



Amman Arab University for Graduated Studies

**DEVELOPMENT AND PERFORMANCE ANALYSIS OF
HYBRID IPSec SSL VPN TECHNOLOGY**

**Prepared by
Mazen Ghazi Juma**

**Supervisor
Prof. Dr. Hilal Al-Bayatti**

**This Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of Master in Computer
Science Presented to the College Deanship of
Computing Studies in Amman Arab University for
Graduated Studies**

**August, 2007
Amman - Jordan**

DELEGATION

I, the undersigned "Mazen Ghazi Juma" authorize hereby Amman Arab University for Graduated Studies to provide copies of this thesis to libraries, institutions, agencies, organizations, individuals and any other parties upon their request.

Name: Mazen Ghazi Juma

Signature:



Date: 11 / 8 / 2007

© Copyright by Amman Arab University for Graduated Studies

APPROVAL

This Thesis titled "Development and Performance Analysis of Hybrid IPsec SSL VPN Technology", has been successfully defended and approved by the examining committee on 11 / 8 / 2007.

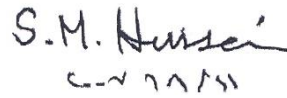
Dr. Khalid Al-Ka'abneh, Chair



Prof. Dr. Hilal Al-Bayatti, Member and Supervisor



Dr. Shakir M. Hussain, Member



Dr. Moayad A. Fadhil, Member



ACKNOWLEDGMENT

My praises and thanks to wisdom donator ALLAH, who provide me the willpower to complete this thesis. Then I would like to thank so much my supervisor Prof. Dr. Hilal Al-Bayatti for his constant encouragement and technical advices to produce this work.

I also thank the examining committee, all my colleagues and relatives for their supporting. Therefore, I present the continuously thanks to all lecturers, administration and staff of Amman Arab University for their help.

Much thanks to my patient mother for her praying and providing supports and special thanks to my beloved wife who kept close to me all the time. My brothers and sisters thank you very much.

There are many friends who have made this work possible that they can not be listed in one page, my thanks to them.

DEDICATION

I dedicate this work to ...

The soul of my father, patient mother, beloved wife, dearest children, impacted brothers and sisters, and supportive friends.

ABSTRACT
DEVELOPMENT AND PERFORMANCE ANALYSIS OF
HYBRID IPSec SSL VPN TECHNOLOGY

Prepared by
Mazen Ghazi Juma

Supervisor
Prof. Dr. Hilal Al-Bayatti

The VPN technology has important role especially in the enterprises environments. Hence, it takes note that the worth issue in the VPN technology is security.

Therefore, several secure VPN technologies are used such as IPSec and SSL VPNs technologies. These technologies have advantages and disadvantages.

This research presents the development of Hybrid IPSec SSL VPN technology as a proposed solution which attainment of the best combinations complementary advantages of IPSec VPN and SSL VPN technologies and elimination their shortcomings.

In this research there is also a representation of the performance analysis for the proposed Hybrid IPSec SSL VPN technology and comparison with separate IPSec VPN and SSL VPN technologies through considered performance measurements included throughput, round trip delay and bandwidth consumption.

The performance measurements benchmark tests results of the proposed hybrid IPsec SSL VPN technology were positively correlated with acceptance levels compared by IPsec VPN and SSL VPN technologies.

The proposed solution decreases the technical administrations which include the deployment and the maintenance of the supporting efforts in networks management. Furthermore, reduces the financial cost that has a high importance in the market.

Table of Content

ACKNOWLEDGMENT.....	IV
ABSTRACT.....	VI
Table of Content	VIII
ACRONYMS AND ABBREVIATIONS.....	XII
LIST OF TABLES.....	XIV
LIST OF FIGURES.....	XV
Chapter One Introduction	1
1.1 Overview	1
1.2 Problem Statement.....	4
1.3 Aims of Thesis.....	5
1.4 Suggested Solution.....	5
1.5 Research Methodology.....	6
1.6 Contributions	6
1.7 Thesis Organization.....	7
Chapter Two Background of IPSec VPN and SSL VPN Technologies	8
2.1 Virtual Private Network	8
2.1.1 Introduction	8
2.1.2 Types of VPN	10
2.1.3 VPN Benefits.....	11
2.1.4 VPN Tunneling and Security	11
2.2 IPSec VPN Technology	12
2.2.1 Introduction	12
2.2.2 IPSec VPN Architecture	14
2.2.3 IPSec VPN Modes	17
2.2.3.1 Transport Mode	17
2.2.3.2 Tunnel Mode	19
2.3 SSL VPN Technology	21

2.3.1 Introduction	21
2.3.2 SSL VPN Architecture	22
2.3.2.1 SSL Session and Connection.....	24
2.3.2.2 The SSL Record Protocol.....	27
2.4 IPsec VPN versus SSL VPN	37
2.4.1 Technical Comparison.....	37
2.4.2 Functional Comparison	38
Chapter Three Related Studies and Works	40
3.1 Introduction	40
3.2 Related VPN Studies and Works	40
3.3 Related IPsec VPN Studies and Works	41
3.4 Related SSL VPN Studies and Works	42
3.5 Related Hybrid IPsec SSL VPNs Studies and Works	43
3.6 Related Ensemble System Mechanism Studies and Works	44
3.7 Related Performance Analysis Studies and Works	46
Chapter Four Development of the Proposed Hybrid	49
IPsec SSL VPNs Technology.....	49
4.1 Introduction	49
4.2 Architecture	50
4.2.1 General Description.....	50
4.2.2 Client Phase Description	51
4.2.3 Proposed Hybrid VPN System Phase Description	54
4.2.4 Server Phase Description	56
4.3 Technical Details	57
4.3.1 General Technical Details.....	57
4.3.2 Technical Details for Client Phase.....	64
4.3.3 Technical Details for the Proposed Hybrid VPN System Phase.....	70
4.3.4 Technical Details for the Server Phase	73
4.4 Algorithms and Implementations.....	74
4.4.1 Introduction	74

4.4.2 Algorithms.....	76
4.4.2.1 Sending Algorithm	76
4.4.2.2 Receiving Algorithm.....	78
4.4.3 Implementations.....	82
4.4.3.1 ML Core Level	82
4.4.3.2 Interface Procedures Level	92
4.4.3.3 Adaptive Applications Level.....	94
4.4.3.4 Network Layers Level.....	96
Chapter Five Performance Analysis of the Proposed Hybrid	99
IPSec SSL VPNs Technology.....	99
5.1 Introduction	99
5.2 Performance Analysis Measurements	100
5.2.1 Throughput Measurement	100
5.2.2 Delay Measurement	100
5.2.3 Bandwidth Consumption Measurement	101
5.3 Benchmarks	101
5.3.1 Instrumentation	101
5.3.1.1 Software and Hardware	101
5.3.1.2 Experimental Scenarios.....	103
5.3.1.3 Variables Discipline.....	106
5.3.2 Throughput Measurement Benchmark.....	107
5.3.3 Delay Measurement Benchmark.....	107
5.3.4 Bandwidth Consumption Measurement Benchmark.....	109
Chapter Six Conclusions and Future Works.....	118
6.1 Introduction	118
6.2 Research Conclusions	118
6.2.1 In General.....	118
6.2.2 Backgrounds, Related Studies and Works	119
6.2.3 Development of Proposed Hybrid VPN System.....	119
6.2.4 Performance Analysis of Proposed Hybrid VPN System	120

6.3 Recommendations for Future Works	120
6.3.1 Development of Proposed Solution Field	121
6.3.2. Performance Analysis of Proposed Solution Field	121
BIBLIOGRAPHY	122
Arabic Summary	126

ACRONYMS AND ABBREVIATIONS

ACK	Acknowledgment
AH	Authentication Header
ATM	Asynchronous Transfer Mode
B2B	Business-to-Business
BBN	Bolt – Beranek - Newman
BITS	Bump in the Stack
CA	Certificate Authority
CCP	Common Case Predicate
DES	Data Encryption Standard
DOI	Domain Of Interpretation
DSA	Digital Signature Algorithm
ESP	Encapsulating Security Payload
FIFO	First In First Out
FTP	File Transfer Protocol
HMAC	Hashed Message Authentication Code
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure
ID	Identifier
IETF	Internet Engineering Task Force
IKE	Internet Key Exchange
Iperf	Illustrative Performance
IPSec	Internet Protocol Security
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
ISAKMP	Internet Security Association Key Management Protocol
ISO	International Standards Organization
ISP	Internet Service Provider
JPL	Jet Propulsion Laboratory
L2F	Layer 2 Forwarding
L2TP	Layer 2 Tunneling Protocol
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
LPE	Logical Programming Environment
MAC	Message Authentication Code
Mbps	Megabits per Second
MD5	Message Digest Version 5
MPLS	Multi-Protocol Label Switching
MPVPN	Multi Path Virtual Private Network

Msec	Millisecond
NAT	Network Address Translation
Netio	Network Input Output
Netperf	Network Performance
NSP	Network Service Provider
OCaml	Objective Caml Programming Language
OSI	Open Systems Interconnection
PC	Personal Computer
PFS	Perfect Forward Secrecy
PPTP	Point-to-Point Tunneling Protocol
POP3	Post Office Protocol Version 3
PSK	Phase Shift Keying
RADIUS	Remote Authentication Dial-In User Service
RDP	Remote Desktop Protocol
RFC	Request For Comments
ROI	Return on Investment
RSA	Rivest – Shamir - Adelman
SA	Security Association
SHA	Secure Hash Algorithm
SSL	Secure Sockets Layer
SYN-ACK	Synchronized Acknowledgment
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
URL	Uniform Resource Locator
VC	Virtual Circuits
VPN	Virtual Private Network
WAN	Wide Area Network

LIST OF TABLES

Table 2.1- The technical comparison between IPSec and SSL VPNs.....	34
Table 2.2 - The functional comparison between IPSec and SSL VPNs ...	35
Table 5.1 - Average throughput for different sized data transfer	94
Table 5.2 - Average delay in both higher and lower UDP traffic	94
Table 5.3 - Bandwidth consumption in different protocol operation phases	9

6

LIST OF FIGURES

Figure 1.1 – Applications for IPSec VPN and SSL VPN	3
Figure 2.1 - The IPSec Roadmap	13
Figure 2.2 -IPSec stack layering	15
Figure 2.3 - Packet format with ESP and AH	17
Figure 2.4 - Packet format with AH and ESP	18
Figure 2.5 - IPSec in tunnel mode	18
Figure 2.6 - IPSec tunneled Mode packet format	19
Figure 2.7 - Nested packet format in tunnel mode	19
Figure 2.8 - The SSL protocol stack	22
Figure 2.9 - Creating a packet under SSL record protocol	27
Figure 2.10 - Handshake protocol action	30

Figure 4.1 – Overview of Hybrid IPsec SSL VPNs Technology System	4
.....	6
Figure 4.2 – Client phase web portal interface shows the different secure connections that delivered to the Hybrid VPN system	47
Figure 4.3 – Hybrid VPN system reverse NAT table operations	49
Figure 4.4 – ISO/OSI model, SSL layer and IPsec layer	51
Figure 4.5 - Packet confusion between SSL layer and IPsec layer	52
Figure 4.6 - General design of the ensemble system mechanism	53
Figure 4.7 – New general design of proposed Hybrid VPN system	54
Figure 4.8 - Technical details for ensemble system mechanism	56
Figure 4.9 – SSL record protocol operation	58
Figure 4.10 - Packet protected by SSL layer forwarded through Hybrid VPN	59
.....	

Figure 4.11 – IPSec cryptography and network security	60
Figure 4.12 - Packet protected by IPSec layer forwarded through Hybrid VPN	61
Figure 4.13 - Protected packets by SSL layer received to Hybrid VPN	62
Figure 4.14 - Protected packets by IPSec layer received to Hybrid VPN	63
Figure 4.15 – IPSec and SSL sessions established via Hybrid VPN	64
Figure 4.16 - Virtual interaction among ML core, interfaces, applications, and network layers	66
Figure 4.17 – Flowchart of sending algorithm	67
Figure 4.18 – Flowchart of receiving algorithm	70
Figure 5.1 - Separate IPSec VPN and SSL VPN Scenario	91
Figure 5.2 - Proposed Hybrid IPSec SSL VPNs System Scenario	92

Figure 5.3 - Throughput measurement test results comparison	96
Figure 5.4 - Delay measurement test results comparison in 1 ms time gap	9
Figure 5.5 - Delay measurement test results comparison in 5 ms time gap	9
Figure 5.6 - Bandwidth consumption measurement test results comparison	10

Chapter One

Introduction

1.1 Overview

The business world today is increasingly dependent on communications which is rapidly becoming a requirement for companies to stay competitive.

This impacted on the information security programs as employees, business partners, vendors and others beyond the traditional enterprise boundaries that require immediate access to a wide array of systems and business information.

These new business communications needs have demonstrated the need for secure communications technologies to meet these goals such as remote Virtual Private Network (VPN) access.

VPN is a connection through a network utilizing encryption to privatize data for transmission between two trusted parties. VPN is a virtual because it can use a public or a private network to transmit data. It is private because it uses encryption that the users control for protection of the data, and it is a network because devices and systems are communicating on a common path [4].

Using of virtual private networks achieves many benefits summarized as follows:

- 1) Lower Costs: combining Internet, Intranet, and Extranet connectivity through the same VPN solution reduces the cost and complexity of managing multiple networks.

- 2) Extended Geographic Connectivity: a VPN connects remote workers to central resources, making it easier than ever to set
- 3) up widely distributed global operations.
- 4) Increased Return on Investment: an effective security solution significantly reduces threats and consequently decreases downtime and lost business.
- 5) Easily scalable: a VPN allows customers to utilize the remote access infrastructure within internet service providers (ISPs) and companies can add a virtually unlimited amount of capacity without adding significant infrastructure [3].

Enterprises are becoming more aware of the opportunities available to meet strategic objectives by leveraging security and technology to deliver services to employees, clients and business partners. The use of VPN technology has matured to a level where it will provide organizations with a solid solution to enable new avenues to drive revenue and reduce operating costs [4].

Several different technologies are used to create security facility to VPNs. Figure (1.1) illustrate the applications of two technologies used to provide remote VPN access which will be focused on this research, Internet Protocol Security (IPSec) VPN and Secure Sockets Layer (SSL) VPN.

IPSec is a suite of protocols that provides security for Internet Protocol (IP) traffic at the network layer. It defines how to provide data integrity, authenticity and confidentiality across a public network like the Internet. It accomplishes these goals through tunneling

, encryption, and authentication, but allows enterprises to select the specific security policy appropriate for their business [26].

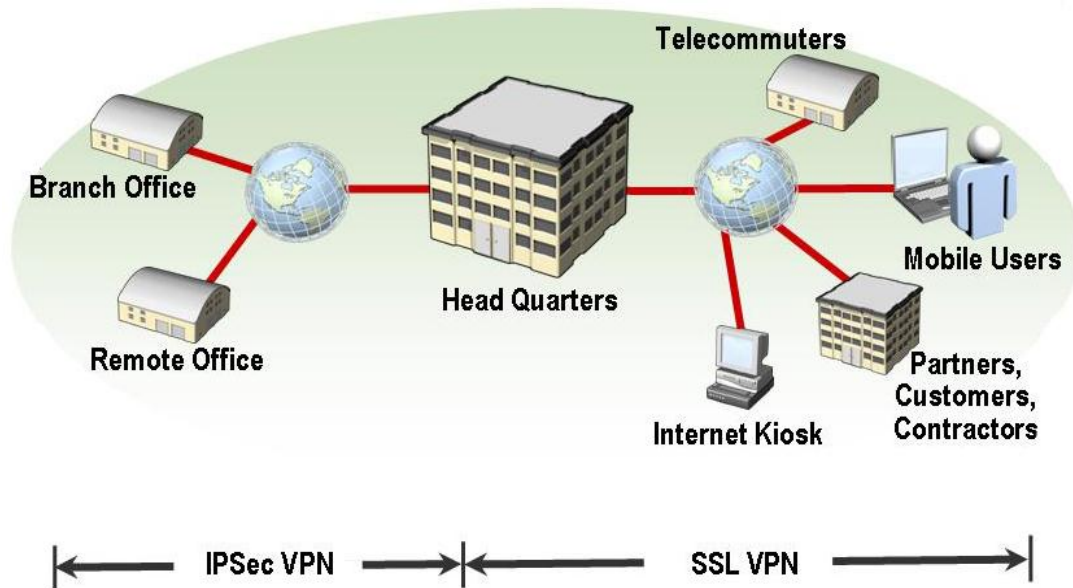


Figure 1.1 – Applications for IPsec VPN and SSL VPN [32]

IPsec VPN revolutionized the way remote workers and business partners connected to an organization by establishing a secure tunnel between a remote worker or business partner and the organization to which they were connecting [25].

There are many advantages and disadvantages for the IPsec VPN among them are:

- 1) Advantages: network to network communication, desk-like network access experience and protocol independent.
- 2) Disadvantages: it does not work through firewalls, difficult to deploy, maintain, and manage, high cost of support and troubleshooting, client IP addresses are visible from the accessed network and network bridging allows network viruses and worms traversal [28]

3) .On the other hand, SSL is a protocol used to secure web based

communications over the internet at the application layer. It uses encryption and authentication to keep communications private between two devices, which are typically a web server and a user machine. Like IPSec, SSL also provides flexibility in allowing enterprises to define the level of security that best meets their needs [33].

SSL VPN allows users to remotely access important enterprise applications, systems, and files from any Internet enabled computer. It enhanced to provide users with secure remote access to internal resources, and promises to improve both mobile user's productivity and convenience [14].

There are many advantages and disadvantages for the SSL VPN among them are:

- 1) Advantages: easy to deploy, clientless and client IP addresses are not visible from the accessed network.
- 2) Disadvantages: works only for web applications that do not use many of the more popular features like ActiveX controls and Java applets. It supports the native corporate applications require creating custom connection and degraded performance of client-server and web applications [3].

1.2 Problem Statement

The purpose of the study is to find out the best combinations of the complementary advantages of the Internet Protocol Security

of the Virtual Private Network, and the Secure Sockets Layer of the Virtual Private Network technologies, and the elimination of their shortcomings.

The combination of these two Virtual Private Network technologies decreases the administration support, financial cost, and optimizes performance element.

Decreasing the administration support element includes the deployment and the maintenance of the technical supporting efforts in networks management.

Reducing the financial cost element has a high importance in the market. However, the optimization of the performance element includes the throughput, delay, and the bandwidth consumption.

1.3 Aims of Thesis

The main aim of this research is the development a proposed Hybrid IPSec SSL VPNs technology as a solution of this research problem statement.

The minor aim of this research is the comparison of the performance analysis of a proposed Hybrid IPSec SSL VPNs technology solution with separate IPSec VPN and SSL VPN technologies solutions through considered performance measurements.

1.4 Suggested Solution

The suggested solution of the problem statement which combine the advantages of IPSec VPN and SSL VPN technologies

and elimination their shortcomings that this research proposed is Hybrid IPSec SSL VPNs technology.

The performance of the suggested solution will be positively correlated with separate IPSec VPN and SSL VPN technologies solutions.

1.5 Research Methodology

This research will use both qualitative and quantitative research methods. Qualitative method will introduced by analysis tools and quantitative method will done through a survey of libraries, collecting data from white papers, books, and websites.

1.6 Contributions

In the research field firstly, the contribution of this research is proposing a unique development model of the Hybrid IPSec SSL VPNs technology and analysis its performance which helps the researchers to test and develop Hybrid VPN technology by using this research resource.

Because the international networks companies whose develop and sell this technology in the market as a black box consider the model design and technical details of this technology as own confidential manner.

In the applied field secondly, the contribution of this research is introducing a new system to benefit from the ensemble system mechanism for participation of Hybrid IPSec SSL VPNs technology development which provide an ideal solution for enterprises whose applications required supporting from both IPSec VPN and SSL VPN technologies at same time.

1.7 Thesis Organization

The thesis organization for the remainder of this research is as follows:

- 1) This chapter overview the VPN technologies, present a problem statement and aims of this thesis, provide the
- 2) suggested solution, research methodology, and end with the contributions of this work.
- 3) Chapter two introduces a deeply background of the VPN technology, focuses on the IPSec VPN and SSL VPN technologies with comparison of the technically and functionality details for each other.
- 4) Chapter three presents the literature review of the related studies and works for this research such as switching protocols and performance implications.
- 5) Chapter four describes the development of the proposed Hybrid IPSec SSL VPNs technology and the presents overall phases of the architecture descriptions, technical details, algorithms and Implementations.
- 6) Chapter five provides a performance analysis of the proposed Hybrid IPSec SSL VPNs technology by introduces the performance measurements, analyzes the testing details, and evaluates the results discussion.
- 7) Chapter six discusses the final conclusions of this research and guidelines for possible areas of the suggested future works.

Chapter Two

Background of IPSec VPN and SSL VPN Technologies

2.1 Virtual Private Network

This section introduces types, benefits, tunneling and security of the virtual private network (VPN).

2.1.1 Introduction

The idea behind VPNs is not new, but VPNs went a step further by offering the opportunity to create dynamic links over different transmission media. In addition to offering a better scalability, a VPN provides a service functionally equivalent to a private network, using the shared resources of a public network.

Typically a VPN uses an existing infrastructure to establish secure communications between trusted associates in a very cost effective way. This infrastructure can be either an IP backbone from a Network Service Provider (NSP), or the whole Internet.

In places, the infrastructure may be frame relay or asynchronous transfer mode (ATM) carrying IP. In other words, a VPN simulates the behavior of dedicated Wide Area Network (WAN) links over leased lines [10].

VPN Definitions

A virtual private network (VPN) is a private communications network often used by enterprises or organizations, to communicate confidentially over a public network.

VPN traffic can be carried over a public networking infrastructure such as the Internet on top of standard protocols, or

over a service provider's private network between the VPN customer and the VPN service provider.

A Bit of History

At the beginning, enterprises leased multiple circuits between geographically dispersed sites to extend their private networks. Physical circuits, leased from carriers, connected pairs of sites to create a point-to-point private communications infrastructure.

Then the circuit-switched private networks have been used for communicating between distant sites. The circuit costs included a one-time setup charge and a periodic recurring charge based on the bandwidth. Because providers supplied fixed bandwidth, customers had exclusive access to leased bandwidth, whether or not they actually used it.

The VPN notion has been around as soon as one started to talk about virtual circuits (VC). The Frame Relay and ATM technology are the most important approaches using the VCs.

These technologies allowed providers to sell less expensive private network services through economies of scale. Both Frame Relay and ATM protocols provided remote site point-to-point connectivity without the need for dedicated bandwidth between sites, since VCs were overlaid on the physical infrastructure.

VPN services were typically not sensitive to distance charges and were much less expensive than leasing dedicated circuits. The use of VPNs enabled enterprises to use the internet infrastructure to deploy their own private networks.

Thus VPNs have become an essential part of business communications between employees, customers, and enterprise partners spread over the Internet, since VPNs enable the creation of secure connections to protect private data and resources when they travel over an un-trusted network like the public internet [8].

2.1.2 Types of VPN

Secure VPNs use cryptographic tunneling protocols to provide the intended confidentiality, sender authentication, and message integrity to achieve privacy. When properly chosen, implemented, and used, such techniques can provide secure communications over unsecured networks; this has been the usually intended purpose for VPN for some years.

Because such choice, implementation, and use are not trivial, there are many insecure VPN schemes available on the market. Secure VPN technologies may also be used to enhance security as a "security overlay" within dedicated networking infrastructures.

The secure VPN protocols include Internet security Protocol (IPSec), Secure Socket Layer and Transport Layer Security (SSL/TLS), Point-to-Point Tunneling Protocol (PPTP), Layer 2 Tunneling Protocol (L2TP), Multi Path Virtual Private Network (MPVPN), Multi-Protocol Label Switching (MPLS), Layer 2 Forwarding (L2F).

In the VPN Quarantine, the client machine at the end of a VPN

could be a threat and a source of attack; this has no connection with VPN design and is usually left to system administration efforts. There are solutions that provide VPN Quarantine services which run end point checks on the remote client [30].

2.1.3 VPN Benefits

A well-designed VPN can provide great benefits for the organization. It can summarize as follows [27]:

- 1) Extend geographic connectivity.
- 2) Reduce operational costs versus traditional WAN.
- 3) Reduce transit time and transportation costs for remote users.
- 4) Provide global networking opportunities.
- 5) Provide telecommuter support.
- 6) Provide broadband networking compatibility.
- 7) Show a good economy of scale.
- 8) Scale well, when used with a public key infrastructure.
- 9) Provide faster Return on Investment (ROI) than traditional carrier leased/owned WAN lines.

2.1.4 VPN Tunneling and Security

Tunneling is the transmission of data through a public network in such a way that routing nodes in the public network are unaware that the transmission is part of a private network. Tunneling is generally done by encapsulating the private network data and protocol information within the public network protocol data so that

the tunneled data is not available to anyone examining the transmitted data frames. Tunneling allows the use of public networks such as the internet, to carry data on behalf of users as though they had access to a private network.

On the other hand, the most important part of a VPN solution is security. The very nature of VPNs raises concerns about potential threats to that data and the impact of data loss. A Virtual Private Network must address all types of security threats by providing

security services in the areas of authentication which is a process of ensuring that a user or system is who the user claims to be. There are many types of authentication mechanisms, but they all use one or more of the following approaches:

- 1) Something the client knows for example a login name, a password, a PIN code.
- 2) Something the client has for example a smart card, a card key.
- 3) Something the client is for example a fingerprint, retinal pattern, iris pattern, hand configuration.

Strong authentication is usually taken to combine at least two authentication components from different above areas [14].

2.2 IPSec VPN Technology

This section presents the overview about IPSec VPN technology, its architecture and modes.

2.2.1 Introduction

The IPSec working group at the IETF has defined 12 Request for Comments (RFC). The RFCs define various aspects of IPSec

Architecture, key management, base protocols, and the mandatory transforms to implement for the base protocols.

Figure (2.1) illustrate the IPsec protocols include Authentication Header (AH), Encapsulating Security Payload (ESP), Internet Key Exchange (IKE), and Internet Security Association and Key Management Protocol (ISAKMP) that define a framework for security association management and cryptographic key establishment for the Internet. This framework consists of defined exchanges and processing guidelines that occur within a given

Domain of Interpretation (DOI).

DOI used to group related protocols using ISAKMP to negotiate security associations. Security protocols sharing a DOI choose security protocol and cryptographic transforms from a common namespace and share key exchange protocol identifiers. They also share a common interpretation of DOI specific payload data content, including the Security Association and Identification payloads.

In order to understand, implement, and use IPsec, it is necessary to understand the relationship among these components. The IPsec roadmap defines how various components of IPsec interact with each other.

IPsec is a suite of protocols and it is important to understand how these protocols interact with each other and how these protocols are tied together to implement the capabilities described

by the IPSec architecture [24].

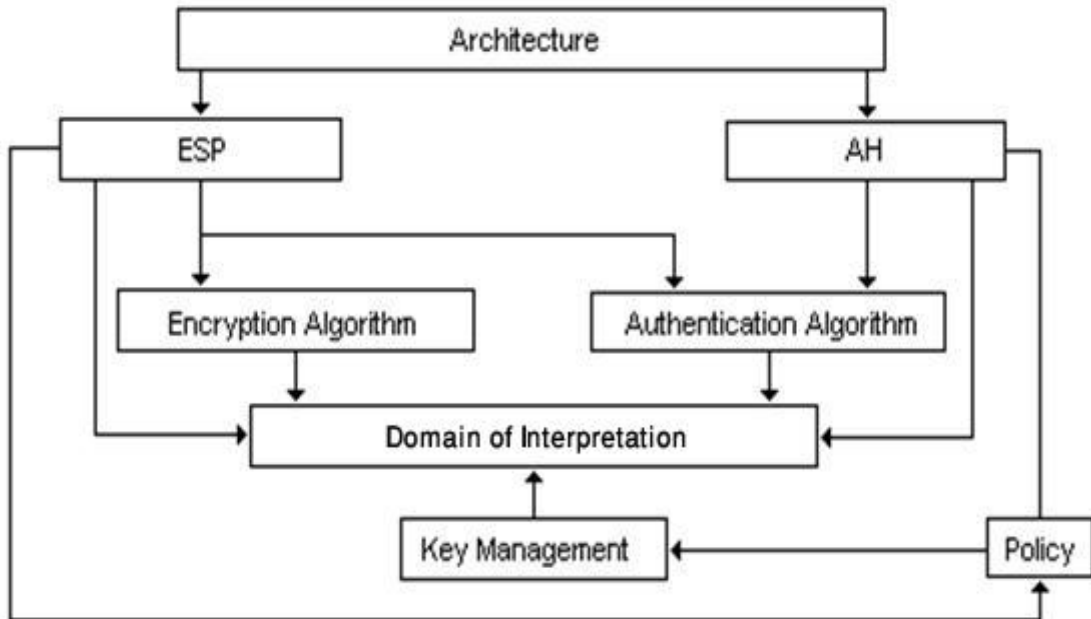


Figure 2.1 - The IPSec Roadmap

2.2.2 IPSec VPN Architecture

IPSec architecture requires the host to provide confidentiality using ESP, and data integrity using either AH or ESP. However, the architecture document does not specify the header formats for these protocols. The architecture discusses the semantics of the IPSec protocols and the issues involved in the interaction among the IPSec protocols and the rest of the Transmission Control Protocol with Internet Protocol (TCP/IP) protocol suite.

The ESP and the AH documents define the protocol, the payload header format, and the services they provide. However, they do not specify the transforms that are used to provide these capabilities. This is because the new transforms can be defined

when the algorithms used by the older transforms are proved to be cryptographically insecure. However, this does not mandate any change to the base protocols.

The transforms define the transformation applied to the data to secure it. This includes the algorithm, the key sizes and how they are derived, the transformation process, and any algorithmic specific information. It is important to be specific about the necessary information so that different implementations can interoperate.

IKE generates keys for the IPSec protocols. IKE is also used to negotiate keys for other protocols that need keys. There are other protocols in the internet that require security services such as data integrity to protect their data.

The payload format of IKE is very generic. It can be used to negotiate keys for any protocol and not necessarily limit itself for IPSec key negotiation. This segregation is achieved by separating the parameters IKE negotiates from the protocol itself. The parameters that are negotiated are documented in a separate document called the IPSec Domain of Interpretation.

An important component that is not yet a standard is policy. Policy is a very important issue because it determines if two entities will be able to communicate with each other and, if so, what transforms to use. It is possible, with improperly defined policies, for two sides to be unable to communicate with each other [27].

OSI Model Integrated

In the host implementation, IPSec may be integrated with the OSI model. It may be implemented as part of the network layer as shown in Figure (2.2). IPSec layer needs the services of the IP layer to construct the IP header. This model is identical to the implementation of other network layer protocols [23].

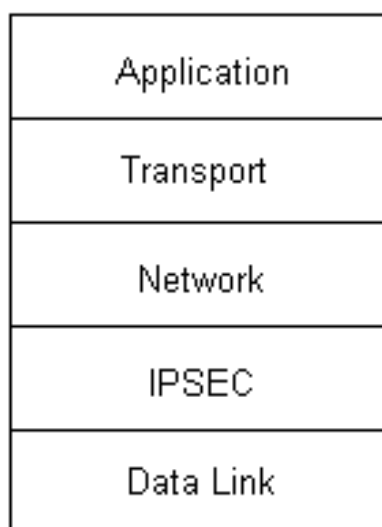


Figure 2.2 -IPSec Stack Layering

Bump in the Stack

For companies providing solutions for VPNs and intranets, OSI model integrated solution has one serious drawback. On the end

hosts, they have to work with the features provided by the OSI model vendors. This may limit their capabilities to provide advanced solutions.

To overcome this limitation, IPSec is inserted between the network and the data link layer. This is commonly referred to as Bump in the Stack (BITS) implementation.

The major issue in this implementation is duplication of effort. It requires implementing most of the features of the network layer, such as fragmentation and route tables. Duplicating functionality leads to undesired complications. It becomes more difficult to handle issues such as fragmentation, and routing. An advantage of BITS implementation is the capability of an implementation to provide a complete solution [17].

2.2.3 IPSec VPN Modes

This section describes how the IPSec protocols, AH and ESP, implement the tunnel and transport modes. There are four possible combinations of modes and protocol: AH in transport mode, AH in tunnel mode, ESP in transport mode, and ESP in tunnel mode. In practice, AH in tunnel mode is not used because it protects the same data that AH in transport mode protects. The AH and ESP header do not change between tunnel or transport mode [23].

2.2.3.1 Transport Mode

In transport mode, AH and ESP protect the transport header. In this mode, AH and ESP intercept the packets flowing from the transport layer into the network layer and provide the configured security.

When security is not enabled, transport layer packets such as TCP and User Datagram Protocol (UDP) flow into the network layer which adds the IP header and calls into the data link layer. When security in transport layer is enabled, the transport layer packets flow into the IPSec component.

The IPSec component is implemented as part of the network layer when intergraded with OSI model. The IPSec component adds the AH, ESP, or both headers, and invokes the part of the network layer that adds the network layer header.

The transport mode of IPSec can be used only when security is desired end to end. As stated earlier, the routers look mostly at the network layer in making routing decisions and the routers do not and should not change anything beyond the network layer header. Inserting transport mode IPSec header for packets flowing through a router is a violation of this rule.

When both AH and ESP are used in transport mode, ESP should be applied first. The reason is obvious. If the transport packet is first protected using AH and then using ESP, the data integrity is applicable only for the transport payload as the ESP header is added later on as shown in Figure (2.3). This is not desirable because the data integrity should be calculated over as much data as possible [5].



Figure 2.3 - Packet format with ESP and AH

If the packet is protected using AH after it is protected using ESP, then the data integrity applies to the ESP payload that contains the transport payload as shown in Figure (2.4).

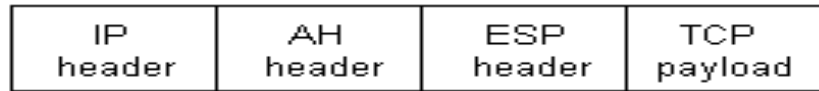


Figure 2.4 - Packet format with AH and ESP

The transport mode for BITS implementation is not as clean, as the ESP and AH headers are inserted after the IP payload is constructed. This implies the BITS implementation has to duplicate the IP functionality because it has to recalculate the IP checksum and fragment the packet if necessary. Many BITS implementations may not support transport mode but support only tunnel mode [17].

2.2.3.2 Tunnel Mode

IPSec in tunnel mode is normally used when the ultimate destination of the packet is different from the security termination point as shown in Figure (2.5) or in case of BITS implementations. The tunnel mode is used in cases when security is provided by a device that did not originate packets as in the case of VPNs or when the packet needs to be secured to a destination that is different from the actual destination.

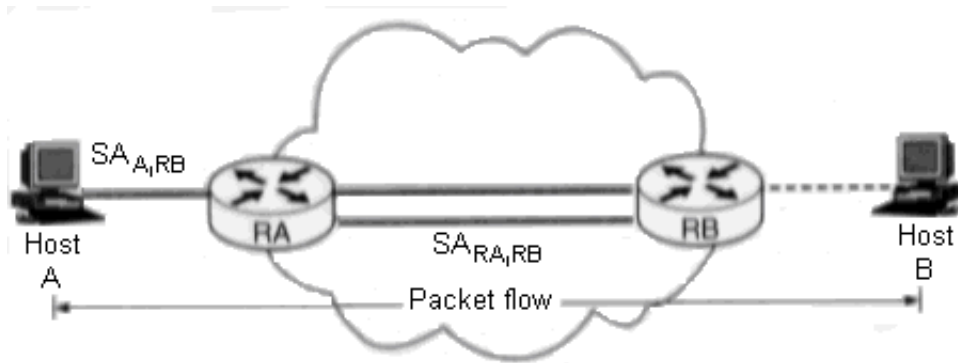


Figure 2.5 - IPsec in tunnel mode

It is also used when a router provides security services for packets it is forwarding. In the case of tunnel mode, IPsec encapsulates an IP packet with IPsec headers and adds an outer IP Header as shown in Figure (2.6).



Figure 2.6 - IPsec tunneled Mode packet format

An IPsec tunneled mode packet has two IP headers, inner and outer. The inner header is constructed by the host and the outer header is added by the device that is providing the security services.

This can be either the host or a router. There is nothing that precludes a host from providing tunneled mode security services end to end. However, in this case there is no advantage to using tunnel mode instead of transport mode. In fact, if the security services are provided end to end, transport mode is better because it does not add an extra IP header.

IPSec defines tunnel mode for both AH and ESP. IPSec also supports nested tunnels. The nested tunneling is where we tunnel a tunneled packet as shown in Figure (2.7).

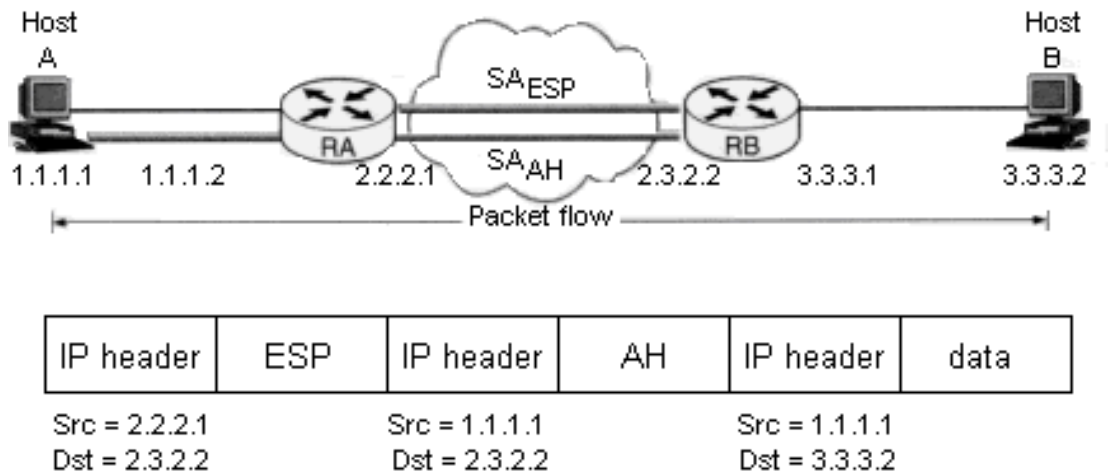


Figure 2.7 - Nested packet format in tunnel mode

In this example, host A is sending a packet to host B. The policy says it has to authenticate itself to router RB. In addition, there is a VPN between the two networks bordered by RA and RB. The outermost header is the tunneled ESP packet. It is carrying a tunneled AH packet.

The tunneled AH packet is carrying the IP packet destined for the host B generated by host A [8].

2.3 SSL VPN Technology

This section presents the overview about SSL VPN technology and its architecture.

2.3.1 Introduction

Security of data in transit over the Internet becomes increasingly necessary because of steadily growing data volume and importance. Nowadays, every user of a public network sends various

types of data, from email to credit card details daily, and he would therefore like them to be protected when in transit over a public network.

To this end, a practical SSL protocol has been adopted for protection of data in transit that encompasses all network services that use TCP/IP to support typical application tasks of communication between servers and clients.

The SSL protocol was originally developed by Netscape, to ensure security of data transported and routed through Hyper Text Transfer Protocol (HTTP), Lightweight Directory Access Protocol (LDAP) or Post Office Protocol 3 (POP3) application layers.

SSL is designed to make use of TCP as a communication layer to provide a reliable end-to-end secure and authenticated connection between two points over a network, for example between the service client and the server.

Although SSL can be used for protection of data in transit in situations related to any network service, it is used mostly in HTTP server and client applications. Today, almost each available HTTP server can support an SSL session, whilst web browsers are provided with SSL enabled client software [21].

2.3.2 SSL VPN Architecture

The main objectives for SSL are:

- 1) Authenticating the client and server to each other:

The SSL protocol supports the use of standard key cryptographic techniques which is public key encryption to authenticate the communicating parties to each other. Though

the most frequent application consists in authenticating the service client on the basis of a certificate, SSL may also use the same methods to authenticate the client.

2) Ensuring data integrity:

During a session, data cannot be either intentionally or accidentally tampered with.

3) Securing data privacy:

Data in transport between the client and the server must be protected from interception and be readable only by the intended recipient. This prerequisite is necessary for both the data associated with the protocol itself securing traffic during negotiations and the application data that is sent during the session itself [24].

SSL is in fact not a single protocol but rather a set of protocols that can additionally be further divided in two layers:

1) The protocol to ensure data security and integrity: this layer is composed of the SSL Record Protocol.

2) The protocols that are designed to establish an SSL connection: three protocols are used in this layer: the SSL Handshake Protocol, the SSL Change Cipher Spec protocol, and the SSL Alert Protocol.

Figure (2.8) illustrate the SSL when uses these protocols to address the task. The SSL record protocol is responsible for data encryption and integrity. It is also used to encapsulate data sent by other SSL protocols. Therefore, it is also involved in the tasks associated with the SSL check data. The other three protocols cover

the areas of session management, cryptographic parameter management and transfer of SSL messages between the client and the server.

Prior to going into a more detailed discussion of the role of individual protocols and their functions, let us describe two fundamental concepts related to the use of SSL [23].

SSL handshake protocol	SSL cipher change protocol	SSL alert protocol	Application Protocol (eg. HTTP)
SSL Record Protocol			
TCP			
IP			

Figure 2.8 - The SSL protocol stack

2.3.2.1 SSL Session and Connection

The concepts as mentioned before are fundamental for a connection between the client and the server, and they also encompass a series of attributes. More details listed as follows [24]:

1) Connection:

This is a logical client-server link, associated with the provision of a suitable type of service. In SSL terms, it must be a peer-to-peer connection with two network nodes.

2) Session:

This is an association between a client and a server that defines a set of parameters such as algorithms used session number etc. An SSL session is created by the Handshake Protocol that allows parameters to be shared among the

connections made between the server and the client, and sessions are used to avoid negotiation of new parameters for each connection. This means that a single session is shared among multiple SSL connections between the client and the server. In theory, it may also be possible that multiple sessions are shared by a single connection, but this feature is not used in practice. The concepts of a SSL session and connection involve several parameters that are used for SSL-enabled communication between the client and the server. During the negotiations of the handshake protocol, the encryption methods are established and a series of parameters of the Session State are subsequently used within the session.

A session state is defined by the following parameters:

- 1) Session identifier: this is an identifier generated by the server to identify a session with a chosen client.
- 2) Peer certificate: X.509 certificate of the peer.
- 3) Compression method: a method used to compress data prior to encryption.
- 4) Algorithm specification termed Cipher Spec: specifies the bulk data encryption algorithm. For example data encryption standard (DES), and the hash algorithm such as MD5, used during the session.
- 5) Master secret: 48-byte data being a secret shared between the client and server.
- 6) "Is Presumable": this is a flag indicating whether the session can be used to initiate new connections.

The SSL connection state is defined by the following parameters:

- 1) Server and client random: random data generated by both the client and server for each connection.
- 2) Server write Message Authentication Code (MAC) secret: the secret key used for data written by the server.
- 3) Client write MAC secret: the secret key used for data written by the client.
- 4) Server write key: the cipher key for data encrypted by the server and decrypted by the client.
- 5) Client write key: the cipher key for data encrypted by the client and decrypted by the server.
- 6) Sequence number: sequence numbers maintained separately by the server for messages transmitted and received during the data session.

The abbreviation MAC used in the above definitions means Message Authentication Code that is used for transmission of data during the SSL session.

The role of MAC will be explained further when discussing the record protocols. A brief description of the terms was necessary to be able to explain the next issues connected with the functioning of the SSL protocol, namely the SSL record protocol [25].

2.3.2.2 The SSL Record Protocol

The SSL record protocol involves using SSL in a secure manner and with message integrity ensured. To this end it is used by upper layer SSL protocols.

The purpose of the SSL record protocol is to take an application message to be transmitted, fragment the data which needs to be sent, encapsulate it with appropriate headers and create an object just called a record, which is encrypted and can be forwarded for sending under the TCP protocol.

The first step in the preparation of transmission of the application data consists in its fragmentation i.e. breaking up the data stream to be transmitted into 16 KB or smaller data fragments followed by the process of their conversion in a record.

These data fragments may be further compressed, although the SSL 3.0 protocol specification includes no compression protocol, thus at present, no data compression is used.

At this moment, creation of the record is started for each data portion by adding a header to it, possible information to complete the required data size and the MAC. The record header that is added to each data portion contains two elementary pieces of information, namely the length of the record and the length of the data block added to the original data.

In the next step, the record data constructed consists of the following elements: primary data, some padding to complete the datagram as required and MAC value.

MAC is responsible for the verification of integrity of the message included in the transmitted record. It is the result of a hash function that follows a specific hash algorithm, for example Message Digest (MD5) or Secure Hash Algorithm (SHA-1). MAC is determined as a result of a hash function that receives the following

data: $MAC = \text{Hash function} [\text{secret key, primary data, padding, sequence number}]$.

A secret key in creation of MAC is either a client write MAC secret or a server write MAC secret respectively, it depends on which party prepares the packet.

After receiving the packet, the receiving party computes its own value of the MAC and compares it with that received. If the two values match, this means that data has not been modified during the transmission over the network.

The length of the MAC obtained in this way depends on the method uses for its computing. Next, the data plus the MAC are encrypted using a preset symmetric encryption algorithm, for example DES or triple DES. Both data and MAC are encrypted [24].

This prepared data is attached with the following header fields:

- 1) Content type: identifies what payload is delivered by the packet to determine which higher protocols are to be used for processing of data included in the packet. The possible values are changed cipher spec, alert, handshake, and application data that refer to the appropriate protocols.

- 2) Major version: establishes the main portion of the protocol version to be used. For example SSL 3.0, the value is 3.
- 3) Minor version: establishes the additional portion of the used version of the protocol. For example SSL 3.0 the value is 0.

With the addition of fields, the process of record preparation is completed. Afterwards, the record is sent to the targeted point. The entire process of preparation of the packet to be sent is illustrated in Figure (2.9) [34].

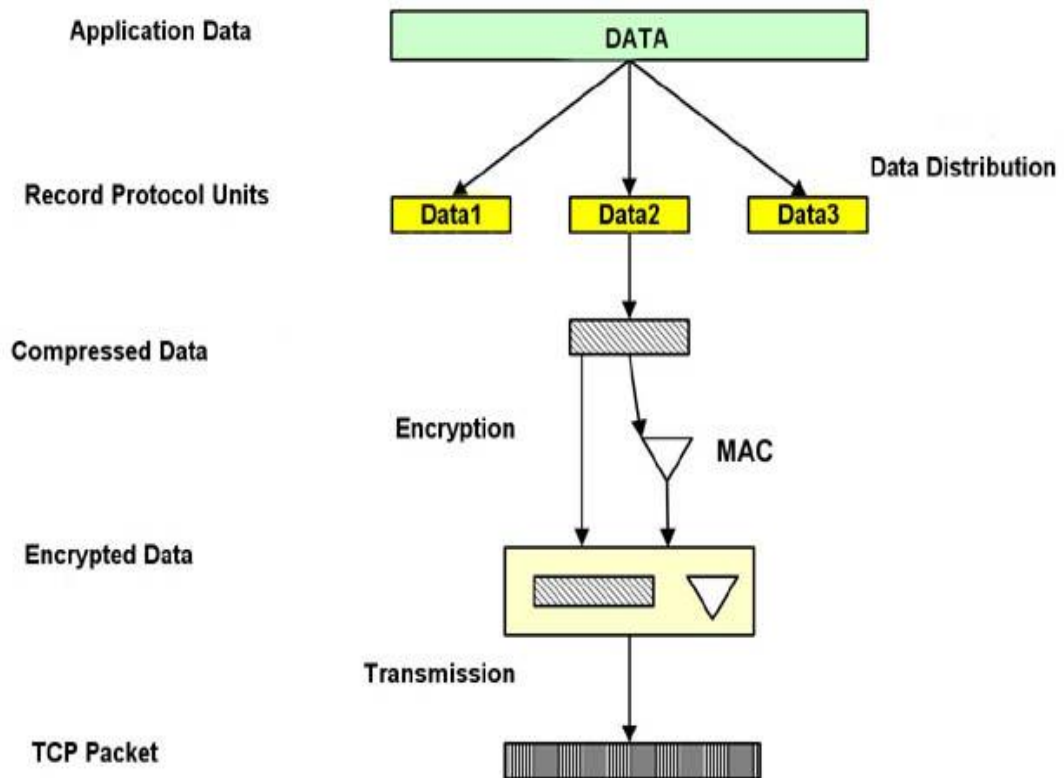


Figure 2.9 - Creating a packet under SSL record protocol

The SSL record protocol is used to transfer any data within a session - both messages and other SSL protocols (for example the handshake protocol), as well as for any application data

The Alert Protocol

The Alert Protocol is used by parties to convey session messages associated with data exchange and functioning of the protocol. Each message in the alert protocol consists of two bytes. The first byte always takes a value "warning", and the second byte always takes a value "fatal", that determines the severity of the message sent.

Sending a message having a "fatal" status by either party will result in an immediate termination of the SSL session. The next byte of the message contains one of the defined error codes, which may occur during an SSL communication session [17].

The Change Cipher Spec protocol

This protocol is the simplest SSL protocol. It consists of a single message that carries the value of one. The sole purpose of this message is to cause the pending session state to be established as a fixed state, which results, for example, in defining the used set of protocols.

This type of message must be sent by the client to the server and vice versa. After exchange of messages, the session state is considered agreed. This message and any other SSL messages are transferred using the SSL record protocol [24].

The handshake protocol

The handshake protocol constitutes the most complex part of the SSL protocol. It is used to initiate a session between the server and the client. Within the message of this protocol, various components such as algorithms and keys used for data encryption

are negotiated. Due to this protocol, it is possible to authenticate the parties to each other and negotiate appropriate parameters of the session between them.

Figure (2.10) illustrate the four divided phases which separated with horizontal broken lines. During the first phase, a logical connection must be initiated between the client and the server followed by the negotiation on the connection parameters.

The client sends hello message to the server containing data such as:

- 1) Version: the highest SSL version supported by the client.
- 2) Random: data consisting of a 32 bit timestamp and 28 bytes of randomly generated data. This data is used to protect the key exchange session between the parties of the connection.
- 3) Session ID: a number that defines the session identifier. A nonzero value of this field indicates that the client wishes to update the parameters of an existing connection or establish a new connection on this session. A zero value in this field indicates that the client wishes to establish a new connection.
- 4) Cipher Suite: a list of encryption algorithms and key exchange method supported by the client [27].

In response, the server sends hello message to the client containing the same set of fields as the client message, placing the following data:

- 1) Version: the lowest version number of the SSL protocol supported by the server.

- 2) Random data: the same fashion as used by the client, but the data generated is completely independent.
- 3) Session ID: if the client field was nonzero, the same value is sent back; otherwise the server's session ID field contains the value for a new session.
- 4) Cipher Suite: the server uses this field to send a single set of protocols selected by the server from those proposed by the client. The first element of this field is a chosen method of exchange of cryptographic keys between the client and the
- 5) server. The next element is the specification of encryption algorithms and hash functions, which will be used within the session being initiated, along with all specific parameters [24].

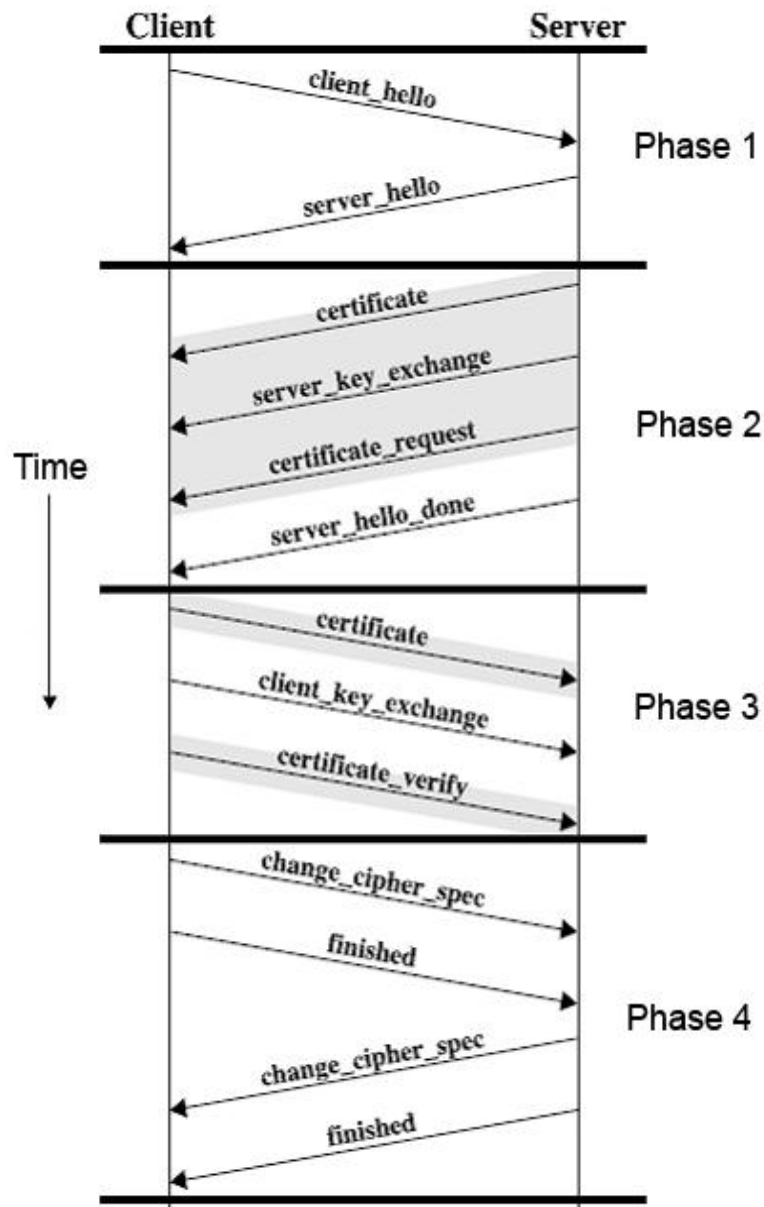


Figure 2.10 - Handshake protocol action

The set of encryption algorithms and key exchange method sent in the Cipher Suite field establishes three components:

- 1) The method of key exchange between the server and client.
- 2) The encryption algorithm for data encryption purposes.
- 3) A function used for obtaining the MAC value.

The server begins the next phase of negotiations by sending its certificate to the client for authentication. The message sent to the client contains one or a chain of X509 certificates.

These are necessary for authentication of both the server and the certification path towards a trusted certification official of the certifying body for the server. This step is not obligatory and may be omitted, if the negotiated method of key exchange does not require sending the certificate.

Depending on the negotiated method of key exchange, the server may send an additional server key exchange message, which is however not required in the case when the fixed Diffie-Hellman method or Ron, Shamir and Adleman (RSA) key exchange technique has been negotiated.

Moreover, the server can request a certificate from the client. The final step of Phase two is the server done message, which has no parameters and is sent by the server merely to indicate the end of the server messages.

After sending this message, the server waits for a client response. Upon receipt of the message, the client should verify the server's certificate, the certificate validation data and path, as well as any other parameters sent by the server in the server hello message [25].

The client's verification consists of:

- 1) Validation date check of the certificate and comparison with the current date, to verify whether the certificate is still valid.
- 2) Checking whether the certifying body is included in the list of trusted Certifying Authorities in possession of the client.
- 3) If the Certificate Authority (CA), which has issued the server's certificate, is not included in the CAs list, the client attempts to verify the CA signature.
- 4) If no information about the CA can be obtained, the client terminates the identification procedure by either returning the error signal or signaling the problem for the user to solve it.
- 5) Identifying the authenticity of the public key of the CA which has issued the certificate: if the Certifying Authority is included in the client's list of trusted CAs, the client checks the CA's public key stated in the server's certificate with the public key available from the list. This procedure verifies the authenticity of the certifying body.
- 6) Checking whether the domain name used in the certificate matches the server name shown in the server's certificate.

Upon successful completion of all steps the server is considered authenticated. If all parameters are matched and the server's certificate correctly verified, the client sends the server one or multiple messages.

Next is the client key exchange message, which must be sent to deliver the keys. The content of this message depends on the negotiated method of key exchange. Moreover, at the server's

request, the client's certificate is sent along with the message enabling verification of the certificate. This procedure ends phase three of negotiations [24].

Phase four is to confirm the messages so far received and to verify whether the pending data is correct. The client sends a Change Cipher Spec message and then sets up the pending set of algorithm parameters and keys into the current set of the same. Then the client sends the finished message, which is first protected with just negotiated algorithms, keys and secrets.

This is to confirm that the negotiated parameters and data are correct. The server in response to the client sends the same message sequence. If the finished message is correctly read by either party this confirms that the transmitted data, negotiated algorithms and the session key are correct.

This indicates that the session has been terminated and that it is possible to send the application data between the server and the client, via SSL. At this point the TCP session between the client and the server is closed; however a session state is maintained, allowing it to resume communications within the session using the retained parameters.

It is worth noticing that both phases two and three are used by both parties to verify the authenticity of the server's certificate and possibly the client's certificate during the handshake step. If the server cannot be successfully authenticated by the client on the basis of the delivered certificate, the handshake terminates and the client will generate an error message. The same will occur at the

server if the client's certificate authenticity cannot be confirmed.

At first glance this process seems to be somewhat complicated, however this takes place at each connection with the server of an SSL enabled service, such as request the address of a website beginning with Secure Hyper Text Transfer Protocol (HTTPS://) [15].

2.4 IPSec VPN versus SSL VPN

In this section, the briefly tables would be show the details for technical and functional comparisons between IPSec VPN and SSL VPN.

2.4.1 Technical Comparison

Table (2.1) shows the technical comparison between IPSec VPN and SSL VPN technologies [2].

Table 2.1- The technical comparison between IPSec and SSL
VPNs

NO.	Optional Clause	IPSec	SSL
1	Authentication Algorithm	RSA_DSA Digital Signature RSA Public Key, PSK	Server : RSA and DSA Client : RSA_DSA Anonymous : none
2	Authentication Method	Mutual Authentication	Server Authentication Client Authentication
3	MAC	HMAC-SHA-1-96 HMAC-MD5-96	HMAC-SHA-1 HMAC-MD5
	Hash Length	12 Byte , 12 Byte	20 Byte , 16 Byte
4	Connection Mode	Tunnel Mode Transport Mode	One connection per one session type
5	Remote Access	One PSK key Aggressive Mode User Authentication	Server : RSA or DSA Client : RSA/DSA

6	Ports	Server : ESP 50/TCP, AH 51/TCP Client : ESP 50/TCP, AH 51/TCP	Server : HTTPS 443/TCP Client : Any		
7	Perfect Forward Secrecy	PFS	PFS		
8	Order of Cryptographic Operations	Encrypt data then create MAC	Create MAC then encrypt data		
9	Cipher List Proposal	Bi Direction	One Direction		
10	Interoperability	Does not integrate well with other IPsec vendors	Trouble free and well integrated		
11	Overhead Size	Tunnel Mode ESP : 32 Byte ESP & AH : 44 Byte	HMAC-MD5 : 21 Byte HMAC-SHA-1 : 25 Byte		
		Transport Mode ESP : 36 Byte ESP & AH : 48 Byte			
12	Residing Layer	Network layer	Application layer		
13	Time of Handshake Process	Main Mode (PSK)	97 msec	Server Auth.	41.7 Msec
		Main Mode (RSA)	170 msec	Client Auth.	74.8 Msec
		Aggressive Mode (PSK)	56 msec	Server (DH)	66.1 msec
				Client (DH)	118.6 msec
14	Session Resumption and Rekeying Time	26 msec	1.3 msec		
15	NAT Traversing	Clients bound to a specific ports	NO specific port		

2.4.2 Functional Comparison

Table (2.2) shows the functional comparison between IPsec VPN and SSL VPN technologies [1].

Table 2.2 - The functional comparison between IPsec and SSL

VPNs

NO.	Function	IPsec	SSL
1	Configuration	Hard	Easy
2	Client Authentication	Must	Option
3	Pre-Shared Key	Yes	No
4	Interoperability Problem	Yes	No
5	TCP Application Support	All	Some
6	UDP support	Yes	No
7	Throughput Rate	High	High
8	Compression Support	Yes	Open SSL
9	Handshake Time	Slow	Fast

Chapter Three

Related Studies and Works

3.1 Introduction

A few studies and works available are represented the core of this research, because the development products as same as this research in the market has no white papers or related technical works are published that can be derived benefits from them.

So, this research depends on the other several studies papers and related trusted works which published so far which have strong link with this research fields as virtual private network, IPSec and SSL VPNs technologies, some companies' white papers produce hybrid VPNs technology, using ensemble system mechanism to build a powerful network system, statistical benchmarks and experimental environments of performance analysis measurements.

3.2 Related VPN Studies and Works

Al-Chaal [1] introduced a Customer Edge based VPN approach that offers different management network services in behalf of customers. Dynamic and easily manageable approach for secure IP VPN environments shifts the management hassles from customer's side to the VPN service provider.

Yet by using this approach service providers have only to care about managing customers edge devices that are the gateways to the customers' networks. This approach is a centralized solution, including VPN creation, deployment and membership management,

is under the control of a single management operation point. This approach focuses on three key aspects such as management, dynamism and security. It investigates the use to offer group communication services and manages secure web services.

Djin [10] presented a solution in the form of a centralized access control framework called an access control service, which can grant remote users network presence and simultaneously aid them in accessing various network resources with varying access control policies.

This solution provides a centralized framework for administrators to manage access to their resources and achieves these objectives using VPN technology, network address translation and by proxy various authentication protocols on behalf of remote users.

3.3 Related IPSec VPN Studies and Works

Perlman [26] analyzed the IPSec key exchange standard and described some issues with the rest of IPSec, such as what services it can offer without changing the applications, and whether the AH header is necessary. He discussed the various protocols of IKE, and make suggestions for improvement and simplification.

Ferguson and Schneider [33] showed cryptographic evaluation of IPSec, and prove that IPSec is far better than any IP security protocol which has come before such as PPTP, L2TP, etc. They believe that it will ever result in a secure operational system. It

is far too complex, and the complexity has lead to a large number of ambiguities, contradictions, inefficiencies, and weaknesses.

It has been very hard work to perform any kind of security analysis; Current evaluation methods cannot handle systems of such a high complexity, and current implementation methods are not capable of creating a secure implementation of a system as complex as this.

The IPSec experience has demonstrated that a committee design process is wholly incapable of creating a useful design for a security system. There is a fundamental connect between the committee process and the property of security systems being only as strong as their weakest link.

The use of IPSec in its current form is more protection of any kind of valuable information, and hope that future iterations of the design will be improved. However, even more strongly discourage any current alternatives, and recommend IPSec when the alternative is an insecure network.

Alshamsi and Saito [2] compared the two protocols IPSec and SSL in technical characteristic and functionality terms to analysis the security and performance properties for them. The study presented that each protocol has unique properties and choosing IPSec or SSL depends on the security needs.

3.4 Related SSL VPN Studies and Works

Mitchell et. al. [24] studied the finite-state analysis of SSL 3.0. The analysis presented using a sequence of incremental

approximations to the SSL 3.0 handshake protocol. This process identified the main shortcomings in SSL 2.0 that led to the design of SSL 3.0, as well as a few anomalies in the protocol that is used to resume a session in SSL 3.0. This study demonstrated the feasibility of using formal methods to analyze commercial protocols.

Guttman [11] presented the comparisons of the most popular application level security protocols such as SSL / TLS.

His paper provided a detailed breakdown and analysis of the performance characteristics of the different protocols, identifying potential performance problem areas and providing guidance for protocol designers and implementers.

Hosner [13] presented his research about open source SSL VPN revolution. It also operates in kernel space providing the opportunity for catastrophic failure. Open VPN rejects the complexity of IPSec by using the battle tested SSL / TLS protocol and cryptographic libraries to provide equal or better function in a simpler package.

Open VPN also operates in user space increasing security and stability. Open VPN is the first real SSL VPN to provide the same function and security as its IPSec predecessors.

3.5 Related Hybrid IPSec SSL VPNs Studies and Works

Bickford et. al. [3] proved the correct generic switching protocol with the Nuprl proof development system. They introduced the concept of Meta properties and use them to formally characterize communication properties that can be preserved by switching.

They designed a generic switching protocol for the construction of adaptive network systems and formally proved it correct with the Nuprl logical programming environment.

Rao [28] discussed the advantages and disadvantages of IPSec VPN and SSL VPN technology as well as compare both to the Net6 Hybrid-VPN Gateway solution. He provided high-level reference to the total cost of ownership business case of comparison.

Finch [9] showed in white paper the deploy IPSec and SSL together for a complete remote access solution. Although some enterprises have found themselves in situations where deploying solely SSL or IPSec is satisfactory, chances are that they will need to support users that want access from trusted and un-trusted environments. As a result, neither VPN type may be a perfect fit for them entire population of remote enterprise access users.

Choosing the right solution that enables employees, partners and others to access the enterprise network securely without inhibiting productivity is critical. The objective is to make it easy, easy for IT staff to deploy and manage an integrated VPN solution in unison, and easy for users to connect regardless of whom they are and where they are.

3.6 Related Ensemble System Mechanism Studies and Works

Hayden [12] began the presenting of the ensemble system architecture as well as background in group communication. He described the various components of the architecture and compared this architecture with that of other layered communication systems.

The ensemble protocols make heavy use of layered micro protocols. He described optimization techniques that greatly reduce the performance overheads introduced by layering and show how the architecture facilitates these optimizations. In addition, he showed how to formalize these optimizations in type theory and implement them using the Nuprl theorem prover. He described how the use of ML impacted the system and presented a wide range of comparisons between ensemble and a similar system implemented in C programming languages.

Leroy [19] documented the objective Caml system manual. This manual overviewed the Caml programming language. It used the interactive system, which is started by running OCaml from the UNIX shell, or by launching the OCamlwin.exe application under Windows.

Rennesse et. al. [29] presented in stature research the building adaptive systems using ensemble in networking and distributed computing which are creating a new generation of applications that must adapt as the environment within which they execute changes.

Examples of adaptation include switching protocols to overcome a security exposure or failure mode seen only in certain settings, changing data rates to accommodate a slow link, or adapting the behavior of the high level application to match the set of participants using the application. They described the ensemble system as a tool for building adaptive distributed programs.

Kreitz et. al. [18] correctly proved with Nuprl proof development

system by the Nuprl logical programming environment (LPE) which is a framework with database of thousands of definitions, theorems and examples for the development of formalized mathematical knowledge that is well suited to support such a formal design of software systems.

3.7 Related Performance Analysis Studies and Works

Miltchev et. al. [5] investigated the performance of IPSec using micro and macro benchmarks. Their tests explored how the various modes of operation and encryption algorithms affect its performance and the benefits of using cryptographic hardware to accelerate IPSec processing.

They compared against other secure data transfer mechanisms, such as SSL. Their experiments had shown that IPSec outperforms all other popular schemes that try to accomplish secure network communications.

Even though this safety comes at a price, which is present no matter which protocol one uses, it is possible to get enough performance for practical use by using dedicated cryptographic hardware. This price may easily be acceptable for many applications and environments, given the remarkable flexibility and transparency offered by IPSec.

Lin et. al. [20] designed, implemented and evaluated the performance of IPSec VPN. They discussed the problems when combine IPSec into current TCP/IP module by porting an IPSec Free SWAN into a router.

In order to understand the impact on router's performance when using various services and hash or encryption algorithms provided by IPSec, they tested the throughput of the router before and after applying IPSec.

Jensen [16] enhanced the SSL performance. This paper presented performance tested of the Open SSL library handling sockets and certificates, and sending HTTP over SSL to Apache. The main result of the paper is an analysis of the effect of caching information at various levels of the SSL connection on the client side, thus providing guidelines for speeding up SSL socket connections.

Since Open SSL and libraries built on Open SSL are ubiquitous, this work will be of interest to anyone writing grid and SSL clients. Often SSL tuning information and options are for the server side, including buying hardware acceleration. SSL is used to stress test servers.

However, the Open SSL library allows optimizations by caching information also on the client side. In this paper they discuss the possibilities and impact.

Rodeh et. al. [31] represented the architecture and performance of security protocols in the ensemble group communication system. Ensemble is a group communication system. It allows processes to create process groups within which scalable reliable First in First out (FIFO) ordered multicast and point-to-point communication are supported.

This paper described the security protocols and infrastructure

of ensemble. Applications using ensemble with the extensions described here benefit from strong security properties. Under the assumption that trusted processes will not be corrupted, all communication is secured from tampering by outsiders.

Chapter Four

Development of the Proposed Hybrid IPSec SSL VPNs Technology

4.1 Introduction

Each type of IPSec VPN and SSL VPN technology has distinct advantages and disadvantages, and differences of technical characteristic and functionality terms. So which VPN technology is the right for use? The quick answer is: it depends, but the probable answer is: a combination of both.

In this research, a Hybrid IPSec SSL VPNs Technology system is proposed as a replacement solution which provides enterprises with the combined advantages of both IPSec VPN and SSL VPN technologies, but none of the shortcomings.

IPSec VPN technology provides network layer access and encryption. SSL VPN provides application layer access and encryption. The proposed Hybrid VPN system combines network layer access with application level encryption together in a combination technology.

This drastically improves the end user experience while significantly reducing the IT security administration support overhead and security risks.

The proposed Hybrid VPN technology relieves enterprises from the burden of maintaining two separate VPNs infrastructures because the Hybrid VPN technology provides the benefits of both.

4.2 Architecture

This section presents general description, client phase description, proposed Hybrid VPN phase description, and server phase description of the proposed Hybrid VPN system.

4.2.1 General Description

Figure (4.1) illustrate the proposed Hybrid VPN system that supports any application includes peer-to-peer sharing, audio-video streaming and real-time applications. In addition, the proposed Hybrid VPN system supports all protocols, traverses firewalls, and hides the IP addresses of the remote network to inhibit the traversal of viruses, worms and other attacks, so the enterprises can now just deploy one solution for their secure remote access needs which reduce the technical support costs.

The proposed Hybrid VPN system support mobility feature which enables clients to work from anywhere, behind any firewall with auto reconnects. A client can be connected at one location, disconnected from the network and be automatically reconnected when they get online at their next location. With the proposed Hybrid VPN system, clients simply access a secure web portal and use their normal authentication credentials.

There is no need to worry about VPN client software or updates. Clients remain productive by having the same network experience and application access available while sitting at their desk.

A session which is encrypted using proven technologies such as SSL will be created between the client computer and the

proposed Hybrid VPN system that installed in the delimiter zone area of the target private network. The proposed Hybrid VPN system participates on two networks: a private network as well as a public network with a publicly routable IP address.

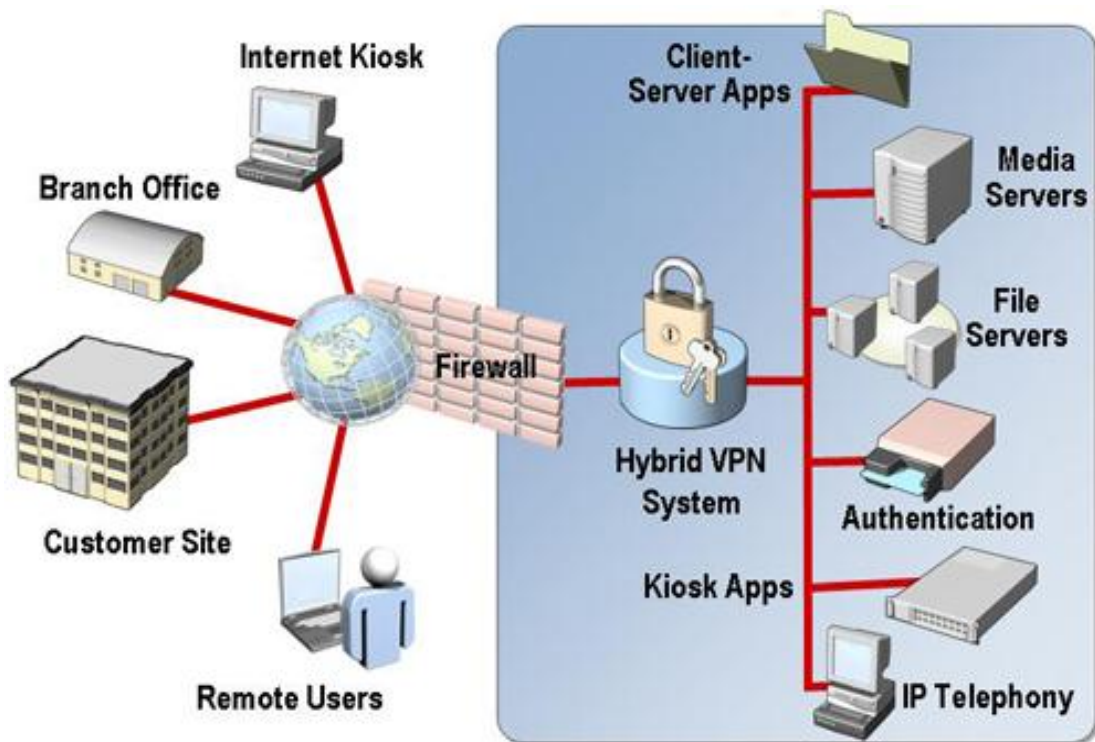


Figure 4.1 – Overview of Hybrid IPsec SSL VPNs Technology System

4.2.2 Client Phase Description

Clients launch the proposed Hybrid VPN system web portal interface by simply accessing a secure web Uniform Resource Locator (URL) over intranet or internet. This is typically the public name of the proposed Hybrid VPN system, which prompts the client for authentication over secure HTTP protocol.

Figure (4.2) illustrate the proposed Hybrid VPN system authorize the client credentials with an enterprise's logon server such as Microsoft LDAP or Remote Authentication Dial-In User Service (RADIUS) environment and if the credentials are correct; the logon server finishes the handshake with the client computer. Otherwise the system will be stopped the restricted URL connection that access from un-trusted client computer.

Once the clients have been launched with SSL VPN remote access which is a default login service, they establish a secure tunnel over HTTP protocol with certain configured port on the proposed Hybrid VPN system using SSL encrypted connection. The SSL VPN clients can be test and validate client phase digital certificates which configured by the proposed Hybrid VPN system.

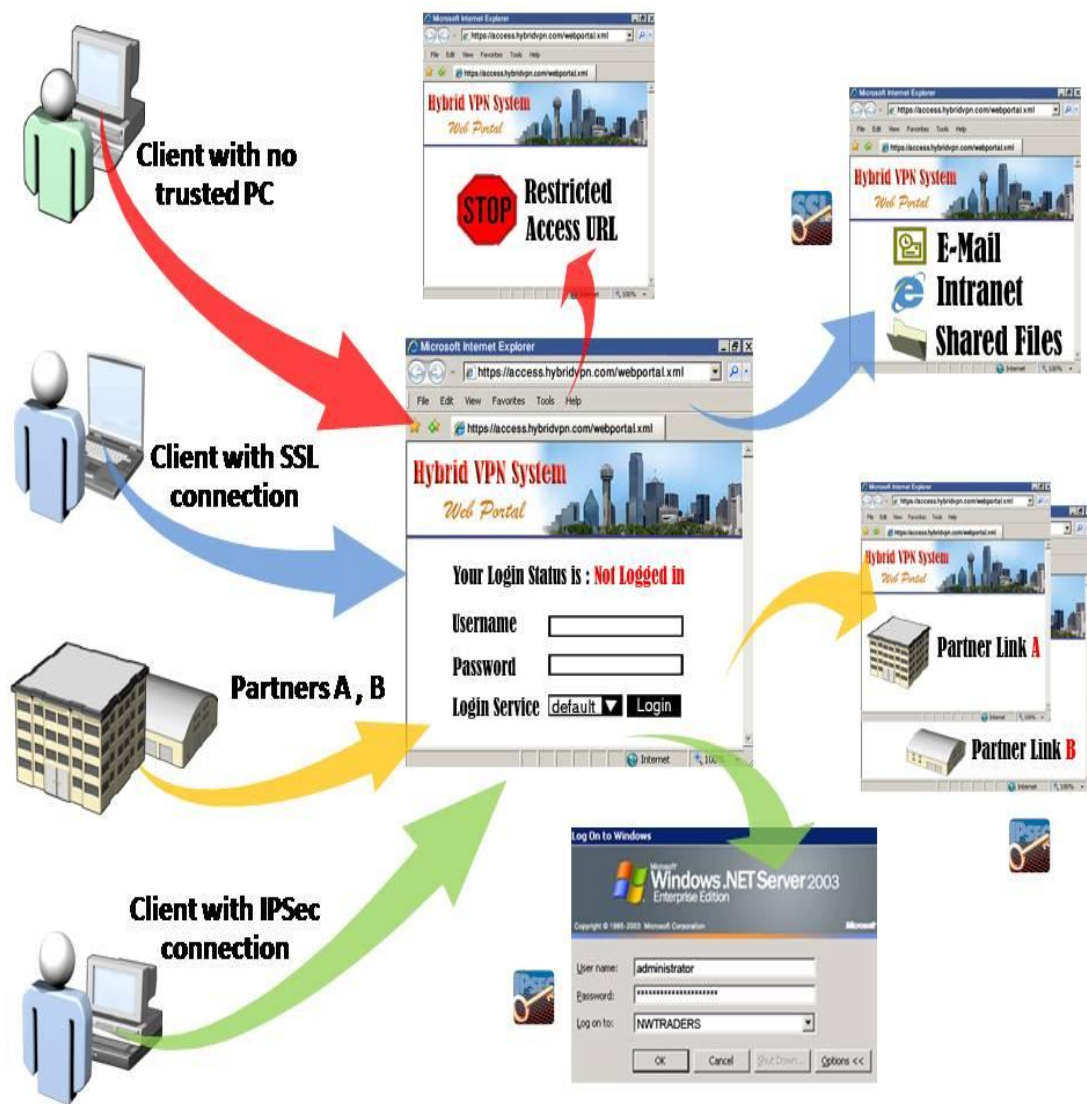


Figure 4.2 – Client phase web portal interface shows the different secure connections that delivered to the Hybrid VPN system

Once the enterprise partners and clients have been launched with IPsec connections, they authenticate information which sent to validate the established tunnel.

The proposed Hybrid VPN system sends configuration information over to the remote access that describe the networks which need to be secured and potentially an IP address if the system

administrator enables IP address visibility.

The web portal interface guidelines helps clients to choose between IPSec VPN and SSL VPN connections with certain application for best performance.

4.2.3 Proposed Hybrid VPN System Phase Description

The proposed Hybrid VPN system handles the SSL tunnel or IPSec tunnel and accepts any incoming packets destined for the private network. If the packets meet the authorization and access control criteria, they are first fixed up the IP headers regenerated to appear from the proposed Hybrid VPN system as a private network IP address range and then they are injected into the assigned network.

Network Address Translation (NAT) also called IP masquerade, is a process of translating the source header of IP packets so they will be routable across wide area networks.

The primary security function that NAT provides is its state inspection of incoming packets. NAT is a process of translating and recording the packet header information of all traffic leaving local network. When incoming traffic arrives, NAT interface compares the header information to its records. If there is a match retranslates, it forwards the packet to the client. If there is no match, it deems the packet as suspicious and will either deny or drop such traffic.

For most small networks, NAT is sufficient security. For larger networks hosting services, the packet filtering is an extended measure of security that can help lock down servers so that only the

necessary services may be used, while all other communication is restricted.

Figure (4.3) illustrate the proposed Hybrid VPN system maintains reverse NAT table operations. The proposed Hybrid VPN system makes firewall friendly and thus allows client computer to access private networks from behind other enterprises firewalls without creating any problems, these NAT firewalls can now create a mapping table that allows them to route secure packets from the proposed Hybrid VPN system back to the client computer.

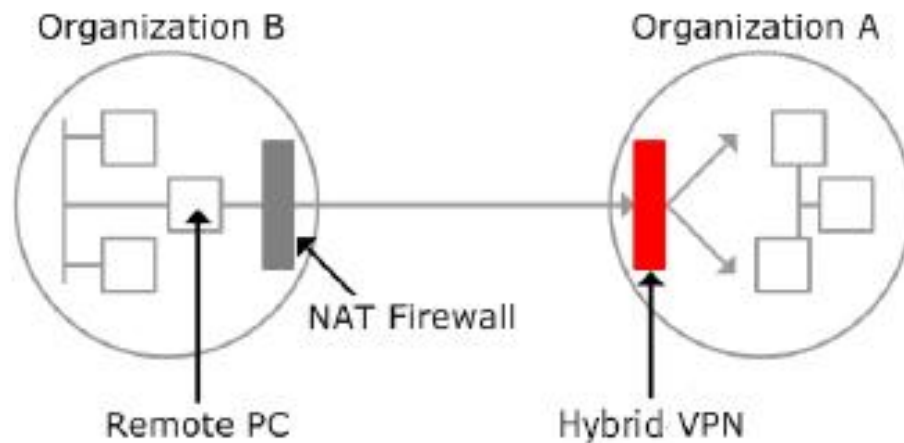


Figure 4.3 – Hybrid VPN system reverse NAT table operations

Once the secure HTTP session is established, the payload of the session is encrypted and carries secure packets to the proposed Hybrid VPN system; each packet is encrypted to provide a very high level of security include any header information such as the IP header.

Target servers view connections as originating from the proposed Hybrid VPN system on the private network, thus hiding client IP address. Locally, on the client computer, all connections

related traffic such as Acknowledgment (ACK) and Synchronized Acknowledgment (SYN-ACK) packets are recreated by the proposed Hybrid VPN system so as to appear from the private server.

Consider TCP connections for example that represent widely known connections from local applications on the client computer are securely tunneled over to the proposed Hybrid VPN system at which point the connections are re-established to the target server.

4.2.4 Server Phase Description

The proposed Hybrid VPN system sits in an enterprise's delimiter zone with access to both the external network and internal network. It maintains a port mapped NAT table, so that connections can be matched. The packets can be sent back over the tunnel to the client with correct port numbers for certain servers.

These servers have installed services that support many applications which use different protocols, such as real-time traffic which implemented over UDP, because of TCP is not appropriate for real-time traffic due to the delay introduced by acknowledge meets and retransmission of lost packets.

The proposed Hybrid VPN system is essentially acting as a low level packet filter with encryption. It drops traffic which does not have authentication or does not have authorization permissions for a particular server or servers on the network.

4.3 Technical Details

This section presents general technical details, technical details for client phase, proposed Hybrid VPN system phase, and server phase.

4.3.1 General Technical Details

Figure (4.4) illustrates the default International Standards Organization and Open Systems Interconnection (ISO/OSI) standard model, the available SSL layer lies over transport layer, and the available IPSec layer lies over data link layer.

Application Layer	Application Layer	Application Layer
Presentation Layer	Presentation Layer	Presentation Layer
Session Layer	Session Layer	Session Layer
Transport Layer	SSL Layer	Transport Layer
Network Layer	Transport Layer	Network Layer
Data Link Layer	Network Layer	IPSec Layer
Physical Layer	Data Link Layer	Data Link Layer
	Physical Layer	Physical Layer

Figure 4.4 – ISO/OSI model, SSL layer and IPSec layer

Figure (4.5) illustrates both SSL layer and IPSec layer lie in the same ISO/OSI model to work with the proposed Hybrid VPN system, but bringing both SSL and IPSec layers together in the same model due to packet treatment confusion.

In sending case, the packet has been protected by SSL layer for example should be passed from IPSec layer as ISO/OSI model

laws which already protected the packet another time. Therefore, any packet in receiving case should be confused of which security layers have been required handling with it.

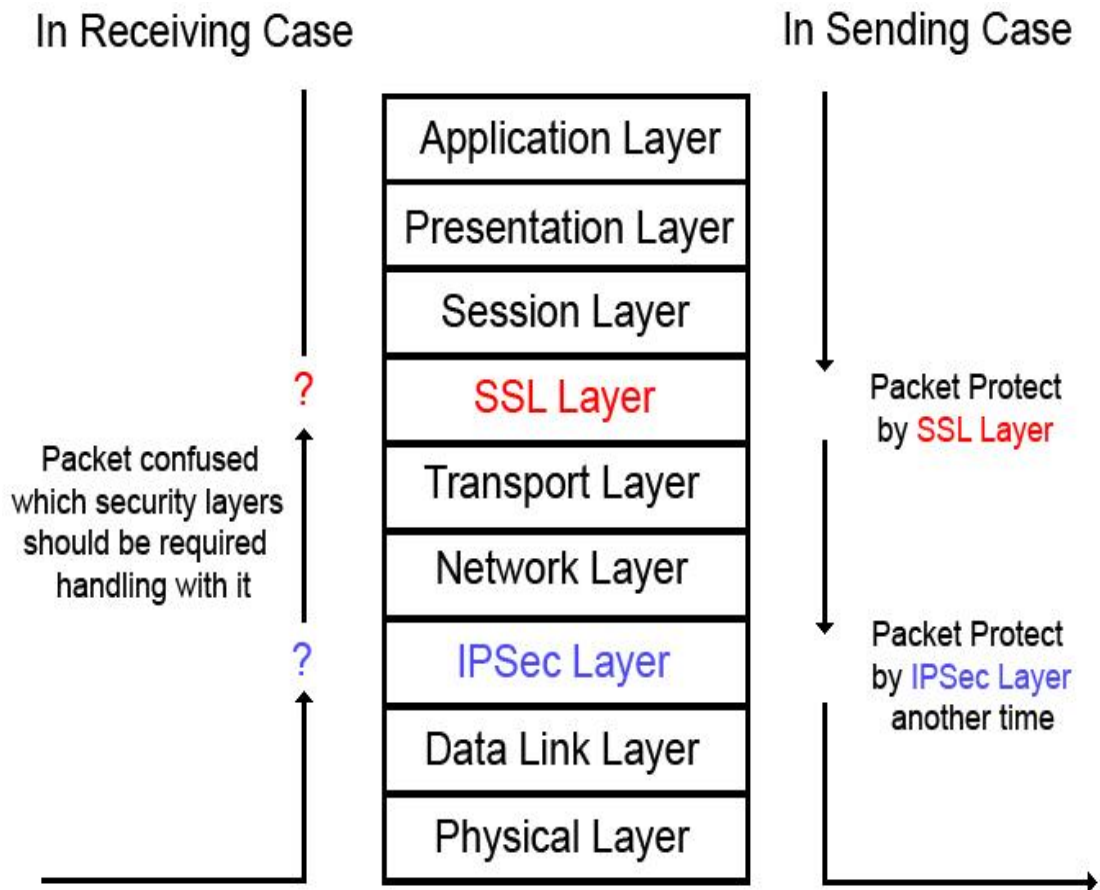


Figure 4.5 - Packet confusion between SSL layer and IPsec layer

To solve these problems, the proposed Hybrid VPN system should have a mechanism to switch between security protocols to avoid packets confusion, so the proposed Hybrid VPN system used the ensemble system mechanism.

Figure (4.6) illustrates the general design of this mechanism that serves as a wrapper for a specified protocol with the same

functionality.

The main function of the ensemble system mechanism that the switch protocol layer shall interact with the upper layer in a transparent fashion, that is, the upper layer cannot tell easily that it is running on the switch protocol rather than on one of the underlying protocols in lower layer, even as the switch protocol switches between protocols.

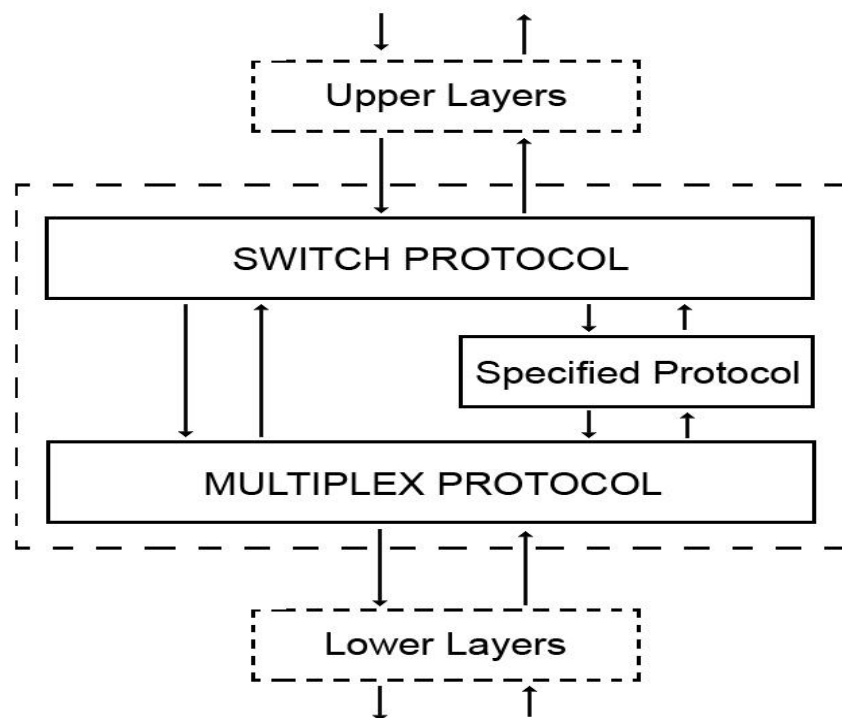


Figure 4.6 - General design of the ensemble system mechanism

In a protocol layering architecture like the one used in ensemble system mechanism, the switch protocol will reside on top of the specified protocol to be switched, coupled by a multiplex protocol below them.

The basic idea of the switch protocol is to operate in one of two modes. In normal mode, it simply forwards packets from the upper

layer to the current protocol and vice versa.

When there is a request to switch to a different protocol the switch protocol goes into switching mode and labels the packet with own special header, during which the switching layer at each process will deliver all packets that were sent under the specified protocol while buffering packets sent under the new one. The switch protocol will return to normal mode as soon as to all packets for the specified protocol has been delivered.

The multiplex protocol simulates multiple connections over a single communication channel to backward the packets to switch protocol layer after checked the special switch protocol header, otherwise the packet would be passed to a specified protocol.

Figure (4.7) illustrates a new general design of the proposed Hybrid VPN system when the IPSec layer wrapper as a specified protocol with the switch layer and multiplex layer.

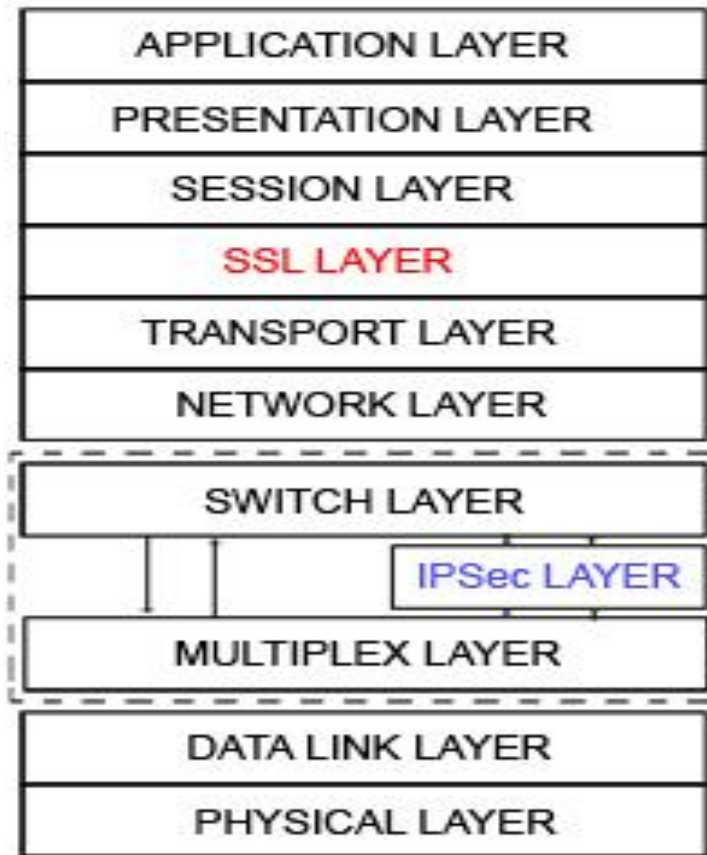


Figure 4.7 – New general design of proposed Hybrid VPN system

In brief technical description for ensemble system mechanism that is high performance network protocol architecture, designed particularly to support group membership and communication protocols. The ensemble system mechanism is constructed from simple micro protocol modules, which can be stacked in a variety of ways to meet the communication demands of its applications.

This mechanism has a micro protocols implement basic sliding window protocols, fragmentation and reassembly, flow control, signing and encryption, group membership, packet ordering, and other functionality.

It includes a library of over sixty of these components. Each module adheres to a common ensemble micro protocol interface. There is a top level and a bottom level part to this interface. The top level part of the interface of a module communicates with the bottom level part of the interface of the module directly on top of it.

Figure (4.8) illustrate the ensemble system mechanism interprets a protocol; it is useful to think of a protocol as a function.

Such this function takes the state of the collected variables maintained by the protocol and an input event such as user operation, an arriving packet, an expiring timer ... etc, and produces an updated state and a list of output events. This function can be optimized if something is known about an input event and the state of the protocol.

This knowledge expressed by a so called Common Case Predicate (CCP) which is a Boolean function on the state of a protocol and an input event. CCP conditions are specified and configured by implementation of the protocol, and typically determined from runtime statistics.

In the Hybrid VPN system case, a CCP condition may be true if the delivered event does not have an equal to the certain sequence number configured in the CCP condition, the event would moved up to bypass code fragments then to the upper layers.

In other words, if they arrived packet does not have the label header which distinguishes the packets protected by SSL layer; the normal mode of the mechanism is worked. In this time, the IPsec layer handles these packets and use the buffer if needed then push the packets to the upper layers.

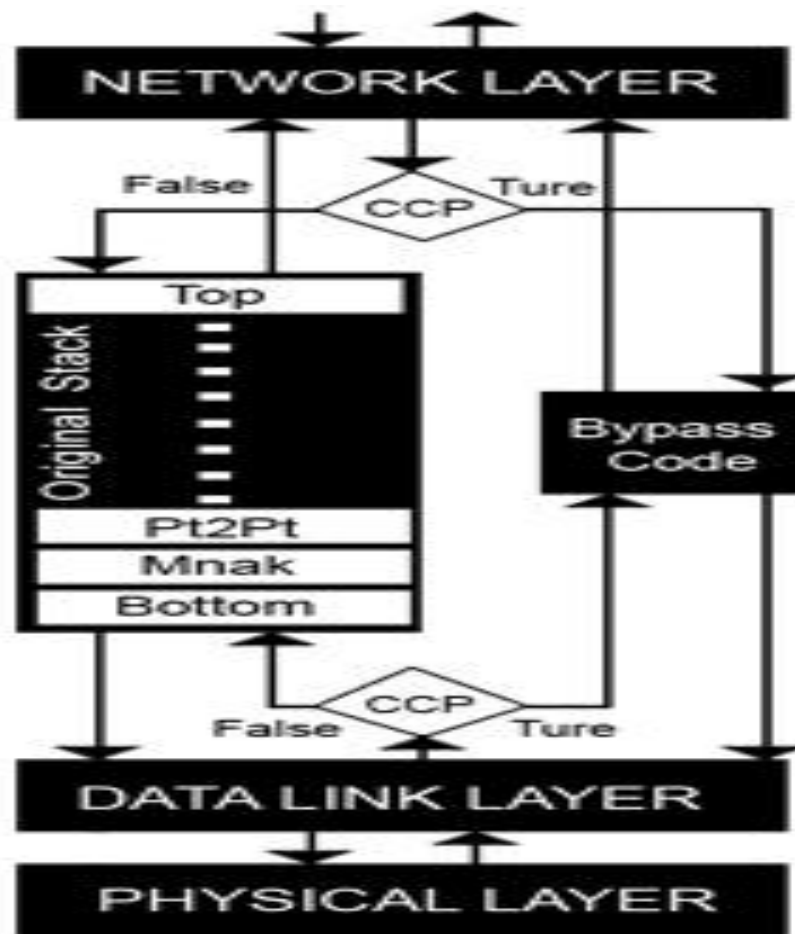


Figure 4.8 - Technical details for ensemble system mechanism

The same CCP conditions that are used at compile time to generate the bypass code are also used at runtime to decide whether to deliver an event to the bypass code or to the original full stack.

CCP condition may be false if the delivered event has an equal to the certain sequence number configured in the CCP condition, the event would move up to original full stack components for processing then to the upper layers.

In other words, if the arrived packet has an expected one byte size label which distinguishes the packet protected by SSL layer, the switching mode of the mechanism worked and switches the packet to the upper layers for handling without passing through IPsec layer.

Often, the header added for the particular packet to be distinguished becomes a constant. The constant header fields in the combined headers of all packets can be combined into a single, short, identifier, thus compressing the header and reducing handling overhead.

Therefore, the ensemble system mechanism is currently used in the Bolt, Beranek, and Newman (BBN) platforms, a fault-tolerant test bed at Jet Propulsion Laboratory (JPL), the Adapt multimedia middleware system at Lancaster University, a multiplayer game by Sega soft, and in the Allier financial database tools.

4.3.2 Technical Details for Client Phase

In this phase, the client launch Hybrid VPN system web portal logon interface and write the username and password then select one of the two login services over SSL VPN or IPsec VPN and then click login event.

Figure (4.9) illustrate the SSL record protocol operation which provides a basis for secure communication confidentiality with packet authenticity when the client selects SSL VPN service.

The most complex part of SSL is handshake protocol which allows the server and client to authenticate each other based on the interchange cryptosystem such as RSA that negotiate encryption included MAC algorithm and cryptographic keys. The handshake

protocol used before any application data is transmitted.

The other SSL protocols such as change cipher spec protocol which creates new cipher parameters during the renegotiation.

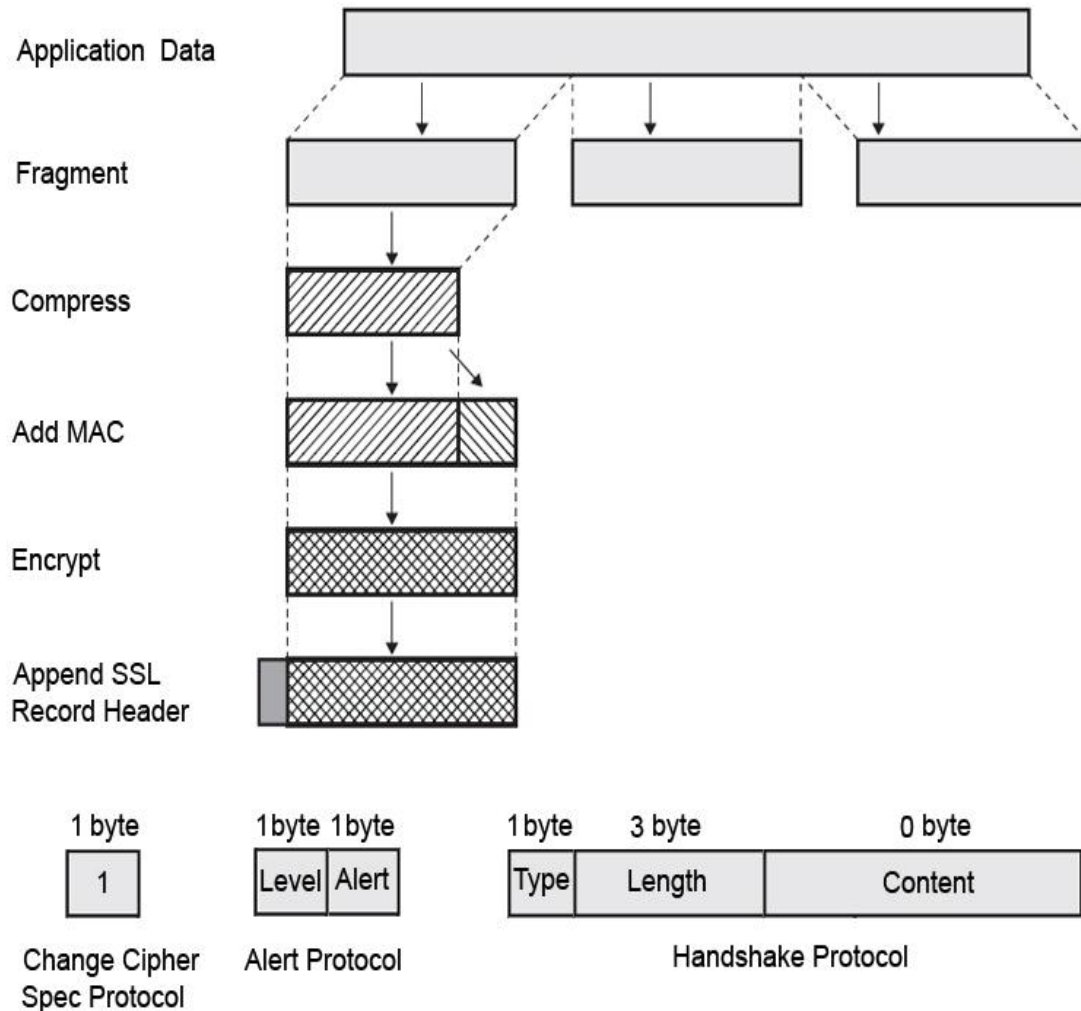


Figure 4.9 – SSL record protocol operation

Another SSL protocol is the alert protocol which sends an alert signals when conditions are unusual i.e. the sender will not send packets anymore and the fatal error has been resulted in disconnection.

The application data such as HTTP message divided into certain fragments and compressed, then MAC would be added to the message. The segment data appended SSL record header after encryption, then forward to transport layer and network layer respectively.

Figure (4.10) illustrate the packet when it forwarded to switch layer through ensemble system mechanism which configured to work properly with the proposed Hybrid VPN system concepts.

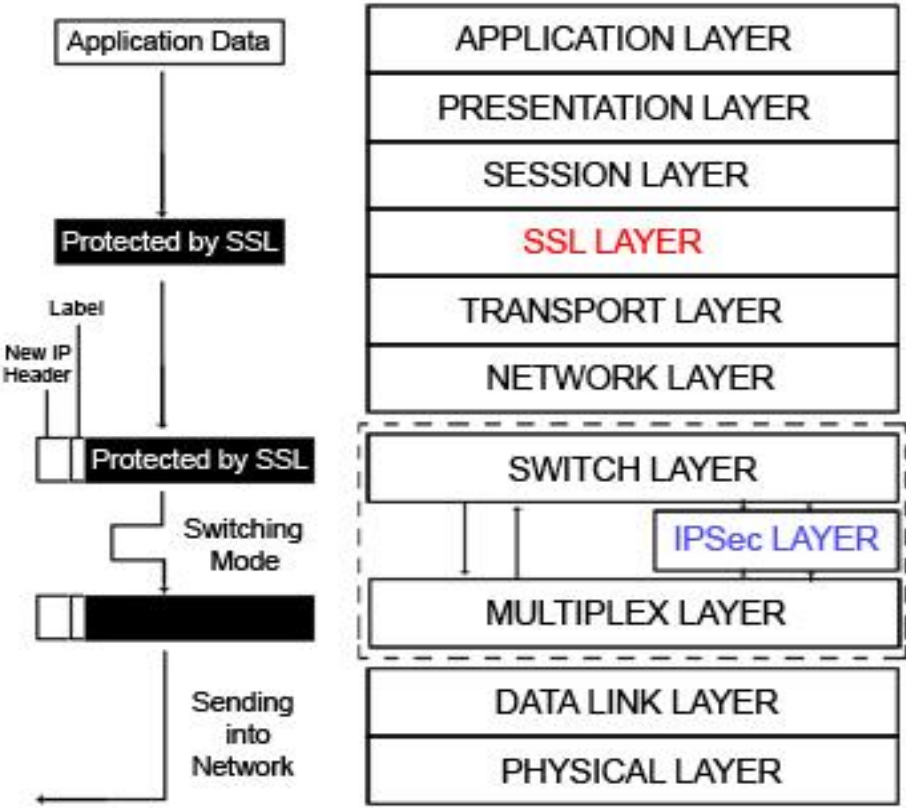


Figure 4.10 - Packet protected by SSL layer forwarded through Hybrid VPN

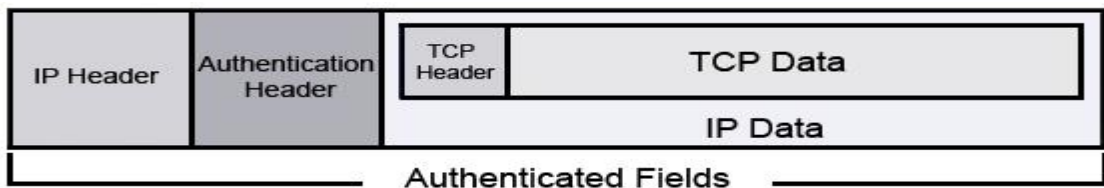
At this time, a new header would be created which sized a one

byte to work as a label to distinguish this packet that protected by SSL layer and added a new IP header to the packet.

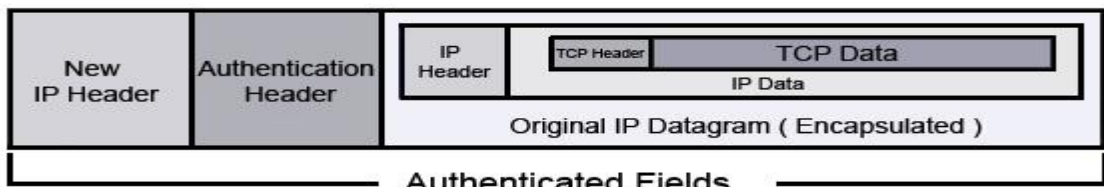
The ensemble system mechanism enter in switching mode and switch the packet to the multiplex layer without passing through IPSec layer and forward to data link layer and physical layer respectively to the destination.

Figure (4.11) illustrates the set of IPSec protocols that encrypts and authenticates all traffic at the IP level and protects all packets sent along a path when the client selects IPSec VPN service.

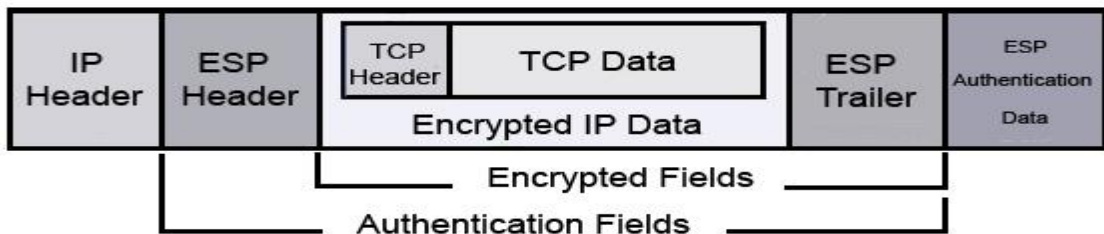
IPSec supports two security services: authentication header (AH) and encapsulating security payload (ESP) which provides packet integrity, origin authentication and anti-replay services, but ESP also can provide confidentiality.



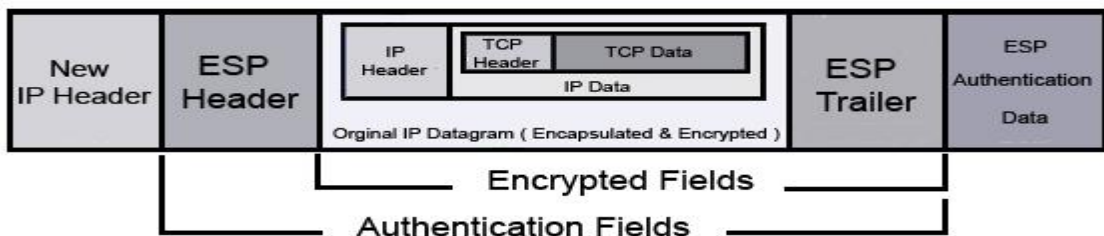
IP AH Datagram Format - IPsec Transport Mode



IP AH Datagram Format - IPsec Tunnel Mode



IP ESP Datagram Format - IPsec Transport Mode



IP ESP Datagram Format - IPsec Tunnel Mode

Figure 4.11 – IPsec cryptography and network security

In order for IPsec to function properly, the sender and receiver must share a public key. This is done through internet key exchange protocol that allows the receiver to get a public key and authenticate the sender.

A fundamental construct in IPsec is the security association (SA), which establishes a basic connection with security services.

IPSec supports two encryption modes: transport and tunnel.

The transport mode encrypts just the upper layer headers and data payload of each packet. The more secure Tunnel mode encrypts the IP header, upper layer headers, and data payload.

Figure (4.12) illustrate the IPSec layer which always interacts with ensemble system mechanism by normal mode.

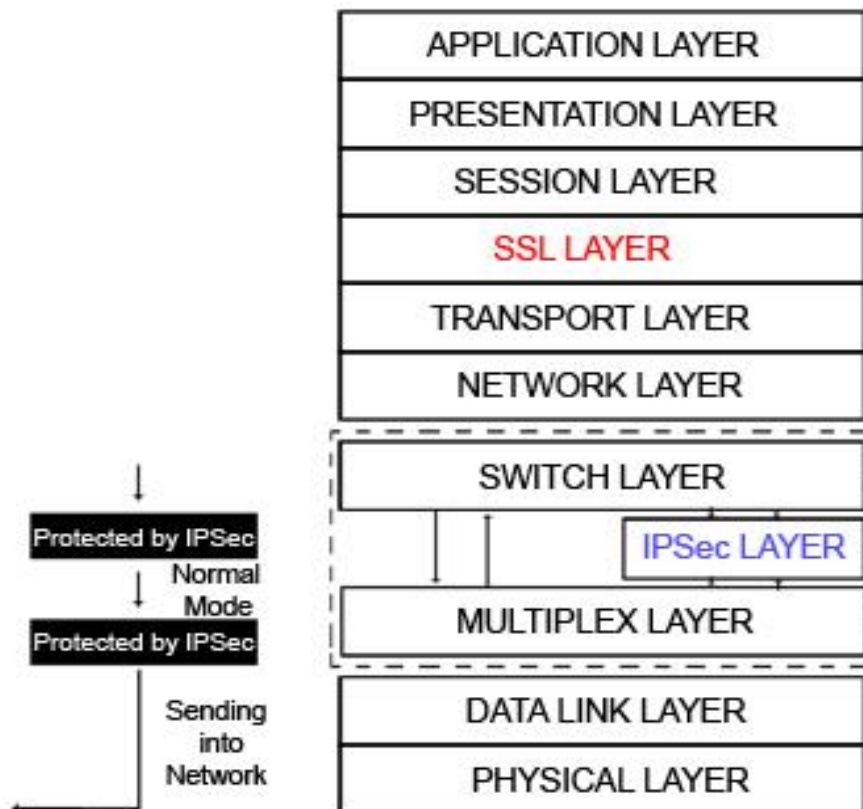


Figure 4.12 - Packet protected by IPSec layer forwarded through Hybrid VPN

The normal mode represents a transparent fashion when the packet protected by IPSec layer and passed through Hybrid VPN system in either sending or receiving cases.

4.3.3 Technical Details for the Proposed Hybrid VPN System Phase

In this phase, all clients requests securely received by the proposed Hybrid VPN system which backward the secure packets through the physical layer and data link layer respectively.

Figure (4.13) illustrate the ensemble system mechanism when checked the header of the received packet to find a label header. If the received packet was protected by SSL layer from the source, the one byte size label header has been founded. The ensemble system mechanism would be entered in the switching mode.

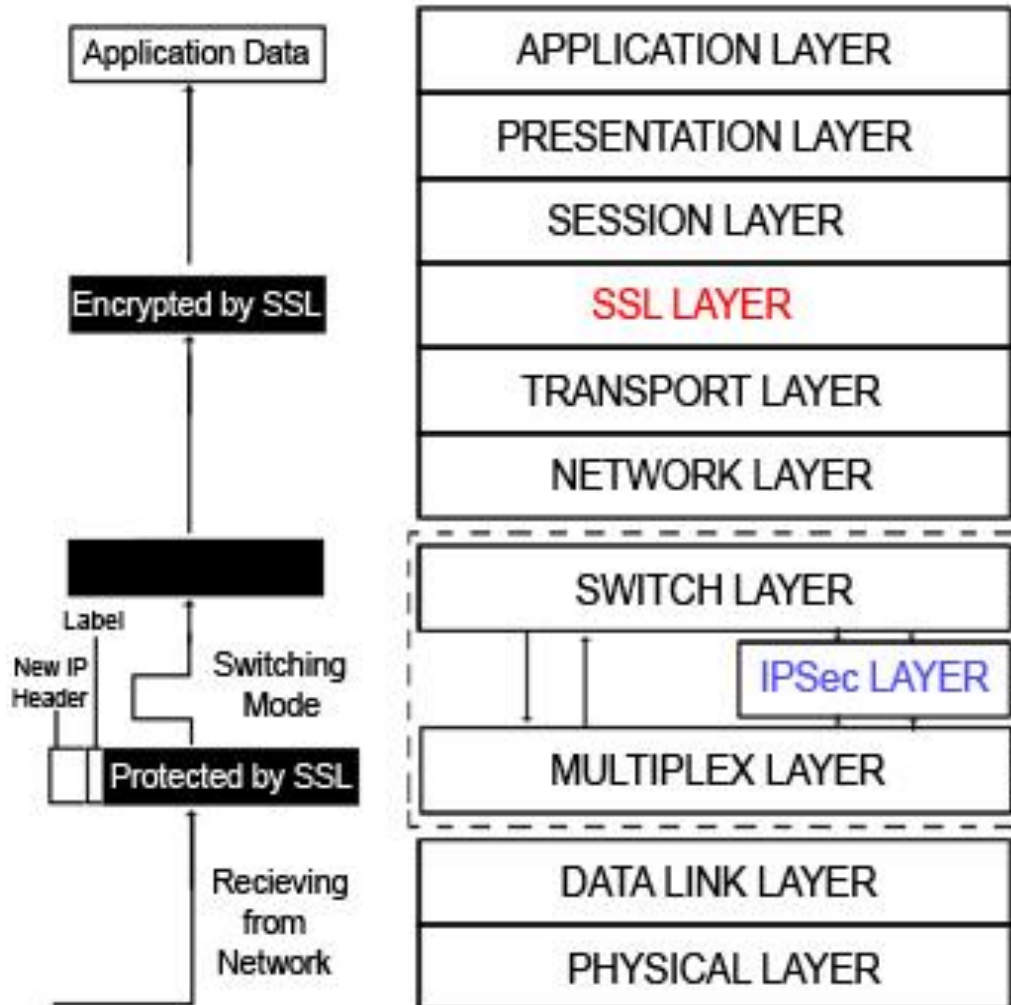


Figure 4.13 - Protected packets by SSL layer received to Hybrid VPN

The packet switch from multiplex layer to the switch layer without passed through IPSec layer, and then backward to the upper layers to receive by SSL layer which encrypted the packet and extracted the payload data that backward to reach the application layer.

If the received packet was protected by IPSec layer from the source, the one byte size label header has not found. The ensemble system mechanism would be entered in the normal mode.

Figure (4.14) illustrate the packet that encrypted by IPSec layer and extracted the payload data, and then backward to the upper layers through the switch layer which interacted as transparent layer.

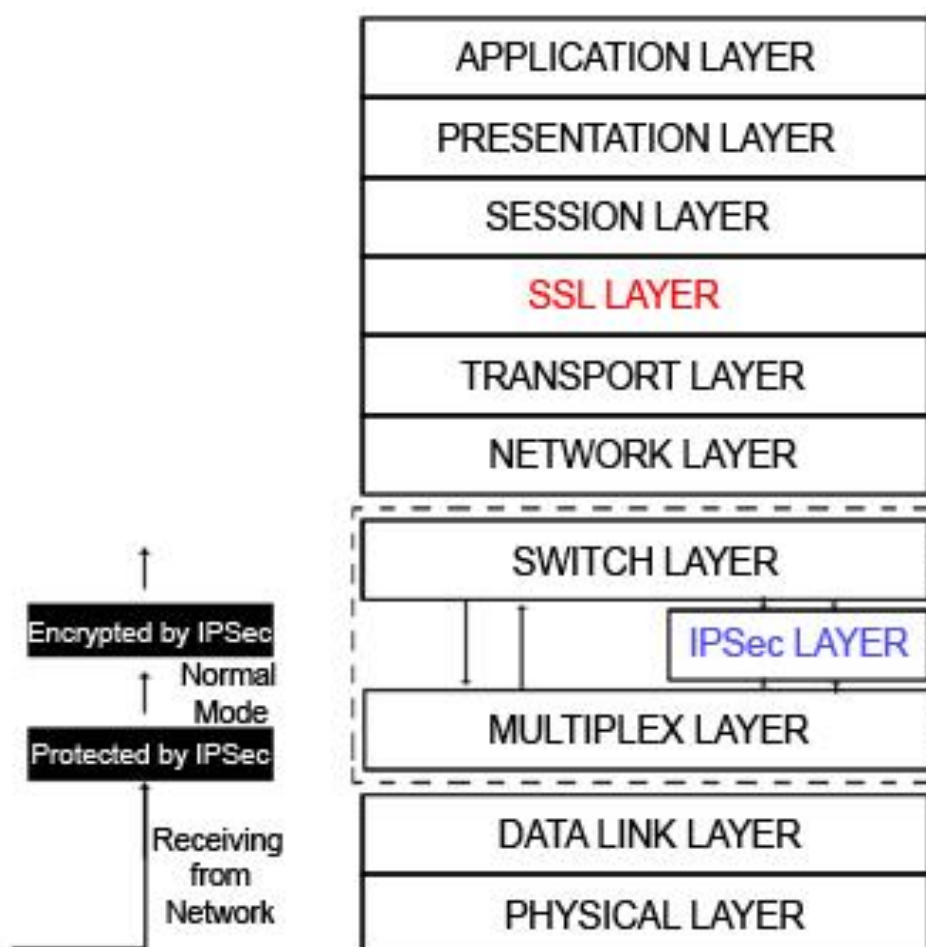


Figure 4.14 - Protected packets by IPsec layer received to Hybrid VPN

All the delivered packets that were sent under the IPsec layer would be buffered during the ensemble system mechanism were in switching mode. After switching processes currently finished, the mechanism would return to the default normal mode to process the buffered packets.

4.3.4 Technical Details for the Server Phase

In this phase, the proposed Hybrid VPN system routes the received packets to the destination servers.

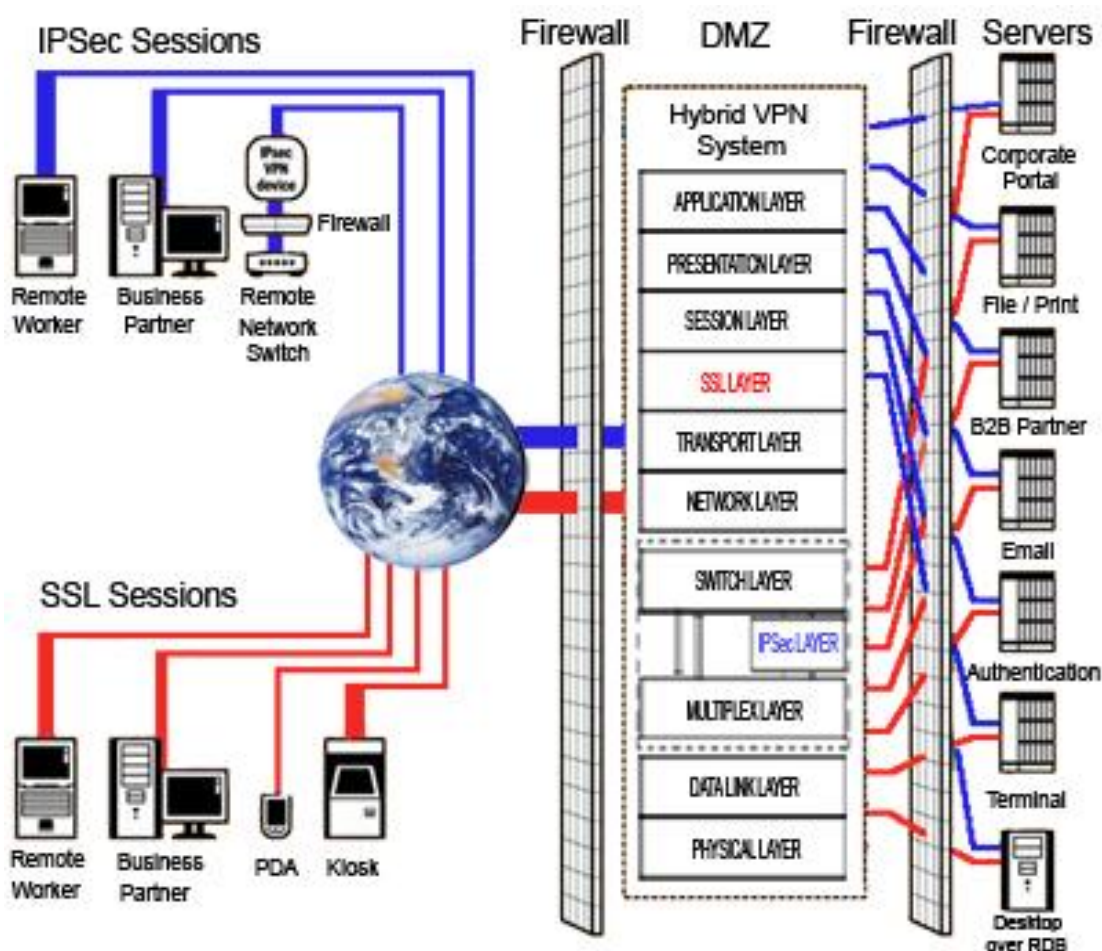


Figure 4.15 – IPsec and SSL sessions established via Hybrid VPN

These servers provide different types of services that need a secure connection either by IPsec protocol or SSL protocol such as authentication server, file server, print server, business-to-business (B2B) partner site server, terminal server, email server, corporate portal server, desktop over remote desktop protocol (RDP) server ... etc.

The secure session would be established between the destination server and source computer through Hybrid VPN system. The packets securely exchange over server client model by IPSec connections and SSL connections together through the one Hybrid VPN system. Figure (4.15) illustrate the secure session created over IPSec layer or SSL layer for the specified application on certain server.

4.4 Algorithms and Implementations

This section presents the algorithms and the implementations of the proposed Hybrid VPN system.

4.4.1 Introduction

The ensemble system mechanism is substantial implemented in the Objective Caml programming language that dialect of ML. The use of ML supports a variety of optimizations that achieve faster communications than Horus which is a previous version of ensemble.

The ensemble system mechanism must be adapted to work properly with the sending and receiving packets algorithms of the proposed Hybrid VPN system. This adaptation will be in the adaptive applications that interact with the interface procedures which joined with the top and the bottom of the ML core.

The ML core has interface procedures such native C programming language interface whereby facilitates that has the same performance as the ML language. The adaptive applications developed in ML, C, C++, or Java.

For an adaptive application builder, the ensemble system mechanism provides a library of protocols that can be used for quickly building complex adaptive applications. An adaptive application registers event handlers with ensemble system mechanism interface procedures, and then the ML core handle the details of reliably sending and receiving packets, transferring state, implementing security, detecting failures, and managing reconfigurations in the ensemble system mechanism.

Figure (4.16) illustrate the virtual interaction model for the four levels of the ensemble system mechanism include the ML core level, interface procedures level, the adaptive applications level, and the network layers level.

All levels can be modified or rebuilt to experiment with new properties or change the performance characteristics of the ensemble system mechanism that makes it a very flexible platform on which to do research.

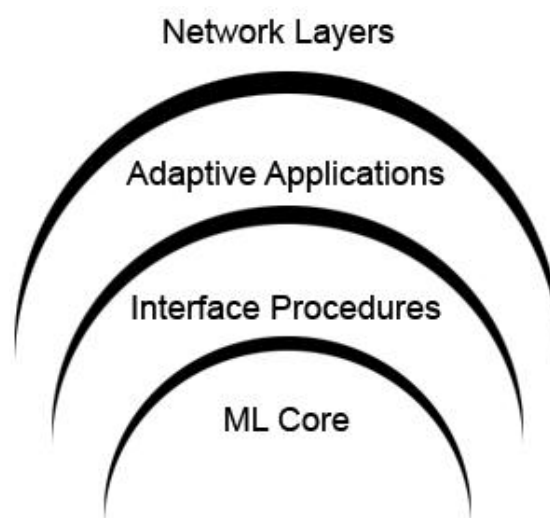


Figure 4.16 - Virtual interaction among ML core, interfaces, applications, and network layers

4.4.2 Algorithms

There are two algorithms that present a packet switching between network layers in both sending and receiving cases throughout different proposed Hybrid VPN system phases.

4.4.2.1 Sending Algorithm

Figure (4.17) illustrates the flowchart of the sending algorithm which has initial values of modes as inputs, and produces the final processes on the packet. Then this algorithm would be explained step-by-step through the pseudo code.

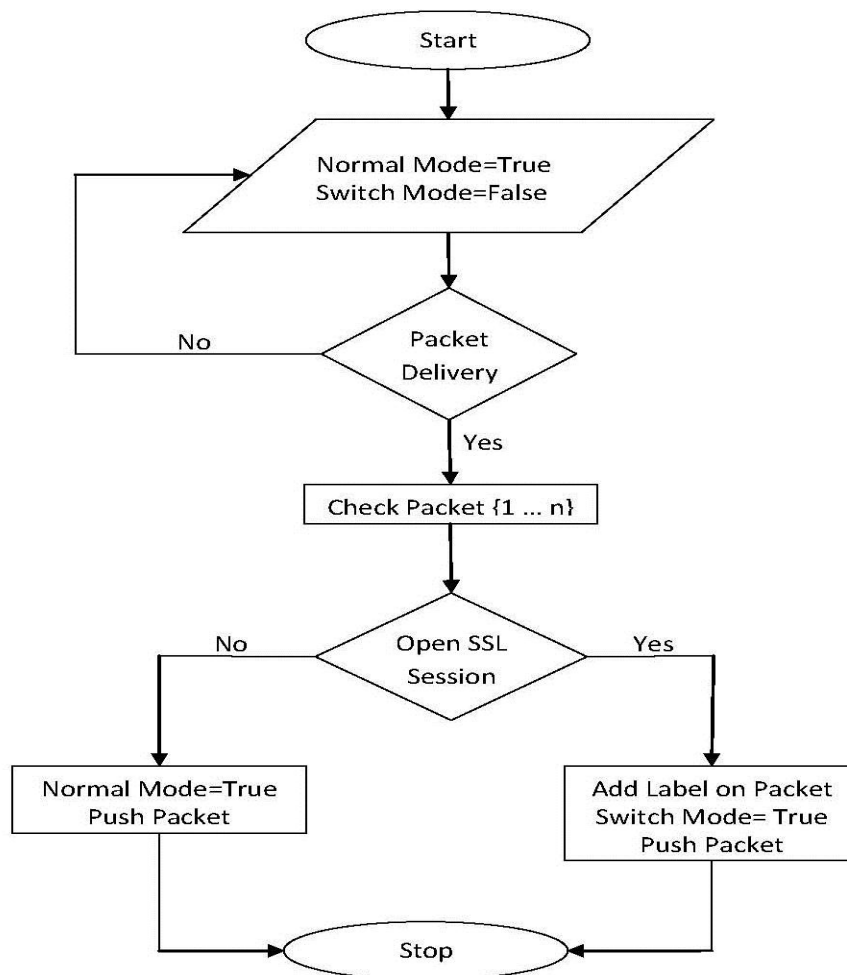


Figure 4.17 – Flowchart of sending algorithm

This algorithm presents the packet handling in the sending case. The default algorithm mode is the normal mode, i.e. all packets will handle with the IPSec layer. When the SSL open session acknowledgment packet arrived, the algorithm at this time will handle the packets with SSL layer that works in different steps showed as followed.

The first step is describing the initial values of the all related properties.

Describing the initial values of the all related properties

1. $deliv \leftarrow 0$ {packet delivering}
 2. $undeliv \leftarrow 0$
 3. $nmMod \leftarrow n-m$ {normal mode}
 4. $swMod \leftarrow 0$ {switching mode}
 5. $switching \leftarrow false$
 6. $pckcheck[1..n] \leftarrow false$
 7. $sslSnAck \leftarrow false$ {ssl session ack packet}
-

The second step is the checking of the SSL layer open session acknowledgment packet and also checks the close packet one, then the SSL layer will create the packets and push them to the switch layer below it.

Checking the acknowledgment packet

1. upon $sslSession(sslSnAck)$ do
 2. if $sslSnAck=Open$ {open ssl session}
 3. $pckcheck[1..n] \leftarrow true$
 4. push {push created packets to lower layer}
-

The third step is adding the SSL layer label to the all packets delivered to switch layer.

Adding label header to the delivered packet

1. upon $sslDeliver(deliv,flag)$ do
 2. $pckcheck[1..n] \leftarrow true$
 3. Add flag{add label header to delivered packet}
-

The fourth step is switching labeled packets from the switch layer to the multiplex layer without passing through the IPSec layer, and then pushes the packets to the data link layer.

Switching the labeled packet

1. upon changeMode(deliv,flag) do
 2. if deliv \leftarrow 1 ^ flag \leftarrow 1
 3. nmMod \leftarrow 0
 4. swMod \leftarrow sw
 5. switching \leftarrow true
 6. pckcheck[1..n] \leftarrow true
 7. push
-

The fifth step is the checking of the SSL layer session acknowledgment packet, if the session close packet delivered, the SSL session terminated. Then the algorithm goes back to the normal mode.

Checking the acknowledgment packet again

1. upon sslSession(sslSnAck) do
 2. if sslSnAck=Close {close ssl session}
 3. pckcheck[1..n] \leftarrow false
 4. nmMod \leftarrow nm
 5. swMod \leftarrow 0
 6. switching \leftarrow false
-

4.4.2.2 Receiving Algorithm

Figure (4.18) illustrates the flowchart of the receiving algorithm which has initial values of modes as inputs, and produces the final processes on the packet. Then this algorithm would be explained step-by-step through the pseudo code.

This algorithm presents the packet handling in the receiving case. All received packets will handle with the IPSec layer except those packets which have SSL label header, the algorithm at this

time enter in switching mode and pull the received packets to SSL layer for handling. These operations works in different steps showed as followed.

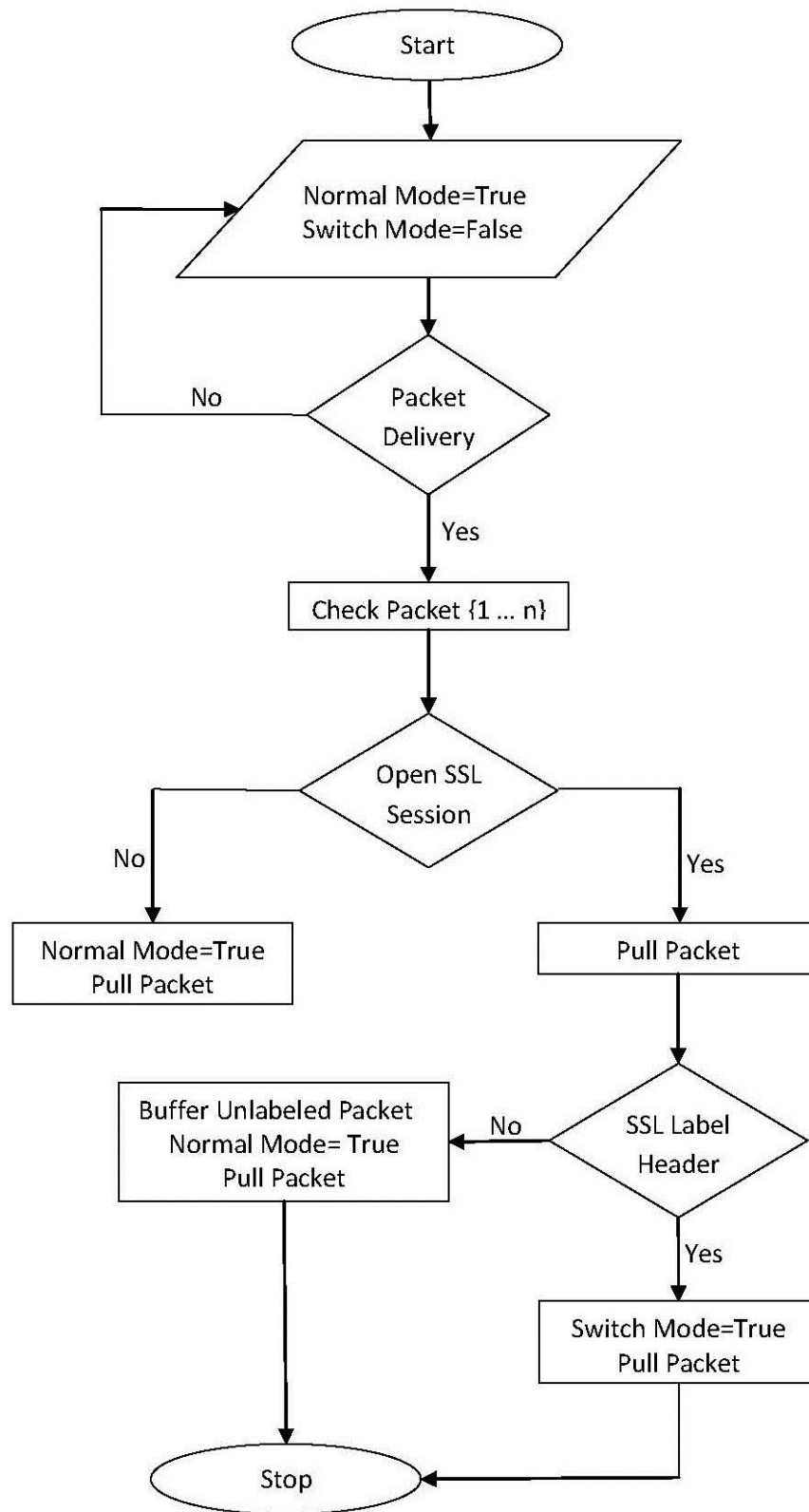


Figure 4.18 – Flowchart of receiving algorithm

The first step is the checking of the SSL layer open session acknowledgment packet and also checks the close packet one, then the algorithm enters in switching mode.

Checking the acknowledgment packet

1. upon sslSession(sslSnAck) do
 2. if sslSnAck=Open {open ssl session}
 3. nmMod \leftarrow 0
 4. swMod \leftarrow sw
 5. switching \leftarrow true
-

The second step is switching the received packets from the multiplex layer to the switch layer without passing through the IPSec layer after check the label header for each packet switched.

Switching the labeled packet

1. upon changeMode(deliv,flag) do
 2. if deliv \leftarrow 1 ^ flag \leftarrow 1
 3. pckcheck[1..n] \leftarrow true
 4. pull {pull delivered packets to upper layer}
-

The third step is buffering the packets which have not the labeled header, i.e. these non label packets will handle later by IPSec layer when the algorithm goes to normal mode.

Buffering the non label packet which handle by IPSec layer

1. upon changeMode(deliv,flag) do
 2. if deliv \leftarrow 1 ^ flag \leftarrow 0
 3. pckcheck[1..n] \leftarrow true
 4. buffer {buffer delivered packets to handle later by IPSec layer}
-

The fourth step is pulling the received packets from the switch layer to reach the SSL layer for handling, and then check the SSL layer close session acknowledgment packet which terminate the session. The algorithm goes back at this time to the normal mode.

Pulling the received packet to the SSL layer and checking the acknowledgment packet again

1. upon sslDeliver(deliv,flag) do
 2. pckcheck[1..n] \leftarrow true
 3. pull {pull delivered packets to upper layer}
 4. upon sslSession(sslSnAck) do
 5. if sslSnAck=Close {close ssl session}
 6. pckcheck[1..n] \leftarrow false
 7. nmMod \leftarrow nm
 8. swMod \leftarrow 0
 9. switching \leftarrow false
-

The fifth step is handling the non labeled packets in the buffer by the IPsec layer in the normal mode.

Handling the non labeled packet in the buffer

1. upon changeMode(buffer) do
 2. if buffer \leftarrow 1
 3. pckcheck[1..n] \leftarrow true
 4. pull {pull buffered packet through IPsec layer}
-

4.4.3 Implementations

The proposed Hybrid VPN system implemented on four levels, ML core level, Interface procedures level, adaptive applications level, and network layers level. These levels will be configuring to work properly with each other. On other hand, the ensemble system mechanism is structured as a client server model with a server providing group communication services through a socket based interface.

4.4.3.1 ML Core Level

The first level is the ML core which represents a server side of the client-server model of ensemble system mechanism which written mostly in the Objective Caml programming language.

The ML core uses the variety of identifiers for a variety of different purposes. Descriptions and type definitions of all identifiers can be found in the files corresponding to their names. Most of the different identifiers support a similar interface procedure for a variety of operations. Several of the identifiers are hidden, in the sense that the interface procedures hide the actual structure of the identifiers. All identifiers have a string of id function defined in a human readable description of their contents. The important ones of these identifiers will be represented as the following:

- 1) Endpoint identifiers: used to specify the points of channels connections which interior layers packets handled sending or receiving among them.
- 2) Connection identifiers: used to route interior layers packets to the precise destination channels by specify the exact destination endpoint. In proposed Hybrid VPN system, there are ten channels connections: two channels (send and receive) between multiplex layer and data link layer, two channels (send and receive) between multiplex layer and switch layer, two channels (send and receive) between multiplex layer and IPSec layer, two channels (send and receive) between IPSec layer and switch layer, and finally two channels (send and receive) between switch layer and network layer.
- 3) Protocol identifiers: used to specify the micro protocols id in ensemble stack and indentified protocols identifier in the proposed Hybrid VPN system which is IPSec layer protocols

- 4) and SSL layer protocols. Having identifiers for protocols is convenient because they can be passed around in interior layers packets.
- 5) Mode identifiers: used to specify the communication domain by corresponding mode. The proposed Hybrid VPN system has two mode identifier, normal mode identifier which is a default and switch mode identifier.
- 6) Stack identifiers: used to distinguish between the various communication domains that sending and receiving packets through protocols stack. Proposed Hybrid VPN system specifies two stack identifiers, stack bypass id for IPsec protocol and stack switch id for SSL protocol.

The ML core has various types of stack layers, each layer named by its function and saved in separate file. This file contains several description items of related layer such as protocol implemented by this layer, parameters required to initialize the layer, properties informal of the layer, sources file implemented in ML language, and lists of generated events of the layer with testing notes. The important layers that will be represented are listed below:

- 1) Tops layer: used to implement the total ordering protocol and control of switching process. Its source file found in tops.ml. The following pieces of ML code represent the main functions of this layer.

Part of header of file: declare the file name and variables.

```
1. let name = Trace.file! "TOPS"
2. type msg_type = bitfield
3. type header = NoHdr | YesTops | Filler
4. type 'abv msg = | Full of 'abv * lovecl.t | Null
5. type 'abv state = {
6.   buf : 'abv msg Queue.t array;      (* The buffer of delayed messages *)
7.   mutable pointer : int;              (* Current place in buf *)
8.   mutable num_msgs : int;            (* Number of messages in buf *)
9.   mutable blocked : bool             (* Am I blocked? *) }
10. let dump vf s = Layer.layer_dump_simp name vf s
11. let init _ (ls,vs) = {
12.   buf = Array.init ls.nmembers (fun _ -> Queue.create ());
13.   num_msgs = 0;}

```

Part of first function: handle the sending and receiving packets with upper layer.

```
1. let up_cast org msg = match msg with | Full(abv,dn_iov) ->
2.   up (castPeerlov name org dn_iov) abv
3.   Null -> () in
4. let next_up () = let msg = Queue.take s.buf.(s.pointer) in
5.   s.num_msgs <- pred s.num_msgs;
6.   up_cast s.pointer msg;
7.   s.pointer <- (succ s.pointer) mod ls.nmembers in
8. let handle_msg msg org = Queue.add msg s.buf.(org);
9.   s.num_msgs <- succ s.num_msgs;
10. while true do next_up () done;
11. with Queue.Empty -> () in
12. let up_hdlr ev abv hdr = match getType ev, hdr with

```

Part of second function: handle the data of the packets.

```
1. ECast iovl, YesTops -> handle_msg (Full(abv, iovl)) (getPeer ev);
2. -> up ev abv and uplm_hdlr ev hdr = match getType ev,hdr with
3.   ECast iovl, Filler -> handle_msg Null (getPeer ev) ;
4.   lovecl.free iovl
5.   -> failwith unknown_local and upnm_hdlr ev = match getType ev with
6.   EBlock -> s.blocked <- true;
7.   upnm ev | EExit -> Array.iter (funq -> Queue.iter (function | Null ->() |
8.     Full(abv, iov) -> lovecl.free iov) q; Queue.clear q; ) s.buf ;
9.   upnm ev | ETimer -> if s.num_msgs > 0
10.  && not s.blocked && s.pointer = ls.rank then

```

```

11.Begin log (fun () -> "Filler");
12.dnlm (castEv name) Filler;
13.handle_msg Null ls.rank;
14.end;

```

Part of Third function: Flush up all packets that are still in the buffer.

```

1. EView -> while s.num_msgs > 0 do
2. while true do next_up () done;
3. with Queue.Empty -> s.pointer <- (succ s.pointer) mod ls.nmembers;
4. EDump -> ( dump vf s ; upnm ev ) | -> upnm ev and dn_hdlr ev abv =
    match
5. ECast iovl when not (getNoTotal ev) -> log (fun () -> "multicasting") ;
6. if ls.nmembers > 1 then (dn ev abv YesTops;
7. handle_msg (Full(abv,(lovecl.copy iovl))) ls.rank; ) else
8. up (Event.set name ev [Peer ls.rank]) abv
9. ESend iovl -> let dest = getPeer ev in
10.if dest = ls.rank then (up (sendPeerlov name ls.rank iovl) abv
11.) else ( dn ev abv NoHdr)
12.-> dn ev abv NoHdr and dnm_hdlr = dnm in
13.let l args vs = Layer.hdr init hdlrs None NoOpt args vs
14.let _ = Layer.install name

```

2) Bottom layer: used to interact directly with the communication transport by sending and receiving packets, scheduling and handling timeouts. Its source file found in bottom.ml. The following pieces of ML code represent the main functions of this layer.

Part of header of file: declare the file name and variables.

```

1. let name = Trace.filel "BOTTOM"
2. type header = | NoHdr | Unrel
3. let string_of_header = function | NoHdr -> "NoHdr" | Unrel -> "Unrel"
4. type state = { mutable all_alive : bool ; alarm : Alarm.t ;
5. mutable failed : bool Arrayf.t ; mutable enabled : bool }
6. let init _ (ls,vs) = { alarm= Alarm.get_hack () ;
7. failed = ls.falses ; all_alive = true ; enabled = true }

```

Part of first function: handle the sending and receiving packets with lower layer.

```
1. let dump = Layer.layer_dump name (fun (ls,vs) s -> []
2. sprintf "rank=%d nmembers=%d\n" ls.rank ls.nmembers ;
3. sprintf "enabled=%b\n" s.enabled ; sprintf "all_alive=%b\n" s.all_alive ;
4. sprintf "failed=%s\n" (Arrayf.bool_to_string s.failed) [])
5. let hdlrs s ((ls,vs) as vf)
6. {up_out=up;upnm_out=upnm;dn_out=dn;dnlm_out=dnlm;dnnm_out=dn
   nm} =
7. let failwith = layer_fail dump vf s name in
8. let log = Trace.log2 name ls.name in
9. let got_merge ev typ vs abv = if true || s.enabled then (
10. let time = Alarm.gettime s.alarm in
11. up (set name ev[Type typ;Time time;ViewState vs]) abv
12.) else ( free name ev ) in
13. let up_hdlr ev abv hdr = match getType ev, hdr with
14. | (ECast iov |ESend iov), NoHdr -> if s.all_alive
15. || not (Arrayf.get s.failed (getPeer ev))
16. Then up ev abv else lovecl.free iov
17. | ECast iov,Unrel -> if Arrayf.get s.failed (getPeer ev) then lovecl.free
   iov
18. Else up (set name ev[Type (ECastUnrel iov)]) abv
19. | ESend iov,Unrel -> if Arrayf.get s.failed (getPeer ev) then lovecl.free
   iov
20. else up (set name ev[Type (ESendUnrel iov)]) abv
```

Part of second function: packets scheduling and handling timeouts.

```
1. | ETimer | EAsync | EGossipExt ->
2. if s.enabled then upnm ev else free name ev
3. | _ -> failwith "bad upnm event" and dn_hdlr ev abv =
4. if s.enabled then ( match getType ev with | EAuth | ECast _ | ESend _ -
   > dn
5. ev abv NoHdr | ECastUnrel _ | ESendUnrel _ -> dn ev abv Unrel
6. | _ -> failwith "bad down event[1]" ) else ( free name ev )
7. and dnnm_hdlr ev = match getType ev with
8. | EGossipExt | EGossipExtDir -> if s.enabled then dnnm ev
9. Else free name ev | ETimer -> if s.enabled then ( let alarm = getAlarm
   ev in
10. if Time.is_zero alarm then ( upnm (create name ETimer[Time
   (Alarm.gettime
11. s.alarm)]) ) else (dnnm ev ) ) else (free name ev ) | EFail -> s.failed <-
12. getFailures ev ; s.all_alive <- Arrayf.for_all not s.failed ;
13. upnm ev | EExit -> if s.enabled then ( s.enabled <- false ;
14. s.all_alive <- false ; s.failed <- Arrayf.create ls.nmembers true ;
15. dnnm ev ; let time = Alarm.gettime s.alarm in
```

- 16.
 17. upnm (create name EExit[Time time]) ;
 - 18.) else (log (fun () -> "2nd Exit (ok in some cases)") ; free name ev)
 19. let l args vs = Layer.hdr init hdlrs None (FullNoHdr NoHdr) args vs
 20. let l2 args vs = Layer.hdr_noopt init hdlrs args vs
 21. let _ = Layer.install name
-

3) Bypass layer: used to send packets from bypass code without going through the stack of layers. Its source file found in bypass.ml. The following pieces of ML code represent the main function of this layer.

Part of header of file: declare the file name and variables.

1. let name = Trace.filel "BYPASS"
 2. let failwith s = failwith (Util.failmsg name s)
 3. let ts_done = Timestamp.add "BYPASS:recv_done"
 4. let procure a = Layer.procure_hide a type 'abv cbypass_env = {
 5. c_appl_intf_recv_cast:origin->lovec.t ->(lovec.t, lovec.t) Appl_intf.action list ;
 6. c_bottom_alive : bool array ; c_bypass_xmit : int -> lovec.t -> unit ;
 7. c_handle_action : (lovec.t, lovec.t) Appl_intf.action -> unit ;
 8. c_iq_add : 'abv lq.t -> 'abv -> unit ;
 9. c_iq_opt_insert_check_doread : 'abv lq.t -> int -> 'abv -> bool ;
 10. c_mnak_casts : 'abv lq.t ; c_mnak_my_casts : 'abv lq.t ;
 11. c_origin : rank ; c_rank : rank ; c_stable_my_row : seqno array ;
 12. c_stable_ncasts: seqno array ;
 13. c_top_appl_s : Top_appl.state ; c_no_bypass : seqno -> lovec.t -> unit
-

Part of function: handle the sending and receiving packets with upper layer.

1. external cbypass_receive :
2. 'abv cbypass_env -> seqno -> lovec.t -> unit = "bypass_receive"
3. and act_next = ref Time.zero let getdns_l dn_ref dn_insert inher =
4. let l {empty_lout=empty;up_lout=up;dn_lout=dn} =
5. dn_ref := dn ; let dn ev msg = if not (dn_insert ev msg) then dn ev msg in
6. let up ev abv = if getType ev = ETimer & !verbose & (getTime ev) >=
7. !act_next then (act_next := Time.add !act_next (Time.of_int 1) ;
8. !act_send_normal !act_send_bypass

```

9. !act_recv_normal !act_recv_bypass_fail !act_recv_bypass_succ
10. !act_fail_mnak_seqno ) ; up ev abv in let rank= !s.rank in
11. let bypass_xmit_r= ref (fun _ _ -> failwith "byp_cast unset") in
12. let bypass_disable_r = ref (fun () -> failwith "byp_disable unset") in
13. let bypass_disable () = !bypass_disable_r () in
14. let dn_nobyp_r= ref (fun _ _ -> failwith "sanity:dn_nobyp_r")
15. and dn_catch_r= ref (fun _ _ -> failwith "sanity:dn_catch_r") in
16. let dn_nobyp ev msg= !dn_nobyp_r ev msg
17. and dn_catch ev msg= !dn_catch_r ev msg in
18. let {empty_lout=empty;up_lout=up;dn_lout=dn} = hdlrs in
19. let dn ev = if getType ev = ELeave then bypass_disable () ; dn ev in
20. let hdlrs = {empty_lout=empty;up_lout=up;dn_lout=dn} in
21. let hdlrs = sub_l hdlrs in
22. let mnak_seqno = lq.tail mnak_my_casts in
23. !bypass_xmit_r mnak_seqno iov ; incr act_send_bypass ;
24. stable_ncasts.(rank) <- succ stable_ncasts.(rank) ;(* BUG!!!! *)
25. lq.add mnak_my_casts (Mnak.Compressedlov(iov)) ;
26. True in dn_catch_r := (fun ev abv -> match getType ev, abv with
27. | ECast, Empty -> ( let iov = lovecl.flatten (getlov ev) in
28. let res = bypass_ECast iov in (* returns bool *)
29. Res ) | _ -> false ) ; let sched = Appl.root_sched in
30. let enqueue a b = Sched.enqueue sched (fun () -> dn_nobyp a b) in
31. let rec handle_actions = function | [] -> ()
32. | hd::tl -> (begin match hd with handle_actions tl ) in
33. let bypass_receive kind origin =
34. let mnak_casts = mnak_casts_v.(origin) in
35. fun mnak_seqno msg_iov -> ( if bottom_alive.(origin)
36. && top_appl_s.Top_appl.state = Top_appl.Normal
37. && lq.opt_insert_check_doread mnak_casts mnak_seqno
38. Mnak.Compressedlov(msg_iov)
39. then ( handle_actions (appl_intf_recv_cast origin msg_iov) ;
40. stable_ncasts.(origin) <- succ stable_ncasts.(origin) ;
41. if mnak_seqno >= stable_my_row.(origin) then
42. stable_my_row.(origin) <- succ mnak_seqno ;
43. ts_done () ; incr act_recv_bypass_succ
44. ) else if bottom_s.Bottom.enabled then ( incr act_recv_bypass_fail ;
45. let abv = expand_bottom mnak_seqno in
46. let l = (l : Layer.state -> ('abv,'abv h9,unit) Layer.basic)
47. let _ = Elink.layer_install name

```

4) Credit layer: used to buffer the packets accord of the based flow control. Its source file found in credit.ml. The following pieces of ML code represent the main functions of this layer.

Part of header of file: declare the file name and variables.

```

1. let name = Trace.file! "CREDIT"
2. type 'a state = { mutable rtotal: int ;      (* total resource *)
3. mutable rremain: int ;      (* unallocated resources *)
4. mutable ntogive: int array;(* # credit to give at a time *)
5. mutable whentogive: int array;(* give more credit when # credit
   remaining *)
6. mutable pntogive: int array;(* # credit to piggyback at a time *)
7. mutable pwhentogive: int array;(* piggyback credit when # credit
   remaining *)
8. crgiven: int array;(* credit given to other members *)
9. mutable ntoask: int ;(* # credit to ask at a time *)
10. mutable whentoask: int ;(*ask for more credit when this # credit
   remaining *)
11. mutable pntoask: int ;(* # credit to ask at a time *)
12. mutable pwhentoask: int ;(*ask for more credit when this #credit
   remaining *)
13. credit: int array ;(* credit given by other members *)
14. buffer: 'a Queue.t ;(* msgs yet to be casted *)
15. mutable nbuffered: int ;(* # msgs in buffer *) sweep: Time.t ;
16. mutable next_sweep : Time.t ; mutable msent_data: int ;(* # data msgs
   sent *)
17. mutable nblocked: int ;(* # data msgs blocked *)
18. msent : int array ;(* # credit reply sent *)
19. mrcvd : int array ;(* # credit reply rcvd *)
20. csent : int array }(* # credit given via reply *)

```

Part of function: buffer the packets accord of based flow control.

```

1. let init (rtotal,ntoask,whentoask,pntoask,pwhentoask,sweep) (ls,vs) =
2. let nmembers= ls.nmembers in
3. let gwhen = if nmembers > 1 then rtotal / (nmembers - 1) else 0
4. let hdlrs s ((ls,vs) as
5. vf){up_out=up;upnm_out=upnm;dn_out=dn;dnlm_out=dnlm;dnnm_out=
   dnnm
6. } = let ack = make_acker name dnm in and give_credit sender style = (
7. let ntogive = s.ntogive.(sender) and whentogive =
   s.whentogive.(sender)
8. and pntogive = s.pntogive.(sender) and pwhentogive =
   s.pwhentogive.(sender)
9. and crgiven = s.crgiven.(sender) and remain = s.rremain
10. and send_credit n = ( s.rremain <- s.rremain - n ;
11. s.crgiven.(sender) <- s.crgiven.(sender) + n ;
12. s.csent.(style) <- s.csent.(style) + n ; n ) in
13. if remain > 0 & ntogive > 0 & (style = ct_piggy or crgiven<= whentogive)
   then

```

```

14. ( if s.rremain >= ntogive then send_credit ntogive
15. Else send_credit remain )
16. if remain > 0 & ntogive > 0 & crgiven <= whentogive then (
17. if s.rremain >= ntogive then send_credit ntogive
18. else send_credit remain ) if remain > 0 & ntogive > 0
19. & ((style = ct_piggy & crgiven <= pwhentogive)
20. or crgiven <= whentogive) then ( if style = ct_piggy & s.rremain >=
    pntogive
21. then send_credit pntogive
22. else if (not (style = ct_piggy)) & s.rremain >= ntogive then
23. send_credit ntogive else send_credit remain
24. ) else 0 ) in let give_gcredit style = (
25. let crarray = array_create name ls.nmembers 0 and max = ref 0 in
26. for i = 0 to pred ls.nmembers do
27. if i <> ls.rank then ( let cr = (give_credit i style) in crarray.(i) <- cr ;
28. if cr > !max then max := cr ) done ;
29. if !max > 0 then ( CR_Many(crarray) )
30. else NoHdr ) and avail_credit () = ( let rec check_credit answer n =
31. if n >= ls.nmembers then answer
32. else if (not (n = ls.rank)) & (s.credit.(n) < answer) then
33. check_credit s.credit.(n) (n + 1)
34. else check_credit answer (n + 1) in if ls.rank = 0 then (
35. if ls.nmembers > 1 then check_credit s.credit.(1) 2
36. else 0 ) else check_credit s.credit.(0) 1 )
37. let do_cast msg = (use_credit 1 ;
38. let hdr = (give_gcredit ct_piggy) in
39. ( match hdr with | CR_Many-> s.msent.(ct_piggy)<- s.msent.(ct_piggy)
    + 1
40. | _ -> () ) ; dn (castEv name) msg hdr )
41. and cast_buffered ntosend = ( for i = 1 to ntosend do
42. let msg = Queue.take s.buffer in if i = 1 then (let hdr = (give_gcredit
    ct_piggy)
43. in ( match hdr with | CR_Many-> s.msent.(ct_piggy)<-
    s.msent.(ct_piggy) + 1
44. | _ -> () ) ; dn (castEv name) msg hdr )
45. let _ = Layer.install name

```

5) Application layer: used as interface definition socket between the interior ML core layers and the interface procedures. Its source file found in appl.ml. The following pieces of ML code represent the main functions of this layer.

Part of header of file: declare the file name and variables.

1. let name = Trace.filel "APPLICATION"
 2. type rank= int; type view= Endpt.id list; type origin = rank; type dests = rank
 3. array; type control =| Leave | Prompt | Suspect of rank list | XferDone | Rekey
 4. of bool | Protocol of Proto.id | Migrate of Addr.set | Timeout of Time.t | Dump
 5. | Block of bool | No_op | type ('cast_msg,'send_msg) action = | Cast of
 6. 'cast_msg | Send of dests * 'send_msg | Send1 of rank * 'send_msg | Control of
 7. control | type cast_or_send = C | S | type blocked = U | B | type 'msg naction =
 8. ('msg,'msg) action | type 'msg handlers = { flow_block : rank option * bool ->
 9. unit ; block : unit -> 'msg naction array ; heartbeat : Time.t -> 'msg naction
 10. array ; receive : origin -> blocked -> cast_or_send -> 'msg -> 'msg naction
 11. array ; disable : unit -> unit }
 12. type 'msg full = { heartbeat_rate : Time.t ; install : View.full -> ('msg naction
 13. array) * ('msg handlers) ; exit : unit -> unit }
-

Part of function: handle the some of interactive operations.

1. let run ()= Arge.parse [] (Arge.badarg name)"mtalk:multiperson talk program";
 2. let view_state = Appl.default_info "mtalk" in
 3. let alarm = Alarm.get_hack () in
 4. let host = gethostname () in
 5. let host = string_of_inet (inet_of_string host) in
 6. sprintf "%s@%s" (getlogin ()) host
 7. with _ -> view_state.name in
 8. let interface = intf name alarm in Appl.main_loop ()
 9. let _ = Appl.exec ["mtalk"] run end;
-

4.4.3.2 Interface Procedures Level

The second level is the interface procedures level which represents a client side of the client-server model of ensemble system mechanism which written mostly in the native C

programming language.

In this level, the connection issues show between the client side and the server side. Pieces code below show the main operations between both sides.

Initialize the connection structure that encapsulates a local socket to the server side and test the pending packets then it will be joining with server operations as a member of structure as shown below.

Part of connection initialize and join with server operations.

```
1. ens_conn_t *ens_Init(void);
2. typedef enum ens_rc_t ; ENS_OK = 0, ENS_ERROR = 1; ens_rc_t;
3. ens_rc_t ens_Poll(ens_conn_t *conn, int milliseconds, int
   *data_available);
4. ens_rc_t ens_Join(ens_conn_t *conn, ens_member_t *memb,
   ens_jops_t *ops,
5. void *user_ctx) ; typedef struct ens_jops_t
6. char group_name[ENS_GROUP_NAME_MAX_SIZE] ; /* The group
   name */
7. char properties[ENS_PROPERTIES_MAX_SIZE] ; /* The set of
   properties */
8. char params[ENS_PARAMS_MAX_SIZE] ; /* The set of parameters */
9. char princ[ENS_PRINCIPAL_MAX_SIZE] ; /*The principal
   name(security)*/
10.int secure ; /* Encryption and authentication*/
11.ens_jops_t ; /* Joining operation completed*/
```

After joining to the server operations, there are several processes that are allowed on their connection such as sending and receiving packets between client and server as shown below.

Part of several operations between client and server after joining.

```
1. ens_rc_t ens_Leave(ens_member_t *memb) ;
2. ens_rc_t ens_Cast(ens_member_t *memb, int len,char *buf) ;
3. ens_rc_t ens_Send(ens_member_t *memb, int num_dests, int *dests,
   int len,
4. char* buf) ; ens_rc_t ens_Send1(ens_member_t *memb, int dest, char*
   buf) ;
```

Pieces code below show the management of the buffer and memory space in the client side by several operations such as suspension, resuming, blocking and allocating the flow of the packets.

Part of several operations of buffer management.

```

1. ens_rc_t ens_Suspect(ens_member_t *memb, int num, int *suspects);
2. ens_rc_t ens_BlockOk(ens_member_t *memb) ;
3. typedef enum ens_up_msg_t
4. VIEW = 1, /* A new view has arrived from the server. */
5. CAST = 2, /* A multicast message */
6. SEND = 3, /* A point-to-point message */
7. BLOCK = 4, /* A block request, prior to the installation of a new view */
8. EXIT = 5 /* A final notification that the member is no longer valid */
9. ens_up_msg_t; typedef struct ens_msg_t
10.ens_member_t *memb; /* endpoint this message belongs to */
11.ens_up_msg_t mtype ; /* message type */ union
12.struct /* The variant for VIEW: */
13.int nmembers; /* the number of members in a view */
14.view; struct /* The variant for a point-to-point message */
15.int msg_size; /* length of a bulk-data */ send;
16.struct /* The variant for multicast message */
17.int msg_size; /* length of a bulk-data */ cast; ens_msg_t;
18.ens_rc_t ens_RecvMetaData(ens_conn_t *conn, ens_msg_t *msg);
19.ens_rc_t ens_RecvView(ens_conn_t *conn, ens_member_t *memb,
20.ens_rc_t ens_RecvMsg(ens_conn_t *conn, int *origin, char *buf);

```

4.4.3.3 Adaptive Applications Level

The third level is the adaptive applications level which represents the implemented previous switching algorithms by converting them from pseudo code to the native C programming language due to high performance communications.

In this level, the implementation of the sending and the receiving adaptive applications represent the interaction between these adaptive applications and the interface procedures level.

Pieces of codes below show the implementation of different operations for different steps.

Initialize the values of parameters responsible on the control of beginning the sending or receiving operations for the packets. The timer function control the different events that associated by the internal connection channels.

Part of initialization of the parameters' values and timer function

1. private void start Reconfiguration () { switching = true ; checked = 0 ;
 2. check = new boolean [vs.view.length] ; isFirstPck = true ;
 3. nullFirst = false ; PckFirst = false ;
 4. NullEventTimer nullt = new NullEventTimer (NULL_TIMEOUT,
 5. "NullEventTimer", currentChannel, Direction.DOWN, this,
 6. EventQualifier.ON) ; nullt.go () ; }
-

The following codes show the label header handling, the packet processes through sending and receiving cases in different steps, management of connection channels.

Part of label header handling, packet processes, and channels management

1. private void handl eGroupSendableEvent (GroupSendableEvent event) {
2. if (event.getDir() == Direction.DOWN) { event.getPacket().pushLong(loc a
3. ISN); if (switching && isFirstPck && !nullFirst) {
4. event.getPacket().pushBoolean(true) ; isFirstPck = false ; PckFirst = true; }
5. else event.getPacket().pushBoolean(false);
6. if(switching) { GroupSendableEventclone = (GroupSendableEvent)
7. event.cloneEvent() ; clone.setChannel (otherChannel) ;
8. clone.setSource (this); clone.init() ; clone.go () ;}
9. event.setChannel (currentChannel); event.setSource (this); event.init () ;
10. event.go () ; } else { boolean flag = event.getPacket ().popBoolean () ;
11. if(event.getChannel() == currentChannel)
12. processCurrent(event , flag); elseif(event.getChannel() == otherChannel)
13. processOther(event); if(switching && checked >= actives) endSwitch() ; } } }

```

14. private void processCurrent(GroupSendableEvent event, boolean flag) {
15. EventContainer cont = new EventContainer(event.orig, event.getPacket(
    ), null);
16. if (switching && flag) { check [event.orig] = true; checked ++; }
17. tryDeliver(event, flag); if (otherList.contains (cont)) otherList.remove
    (cont); }
18. private void processOther (GroupSendableEvent event) {
19. EventContainer cont = new
    EventContainer(event.orig, event.getPacket());
20. if (cont.sn > lastDelivered [event.orig]) otherList.add (cont); }

```

Finally, the following codes show the part of buffer management in sending and receiving cases. In addition, part of sessions termination processes.

Part of buffer management and sessions termination processes

```

1. private void clean Buffers ( ) { Iterator it = otherList.iterator ( );
2. while (it.hasNext ( ) ) { EventContainer cont = (EventContainer) it.next (
    );
3. cont.event.setChannel(applChannel); cont.event.setSource(this);
4. cont.event.init( ); cont.event.go( ); lastDelivered [cont.source] = cont.sn;
    } }
5. private void endSwitc ( ) { clean Buffers ( );
6. if (currentChannel == defaultChannel) { currentChannel =
    secondChannel;
7. otherChannel = defaultChannel; } else { currentChannel =
    defaultChannel;
8. otherChannel = secondChannel; } switching = false; }

```

4.4.3.4 Network Layers Level

The last level is the network layers level which represent the highest level of the virtual interaction model in the ensemble system mechanism which embedded in the proposed Hybrid VPN system.

At this level, the SSL layer has not reconfigured because it lies far away over TCP layer. But the IPSec layer is wrapper by ensemble system mechanism layers. The interaction between IPSec layer and the sending and the receiving adaptive applications which its role is interact with interface procedures through application layer to reach

the ML core layer configurations.

The IPSec layer lies usually over data link layer and interfaces to the network layer, in the proposed Hybrid VPN system reconfigure the communication channels to become between IPSec layer and switching layers in hand, then between switching layers and data link layer with network layer in the other hand.

The configuration settings connect the switching layers in the ML core level with the tops and bottom layers. The IPSec layer interact with bypass layer directly through the adaptive applications and interface procedures configurations.

The packets which handle with IPSec layer pass transparency through the virtual interaction model in the ensemble system mechanism. Pieces code below show the main configuration of the IPSec interaction operations.

Part of several operations of packet handling and buffer management.

```
1. #include <stdio.h>
2. #include <ctype.h>
3. #include <assert.h>
4. int do_file( char *, char * ) ;
5. main(int argc, char **argv) {
6. FILE *in ; char *programe; long lsize ; size_t size, nread;
7. char *buffer, *bufend ; programe = *argv ; if( argc != 2 )      {
8. fprintf(stderr,"usage: %s input-file\n", programe); exit(1) ; }
9. if( (in = fopen(argv[1],"r")) == NULL ) {
10. fprintf(stderr,"%s Can't open input file\n", programe); exit(2) ; }
11. if( (lsize = fseek(in, 0L, SEEK_END)) < 0L ) {
12. fprintf(stderr,"%s fseek() fails\n", programe); exit(3) ; }
13. lsize = ftell(in) ; rewind(in) ; size = (size_t) lsize ;
14. if( lsize != (long) size ) { fprintf(stderr,"%s file too large\n", programe);
15. exit(4) ; } if( (buffer = (char *) malloc(size)) == NULL ) {
16. fprintf(stderr,"%s malloc() failed\n", programe); exit(5) ; }
17. bufend = buffer + size ; if( (nread = fread(buffer, size, 1, in)) != 1 ) {
18. fprintf(stderr,"%s fread() failed\n", programe); exit(6) ; }
19. do_file(buffer,bufend); } do_file(char *start, char *end) {
```

20.
21. char *p, *q ; int value ; for (p = q = start ; p < end ; q = (q<end) ? (q+1) :
q) {
22. if(q < p) continue ; if(isalnum(*q)) continue ;
23. switch(*q) { case '!': case '_': case '-': case '(': continue ; break ; default:
24. for(; p <= q ; p++) putchar(*p); break ; case ')': value = do_name(p,q);
25. if(value) { p = q ; p++ ; } else for(; p <= q ; p++)
26. putchar(*p); break ; } } return 1 ; }

Chapter Five

Performance Analysis of the Proposed Hybrid IPSec SSL VPNs Technology

5.1 Introduction

In this chapter, the performance of the proposed solution is analyzed and compared with the separate IPSec VPN and SSL VPN technologies.

It would show a technical details of the performance analysis measurements such throughput, delay, and bandwidth consumption. Rather for, the description of experimental platforms, setups and collection of tests results. The objective behind the collection of tests results is to obtain a comprehensive set of measurements for the analysis and evaluation of the performance of IPSec, SSL, and proposed Hybrid VPN technologies infrastructure under various scenarios.

The first section of this chapter presents the basic performance measurements considered for this research. Following this is a section providing a detailed description of benchmark tests included all implementations, techniques, and approaches used in the setup of these experimental platforms.

The final section of this chapter takes the results comparison further by stating the detailed procedures through which the VPN technologies are performed dependent on the certain measurements. It also provides the collected tests data of various measurements in a convenient tabular format such as tables and charts.

5.2 Performance Analysis Measurements

This section presents the performance analysis measurements used by this research. The comprehensive set of these measurements relating to VPN technology is considered. These measurements include the application throughput, round trip delay and the bandwidth consumption. The descriptions of these measurements show the particular definition, the reasons of usage and detailed parameters.

5.2.1 Throughput Measurement

The throughput is the average amount of data payload transmitted and received over a sampling period between two points. The throughput metric is expressed in megabits per second (Mbps). This measurement reflects the data throughput rates available for TCP based services such as File Transfer Protocol (FTP) downloads or UDP based services such as audio streaming.

5.2.2 Delay Measurement

The delay measurement is the average time taken by a packet to complete one full round trip from the source to the destination and back. The delay metric is expressed in milliseconds (ms). This measurement describes how approximate speculations can be made on the regions where queues are formed. Such information may also point to various bottlenecks in the comparison VPN technologies. The spent time which consume during packet stay in the waiting state because delay would be decreased the overall performance of the system.

5.2.3 Bandwidth Consumption Measurement

This measurement is the amount of data that can be transferred over the network connection in a fixed amount of time. The bandwidth metric is expressed in bits per second (bps). This measurement affects the VPN technologies, because the bandwidth consumption can be a major element for a quantitative analysis due to delay of creation of the packets which may be severely impacted by available bandwidth. Also increasing the link bandwidth to solve the performance problem may be either increasing the financial costs away or focusing bandwidth expenditures on the wrong part of the VPN network.

5.3 Benchmarks

This section presents the experimental environment which includes hardware and software. In addition, the tests setup of the benchmark of the certain performance measurements with IPsec VPN, SSL VPN and proposed Hybrid VPN system technologies.

5.3.1 Instrumentation

This research used the minimal experimental hardware to reduce the financial cost, so the researchers who would reuse these tests can have benefits in their experimental environments.

Also in this research, the usage of the free and the open source software which widely available allow the large scale of modification and development to fix the experimental requirements. Rather for reducing costs of money and technical support.

5.3.1.1 Software and Hardware

The details of software and hardware for conducting the benchmark tests are as follows:

Software:

A. Server side:

- 1) Operating System: Red Hat Fedora Linux (kernel 2.6.14 - 1.1653).
- 2) Open SSL and Stunnel: Used for SSL VPN secure data exchange (version 0.9.8e).
- 3) Open Swan: Used for IPSec VPN secure data exchange (version 2.4.7).
- 4) Proposed Hybrid VPN system: Used for IPSec SSL VPNs secure data exchange (core ensemble mechanism system sever side version 2.01).
- 5) Web server: standard Apache (version 2.0.43).
- 6) Network Performance Tool (Netperf): Used for tests for end-to-end latency, unidirectional throughput and bandwidth efficiency (version 2.4.1).
- 7) Illustrative Performance Tool (Iperf): Used for observing any datagram loss and bandwidth consumption (version 2.0.2).
- 8) Network Input Output Tool (Netio): Used for observing performance while sending data with different packet sizes (version 1.26).
- 9) Firewall: Open source Seattle firewall (version 4.1).

B. Client side:

- 1) Operating System: Red Hat Fedora Linux (kernel 2.6.14 - 1.1653).

- 2) Open Swan: Used for IPSec VPN secure data exchange (version 2.4.7).
- 3) Proposed Hybrid VPN system: Used for IPSec SSL VPNs secure data exchange (core ensemble mechanism system client side version 2.01).
- 4) Web browser: Mozilla (version 1.7.13).

Hardware:

A. Server side:

- 1) Processor: Intel(R) Pentium(R) 4, 3.2 GHz or later.
- 2) Memory: 1 GB RAM or later.
- 3) Network Interface Card: Broadcom Gigabit configured to 100 Mbps.

B. Client side:

- 1) Processor: Intel(R) Pentium(R) III, 1.7 GHz or later.
- 2) Memory: 128 MB RAM or later.
- 3) Modem: U.S. Robotics PCI 56 Kbps.
- 4) WAN Link: Dial up 56 Kbps.

C. Internal Network Configuration:

- 1) Local Area Network (LAN): 100 Mbps closed switched intranet.
- 2) WAN link: Dial up 128 Kbps.
- 3) Switch: Standard base 10/100 Mbps.

5.3.1.2 Experimental Scenarios

The nature of data exchange between the VPN client personal

computers (PCs) and the VPN servers in web services scenarios can vary.

For the sake of flexibility in parameters and simplicity, the data exchange in the tests was experimented as a file transfer between the VPN client and VPN server.

The size of files transferred at different occasions varied from 1 KB to 1 MB (1024 KB). The data transfer process was initiated and controlled by an external script that was also responsible for clearing the cache at regular intervals.

The two experimental scenarios were considered to evaluate and compare performance are the separate IPsec VPN and SSL VPN scenario and proposed Hybrid IPsec SSL VPN system scenario in a networked environment. These scenarios repeated several time during experiments at overall the benchmark tests for certain VPNs technologies.

First Scenario: Separate IPsec VPN and SSL VPN

Figure (5.1) illustrate the experimental scenario for the separate IPsec VPN and SSL VPN that the IPsec client PC connect to the internet with 56 kbps dial up link and the SSL client PC connect in the same way.

The tunneled communication connect to firewall of internal network with 128 kbps dial up link, then the request switched via standalone IPsec VPN server or via standalone SSL VPN server to

the destination server.

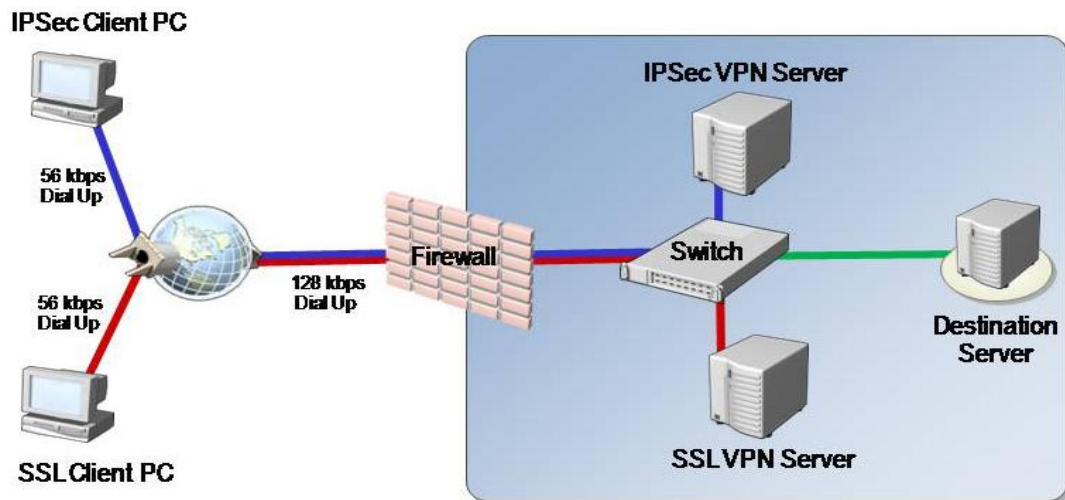


Figure 5.1 - Separate IPsec VPN and SSL VPN Scenario

Both the standalone IPsec VPN and SSL VPN servers handling the secure requests and back the response from the destination server through firewall over internet to the certain VPN clients.

Second Scenario: Proposed Hybrid IPsec SSL VPNs

System

Figure (5.2) illustrate the experimental scenario for the proposed Hybrid IPsec SSL VPNs system scenario that the IPsec client PC connect to the internet with 56 kbps dial up link and the SSL client PC connect in the same way.

The tunneled communication connect to the firewall of the internal network with 128 kbps dial up link, then the request switched

via the proposed Hybrid VPN system server to the destination server.

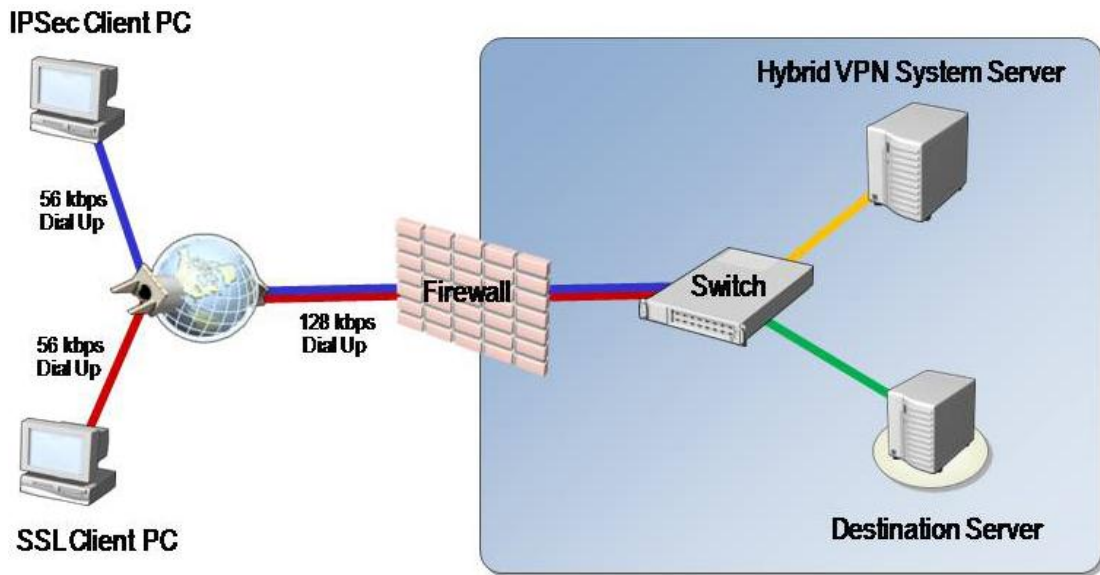


Figure 5.2 - Proposed Hybrid IPsec SSL VPNs System Scenario

The proposed Hybrid VPN system server handles the secure requests and backs the response from the destination server through firewall over internet to the certain VPN clients.

5.3.1.3 Variables Discipline

The following experimental variables were disciplined and controlled to cause the minimum effects on the experiments setup.

- 1) System Load: While performing the experiments, only the required services and processes for VPN systems and network connectivity were run. This prevented any fluctuations in system load due to external services or processes.
- 2) Network Load: All network services were disabled while performing the experiments.

- 3) Caching: An external script was used to clear the buffer and cache after each file transfer.

5.3.2 Throughput Measurement Benchmark

In this benchmark, the test dependent on securely transferring the files in different sizes which created using random binary data generated by random number generator algorithm that found on Red Hat Fedora Kernel 2.6.14 - 1.1653 at /dev/random directory.

The command line was used to write specified number of bytes to the test files. The encrypted files would be transferred between the VPN client PC and the VPN server in different experiments settings.

The file transfer was monitored using Netperf, Iperf and Netio tools for throughput. Multiple runs were performed in two scenarios and data was aggregated to improve accuracy. Table (5.1) below shows the average throughput for different sized data transfer.

Table 5.1 - Average throughput for different sized data transfer

Scenarios	Data File Transfer				
	1 KB	10 KB	100 KB	500 KB	1 MB
First Scenario					
IPSec VPN	8.12	7.76	7.82	7.91	7.65
SSL VPN	8.54	8.12	7.96	8.03	8.01
Second Scenario					
Proposed Hybrid VPN	7.39	7.22	7.26	7.15	7.20

5.3.3 Delay Measurement Benchmark

In this benchmark, the test dependent on the UDP traffic generator at the VPN client PC which configured to generate different streams of encrypted UDP traffic on VPN server.

Table 5.2 - Average delay in both higher and lower UDP traffic

Payload Size (Byte)	First Scenario				Second Scenario	
	IPSec VPN		SSL VPN		Proposed Hybrid VPN	
	Gen. Gap 1 ms	Gen. Gap 5 ms	Gen. Gap 1 ms	Gen. Gap 5 ms	Gen. Gap 1 ms	Gen. Gap 5 ms
16	342.2	62.18	281.4	53.01	393.5	78.60
32	332.8	63.85	292.6	56.15	404.3	79.22
64	340.5	62.96	302.3	59.21	413.1	82.60
128	353.6	70.60	320.5	64.00	429.4	84.11
256	369.6	69.62	339.8	65.19	444.5	87.05
512	401.1	80.20	368.9	72.16	485.3	91.50
1024	474.3	91.15	426.2	80.37	550.6	101.85
2048	541.7	100.18	471.1	85.63	631.8	114.72

The data streams consist of 10000 packets of UDP traffic with a fixed repeating packet content value. The experiment runs for a higher UDP flow with inter packet generation gap of 1 ms and repeated with a relatively lower UDP flow with inter packet generation gap of 5 ms.

The starting UDP payload size is 16 bytes and increased doubling step-by-step up to a maximum of 2048 bytes to reach higher traffic levels.

Each reading is taken at least five times and the mean value of this sample is recorded to increase the accuracy.

Table (5.2) shows the readings of the average full round trip delay for the two scenarios in both higher and lower UDP securely traffic.

5.3.4 Bandwidth Consumption Measurement Benchmark

In this benchmark, the test bandwidth consumption dependent on the operation of a security protocol which can be broadly split into four phases: idle, initialization, negotiation, and data transfer phase.

The idle Phase consists of launching the web portal interface by the URL, writing the username and password, selecting the security services, submitting, checking prerequisites on the VPN client PC such as IP address and available ports.

The initialization phase marks the beginning of data communication as the protocol initiates a communication setup. The negotiation phase consists of generating of keys and other security materials and exchanging parameters between VPN client PC and VPN server.

This phase is bound to be bandwidth intensive as the generation of keys and security parameters require processing on the WAN link. Finally, in the data transfer phase, the data transfer begins.

The VPN client PC and VPN server nodes continuously monitored using Iperf tool to observe the pattern of bandwidth consumption.

Table (5.3) below shows the bandwidth consumption observed at different phases of protocol operations.

Table 5.3 - Bandwidth consumption in different protocol operation phases

Scenarios	Protocol Operations			
	Idle	Initialization	Negotiation	Data Transfer
First Scenario				
IPSec VPN	9.5	22.5	56.2	11.9
SSL VPN	9.3	20.5	44.9	10.9
Second Scenario				
Proposed Hybrid VPN	9.8	26.5	82.7	15.2

5.4 Results Comparison

This section presents the comparison of the benchmark tests results by the charts and discuss these results for the certain measurements such throughput, delay, and bandwidth consumption over IPSec VPN, SSL VPN, and proposed Hybrid VPN system.

5.4.1 Throughput Measurement Test Results

Comparison

Throughput is one of the most important measurements used to evaluate the performance of VPN data networks. Figure (5.3) illustrate the comparison of the throughput among IPSec VPN, SSL VPN, and the proposed Hybrid VPN system.

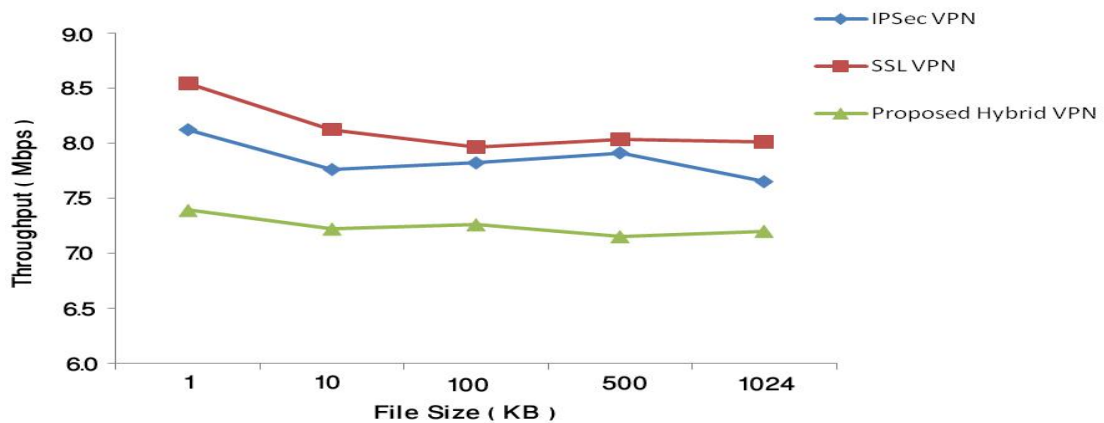


Figure 5.3 - Throughput measurement test results comparison

The throughput results are plotted as chart has been analyzed the comparisons detailed as follows:

- 1) The throughput coefficient in general matches backward with the file size coefficient for all VPNs technologies used. Because of the processing of encryption and decryption operations between the VPN client PC and VPN server decreased whenever the protected payload size increased.
- 2) The SSL VPN has the highest throughput, followed by IPSec VPN and proposed Hybrid VPN respectively. Because the IPSec suite protocols was not working properly behind firewall systems while the SSL has an advantage in this case.
- 3) The throughput of proposed Hybrid VPN has small gapping than IPSec and SSL VPNs, because first, it wrappers a built in IPSec layer which has disadvantage behind firewall working, and secondly the reason of the internal protocol switching operations due to the delay of packet handling.

4) Finally, although the proposed Hybrid VPN throughput was lower than SSL and IPsec VPNs, but it is relatively in an acceptance level.

5.4.2 Delay Measurement Test Results Comparison

The delay measurement describes the various bottlenecks in the VPN as a result of the formed packets queues.

Figure (5.4) illustrates the comparison of the round trip delay with an inter packet generation gap of 1 ms among IPsec VPN, SSL VPN and proposed Hybrid VPN system.

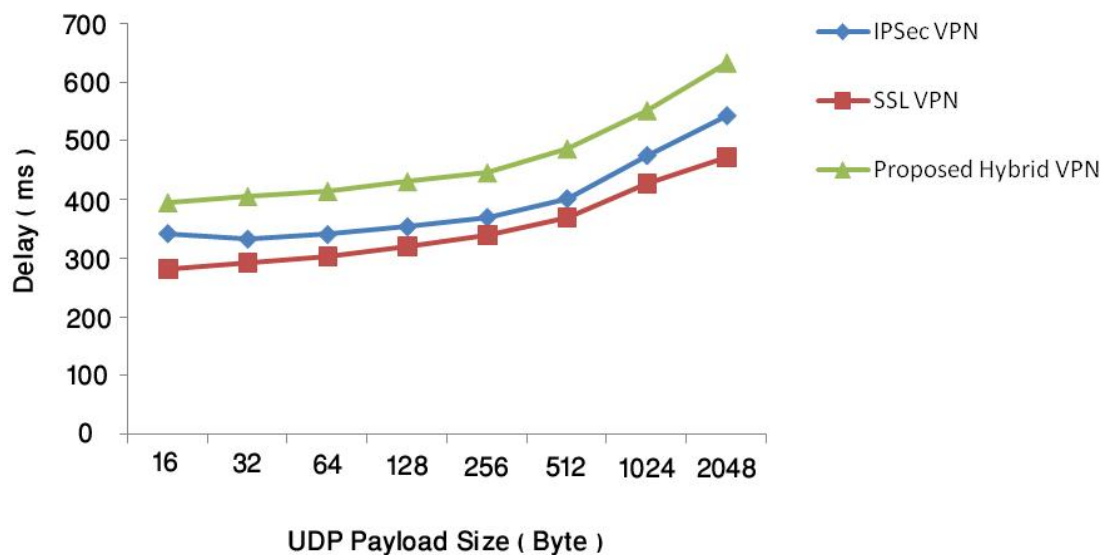


Figure 5.4 - Delay measurement test results comparison in 1 ms time gap

A comparative analysis of the round trip delay in 1 ms gapping is carried out in the following points:

- 1) The delay coefficient in general matches backward with the UDP payload size coefficient for all VPNs technologies used.

- 2) Because firstly, the processing of encryption and decryption operations between the VPN client PC and VPN server, and secondly the reason of relatively high UDP flow with small time gapping between the UDP streams which increased the traffic and consequently increased the round trip delay.
- 3) The SSL VPN has the lowest round trip delay, followed by IPsec VPN and proposed Hybrid VPN respectively. Because the nature of SSL encryption approaches was handling lightly than IPsec encryption approaches.
- 4) The delay of proposed Hybrid VPN was the highest, because of the internal protocol switching operations between the IPsec and SSL layers to handle packets.

Figure (5.5) illustrate the comparison of the round trip delay with inter packet generation gap of 5 ms among IPsec VPN, SSL VPN and proposed Hybrid VPN system.

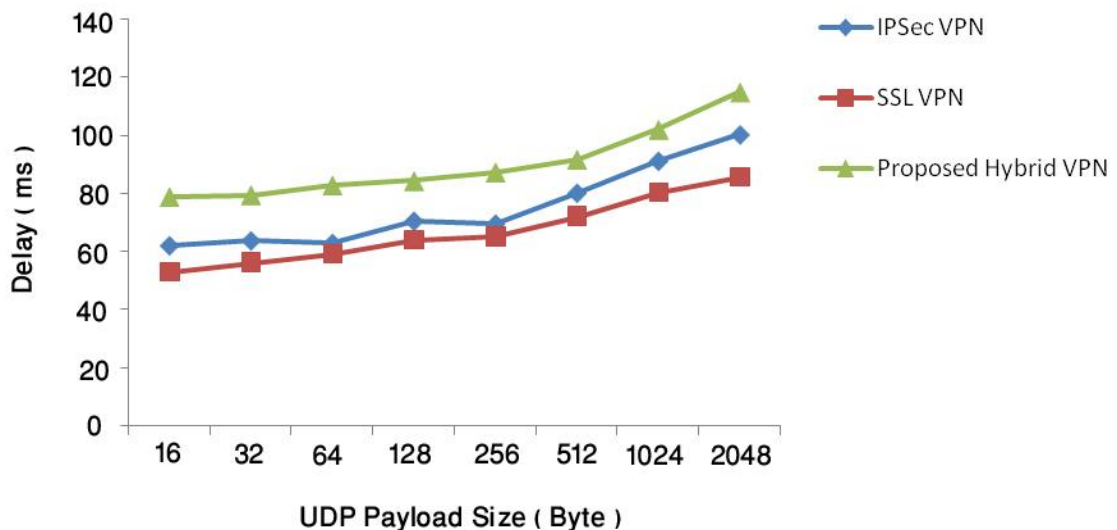


Figure 5.5 - Delay measurement test results comparison in 5 ms
time gap

Such information may also reference the effects of change the inter packet generation time gap which investigated and pointed as follows:

- 1) The delay coefficient in general matches backward with the UDP payload size coefficient for all VPNs technologies used, for the same reason discussed above in Figure (5.4).
- 2) The delay measurement decreased clearly when the inter packet time gap increased from 1 ms to 5 ms for all VPNs technologies used because this gapping time give the VPNs engines the chance to handle their formed packets queues which buffered during the connection operation due to encryption and decryption processes.
- 3) Increasing the inter packet time gap decrease the round trip delay measurement, so to control on the UDP flow should limit the number of the sessions that established between the VPN client PCs and VPN server.
- 4) The approximate speculations delay bottleneck in the proposed Hybrid VPN system can be made on the regions where the internal protocol switching operated.
- 5) Finally, although the proposed Hybrid VPN round trip delay measurement was the highest compared with the SSL and IPSec VPNs in both inter packet time gaps, but it is relatively in acceptance level.

5.4.3 Bandwidth Consumption Measurement Test Results

Comparison

The bandwidth significant affect on the overall network connections specially which is used in a secure communication channels such as VPN, and therefore bandwidth consumption should be controlled to reach a best performance.

Figure (5.6) illustrate the comparison of the bandwidth consumption in different phases of protocol operation during the secure connection established among IPSec VPN, SSL VPN and proposed Hybrid VPN system.

The detailed analysis of the bandwidth consumption test results from the experimentation described in the following points:

- 1) The bandwidth consumption coefficient percentage is the amount of the bandwidth consumption for each protocol operational phase divided by the total of the bandwidth consumption linkage.
- 2) The bandwidth consumption percentage is the highest in the negotiate phase for all VPNs technologies used. Because the negotiation phase consists of generating of keys and other security materials and exchanging parameters between VPN client PC and VPN server. This phase is bound to be bandwidth intensive as the generation of keys and security

3) parameters require processing on the link.

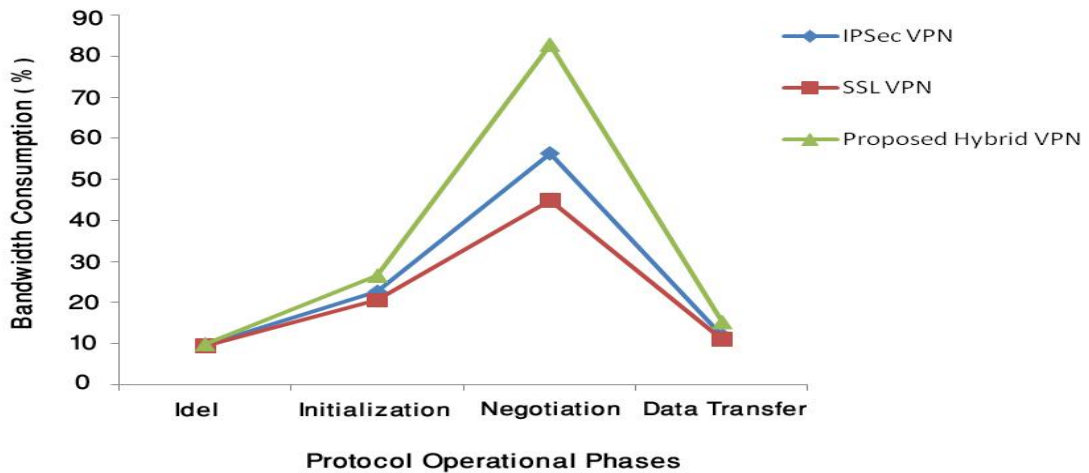


Figure 5.6 - Bandwidth consumption measurement test results comparison

- 4) The SSL VPN has the lowest bandwidth consumption, followed by IPSec VPN and proposed Hybrid VPN respectively. Because the SSL protocol operations was handling lightly than IPSec protocol operations.
- 5) The bandwidth consumption of proposed Hybrid VPN was a highest, because it consists both types of the IPSec and the SSL protocol operations on the same system which communicate by several sessions in different types between the VPN client PCs and the proposed Hybrid VPN server. These protocol operation connections consume to a considerable extent the bandwidth linkage.
- 6) Although the proposed Hybrid VPN bandwidth consumption measurement was the highest comparing with the SSL and the IPSec VPNs, but it is relatively in acceptance level.
- 7) Possibly, to increase the performance over linkage which

- 8) connect the VPN client PCs and VPN server is upgrade the bandwidth of the link between them, but it is financially expensive.

Chapter Six

Conclusions and Future Works

6.1 Introduction

The aim of this research was to develop a solution to attain the best combinations complementary advantages of IPsec VPN and SSL VPN technologies and eliminate their shortcomings.

The proposed solution had been implemented and tested; its performance analysis compared with separate IPsec VPN and SSL VPN technologies through considered performance measurements included throughput, round trip delay and bandwidth consumption.

The conclusions of each part of this research were summarized. The recommendations for future works and suggestions studies in the development and performance analysis fields were also presented.

6.2 Research Conclusions

This section will revisit parts of this research including introduction, backgrounds, related studies and works. Furthermore, conclude the development and performance analysis of the proposed solution.

6.2.1 In General

The VPN technologies have important role especially in the enterprises environments. The Hybrid IPsec SSL VPN technology presented as proposed solution for this research problem statement.

The performance analysis was positively correlated with separate IPsec and SSL VPN technologies. This research strongly contributes in research field and in applied field.

6.2.2 Backgrounds, Related Studies and Works

Several VPN technologies found this research focuses deeply on technical and functionality details on both IPsec VPN and SSL VPN technologies.

A few studies and works available are represented the core of this research, but several studied papers and related trusted works published so far have strong link with this research fields that depended on.

There are studies and related works using ensemble system mechanism to build a powerful network system, statistical benchmarks and experimental environments of performance analysis measurements.

6.2.3 Development of Proposed Hybrid VPN System

The proposed Hybrid IPsec SSL VPNs technology relieved enterprises from the burden of maintaining two separate VPNs infrastructures because it fitted the benefits of both.

This technology served a broad level of the client applications and provided with many server services through web portal interface which reduce the financial costs and administration supports.

In the client phase, the client chose either SSL VPN or IPsec VPN remote access service to establish the secure connections with certain server through the proposed Hybrid VPN system. In the proposed Hybrid VPN system phase, handling protected packets by security layers and routes the requests to the destination servers.

In the server phase, the servers provide different types of services needed to client computer by either IPSec protocol or SSL protocol such as authentication server, B2B partner site server, and email server.

Representation of the sending and receiving packets algorithms and show the four levels of the virtual interaction model of the proposed Hybrid VPN system, Implementation of the switching algorithms and interaction four levels presented in ML and native C programming languages.

6.2.4 Performance Analysis of Proposed Hybrid VPN System

The comprehensive set of the performance measurements relating to VPN technology which considered are throughput, round trip delay and bandwidth consumption.

Several benchmark tests were experimented for the certain performance measurements and observed the revised data which organized in tables. The benchmark tests results compared graphically with each other for every VPN technologies used.

The deductions points of these results discussed in details to explain the truly reasons for the relationship among the certain performance measurements and different VPN technologies used.

6.3 Recommendations for Future Works

This section would be show the suggestions which can be attended for future works. In addition, these recommendations will encourage the researchers to present further studies in this research

fields such development and performance analysis for proposed solution.

6.3.1 Development of Proposed Solution Field

For future works, the proposed solution presented can be improved by carrying out the following:

- 1) Develop the proposed solution with other network security protocols such as network authentication protocol (Kerberos).
- 2) Solve this research problem statement by another protocols switching mechanism such as Horus system mechanism.
- 3) Apply the proposed solution under close source software such as UNIX, UNIX IPSec, and VeriSign SSL Certificate.
- 4) Compare between the proposed solution and the same ready made products in the market such as Cisco ASA 5500 Series SSL/IPSec VPN Edition.

6.3.2. Performance Analysis of Proposed Solution Field

The presented performance analysis in this research can be readily extended to include:

- 1) Research for new algorithms to decrease the internal protocol switching overheads in the proposed solution design to increase the performance.
- 2) Improve the working of IPSec protocols suite behind the network firewalls which enhance the overall performance.
- 3) Analyze the effects of the performance measurements when applied in internet protocol sixth version (IPv6) environment.
- 4) Study the proposed solution performance measurements through wireless networks.

BIBLIOGRAPHY

- [1] Al-Chaal, Lina. "Dynamic and Easily Manageable Approach for Secure IP VPN Environments".PH.D. Thesis, Institute National Poly Technique De Grenoble, 2005.
- [2] Alshamsi, AbdelNasir and Takamichi Saito. "A Technical Comparison of IPSec and SSL". Tokyo University of Technology, 2004.
- [3] Bickford, M, C. Kreitz, R. V. Renesse, and X. Liu. "Proving Hybrid Protocols Correct". Department of Computer Science, Cornell University, Ithaca, NY, 2001.
- [4] Broderick, J.S. "What is VPN?" and "VPN Security Policy". Director of C.S. Department, Symantec Inc., I.S.T. Report Journal, Vol. 6, No. 1 pages (15-22) and pages (31-34) respectively, 2001.
- [5] Cisco Team. "IPSec VPN WAN Design Overview". Data Sheet, Cisco Systems Inc., USA, 2006.
- [6] Derek, Ban. "IPSec vs. SSL: Why Choose?".Head of Research Department, Open Reach, Inc. 2002.
- [7] Djin, Twum. "Managing Access Control in Virtual Private Networks". Thesis, Department of C.S., Dartmouth College, 2005.
- [8] Doraswamy, Naganand and Harkins, Dan. "IPSec Architecture". Articles, the new security standard for the internet, intranets and virtual private networks, Prentice Hall, 2004.
- [9] Finch, Derek. "Deploy IPSec and SSL Together for a Complete Remote Access Solution". White Paper, Fiber link Inc., 2003.
- [10] Fisli, Rezan. "Secure Corporate Communications over VPN-

Based WANs". Department of Numerical Analysis KTH and Computer Science, Stockholm Royal Institute of Technology, Sweden, 2005.

[11] Guttman, Peter. "Performance Characteristics of Application-level Security Protocols". University of Auckland, 2004.

[12] Hayden, Mark. "The Ensemble System". PHD Dissertation, the faculty of the graduate school of Cornell University, 1998.

[13] Hosner, Charlie. "Open VPN and the SSL VPN Revolution". White Paper, SANS Institute, USA, 2004.

[14] IT Team. "IPSec and SSL: Complementary VPN technologies for Universal Remote Access". National Web cast Initiative, 2005.

[15] IT Team. "THE NEXT-GENERATION SSL VPN". Technical Study, Watch Guard Technologies, Inc., 2005.

[16] Jensen, Jens. "Enhancing SSL Performance". CCLRC Rutherford Appleton Laboratory, 2006.

[17] Kak, Avi. "Security for Internet Applications (PGP, IPSec, SSL/TLS)". Master Thesis, Purdue University, 2006.

[18] Kreitz, Christophe and Liu, Xiaoming. "Proving Hybrid Protocols Correct". Department of Computer Science, Cornell University, Ithaca, NY, U.S.A.2001.

[19] Leroy, Xavier. "The Objective Caml system". Documentation and user's manual, Institute National, 2005.

[20] Lin, Cherg, Ching Chang and Tao Chung. "Design, Implementation and Performance Evaluation of IP-VPN". Department of Computer Science and Engineering Tatung University, China, 2003.

- [21] Miltchev, Stefan, Sotiris Ioannidis and Angelos Keromytis. "A Study of the Relative Costs of Network Security Protocols". University of Pennsylvania and Columbia University, 2000.
- [22] Mitchell, J.C., V. Shmatikov and U. Stern. "Finite-State Analysis of SSL 3.0". Stanford University, Stanford, 1998.
- [23] Nicklos, Phillip. "IPSec vs. SSL VPNs for Secure Remote Access". White paper, Winfrasoft Company, 2006.
- [24] Onyszko, Tomasz. "SSL Architecture". Articles, the new security standard for the internet, intranets and VPN, Prentice Hall, 2004.
- [25] Oppliger, Rolf. "Security Technologies for the World Wide Web", Second Edition. Artech House, 2003, ISBN: 1580533485.
- [26] Perlman, Radia. "Analysis of the IPSec Key Exchange Standard". Sun Microsystems Laboratories, Iris Associates, 2001.
- [27] Product Bulletin. "SSL VPN Module 1000". Nortel Networks business without boundaries, 2004.
- [28] Rao, Goutham. "Net6 Hybrid-VPN Gateway". Chief Architect, Net6 Release, Inc.2004.
- [29] Renesse, Robbert, Ken Birman, Mark Hyden, Alexey Vaysburd, and David Karr. "Building Adaptive Systems Using Ensemble". Department of Computer Science, Cornell University, 1999.
- [30] Rissler, Roslyn and Sorensen, Sarah. "VPN Decision Guide". White paper, Juniper Networks, Inc., USA, 2005.
- [31] Rodeh, Ohad, Kenneth Birman, and Danny Dolev. "The Architecture and Performance of Security Protocols in the Ensemble Group Communication System". IEEE Computer Society Press, 2000.

[32] Root, Don and Roslyn Rissler. "IPSec and SSL VPN Decision Criteria". Juniper Networks, Inc. 2006.

[33] Schneier, Bruce and Ferguson, Niels. "A Cryptographic Evaluation of IPSec". Counterpane Internet Security, Inc., San Jose, 1999.

[34] Steinberg, Joseph. "SSL VPN Security". Director of Technical Services, Whale Communications, 2003-14.

تطوير و تحليل أداء تكنولوجيا الشبكات الخاصة الافتراضية لتقنيات البروتوكولات الآمنة IPsec SSL المهجنة

إعداد

مازن غازي جمعة

إشراف

الأستاذ الدكتور هلال البياتي

Arabic Summary

تلعب تكنولوجيا الشبكات الخاصة الافتراضية دوراً مهماً خاصة في بيئات العمل الكبيرة للشركات. و يعتبر الأمن من أهم القضايا لهذه التكنولوجيا، لذا كانت هناك العديد من التقنيات التي تقدم خاصية الأمن مثل تقنيات البروتوكولات الآمنة IPsec و SSL، و لهذه التقنيات الكثير من الفوائد و كذلك العيوب. يُقدّم هذا البحث تطويراً لتكنولوجيا الشبكات الخاصة الافتراضية لتقنيات البروتوكولات الآمنة IPsec SSL المهجنة كحل مقترح لنيل أفضل مجموعة متكاملة من فوائد تقنية بروتوكول IPsec و تقنية بروتوكول SSL و إزالة عيوبهم.

كما يُقدّم البحث تحليلاً مفصلاً لأداء تكنولوجيا الشبكات الخاصة الافتراضية لتقنيات البروتوكولات الآمنة IPsec SSL المهجنة المقترح و مقارنته بشكل منفصل مع تقنيات البروتوكولات الآمنة IPsec SSL كلاً على حده من خلال مقاييس الأداء المُعتَبَرة و التي تَصمّنَت مقياس الإنتاجية و التأخير و إستهلاك سعة الخط.

و لقد اجريت الإختبارات السابقة لقياس مستوى الأداء على تكنولوجيا الحل المهجن المقترح ، و قورنت النتائج مع نتائج مقاييس الأداء لتقنيات البروتوكولات الآمنة IPsec SSL و تبين أن نتائج مقاييس الأداء للحل المهجن المقترح كانت إيجابية و بمستويات مقبولة.

إن تكنولوجيا الحل المهجن المقترح في هذا البحث يخفض عبء الدعم الفني و ضغط إدارة الشبكات و يشمل جهود الصيانة و توزيع البرمجيات ، كذلك تخفض هذه التكنولوجيا التكاليف المالية الباهظة بشكل ملحوظ مما له أثر إيجابي على مستوى السوق.

