RFC 768, August 1980
[20] Defence Advanced Research Projects Agency, “Transmission Control Protocol”, Internet Engineering Task Force, RFC 793, September 1981.