ABSTRACT

Zhou, Dong. Simulation of Transaction Capabilities Application Part (TCAP) over IP. (Under the direction of Dr. S. Felix Wu.)

In order to internetwork with Public Switched Telephone Network (PSTN), Internet telephony must process ISDN User Part (ISUP) and Transaction Capabilities Application Part (TCAP) messages. To allow the interoperability between the two networks, it is necessary for the performance of the signaling over IP to be comparable to that the current PSTN users can get to avoid introducing degradation in the service. At the same time, the signaling over IP needs to deal with the attack that comes from the IP domain.

In this research, several activities have been conducted. They are as follows. (1) Performance requirements for TCAP over IP are derived from the current PSTN standards. (2) A simulation of TCAP over IP based on STIPP is designed and implemented. (3) An experiment for getting the processing time of IPSEC is implemented. (4) Several experiments are implemented on the simulation testbed for getting the performance of TCAP over IP under flood attack.

According to the results, we find the delay of TCAP over IP is bigger than that we get in PSTN network. However, it is still in the range of the delay requirement of PSTN standards. We also find it is necessary to apply the Quality of Service (QoS) and security mechanism to IP network in order to provide a more reliable and safe network for signaling transmission.

THE SIMULATION OF TCAP OVER IP

by

Dong Zhou

A dissertation submitted to the Graduate Faculty of

North Carolina State University

in partial fulfillment of the

requirements for the Degree of Master of Science

Raleigh

2000

APPROVED BY:

_____	_____
Dr. Douglas S. Reeves	Dr. Wenke Lee

_____

Dr. S. Felix Wu

Chair of Advisory Committee

Contents

ACRONYM LISTS **9**

CHAPTER 1 INTRODUCTION **10**

1.1 WHY TCAP OVER IP? 10

1.2 WHAT DO WE CONCERN? 10

1.3 WHAT DOES THIS THESIS DO? 11

1.4 WHAT ARE THE THESIS CONTRIBUTIONS AND HOW IS IT ORGANIZED? 11

CHAPTER 2 BACKGROUND **13**

2.1 SS7 AND TCAP 13

2.2 SS7 PROTOCOL STACK 13

2.2.1 Message Transfer Part 14

2.2.2 ISDN User Part (ISUP) 14

2.2.3 Telephone User Part (TUP) 14

2.2.4 Signaling Connection Control Part (SCCP) 15

2.3 MORE ABOUT THE TRANSACTION CAPABILITIES APPLICATION PART (TCAP) 16

2.3.1 Transaction Portion 18

2.3.2 Component Portion 18

2.4 THE SIMPLE SS7-TCAP/IP PROTOCOL (STIPP) 19

2.4.1 TCAP-IP Gateway Function 20

2.4.2 IP Entity Function 23

2.5 IPSEC 23

2.5.1 Advantages of IPSEC 24

2.5.2 Limitations of IPSEC 24

2.5.3 Service of IPSEC 24

2.5.4 IPSEC modes 27

2.5.5 Some uses of IPSEC 27

2.6 NETWORK SIMULATION AND NETWORK SIMULATOR (NS2) 28

2.7 FREESWAN BACKGROUND 29

CHAPTER 3 MOTIVATION AND PROBLEMS **32**

3.1 THE DELAY REQUIREMENT FOR TCAP OVER IP 32

3.2 THE DELAY OF THE TCAP OVER IP ON TRADITIONAL IP NETWORK 32

3.3 THE DELAY OF THE TCAP OVER IP ON TRADITIONAL IP NETWORK UNDER FLOOD ATTACK 32

3.4 THE DELAY OF THE TCAP OVER IP IN DIFFSERV ENVIRONMENT 33
3.5 THE DELAY OF THE TCAP IN DIFFSERV ENVIRONMENT OVER IP UNDER FLOOD ATTACK..... 33
3.6 THE DELAY OF THE TCAP OVER IP IN IPSEC ENVIRONMENT 33
3.7 THE DELAY OF THE TCAP OVER IP IN IPSEC ENVIRONMENT UNDER FLOOD ATTACK..... 34
3.8 THE DELAY OF THE TCAP OVER IP IN DIFFSERV + IPSEC ENVIRONMENT 34
3.9 THE DELAY OF THE TCAP OVER IP IN DIFFSERV + IPSEC ENVIRONMENT UNDER FLOOD ATTACK 34

CHAPTER 4 THE PARAMETERS AND DATA COLLECTION FOR THE SIMULATION 35

4.1 THE DELAY REQUIREMENT FOR TCAP 35
4.2 THE PROCESSING DELAY OF IPSEC..... 37
4.2.1 Test scenario 39
4.2.2 Flood attack through the IPSEC tunnel..... 41

CHAPTER 5 TCAP OVER IP SIMULATION 48

5.1 THE OBJECTIVE OF THE SIMULATION..... 48
5.2 THE ARCHITECTURE OF THE SIMULATION..... 48
5.2.1 The node and link in the simulation..... 49
5.2.2 The agents in the simulation..... 50
5.3 THE PROTOCOL IN THE SIMULATION..... 56
5.3.1 STIPP header..... 56
5.3.2 Scenarios..... 60

CHAPTER 6 TCAP OVER IP EXPERIMENTS AND RESULTS 64

6.1 TCAP OVER IP ON TRADITIONAL IP NETWORK 64
6.1.1 Query response delay simulation..... 64
6.1.2 Flood attack simulation..... 70
6.2 TCAP OVER IP IN DIFFSERV ENVIRONMENT 73
6.2.1 Query response delay simulation..... 73
6.2.2 Flood attack simulation..... 79
6.3 TCAP OVER IP IN IPSEC ENVIRONMENT 81
6.3.1 Query response delay simulation..... 82
6.3.2 Flood attack simulation..... 85
6.4 TCAP OVER IP IN DIFFSERV + IPSEC ENVIRONMENT..... 91
6.4.1 Query response delay simulation..... 91
6.4.2 Flood attack simulation..... 94

CHAPTER 7 CONCLUSIONS AND FUTURE WORK.....98

CHAPTER 8 REFERENCES 102

List of Figures

Figure 2.1: The OSI Reference Model and the SS7 Protocol Stack ____ *13*

Figure 2.2: TCAP in SS7 stack ____ *17*

Figure 2.3: STIPP architecture ____ *19*

Figure 2.4: TCAP messaging from SS7 network to IP network ____ *21*

Figure 2.5: TCAP messaging from IP network to SS7 network ____ *21*

Figure 2.6: TCAP messaging from TIPG to IP Entity ____ *23*

Figure 4.1: IPSEC tunnel between source and destination ____ *38*

Figure 4.2: The result of the delay simulation in IPSEC environment ____ *41*

Figure 4.3: Flood attack through IPSEC tunnel ____ *42*

Figure 4.4: The partial result of the delay simulation under one node flood attack through the IPSEC tunnel ____ *43*

Figure 4.5: The three nodes flood attack through IPSEC tunnel ____ *44*

Figure 4.6: Three nodes flood attack out of band of the IPSEC tunnel ____ *45*

Figure 5.1. Simulation topology ____ *48*

Figure 5.2: The NS2 node architecture ____ *49*

Figure 5.3: The simulation of queuing ____ *52*

Figure 6.1: Signaling internetworking between SS7 and IP network ____ *65*

Figure 6.2: Simulation of STIPP ____ *66*

Figure 6.3: The topology of the delay simulation of STIPP ____ *67*

Figure 6.4: The procedure of the delay simulation of STIPP ____ *68*

Figure 6.5: The results of the delay simulation of STIPP ____ *70*

Figure 6.6: An attack node floods Query requests to the destination node ____ *71*

Figure 6.7: The topology of the simulation of flood attack to TIPG ____ *71*

Figure 6.8: The results of the simulation of flood attack to TIPG ____ *73*

Figure 6.9: Testbed for Diffserv ____ *74*

Figure 6.10: Diffserv applied on the testbed ____ *75*

Figure 6.11: Topology for Diffserv testbed. ____ *76*

Figure 6.12: Partial results of the experiments on Diffserv testbed. ____ *78*

Figure 6.13 Flood attack to TCAP over IP on Diffserv ____ *79*

Figure 6.14. The delay of TCAP over IP under flood attack in Diffserv environment ____ *81*

Figure 6.15: Setup an IPSEC tunnel between two TIPGs. All TCAP messages are transferred through the tunnel between the two TIPGs. ____ *82*

Figure 6.16: The topology for TCAP over IP on IPSEC ____ *83*

Figure 6.17: Delay of TCAP query over IP on IPSEC ____ *84*

Figure 6.18: An attack node floods TCAP query messages to the destination node through IPSEC tunnel ____ *85*

Figure 6.19: The topology of TCAP delay simulation under one node flood attack through IPSEC tunnel ____ *86*

Figure 6.20: The partial results of TCAP delay simulation under one node flood attack through IPSEC tunnel ____ *88*

Figure 6.21: IPSEC applies authentication between attached node and TIPG ____ *88*

Figure 6.22: The results of TCAP delay simulation under flood attack. The TIPG applies policy to drop the flood attack packets. ____ *91*

Figure 6.23: The TCAP over IP in Diffserv + IPSEC environment ____ *91*

Figure 6.24. The delay of TCAP over IP on Diffserv + IPSEC ____ *94*

Figure 6.25. The flood attack to TCAP over IP in Diffserv + IPSEC environment ____ *94*

Lists of Tables

Table 4.1: Maximum Number of Signaling Points and STPs in a National Component (Source: ITU-T Recommendation Q.709, Table 3) _____ *35*

Table 4.2: Switch Response Time Assuming Typical Traffic Mix and Message Lengths (Source: Telcordia GR-1364-CORE, Table 5-1) _____ *36*

*Table 4.3: ITU-T Cross-Switch Transfer Time (Source: ITU-T Recommendation Q.706, Table 5)*_____ *36*

Table 4.4: Maximum End to End TCAP Signal Transfer Delays for National Component _____ *37*

Table 4.5: The Round-trip Delay in No-IPSEC and IPSEC _____ *40*

Table 4.6: The Round-trip Delay in No-IPSEC and IPSEC under one node flood attack _____ *42*

*Table 4.7: The Round-trip Delay in No-IPSEC and IPSEC under three nodes flood attack*_____ *45*

Table 4.8: The Round-trip Delay in No-IPSEC and IPSEC under flood attack out-band of the IPSEC tunnel _____ *46*

*Table 6.1: Network Element Processing Time in Simulation*_____ *67*

Table 6.2: The result of the experiments on Diffserv testbed _____ *77*

Table 7.1: The performance of TCAP over IP in different environment _____ *98*

Acronym Lists

Diffserv	Differentiated service
IN	Intelligent Network
INAP	Intelligent Network Application Part
ISUP	ISDN User Part
GTT	Global Title Translation
MAP	Mobile Application Part
OSI	Open Systems Interconnect
PCS	Personal Communications Services
PSTN	Public Switched Telephone Network
RSVP	ReSerVation Protocol
SCP	Service Control Point
SCCP	Signaling Connection Control Part
SSN	SubSystem Number
SS7	Signaling System #7
STIPP	Simple TCAP-IP Protocol
TCAP	Transaction Capabilities Application Part
TIPG	TCAP/IP Gateway
TUP	Telephone User Part
VoIP	Voice over IP

Chapter 1 Introduction

1.1 Why TCAP over IP?

Today, most countries' PSTN networks use SS7 as the signaling network. TCAP is currently used extensively in SS7 networks to exchange user sensitive signaling information between application and plays an important role in Intelligent Network (IN). Some examples of those applications can be Intelligent Network Application Part (INAP) in IN systems, Mobile Application Part (MAP) in GSM or IS-41 in AMPS systems. The application on top of TCAP takes advantage of the international support of SS7 systems in many countries. As IP Telephony becomes widely used in the telecommunication industry, more and more IP entities are involved in telephone systems. Those technologies, such as Internet Dial-up Access, Voice over IP (VoIP), have identified the need for SS7/IP internetworking.

PSTN call processing involves both the ISUP and TCAP. The ISUP is responsible for the basic setup, management and teardown of a telephone. In the current market, there are several products implementing the SS7-ISUP inter-work with IP. TCAP is mainly used for transaction related applications such as in 800/888, calling card and local number portability service. IP Telephony users also require those services. That makes TCAP over IP a critical requirement for VoIP. However, some of the service entities have been planned or implemented in the IP network without the standardization of the TCAP/SS7 interworking with IP.

1.2 What do we concern?

The performance of the TCAP over IP including delay, jitter and loss rate certainly affects the success of the call processing procedure, then further decides the quality of service of the IP telephony. Traditional Internet provides a best-effort service to all upper applications and is not appropriate to deliver QoS. However, the network community has worked hard to develop new service models like RSVP and Diffserv to make Internet provide some degree of QoS.

What kind performance we can get if we implement TCAP over IP based on those new service models becomes an interesting topic.

On the other hand, the SS7 system is almost isolated from the outside world. Hackers hardly attack a PSTN system since they rarely could access an entity in that network. The explosive growth of the Internet and IP networks makes the integration of SS7 entities in the IP network more practical. This integration also provides a bridge for hackers to jump into the SS7 world. We can imagine that a flooding of TCAP query packets from IP domain denies a SCP service in SS7 world. Therefore, the security risk and solution in implementing the TCAP over IP is another concern.

1.3 What does this thesis do?

This thesis analyzes the delay requirement and security issues for TCAP signaling over IP protocol and does a simulation of a simple TCAP over IP protocol. And this thesis uses network simulation technology to build up a TCAP over IP testbed and does some experiments on it. Those experiments are related to the performance of TCAP on different service models and under flood attack.

1.4 What are the thesis contributions and how is it organized?

This thesis:

1. Analyzes the delay requirement for TCAP over IP.

2. Does an experiment for getting the IPSEC processing time.

3. Implements a simulation of a simple TCAP over IP protocol and does the query response time experiments on normal environment, IPSEC environment and Differentiated Service environment.

4. Analyzes of the performance of TCAP over IP under flood attack on different environments.

The thesis is organized as follows. Chapter 2 gives a brief background of the technologies that the thesis involves. Chapter 3 describes the motivation of the thesis and lists the problems we researched in this thesis. Chapter 4 does the parameter and data collection for the simulation. This chapter analyzes the current PSTN standards and derives a query response delay requirement for TCAP over IP. It also describes the experiment for getting the processing time of IPSEC. The processing time of implementing the IPSEC is a very important parameter in the following experiments. Chapter 5 describes the TCAP over IP simulation. Chapter 6 describes the experiments that run on the simulation testbed and gives the analysis of the results. Chapter 7 summarizes our results, gives suggestions for the future work and concludes the thesis.

Chapter 2 Background

This chapter gives a brief background for the techniques we will meet in the following chapter.

2.1 SS7 and TCAP

2.2 SS7 protocol stack

The hardware and software functions of the SS7 protocol are divided into several levels (Fig 2.1). These levels map loosely to the Open Systems Interconnect (OSI) 7-layer model.

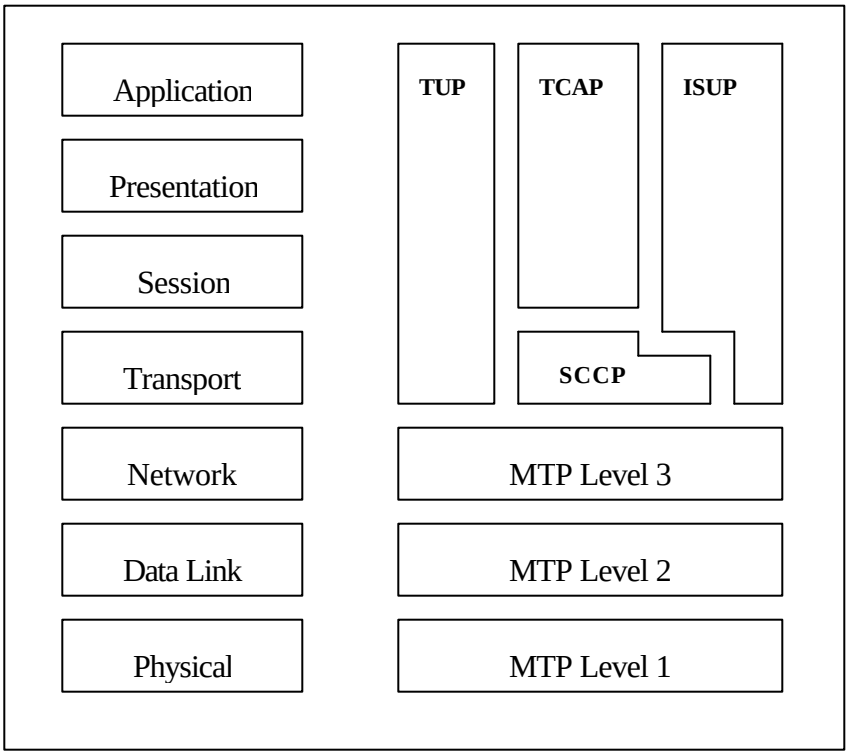


Figure 2.1: The OSI Reference Model and the SS7 Protocol Stack

2.2.1 Message Transfer Part

There are three sub layers in the Message Transfer Part (MTP). The lowest level, MTP Level 1, is equivalent to Physical Layer in OSI model. MTP Level 1 defines the physical, electrical, and functional characteristics of the digital signaling link. The physical interfaces defined include E-1 (2048 kb/s; 32 64 kb/s channels), DS-1 (1544 kb/s; 24 64kb/s channels), V.35 (64 kb/s), DS-0 (64 kb/s), and DS-0A (56 kb/s).

MTP Level 2 is equivalent to the OSI Data Link Layer. It is responsible for accurate end-to-end transmission of a message across a signaling link. It implements flow control, message sequence validation, and error checking. When an error occurs on a signaling link, the message is retransmitted.

MTP Level 3 is equivalent to the OSI Network Layer. MTP Level 3 provides message routing between signaling points in the SS7 network. Like the function in OSI network layer, MTP Level 3 reroutes traffic away from failed links and signaling points and does the congestion control.

2.2.2 ISDN User Part (ISUP)

The ISDN User Part (ISUP) is the protocol that handles the trunk circuits set-up, management, and release. ISUP is used for both ISDN and non-ISDN calls.

2.2.3 Telephone User Part (TUP)

Some countries (e.g., China, Brazil) use the Telephone User Part (TUP) to support basic call setup and teardown. TUP handles analog circuits only. In many countries, ISUP has replaced TUP for call management.

2.2.4 Signaling Connection Control Part (SCCP)

SCCP is above MTP Level 3 to provide connectionless and connection-oriented network services. While MTP Level 3 uses point codes as addresses to transfer messages to specific signaling points, SCCP provides subsystem numbers to allow messages to be addressed to specific applications (called subsystems) at these signaling points. In some TCAP-based services such as 800/888, calling card and local number portability, SCCP is used as the transport layer such as in the wireless roaming, and personal communications services (PCS). SCCP also provides the means by which an STP can perform global title translation (GTT), a procedure by which the destination signaling point and subsystem number (SSN) is determined from digits (i.e., the global title) present in the signaling message. The global title digits may be any sequence of digits (e.g., the dialed 800/888 number, calling card number, or mobile subscriber identification number) pertinent to the service requested. Because an STP provides global title translation, originating signaling points do not need to know the destination point code or subsystem number of the associated service. Only the STPs need to maintain a database of destination point codes and subsystem numbers associated with specific services and possible destinations.

2.2.4.1 Transaction Capabilities Applications Part (TCAP)

In the SS7 network, TCAP uses the SCCP connectionless service. Queries and responses sent between SSPs and SCPs are carried in TCAP messages. For example, an SSP sends a TCAP query to determine the routing number associated with a dialed 800/888 number or to check the personal identification number (PIN) of a calling card user. In mobile networks (IS-41 and GSM), TCAP carries Mobile Application Part (MAP) messages sent between mobile switches and databases to support user authentication, equipment identification, and roaming. More details of TCAP are described in the following section.

2.2.4.2 Operations, Maintenance and Administration Part (OMAP) and ASE

OMAP and ASE are areas for future definition. Presently, OMAP services may be used to verify network routing databases and to diagnose link problems.

2.3 More about the Transaction Capabilities Application Part (TCAP)

The Transaction Capabilities Application Part (TCAP) is designed for non-circuit-related messages. Database entities as well as actual end office switches can be the destination of those TCAP messages. The TCAP protocol is critical for reliable transferring of information from one application at a switch location to another application within another network entity.

The first usage of TCAP protocol was 800 number translation. When a user calls an 800 number, the call cannot be routed through the telephone network because the area code "800" does not specify any particular exchange. In order to overcome this problem, a database for 800 numbers provides a routing number that the local office can use to route the call through the PSTN network. In most cases, the database is centrally located within the service provider's network. In today's networks, all 800s must be visible for all carriers. That means all telephone companies need the capability to access other companies' 800 databases. This is known as transportable 800 numbers. The TCAP protocol provides a message type and parameters that the central office switch can use to query the database for the specific information.

Another feature can be the automatic callback service. When a subscriber dials a number that is busy, the subscriber can enter a feature code and hung up. When the dialed number is available, the local exchange notifies the caller's local switch by sending a TCAP message. This TCAP message allows the local switch to ring the telephone of the caller. The distant switch has reserved the called party's line so no other telephone calls can be routed to it. When the calling party answers the telephone, normal call setup procedure is used to establish the connection between the two exchanges. TCAP, in this case, serves as an alerting mechanism,

sending an informational message (not circuit related) to another entity within the network. This can be extended to many other types of applications where remote invocation is an option.

TCAP query also can be used to check the personal identification number (PIN) of a calling card user. In mobile networks (IS-41 and GSM), TCAP carries Mobile Application Part (MAP) messages sent between mobile switches and databases to support user authentication, equipment identification, and roaming.

In short, TCAP provides a way for end users in the SS7 network to access other end users on a peer-to-peer level. The end users can be database entities, local central offices and Service Control Points (SCP). The access can be database query or operation invocation.

In SS7 protocol stack, the TCAP is above the SCCP. The TCAP is using the services of the Signaling Connection Control Part protocol to invoke an operation and return the results of the operation. SCCP provides the network services, while the Message Transfer Part provides the physical layer and data link layer functionality (Fig 2.2).

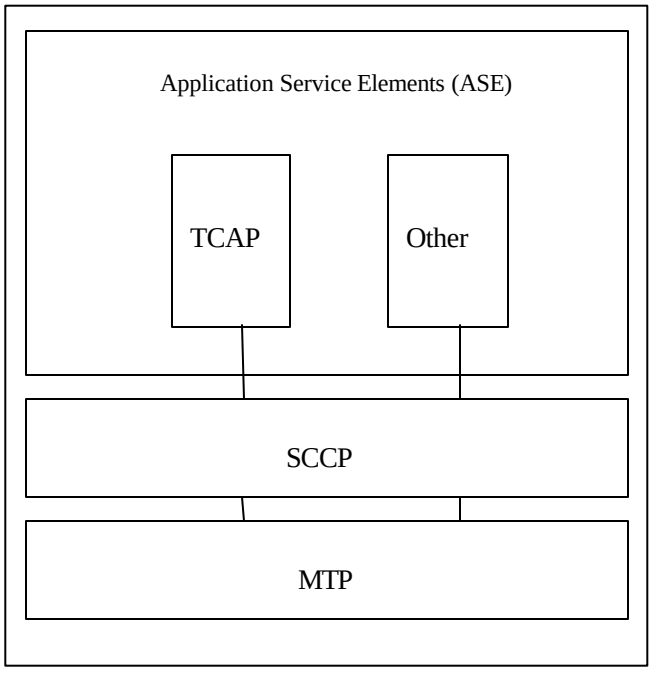


Figure 2.2: TCAP in SS7 stack

TCAP messages are contained within the SCCP portion of an MSU. A TCAP message is comprised of a transaction portion and a component portion.

2.3.1 Transaction Portion

The transaction portion contains the package type identifier. There are seven package types:

Unidirectional: Transfers component(s) in one direction only (no reply expected).

Query with Permission: Initiates a TCAP transaction (e.g., a 1-800 query). The destination node may end the transaction.

Query without Permission: Initiates a TCAP transaction. The destination node may not end the transaction.

Response: Ends the TCAP transaction. A response to an 1-800 query with permission may contain the routing number(s) associated with the 800 number.

Conversation with Permission: Continues a TCAP transaction. The destination node may end the transaction.

Conversation without Permission: Continues a TCAP transaction. The destination node may not end the transaction.

Abort: Terminates a transaction due to an abnormal situation.

The transaction portion also contains the Originating Transaction ID and Responding Transaction ID fields which associate the TCAP transaction with a specific application at the originating and destination signaling points, respectively.

2.3.2 Component Portion

The component portion contains components. There are six kinds of components:

Invoke (Last): Invokes an operation. For example, a Query with Permission transaction may include an Invoke (Last) component to request SCP translation of a dialed 800 number. The component is the "last" component in the query.

Invoke (Not last): Similar to the Invoke (Last) component except that the component is followed by one or more components.

Return Result (Last): Returns the result of an invoked operation. The component is the "last" component in the response.

Return Result (Not last): Similar to the Return Result (Last) component except that the component is followed by one or more components.

Return Error: Reports the unsuccessful completion of an invoked operation.

Reject: Indicates that an incorrect package type or component was received.

2.4 The Simple SS7-TCAP/IP Protocol (STIPP)

This section addresses a Simple TCAP-IP Protocol (STIPP) for the interworking of SS7-TCAP and IP that was originally proposed by Nortelnetworks [1]. Figure 2.3 shows the architecture of the protocol.

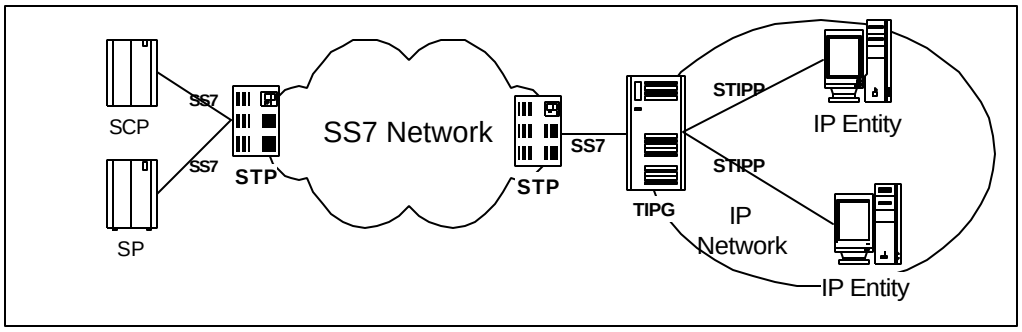


Figure 2.3: STIPP architecture

The idea behind the protocol is a convergence layer- Simple TCAP/IP Interworking Part (STIPP) being added between TCAP and IP layer. A TCAP/IP Gateway (TIPG) is introduced into the system to provide an interface between the SS7 network and IP network. The TIPG has two basic hardware interfaces. The SS7 interface supports the SS7 protocols including SCCP and TCAP to connect to the SS7 network. The IP interface supports IP protocols to connect to IP network. IP entities can be application servers with TCAP capability, such as a call server. All TCAP messages that are transferred in IP domain are encapsulated in STIPP packets. When an IP entity needs to send TCAP messages to the SS7 side, it will send the TCAP messages in using STIPP packet to TIPG. TIPG forwards the TCAP messages using MTP packets. When an entity on the SS7 side, say a SCP, sends a TCAP message to an IP entity, it sends the TCAP message to TIPG in a MTP packet. TIPG extracts the TCAP messages, encapsulates it into STIPP packets and forwards them to the IP Entity. The TCAP-IP gateway and IP Entity functions are described as below.

2.4.1 TCAP-IP Gateway Function

2.4.1.1 SCCP Flow Control

In SS7 network, TCAP only uses the connectionless service provided by SCCP. There are four SCCP messages that are used in the SCCP connectionless services: Unitdata (UDT), Extended Unitdata (XUDT), Unitdata Service (UDTS), and Extended Unitdata Service (XUDTS). The UDT message is normally used to transmit TCAP messages. The XUDT message is used if the TCAP message is too big to fit into the SCCP message. When the TIPG receives an UDT or XUDT message but the destination IP Entity for this message is not properly functional, TIPG must send back the UDTS of XUDTS to the message originator to indicated if the IP Entity is congested or failed.

2.4.1.2 Encapsulation/Decapsulation

The TIPG must provide the encapsulation/decapsulation functions at both directions from the SS7 network to the IP network and vice versa.

I. SS7 network originates a TCAP message into the IP network:

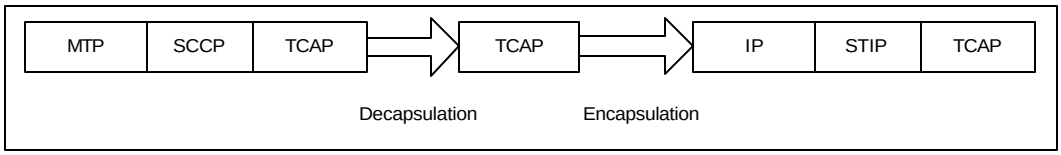


Figure 2.4: TCAP messaging from SS7 network to IP network

The SS7 message is decapsulated to remove the MTP and SCCP headers. The remaining TCAP message is encapsulated with the IP header and the STIPP header, which contains the required information from the MTP and SCCP headers. The STIPP header will be defined in protocol part.

II. IP network originates a TCAP message into the SS7 network:

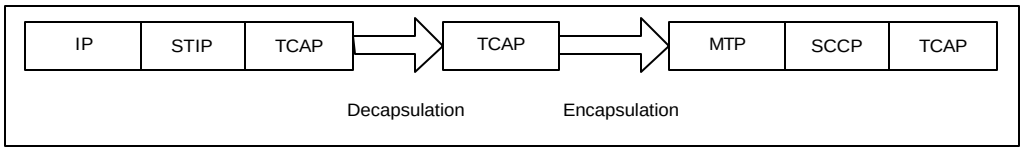


Figure 2.5: TCAP messaging from IP network to SS7 network

The IP message is decapsulated to remove the STIPP header and the IP header. The remaining TCAP message is encapsulated with the MTP and SCCP. The STIPP header will be defined in protocol part.

III. IP network transports a TCAP message between two SS7 networks:

The TCAP message originates from one SS7 network and is terminated to a far-end SS7 network. The IP network only provides signaling transport functions.

2.4.1.3 Address Translation

The TIPG must provide the address translation between the Point Code (PC) /Subsystem Number (SSN) and the IP address. On the SS7 side, the SS7 network addresses each IP entity by a PC and a SSN. The point code is identical for some of IP entities, but the SSN must be different to correctly address each of the IP entities. On the IP side, each IP entity as well as the TIPG is pre-assigned with a unique IP address. Each IP entity should be pre-configured with the IP address of the TIPG. The TIPG should maintain the mapping between the PC and/or SSN and the IP address of the IP entity.

2.4.1.4 Global Title Translation (GTT)

Global Title (GT) Address in SS7 network refers to E.164 or E.214 Mobile numbers. The GTT function performed by SCCP is to translate GT to find a destination node in the SS7 network. Since TIPG is acting as a transit signaling point to IP entities, the GTT function may be required and extended in TIPG. Possible GTT outcomes:

- A destination node in the SS7 network - DPC (+SSN or +GT)

- A destination node in the IP network - IP address + port number

2.4.1.5 Protocol Discrimination

In theory, the TIPG must provide the protocol discrimination function for all messages received from the SS7 network. In this simulation I just implement part of those messages. A TCAP message received from the SS7 network can be defined by several standards. It can be defined by the TCAP standard such as Charging, Provide Instruction, Connection Control, Network

Management, etc. On the other hand, it can be defined by the upper-layer protocols such as IS-41, MAP, OMAP, etc.

The messages received from the SS7 network are the TCAP messages with the Invoke Component. These messages can be discriminated by the Operational Code Identifier and the Operational Family of the Operation Code. For example, the IS-41 messages have the Operational Code Identifier of Private TCAP and the Operational Family of 9. Other TCAP messages (with Return Result, Return Error, and Reject Component) are sent into the SS7 network and do not require the protocol discrimination. For these messages, the TIPG maps the source IP address of the IP packet to the corresponding SS7 PC/SSN and sends those messages to the SS7 network.

2.4.2 IP Entity Function

The IP entity has one basic interface that supports the IP network. For the function aspect, the IP entity is responsible for processing the TCAP message. The IP Entity communicates the TIPG via STIPP protocol. If IP Entity receives a TCAP message that is not in its supported protocols, it should send back a TCAP Data Error message. The IP entity must provide the encapsulation/decapsulation functions at both directions from the TIPG to the IP entity and vice versa.

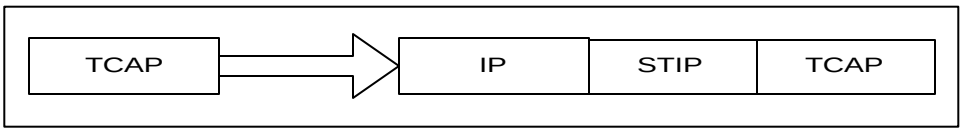


Figure 2.6: TCAP messaging from TIPG to IP Entity

2.5 IPSEC

In brief, IPSEC provides encryption and authentication services at the IP level of the protocol stack.

2.5.1 Advantages of IPSEC

Today IPSEC is the most general way to provide encryption and authentication services for the Internet. The big difference between other security protocols and IPSEC is IPSEC working on IP layer while others works on higher layers or lower layers. Higher-level services protect a single application protocol; for example, PGP protects mail. Lower level services protect a single physical medium; for example, a pair of encryption boxes on the ends of a line makes wiretaps on that line useless unless the attacker is capable of breaking the encryption. IPSEC, however, can protect protocols running above IP and mediums used below IP. Further more, it can protect a mixture of protocols running over a complex combination of media.

2.5.2 Limitations of IPSEC

IPSEC is not end-to-end solution. IPSEC cannot provide the same end-to-end security as some security system working at higher levels. For example, if you need encrypt email from the sender's desktop to the recipient's desktop and only the recipients can decrypt the email, use PGP or another such system.

At best, IPSEC encrypts information as it leaves the sender's machine and decrypts it on arrival at the recipient's machine. Suggestion is that if you need end-to-end encryption, you should use software specifically designed for this end-to-end service. In another common setup, IPSEC encrypts packets at a security gateway and decrypts them on arrival at the gateway to the recipient's site. However, it is not a strict end-to-end service. Anyone with appropriate privileges on either site's LAN can intercept the message in unencrypted form.

2.5.3 Service of IPSEC

IPSEC offers two services, authentication and encryption. These can be used separately but are often used together.

Authentication

Authentication allows a packet to be confidently sent from a particular. No attempt is made to conceal or protect the contents, only to assure its integrity.

Authentication can be provided separately using an Authentication Header (AH) or it can be included as part of the Encapsulated Security Payload service (ESP).

Encryption

Encryption allows protecting the contents of a packet from eavesdroppers.

IPSEC does the encryption using a secret key. In most setups, keys are automatically negotiated, and periodically re-negotiated. A protocol, know as the IKE (Internet Key Exchange), handles the key management.

The Authentication Header (AH)

Authentication service can be provided by adding an authentication header (AH) after the IP header but before the other headers on the packet [16].

Normally, headers on a packet are connected by a linked list where each header contains a "next protocol" field telling the system what header to look for next. IP headers generally have either TCP or UDP in this field. While IPSEC authentication is used, the packet IP header has AH in the field and the authentication header has the next header type -- usually TCP, UDP or encapsulated IP.

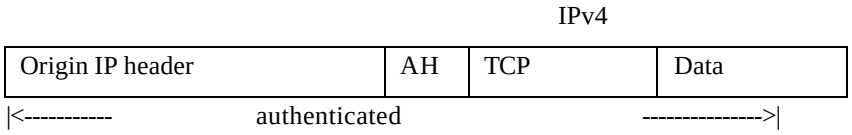
IPSEC authentication can be added in transport mode, as a modification of standard IP transport. This is shown in this diagram from the RFC2402[16]:

Before applying AH

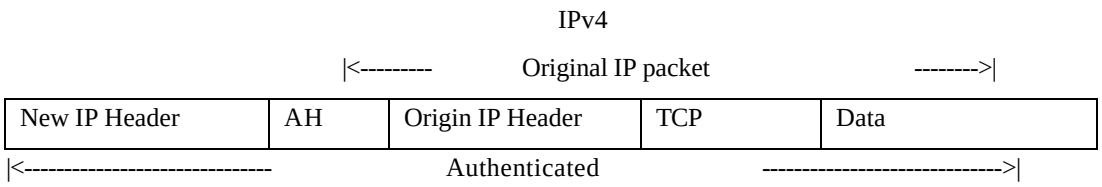
IPv4

Origin IP header	TCP	Data

After applying AH



Authentication can also be used in tunnel mode. The tunnel encapsulates the underlying IP packet beneath AH and an additional IP header.



This would normally be used in a gateway-to-gateway tunnel. The receiving gateway strips the outer IP header and the AH header and forwards the inner IP packet.

Keyed MD5 and Keyed SHA

The actual authentication data in the header is typically 96 bits long and depends both on a secret key shared between sender and receiver and on the contents of the data being authenticated.

The algorithms involved are the MD5 Message Digest algorithm [17] or the Secure Hash Algorithm [18]. These algorithms are designed to make it nearly impossible for anyone to alter the authenticated data in transmission. The sender calculates the hash value from the data in the packet and includes the result in the authentication header. The recipient does the same calculation and compares results. If the data are unchanged, the results will be identical. The

hash algorithms are used to make it extremely difficult to change the data in any way and still get the correct hash.

Sequence numbers

The authentication header includes a sequence number field. The sender increases the sequence number for each packet. The receiver may use it to check that packets are indeed arriving in the expected sequence.

Using sequence numbers provides partial protection against replay attack. It cannot provide complete protection since it is necessary to handle legitimate packets that are lost, duplicated, or delivered out of order, but use of sequence numbers makes the attack much more difficult.

Encapsulated Security Payload (ESP)

The ESP protocol is defined in RFC 2406 [19]. It can provide encryption and authentication. So, it may be used with or without AH authentication.

2.5.4 IPSEC modes

Tunnel mode

Tunnel mode connections requires security gateway support. The gateway provides a secure tunnel for all client machines behind the gateway. The client machines need not do any IPSEC processing. They only need to route packets to gateway.

Transport mode

As opposed to tunnel mode, transport mode asks the host machines to do its own IPSEC processing and routes some packets via IPSEC.

2.5.5 Some uses of IPSEC

ESP for encryption and authentication

AH for authentication alone

Other variants are allowed by the standard, but not much used:

ESP encryption without authentication

ESP encryption with AH authentication

Authenticate twice, with AH and with ESP

ESP authentication without encryption

Multiple layers of IPSEC processing are possible

The above describes combinations possible on a single IPSEC connection. A complex network may have several layers of IPSEC in play, with any of the above combinations at each layer.

2.6 Network Simulation and Network Simulator (NS2)

This thesis uses a network simulator to test the performance of TCAP over IP. Using a network simulator has some advantages. Normally, testbeds are expensive to build. Testbeds and labs can be difficult to reconfigure and share, and they have limited flexibility. On the other hand, multi-protocol network simulators can provide a rich environment for experimentation at low cost. In addition, users can very easily add some simulation component may be easily constructed by those who know most about the particular protocol represented by the component.

Network simulation has a very long history. NS2 (Network Simulator 2) itself is derived from REAL [5], which is derived from NEST [6]. NS2 is an object-oriented simulator, written in C++ and Otcl. The simulator supports a class hierarchy in C++(also called the complied hierarchy) and a class hierarchy in Otcl (also called the interpreted hierarchy). The two hierarchies are closely related to each other. From the user's perspective, there is a one-to–one correspondence between a class in the interpreted hierarchy and one in he compiled hierarchy.

TclObject is the base class for most of the other classes in the interpreted and compiled hierarchies. The user creates every object in the class TclObject from within the interpreter. An equivalent shadow object is created in the compiled hierarchy. The class TclClass contains the mechanism that performs this shadowing. In most cases, access to compiled member variables is restricted to compiled code, and access to interpreted member variables is likewise confined to access via interpreted code. However, NS2 can establish bi-directional bindings such that both the interpreted member variable and the complied member variable can access the same data and change the value of either variable to the same value when changing the value of the corresponding paired variable. The compiled constructor establishes the binding when that object is instantiated; it is automatically accessible by the interpreted object as an instance variable. In this way, users can not only program in Otcl to define the elements and behavior of the network, but also dynamically get the network status through those binding variables. NS2 also provides functions to trace data on a simulation. Generally, trace data is either displayed directly during execution of the simulation, or (more commonly) stored in a file to be post-processed and analyzed.

2.7 FreeSwan Background

Linux FreeSwan is an implementation of IPSEC & IKE for Linux. The objective is to help make IPSEC widespread by providing freely available source code. FreeSwan is an initiative of RSA Data Security and a number of other companies, mainly vendors of firewalls and other security products. The focus is on testing interoperability and sorting out details so that the players' implementations of IPSEC will work with each other.

IPSEC is designed to support VPNs, which allows multiple sites from an organization to communicate securely over an insecure Internet by encrypting all communication between the sites.

The authentication algorithms in FreeSwan are the same ones used in the IPSEC authentication header. FreeSwan implements both of the required encryption algorithms but also triple DES or 3DES. The triple DES is very strongly recommended since DES is insecure.

FreeSwan parts

KLIPS: KLIPS is the kernel of the IPSEC Support. Linux kernel is necessarily modified to for IPSEC support.

The Pluto daemon: Pluto is a daemon that implements the IKE protocol. It verifies identities, chooses security policies, and negotiates keys for the KLIPS layer.

The ipsec (8) command: The ipsec (8) command is a front end that allows control over IPSEC activity.

Linux FreeS/WAN configuration file: The configuration file for Linux FreeSwan is /etc/ipsec.conf

Key management: IPSEC can manage keys in several ways. Not all are implemented in Linux FreeSwan. Currently Implemented Methods include manual keying and automatic keying.

Manual keying

IPSEC allows keys to be manually set. In Linux FreeSwan, such keys are stored with the connection definitions in /etc/ipsec.conf. Manual keying is useful for debugging since it allows developers to test the KLIPS kernel IPSEC code without the Pluto daemon doing the key negotiation.

In general, however, automatic keying is preferred because it is more secure.

Automatic keying

In automatic keying, the Pluto daemon negotiates keys using the IKE Internet Key Exchange protocol. Connections are automatically re-keyed periodically.

Chapter 3 Motivation and problems

This chapter lists the problems of concern in the simulation. The analysis and experiments related to those problems are in the following chapters.

3.1 The delay requirement for TCAP over IP

The delay requirement for TCAP over IP is our first concern. Two reason that require the TCAP over IP to provide a comparable delay. The first reason is the user has a delay requirement for TCAP. TCAP is used for fetching routing information for 800/888 service. If the delay of TCAP over IP is too long, the call set up procedure may fail due to call set up procedure timeout. The second reason can be the internetworking of the two networks. One TCAP conversation can involve entities in two networks. When an entity sends out a TCAP message, it starts a timer. The period of timeout in both sides should be comparable. Chapter 4.1 analyzes the current SS7 standards and derives a delay requirement for TCAP in SS7 world. We expect our result of the delay of TCAP over IP experiments can be comparable with that value.

3.2 The delay of the TCAP over IP on traditional IP network

In the simplest case, we assume the IP network bandwidths are ample and the number of hops along the TCAP path is comparable to that in SS7. How much is the delay of one normal TCAP query? Does it match the requirement we get in chapter 4.1? Chapter 6.1.1 describes an experiment we did based on the testbed that is build in chapter 5.

3.3 The delay of the TCAP over IP on traditional IP network under flood attack

The flood attack is a common attack in the IP domain that cam deny the service of the application server on the Internet. Flood attack is when the attack nodes sending a great

quantity requests to one application server. The server's incoming buffer is full of the useless packets from the attack nodes. The queuing delay of the normal request will increase, and most normal requests fail due to time out. How much does the flood attack affect the performance of the TCAP over IP? Chapter 6.1.2 describes the experiments of the performance of TCAP over IP under flood attack.

3.4 The delay of the TCAP over IP in Diffserv environment

In traditional Internet, we cannot expect that the network resources are ample due to the traditional best-effort architecture. Currently, we need to choose some service models for getting some degree QoS. Those service models can be Diffserv and RSVP. However, when those service models promise a bounded of delay for TCAP over IP, they also introduce additional delays for tagging and scheduling. So, how much is the tradeoff? Diffserv is simpler than RSVP. We expect Diffserv to bring less delay. Chapter 6.2.1 describes the experiment on an enhanced testbed about the performance of TCAP over IP in Diffserv environment.

3.5 The delay of the TCAP in Diffserv environment over IP under flood attack

The Diffserv can provide a QoS mechanism for TCAP signaling transmission on traditional IP network. In Diffserv, packets will be marked by different TOSs according to different profiles. For example, TCAP message can be marked as EF since it has a bounded delay requirement. We assume the conditioner and scheduler are safe and they cannot be compromised. However, how much flood attack affects the performance of the TCAP over IP? Chapter 6.2.2 describes the experiment about the performance of TCAP over IP under flood attack in Diffserv environment.

3.6 The delay of the TCAP over IP in IPSEC environment

Flood attack can increase the delay of TCAP over IP and further deny the service of TCAP. Currently there are several security protocols working on IP domain to detect the malicious attack and protect the system. However, if we apply some security protocols on TCAP over

IP, how much is the tradeoff? This thesis simulates the TCAP over IP in IPSEC environment. Chapter 4.2 describes the experiments for getting the processing delay of IPSEC on a real machine. Chapter 6.3.1 presents the experiments we did on the simulation testbed for the delay of TCAP over IP in IPSEC environment.

3.7 The delay of the TCAP over IP in IPSEC environment under flood attack

On normal Internet, a flood attack can easily deny service of TCAP gateway or some application servers that involve in the TCAP over IP service. What happens if we build the TCAP over IP on an IPEC tunnel? Does the IPSEC protect the TCAP over IP from flood attack? Chapter 6.3.2 describes the experiments about the performance of TCAP over IP under flood attack in IPSEC environment.

3.8 The delay of the TCAP over IP in Diffserv + IPSEC environment

In the real world, in order to satisfy the delay requirement of TCAP over IP, we expect that the TCAP over IP will be built on an enhanced IP layer that can provide a QoS and security function. This thesis simulates an enhanced IP network based on Diffserv and IPSEC. Chapter 6.4.1 describes the experiment we did on that simulation testbed about the delay of TCAP over IP.

3.9 The delay of the TCAP over IP in Diffserv + IPSEC environment under flood attack

We did the experiments of the performance of TCAP over IP under flood attack on our simulation testbed that implies both Diffserv and IPSEC. Chapter 6.4.2 describes the experiments.

Chapter 4 The parameters and data collection for the simulation

4.1 The delay requirement for TCAP

To gain user acceptation of VoIP services and to enable interoperability between Switched Circuit Networks (SCN) and VoIP systems, it is imperative that the VoIP signaling performance be comparable to that of the current SCNs. The call setup delay, also know as the Post Dial Delay (PDD), in an ISDN-SS7 environment is the period that starts when an ISDN user dials the last digit of the called number and ends when the user receives the last bit of the Alerting message. Although the PDD is not explicitly given in the existing SCN performance requirements, it is still a very important parameter in the research of the requirement for TCAP over IP. A VoIP subscriber can dial an 800 number and hope to get a response from the service provider within a holding time as he dials a common telephone number. There is at least one TCAP query transaction in this call setup procedure. We definitely do not want the delay of the TCAP over IP to be the bottleneck. We can derive the TCAP requirements using ITU-T's SS7 Hypothetical signaling Reference Connection (HSRC) [2], cross-STP (Signaling Transfer Point) time [3], Telcordia's switch response time generic requirements [4], and a simple ISDN-SS7 call flow. The maximum number of signaling points and STPs allowed in a national component and an international component are listed in Table4.1.

Table 4.1: Maximum Number of Signaling Points and STPs in a National Component (Source: ITU-T Recommendation Q.709, Table 3)

Country size	Percentage of connections	Number of STPs	Number of signaling points
Large-size	50%	3	3
	95%	4	4
Average-size	50%	2	2

	95%	3	3

The switch response time is given in Table 4.2. The switch response time is the period that starts when a stimulus occurs at the switch and ends when the switch completes its response to the stimulus. The occurrence of a stimulus often means the switch receives the last bit of a message from an incoming signaling link. The completion of a response means the switch transmits the last bit of the message on the outgoing signaling link. Telcordia GR-1364 specifies switch response time using switch call segments as a convenient way to refer to the various phase of call processing that switches are involved in. Listed in Table 3.2 is the TCAP message call segments that involve the switch sending a TCAP message as result of a stimulus.

Table 4.2: Switch Response Time Assuming Typical Traffic Mix and Message Lengths (Source: Telcordia GR-1364-CORE, Table 5-1)

Type of Call Segment	Switch Response Time (ms)	
	Mean	95%
TCAP Message	210-222	<=342-354

Message delay through a STP is specified as the cross-STP delay. It is the interval that begins when the STP receives the last bit of a message from the incoming signaling link, and ends when the STP transmits the last bit of the message on the outgoing signaling link.

Table 4.3: ITU-T Cross-Switch Transfer Time (Source: ITU-T Recommendation Q.706, Table 5)

STP signaling traffic load	Cross-Switch Transfer Time (ms)	
	Mean	95%
Normal	20	40
+15%	40	80

Using the Maximum Number of Signaling Points and STPs in a National Component, switch response time, and cross-STP delays, we can compute the maximum signaling transfer delays for TCAP message under normal load. The delay is end to end. The results are listed in Table 4.4. As with Telcordia GR-1364, it is assumed that the distribution of switch response time for call segment is approximately a normal distribution. It is further assumed that switch response time of different switches is independent.

Table 4.4: Maximum End to End TCAP Signal Transfer Delays for National Component

Country size	Percent of connections	Delay (ms)	
		Mean	95%
Large-size	50%	690-726	1086-1122
	95%	920-968	1448-1496
Average-size	50%	460-484	724-748
	95%	690-726	1086-1122

The requirements derived in Table 4.4 should be interpreted as the worst-case requirements. At least in the United States, users of SCNs typically experience far less delays then the derived delay requirements. However, the TCAP delays that we get in Table 4.4 are still in the range of the ITUT standards for PSTN. However, from the subscriber experience, we do hope the TCAP over IP to achieve the same level of delay in the Table 3.4. We will use the 690-762ms in the following chapter as the delay requirement for TCAP over IP, since most calls fall in this range in PSTN.

4.2 The processing delay of IPSEC

This chapter describes the experiments for getting the processing delay of implementing IPSEC. We did the experiment because the processing delay of implementing the IPSEC is used as input for the following experiments on the TCAP over IP simulation. The idea is that the

simulation uses IPSEC to set up a secure and efficient tunnel between two entities. All TCAP messages are transferred through IPSEC tunnels. IPSEC provides the encapsulation and authentication functions to protect the communication. In order to find how much delay the IPSEC introduces into the whole system, we need the processing delay of implementing the IPSEC algorithm.

In order to get this important parameter, We use FreeS/WAN to set up an IPSEC testbed (Fig 4.1) and do some experiments for the delay of processing the IPSEC encryption and decryption.

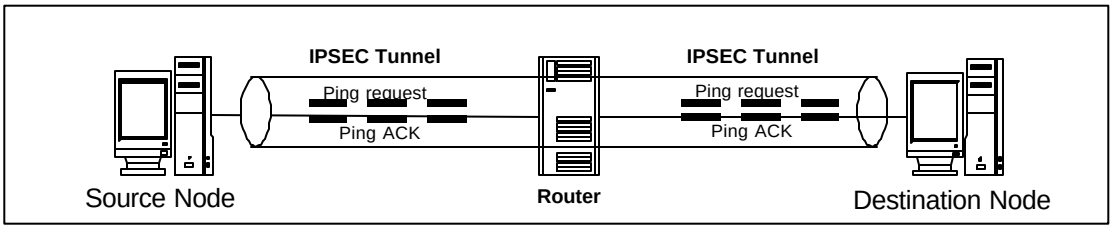


Figure 4.1: IPSEC tunnel between source and destination

In the experiment, both the source and destination nodes were run on Linux and FreeSwan. Both nodes are running on the IPSEC tunnel mode. That means all network traffic between the two nodes goes through an IPSEC tunnel. There is a one router between the two nodes. For simplicity, we use the manual keying at both sides. The processing time of IPSEC depends on the performances of host hardware and operating system. In this experiment, the hardware of the source node and destination node are identical. The network is a LAN, and there is no other traffic in the LAN.

Test platform:

CPU	Pentium III 450 MHz
Memory	128Mb SDRAM
Hard	12961MB

Cache Ram	512 KB
Operating system	Red Hat Linux 6.0
IPSEC implementation	FreeS/WAN

4.2.1 Test scenario

This experiment uses IPSEC ESP to encrypt the traffic between source and destination nodes. Normally, the processing delay for authentication is far less that the processing delay for encryption and decryption. For simplicity, this experiment uses ESP without authentication. Ping is used to generate ICMP traffic between source and destination nodes. We assume the ping processing delay and IPSEC processing delay independently. We run the "Ping" in the No-IPSEC environment and get an average round-trip delay. Then, we run the "Ping" in the IPSEC environment and get another average round-trip delay.

The following equalization is assumed.

No-IPSEC round-trip delay = source node ping processing delay + network delay + destination node ping processing time.

IPSEC round-trip delay = source node IPSEC encryption delay + source node ping processing delay + network delay + destination node IPSEC decryption delay + destination node ping processing delay.

We assume the network delay includes the transmission delay, queuing delay, propagation delay and router processing delay for round-trip. In addition, in the test, the network element in No-IPSEC environment is the same as it is in IPSEC environment. Based on the above assumption, we can derive the IPSEC delay as follows:

IPSEC encryption delay + IPSEC decryption delay = IPSEC round-trip delay – NO-IPSEC round –trip delay

Because the ESP header will apply on the original IP packet in the IPSEC environment, in order to reduce the effect of the ESP header, we let ping packet size be 992 bytes. The ping program adds 8 bytes ICMP header in ping packets. So, there is a total of 1000 bytes IMAP load.

Table 4.5 lists the round-trip-delay for both No-IPSEC and IPSEC environment. The source node IPSEC encryption plus the destination node IPSEC decryption delay is around 0.8ms.

Table 4.5: The Round-trip Delay in No-IPSEC and IPSEC

Round-trip Delay(ms) (min/avg./max/loss rate)	The number of Ping Packets		
	Ping 1,000 times	Ping 10,000 times	Ping 1,000,000 times
No-IPSEC	4.7/4.7/4.8 ms/0%	4.7/5.0/6.0 ms/0%	4.7/5.0/9.9 ms /0%
IPSEC	5.8/5.9/49.6 ms/0%	5.8/5.8/6.8 ms/0%	5.8/5.8/6.5 ms 0%

Figure 4.2 shows partial records of the delay. In the picture, we can see without implementing the IPSEC, the round-trip-delay is about 4.7ms. The average of the round-trip-delay within implementing IPSEC is in the range of 5.8ms to 5.9ms. For clarity, Figure 4.2 only shows the result of Ping 1000 times. In the table 5, we can get the delay of implementing IPSEC encryption and decryption around 0.8ms. Compared to the 222ms TCAP processing time, the overhead of IPSEC is cheap and affordable.

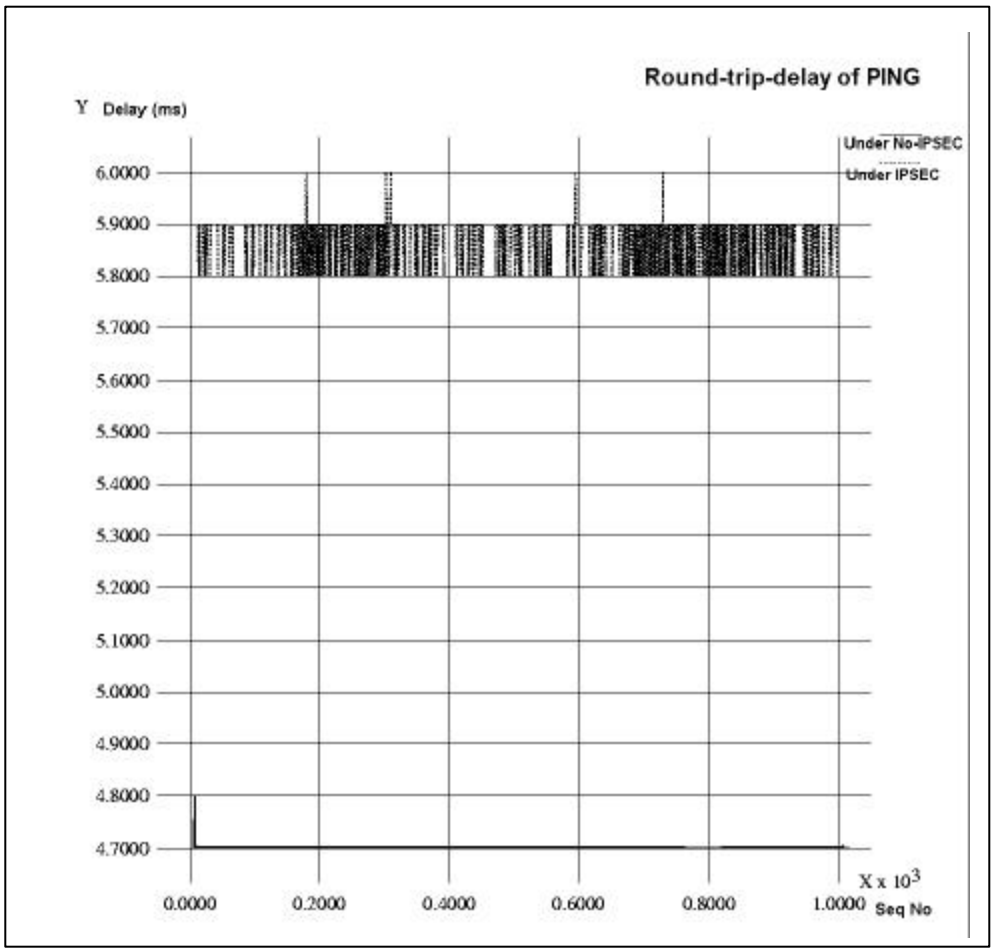


Figure 4.2: The result of the delay simulation in IPSEC environment

4.2.2 Flood attack through the IPSEC tunnel

In order to see the processing time of IPSEC under flood attack, we introduce an attack node into the testbed (Fig 4.3). The attack node floods Ping requests through the source node to the destination node. Since the source node uses IPSEC tunnel to transmit Ping packets to destination node, all flood requests will be encrypted at the source node and sent to the destination node through PSEC tunnel.

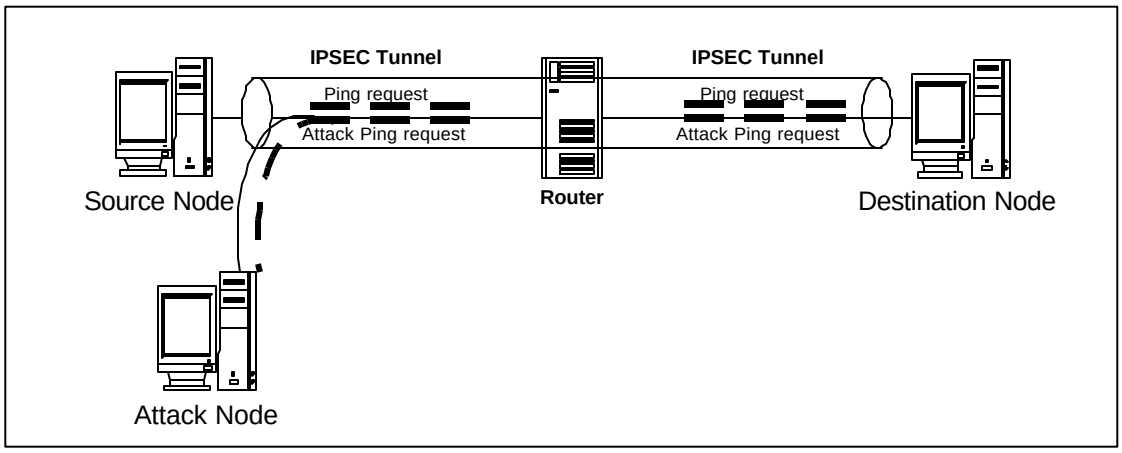


Figure 4.3: Flood attack through IPSEC tunnel

Test platform:

Source and destination nodes are the same machines as in the previous test. The attack node floods in 992Bytes requests at the interval of 100ms to the destination node.

Test results:

Table 4.6 lists the round-trip-delay under flood attack. From the results, we can see the flood attack through the IPSEC tunnel seriously increases the round-trip delay. Figure 4.4 shows partial records of the round-trip-delay.

Table 4.6: The Round-trip Delay in No-IPSEC and IPSEC under one node flood attack

Round-trip Delay(ms) (min/avg./max/loss rate)	Ping 1,000 times	Ping 10,000 times	Ping 1,000,000 times
No-IPSEC	5.5/100.1/198.3 ms/0%	95.2/181.5/305.8 ms/57%	94.3/181.2/307.5 ms /57%
IPSEC	6.2/180.5/207.3 ms/0%	6.0/189.3/297.4 ms/72%	6.0/190.9/304.6 ms /71%

From the diagram, we can see the delay keeps increasing with time. If we subtract the round-trip-delay in IPSEC environment by the delay in the No-IPSEC environment, we can see the difference between the two delays is not increasing so sharply as the delay itself. The encryption and decryption are CPU dominating work. The flood does not increase the complexity of the encryption and decryption algorithm. We can assume the flood attack does not increase the processing time for the CPU to encrypt and decrypt the packets. Based on those assumptions, we can derive that the increase of the round-trip-delay is mainly from the increase of the queuing delay. What happens, if we increase the number of flooding requests?

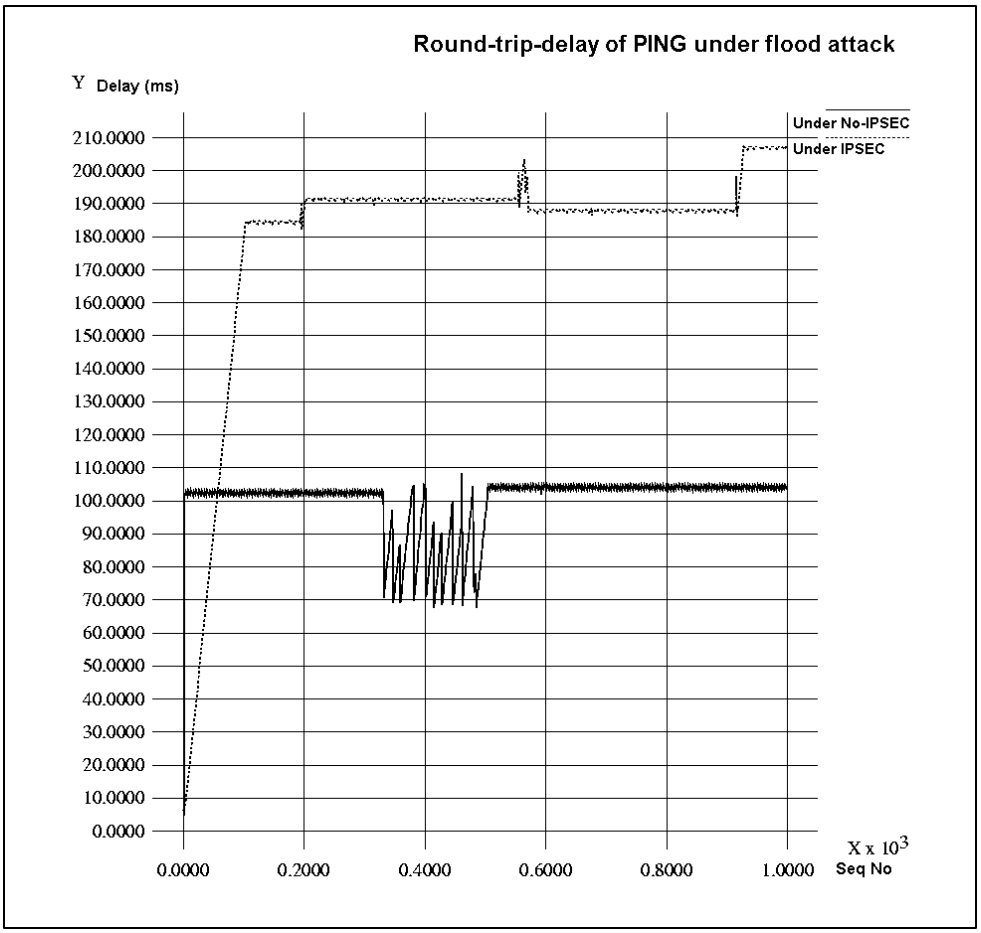


Figure 4.4: The partial result of the delay simulation under one node flood attack through the IPSEC tunnel

We introduce other two attack nodes into the topology (Fig 4.5). The additional two attack nodes connect to the source node in the same way as the first attack node. The source node functions like an IPSEC gateway. The entire flood requests are going to the destination node through the IPSEC tunnel, and are encrypted at the source node. The two additional attack nodes increase the source node CPU burden and increase the source node's receiving buffers overflow.

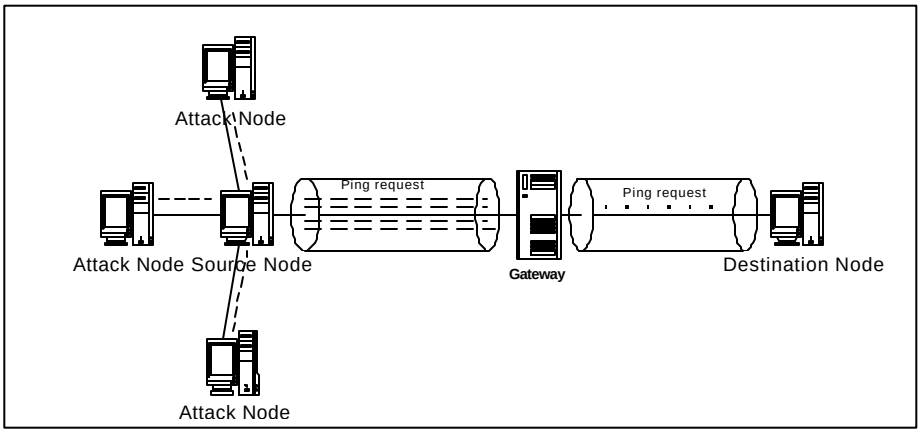


Figure 4.5: The three nodes flood attack through IPSEC tunnel

Table 4.7 lists the round-trip-delay. From the result, we can see the delay becoming larger when the flood becomes more intensive. The difference between the delay for IPSEC and No-IPSEC is increasing. That makes sense, since at the time the CUP is processing the encryption and decryption more packets arrive at the receiving buffer at the source node. In this case, the queuing delay increases sharply, and more packets are dropped at the incoming packets buffer. In these two experiments, all attack traffic passes through the IPSEC tunnel. What happens, if the attack could not pass the gateway authentication and behaves like a man-in-middle?

Table 4.7: The Round-trip Delay in No-IPSEC and IPSEC under three nodes flood attack

Round-trip delay(ms) (min/avg./max/loss rate)	Ping 1000 times	Ping 10,000 times	Ping 1,000,000 times
No-IPSEC	179.4/179.5/180.7 ms/84%	179.0/180.5/348.1 ms/78%	179.4/180.2/512.8 ms /79%
IPSEC	175.7/234.2/407.9 ms/95%	48.0/242.4/518.2 ms/94%	6.2/323.2/723.0 ms 74%

We connect three attack nodes to the router that is in the middle way of the IPSEC tunnel (Fig 4.6). The source and destination nodes use the same machines as in the previous experiment. The three attack nodes flood 992 Bytes Ping packets at the interval of 100ms, respectively. The attack traffic does not go through the IPSEC tunnel, but it goes to the destination through the gateway.

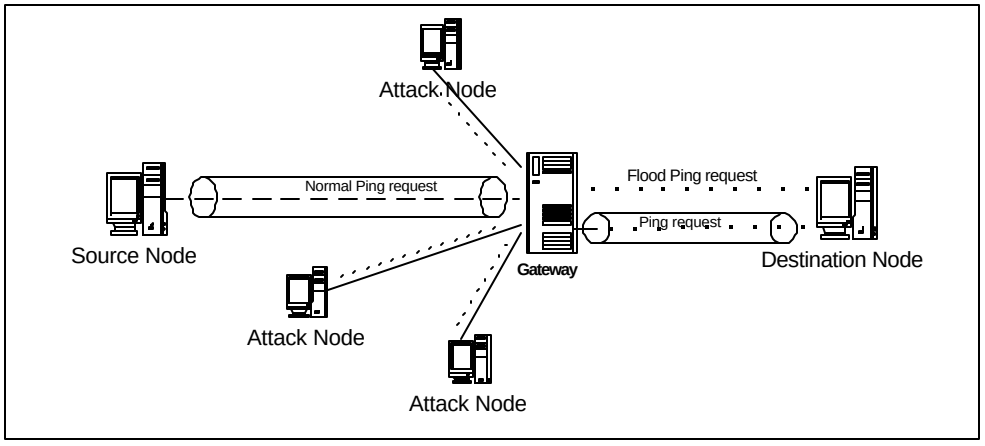


Figure 4.6: Three nodes flood attack out of band of the IPSEC tunnel

Table 4.8 lists the round-trip-delay under the flood attack out of band of the IPSEC tunnel. From the result, we can see the flood attack doesn't increase the round-trip-delay seriously. There are two reasons. The first one is the flood attack is not intensive enough to make the

gateway overburden. The flood attack may increase the average queuing delay at the gateway a little bit, but not so seriously. The other reason that is more important is the FreeS/WAN on the destination node uses different buffers for the packets from the IPSEC tunnel and other packets. The flood requests reach the destination node, but they wait in different buffers from the ICMP requests that are from the source node. Flood traffic does not affect the queuing delay of the packets through the IPSEC tunnel. That means if we configure the system properly, the IPSEC tunnel can isolate the effect from the flood attack. That requires the IPSCE tunnel use authentication to drop the unauthenticated packet to enter the tunnel. Another aspect is that we need to provide some mechanism to classify the normal request and the flood request. There are several network protocols like RSVP and Diffserv that can provide that kind of quality of service. However, only applying those resource reservation protocols is not enough. Flood attack can steal the reserved resource by identifying itself as the normal packet or high priority packet. IPSEC or other security mechanisms must be implied to secure the reserved resource. Of course, the IPSEC tunnel does not provide the complete solution for signaling transmission. If the flood comes from behind the IPSEC gateway, the flood attack still can degrade the service.

Table 4.8: The Round-trip Delay in No-IPSEC and IPSEC under flood attack out-band of the IPSEC tunnel

Round-trip Delay(ms) (min/avg./max/loss rate)	Ping 1000 times	Ping 10000 times	Ping 1,000,000 times
No-IPSEC	5.2/5.5/6.1 ms/0%	5.2/5.9/6.3 ms/0%	5.2/5.8/13.3 ms /0%
IPSEC	5.9/6.1/6.7 ms/0%	5.9/5.9/6.6 ms /0%	5.9/5.9/18.8 ms /0%

From the processing delay of IPSEC experiment, we get the processing time of implementing the IPSEC ESP algorithm on our testbed is around 0.8ms. It is comparably cheap and affordable for TCAP over IP. We also can see the flood attack can seriously increase the

queuing delay and degrade the service, especially when the flood traffic can enter the IPSEC tunnel. However, if we can identify the normal request, we can apply some mechanism to isolate the affect of the flood. The authentication is critical for the signaling transmission.

Chapter 5 TCAP over IP simulation

5.1 The objective of the simulation

The simulation implements the operations and functions to provide the TCAP-IP interworking. It describes the operations and functions of the TCAP-IP Gateway (TIPG) and the protocols between the TIPG and an IP entity. The TCAP-IP gateway provides the TCAP-IP interworking functions between applications in PSTN/IN/Wireless and IP domain. The basic goal is to provide a virtual environment using network simulation technology for testing the protocol of TCAP over IP.

5.2 The architecture of the simulation

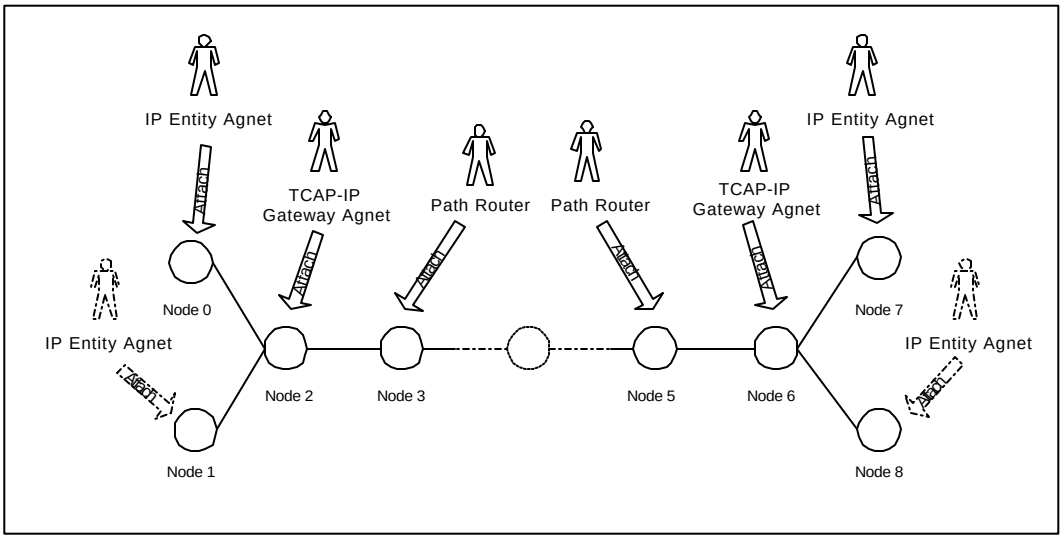


Figure 5.1. Simulation topology

In the simulation, several nodes and links simulation the IP network (Fig 5.1). Those nodes and links are subclasses of NS2 nodes and links. TCAP-IP Gateway, router and IP Entity are implemented as Agents. Those agents are attached to nodes to simulate different entities in the real world.

The network topology and objects in the networks are defined in a TCL file. The NS2 has an imbedded TCL interpreter to receive the TCL file from a console. The interpreter evaluates each line in the TCL file in a globe context.

The TCL file has three parts. The first one is to define the network topology. This part declares nodes and links in the network and sets the relation of each pair of nodes and links. The second part declares and initializes agents and attaches the agents to nodes according to the topology. For each object declared in the TCL file, the interpreter builds a compiled object and inserts the compiled object into the NS2 object table. All parameters for each object are specified in the TCL file. The third part of TCL file defines the event in the simulation. This part is a composite of commands to agents, links and systems. A sample TCL file is given in appendix A.

5.2.1 The node and link in the simulation

The node itself is a standalone class in Otcl. However, most of the components of the node are themselves TclObjects. The typical structure of a node is as shows in Fig 5.2.

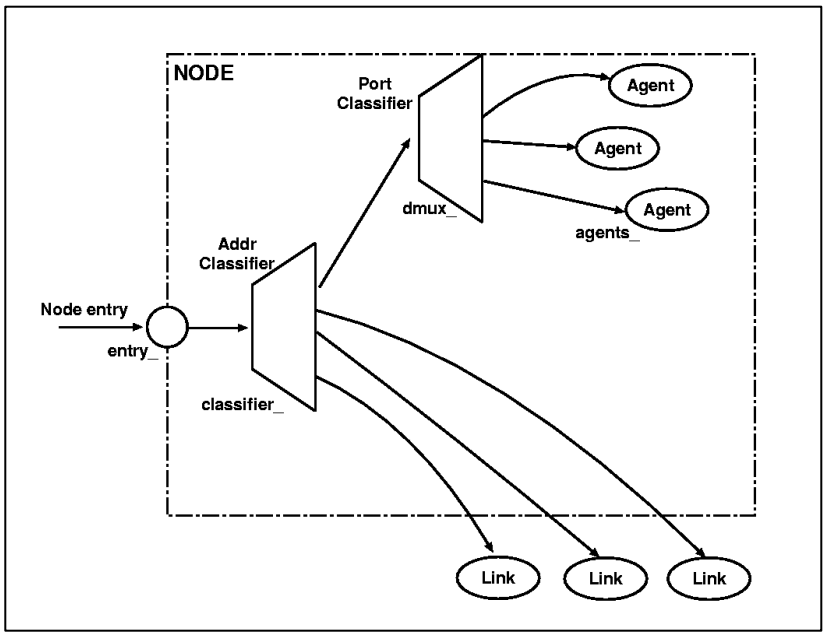


Figure 5.2: The NS2 node architecture

The function of a node when it receives a packet is to examine the packet's fields, usually its destination address, and on occasion, its source address. It should then map the values to an outgoing interface object that is the next downstream recipient of this packet. In NS2, a simple classifier object performs this task.

Most links in this simulation are simplex-links. This class forms a unidirectional link from one node to another with an associated queue and delay.

5.2.2 The agents in the simulation

In NS2, agents represent endpoints where network-layer packets are constructed or consumed, and are used in the implementation of protocols. In this simulation, most protocol functions are implemented on three agents. The are TCAP-IP Gateway, IP Entity and Path Router.

5.2.2.1 TCAP-IP Gateway (TIPG) function

TCAP-IP Gateway functions as follows.

1. The TIPG can receive command and parameters from the TCL interpreter. When it receives a TCL command, it needs to parse the command and do some operation. TCL interpreter can set the TIPG's basic processing delay, workload processing delay, receiving buffer size, queuing algorithm on receiving buffer and default gateway and start or stop the TIPG. TIPG has a command() function for handling the TCL command from the NS2 imbedded TCL interpreter.

2. The TIPG encapsulates/decapsulates the TCAP messages.

3. The TCAP-IP gateway must provide the encapsulation/decapsulation functions at both directions from the TIPG to the IP entity and vice versa.

4. The TIPG builds the route table.

5. The TIPG sends the registration message to other gateway at the beginning of the simulation. The registration message has the sender's address. When the TIPG receives the registration message, it will update its route table.

6. The TIPG routes the STIPP messages.

7. The TIPG forwards the STIPP messages from its attached IP entities to other TIPG gateway according to the STIPP addresses and its route table. If it finds the STIPP destination is not on the route table, it drops the STIPP message.

8. The TIPG applies the processing delay and queuing delay

   TIPG has a receiving buffer. The buffer maintains a FIFO queue. All the incoming packets will first be inserted into the queue. The TIPG will examine the packet type, compute processing delay for this packet and if this packet is routable, the TIPG will schedule a send event. For example, a TCAP request arrives at the queue. If the TIPG is not processing another packet, it will be notified that a packet arrives. The gateway gets the packet from the queue, examines its type and knows it is a TCAP request. Then the gateway does the necessary operation on this packet and scans a processing timetable to get the processing time for this packet. Finally, the TIPG schedules a send event and falls into idle. When the send event happens, the gateway will wake up, send the packet and ready for the next packet. If the workload related processing delay has been set, the gateway can handle up to k packets at the same time. That means the gateway can get up to k packets from the queue at one time if there are more than k packets in the queue. This simulates the multi processor gateway in the real world.

   The total delay for one packet can be the sum of the basic processing delay, queuing delay and workload related processing delay (Fig 5.3).

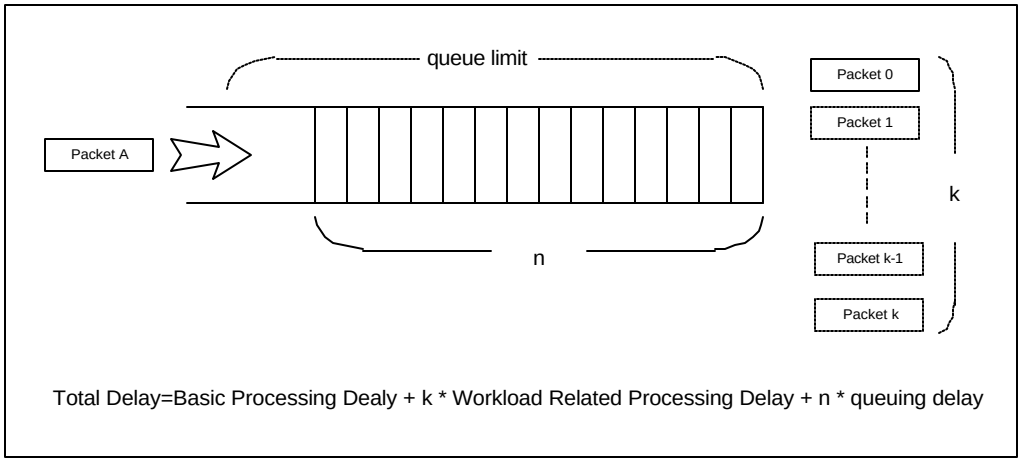


Figure 5.3: The simulation of queuing

The Basic Processing Delay, Workload Related Processing Delay and queuing delay are parameters of TIPG and can be set in the TCL file.

9. The TIPG monitors the status of the receiving queue.

   The TIPG keeps track of the length of the queue in the receiving buffer. If the length is equal to the buffer limit length, the following incoming packets will be dropped. TIPG records the number of received packets, forwarded packets and dropped packets.

10. The TIPG generates statistic report.

    TIPG can generate a statistic report for the period from the beginning of the simulation to the time it receives a report command. The report includes:

    The number of STIPP packets the TIPG received.

    The number of STIPP packets the TIPG forwarded.

    The number of STIPP packets the TIPG dropped due to the oversize of the queue.

The drop rate that equals (the number of STIPP packets TIPG dropped) / (the number of STIPP packets the TIPG received).

5.2.2.2 IP Entity functions

In the real network, the IP entity can be a media gateway, application server or other program that will use the TCAP to communicate with the SS7 network. Its functions are as follows.

1. The IP entity receives commands and parameters.

   The IP entity can receive commands and parameters from TCL interpreter. IP entity has a command function for handling the TCL command from the NS2 imbedded TCL interpreter. When it receives a TCL command, it needs to parse the command and do some operation. The TCL interpreter can set the IP entity's basic processing delay, workload processing delay, receiving buffer size, queuing algorithm on receiving buffer and default TCAP-IP gateway and start or stop the IP Entity sending the STIPP packets.

2. The IP entity encapsulates/Decapsulates the TCAP messages.

   The IP entity must provide the encapsulation/decapsulation functions in both directions from the TIPG to the IP entity and vice versa.

3. The IP entity generates STIPP traffic.

   IP entity can generate STIPP traffic. The speed and STIPP package size is configured in the TCL file. In this simulation, the STIPP traffic is in TCAP query type.

4. The IP entity monitors the TCAP transaction.

   In this simulation, the IP entity generates the STIPP traffic. The STIPP traffic is in TCAP query type. When the destination IP entity receives this TCAP query, it applies a processing delay on it and sends a result back. The source IP entity starts a timer when it sends out a TCAP query message. If it doesn't receive a result or error message from the

destination IP entity within a pre-fixed time, the timer will timeout, and this TCAP transaction fails.

5. The IP entity processes the TCAP queries and results.

   IP entity has a receiving buffer. The buffer maintains a FIFO queue. All the incoming STIPP packets will first be inserted into the queue. In this simulation, STIPP traffic is in TCAP query type. IP entities process the packages in the receiving buffer. If the incoming STIPP package is a TCAP query, the IP entity computes the queuing delay and processing delay for this packet, generates a TCAP result package, inserts it into the out-going buffer and schedules a send event. If the incoming STIPP package is a TCAP query, IP entity counts the delay that equals the processing time of the TCAP result package minus the sending time of the TCAP query. If the delay is in the affordable range, this TCAP transaction succeeds. Otherwise, the TCAP transaction fails. If the incoming STIPP package is a TCAP result error, the TCAP transaction fails.

6. The IP entity generates a statistic report.

   IP entity can generate a statistic report for the period from the beginning of the simulation to the time it receives a report command. The report includes:

   The number of STIPP packets the IP entity received.

   The number of TCAP queries the IP entity sent out.

   The number of TCAP acknowledges the IP entity sent out.

   The number of TCAP errors the IP entity sent out.

   The number of STIPP packets the IP entity dropped due to the oversize of the queue.

   The drop rate that equals (the number of STIPP packets IP Entity dropped) / (the number of STIPP packets the IP Entity received).

The rate of successful TCAP transactions that equals (the number of TCAP results the IP entity received)/(the number of TCAP results the IP entity sent out).

5.2.2.3 Path Router functions

The path router simulates the real world router. It forwards the IP packets between TIPGs. It implies queuing delay and processing delay. Its functions are as follows.

1. The path router receives commands and parameters.

   The path router can receive commands and parameters from TCL interpreter. Path router has a command function for handling the TCL command from the NS2 imbedded TCL interpreter. It can begin or stop forwarding IP packets between TIPGs according to commands.

2. The path router forwards the IP packets.

   Path router has a receiving buffer to hold a FIFO queue for incoming packets. It implies the queuing delay and processing delay for all IP packets using the same algorithm as the TIPG (Fig 5.3). If the control drop rate is set in the TCL file, the router will drop IP packets according to the control drop rate. The control drop rate is the probability of the router dropping IP packets. For example, if control drop rate is 0.001, the router will randomly drop one packet in 1000 packets.

3. The path router generates a statistic report.

   Path router can generate a statistic report for the period from the beginning of the simulation to the time it receives a report command. The report includes:

   The number of IP packets the router received.

   The number of IP packets the router forwarded.

The number of IP packets the router dropped.

The control drop rate that equals (the number of IP packets the router intended dropped) / (the number of IP packets the router received).

The total drop rate that equals (the number of IP packets the router dropped) / (the number of IP packets the router received).

5.3 The protocol in the simulation

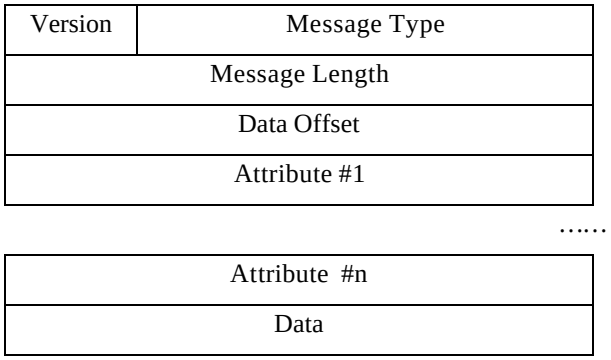
The main interest of this simulation is to test the delay performance of the STIPP kind TCAP over IP protocol, find the security issue and test the performance of the protocol in the IPSEC environment.

This simulation implements a subset of STIPP as the TCAP communication protocol between TIPG and IP entity [1].

5.3.1 STIPP header

5.3.1.1 STIPP header format

The format of the STIPP Header is shown below. The fields are transmitted from left to right.



* Version

The Version field indicates the version number of the received STIPP packet. In this simulation, this field must set to 1 to indicate STIPP Version 1.

* Message Type

The Message Type field is twelve bits and identifies the type of the STIPP packet. The STIPP Message Types are assigned as follows:

Type number	Type Name
1	Login
2	Login Acknowledgment
3	TIPG Status
4	IP Entity Status
5	TCAP Data
6	TCAP Data Error

* Message Length

The Message Length field indicates the total length of the packet excluding the length of Version, Message Type, and Message Length fields.

* Data Offset

The Data Offset field indicates the offset to the data portion from the first byte of the Attribute #1.

* Attribute

See Attribute Format section for more information of parameter formats.

* Data

The format of the Data field is message-dependent and can be NULL.

Attribute Format

The STIPP attributes carry the specific information for each message type. The format of the attribute is shown below.

Type	Length	Value

* Type

The Type field is one octet. The current assigned values are as follows:

Type number	Type name
1	System Name
2	Subsystem Number
3	TIPG Status
4	IP Entity Status
5	Additional Status Information
6	Protocol Type
7	IP Address
8	Calling Party Address
9	Called Party Address
10	Error Reason

* Length

The Length field indicates the length of the Value field.

* Value

The Value field is zero or more octets and contains information specific to the attribute. The Type field determines the format of the Value field.

Attribute name	Attribute value
System Name	Contains the system name to be authenticated by the TIPG. The Login message type normally uses this field.
Subsystem Number	Indicates to the Subsystem Number of an IP Entity. The Login message type normally uses this field.
TIPG Status	Contains the status of the TIPG and is normally used within the TIPG Status message type.

IP Entity Status	Contains the status of the IP Entity and IP Entity Status message type normally uses it.
Protocol Type	Contains the protocol type of the TCAP message.
IP Address	Contains the IP address of the IP Entity. It is optional and used within the message types sent from IP Entity to the TIPG.
Calling/Called Party Address	Contains the SCCP Party Address parameters.
Error Reason	Contains the reason that the IP Entity or the TIPG cannot handle a TCAP message. It is normally used within the TCAP Data Error message type. The Error Reason would be mapped to Error in UDTS or XUDTS message.

5.3.1.2 Protocol Messages

The following defines what attribute is mandatory or optional in each message type.

Login

IP Entity sends this message type to the TIPG to begin the login process. The TIPG gets the subsystem number of IP entity from this login message and puts it in the addressing table for routing the traffic from TIPG to IP Entity.

Attribute
System Name
Subsystem Number
Identification
Protocol Type

Login Acknowledgment

TIPG sends this message type to the IP Entity in response to a Login message.

TIPG Status

TCAP Data

This message type is sent in both directions from the TIPG to the IP entity or from the IP Entity to the TIPG. Both the TIPG and IP entity use this message type to carry the actual TCAP data. The Data portion contains the actual TCAP message.

Protocol Type
Called Party Address
Calling Party Address

TCAP Data Error

This message type is sent in both directions from the TIPG to the IP entity or from the IP Entity to the TIPG if the IP Entity or the TIPG cannot handle a TCAP message. The Data portion is a copy of the Data portion of the corresponding message of the TCAP Data message type.

Error Reason
Called Party Address
Calling Party Address

5.3.2 Scenarios

The following section describes scenarios to illustrate how this subset STIPP works.

5.3.2.1 IP Entity-Initiated Transaction

This scenario describes a simple exchange of TCAP messages between IP Entities under the same TIPG.

1. The IP Entity builds a Query with Permission TCAP message and sends to the TIPG using the TCAP Data message type.

2. The TIPG performs the IP decapsulation. From the STIPP Header, TIPG gets the Called Party Address. TIPG uses the Called Party Address as the index for checking the route table for the next hop address. If the next hope is an IP address, the TIPG performs IP

encapsulation, puts the IP address into the destination IP address in the IP header and forwards the IP Packets to the next IP node.

3. If the next hope is a router, the router will forward the IP packet to the next hop according to its route table. If control drop rate has been set on this router, this route may drop this packet.

4. The destination IP entity receives the TCAP message and sends it back with a Respond TCAP message to the TIPG.

5. The TIPG checks the destination IP address of the IP packet. If the destination is one of its registered IP entities, it forwards the IP packets to the IP entity. If the destination address is not on the list of the registered IP entities, the TIPG drops it.

6. The IP Entity receives the Response TCAP message and completes the transaction.

This scenario describes a simple exchange of TCAP messages between IP Entities under different TIPGs.

1. The IP Entity builds a Query with Permission TCAP message, and sends it to the TIPG using the TCAP Data message type.

2. The TIPG performs the IP decapsulation. From the STIPP Header, TIPG gets the Called Party Address. TIPG uses the Called Party Address as an index for checking the route table for the next hope address. The next hop should be the IP address of the next TIPG. The TIPG performs IP encapsulation, puts the IP address into the destination IP address in the IP header and forwards the IP Packets to the next TIPG.

3. If the next hop is a router, the router will forward the IP packet to next hop according to its route table. If control drop rate has been set on this router, this route may drop this packet.

4. The destination TIPG receives the IP packets; it checks the destination IP address of the IP packet. If the destination is one of its registered IP entities, it forwards the IP packets to the IP entity. If the destination address is not on the list of the registered IP entities, the TIPG dropps it.

5. The destination IP entity receives the TCAP message and sends it back with a Respond TCAP message to the TIPG.

6. The TIPG forwards the IP packet to the source TIPG.

7. The source TIPG forwards the IP packet to the source IP entity.

8. The IP Entity receives the Response TCAP message and completes the transaction.

5.3.2.2 IP entity generated erroneous TCAP messages

This scenario describes an example of the IP Entity sending an erroneous TCAP message.

The IP Entity builds a TCAP message and sends to the TIPG using the TCAP Data message type.

The TIPG performs the IP decapsulation. From the STIPP Header, TIPG gets the Called Party Address. TIPG uses Called Party Address as an index for checking the route table for the next hop address. The next hop should been the IP address of the next TIPG. The TIPG performs IP encapsulation, puts the IP address into the destination IP address in the IP header and forwards the IP Packets to the next TIPG.

If the next hop is a router, the router will forward the IP packet to the next hop according to its route table. If the control drop rate has been set on this router, this route may drop this packet.

The destination TIPG receives the IP packets. It checks the destination IP address of the IP packet. If the destination is one of its registered IP entities, it forwards the IP packets to the IP entity. If the destination address is not on the list of the registered IP entities, the TIPG drops it.

The destination IP entity receives the TCAP message. It determines that it cannot process the message. It sends it back to the TIPG with an UDTS (or XUDTS) message.

The TIPG forwards the IP packet to the source TIPG.

The source TIPG forwards the IP packet to the source IP entity.

The IP Entity receives the Response TCAP message and performs its own error handling.

Chapter 6 TCAP over IP Experiments and results

6.1 TCAP over IP on traditional IP network

6.1.1 Query response delay simulation

As analyzed in chapter 4.1, the TCAP has a delay requirement. In PSTN, the TCAP has an average delay requirement in range of 690-726 ms. In VoIP system, a subscriber can call an 800 number (Fig 6.1). An intelligent network model that has a TCAP query capability resides in the Call Server (CS). When the Call Server receives the 800 number, the intelligent model builds a Query TCAP message and sends it to the TIPG. The TIPG forwards the TCAP query to SCP. The SCP receives the TCAP message and sends it back with a Respond TCAP message to the TIPG. TIPG performs the SS7 decapsulation/IP encapsulation and then sends it to the CS using the TCAP data message type. The intelligent model in CS receives the TCAP message and retrieves the destination telephone number then returns the number to CS. CS uses the destination number to route the ISUP message.

It is critical for TCAP over IP protocols to provide a comparable delay for TCAP transaction. Otherwise, the increased TCAP delay can make the whole call processing procedure fail due to over time.

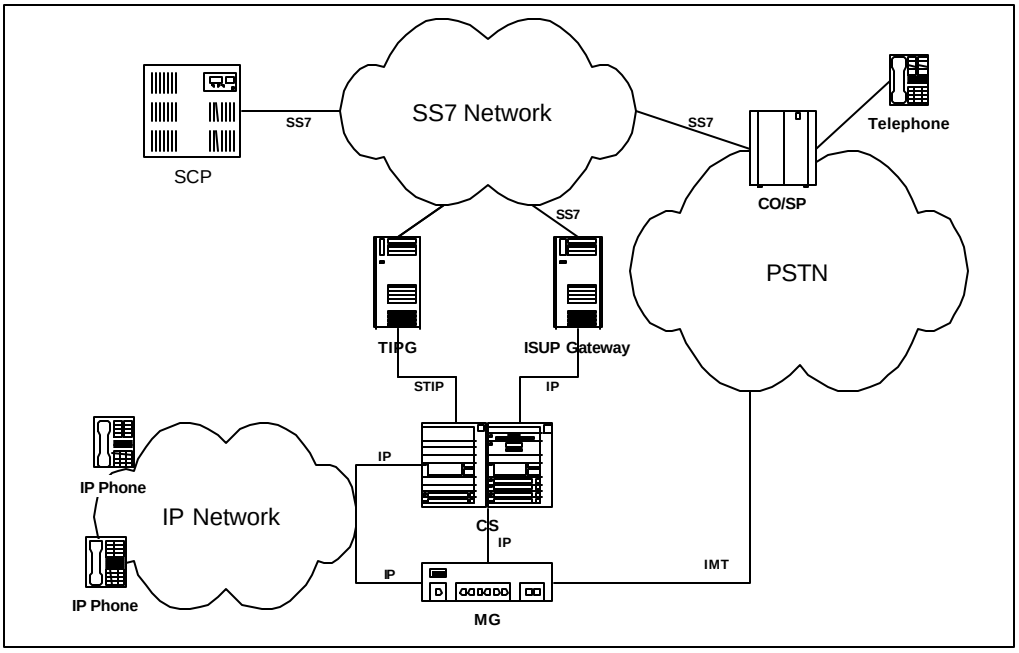


Figure 6.1: Signaling internetworking between SS7 and IP network

6.1.1.1 Topology and parameters

This delay simulation uses NS2 to set up a test bed to simulate a TIPG to precede a TCAP query transaction. In the Query message, a time stamp is inserted to record the sending time when the IP entity sends the Query message to TIPG. When TIPG gets the Query message, it waits for a switch TCAP response time and sends back a Response message. TIPG also copies the sending time in the Response message. All queuing delay and processing delay will be applied in each node that the Query messages and Response passes by. When the IP entity gets the Response message, it subtracts the starting time from the current system time in the Response message. Therefore, we can get an end-to-end TCAP delay for TCAP over IP. Figure 6.2 shows the topology for this simulation. This simulation is to check whether the STIPP can meet the delay requirements for TCAP and illustrates the TCAP flow that contributes to the delay and analyzes the total TCAP delay in terms of delay in network elements.

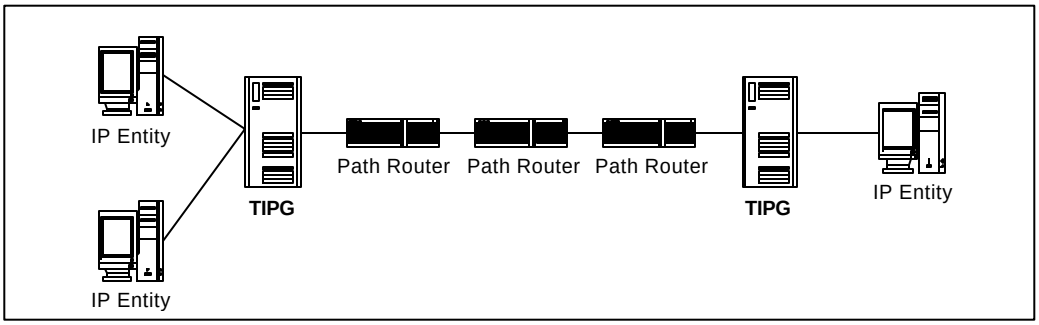


Figure 6.2: Simulation of STIPP

For simplicity, the simulation uses an IP entity to simulate the intelligent model in CS and assumes a resource in the IP network that can provide the same function as SCP. The simulation uses another IP entity to simulate this resource. For comparing the performance of the TCAP-IP system with that of the PSTN, we assume network elements in each system have comparable processing time for executing the same or similar functions. The comparison then can be made based on the system complexity (i.e. number of components) and the set of messages (i.e. the number and the type of commands) need to be exchanged and executed. More specifically, we assume the TCAP response time at IP entity to process a TCAP query is comparable with that of a SCP, which has a mean value of 210-220 ms and a 95% of probability not exceeding 359ms. The TIPG relays signaling messages between the PSTN and CS. It is assumed to act like a STP in the PSTN, and the Cross-STP delay is used for the basic processing delay in TIPG. The transmission Delay in an IP network has different characteristics from that in an SS7 network. Some experimental data were gathered from Cisco router and were applied them for the purpose of this simulation.

For ease of manipulation, we define:

T_ip_p: IP Entity processing time for handling the TCAP query.

T_ip_q: IP Entity queuing delay parameter.

T_tipg_p: TIPG processing time for forwarding the TCAP message.

T_router_p: Path Router processing time for forwarding the IP packet.

In summary, the tentative statistics I use for this simulation is as in table 6.1:

Table 6.1: Network Element Processing Time in Simulation

	50%	95%
T_ip_p	222	354
T_ip_q	1	1
T_tipg_p	20	40
T_router_p	3	3

The topology in the simulation is as follow:

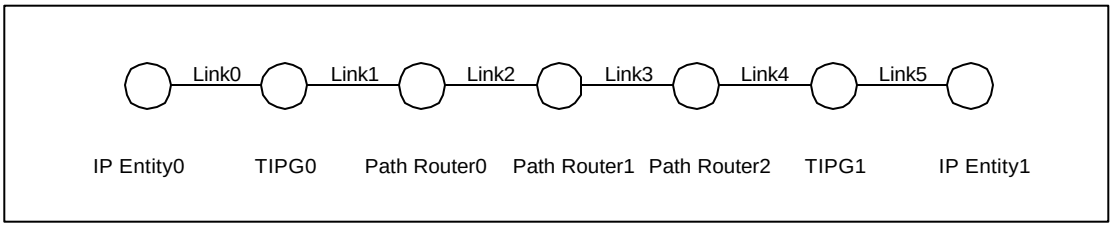


Figure 6.3: The topology of the delay simulation of STIPP

The whole simulation procedure is as follows (Fig 6.4):

1. Network Initialization, including IP Entities login, TIPG multicast status, set up route table for TIPGs and Path Routers.

2. IP Entity0 sends a TCAP Query with Permission message to IP Entity1.

3. TIPG0 gets the Query message, looks up the route table and forwards the message to TIPG1 through Path Router0, Path Router1 and Path Router2.

4. Path Router0, Path Router1, Path Router2 apply the queuing delay and processing delay.

5. TIPG1 gets the Query message, looks up the route table, does the decapsulation and forwards the message to IP Entity1.

6. IP Entity1 applies TCAP response time and queuing time, copies the sending time from the receiving Queuing Message to Response Message, sends the Response message to IP Entity0.

7. TIPG1 gets the Response message, looks up the route table, does the encapsulation and forwards the message to TIPG0 through the Path Router2, Path Router1 and Path Router2.

8. Path Router0, Path Router1, Path Router2 apply the queuing delay and processing delay.

9. TIPG0 gets the Response message, looks up the route table, does the decapsulation and forwards the message to IP Entity0.

10. IP Entity0 receives the Response message, uses the sending time in the Response time to subtract the current system time and puts the delay of TCAP Query into trace file.

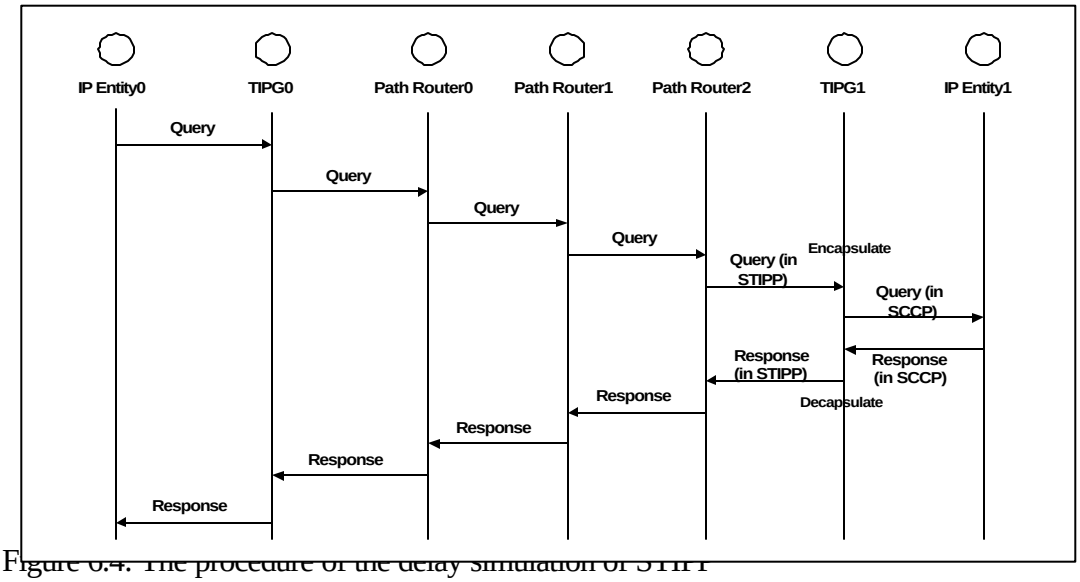


Figure 6.4: The procedure of the delay simulation of STIPP

The parameters for each object are as follows:

Parameter name	Parameter value
The bandwidth for each link	1Mbps
The delay for each link	5ms
The queue attached on each link	DropTail
The basic processing delay for IP Entity1	222 (50% pro.) / 354 (95% pro.)
The workload related processing delay factor for IP Entity1	1
The basic processing delay for IP Entity0	1ms
The queuing delay factor for IP Entity	1ms
The queue size for each IP Entity	1000
The TCAP Query messages package size	1024 bytes
The basic processing delay for TIPG	20 ms
The workload related processing delay factor for TIPG	1
The queue size for each TIPG	1000
The processing delay for each router	2ms
The queuing delay factor for TIPG	1ms
The control drop rate on each node	0

6.1.1.2 Results of the experiments

The Figure 6.5 shows the results of the simulation. The results show that the delay for TCAP query has a mean value of 441ms and a 95% of probability not exceeding 573ms. The current PSTN user can get a call set-up delay around 500ms and one TCAP query delay around 200-300ms. Although the delay of TCAP over IP is larger than this value, the delay of TCAP over IP is still in the range of the requirement of TCAP delay. Moreover, as the hardware capabilities of TIPG and routers increase, we expect better performance for the delay of TCAP over IP.

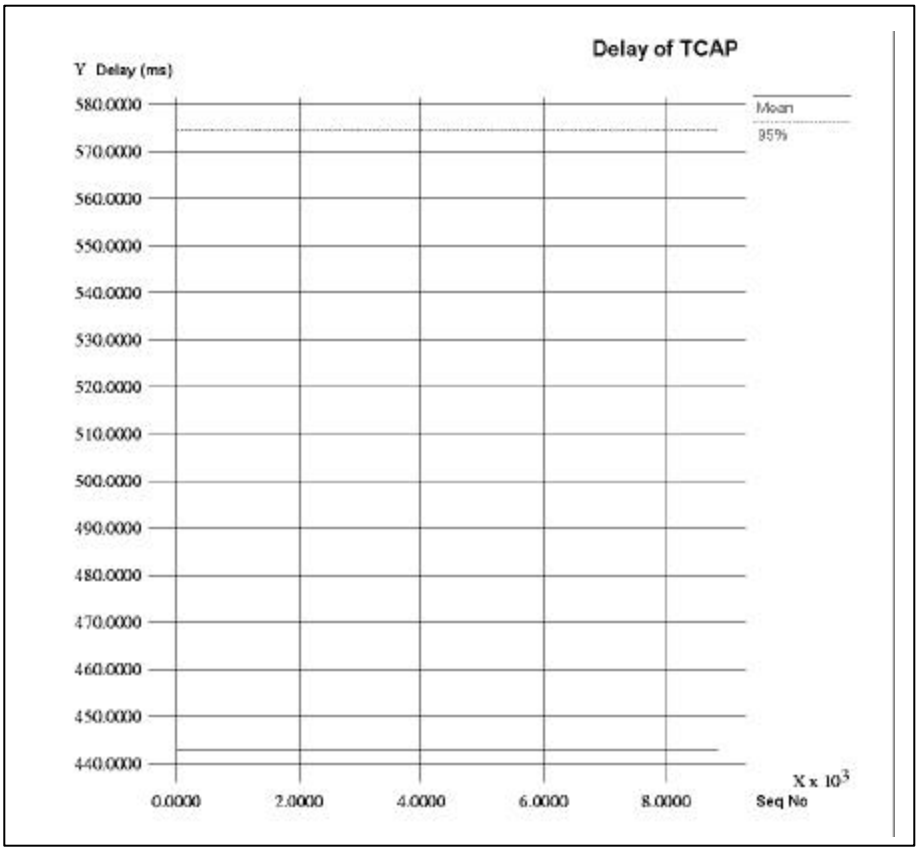


Figure 6.5: The results of the delay simulation of STIPP

6.1.2 Flood attack simulation

The performance of the TCAP-IP protocol under flood attack is another interesting topic in this simulation. In the SS7 network, SCP is in a high secure environment because few people can access the network device. However, in an IP network, most gateways and routers are in an explosive environment. The most common attack is to deny service. We can imagine hackers floods a fatal number of TCAP Query messages to TIPG. Most normal TCAP Query messages will be dropped at the receiving buffer at the routers or TIPGs.

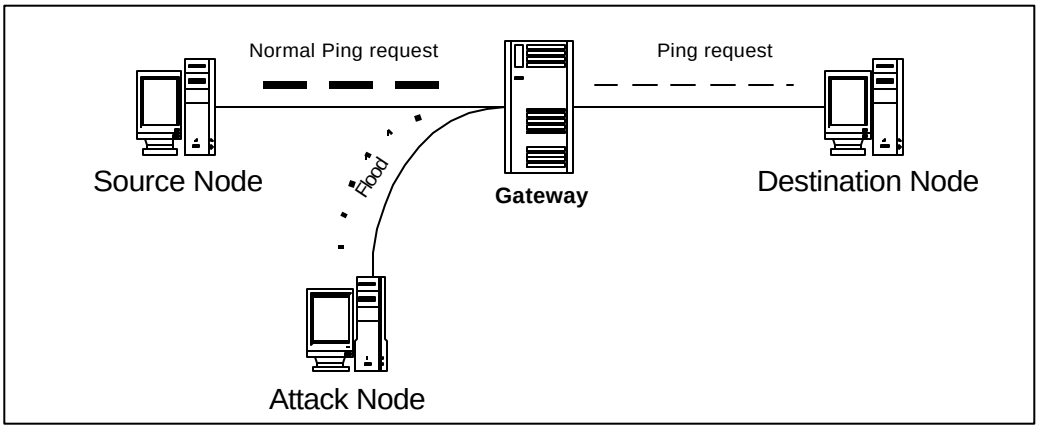


Figure 6.6: An attack node floods Query requests to the destination node

6.1.2.1 Topology and parameters

In order to see the effect of the flood attack on the performance of TCAP over IP, we introduce an IP entity to simulate a malicious attack node to flood TCAP message to one TIPG (Fig 6.6).

The topology in the simulation is as follows:

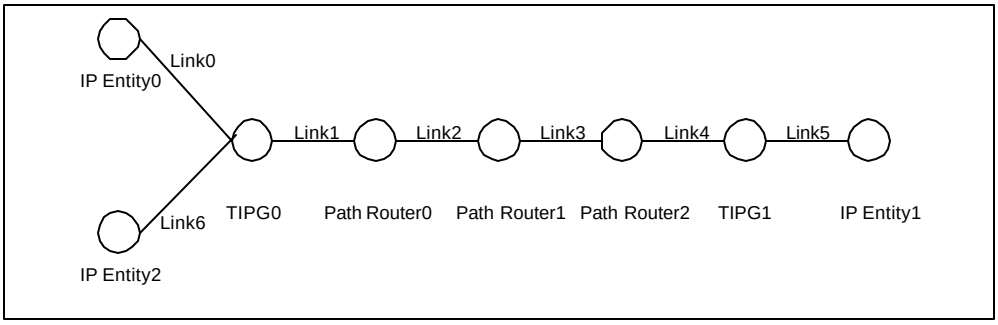


Figure 6.7: The topology of the simulation of flood attack to TIPG

The simulation plays the same procedure as delay simulation (Fig 6.4), plus the IP Entity2 floods Query messages to IP Entity1.

The parameters in the simulation are as follows:

Parameter name	Parameter value
The bandwidth for each link	1Mbps
The delay for each link	5ms
The queue attached on each link	DropTail
The basic processing delay on IP Entity1	222 (50% pro.) / 354 (95% pro.)
The basic processing delay on IP Entity0	1ms
The workload related processing delay factor for IP Entity1	1
The queuing delay factor for IP Entity	1ms
The queue size for each IP Entity	1000
The TCAP Query messages package size	1024 bytes
The processing delay for each router	2ms
The queue size on each TIPG	1000
The queuing delay factor on TIPG	1ms
The workload related processing delay factor on TIPG	1
The basic processing delay on TIPG	20 (50% pro.) / 40 (95% pro.)
The interval for IP Entity2 floods Query message	100ms
The control drop rate on each node	0

6.1.2.2 Results of the experiments

The results of the simulation (Fig 6.8) show the flood attack lets the delay of TCAP increase sharply. The TIPG almost is denied of service.

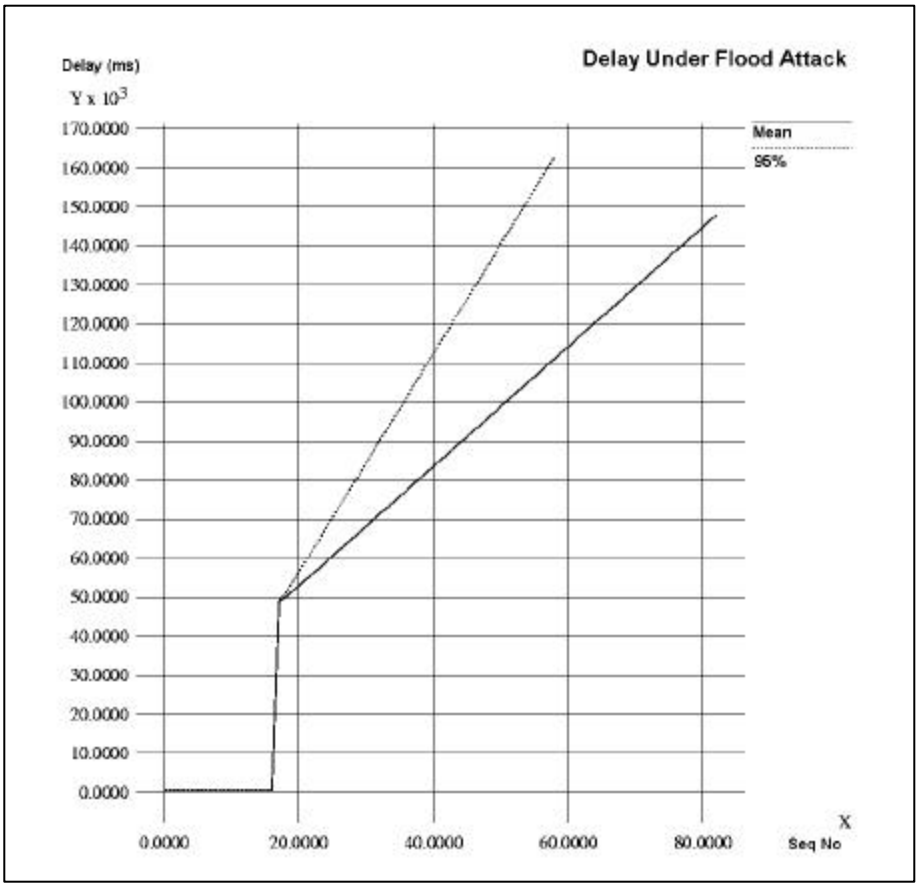


Figure 6.8: The results of the simulation of flood attack to TIPG

6.2 TCAP over IP in Diffserv environment

6.2.1 Query response delay simulation

In most cases, the signaling transmission over IP has a high priority and a QoS mechanism should be applied to provide a bounded end to end delay, jitter and throughput capacity. Currently, several QoS models can provide some degree of QoS control service. This paper applies the Differentiated Service (Diffserv) patch on NS and simulates the TCAP over IP on

Diffserv. The advantage of Diffserv is it is simple to implement and brings lower overhead compared to RSVP. In RSVP, each participating node needs to keep states for every data flow individually and perform signaling at each hop. By contrast, the Diffserv defines a scalable service discrimination policy without the need for per–flow state and signaling at each hop. Instead of maintaining and processing the states for each flow in each node on the path, Diffserv defines a way for the network edge to communicate service requirements to the Internet backbone providers.

This simulation builds a testbed as Figure 6.9.

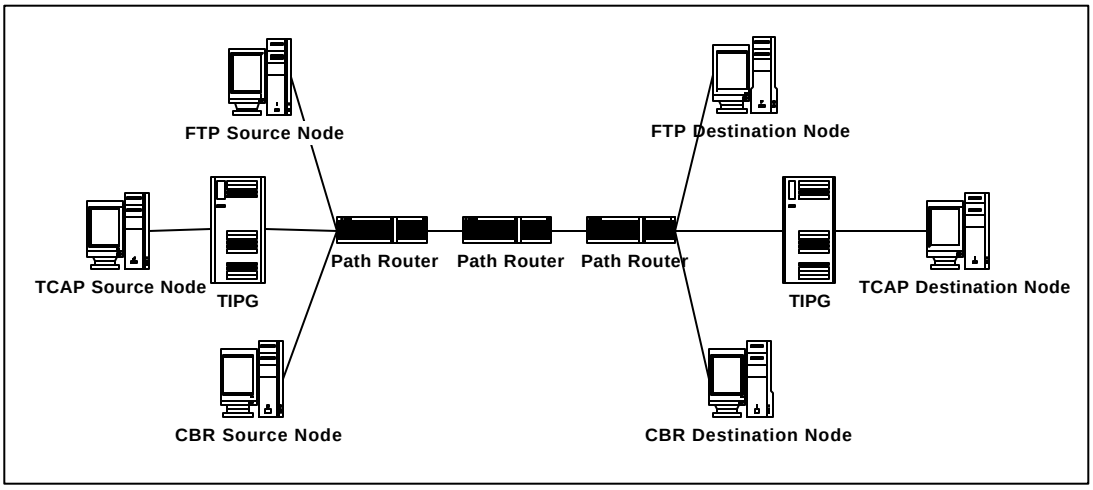


Figure 6.9: Testbed for Diffserv

6.2.1.1 Topology and parameters

We introduce two pair of nodes into TCAP over IP system. One pair of nodes uses FTP to transfer files in the same path that the TCAP message goes through. The other pair of nodes generates CBR traffic in the same path that the TCAP message goes through. We did three experiments on this testbed. In one experiment, all traffic runs on the normal IP network. The network promises nothing but best-effort services. It is the same as what we meet on the

current Internet. In other two experiments, we apply Diffserv on the network. All traffic from source nodes is marked, and conditioners and schedulers are inserted as in Figure 6.10.

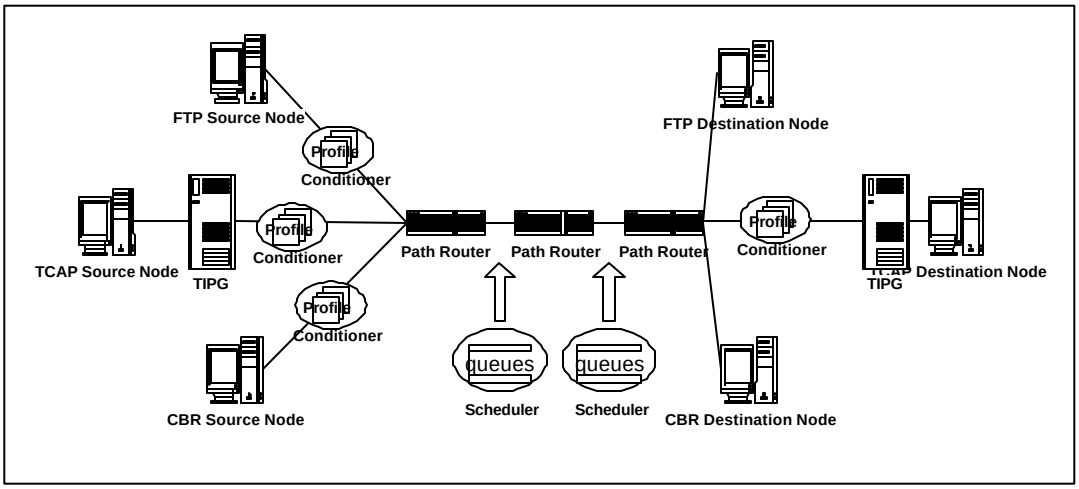


Figure 6.10: Diffserv applied on the testbed

The conditioner is installed between a link and a node. This can be viewed as being equivalent to installing a conditioner at the input interface to a node. The conditioner is installed before the node because of the way the node is implemented. All the traffic enters the node at the same point once the traffic has entered the node. Profiles are added into a conditioner. The profile is the essential notion of the Diffserv. In the Diffserv architecture, the customers specify how much of the network resources they will require using a profile. The profile then contains information describing one customer's traffic requirements, or a component of the traffic requirements. In the simulation, the scheduler is a new NS class that contains 3 queues. One is for EF. One is for AF, and one is for BE traffic. The EF queue is a simple drop-tail queue; the AF queue is a RIO queue; and the BE queue is a RED queue. The scheduler scans the traffic on the link, puts incoming packets into different queues and applies a WRR algorithm to service those three queues and schedules the packet forwarding according to the configuration. Upon the result of the experiments on lab, the route normally needs additional 10ms for tagging and implementing

the WRR respectively. In our Diffserv experiment 1, we make all the TCAP, FTP and CBR traffic marked as BE. In Diffserv experiment 2, we make the TCAP traffic as EF and both the FTP and CBR traffic as BE. The experiment topology is show as Figure 6.11. The system parameters are as follows.

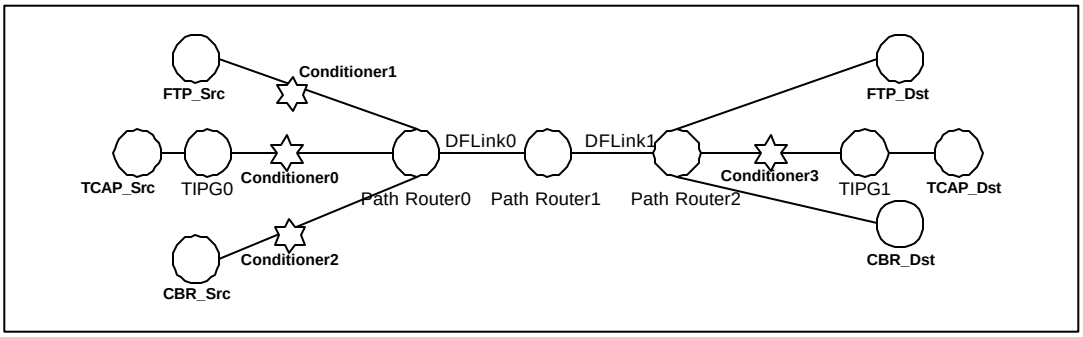


Figure 6.11: Topology for Diffserv testbed.

The parameters in the simulation are as follows:

Parameter name	Parameter value
The bandwidth on each link	1Mbps
The delay on each link	5ms
The queue attached on each normal link	DropTail
The TCAP Query messages package size	1024 bytes
The basic processing delay on IP Entity1	222 (50% pro)
The basic processing delay on IP Entity0	1ms
The workload related processing delay factor for IP Entity1	1
The queuing delay factor on IP Entity	1ms
The queue size on each IP Entity	500
The basic processing delay on TIPG	20ms (50% pro)
The workload related processing delay factor on TIPG	1
The queuing delay factor on TIPG	1ms
The queue size on TIPG	1000
The TOS of the FTP and CBR traffic	BE
The TOS of the normal TCAP message	EF
The CBR sending rate	1Mbps

The processing delay on each router	12ms(mean)
The control drop rate	0
The scheduling algorithm on Diffserv	WRR
The processing time for conditioner	10ms

6.2.1.2 Results of the experiments

The results are listed in Table 6.2 and partial results are as show in Figure 6.12.

Table 6.2: The result of the experiments on Diffserv testbed

	TCAP Query Response Delay (ms), based on 1000000 Queries. Min/Max/Mean	
	Before the FTP and CBR start	After the FTP and CBR start
Normal IP Network	447/447/447	483/890/631
Diffserv Network, all traffic is marked as BE (Diffserv 1)	522/522/522	643/1307/831
Diffserv Network, TCAP messages are marked as EF, attack traffic is marked as BE (Diffserv 2)	522/522/522	524/537/527

Comparing the results of normal IP network and the Diffserv 1, we can see Diffserv brings overheads for tagging and scheduling. Moreover, the best-effort network cannot guarantee a bounded network delay for the signaling. After the FTP and CBR traffic enters the network, the TCAP delay increases. There are almost 400-500ms additional delays. The TCAP cannot afford it. The signaling transmission definitely requires a QoS mechanism on the network. Comparing the results of Diffserv 1 and Diffserv 2, we can see the delay due to the effect of FTP and CBR traffic is reduced to 15ms. That means Diffserv reduces the effect between different priority traffic and provides some kind QoS for the EF traffic.

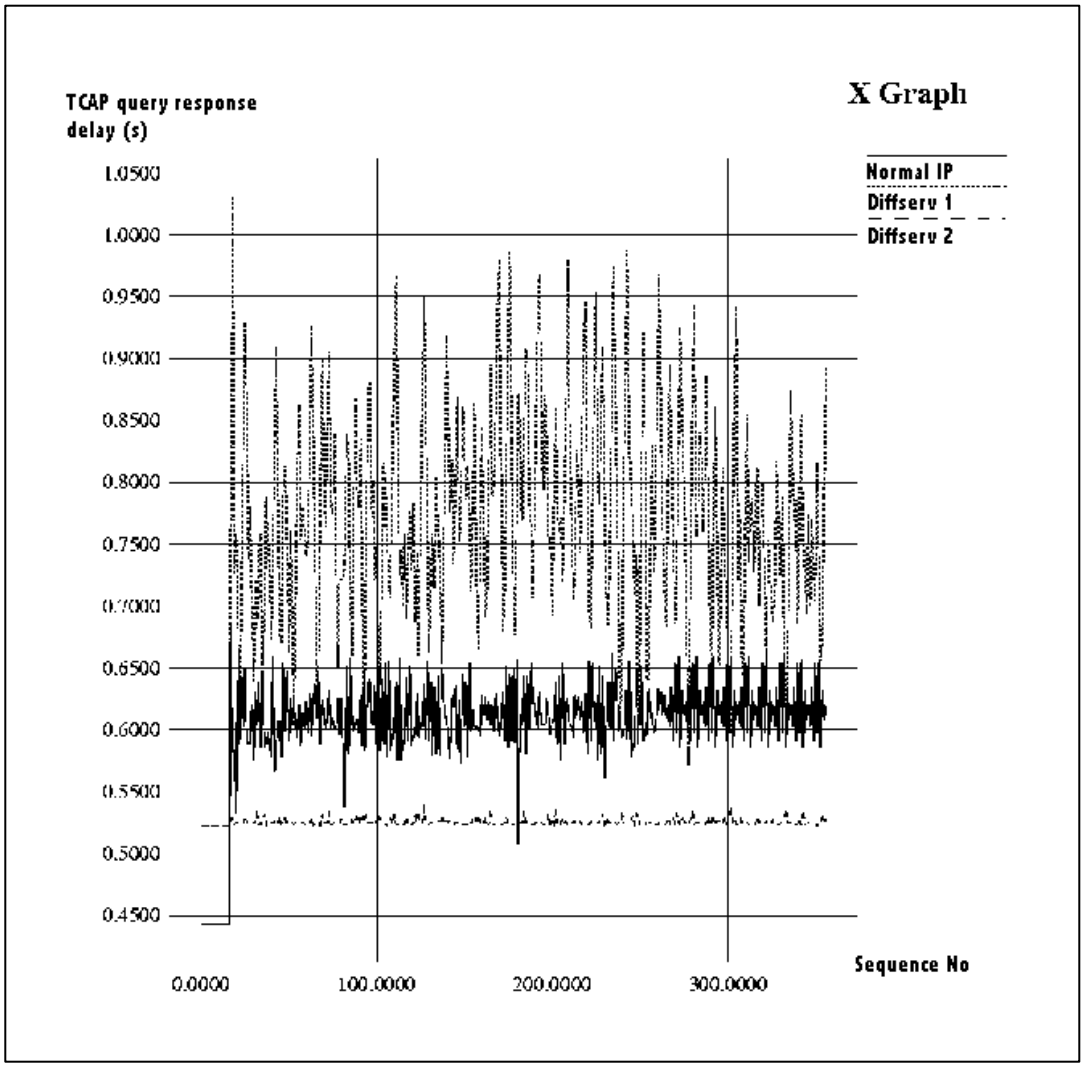


Figure 6.12: Partial results of the experiments on Diffserv testbed.

6.2.2 Flood attack simulation

6.2.2.1 Topology and parameters

In order to get the performance of TCAP over IP in Diffserv environment, we introduce three attack nodes into the system (Fig 13). Two attack nodes are attached to the TIPG with the TCAP Source node. One additional TIPG and attack node are connected to the router. We launch two flood attacks in the experiments. One is from the two attack nodes attacked to TIPG0. This simulates the flood attack from the source side. The other is from the attack node that is attached to TIPG2. That simulates an attack in the middle of the path. The topology shows in Figure 6.13.

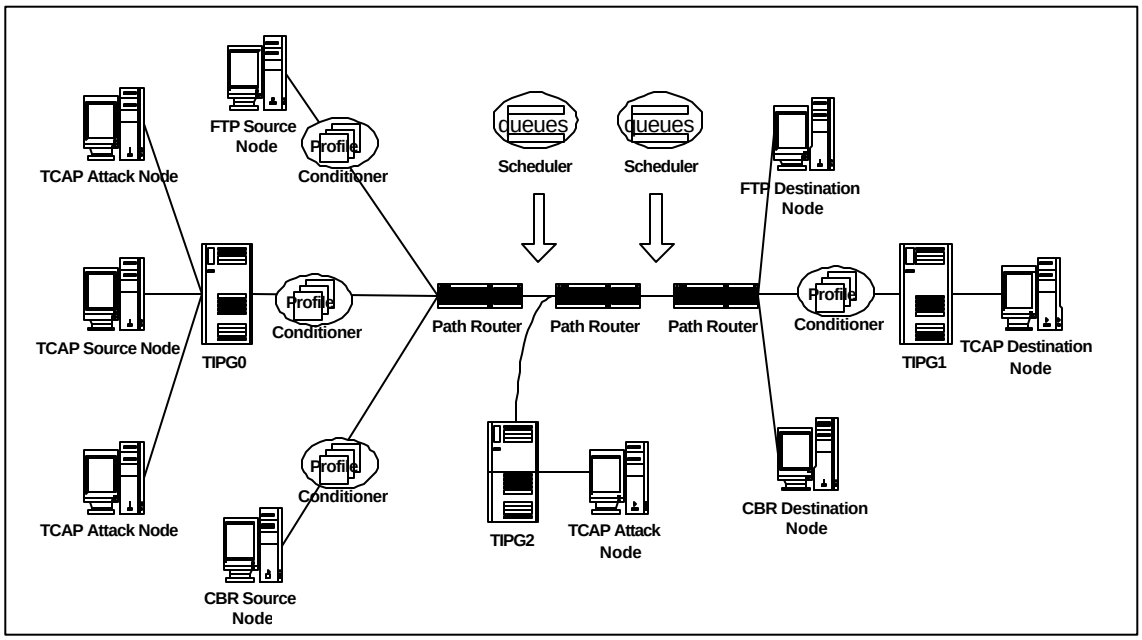


Figure 6.13 Flood attack to TCAP over IP on Diffserv

The parameters in the simulation are as follows:

Parameter name	Parameter value
The bandwidth on each link	1Mbps

The delay on each link	5ms
The queue attached on each normal link	DropTail
The TCAP Query messages package size	1024 bytes
The basic processing delay on TCAP destination node	222ms (50% pro)
The basic processing delay on TCAP source & attack nodes	1ms
The workload related processing delay factor on TCAP nodes	1
The queuing delay factor on TCAP nodes	1ms
The flood interval on the TCAP attack nodes	100ms
The TOS of the normal TCAP message	EF
The TOS of the attack TCAP message	BE
The TOS of the FTP and CBR traffic	BE
The CBR sending rate	1Mbps
The queue size for each TCAP node	1000
The basic processing delay for TIPG	20ms(50% pro)
The workload related processing delay factor for TIPG	1
The queuing delay factor for TIPG	1ms
The queue size for each TIPG	1000
The processing delay for each router	12ms(mean)
The control drop rate	0
The scheduling algorithm on Diffserv	WRR
The processing time for conditioner	10ms

6.2.2.2 Results of the simulation

From the results, we can see that both the attacks can increase the delay and deny the service of TCAP over IP. The curve of dsflood1 represents the flood attack from the two attack nodes attached to TIPG0. After the flood attack starts, the delay increases linearly. The curve of dsflood2 represents the flood attack from the attack node attached to TIPG2. After the flood attack starts, the delay jumps to almost 60s.

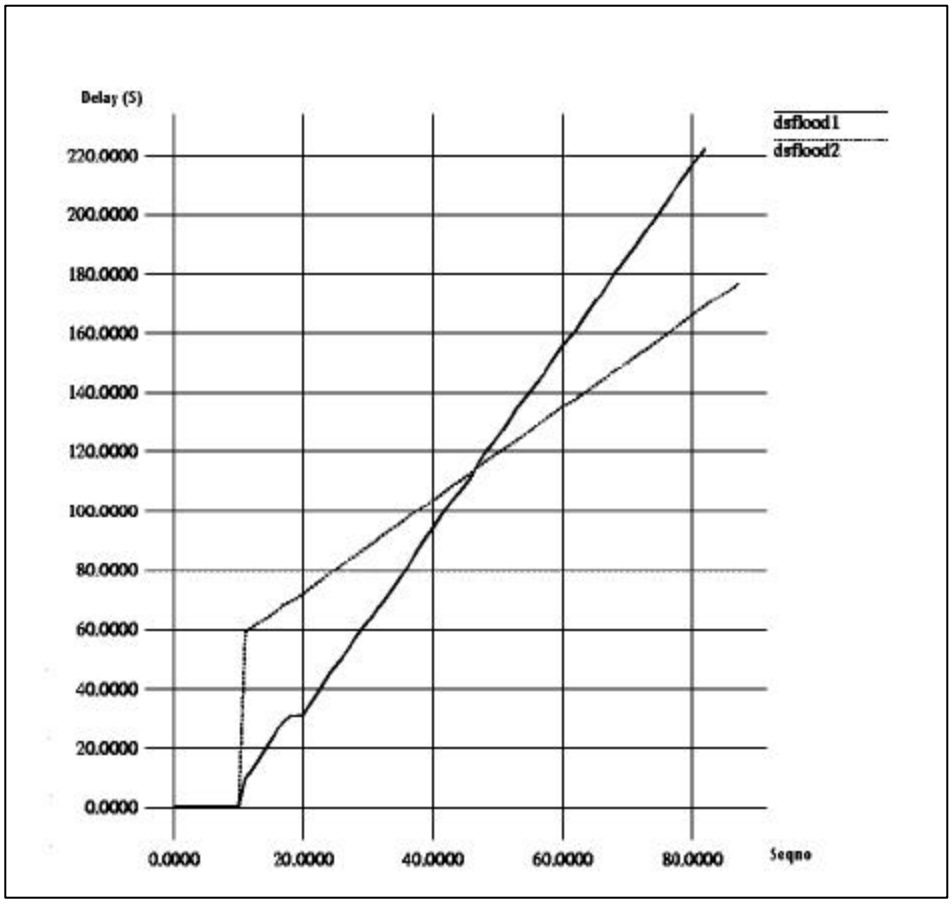


Figure 6.14. The delay of TCAP over IP under flood attack in Diffserv environment

6.3 TCAP over IP in IPSEC environment

Flood attack can seriously degrade the performance of TCAP over IP. Due to the flood attack, a TIPG can be easily denied of service. Since, the call set up procedure may involve TCAP transactions (i.e. 800 number); the flood attack to TIPG can fatally affect the performance of the VoIP system. The solution for preventing the flood attack could be applying security mechanisms to protect the TIPG and routers. The protection of routers from flood attack is

beyond the topic of this thesis. This thesis focuses on the mechanism to protect the TIPG and analyze the effect that the security mechanism brings to the performance of TCAP over IP.

This thesis selects the IPSEC as the protocol to provide the security mechanism for TCAP over IP and analyzes the effect that it brings to the performance.

6.3.1 Query response delay simulation

From chapter 4.2, we get the processing delay of applying IPSEC. In order to get the performance of the TCAP over IP protocol in IPSEC environment, we do a simulation as follows.

6.3.1.1 Topology and parameters

We set up an IPSEC tunnel between two TIPGs (Fig 6.15). We assume the TIPG support IPSEC. TIPG applies the ESP and AH on the IPSEC tunnel. For simulating the IPSEC operation on the TIPG, we increase the basic processing delay with 0.8 ms on TIPG.

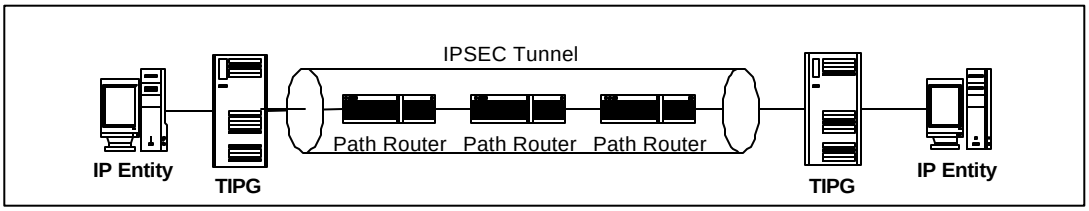


Figure 6.15: Setup an IPSEC tunnel between two TIPGs. All TCAP messages are transferred through the tunnel between the two TIPGs.

The topology in the simulation is as follows:

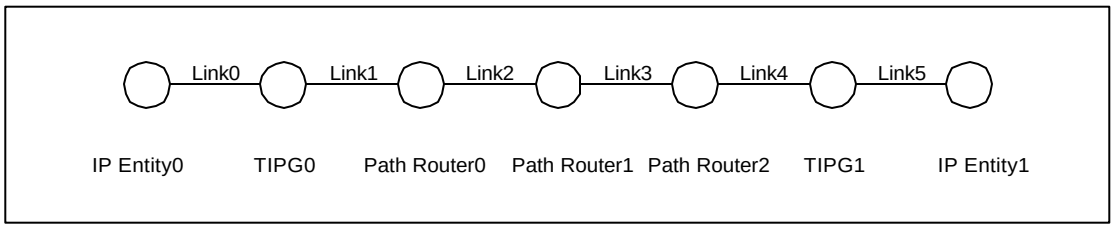


Figure 6.16: The topology for TCAP over IP on IPSEC

The simulation procedure is the same as Figure 6.4. The parameters for each object are as follows:

Parameter name	Parameter value
The bandwidth on each link	1Mbps
The delay on each link	5ms
The queue applies on each link	DropTail
The TCAP Query messages package size	1024 bytes
The basic processing delay on IP Entity1	222 (50% pro.) / 354 (95% pro.)
The basic processing delay on IP Entity0	1ms
The workload related processing delay factor on IP Entity1	1
The queuing delay factor on IP Entity	1ms
The queue size on each IP Entity	500
The basic processing delay on TIPG	20 (50% pro.) / 40 (95% pro.)
The workload related processing delay factor on TIPG	1
The queuing delay factor for TIPG	1ms
The queue size for each TIPG	1000
The processing delay for each router	2ms
The processing delay for IPSEC on IP Entity0 and TIPGs	0.8ms
The control drop rate	0

6.3.1.2 Results of the experiments

Figure 6.17 shows the TCAP delay in the IPSEC environment. The result shows that the delay for TCAP query has a mean value of 444.0 ms and a 95% of probability not exceeding 576ms. It shows that the IPSEC implementation does not bring much increase to delay of TCAP. It is very possible to use the IPSEC to provide the security mechanism for signaling transmission of TCAP over IP.

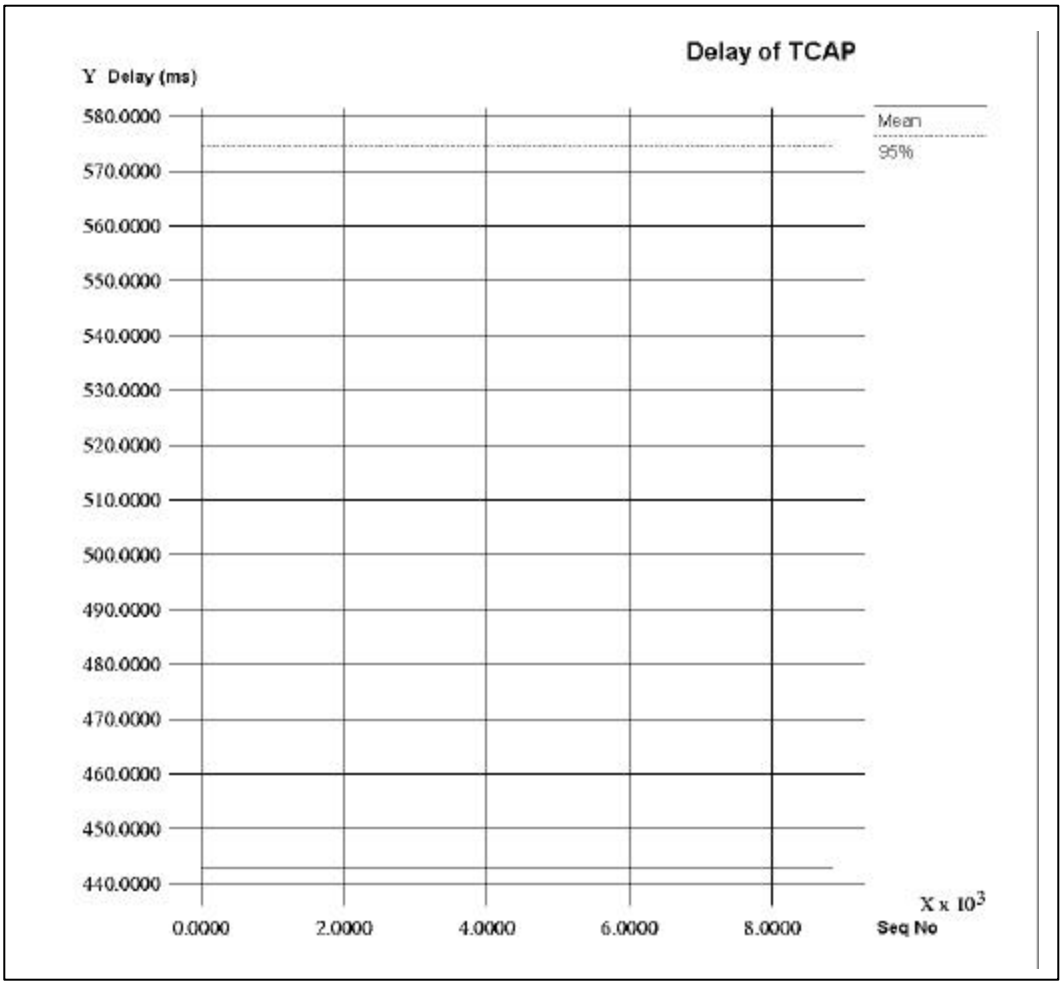


Figure 6.17: Delay of TCAP query over IP on IPSEC

6.3.2 Flood attack simulation

6.3.2.1 TCAP over IP on IPSEC tunnel under flood attack

In order to test the performance of the TCAP over IP under flood attack in IPSEC environment, we introduce an IP entity to simulate a malicious attack node to flood TCAP messages to one TIPG (Fig 6.18).

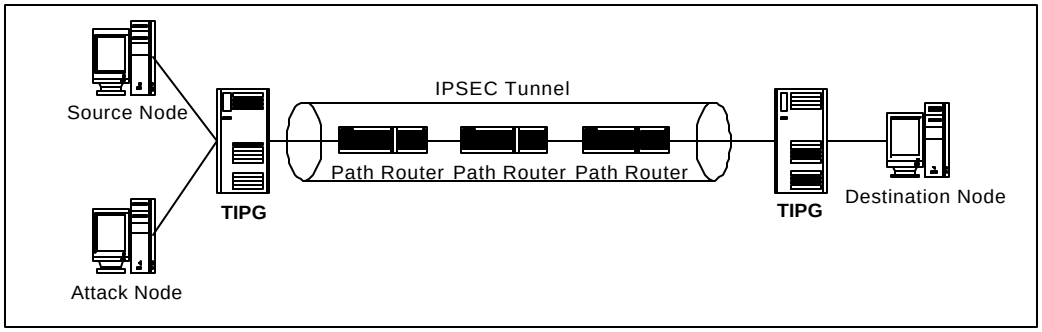


Figure 6.18: An attack node floods TCAP query messages to the destination node through IPSEC tunnel

6.3.2.1.1 Topology and parameters

The topology in the simulation is as follows:

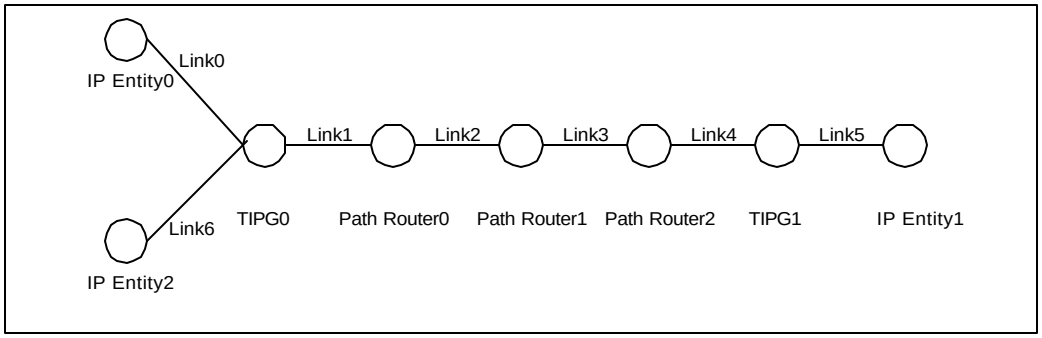


Figure 6.19: The topology of TCAP delay simulation under one node flood attack through IPSEC tunnel

IP Entity2 floods the TCAP Query message to IP Entity1 through the IPSEC tunnel. The simulation procedure is the same as Figure 6.4.

The parameters in the simulation are as follows:

Parameter name	Parameter value
The bandwidth on each link	1Mbps
The delay on each link	5ms
The queue attached on each link	DropTail
The TCAP Query messages package size	1024 bytes
The basic processing delay on IP Entity1	222 (50% pro) / 354 (95% pro)
The basic processing delay on IP Entity0	1ms
The workload related processing delay factor on IP Entity1	1
The queuing delay factor on IP Entity	1ms
The queue size for each IP Entity	1000
The basic processing delay on TIPG	20ms (50% pro.) / 40ms (95% pro.)
The workload related processing delay factor on TIPG	1
The queuing delay factor for TIPG	1ms
The queue size for each TIPG	1000
The processing delay for each router	2ms
The processing delay for IPSEC on TIPGs	0.8ms
The control drop rate	0

6.3.2.1.2 Results of the experiments

Figure 6.18 shows that the TCAP delay increases sharply under the flood attack. Most TCAP Query messages wait in the TIPG receiving buffer. The queuing delay increases sharply. That means that only using IPSEC to protect the packet transmission between TIPGs is not enough. We cannot assume that the IP entities and TIPG are in a secure environment like the SS7 world. IP entities can locate in different sites from the TIPG. In normal cases, the TIPG is centralized

in the service providers'networks. The IP entities can be within other IP routers, application servers, web application servers and so on. So between IP entities and TIPGs there is an explosive world. Service providers need to provide the packet transmission between IP entities and TIPGs, especially protecting the STIPP packets securely exchanged between the IP entities and TIPGs.

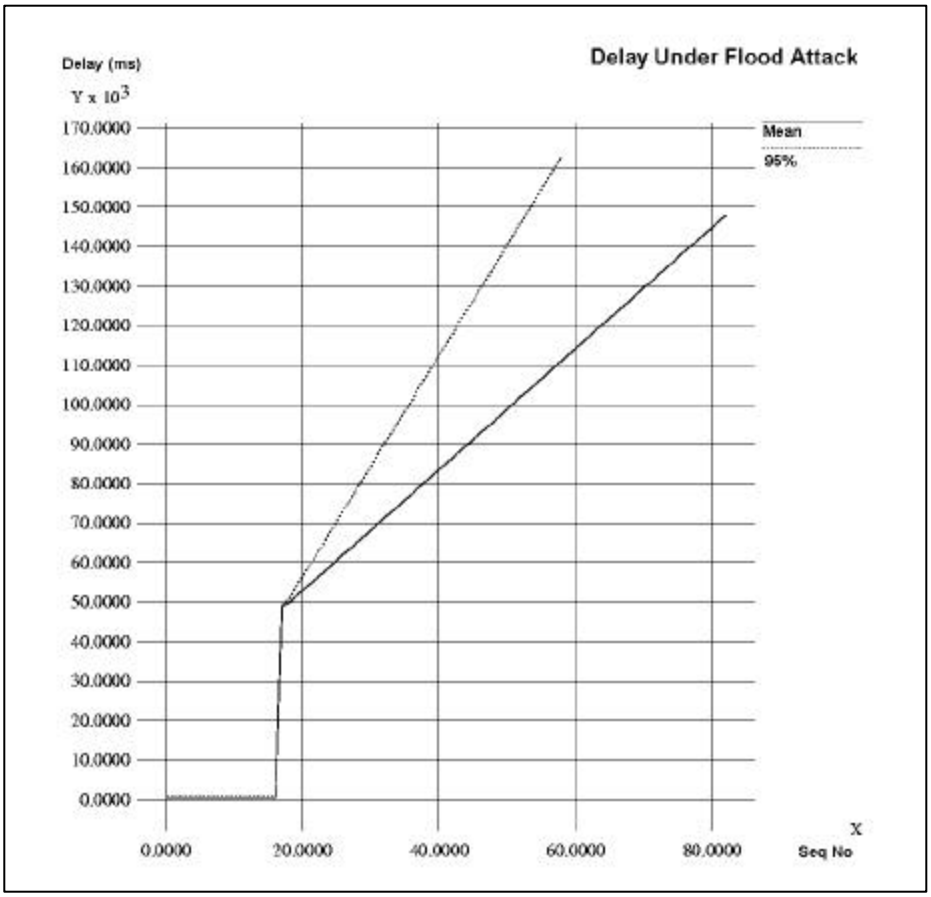


Figure 6.20: The partial results of TCAP delay simulation under one node flood attack through IPSEC tunnel

6.3.2.2 TCAP over IP on IPSEC tunnel (between TIPGs and IP Entities) under flood attack

6.3.2.2.1 Topology and parameters

For protecting the system from the flood attack, we apply the IPSEC tunnel between IP entities and TIPGs (Fig 6.21). All packets that are not from the IPSEC tunnel will be dropped at the TIPG. In order to simulate the IPSEC processing on IP Entity. We increase the basic processing delay at IP entity by 0.8ms. In order to simulate the authentication at the TIPG, all incoming packets are classified by the source address at TIPG receiving buffer. The packets that do not come from the attached IP entity will be dropped.

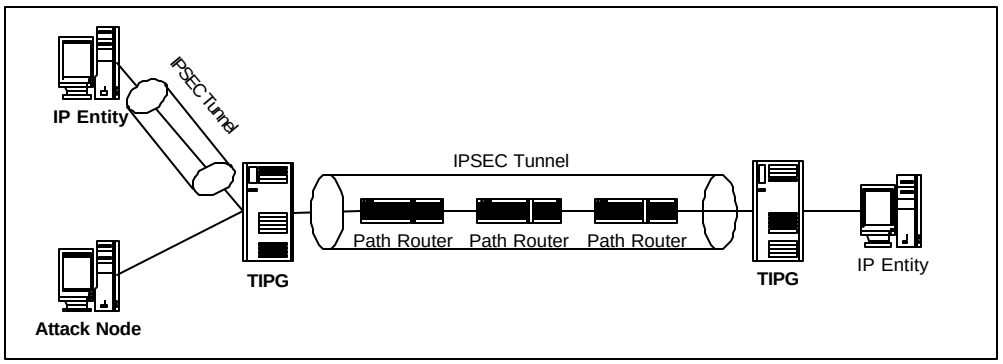


Figure 6.21: IPSEC applies authentication between attached node and TIPG

The simulation procedure is the same as Figure 6.14. In order to see the difference between before and after applying the IPSEC tunnel between IP Entity and TIPG, the simulation does not activate the IPSEC tunnel between the IP Entity and TIPG at the beginning of the simulation. When the flood attack happens for a while, the simulation actives the IPSEC tunnel between IP

Entity and TIPG and applies the policy at the TIPG. TIPG drops all packets that are not from the IPSEC tunnel.

The parameters in the simulation are as follows:

Parameter name	Parameter value
The bandwidth on each link	1Mbps
The delay on each link	5ms
The queue implies on each link	DropTail
The TCAP Query messages package size	1024 bytes
The basic processing delay on IP Entity1	222.8 (50% pro) / 354.8ms (95% pro)
The basic processing delay on IP Entity0	1ms
The workload related processing delay factor on IP Entity1	1
The queuing delay factor on IP Entity	1ms
The queue size on each IP Entity	500
The basic processing delay on TIPG	20.8 (50% pro) / 40.8 (95% pro)
The processing delay for IPSEC on IP Entity and TIPGs	0.8ms
The workload related processing delay factor on TIPG	1
The queuing delay factor on TIPG	1ms
The queue size for each TIPG	1000
The processing delay on each router	2ms
The control drop rate	0

6.3.2.2.2 Results of the experiments

Figure 6.22 shows the TCAP delay in this simulation. It shows the IPSEC can effectively reduce the effect of the flood attack. After we apply the policy at TIPG, the TCAP delay almost drops to the normal value. It depends on two things. The most important factor is that the TIPG can classify the packets from the different resource. The IPSEC uses ESP and AH to provide the authentication. The TIPG use the IPSEC tunnel to make certain that all packets that identify themselves from the IP entity are really the from the IP entity. The TIPG also applies a policing to all incoming packets. If the packet is not from the registered IP entity, the packet

will be dropped. Another important factor is that the TIPG applies a mechanism to apply the classification when the packets arrive at the buffer.

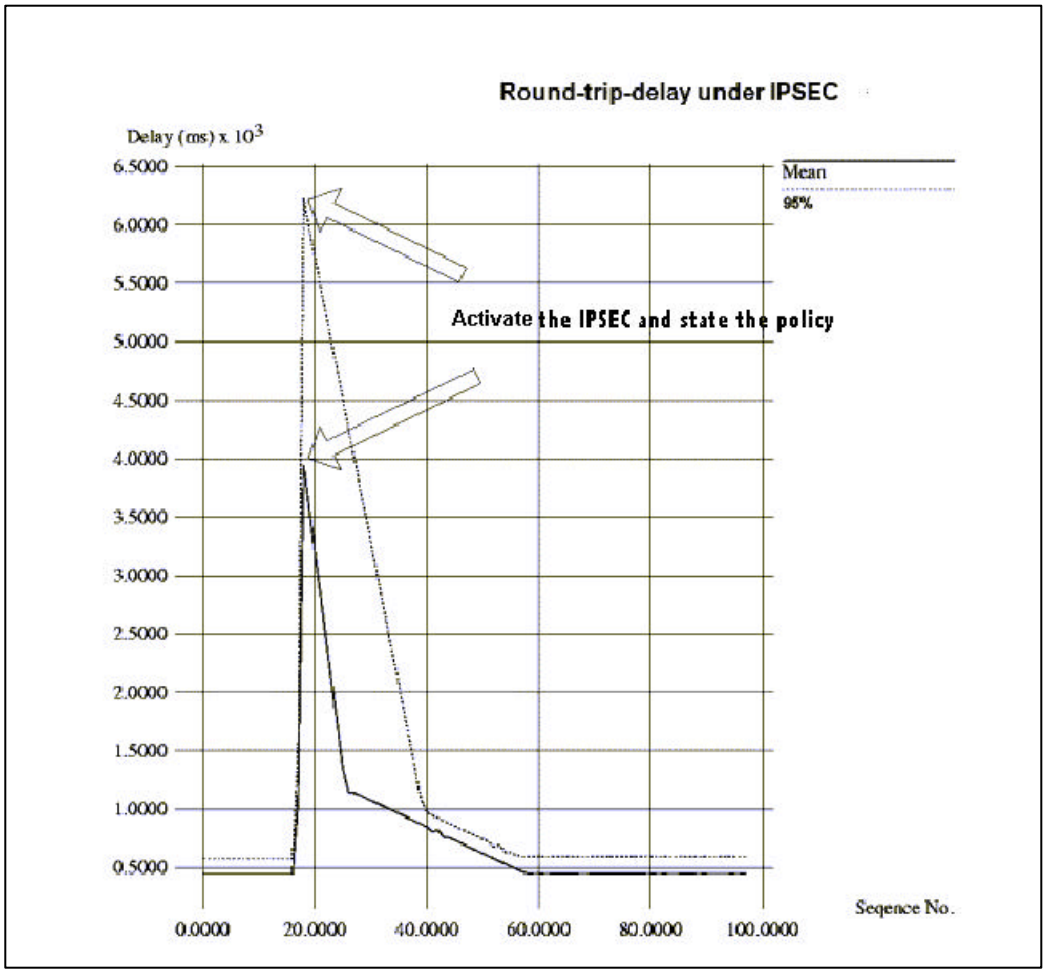


Figure 6.22: The results of TCAP delay simulation under flood attack. The TIPG applies policy to drop the flood attack packets.

6.4 TCAP over IP in Diffserv + IPSEC environment

6.4.1 Query response delay simulation

In order to make the simulated testbed closer to what we expect in the real world, we apply both Diffserv and IPSEC on our normal IP network testbed. We hope the Diffserv can provide a QoS capability network service for TCAP signaling transmission while the IPSEC can provide a security mechanism for protecting the service.

6.4.1.1 Topology and parameters

The topology in the simulation is shown in Figure 6.23. Two IPSEC tunnels are set up. One is between the TCAP source node and TIPG0. The other is between the two TIPGs.

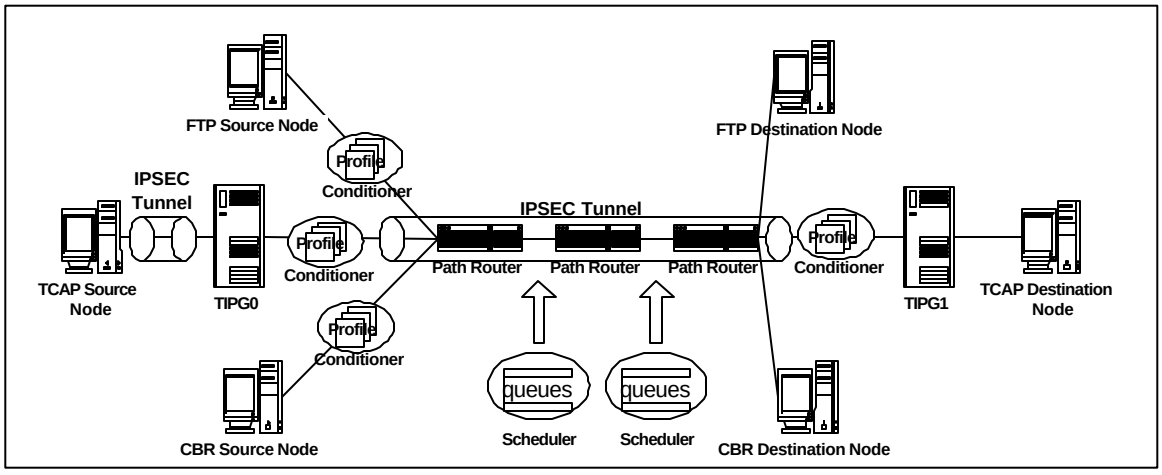


Figure 6.23: The TCAP over IP in Diffserv + IPSEC environment

The parameters in the simulation are as follows:

Parameter name	Parameter value
The bandwidth on each link	1Mbps
The delay on each link	5ms
The queue applies on each normal link	DropTail
The TCAP Query messages package size	1024 bytes
The basic processing delay on TCAP destination node	222ms (50% pro)
The basic processing delay on TCAP source node	1ms
The processing delay for IPSEC on TCAP source node	1ms
The workload related processing delay factor on TCAP nodes	1
The queuing delay factor on TCAP nodes	1ms
The queue size on each TCAP node	1000
The basic processing delay for TIPG	20 ms(50% pro)
The workload related processing delay factor for TIPG	1
The queuing delay factor for TIPG	1ms
The queue size for each TIPG	1000
The processing delay for IPSEC on TCAP source node and TIPGs	0.8ms
The TOS of the normal TCAP message	EF
The TOS of the attack TCAP message	BE
The TOS of the FTP and CBR traffic	BE
The CBR sending rate	1Mbps
The processing delay for each router	2ms(mean)
The control drop rate	0
The scheduling algorithm on Diffserv	WRR
The processing time for conditioner	10ms

6.4.1.2 Results of the simulation

The partial results are shown in Figure 6.24. We can see the delay is about the 532ms which is more expensive that the customer can get from PSTN but still in the range of requirement. The most contributions are the basic processing delay on TCAP destination node, the processing time for tagging on the conditioner and the time to implementing the WRR on each router. How to implement the Diffserv is still under research. As the function of tagging packets and the

scheduling algorithm are being implemented on hardware, we expect to get a better performance of TCAP over IP in the near future.

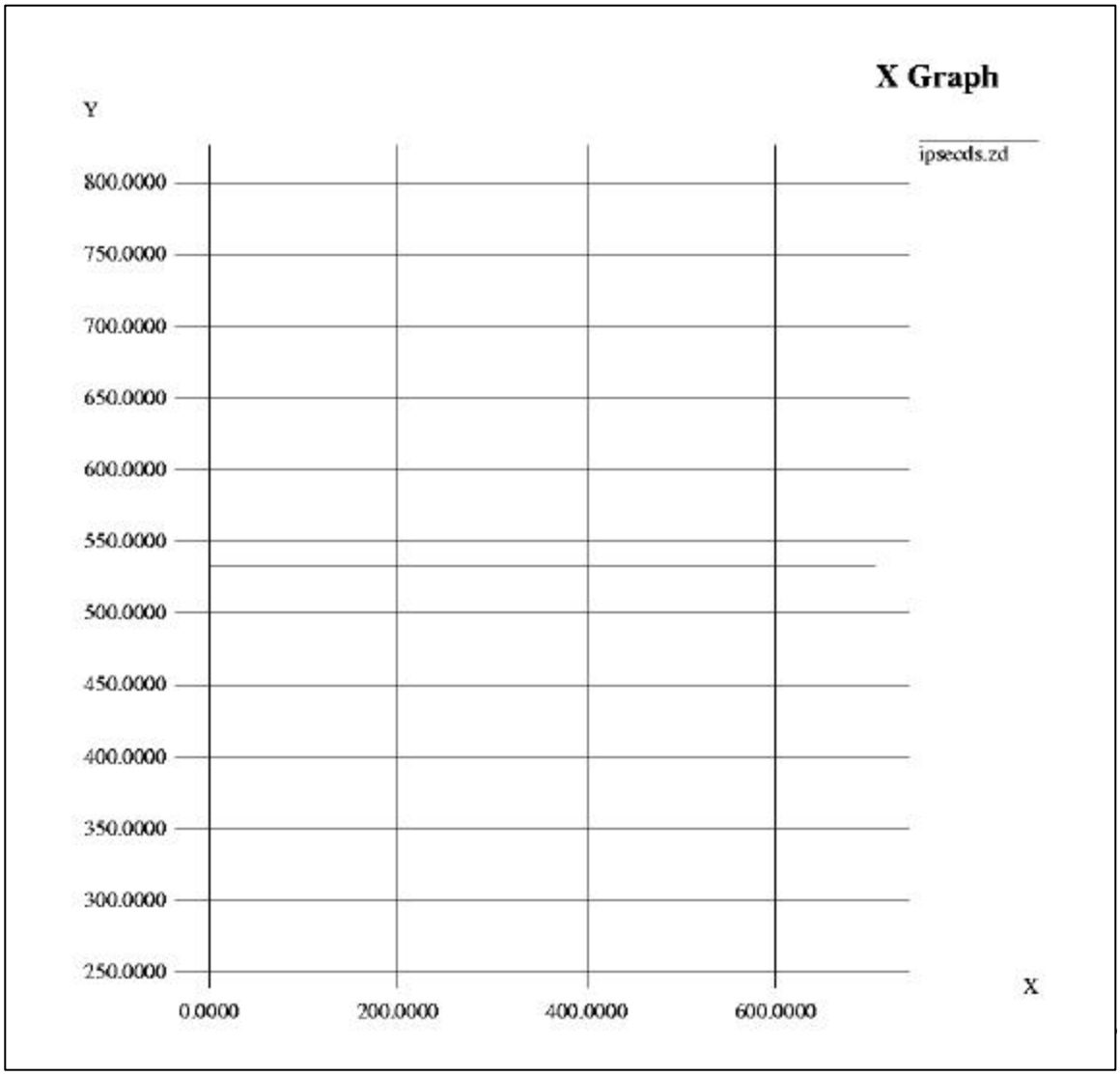


Figure 6.24. The delay of TCAP over IP on Diffserv + IPSEC

6.4.2 Flood attack simulation

6.4.2.1 Topology and parameters

In order to get the effect of the flood attack to TCAP over IP, we introduce three attack nodes into the testbed we built in chapter 6.4.1.1. Two attack nodes are attached to TIPG0 to simulate the flood attack behind the TIPG. Another attack node and TIPG are connected to the router to simulate the man-in-middle flood attack. The topology is shown in figure 6.25.

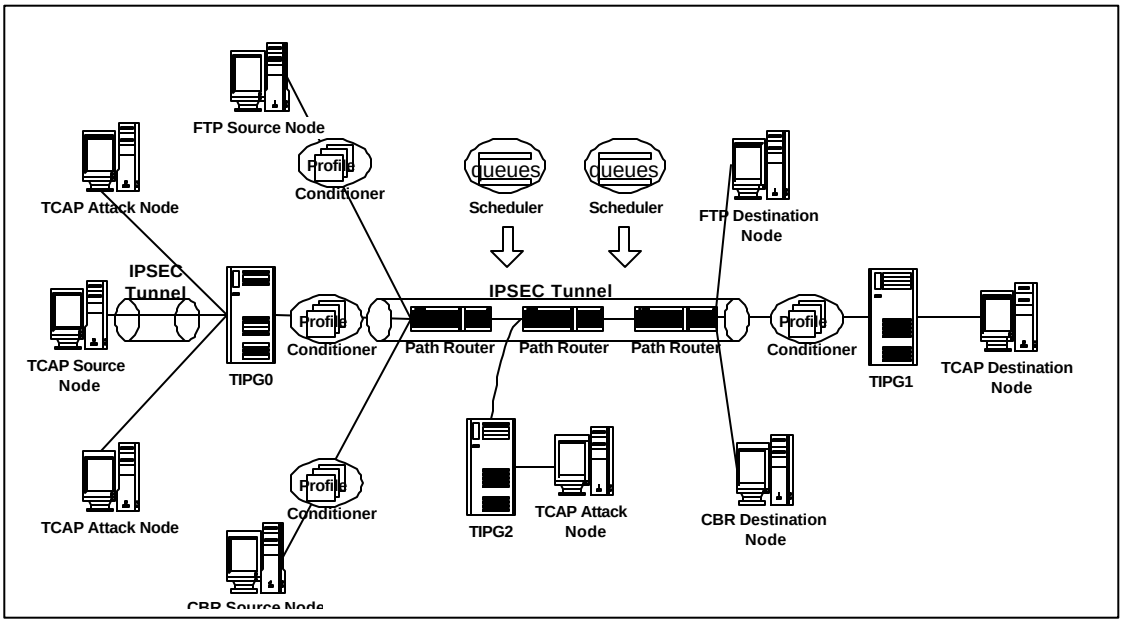


Figure 6.25. The flood attack to TCAP over IP in Diffserv + IPSEC environment

In order to see the difference of the performance between the IPSEC active and IPSEC inactive, we activate the IPSEC 1 second after the flood begins.

The parameters in the simulation are as follows:

Parameter name	Parameter value
The bandwidth on each link	1Mbps
The delay on each link	5ms
The queue attached on each normal link	DropTail
The TCAP Query messages package size	1024 bytes
The basic processing delay for TCAP destination node	222ms (50% pro)
The basic processing delay for TCAP source & attack nodes	1ms
The workload related processing delay factor for TCAP nodes	1
The queuing delay factor for TCAP nodes	1ms
The flood interval of the TCAP attack nodes	100ms
The TOS of the normal TCAP message	EF
The TOS of the attack TCAP message	BE
The TOS of the FTP and CBR traffic	BE
The CBR sending rate	1Mbps
The queue size for each TCAP attack nodes	1000
The basic processing delay for TIPG	20ms(50% pro)
The workload related processing delay factor for TIPG	1
The queuing delay factor for TIPG	1ms
The queue size for each TIPG	1000
The processing delay for each router	12ms(mean)
The scheduling algorithm on Diffserv	WRR
The control drop rate	0
The processing time for conditioner	10ms

6.4.2.2 Results of the simulation

The partial results of the experiments are shown in Figure 2.27. The first large increase of delay is due to the flood attack from the two attack nodes from the TIPG. When the flood attack began, the IPSEC tunnel between the TCAP source node and TIPG are inactive. All flood

attack traffic gets to the TCAP destination node. The queuing delay increases sharply. The TCAP service is denied. Then, 1 second later, the IPSEC tunnel between the TCAP source node and TIPG0 is activate. All flood attack traffic is dropped at the TIPG0 due to failure of authentication. The delay of TCAP decreases to the normal value. The second large increase of delay is due to the flood attack from the attack node in the middle. The before the IPSEC tunnel is activated, all flood attacks get to the TCAP destination node. The delay of the TCAP over IP jumps to almost 1.8 seconds. Once the IPSEC tunnel between the TIPGs is active, all flood traffic is dropped at TIPG1 due to failure of authentication. The delay of TCAP over IP decreases to the normal value. Because the flood attack packets are marked as BE, we can see that although the flood attack gets to router 1 and router 2 it does not bring too much additional delay to the whole TCAP delay. There is a little increase for the delay of TCAP over IP. Most of the increase is from implementing the WRR algorithm on each router.

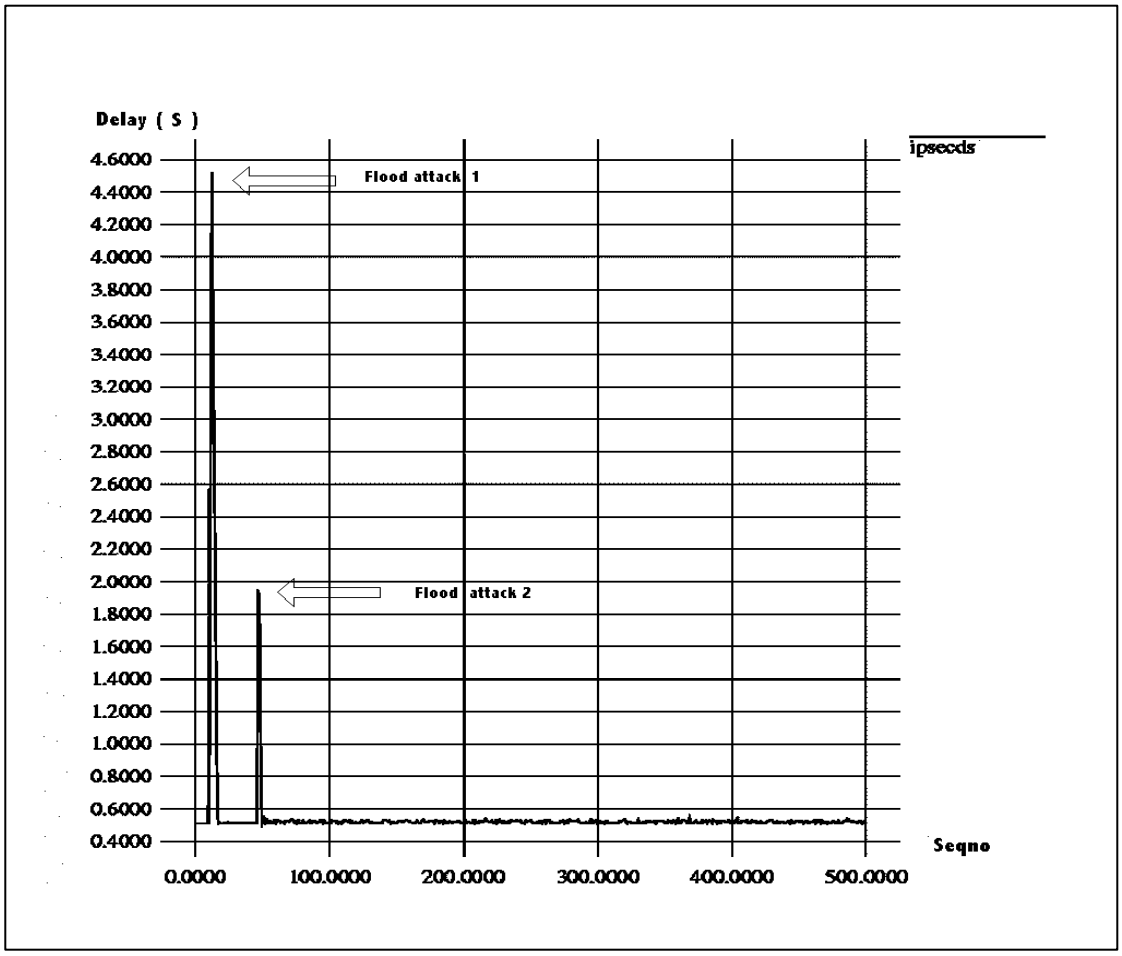


Figure 2.27. The delay of TCAP over IP under flood attack in Diffserv + IPSEC environment

Chapter 7 Conclusions and future work

From the results of the experiment we did on the simulation testbeds, we can draw some conclusion.

1. The delay of TCAP over IP is affordable for the internetworking of SS7 and IP networks.

Based on our simulation, the delay of implementing TCAP over IP is larger than what the current PSTN subscribers get from the SS7. However, it is acceptable and can satisfy the requirements of the VoIP service. The current IP network provides a pure best-effort service that is not suitable for real time signaling transmission. The Diffserv can provide a QoS mechanism on the IP layer to provide a bounded delay for the TCAP signaling transmission. Based on the results of the delay experiments, we can see the most significant tradeoff of Diffserv is that it brings 10ms additional delay for tagging packets and 10ms additional delay for processing the WRR on each router. As those functions are implemented on hardware, we expect the delay of TCAP over IP can be reduced further. In order to compare the performance of TCAP over IP in different environments, we list the results of our experiments in table 7.1.

Table 7.1: The performance of TCAP over IP in different environment

	Mean delay of TCAP of IP (ms)	Invulnerability to flood attack	Mean delay of TCAP of IP under flood attack (ms)	Provide QoS
Normal IP network	442.8	No	Keep increasing	No
Diffserv	522.0	No	Keep increasing	Yes, but vulnerable to the attack that can mark itself as EF.

IPSEC tunnel	444.0	It can protect the flood attack outside the IPSEC tunnel, but vulnerable to the flood attack that can enter the tunnel.	452.0	No
Diffserv + IPSEC tunnel	532.6	It can protect the flood attack outside the IPSEC tunnel, but vulnerable to the flood attack that can enter the tunnel.	540.4	Yes

2. The TCAP over IP brings security risks to SS7 and IP Telephony.

During implementing the signaling transmission between the SS7 and IP domain, the attack can be launched from either side. For TCAP over IP, the attack can drop, delay, and temper the TCAP message or flood TCAP message to SCP and TIPG. From the results of our experiment, we can see the flood attack can easily deny the service of the IP Entity that does not apply the security mechanism. Flood attackers can be categorized into two different classes: insiders and outsiders. Outsider attacks can usually be prevented effectively by access control mechanism. However, the insider attacks are much more difficult to deal with as the insider flood attacks can typically pass through some authentication and access certain trusted components.

3. IPSEC can effectively reduce the risk of attack and the overhead of IPSEC is affordable.

IPSEC can provide the encryption and authentication at the IP layer for TCAP over IP. The processing delay of implementing IPSEC is comparably cheap and affordable to TCAP over IP. Moreover, as the IPSEC function is implemented by hardware, we expect those delays can be reduced further. However, IPSEC does not provide a complete solution. There are two problems. One is that IPSEC needs to combine with the Quality of Service mechanism to satisfy the requirement of signaling transmission. We expect the IPSEC function can be moved

to a high performance router or gateway that is located in front of the TIPG and supports the QoS. However, the implementation of the quality of service in IPSEC is still under research. The other problem can be the policy of the game. Right now, it is hard to set up a trusted relationship between different domains, especially for different counties. IPSEC tunneling can be easily set up for a VPN. However, TCAP over IP requires the capability of interoperability of different networks. How to setup the security association between different countries and different domains is still under discussion.

All in all, TCAP over IP is very critical for the internetworking between the SS7 and IP worlds. Its performance and safety will affect the entire quality of VoIP service. This paper provides a simulation model for the research in this field. Future work can be done in following the fields.

- Enhance the simulation tool.

We expect the simulation tools can be enhanced in following fields.

Routing Protocol

The PathRouter currently supports static routing. The route table has to be manually set up during the network initialization. We hope that in next step the path router can support dynamic routing protocols such as OSPF and RIP.

RSVP

The RSVP is a receiver-oriented reservation protocol. It is designed to provide a general means for an application to communicate its QoS requirements to an Integrated Services Internet infrastructure. It is neither a data transport protocol nor a routing protocol, but a protocol that signal QoS requests on behalf of a data flow. In the future work, we hope RSVP will be supported and the experiments of the delay of TCAP over IP can be run on RSVP.

- Research the attack from SS7.

Most attacks in this paper are from the IP domain. We assume the SS7 side is secure. If some hackers use compromised SCP to attack the resource on IP side, it is a nightmare for the IP Telephony service provider. How to manage key and authentication information between the two networks is an interesting topic.

- Detect the drop attack.

Drop attack is another common attack in IP domain. In our simulation testbed, the path router can receive command from TCL interpreter to randomly drop the TCAP messages that it received. We hope that in the future a network monitor function can be added to TIPG and path routers. Therefore, we can set up a mechanism to detect the drop attack by periodically exchanging the network status messages.

- New application based on TCAP over IP

In the future work, new applications that are based on TCAP over IP can be tested on our simulation testbed.

Chapter 8 References

[1] Monlling Liao, Liem Le, Rambabu Tummula,Emad Qaddoura, Don Wurch. Simple SS7-TCAP/IP Protocol. Internet Draft, IETF, September 1999.

[2] ITU-T Recommendation Q.709, Specifications of Signaling System No.7--Hypothetical Signaling Reference Connection, March 1993.

[3] Telcordia Technologies Generic Requirements GR-1364-CORE, Issue 1,LSSGR: Switch Processing Time Generic Requirements Section 5.6, June 1995.

[4] ITU-T Recommendation Q.706, Specifications of Signaling System No.7--Message Transfer Part Signaling Performance, March 1993.

[5] KESHAV, S. REAL: A network simulator. Tech. Rep.88/472, University of California, Berkeley, Dec.1988

[6] DUPUY, A., SCHWARTZ. J., YEMINI, Y., AND BACON, D. NEST: A network simulation and prototyping testbed. Communication of the ACM 33,10 (Oct, 1990), 64-74.

[7] ITU-T Recommendation Q.709, Specifications of Signaling System No.7--Hypothetical Signaling Reference Connection, March 1993.

[8] ITU-T Recommendation Q.709, Specifications of Signaling System No.7--Hypothetical Signaling Reference Connection, March 1993.

[9] Load Coene and Siemens Atea. SS7 over internet applicablity statement. Internet Draft, IETF, May 2000.

[10] G. Sidebottom, J. Yoakum. Open Architecture for IP/SS7 Interworking. Internet Draft, IETF, January 2000.

[11] Huai-An P. Lin, Kun-Min Yang, Taruni Seth, Christian Huitema. VoIP Signaling Performance Requirements and Expectations. Internet Draft, IETF, March 2000.

[12] David Sanchez, Miguel A. Garcia. A Simple SCCP Tunneling Protocol (SSTP). Internet Draft, IETF, July 1999.

[13] Travis Russell. Signaling System #7. McGraw-Hill, pp267-320.

[14] ITU-T Recommendation Q.774, Specifications of Signaling System No.7-- Transaction Capabilities Procedures, March 1993.

[15] Lloyd Armstrong, Jr. Virtual InterNetwork Testbed (VINT): methods and system. Information Science Institute University of Southern California. May 1996.

[16] S. Kent and R. Atkinson IP Authentication Header. RFC2402, IETF, November 1998.

[17] P. Metzger and W. Simpson. IP Authentication using Keyed MD5. RFC1828, IETF, August 1995.

[18] P. Metzger and W. Simpson. IP Authentication using Keyed SHA. RFC2402, IETF, September 1995.

[19] S. Kent and R. Atkinson. IP Encapsulating Security Payload (ESP). RFC2406, IETF, November 1998.