Virtual Private Networks

DM71 Project

University Of Southern Denmark - Odense University
IMADA

Anders Porsbo ★*150379*

May 8 2006

Contents

1 Introduction

This report will follow the disposition of the presentation from May 1 2006.

When IPv4 was first proposed, it was asumed that the IP protocol, would be used on trusted networks, so very little effort was put into securing communictions.

Realizing that the internet, which is currently mostly running IPv4, is not the right environment to trust, the IPv6 standard was proposed. IPv6 has in contrast to IPv4 been designed with security and modularity in mind.

When cooporations, small offices, or ordinary people want to communicate securely from one site to another site a private network will definetely provide great protection.

Unfortunately private networks are created by leasing a copper-cable from site A to site B, which is of course expensive. The alternative is to make a "virtual" private network [VPN], passing through the internet.

This is complicated a bit due to the nature of IPv4, and solutions will be discussed in the following. Much information on this topic can be found in the book by Ruixi Yuan [2001].

2 Definition Of A VPN

In order to discuss VPNs we need to define what a VPN is.

It seems there is a trend in VPNs so the definition is a bit sloppy as the term is used in many cases, where it is not appropriate, but especially Hosner [2004] and Ruixi Yuan [2001] seems to take the definition seriously[1].

There is at least three features a VPN must have to be what will be considered a VPN in the following discussion. The following bullets, show what a VPN should be able to do:

- provide a site-to-site connection.
- encapsulate traffic between hosts or networks regardless of application or protocol.
- provide "Private" transfers (see Security Goals on the following page).

It is import that the VPN can create site-to-site connections, this enables VPN gateways to process and redirect the traffic through the VPN, while beeing seemingly transparent to all hosts on the respective networks. Unfortunately many VPN vendors, do not actually sell a VPN solution, but only products enabling "secure" communication between two hosts — this is not a VPN. Of course a proper VPN should have this feature too, but not as the only possibility.

Furthermore a VPN should be able to encapsulate traffic regardless of application or protocol. This is yet again a definition nescessarry because of the VPN

[1] None of the authors seems to be affiliated with cooporations selling VPN solutions

vendors. Often the VPN product will acutally just work as an “secure” application level gateway, which means that some service (not an arbitrary service or protocol), is using for example SSL to authenticate and encrypt traffic, and example is a HTTPS server, which just provides SSL encryption to the HTTP traffic, without providing anything but communications to some application, in this case a webserver.

Finally the traffic passing through the VPN should be private, this means a set of security precautions should be met — these will be precaution elaborated in 3 on the current page.

3 Security Goals

The goal of a VPN is to communicate securely through an insecure network. To meet this goal the following subjects should be considere when choosing/designing a VPN product.

- Confidentiality
- Integrity
- Authentication
- Non-repudiation
- Including:
  - Perfect Forward Secrecy
  - Replay Resistance

Confidentiality means that what we send should be readable by the intended receipent only, so this means encryption should be applied.

Secondly the integrity of the data sent must also be preserved, that is if the message is changed while in transit, we should be able to detect it — this means some sort of message digest should be applied.

Furthermore we want to communicate only with the intended receipent, not someone *pretending* to be the intended receipent, this will be handled by the authentication mechanism.

Non-repudiation, means that it should be possible to be held responsible for traffic/messages sent, without being able to state that you did not sent that information — this is a topic typically solved with digital signatures.

Finally if public key cryptography is used, and the long-term secret key is disclosed, all previous communication keys should not be compromised, this is known as perfect forward secrecy. Secondly it should not be possible to replay a previous communication, without the receipent detecting it — this is often obtained by letting both parties add unpredictable information to the authentication, which the authentication and later communications depend on.

4 Available Technologies

When studying VPNs, it becomes evident that quite a lot of different technologies exist, which may provide the features required according to the VPN definition and security goals. A few of them are mentioned here.

Both Transport Layer Security [TLS] and Internet Protocol Security [IPSec] based VPNs will be elaborated in 5.1 to 5.2 on pages 4–6.

CIPE (Crypto IP Encapsulation)

CIPE [CIPE, 2006a] is a VPN solution using blowfish, but not widely used at this point in time.

This is a quote from CIPE [2006b]:

> It is planned to replace the simple secret key based key exchange process described above by a signed Diffie-Hellman scheme, which would eliminate the need for secret keys.

Unfortunately there is not a proper key exchange mechanism, and the security is dependent on the secrecy of a secret key — so this do not meet the security goals.

PPTP (Point-to-Point Tunneling Protocol)

PPTP is an open Microsoft standard, which for tunneling use the Generel Routing Encapsulation [GRE] , and for encryption the Microsoft Point-to-Point Encryption Protocol [MPPE] is used.

Since the standard is open, and seem to be widely used, PPTP was among the candidates for a more elaborate description, this was later discouraged by Bruce Schneiers security analysis [sch, 1998] of the MS-PPTP implementation.

Schneier states that the PPTP protocol itself is not found to be weak, but the MS-PPTP implementation itself is very poor. The following bullet list is more or less taken directly from sch [1998]:

- Passwords could be sniffed accross the PPTP network.
- Broken encryption scheme.
- Denial Of Service attacks on PPTP servers.
- Schneier quote -”... The mistakes they made are not subtle; they’re “kindergarten cryptographer” mistakes.”
- And it even runs in the kernel to boost performance

These statement taken into account, and the fact that most PCs are running some Microsoft OS (and use MS-PPTP), did discourage further investigation of PPTP as a VPN solution, as many users probably would get a false sense of security.

Other

Additionally these are also occasionally mentioned in VPN reviews L2F (Level 2 Forwarding) and L2TP (Level 2 Tunneling Protocol). It seems a combination of PPP and SSH can be combined into a VPN too.

5 Two VPN Technologies Elaborated

It soon became clear, that IPSec and SSL/TLS would be the most interesting VPN techonologies to study. IPSec [Kent & Atkinson, 1998c] is a standard designed for IPv6 [Deering & Hinden, 1995], and is very widely used. SSL VPNs are not that widely used, but it seems there are many good reasons to use it. These contrasts made it interesting to analyse and compare these two specific solutions.

5.1 IPSec

IPSec has for a long time been the most widely used VPN solution. This is probably because it has been the only possibility for many years [Hosner, 2004]. Furthermore IPSec is a standard described in a RFC [Kent & Atkinson, 1998c], which increases the potential of interoperability, because implementers from different operating systems have the choice to implement it when they know the protocol.

Both Gollmann [1999] and Ruixi Yuan [2001] explain IPSec reasonably detailed, however the RFCs are much more detailed [Deering & Hinden, 1995; Kent & Atkinson, 1998c,a,b; Harkins & Carrel, 1998].

In IPv4, IPSec can be used optionally, but in IPv6 IPSec is mandatory. This however does not mean that all communications with IPv6 and IPSec works as a VPN. The Authentication Header [AH] [Kent & Atkinson, 1998a] will not provide anything but authentication and integrity, and this is probably the most commonly used protocol, when IPSec is used. If however a VPN should be created using IPSec the Encapsulating Security Payload [ESP] [Kent & Atkinson, 1998b] should be used, as this both provide Authentication, Integrity, and Confidentiality.

Furthermore IPSec can be used in both transport- and tunnelmode, and the tunnel mode should be used for VPNs. In figure 5.1 on the following page the encapsulation of a normal IP packet in a tunnel-mode ESP packet is shown.

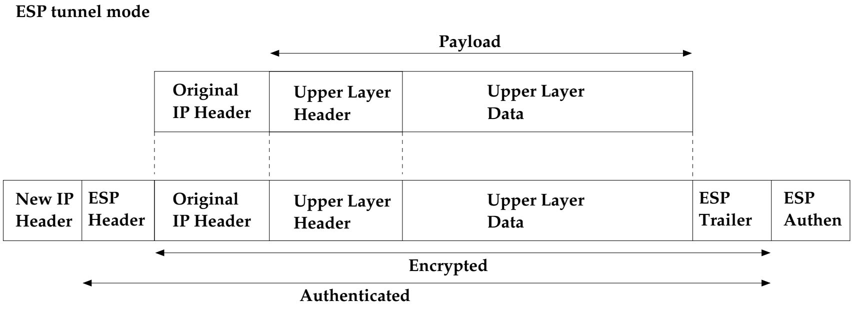

Figure 5.1: ESP Tunnel Mode

Note that the destination address of the original IP packet is in the payload of the ESP packet, when tunnel-mode is used. The new IP Header will get a destination address of the VPN server at the other end of the communiction. This can sometimes give problems with routers and Network Address Translation [NAT], if the routing hardware is not IPSec-aware. Because the headers of the packet are changed when NAT is applied, and thus breaks the message digest used [2].

In figure 5.2 more details of the ESP packet are shown. The part labeled header contains the Security Parameter Index [SPI], which contains values that identifies a Security Association [AS] (negotiated by IKE and guided by the Security Policy Database [SPD][3]).

In the part labeled trailer in figure 5.2 a few important fields exist. The Integrity Check Value [ICV] contains the message digest, and prevents that the packet can be altered without the receipent knowing about it.

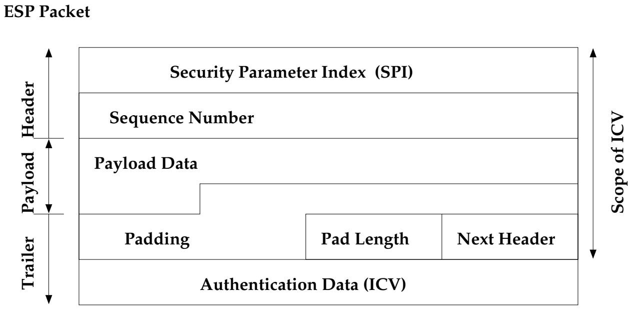

Figure 5.2: ESP Packet

The Payload is the entire IP Packet tunneled through this VPN session. To prevent replay attacks a anti-replay-window can be used with IPsec. Furthermore should the Internet Key Exchange [IKE] protocol provide a secure

[2] could be SHA1, MD5, or any digest negotiated with IKE

[3] Of course the SPD decides what can be suggested and accepted be IKE

establishment of symmetric ciphers to use, and the keys that should be used for *this* session. When using Certificates/Digital Signatures with IKE, it is not possible to repudiate the communication[4].

5.1.1 Evaluating IPSec

So IPSec meet all the security goals listed in 3 on page 2, and this is great.

The two most important pros of IPSec is that it is a open standard, that takes modularity into account. This means that if a weakness is found in any applied cipher or key exchange method, it can simply be turned of in the Security Policy Database. IPSec is also the most widely used VPN technology which increases interoperability — in theory. Unfortunately the complexity of IPSec makes it hard to make different implementation interoperate in practice.

It is often said that complexity is securitys worst enemy [Gollmann, 1999], and it shows in the history of IPSec. Complexity increases the probability of bugs in implementations, and there are regularly reported bugs in implementations for both FreeBSD and Linux[5]. Next problem is the complexity of maintaining the Security Policy Database, and in general configure and administere IPSec securely [Hosner, 2004].

When considering the secure OS Ring Architecture, it is not optimal to run this complex system in the kernel (Ring0), which it in fact does.

Furthermore problems have been reported for windows users trying to use more than one concurrent VPN, and even using another VPN while IPSec is installed.

All these problems suggest a simpler solution should be analysed.

5.2 SSL/TLS VPN

The Secure Socket Layer [SSL] invented by Netscape®, and later updated and standardised through Dierks & Allen [1999] is now known under the name Transport Layer Security [TLS].

Some confusion about SSL and VPNs exist. Mostly due to the fact that IPSec “commercials” tend to state that SSL will only provide a application to application connection, and that SSL VPNs are only application level gateways which depend on application and protocol, often only creating a tunnel on a redirected port. This is often true, but not always. The SSL VPN topic is further obscured by the fact that many SSL VPN vendors actually provide just an application level SSL gateway.

Luckily some bright minds have developed real VPN solutions using the SSL/TLS protocol. These VPNs provide all the features which can be found in IPSec, but are much easier to implement and configure.

One such SSL VPN is the OpenVPN implementation[6], which is free and GPL licensed.

[4] Unless the certificates are disclosed

[5] No attempts have been made to find bugreports on Microsoft IPSec Implementations

[6] Other real SSL VPNs are: Checkpoint, NetScreen, Tinc, OpenSSH v.4.3

OpenVPN utilize the OpenSSL library, and threrefore all cryptographic algorithms are provided through a library that has been used for a long time and hardened during the years — this minimizes the possibility of implementation bugs in the cryptographic algorithms.

Furthermore OpenVPN forces you to use Certificates and can itself work as a Certificate Authority. Keyexchange is done with RSA, and symmetric ciphers and keys are negotiated through the RSA authentication — so both confidentiality, authentication, integrity and non-repudiation is provided by OpenVPN through OpenSSL functions.

TLS consists of the SSL Handshake Protocal (RSA,CA) and the SSL Record Protocol (ciphers and message digests), and the orientation in the network layers, can be seen in figure 5.3.

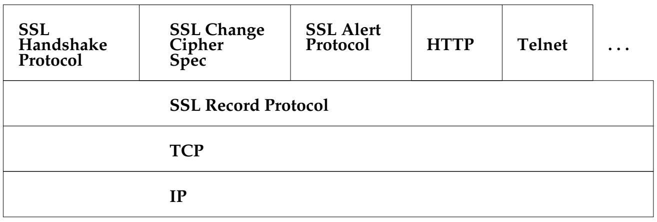

Figure 5.3: TLS in the Network Layers

In figure 5.3 it is also obvious that SSL can rely on the underlying layer TCP, which actually calls for a workaround in OpenVPN. OpenVPN can both tunnel TCP and UDP packets, but most often the tunneled protocols use TCP them selves. It is a known problem that tunneling TCP in TCP can be the source of network congestion, and build up of packets to retransmit [7]. Therefore OpenVPN is most often used in UDP mode, where it relies on the tunneled protocol using TCP. This gives a problem as SSL needs to see a TCP layer beneath it — therefore OpenVPN provides a TCP/UDP multiplexer, so SSL will see a TCP layer beneath it, and OpenVPN can send in UDP mode anyway.

All aspects of the security goals in 3 on page 2 are met with OpenVPN, as they are with OpenSSH which works in a very similar fashion.

5.2.1 Evaluating SSL VPN (OpenVPN)

OpenVPN have been implemented very elegant, and as other SSL VPNs it runs in user-space, which apply with the secure OS Ring Architecture. Even though OpenVPN can be run in user-space it is also possible to downgrade the user running the application. The initial user executing the OpenVPN application, and creating the TUN device need to be root or another privileged user, but in any nix OS the privileges can be downgraded to that of the special unprivileged

[7] The problem arise once the encapsulating packet needs to be retransmitted, because at the time it has been retransmitted, the inner packet is likely to have timed out, and would therefore require to be retransmitted.

user and group “nobody”, which creates yet a layer of security — if an exploit in OpenVPN is found, the damage is limited, because it runs as “nobody”.

Furthermore it utilizes the SSL protocol, which is the most widely deployed security protocol in the world [Hosner, 2004]. Implementations using the SSL protocol often use the long tested SSL libraries, and have no known security related bugs.

The OpenVPN interoperability is simply, as long as the OS can create a virtual point-to-point IP link[8] a port of OpenVPN can be made for that OS. Currently my home network have a FreeBSD OpenVPN server, and Linux and Windows clients connecting to it.

As required by the definition site-to-site connections must be possible, which often result in networks communicating through VPN gateways. If a gateway have a high bandwidth load some hardware support from cryptographic coprocessors might become useful, and this is provided by the OpenSSL library.

Furthermore the configuration of OpenVPN is extremely simple, as can be seen of the configuration files in A on page 10. The two configuration files are the ones currently being used in my home network.

Additionally it is no problem running multiple concurrent VPNs with OpenVPN.

The only con I can think of is that in order to use OpenVPN you need to know it exist, and explicitly install it yourself.

5.3 Conclusion

Both VPN technologies meet both the requirements of the definition and the security goals, so considerations beside security, must decide which VPN solution to prefer. Among these how they work in practice.

After the evaluation of both IPSec and OpenVPN, it is clear that the pros of IPSec is also pros of OpenVPN and that the cons of IPSec negates the pros of IPSec.

Furthermore all the cons of IPSec is handled well by OpenVPN. It should not be hard to realize that it makes sense to prefer OpenVPN over IPSec.

Additionally OpenVPN provides hardware support out of the box, where IPSec on the contrary does not specify how to make hardware support Kent & Atkinson [1998c], it is however stated that it is possible and decissions regarding hardware support will be postponed to the future.

6 Deploying OpenVPN

Deploying OpenVPN is very easy, just download the program from the homepage [OpenVPN, 2006], or install it through your packet manager.

Generate Diffie-Hellmann parameters, public/private keypairs, and a certificate.

Use the configuration files in appedix A on page 10.

[8] Often known as a tun device.

Below is shown my routing table before OpenVPN is started.

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.0.0     0.0.0.0         255.255.255.0   U         0 0          0 eth0
0.0.0.0         192.168.0.1     0.0.0.0         UG        0 0          0 eth0
```

When OpenVPN has setup the VPN, my routing table has changed to:

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
10.8.0.5        0.0.0.0         255.255.255.255 UH        0 0          0 tun0
10.8.0.1        10.8.0.5        255.255.255.255 UGH       0 0          0 tun0
10.10.0.0       10.8.0.5        255.255.255.0   UG        0 0          0 tun0
192.168.0.0     0.0.0.0         255.255.255.0   U         0 0          0 eth0
0.0.0.0         192.168.0.1     0.0.0.0         UG        0 0          0 eth0
```

The 10.8.0/24 network is the real VPN going through the TUN, and the extra route for the 10.10.0/24 network passes through the TUN as well. The 10.10.0/24 route makes my entire home network reachable through the VPN, and I may use any protocol and application to computers behind the FreeBSD OpenVPN gateway[9] at home.

This gives me the ability to run a virtual desktop with VNC just by this command:

```
vncviewer 10.10.0.1:1
```

A windows share can be mounted just by:

```
mount /mnt/platon_all
```

The mount command will look in the /etc/fstab file and see the following:

```
//10.10.0.1/all /mnt/platon_all  smbfs username=stylie,noauto,uid=1000,rw,fmask=700 0 0
```

Furthermore it is worth noting that OpenVPN can be used as encryption for a wireless network, just by substituting the line:

```
push "route 10.10.0.0 255.255.255.0"
```

with this one:

```
push "redirect-gateway"
```

The redirect-gateway line, will make the client change it's default route to pass through the VPN, hence all traffic is “secured”.

[9] Which have a Hi/fn 7955 security accelerator chip (cryptogarphic coprocessor)

A OpenVPN Configuration

A.1 Client

```
client
dev tun
proto udp
remote www.the_domain_name.dk 1194
resolv-retry infinite
nobind
tun-mtu 1500
ca ca.crt
cert archimedes.crt
key archimedes.key
ns-cert-type server
cipher AES-256-CBC
comp-lzo
verb 4
ping 10
ping-restart 60
```

A.2 Server

```
port 1194
proto udp
dev tun
ca ca.crt
cert platon.crt
key platon.key  # This file should be kept secret
dh dh2048.pem
server 10.8.0.0 255.255.255.0
ifconfig-pool-persist ipp.txt
push "route 10.10.0.0 255.255.255.0" # add this network to clients route table
;push "redirect-gateway" # change default route on clients (AccessPoint)
push "dhcp-option DNS 10.8.0.1"
push "dhcp-option WINS 10.8.0.1"
keepalive 10 120
cipher AES-256-CBC # BF-CBC AES-128-CBC DES-EDE3-CBC
comp-lzo # Enable compression on the VPN link.
verb 4
```

References

(1998). www.schneier.com/pptp-faq.html.

CIPE (2006a). http://sites.inka.de/ bigred/devel/cipe.html.

CIPE (2006b). http://sites.inka.de/bigred/devel/cipe-protocol.txt.

Deering, S. & Hinden, R. (1995). Internet Protocol, Version 6 (IPv6) Specification. RFC 1883 (Proposed Standard), obsoleted by RFC 2460.

Dierks, T. & Allen, C. (1999). The TLS Protocol Version 1.0. RFC 2246 (Proposed Standard), obsoleted by RFC 4346, updated by RFC 3546.

Gollmann, D. (1999). *Computer security*. John Wiley & Sons, Inc., New York, NY, USA.

Harkins, D. & Carrel, D. (1998). The Internet Key Exchange (IKE). RFC 2409 (Proposed Standard), obsoleted by RFC 4306, updated by RFC 4109.

Hosner, C. (2004). Openvpn and the ssl vpn revolution.

Kent, S. & Atkinson, R. (1998a). IP Authentication Header. RFC 2402 (Proposed Standard), obsoleted by RFCs 4302, 4305.

Kent, S. & Atkinson, R. (1998b). IP Encapsulating Security Payload (ESP). RFC 2406 (Proposed Standard), obsoleted by RFCs 4303, 4305.

Kent, S. & Atkinson, R. (1998c). Security Architecture for the Internet Protocol. RFC 2401 (Proposed Standard), obsoleted by RFC 4301, updated by RFC 3168.

OpenVPN (2006). www.openvpn.net.

Ruixi Yuan, W.S. (2001). *Virtual Private Networks - Technologies and Solutions*. Addison-Wesley Professional Computing Series, Canada.