Secure E-Commerce Transactions for Multicast Services

Anil Kumar Venkataiahgari

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Computer Science at
Concordia University
Montreal, Quebec, Canada

December 2005

Abstract

Secure E-Commerce Transactions for Multicast Services

Anil Kumar Venkataiahgari

With the advent of an increasing number of multicast services such as audio/video streaming, news broadcast and software distribution, accounting for the resources consumed by subscribers becomes essential. However, multicast environments are inherently more susceptible to attacks because there is as yet no proper foundation for securing multicast networks. However, any e-commerce operational environment requires support for the security properties such as authentication, authorization, data confidentiality, and non-repudiation. As secured multicast services gain popularity in the Internet, at one point there would be a significant response from the service providers to support e-commerce through multicast networks. In this thesis, a Multicast Security Protection Profile (MSPP) has been presented to identify and secure the operating environment of multicast sessions. Although e-commerce protocols such as SSL, TLS, and SET protocols offer some forms of security in e-transactions, they cannot be directly extended to provide security for e-commerce sessions in the multicast (point-to-multipoint) environment. Therefore, we have designed an architectural framework, called the Secure E-Commerce Transactions for Multicast Services (SETMS), to secure e-commerce sessions for multicast environments. The SETMS framework provides authentication of host through HIP protocol, authorization of subscriber and his e-payments through a variant of the 2KP protocol, a procedure to account for the subscriber's resource consumption, and supports non-repudiation of principals parties through PKI. The SETMS framework has been evaluated using the Common Criteria based MSPP security specification and formally validated using the AVISPA tool.

Acknowledgements

I would like to extend my sincere gratitude and appreciation to my research supervisors Dr. J. W. Atwood and Dr. Mourad Debbabi for providing me an opportunity to work both with multicast and e-commerce security protocols. I cannot stop appreciating the patience, encouragement and support at all levels that they have extended towards me when I got stuck with my research work.

Dr. Atwood’s expertise in multicast security protocols and Dr. Debbabi’s expertise in e-commerce security protocols has lead me in figuring out the converging aspects of multicast and e-commerce environments, which ultimately lead me to carry out this research work. Without their mentorship and moral support, my research work would have gone in vain.

I would like to thank my friends Mursalin, Anirban, Ritesh and Salekul for showing their interest in my research and coming up with some generous ideas that provided me some valuable hints to carry out my research work. Special thanks to my friends Amar, Bhushan, Sujoy, Ali, Hamdi, Anand and all other friends, who have encouraged me in carrying out this work. Without their kind heartedness, it would have probably taken me a few more months to reach the finish line.

Finally, I would like to express my deepest appreciation to my parents and my brothers for their constant love, encouragement and assistance.

Table of Contents

List of Figures

List of Tables

List of Acronyms

AH	Authentication Header
AAA	Authentication, Authorization and Accounting
AVISPA	Automated Validation of Internet Security Protocols and Applications
B2B	Business-to-Business
B2C	Business-to-Customer
CC	Common Criteria
CC#	Credit Card Number
CA	Certification Authority
CHAP	Challenge-Response Authentication Protocol
CL-AtSe	Constraint-Logic-based Attack Searcher
CP	Certificate Policy
CPS	Certificate Practice Statement
CRL	Control Revocation List
DTLS	Datagram Transport Layer Security
DNS	Domain Name Service
DoS	Denial of Service Attacks
DSA	Digital Signature Algorithm
ELSD	Equal Link Split Downstream
EMSS	Efficient Multi-chained Stream Signature
ESP	Encapsulated Security Payload
FEC	Forward Error Correction
HI	Host Identifier

HIP	Host Identity Protocol
HIT	Host Identity Tag
HLPSL	High Level Protocol Specification Language
HMAC	Hashed Message Authentication Code
I^3	Internet Indirection Infrastructure
IANA	Internet Assigned Numbers Authority
ICT	Information and Communication Technology
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
iKP	i-Key-Protocol, i = 1, 2, and 3
IP	Internet Protocol
IPSec	IP Security Protocol
LGMP	Local Group based Multicast Protocol
LSI	Local Scope Identifier
MCP	Merchant's Content Provider
MD	Message Digest
MLD	Multicast Listener Discovery
MSC	Message Sequence Chart
NAS	Network Access Server
OCSP	Online Certificate Status Protocol
OFMC	On-the-fly Model Checker
PAP	Password Authentication Protocol
PG	Payment Gateway
PGM	Pragmatic General Multicast
PKI	Public Key Infrastructure
PP	Protection Profile

PS	Policy Server
RA	Registration Authority
RADIUS	Remote Access Dial-In User Service
RMP	Reliable Multicast Protocol
RMTP	Reliable Multicast Transport Protocol
RTFM	Real-time Traffic Flow Measurement
RTP	Real-time Transport Protocol
SA	Security Association
SATMC	SAT-based Model Checker
SET	Secure Electronic Transactions
SHA	Secure Hash Algorithm
SPI	Security Parameter Index
SRM	Scalable Reliable Multicast Protocol
SRTP	Secure Real-time Transport Protocol
SSL	Secure Socket Layer
TA4SP	Tree Automata-based Protocol Analyser
TACACS	Terminal Access Controller Access Control Systems
TCP	Transmission Control Protocol
TESLA	Timed Efficient Stream Loss-Tolerant Authentication
TLA	Temporal Logic of Actions
TLS	Transport Layer Security
TOE	Target of Evaluation
TTL	Time to Live
TTP	Trusted Third Party
UDP	User Datagram Protocol
XTP	Xpress Transport Protocol

Chapter 1

Introduction

E-commerce is basically described as a transaction processing via various devices and communication technologies. In general, in e-commerce transactions an unknown subscriber and an unreliable merchant participate in business for mutual benefits. Usually, the merchant participates in e-commerce for monetary benefits while the subscriber gets involved if the merchant offers cheaper service rates comparatively to his competitor’s norms. The primary requirement for principals to communicate is via the Internet or Intranet where a merchant publishes his services on a web server that could be accessible to its subscribers via the World Wide Web (WWW). E-commerce systems can be broadly categorized into Business-to-Business (B2B) and Business-to-Customer (B2C) commerce. In B2B commerce, business parties are at minimum two merchants (content providers) while in B2C commerce, principals involved are at minimum a merchant and a subscriber where the merchant offers services to its subscribers and customers subscribe to buy service from the merchant. For example, a simple to complex exchange of goods between two merchants is B2B commerce, while a customer subscription to a service offered by a merchant is B2C commerce.

In this thesis we focus on laying a strong foundation for the applicability of B2C commerce for multicast environments, which can be later extended to B2B commerce. At first, let us understand the basic concepts involved in multicasting. IP Multicast is a different way of routing datagrams invented by Steve Deering (PhD. 1991) and specified in RFC 1112 [1]. An IP multicast group address uses a Class D address between 224.0.0.0 and 239.255.255.255. Addresses between 224.0.0.0 and 224.0.0.25 have been designated by Internet Assigned Numbers Authority (IANA) for multicast control messages but Standards are still evolving. Multicast requires a source to send only one packet, which is then replicated on its way to all recipients instead of sending ‘n’ data packets to ‘n’ recipients. Thus, multicast networking saves source bandwidth, reduces routing processing requirements and latency observed by receiver, and yields lower costs to network providers when used to deliver services (content) to subscribers having common service delivery preferences. IP Multicast uses UDP datagrams at the transport layer, which offers best effort data delivery service to its hosts.

Some of the popular multicast routing protocols that are used by multicast routers for forwarding multicast data packets are Distance Vector Multicast Routing Protocol (DVMRP), Protocol Independent Multicast (PIM), Multicast Open Shortest Path First (MOSPF), Routing Information Protocol (RIP2), and Core Based Tree (CBT). These routers exchange routing information as well as multicast group membership data. More information on IP multicast routers and their protocols supporting multicast host groups can be found in [36]. The host group is identified by a multicast group address, which is shared among all hosts who have requested multicast service from the nearest multicast router by issuing a JOIN request to it. The host can issue JOIN and LEAVE signals for expressing its interest to subscribe to or unsubscribe from a multicast group. These control signals are supported by the Internet Group Management Protocol (IGMP) or Multicast Listener Discovery (MLD), which runs over the IP layer. The source data packet hop count is limited by using the Time-to-live (TTL) field in the multicast IP header.

The TTL prevents packets from unnecessary transiting to regions other than that of the designated subnet(s) containing the multicast group members. The TTL of each multicast packet is decremented by 1 whenever a router forwards the packet to its successor. The default IP TTL field is set to 1, which implies that all members, which are 1 hop-count distant from the multicast host group, will receive multicast packets. If the TTL is greater than 1, multicast routers take the responsibility of forwarding packets to other internetwork routers that have members of the destination host group. When the packet’s TTL reaches 0, it is considered as an expired packet and the packet will be dropped by any router holding that multicast packet without issuing an ACK message to the packet source. Since hosts can send data at any time to any group, routers must be prepared to receive packets on all link-layer group addresses and know when to forward or drop packets. The router is responsible for keeping track of interfaces leading to subscribers.

1.1 Motivation, objectives and contribution

Security is a significant concern when someone discusses the applicability of e-commerce for multicast services. It is well known that a multicast network is inherently more susceptible to attacks because it presents more opportunities for interception of its network traffic as the data are now open to a much wider population. In IP multicast networks as specified in RFC 1112, there is no mechanism for restricting non-group members from accessing the group communication. This basic drawback of multicast will result in improper and unauthorized access to the service resources. In addition, since multicast addresses are generally well advertised and made public, it becomes easier for an attacker to launch an attack. If the attack is successful, a potentially large number of principals is affected. In contrast, e-commerce protocols must meet specific requirements such as authentication of data and principal parties, data confidentiality, money atomicity, etc.

Liability will also become a significant issue because the subscriber has to share his sensitive credentials such as credit card information with unknown principals while merchants have to make sure that they are dealing with an authenticated customer. Figure 1 depicts a generic scenario of e-commerce for multicast environment. It shows the subscribers wishing to participate in e-commerce for the multicast service offered by a merchant.

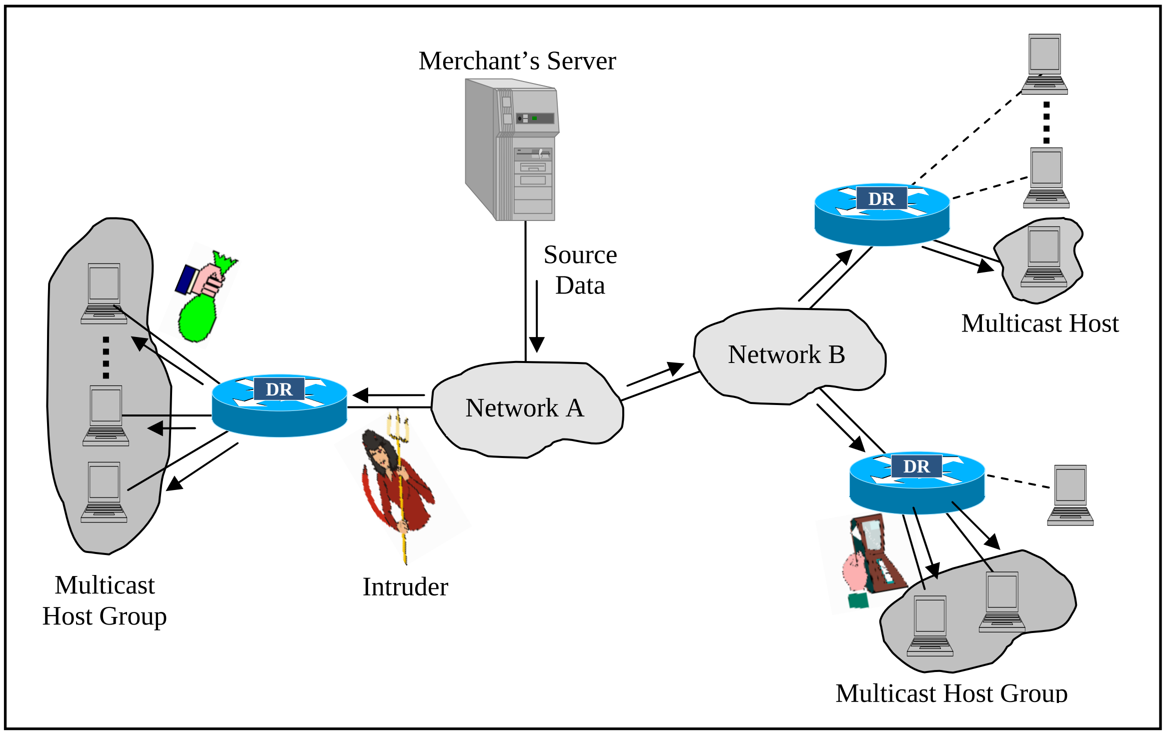


Figure 1: Generic scenario of e-commerce for multicast environment

If the multicast service is offered on the basis of pay-per-view, accounting for the resource usage becomes very important as subscribers are to be charged in a fair manner for their consumed resources. Principal parties would have many questions for which they seek an answer. For example, the concerned principal could have the following questions:

- Who will conduct the authentication process?
- How to verify the identity of hosts and subscribers, and legitimacy of merchants?

- What kinds of information must be verified?
- What authority and access privileges should be enforced?
- What kinds of audits are required, if an error or session compromise occurs?
- What is the liability and risk involved in the transaction processes?
- What are the technical and non-technical issues, and performance tradeoffs to consider when applying security and key management techniques in support of multicast environments?

For example, an e-payment system should be reliable enough to carry out e-commerce transactions between a merchant and its subscriber. Thus, securing e-commerce transactions for multicast environments implies that there should be mechanisms for authenticating an end-host and its subscriber, authorization of subscriber and his payments, encryption of data for confidentiality protection, secure key distribution to the multicast groups and its hosts, and accounting for e-commerce sessions. If vulnerabilities in any of these protocol components persist, the security of the applied infrastructure could be compromised.

The objectives of our research work can be put forth as follows:

- What support does the multicast literature offer to secure multicast sessions and e-commerce transactions?
- What are the prominent approaches in e-payment platforms and protocols that could be applicable to e-payments for multicast environments?
- How to capture the security requirements for multicast sessions in e-commerce operational environment?
- How to elaborate an architecture for e-payments applicable to multicast services that is scalable, flexible and secure?
- How to assess the architecture with respect to the aforementioned security requirements for e-commerce transactions for multicast environments?

To address these aforesaid objectives, this thesis makes the following contributions:

- Survey of multicast literature that could be applicable to secure e-commerce transactions for multicast environments
- Comparative study of the main approaches in e-payment platforms and protocols to understand their applicability for multicast environments
- Propose a Protection Profile that captures the multicast security requirements to secure multicast sessions in the e-commerce operational environment
- Propose a scalable, flexible and secure architectural framework that offers authentication of host and its user, authorization of subscriber and his e-payments, accounting for each e-commerce transaction, and non-repudiation of principal parties in e-commerce
- Common Criteria (CC) evaluation of the aforementioned architecture with respect to Protection Profile (PP) and formal validation of the protocol suite using the AVISPA tool

1.2 Organization of the Thesis

The thesis is made up of eight chapters. The structure of each chapter is briefly described as follows.

Chapter 1 presents details on the fundamentals of IP multicast networks and e-commerce. This is followed by describing what has motivated us to carry out this research work. It then follows up by stating the objectives and contribution of this thesis work.

Chapter 2 details the support of the multicast literature to secure multicast sessions and e-commerce transactions. It presents identification of multicast applications based on service type, and classification of multicast protocols based on their scalability to the audience. It is followed by reviewing the state of art in identification of end-hosts and data source, detection of lost packets, and description of popular accounting models.

Chapter 3 describes popular electronic payment platforms and protocols. It is followed with a comparative study of these systems to evaluate their support for multicast environments.

Chapter 4 describes the key concepts involved in the Common Criteria (CC) that help the reader in understanding the Protection Profile (PP) concepts. The reader who is already well aware of the CC concepts may skip this chapter. This chapter lays the background concepts that will help the reader in understanding chapter 5.

Chapter 5 details the Multicast Security Protection Profile (MSPP) for e-commerce as the Target of Evaluation (TOE). It presents security specifications that are required to secure multicast sessions in an e-commerce environment. Section 5.1 provides the identification and overview of the MSPP. Section 5.2 provides the description of the TOE. Section 5.3 provides a discussion of the expected environment for the TOE. This section also details the threats to security, sources of vulnerability and sources of attack that could target the operating environment of the TOE. It concludes by stating the security policies that must be enforced by the TOE to secure the operating environment. Section 5.4 identifies the risks to the TOE that have been derived from statements of the security environment defined in section 5.3. Section 5.5 concludes the chapter by defining the security objectives for the TOE.

Chapter 6 presents the proposed Secure E-commerce Transactions for Multicast Services (SETMS) architectural design framework. It describes the assumptions and scope, system components and operational infrastructure of SETMS framework that are required to secure e-commerce transactions for multicast environments.

Chapter 7 presents an evaluation and formal validation of the SETMS framework. Section 7.1 describes the Common Criteria evaluation of SETMS framework with respect to the MSPP documentation (discussed in chapter 5) and through the support of the multicast and e-commerce literature. Section 7.2 presents a formal validation of the SETMS protocol suite using the AVISPA tool.

Chapter 8 discusses the conclusion of this thesis work, stating the advantages and limitations of the SETMS framework. It is followed with directions for future work.

Chapter 2

Multicast support for e-commerce

In this chapter, we detail the technological support for multicast networks and the protocols used in a multicast e-commerce environment. Firstly, we classify multicast protocols that could be possibly deployed in e-commerce. In the next section, we discuss the state of the art in areas of identification of multicast end-host and data source. This is followed by a discussion of the literature on multicast support for detecting lost packets of the source data. Mechanisms to detect lost packets are essential when multicast services are deployed in an e-commerce environment. In the last section, we detail various multicast resource accounting models that are available in the multicast literature.

2.1 Multicast protocols classification

As no one has attempted to classify the multicast services that could be deployed in e-commerce, we have made an attempt to identify most of the multicast applications that may suit an e-commerce environment. The nature of the operating conditions for the multicast applications could be such as large number of participants with both known and unknown, heterogeneous participants with varying requirements, subscribers with varying financial interests, high value or high volume transaction-based commerce, and diversity of multicast applications. Broadly, multicast applications can be classified into 1-to-N and M-to-N applications. These applications may consist of a small, medium to large group with static or dynamic multicast groups supporting data oriented services and information dissemination services. These applications could be operational in e-commerce based on pre-paid, on-demand, scheduled, and post-paid multicast services.

1-to-N applications require sending data originated from a single source to multiple subscribers. 1-to-N applications include stock quotes updates, distance learning, advertising, airline flight information updates, news feeds/updates, betting services, online auctions, and audio/video broadcast services. M-to-N applications require sending data originated from multiple sources to multiple subscribers. Usually, most of the M-to-N applications operate in an M-to-M mode [6], which implies all senders are also subscribers. M-to-N applications include online chat groups, multi-player gaming, distributed interactive simulations, multimedia conferencing such as whiteboard, audio/video streaming, resource synchronization applications such as database replication and software distribution. Each of these multicast applications within each type varies in its requirement of degree of robustness and reliability. For example, M-to-N applications cannot absorb high latencies and require the host machine to buffer the source data. Applications like streaming database replication and software distribution services require strict reliability and ordering of data packets. Whiteboard applications require collaboration among all nodes.

J. W. Atwood [6] presented a classification of the multicast protocols based on the end-user requirements, and the architectures, mechanisms, communication patterns, and policies that are used to satisfy these requirements. His survey reveals that XTP, LGMP, RMTP and LPC are 1-to-N multicast protocols while RMP, SRM and PGM are M-to-N multicast protocols. The scalability of each of these protocols is described as follows. XTP protocol could be deployed for small to medium sized multicast groups. LGMP and RMTP are designed for deploying to large groups (several hundreds). LPC is designed to scale to very large groups (tens of thousands of group members). RMP, SRM and PGM are designed for deploying in medium to large sized multicast groups. RMTP-II has recently been developed; it supports multiple senders into a group, but basically remains a 1-to-N protocol.

Designing a reliable multicast protocol that fits for each and every application is considered impossible. Issues of protocol reliability, simplicity, stability, interoperability, resource consumption, and service cost usually dictate multicast applications applicability to an e-commerce environment.

2.2 End host and data source identity

Several researchers have proposed several ways of authenticating the multicast host and data source that are applicable to different operating environments. In this section, we present a survey of the research papers that have some mechanisms to authenticate end hosts and data sources.

The Host Identity Protocol (HIP) has been developed jointly by IETF and IRTF working groups, but HIP Standards are still evolving. HIP specifications are discussed in [2]. The HIP architecture introduces a new namespace namely, Host Identity namespace in between the network layer and the transport layer. This namespace introduces new Host identifiers (HIs), which are cryptographic public keys. HIP decouples transport UDP association with network IP addresses and couples it with HI. This decoupling followed by coupling splits the dual role of IP addresses, facilitating the separation of host identity role of HI from host locating role of IP addresses. HI being a public key is considered to be long and impractical to be used as a host identity in an operational environment. A hash of HI, namely Host Identity Tag (HIT) of 128-bit length is generated for representing the host identity in IPv6 sized address structures. Similarly, a 32-bit length Local Scope Identifier (LSI) is generated for representing the host identity in IPv4 sized address structures.

In [4], R. Moskowitz et al. have discussed the HIP base message exchanges, which use a Diffie-Hellman authentication procedure for establishing a HIP Security Association (SA) between communicating hosts. More on the HIP state machine, HIP packet formats, packet processing, and HIP policies can be found in [4]. While the HIP base message exchange sets up a HIP SA between two hosts, it does not provide any mechanism for protecting data communication between the hosts. IPSec ESP (Encapsulated Security Payload) [9] is used to securely carry the data packets of HIP between hosts.

In [5], P. Nikander and J. Laganier proposed Domain Name System (DNS) extensions for HIP. They have defined two new resource records (RRs) in DNS that could be used with HIP. These RRs facilitate storing of domain name and HIs in the DNS. One of the RR is called HIPHI, which consists of an HI type, a hash algorithm type, an HIT, and a public key algorithm. The other RR supports host mobility and therefore it is not discussed here.

In [8], I. Stoica et al. proposed Internet Indirection Infrastructure (I^3). It is similar to HIP model where the routing information is separated from the host identity. In I^3, each multicast source data packet is identified using a source identifier. Any host that wishes to receive this traffic must register its IP address at a rendezvous server. The rendezvous server takes the responsibility of forwarding the source data packets to the registered hosts.

In [15], A. Perrig et al. proposed two schemes, namely TESLA and EMSS, which provide solutions to the problems associated with source authentication of multicast data streams. TESLA

(Timed Efficient Stream Loss-tolerant Authentication) performs delayed authentication of the host and loose initial time synchronization mechanisms to achieve objectives such as source authentication, high scalability, strong loss robustness, and minimal overhead as described in [15, 16]. It relies on symmetric cryptography for key exchange to lower the computation costs and reduce the transmission overhead of packets. A MAC, which is based on a secret key, is attached to each message. The basic principle used is that each message received by the receiver can be used to authenticate the previous message, i.e., the key for P_i is revealed in P_{i+1}, indirectly delaying the data authentication process but directly speeding the computational process. However, the delayed authentication constrains that the receiver should buffer the packets until authentication is completed. TESLA is also being considered by the IETF for securing multicast sessions. TESLA is even used by Secure RTP (SRTP) for authenticating the source data stream as described in [14].

The EMSS (Efficient Multi-chained Stream Signature) protocol suite addresses the problems in non-repudiating principals as described in [15]. It supports non-repudiation by making use of a public key signature algorithm to compute signatures and distribute them to its subscribers for verification of its identity. The EMSS scheme is robust enough as it computes a signature over the hashes of several packets, i.e., any packet P_i can be associated with several hash values of P_j where $j<i$, and thus facilitates receiver authentication of P_j for which a signature was computed. It was observed that the overhead could be about 20 bytes per packet under realistic scenarios.

In [21], S. Xu and R. Sandhu proposed a scheme for computing nodes such as hosts and routers to authenticate the content provider’s data packets regardless of the packet drops that occur in networks. The scheme uses two random key pairs; one for encrypting the MAC keys and another for authenticating source packets.

In [48], A. Perrig et al. presented a technique for buffering packets at the sender side instead of receiver side as employed by TESLA. This technique facilitates receiver authentication of packets immediately upon receipt.

2.3 Lost packet detection

It is very essential to have some provision in place to detect the lost packets that do not reach a destined subscriber. As the multicast packets are delivered to the end-host using unreliable UDP datagrams that offer "best-effort" service delivery, there is a high probability that some packets may be dropped by the network before reaching the end-host. If the subscriber is charged for a multicast service which he claims to have never received, there should be some provision to track the lost packets. Several researchers have proposed a few protocols that offer some methods to track the lost packets and we present them as follows.

TLS [42] and IPSec [7] require the underlying transport protocol to be reliable, but as multicast uses UDP transport, TLS and IPSec cannot be used for securing the communication link in multicast networks. Therefore, if transport reliability in the service delivery is the key issue for an application, hosts and the routers must use reliable multicast protocols to track lost packets.

A new Datagram TLS (DTLS) has been proposed by Gutman [59] in IETF. His idea is to make extensions to the most of the re-usable records of TLS to offer reliability to the datagrams at the transport layer. DTLS adds new records in the record layer to detect lost packets and facilitates re-ordering of packets. It adds sequence numbers to each record in the record layer, and relies on message acknowledgements in the handshake process. But significant modifications are required in the handshake process as well as in the TLS point-to-point approach for its applicability to multicast services.

DTLS incurs high latency in situations when the multicast host group is large causing feedback implosion. Feedback implosion occurs due to repeated negative acknowledgement messages exchanged between hosts and the data source to report packet losses. Multicast applications that operate on the basis of pay-as-you-watch streaming audio/videos usually cannot absorb this kind of high latency. They require some error resilient methods to prevent feedback implosions that lead to network congestion. One of such kind is the Forward Error Correction (FEC) technique. FEC offers reliable multicast data delivery without requiring the source to retransmit the lost packets. In [58], D. Li and D. Cheriton have evaluated the advantages of FEC for reliable multicast. They have simulated the FEC utility based on parameters such as packet error rate, average length of bursts, group density. The simulation results monitor the delivery latency and the bandwidth consumption.

The Real-time Transport Protocol (RTP) could be used for transport of audio and video streams of the multicast service. RTP/RTCP [56] provides mechanisms to detect packet loss and errors. Though RTP can work with UDP at transport, the Internet security protocols such as TLS do not work with UDP. Also, IPSec prevents header compression, and does not support unequal header compressions as it uses block ciphers. Hence, using Secure Real-time Transport Protocol (SRTP), as stated in RFC 3711, [57] can be used for securing the transport data. SRTP was designed to protect the real-time data transport over RTP as defined in IETF RFC 3711. It supports both point-to-point and multicast networks. It is also possible to use both RTP and SRTP over UDP in the IP-stack as specified in [68], where the communication between the RTP and SRTP can be established through the use of the Session Description Protocol (SDP) [60].

2.4 Resource accounting models

Accounting is nothing but collection of data about resource usage. Accounting for consumed resources depends on many parameters such as accounting type, accounting mode, and data collection type. Accounting types could be classified into on-demand, prepaid, postpaid, contract, and scheduled accounting. Accounting mode could be real-time or batch accounting. Data collecting methods refer to precise or approximate collection. The easiest accounting model to implement is a flat rate accounting, which would serve most of the purposes. Researchers have proposed several accounting models that are applicable to different operating environments. In this section, we present a survey of popular accounting models that are relevant to multicast networks.

Measuring the resource usage is difficult to achieve due to the dynamic nature of the multicast groups and distributed nature of receivers. Real-time Traffic Flow Measurement (RTFM) working group of IETF [22] developed an architecture describing various components for measuring traffic flows. It specifies the quantities and parameters that the RTFM framework needs to collect.

In [41], L. R. Dondeti et al. proposed a framework for collecting router status and flow statistics and storing them in a database. These data are later used for accounting and billing purposes. They have also presented an algorithm for accounting based on ELSD (Equal Link Split Downstream). ELSD is a multicast accounting mechanism where the cost of each link is split equally among all the downstream receivers.

The AAA working group of IETF [18] has presented a general AAA framework for securing inter-domain infrastructure. The most important components of the AAA framework are AAA server, Network Access Server (NAS), and AAA protocols. The AAA server provides distributed authentication, authorization and accounting services to subscriber's sessions. It has distributed AAA clients across networks, namely NAS [47], for providing the functionality of AAA services. The AAA services are shared securely with NAS through AAA protocols such as Diameter [71], RADIUS [69], and TACACS [70]. The AAA server accounts for the end-host's sessions by collecting and storing information about the host's session timestamps, and consumed resources.

A. Celik et al. [37] proposed a dynamic subscription strategy that allows access-based accounting for IP multicast networks. I-DG introduces two servers, one for subscribing subscribers (Subscription server) and another for indexing the source broadcast data (Broadcast server). This mechanism allows the two servers to exchange the information specific to the indexed data items. The Subscription server will assign the specific multicast group to subscribers so that the subscribers are entitled to receive only the indexed broadcast channel from the Broadcast server. This mechanism allows a subscriber to customize his service usage timing as the Broadcast server sends disseminated data items. But nothing has been mentioned about I-DG applicability to an e-commerce environment.

In [38], P. Chrysanthis et al. proposed a middleware architecture for data management, which interfaces with both of the transport and application layers. The middleware holds the content server to serve a large number of hosts. The content server is used for disseminating data through combinations of the following schemes: multicast push, multicast pull, and unicast push. In multicast push scheme, the hosts receive multicast data from the content server without any explicit request. Unlike push scheme, multicast pull scheme requires the hosts to explicitly request the resources from the content server. The content server aggregates these requests and broadcasts the data only once. The network takes the responsibility for replicating the data packets and directing them to the group members. In unicast push, hosts requests do not aggregate, forcing the server to unicast the service resources to each and every subscriber.

In [40], H. Sallay and O. Festor presented a highly distributed management architecture for dynamic IP multicast. Accounting service is provided through the managerial functions that rest on top of a three level hierarchy. The hierarchy is defined as source level, intermediate node level and edge node level. They argue that their architecture originated from the basic drawbacks in multicast dynamics such as JOIN/LEAVE operations to a multicast group, and routers leaving or joining the multicast tree.

In [39], M. Kouadio and W. Pooch classified the characteristics of several accounting models such as accounting for optimal pricing models, edge pricing architecture, zone-based cost sharing architecture, etc. Some of the characteristics that were used for categorizing these models are resource monitoring, payment support, cost sharing scheme, methods of data collection, supporting network protocols, scalability, and flexibility.

However, we did not find any literature stating on how to secure multicast sessions in e-commerce operational environment as well as how to secure e-commerce transactions for multicast services.

Chapter 3

E-payment platforms and protocols

E-payment systems can be broadly classified into e-payment platforms and e-payment protocols. The governing protocols of any e-payment system should have precise solutions to the inquiries from principal parties such as:

- Are the subscriber's e-payment credentials secured?
- Is the subscriber anonymity supported by the e-payment system?
- Does it guarantee fair accounting for the consumed resources?
- Does it non-repudiate the principal parties?
- Does it support low-cost transactions?

The answers to the aforementioned questions could be possible only after a thorough investigation into the existing e-payment platforms and protocols, followed by identification of their supporting features. In the following sections, we will discuss in detail, the popular e-payment platforms and protocols and their supporting features. We will follow up with a comparative study of these platforms and protocols to understand their pros and cons. These comparisons facilitate in identifying the e-payment standards that could be used for multicast environments.

3.1 E-payment platforms

The e-payment platforms can be classified into credit card based, e-Cash based, e-Cheque based, Smartcard based and Micropayments based platforms. These platforms are classified based on the kind of e-payments they accept to favour e-commerce between a subscriber and a merchant. The e-payment platform types with examples are as shown in Table 1. Each of these e-payment platforms is discussed in detail in the following sections with suitable examples for each of these platforms.

Table 1: E-payment platform types with examples

E-payment platform Type	Examples
Credit card	• First Virtual, CyberCash
E-Cash	• DigiCash, CyberCoin
E-Cheque	• NetCheque, NetCash
Micropayments	• CyberCoin, NetBill, Millicent, MicroMint, Pay Word
Smartcard	• Mondex, CAFE

3.1.1 Credit card based e-payment platforms

In credit card based e-payment platforms, the subscriber possesses a credit card (debit card, or any other e-card issued by an authorized financial institution) as a means of conducting e-commerce with merchants. First Virtual and CyberCash are two of the popular platforms of this type, which are discussed in detail in [45]. In this section, we will discuss First Virtual e-payment platforms in detail. First Virtual was one of the first e-payment platforms; it was fully operational since October 1994. It claims to have more than 180,000 subscribers [45]. In this e-payment platform neither subscribers nor merchants are required to install new software to participate in e-commerce. Anyone having access to Internet e-mail can do e-commerce over the Internet using this platform. First Virtual issues VirtualPINs to its subscribers in exchange for credit card details of his bank account. This First VirtualPINs offer security to subscriber’s online transactions because any intruder compromising sessions can only gain First VirtualPINs, which are of no real value to him. This is because an e-mail will be sent to the genuine subscriber by the First Virtual server to authorize the payments claimed by the purchaser. Similarly, merchants use either e-mail, telnet, or any other automated program that makes use of First Virtual's Simple MIME Exchange Protocol (SMXP) to authenticate subscriber’s VirtualPINs. Once the payments are authorized by the subscriber, First Virtual server takes the responsibility for transferring the funds into the merchant’s account. Figure 2 illustrates the process involved in a typical First Virtual payment transaction.

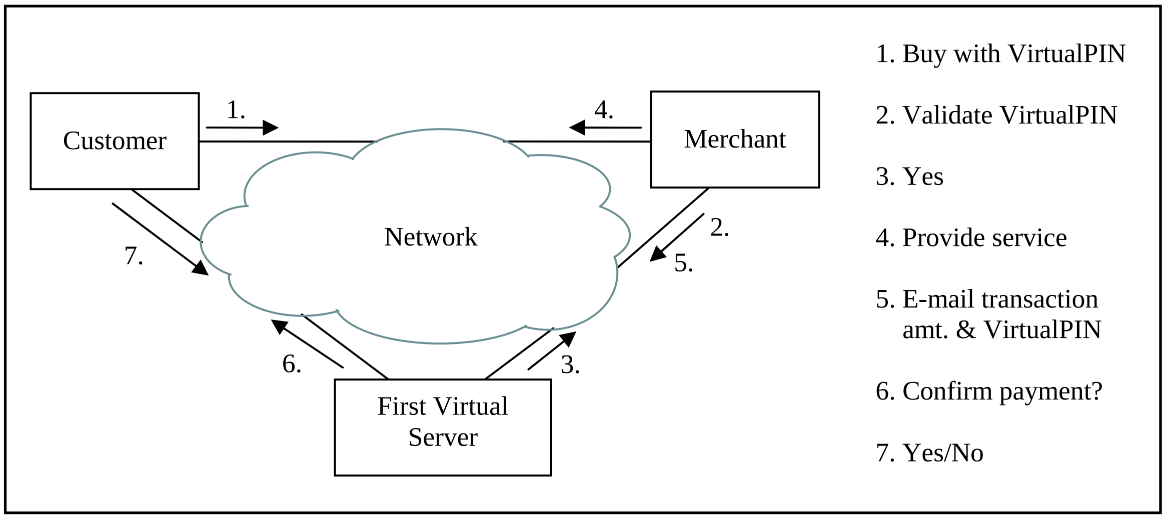


Figure 2: A typical First Virtual e-payment transaction processing

The problem with these kinds of platforms is that each of these platforms have their own bounded subscribers and basically lack interoperability with other platforms. Even the platforms are subject to the international proprietary laws and governing mandates, which further restrict their scalability.

3.1.2 E-Cash based e-payment platforms

E-cash is digital money that is provided by a Certified Bank to enable its subscribers to participate in e-commerce. Customers interested in buying digital money from an e-Cash platform must install a software package in their host machines. This software allows a subscriber to download digital cash into his cash wallet on his host machine. When a subscriber agrees to the terms of the service, the subscriber exchanges his digital cash with the merchant in return for the service offered by the merchant. The merchant can deposit this digital money in a Certified Bank that accepts digital cash. This is illustrated in the following Figure 3.

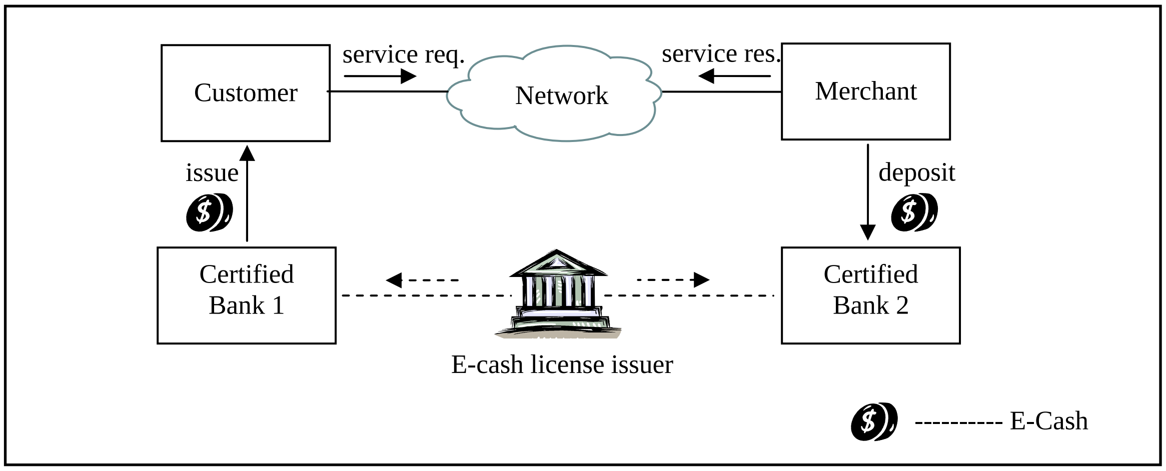


Figure 3: A typical e-Cash payment platform

A few examples of the systems that make use of the E-Cash payment platform are DigiCash and CyberCoin. DigiCash was developed by CyberCash Inc. to support anonymous payments. E-Cash platforms can operate without relying on any third party for verifying the credentials of the principal parties. Any subscriber holding e-cash acceptable by the merchants' Acquirer can directly participate in e-commerce. It requires a minimum communication cost and scales better for multicast environments. It offers total anonymity of the subscriber making it an ideal platform for applications that provide service as long as the subscriber is paying for the consumed resources. This platform resembles a real money transaction between principals much alike to paying cash at a grocery store to buy some groceries. E-Cash cannot scale to a large audience as there is no single system throughout the Internet that offers digital money. Any operational region must comply with its governing mandates, which limits the platform's scalability.

3.1.3 E-Cheque based e-payment platforms

E-Cheque operational infrastructure is similar to the e-Cash, though there are a few differences that exist in their usage. Instead of using electronic cash, the subscriber uses e-Cheque as the means of participating in e-commerce. E-Cheque serves as a means for authenticating and authorizing the payments made by the subscriber. E-Cheque contains information such as subscriber's name, account number, cheque number, the signing amount for transaction, and date of authorization by the subscriber. This information implicitly serves as evidence to non-repudiate subscriber's involvement in e-commerce transactions. Any e-Cheque that is signed by the subscriber and used as a means for participating in e-commerce is validated by both the merchant's bank and e-Cheque issuing bank. A few examples of the systems that are based on e-Cheque platforms are NetCheque and NetCash. More on e-Cheque platform design can be found in [44, 45].

3.1.4 Micropayments based e-payment platforms

Multicast services would scale better to a large number of audience members in an e-commerce environment provided the underlying payment platform is capable of supporting low-cost transactions. Low-cost transactions are Micropayment transactions that could worth just a few cents. Due to the technological and manual costs that recur in processing each payment transaction, there is currently no payment platform that is willing to support Micropayment transactions. A few of the payment platforms that are developing Micropayment technology are CyberCoin, NetBill, Millicent, Subscrip, PayWord, and MicroMint. NetBill is being developed by Carnegie-Mellon University. We will explain in detail how CyberCoins are used for participating in low-cost items transactions.

CyberCash Inc. has been operational over the Internet since 1995. It launched CyberCoin service in September 1996. Firstly, CyberCash sells software packages to principal parties that are involved in Micropayment transactions. The subscriber must download and install CyberCash Wallet software program onto his host machine. The subscriber must create a personal wallet ID and a password, which stand as an identity of his wallet when participating in e-commerce. The subscriber binds his credit card details to this wallet, which is required by CyberCash server to verify the subscriber’s credentials. The credit card binding is a procedure of binding credit card processing information such as credit card number, expiry date, billing address with each wallet. These credentials are then registered at CyberCash server. Once the wallet ID is established, and registered at CyberCash server, the subscriber can use his Virtual wallet as a source of CyberCoins. These CyberCoins facilitate the subscriber to make Micropayments in e-commerce transactions. For merchant to be able to accept payments from CyberCash buyers, he must install a Virtual CashRegister, a component of CyberCash Internet payment service: SMPS. This Virtual CashRegister provides the merchant with the ability to receive payments from his CyberCash subscribers and process them. The infrastructure of the CyberCash payment platform is depicted in Figure 4.

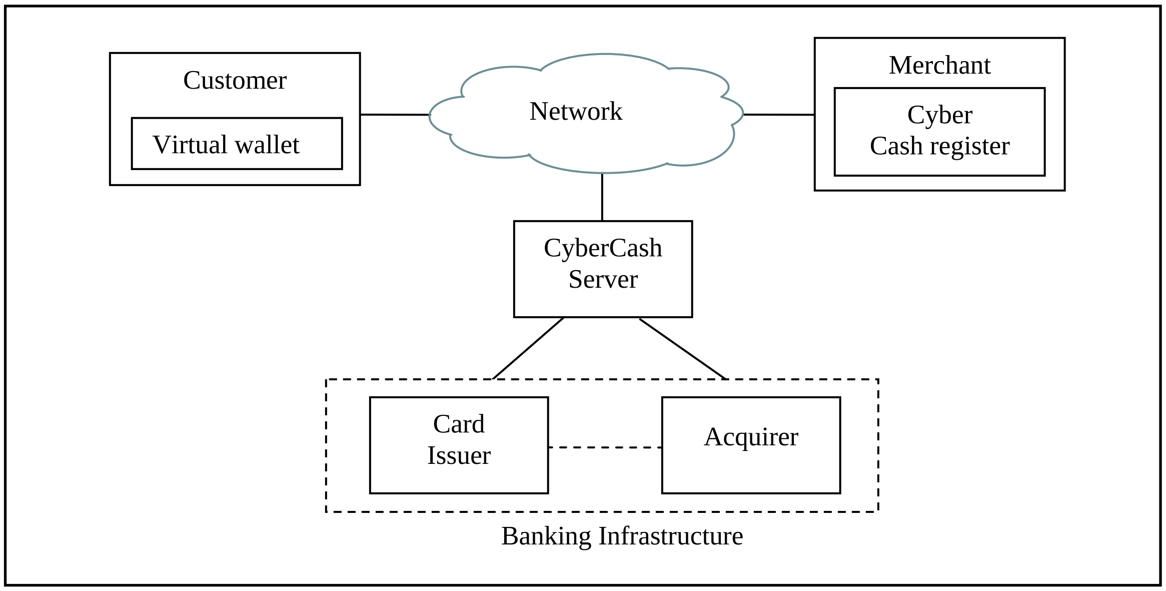


Figure 4: CyberCash payment platform infrastructure

In addition, the merchant must establish an account with an Acquirer, a bank that supports Internet transactions using CyberCash's SMPS. This is because CyberCash server can only communicate with banks that use this payment platform. CyberCash forwards the merchant’s data to Payment Gateway for processing payments of its subscribers. Payment Gateway has secure interfaces to financial institutions and it can validate subscriber’s payments. Once the payments are authorized, Payment Gateway deposits subscriber’s Micropayments into the acquiring bank of the corresponding merchant.

3.1.5 Smartcard based e-payment platforms

Smartcard or electronic purse is a new software or hardware component such as a memory chip or circuit board that needs to be integrated to the computer hardware. Smartcards use offline e-payment methods, which resemble a prepaid payment scheme. These cards usually have their own operating system and file system. It implicitly facilitates subscriber’s authentication in e-commerce. Using smartcards, the subscriber and the merchant can participate in e-commerce transactions at their will and wish. The smartcard technology will bring Micropayments technology into fully operational mode thus bringing a new revolution in the payment technology.

However, the primary problem with this platform is its technology is still not matured enough for widespread adoption. It lacks interoperability with other payment platforms and protocols. Also, it is not cost effective for a subscriber who rarely participates in e-commerce to buy a smartcard. A few examples that make use of smartcard technology are Mondex, and CAFE. The logical and physical design issues for Smartcard databases can be found in [43].

3.2 Credit card based e-payment protocols

In this section we will discuss the most popular e-payment protocols that are deployed in the internet or intranet domain. Specifically, we will discuss SSL/TLS, SET, iKP and Micropayments based e-payment protocols.

3.2.1 SSL e-payment protocol

Today, e-commerce transactions are typically protected using Secure Socket Layer (SSL) protocol. SSL is the simplest e-payment protocol that provides secure end-to-end communication over insecure channels above TCP/IP. SSL is an application layer independent protocol that does not require any additional plug-in to the client software. It supports authentication of the content provider, data encryption and message integrity. However, it does not provide subscriber authentication and thus non-repudiation is not supported. SSL uses the RSA digital signature to authenticate the principal parties. SSL allows server and client to authenticate each other by negotiating on session keys. SSL utilizes standard certificates for authentication and data encryption to ensure privacy or confidentiality. Transport Layer Security (TLS), as of 1996, is a direct descendent of SSL 3.0. SSL/TLS is supported by the majority of Internet browsers such as Netscape and Microsoft Explorer. In SSL/TLS, only merchants are required to possess public-key certificates while subscribers could be anonymous.

Since SSL/TLS protects data only while it is being transmitted, the content provider has access to the subscriber's sensitive information such as the credit card details. The plaintext format of the credit card information at the content provider server therefore represents a security breach that is not currently addressed by the use of the SSL/TLS. More on the TLS handshake process, client key exchange and certificate verification process can be found in [42]. The authors in [42] have performed an inductive analysis of TLS, using the theorem prover Isabelle to prove TLS security goals.

3.2.2 SET e-payment protocol

The Secure Electronic Transactions (SET) specification is an open technical standard for the commerce industry developed by Netscape, Visa, MasterCard and other partners as a way to facilitate secure e-payment transactions over the Internet. Digital Certificates create a trust chain throughout the transaction, verifying subscribers and merchants' authenticity, a process unparalleled to any other Internet security solutions. The SET payment process is slightly more complicated than the other payment protocols because it requires distributing public keys among all principal parties and verifying certificates during each transaction. A typical transaction of the SET protocol is illustrated in Figure 5.

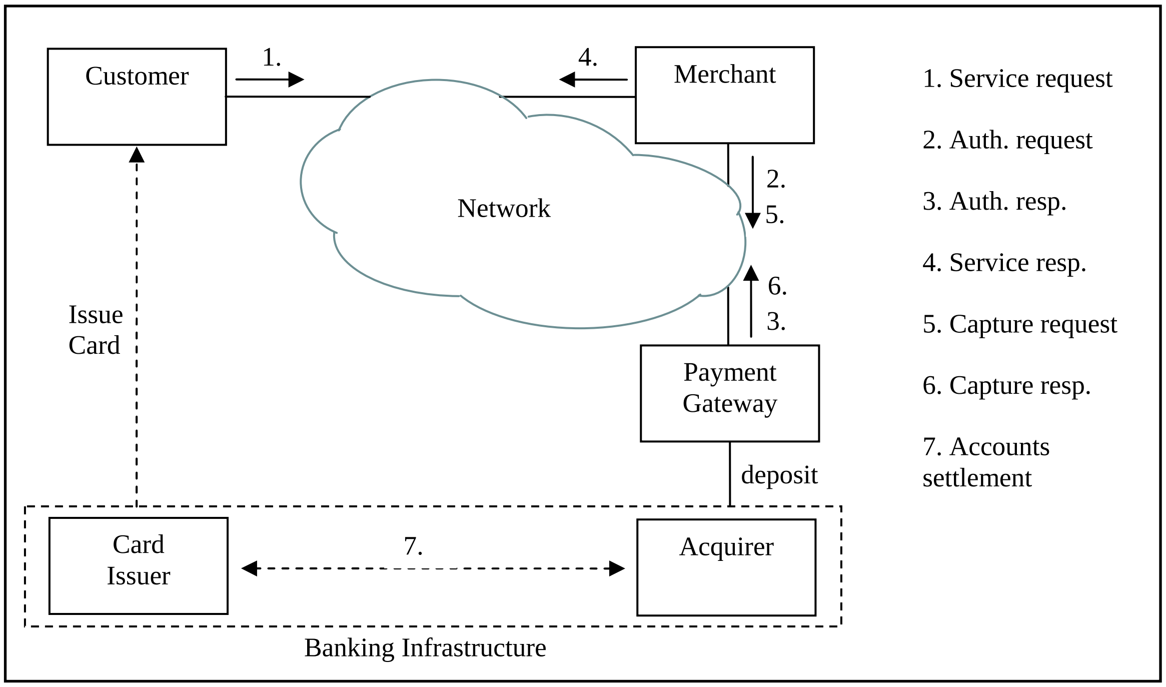


Figure 5: The SET protocol payment method

SET relies on cryptography and digital certificates to ensure message confidentiality and security. Authentication and non-repudiation are provided through the use of digital signatures. In the SET protocol, message data are encrypted using a randomly generated 56-bit DES session key that is further encrypted using the recipient’s public key. This is referred to as the "digital envelope" of the message and is sent to the recipient with the encrypted message. The recipient decrypts the digital envelope using a private key and then uses the symmetric key to decipher the original message. The computational cost of asymmetric encryption is cited as reason for using weak 56 bit DES (IBM, 1998).

SET overcomes the drawbacks of SSL as it offers protection of all the transaction data. SET offers non-repudiation of all the principal parties involved in e-commerce transaction. Unfortunately, SET is difficult to deploy for widely distributed audiences with varying constraints in terms of host computing power, communication overhead, bandwidth limitations, network support, and service requirements. Other technical reasons that are hampering its deployment are overhead in initializing and implementing the SET protocol. The communication overhead is due to the computationally complex procedure that is involved in authentication of the subscriber and authorization of his payments. The procedure involves exchange of digital signatures and certificates, encryption of data, and verifying the payment credentials. The SET business description, programmer’s guide, formal protocol definition, and protocol description can be found in [30].

3.2.3 iKP family of secure e-payment protocols

The iKP protocols were first developed in 1995 by a group of researchers at the IBM research labs [29]. It is a multi-party secure scheme where no party is forced to trust other parties with no reason. iKP protocols are based on public-key cryptography. The protocols are named 1KP, 2KP, and 3KP depending on the number of public key-pairs used in each payment system involving the merchant, subscriber and Payment Gateway (PG) that processes e-payment details of the subscriber. In the 1KP protocol, the PG alone possesses a public key while subscribers and merchants only need to have authentic copies of the PG's public key, reflected in a public key certificate. 1KP involves a minimal PKI as CA issues certificates only to the PG. This protocol does not provide non-repudiation for the messages exchanged between subscriber and merchant. In the 2KP protocol, the merchant along with PG possess public key-pairs where e-payments are authorized only after verifying the credit card number (CC#). In the 3KP protocol, all the three players possess public key-pairs where e-payments are authorized only after validating both the CC# as well as digital signature of the subscriber.

3.2.4 Micropayments based e-payment protocols

The Micropayments based e-payment protocols are being developed to support low-cost transactions. The advent of this technology fundamentally benefits the content providers who are presently serving their web contents for free to their potential content visitors. For example, application servers that could be benefited from this technology are not limited to advertisements, online news, online chat rooms, online games, voice over IP phone calls, etc. If the cost per viewing or downloading content on the Web is very low (one tenth of a cent to a dollar), the customer would not mind to pay for the merchant's services. Micropayments protocols will enable merchants to collect very small remunerations such as a few cents from its potential subscribers. A few examples that use Micropayments based payment protocols are µ-iKP, Compaq's Millicent, NetCard, PayWord, PayTree, and W3C'S MPTP.

3.3 Comparative study of e-payment systems

A comprehensive comparison of the e-payment platforms and protocols is presented in this section, which provides thorough knowledge on their usage and applicability in the Internet or Intranet.

3.3.1 E-payment platforms comparison

Table 2 provides comparison of e-payment platforms based on the features such as accepted payment type, payment mode and payment support, contract type, authentication, interoperability, communication cost, anonymity support, and platforms usage trend.

Table 2: Comparison of the e-payment platforms

S.No	Feature	Card based	E-Cash	E-Cheque	Smartcard
1	Payment Type	Card based	E-Cash	E-Cheques	E-Cash
2	Payment Mode	Online	Online	Online	Offline
3	Payment Support	Macro	Micro/Macro	Macro	Micro/Macro
4	Contract Type	All	Pay-before	Post-paid	Pre-paid
5	Cryptography Use	Yes	No	Yes	No need
6	Authentication	Customer, Merchant	None	Customer	Customer
7	Non-repudiation	Yes	No	Yes	No
8	Anonymity	Yes/No	Yes	No	Yes/No
9	New H/W, S/W	No	Yes	No	Yes
10	TTP Required	Yes	No	No	No

S.No	Feature	Card based	E-Cash	E-Cheque	Smartcard
11	Interoperability	No	No	No	No
12	Comm. Cost	Nominal	Minimum	Nominal	High
13	Amount Restriction	Yes	Yes	No	No
14	Usage Trend	Most	Little	Little	Little

From the above comparative study, we can notice why credit card based e-payment platforms are most popular than other payment platforms in the current Internet infrastructure. This is because of the simplicity, flexibility and ease in making use credit card, which are primary driving forces that any principal party will look for before participating in e-commerce. This is the only platform that can hold any type of contract ranging from prepaid, postpaid, scheduled or on-demand payment. In contrast, E-Cash is favorable in situations when the customer wants to keep anonymous to its merchant. As long as the merchant receives digital cash for his services to its customer, the platform serves its purpose. As the platform supports both micro and macro e-payments, it becomes advantageous to rely on this platform for making low-cost transactions. But the problem is it does not support security features such as authentication of principal parties, and non-repudiation; thus cannot be operated in infrastructures that need authentication of principals before their access to service resources.

3.3.2 E-payment protocols comparison

In this section we compare the following protocols: SSL, SET, and iKP protocols based on the security features such as authentication, payment authorization, communication link protection, confidentiality, integrity, money atomicity, non-repudiation, anonymity, and simplicity. This comparison has been drawn to better understand the suitability of each of these e-commerce protocols to secure e-commerce transactions. Table 3 provides a comprehensive comparison of the SSL, SET and iKP support to secure e-payments. The principal parties for which this comparison is illustrated below are the subscriber and the merchant. We would not discuss in further the comparison results of the below Table 3 as it is self explanatory to any reader.

Table 3: Comparison of the security features support by e-payment protocols

S.No	Security feature	SSL	SET	1KP	2KP	3KP
1	Authentication	Merchant	All principals	Customer	Customer, Merchant	All principals
2	Payment Authorization	No	Yes	Yes	Yes	Yes
3	Comm. Link Protection	Yes	Yes	No	No	Yes
4	Confidentiality	Merchant	Customer, Merchant	Merchant	Customer, Merchant	Customer, Merchant
5	Integrity	Yes	Yes	No	No	No
6	Money Atomicity	No	Yes	No	Partial	Yes
7	Non-repudiation	No	Yes	No	Yes	Yes
8	Anonymity	Merchant	No	Merchant	No	No
9	Simplicity	Simple	Complex	Simple	Medium	Complex

Chapter 4

Common Criteria

This chapter provides an overview of the key Common Criteria (CC) concepts that help the reader in understating the Protection Profile (PP) concepts. Discussion on the security functionality and security assurance specifications [62, 63] is out of scope. The discussion in this chapter is primarily intended for readers who have no background on PP concepts, to prepare them for the multicast security PP, which is presented in chapter 5.

4.1 Common Criteria basics

The official name of Common Criteria is “The Common Criteria for Information Technology Security Evaluation”, but it is often referred as Common Criteria (CC). The CC is the first security standard that has been widely recognized by the international community. It has evolved to unify IT certification efforts that were put forth by several organizations. It removes the need for an information and communication technology (ICT) product or system developer to undergo multiple evaluations of their product in order to sell them internationally. The CC were developed by several national security organizations: AISEP (Australia and New Zealand), BSI (Germany), CESG (UK), CSE (Canada), NIST (USA) and NSA (USA), NLNCSA (Netherlands), and SCSSI (France). CC version 1.0 was first published for comment in January 1996. After extensive review, CC version 2.0 was published in May 1998. The International Organization for

Standardization (ISO) then adopted CC version 2.0 as a Final Committee Draft (FCD). After some editorial and minor technical changes, the CC became ISO 15408 in June 1999. Later, CC version 2.0 was updated to CC version 2.1, which conforms completely to the International Standard. An ICT product or system that has been certified through the CC standard in a nation is automatically recognized in all other participating countries. Thus, it behaves as a standard to eliminate the redundancy in a product or system evaluation process.

The CC provide a common set of requirements for the security functions of ICT products and systems and for assurance measures applied to them during a security evaluation. The ICT product or system (including hardware, software and documentation) that is being subject to evaluation is referred to as the Target of Evaluation (TOE). The CC provides a basis for evaluating the security properties of the TOE. Such a standard evaluation of security properties will be meaningful to a wider audience. The CC evaluation scheme establishes a confidence level that the security functions of such a TOE and the assurance measures applied to them meet these requirements. The CC evaluation scheme favors comparability between the results of independent security evaluations. Thus, the CC evaluation results may help consumers to determine whether the TOE is secure enough for their intended application and whether the security risks implicit in its use are tolerable. The CC focuses on threats to the ICT information that arise due to the malicious and non-malicious human activities, but may be applicable to some non-human threats as well.

The CC are used to create two kinds of documents, namely a Protection Profile (PP) or a Security Target (ST). A PP is a document that identifies the desired security properties of an ICT product or system. The PP allows consumers or developers to define sets of security requirements that are known to be useful and effective in meeting the identified objectives. An ST is a document that identifies the security-relevant aspects of what an ICT product or system (or a subset of it) actually does. It contains the security features for a specifically identified TOE and defines the functional and assurance measures offered by that target to meet the necessary security level. It is quite possible that an ST does not meet the requirements of any particular PP, but it could meet the requirements of one or more PPs. The layered representation of different levels of CC can be depicted in Figure 6 [61].

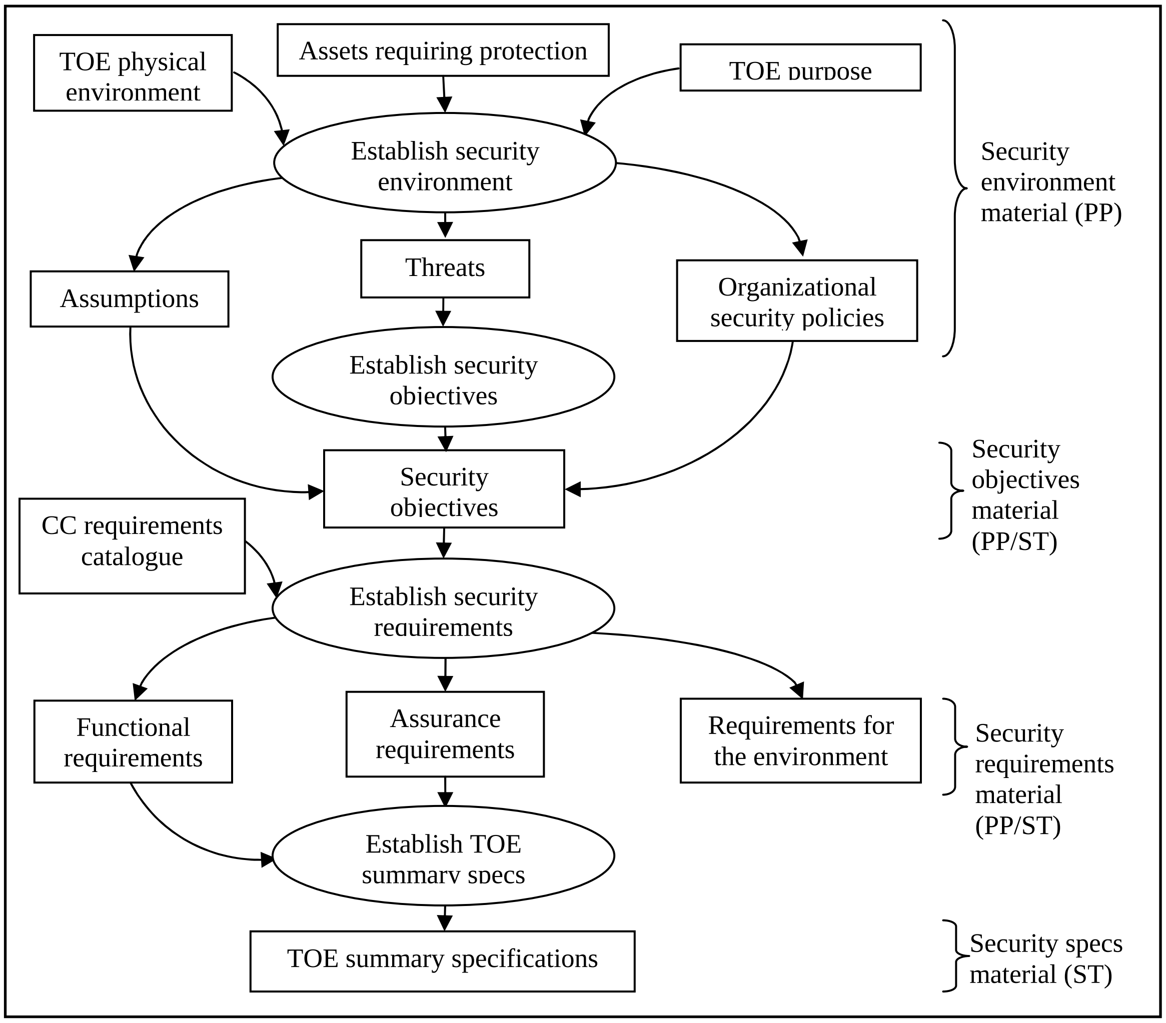

TOE physical environment
Assets requiring protection
TOE purpose
Establish security environment
Assumptions
Threats
Organizational security policies
Security environment material (PP)
Establish security objectives
Security objectives
Security objectives material (PP/ST)
CC requirements catalogue
Establish security requirements
Functional requirements
Assurance requirements
Requirements for the environment
Security requirements material (PP/ST)
Establish TOE summary specs
TOE summary specifications
Security specs material (ST)

Figure 6: Derivation of requirements and specifications

In short, the TOE security threats, objectives, requirements, and security functional specifications and assurance measures together form the primary inputs to the ST. More information on ST can be found in [61, 62, 63]. The Figure 6 shows how the security requirements and specifications can be derived when deploying a PP or ST. The first layer shows how security objectives can be derived for a PP. The security environment is established based on the specifications of the TOE physical environment, assets to protect, and the TOE purpose. The threats are identified for the established security environment that ultimately helps in establishing the security objectives. These threats in the above figure do not bind the means by which PPs and STs are developed, but it shows how the results of some analytic approaches relate to the content of PPs and STs.

4.2 Protection Profile specifications

In this section, we will outline the basic concepts that are necessary to document a PP. In the following sub-sections we will define a PP followed by the definitions of the terminologies that are necessary to completely understand how a PP can be documented.

4.2.1 Definition

A PP is a statement of an implementation-independent set of security requirements and objectives for a category of TOEs. It identifies the security problem that a compliant ICT product or system must solve. Its concept has evolved to support the definition of functional standards, and as an aid to establish TOE summary specifications. It serves as a reusable component in evaluating an ICT product or system.

4.2.2 Content

The basic components that are necessary to define a protection profile are as follows: PP identification and overview, TOE description, TOE security environment, security objectives, and IT security requirements. The hierarchical representation of the components that completely define PP contents are depicted in Figure 7 [61].

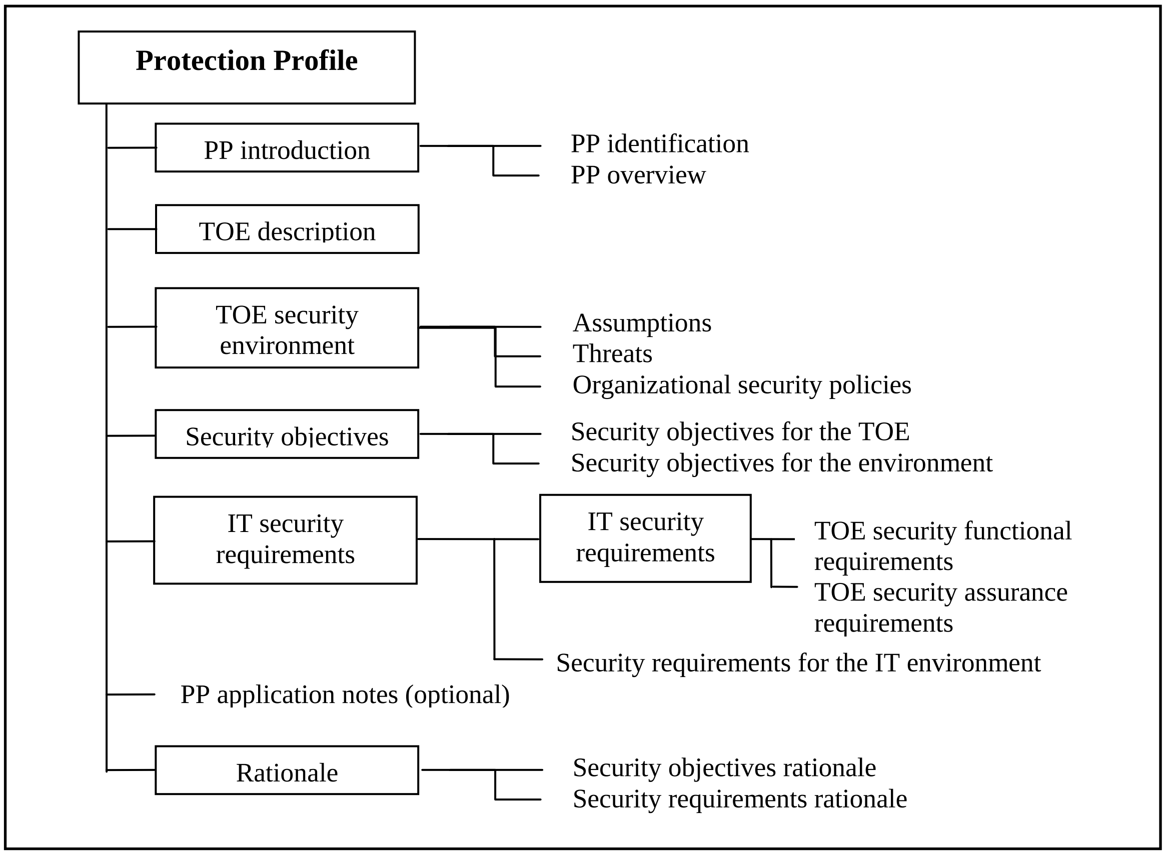

Protection Profile
PP introduction
PP identification
PP overview
TOE description
TOE security environment
Assumptions
Threats
Organizational security policies
Security objectives
Security objectives for the TOE
Security objectives for the environment
IT security requirements
IT security requirements
TOE security functional requirements
TOE security assurance requirements
Security requirements for the IT environment
PP application notes (optional)
Rationale
Security objectives rationale
Security requirements rationale

Figure 7: Protection Profile content

Each of these components is explained as follows.

- *PP identification and overview:* The PP identification provides labeling and descriptive information necessary to identify, catalogue, register, and cross reference a PP. The PP overview summarizes the PP in narrative form. The overview should be sufficiently detailed for a potential user of the PP to determine whether the PP is of interest. The overview should also be usable as a stand alone abstract for use in PP catalogues and registers.

- *TOE description:* It describes the TOE as an aid to the understanding of its security requirements, and shall address the product or system type and the general IT features of the TOE. The TOE description provides context for the evaluation. The information presented in the TOE description will be used in the course of the evaluation to identify inconsistencies. As a PP does not normally refer to a specific implementation, the described TOE features may be assumptions. If the TOE is a product or system whose primary function is security, this part of the PP may be used to describe the wider application context into which such a TOE will fit.
- *TOE security environment:* It describes the security aspects of the environment in which the TOE is intended to be used and the manner in which it is expected to be employed. This statement shall include a description of assumptions, threats, and organizational security policies. Each of these is explained as follows:
    - *Assumptions:* It includes information about the intended usage of the TOE, including such aspects as the intended application, potential asset value, and possible limitations of use; and information about the environment of use of the TOE, including physical, personnel, and connectivity aspects.
    - *Threats:* It describes the threats to the assets against which specific protection within the TOE or its environment is required. A threat shall be described in terms of an identified threat agent, the attack method, any vulnerability that is the basis for the attack, and the asset that is the subject of the attack. Threat agents should be described by addressing aspects such as expertise, available resources, and motivation. Attacks should be described by addressing aspects such as attack methods, any vulnerabilities exploited, and opportunity.

    - *Organizational security policies:* It identifies and/or states the organizational security policies with which the TOE must comply. These policies serve as statements to lay out clear security objectives.

- *Security objectives:* It is a statement of the security objectives for the TOE and its environment. The security objectives shall address all of the security environment aspects identified. The security objectives shall reflect the stated intent and shall be suitable to counter all identified threats and cover all identified organizational security policies and assumptions. The following categories of objectives shall be identified.
    - *Security objectives for the TOE:* They are clearly stated statements that are intended to counter the identified threats and/or organizational security policies and assumptions to be met by the TOE.
    - *Security objectives for the environment:* They are clearly stated statements identifying threats that are not completely countered by the TOE and/or organizational security policies or assumptions not completely met by the TOE. It usually re-states, in part or whole, the statements of the TOE security environment assumptions portion.

- *IT security requirements:* They describe the ICT security requirements that shall be satisfied by the TOE or its environment. These statements are written so that they meet the security objectives of the TOE or its environment. The ICT security requirements are categorized into two kinds, namely functional requirements (what an ICT product or system is intended to do), and assurance requirements (what are the intended measures to be performed to make sure that the objectives have been met). More information on the IT security requirements can be found in [61].

- *Application notes:* It is an optional component of the PP that may contain additional supporting information that is considered relevant or useful for the construction, evaluation, or use of the TOE.

- *Rationale:* This part of the PP presents the evidence used in the PP evaluation. This evidence supports the claims that the PP is a complete and cohesive set of requirements and that a conformant TOE would provide an effective set of IT security countermeasures within the security environment. The rationale can be categorized into security objectives rationale and security requirements rationale. The security objectives rationale shall demonstrate that the stated security objectives are traceable to all of the aspects identified in the TOE security environment and are suitable to cover them. The security requirements rationale shall demonstrate that the set of security requirements of TOE and its environment is suitable to meet and traceable to the security objectives. More information on the security requirements rationale can be found in [61].

Once a PP has been developed, it can go through a formal evaluation. The formal evaluation of a PP ensures that it meets various documentation rules and security checks. A consumer can specify the desired security functionality of an ICT product or system as a PP and is asked to select an evaluation assurance level (EAL) from a defined set of increasing assurance levels (i.e., from EAL 1 to EAL 7), which is described in the following section 4.3.

4.3 CC evaluation assurance levels

Any ICT product or system can be evaluated through the CC evaluation measure namely, evaluation assurance level (EAL) and the corresponding security functional requirements, which the ICT product or system fulfills. An EAL can be interpreted as a confidence level in the security functions of an ICT product or system. There are seven hierarchical EALs, ranging from EAL 1 to EAL 7. It should be noted that with an increase in assurance level, there would be increase in the requirements and information for each certification. We describe each of these EALs as follows:

- *EAL 1 – Functionally tested:* It offers the most basic level of assurance. It is certified where some confidence level for the product is required, but the threats to security of product are considered insignificant or not serious. The product evaluating laboratory performs a review of the product's document and performs tests on a subset of the product.
- *EAL 2 – Structurally tested:* It offers a low to moderate level of assurance. At this level, the product developer furnishes design information and test results. The product evaluating laboratory performs a review of the product's high level design.
- *EAL 3 – Methodically tested and checked:* It offers a moderate level of security assurance. At this level, the TOE and its development are thoroughly investigated and evidence of an independent product's evaluation results conformance with the product developer's test results is gathered. To meet this confidence level, the development environment controls and the TOE configuration management are also necessary.
- *EAL 4 – Methodically designed, tested, and reviewed*: It offers a moderate to high level of assurance. At this level, the product evaluating laboratory reviews the low level design and performs an independent vulnerability analysis. The product should incorporate some informal test models of security policies. These models should be capable of automated configuration management.
- *EAL 5 – Semiformally designed and tested,* EAL 6 – *Semiformally verified design and tested,* and EAL 7 – *Formally verified designed and tested* are not practically certifiable to existing commercial products. This is because retrofitting an existing product line is not feasible beyond EAL 4 as specified in [61].

Therefore, EAL 4 is treated as the highest level that any CC participating nations mutually recognize. More information related with evaluation assurance levels can be found in [61, 62, 63].

Chapter 5

Multicast Security Protection Profile

This chapter details a Multicast Security Protection Profile (MSPP) that captures multicast security requirements to secure multicast sessions in an e-commerce environment as Target of Evaluation (TOE). The MSPP is prepared in accordance with the Common Criteria (CC) Version 2.1 as specified by ISO 15408. The MSPP states the requirements that multicast sessions must satisfy in order to respond to the needs of e-commerce. The TOE security environment, which is composed of threat agents, source of vulnerability, sources of attack and threats description for the TOE, is described in detail. It is followed by a description of the administrative security policies that are necessary to safeguard the TOE or its operating environment. The risks to the TOE are identified and are followed with a statement of the security objectives for the TOE.

5.1 MSPP identification and overview

In this section, first we present the identification and control information for this MSPP document. MSPP identification provides the necessary information to identify this PP. Any reader could become aware of the developed PP name and version, and the CC version that has been used in writing this PP. The following information furnishes these details.

MSPP Title:	Multicast Security Protection Profile
MSPP Version:	1.0, dated August 2005
CC Version:	CC Version 2.1 (ISO 15408)
Author:	A. Venkataiahgari

The MSPP overview provides the descriptive information necessary to identify and catalogue the multicast security issues for e-commerce sessions. The overview is a statement of multicast security problems that exist in Intranet, Internet or Extranet. The MSPP issues (both technical and non-technical) are identified in Table 4.

Table 4: MSPP issues

Issue	Description
Membership Dynamics	Network must be capable of tracking current membership during a session's lifetime. More on multicast members dynamics can be found in [12, 26]
Authentication	Authenticate the identity of principals such as end-host, subscriber, merchant, and messages in transit. More on authentication of source data can be found in [48]
Privacy	Protect the privacy of honest principals data
Authorization	Authorize e-payments, and subscriber's access to resources (only authorized subscribers should gain access to multicast groups)
Resource Accounting	• Identify who is communicating with whom, how much, how often? • Provide audit mechanisms to non-repudiate principal parties

Issue	Description
Session Management	• Access to the session key material should be restricted through a registration and authentication process for secure group communication. A survey on key management for secure group communication can be found in [11] • Manage key distribution since hundreds of groups with only a few group members in each may cause network congestion • Method to overcome from 1-affects-n type multicast failure [17], i.e., when a new member joins the group, a new source based tree needs to be reconstructed
Scalability	Mechanism to limit the overhead incurred due to control and data signals among various principals. Multicast scalability issues can be found in [17, 32, 35]
Robustness	Ability of the multicast networks to sustain operation under heavy loads since a framework that has a high frequency of signaling with poor communication controls may have its components function unreliably due to network congestion

5.2 Target of Evaluation

This section provides context for the Target of Evaluation (TOE) by stating the features that are outside the scope of TOE discussion. The TOE aims to outline e-commerce security features such as authentication and authorization of e-commerce parties, protection of data confidentiality, data integrity, non-repudiation of parties and accountability.

5.2.1 Scope of the TOE

There are several requirements of securing multicast sessions in an e-commerce environment. But, addressing each requirement requires a lot of insight into several research areas. Therefore, we separate the features that this PP addresses from those that are outside the defined TOE. The TOE features that are outside the scope of the defined TOE are stated in Table 5.

Table 5: Security features that are outside the scope of the TOE

Feature	Description
Physical security	Protection of the components at physical layer
Multicast group management	The technical and non-technical complexity involved in managing the multicast groups [17]
Multicast session key management	Multicast session key distribution that relates to the multicast key management [11]
External environment	Threats, risks, and security objectives evolving from outside the defined TOE
Certified delivery	The issues related with the guaranteed goods/service delivery for the subscribers e-payment

5.2.2 Security features of the TOE

The security features of the TOE are the properties that govern the security infrastructure of e-commerce interaction. The following security features as stated in Table 6, are identified as essential features that the TOE is intended to provide to secure multicast sessions.

Table 6: Summary of the TOE security features

Feature	Description
Authentication	Authenticate subscribers, merchant's content providers, messages in transit and principals governed by the protocols used for conducting e-commerce. More on authentication in e-commerce can be found in [46]
Confidentiality	Prevent or detect leakage of sensitive data
Integrity	Prevent or detect corruption of sensitive data
Authorization	Authorize subscribers and their e-payments
Non-repudiation	Collect a non-refutable proof of principal's involvement in e-transactions such as identifying and time stamping who has communicated with whom, how much and how often.
Accounting	Define revenue collecting method such as revenue collection at the sender-side or receiver-side
Money atomicity	Guarantees that e-commerce protocols neither create nor destroy money. More on issues related to atomicity in e-commerce can be found in [55]

Feature	Description
Good atomicity	Guarantee that the merchant receives e-payments only if the subscriber receives the goods/service
Anonymity	Provide support to allow anonymous events (transactions, identity of subscriber, description of goods)

5.3 TOE security environment

The TOE security environment classifies the nature of security problems the TOE is intended to address. It describes the assumptions, threat agents, vulnerability sources, sources of attack, and threats to the TOE or its security environment. It concludes by stating the organizational security policies with which the TOE is intended to be operational.

5.3.1 Assumptions

The TOEs usage assumptions outline the assumptions that are made in the TOE security environment. In Table 7, we outline the assumptions that dictate the operating environment of the TOE.

Table 7: Secure usage assumptions

Assumption	Description
Privileged user	Authorized administrators or the trusted third persons (e.g., TTP) are implicitly assumed to be trustworthy; however, they can act unfairly upon

Assumption	Description
	flaws in their governance
Outside threat	The TOE operating environment is assumed to have the necessary security protocols to resolve any security threat that emerged from outside the defined TOE
Network route	The network protocols are assumed to forward the multicast packets in situations where the receiving node is not capable of forwarding multicast packets

5.3.2 Threats to security

This section identifies the threat agents that are responsible for causing damage to the TOE security. This is followed by identifying the sources of vulnerabilities and the attacks that could possibly threaten the security of the TOE or its environment. The intended organizational security policies are described in the last section to counter these security threats.

5.3.2.1 Threat agents

A threat agent is a computing or communicating principal that is identified as generating a specific class of threats to the TOE security environment. In this section, we classify threat agents based on principal party's expertise requirement, resource usage, and the motivation of their threat. The threat agents relevant to the TOE or its operating environment are described in Table 8.

Table 8: Threat agents for the TOE

Threat agent label	Description			
	Threat agent	Expertise	Resources	Motivation
TA.INSIDER	TTP, RA/CA, policy server, service provider, merchant	Low/High	Moderate/ Substantial	Malicious/ Non-malicious
TA.BAD_INSIDER	Malicious service provider, malicious merchant, malicious subscribed customer	Low/High	Moderate/ Substantial	Malicious
TA.LEGAL_HOST	Subscribed customer	Low/High	Moderate/ Substantial	Malicious
TA.PREV_NEW_INSIDER	Former subscribed customer, or new subscriber	Low/High	Moderate	Malicious
TA.INVADER	Unprivileged external sources	High	Minimal/ Moderate	Malicious
TA.NATURE	Earthquake, fire, flood, etc	N/A	Substantial	N/A

5.3.2.2 Sources of vulnerability

Several vulnerabilities could appear due to improper system design, underlying protocol weaknesses, improper application of security protocols, improper or unclear service parameters, single point of failure, etc. In Table 9, we state the sources of vulnerability that are applicable to the TOE or its operating environment.

Table 9: Sources of vulnerability for the TOE

Vulnerability label	Vulnerability	Description
V.ARCH_DES	Improper system architecture and design supporting multicast services	Poor system architecture and design leading to disputes by the subscribers, merchants, payment systems over the service charges; overall resulting in unreliable e-commerce with security holes and breaches
V.AUTH_PROT	Use of clear text authentication protocols	Weak remote user authentication and authorization techniques leading to severe security breaches in the framework
V.PAY_PROT	Poor development of e-payment systems	Inefficient payment platforms or protocols application to e-commerce without proper quality testing will jeopardize security features such as confidentiality, integrity, money atomicity and availability
V.MIDDLEWARE	Poor middleware des-	Improper middleware application to e-

Vulnerability label	Vulnerability	Description
	ign and development	commerce leading to inefficient data access and/or security breaches in the framework
V.SERV_PARAM	Not clearly defined multicast service policies, service para-meters	Merchants trust relation with the subscriber at risk due to unclear specifications of the multicast service policies, unclear session parameters jeopardizing availability of the services
V.POLICIES	Insufficient and unclear plans, procedures and policies	Overstated or undermined plans, procedures and policies with no formal validation or justification will result in substantial loss of resources and time
V.SPOF	Single Points of Failure	Inefficient routing protocols usage or improper design layout resulting in points of failure at a source of data transfer resulting in service disruption

5.3.2.3 Sources of attack

The sources of attack arise not only due to the operating environment of the TOE but also due to the weaknesses in the underlying protocols that dictate the TOE operating environment. It is possible that there exist several kinds of sources of attack on different operating environments. In

Table 10, we define the possible attack sources that are specific to the TOE or its operating environment.

Table 10: Sources of attack for the TOE

Attack label	Description
A.REPLAY	Attack exploits the weaknesses in the system design and implementation of the protocols to launch an attack such as replay attacks as identified in [50]
A.DOS	DoS and DDoS attacks mostly involves either resource exhaustion or corruption of the OS runtime environment such as UDP bombing, ICMP or Smurf attacks or memory overflow attacks, TCP SYN flooding, CGI bin attacks. More information on DoS and DDoS attacks can be found in [19, 21, 49]
A.SNIFF	Sniffing captures data packets on the network and favors analysis of the network protocol to launch further attacks. There are many free tools available on Internet such as DSniff, AirSnort, etc. More on sniffing can be found in [51]
A.SPOOF	Spoofing exploits weaknesses in the user authentication protocols. More on how spoofing works can be found in [52]
A.BRUTE	Attack exploits weaknesses in the underlying cryptographic building blocks used in the payment system such as cryptanalysis as stated in [44]. More on how Brute force attacks work can be found in [53]
A.SOCIAL_ENGG	Compromising network security by counterfeiting a legitimate principal

Attack label	Description
	to give away hints enough to break into system and launch malicious attacks such as virus attacks, data modification. More on these kinds of attacks can be found in [54]
A.MALCODE	Compromised network may lead to spreading virus or worms (malicious codes) to the service provider, its subscribers or other interacting principals. A good insight into popular viruses, Trojan horses and worms in the Internet can be found in [19]
A.BUFFER_FLOW	Attacker places malicious script/code in the buffer's overflowing area that may pose as an attack at the time of program execution. More on buffer overflow attacks can be found in [20]
A.SQL_INJECT	The attacker injects a code that does not filter input that is being entered directly into a form. This injected code poses as an attack once the attacker gets access to protected data. More on SQL injection attacks can be found in [31]
A.FORMAT_STR	The attack is due to the use of unfiltered user input as the format string parameter in programming language functions that perform formatting. For example, in C language, the printf() function can be exploited by a user to crash the program or execute malicious code. More on format string attacks can be found in [64]
A.ERROR	A privileged user accidentally issues a bad command, which results in disputes among principals

Attack label	Description
A.NATURE	Natural disaster such as earthquake, fire causing service unavailability

5.3.2.3.1 Threat description

A threat in an operating environment could arise due to its dishonest staff and trusted staff, authorized user and unauthorized user, natural calamities, virus or worm spreading, etc. In this section, we describe a threat in terms of the threat agents that cause attacks. The threat description would also provide a statement of the assets that will be prone to a threat. In Table 11, we describe the threats that are intended to be addressed by the TOE, based on the sources of threats possibly from threat agents and sources of attacks that were captured in the previous sections.

Table 11: Threats addressed by the TOE

Threat label	Description
T. MAL_ANALYSIS	Malicious users (TA.BAD_INSIDER, TA.PREV_NEW_INSIDER, TA.INVADER) do an attack (A.SNIFF, A.SOCIAL_ENGG) to acquire sensitive information stored in host machines, servers or data in transit
T.MAL_MODIFY	Malicious users (TA.BAD_INSIDER, TA.PREV_NEW_INSIDER, T.INVADER) do an attack (A.SNIFF, A.SPOOF, A.MALCODE, A.BUFFER_OVER, A.SQL_INJECT, A.FORMAT_STR) to modify sensitive information

Threat label	Description
T.CONFIDENTIAL	Malicious and non-malicious principal parties (TA.BAD_INSIDER, TA.LEGAL_HOST, TA.PREV_NEW_INSIDER, TA.INVADER) may impersonate other principals by an attack (A.SPOOF, A.SOCIAL_ENGG, A.SQL_INJECT, A.FORMAT_STR) for pleasure or monetary benefits
T.COVERT_CHANNEL	Malicious and non-malicious principal parties (TA.BAD_INSIDER, TA.LEGAL_HOST, TA.PREV_NEW_INSIDER, TA.INVADER) may encapsulate a malicious protocol within a given protocol that bypasses the security policy of the protected network's firewall. This malicious protocol would then be accepted by the receiving program in the protected network that could subsequently pose as a source of an attack (A.SOCIAL_ENGG, A.MALCODE, A.SQL_INJECT, A.FORMAT_STR)
T.BAD_COMMAND	Trusted principal parties (TA.INSIDER) accidentally issue a bad command that may pose as an attack (A.ERROR) resulting in the disruption, modification of the service parameters
T.SPOOF	Malicious and non-malicious principal parties (TA.BAD_INSIDER, TA.LEGAL_HOST, TA.PREV_NEW_INSIDER, TA.INVADER) do an attack (A.SPOOF, A.SOCIAL_ENGG) to obtain sensitive information
T.REPUDIATE	Authorized and trusted principal parties (TA.INSIDER, TA.LEGAL_HOST) deny their participation in a multicast session due to an attack (A.REPLAY, A.DOS, A.SNIFF, A.SPOOF, A.SOCIAL_ENGG,

Threat label	Description
	A.MALCODE) that results in disputes with the suspected principals
T.DOS	Malicious principal parties (TA.BAD_INSIDER, TA.PREV_NEW_INSIDER, TA.INVADER) do an attack (A.DOS) to temporarily halt the service to the legitimate subscribers
T.CALAMITY	A natural disaster (TA.NATURE) causing disruption in service to the legitimate subscribers due to (A.NATURE) type of attacks
T.ERROR_RECORD	Trusted principal parties (TA.INSIDER) do an attack (A.BUFFER_FLOW, A.ERROR) that was later on ignored, undetected, and therefore the threat remains unaddressed
T.VIRUS	Principal parties (TA.INSIDER, TA.BAD_INSIDER, TA.LEGAL_HOST, TA.PREV_NEW_INSIDER, TA.INVADER) spread a virus/worm to the principal machines causing data corruption

5.3.3 Administrative security policies

This section describes the administrative security policies that define in broader context, the organizational support and governance to maintain the safety/security of the TOE or its operating environment. The following administrative security policies are specified in Table 12.

Table 12: Administrative security policies

Policy Label	Description
P.MONITOR	The administrating body shall monitor security events to ensure conformity with security policies in place
P.CONFIG	The administrating body shall have managerial and operational security controls necessary to configure protocols during their operations. The configuration of the framework while its protocols are in operation should not jeopardize the security of the framework
P.AWARE	The administrating body shall have in place policies, personnel training sessions, and reporting and enforcement mechanisms such that agents know their security role in the in place framework
P.ACCOUNT	The administrating body shall define the revenue collecting type: precise or approximate, preplanned or on-demand accounting, etc
P.DOCUMENT	The administrating body shall provide detailed documentation of the framework and its underlying protocols to understand in depth the security infrastructure offered by the framework
P.RISK	The administering body shall follow up with necessary risk management policies necessary to counter any failure of the system components. Adaptive approach covering the identification, assessment and treatment of new or existing vulnerabilities in accordance with risk management policy for e-transactions

Policy Label	Description
P.CALAMITY	The administering body shall have the necessary in place operational measures for business continuity in case of natural calamity

5.4 Risk categories for the TOE

Each risk to a principal party or any computing device could be categorized by assessing sources of threats and sources of vulnerability that we have discussed in section 5.3.2.3 and section 5.3.2.2 respectively. The categories of security risks relevant to the TOE are defined as in Table 13.

Table 13: Identified risk categories for the TOE

Risk category label	Description	Threat	Vulnerability
R.FAIR_HOST	Risks associated with the subscribed users	T.CONFIDENTIAL, T.BAD_COMMAND, T.REPUDIATE, T.VIRUS, T.ERROR_RECORD	V.ARCH_DES, V.AUTH_PROT, V.PAY_PROT, V.POLICIES
R.OUTSIDER	Risks associated with new sub-scribers or unsubscribed users	T.MAL_ANALYSIS, T.MAL_MODIFY, T.CONFIDENTIAL, T.SPOOF, T.DOS, T.VIRUS	V.ARCH_DES, V.AUTH_PROT, V.PAY_PROT, V.MIDDLEWARE, V.SERV_PARAM, V.POLICIES
R.SELLER	Risks associated with the merchant or	T.MAL_ANALYSIS, T.MAL_MODIFY,	V.ARCH_DES, V.AUTH_PROT,

Risk category label	Description	Threat	Vulnerability
	merchant's content provider	T.CONFIDENTIAL, T.BAD_COMMAND, T.SPOOF, T.REPUDIATE, T.CALAMITY, T.ERROR_RECORD, T.VIRUS	V.PAY_PROT, V.MIDDLEWARE, V.SERV_PARAM, V.POLICIES, V.SPOF
R.SECPOLICY	Risks associated with the security policies in-place and with the new ones as they evolve by appending, replacing or modifying existing policies	T.CONFIDENTIAL, T.BAD_COMMAND, T.REPUDIATE, T.ERROR_RECORD, T.VIRUS	V.ARCH_DES, V.PAY_PROT, V.MIDDLEWARE, V.SERV_PARAM, V.POLICIES, V.SPOF
R.UNAVAILABLE	Risks associated with unavailability of service resources	T.CONFIDENTIAL, T.BAD_COMMAND, T.SPOOF, T.REPUDIATE, T.DOS, T.CALAMITY, T.ERROR_RECORD, T.VIRUS	V.ARCH_DES, V.AUTH_PROT, V.PAY_PROT, V.MIDDLEWARE, V.SERV_PARAM, V.POLICIES, V.SPOF
R.UNFAIR_CHARGE	Risks associated with the payment principles for the subscribed	T.MAL_ANALYSIS, T.MAL_MODIFY, T.CONFIDENTIAL, T.BAD_COMMAND,	V.ARCH_DES, V.PAY_PROT, V.SERV_PARAM, V.POLICIES, V.SPOF

Risk category label	Description	Threat	Vulnerability
	sessions such as collection of revenue for un-available service	T.SPOOF, T.REPUDIATE, T.DOS, T.CALAMITY, T.ERROR_RECORD, T.VIRUS	
R.CALAMITY	Risks associated with the cause of natural disaster such as an earth-quake, floods, fire	T.CALAMITY	V.ARCH_DES, V.POLICIES
R.MAL_SERV	Risks associated with the illegal use, modification or de-struction of service provider’s service	T.MAL_ANALYSIS, T.MAL_MODIFY, T.CONFIDENTIAL, T.BAD_COMMAND, T.SPOOF, T.ERROR_RECORD, T.VIRUS	V.AUTH_PROT, V.PAY_PROT, V.MIDDLEWARE, V.SERV_PARAM, V.POLICIES, V.SPOF
R.PRIVACY	Risks associated with the threat to privacy, message confidentiality	T.MAL_ANALYSIS, T.SPOOF, T.CONFIDENTIAL	V.ARCH_DES, V.AUTH_PROT, V.MIDDLEWARE, V.SERV_PARAM, V.POLICIES, V.SPOF
R.COMPONENTS	Risks associated with the trusted protocols or components mis-	T.CONFIDENTIAL, T.REPUDIATE	V.ARCH_DES, V.AUTH_PROT, V.PAY_PROT, V.SERV_PARAM,

Risk category label	Description	Threat	Vulnerability
	behavior in the framework		V.POLICIES

5.5 Security objectives for the TOE

The security objectives are intended to provide shielding against the security threats, attack sources, and vulnerability sources that arise due to breaches in the TOE or its operational environment. The security objectives for the TOE are described in Table 14.

Table 14: Security objectives for the TOE

Objective label	Description
O.PHYSICAL	The access to and from the trusted devices must be bounded and shall be free of unauthorized logins
O.BACKUP	The TOE must include provisions for multicast session’s data and control signals to possess the capabilities for timely recovery to an operating state if the session is compromised or damaged in transactions
O.RISK_ANALYSIS	The TOE shall perform multicast session’s security risk analysis for random transactions to evaluate the future continuity of e-commerce
O.AUTEHNTICITY	The TOE shall authenticate the principal parties and devices that

Objective label	Description
	are essential for continuity of e-commerce
O.CONFIDENTIALITY	The TOE protocols shall protect the confidentiality of information of subscriber, merchant, or any other principal’s asset
O.PRIVACY	The TOE protocols shall protect the privacy of information of subscriber, merchant, or any other principal’s asset
O.COMPLY	The administrating body, underlying protocols and computing systems shall comply with the International regulations, governing mandates, policies and controls that govern the deployment of the designed framework and its underlying protocols. This ensures the safety of the system and its operators
O.AUTHORIZE	The TOE shall have the policies in-place to authorize all the principal parties involved in the e-commerce transactions as well as it must have flexibility to authorize new components as they evolve (due to change in administrative policies)
O.AVAILABILITY	The TOE shall have the necessary protocols that ensure that the concerned principal parties have received the session keys that dictate the access to the multicast services
O.INTEGRITY	The TOE shall have the necessary protocols that protect from or detect corruption of the distributed keys that ultimately dictate the

Objective label	Description
	state of the e-commerce transaction
O.ACCOUNTABILITY	The TOE shall have necessary protocols to make sure that the subscriber is charged only if he receives the service
O.UPGRADE	Mechanism to anticipate the network growth and plan upgrades to increase the multicast service components such as number of routing devices, ports
O.SCALABILITY	The TOE shall have the necessary infrastructure and protocols for reducing the signaling overhead among various principals in an e-commerce transaction

Chapter 6

Proposed SETMS framework

We present our proposed Secure E-Commerce Transactions for Multicast Services (SETMS) framework that secures the e-commerce sessions for multicast environments. The framework provides the following features:

- Dynamic subscription of the host to multicast services
- Reliable authentication of all principals that are involved in an e-commerce transaction
- Secure e-payments and user identity
- Management of e-payments
- Authorization based on service policies
- Accounting for the subscriber's consumed resources
- Non-repudiation of principals involved in an e-transaction
- Opportunity for new merchants to participate in e-commerce
- Wide-scale interoperability, adaptability, acceptability and scalability of the distributed multicast services

In the following sections, we will describe the SETMS assumptions, system components and the SETMS system operational infrastructure that secures e-transactions for multicast environments.

6.1 SETMS assumptions

The SETMS framework assumes the availability of standard crypto algorithms and protocols, but it does not stipulate the use of any specific algorithm or protocol. We assume that the host identity details of the subscriber that are available at content providers are not a possible source of threats to e-commerce transactions. The subscriber's e-payment method is assumed to be a credit card in the "Access Control Processes" section 6.3.2. However, e-Cash based, e-Cheque based, micropayments based or Smartcard based e-payment platforms could also be used with the SETMS framework. The integrity of messages exchanged between merchant's content provider and payment gateway is assumed to be well protected by unicast protocols. The framework is assumed not to make any discretionary assumptions concerning the services offered by the network or the communication protocol, i.e., it should be possible to implement this framework on any network that supports "best-effort" multicast services.

6.2 SETMS system components

The SETMS framework is made up of the following components: 1) Subscriber, 2) AAA server, 3) Merchant's content provider, 4) Policy server, 5) Payment gateway, and 6) E-commerce trust model components. Each of these components is explained in the following subsections. The architecture of the SETMS framework is as shown in Figure 8.

6.2.1 Subscriber

A subscriber is an end user (card holder) who has requested a multicast service after agreeing with the service terms put forth by the merchant. As we have already stated in the assumptions section 6.1 that the subscriber's e-payment method would be through a credit card, we presume from now on that the subscriber is fundamentally required to possess a credit card to gain access to e-commerce services. A credit card can be obtained from an 'Issuer', which is typically an authorized financial institution that issues credit cards to its customers. The possession of these cards allows customers to participate in electronic transactions.

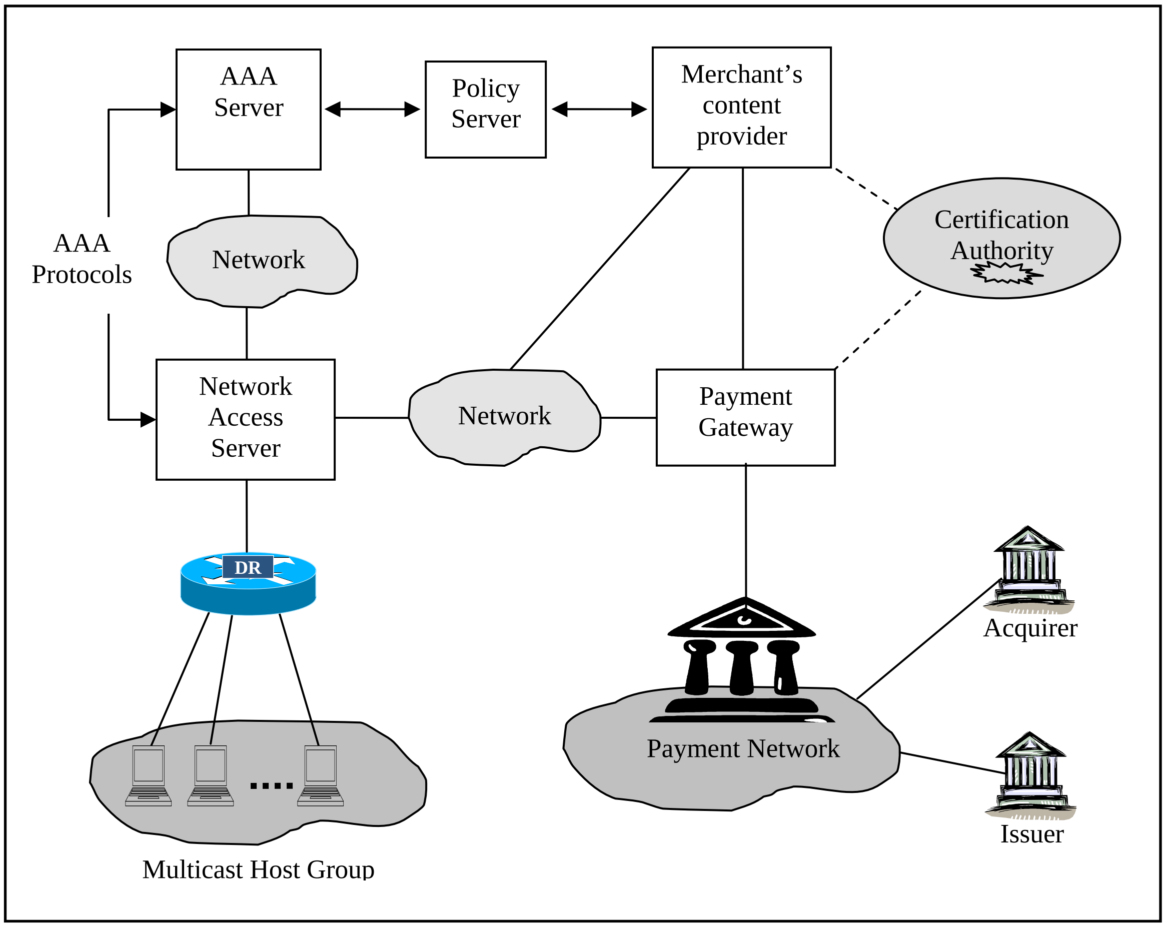


Figure 8: The SETMS system architecture

6.2.2 AAA server

The AAA server is a server that provides distributed authentication, authorization and accounting services to end-host's sessions. It also maintains a database for storing the details of legitimate subscribers. It has distributed AAA clients namely, NAS for providing the functionality of AAA services. The interface between the AAA server and the NAS is secured through AAA protocols such as Diameter [71], RADIUS [69], or TACACS [70]. These protocols are capable of carrying the responses of end hosts to the NAS. The attributes could be such as a challenge from the NAS, a response from a host, digital certificates, the host's public key. Additionally, AAA protocols must support selective encryption of attributes within a message for securing host's identification details. More on AAA protocols, their attribute-value representation of AAA data and their bit lengths can be found in [23].

6.2.3 Merchant's content provider

Merchant's Content Provider (MCP) is a server that provides services of a merchant to its subscribers. It could be solely operated by the merchant himself or authorize a content provider to advertise his services. It also validates the authorization requests of its subscribers. In most occasions, it may store subscriber's identification details to track the usage pattern of its services. It defines the service policies for each of its multicast applications. These policies dictate the access control privileges of its subscribers.

6.2.4 Policy server

The Policy Server (PS) is a server that stores the service parameters of each merchant. It accepts the attribute-value pairs sent by the MCP, denoting the multicast service parameters of its merchants. It provides for systematic representation of merchant’s service parameters. The PS decentralizes the content provider’s authorization procedure enabling multicast services scalability under heavy loads. This decentralized procedure is discussed later in the “Authorization Process” section of this chapter. A generic interface of the PS with content providers and AAA clients is shown in Figure 9. The PS behaves as an intermediate component that could do the necessary transformations to represent the service parameters in a uniform pattern, which could be readily accepted by the AAA server. The PS needs administering personnel to monitor the necessary transformations.

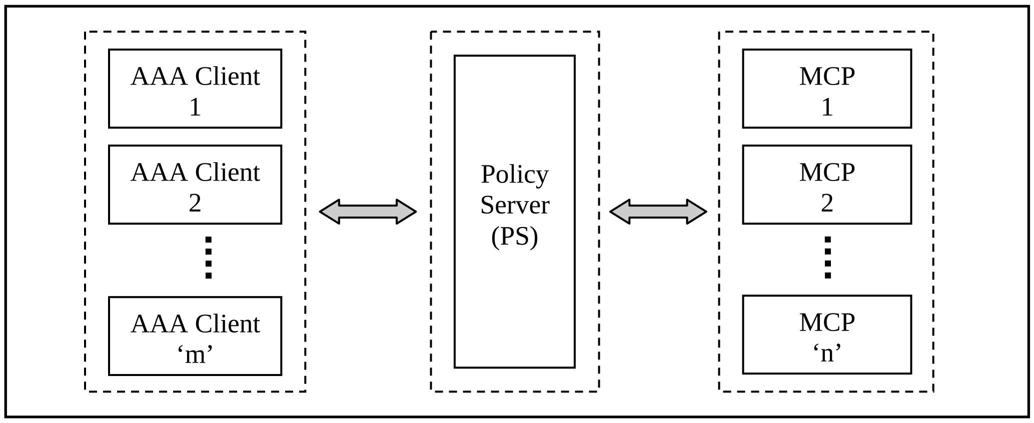


Figure 9: Generic interface of PS with AAA clients and content providers

Hence, the PS behaves as a filter of unnecessary attribute-values and forwarder of essential service parameters of multicast groups to the AAA server. It also acts as a mutual authenticator between the AAA server and the MCP; thus eliminating the probability of fake merchants posing as genuine content providers.

6.2.5 Payment gateway

The Payment Gateway (PG) is a gateway that resides between the MCP and the acquiring bank of the merchant, namely Acquirer. The PG has access to the issuer and the acquirer via secured payment networks, which is the hub for financial institutions. In e-commerce, it is used for authenticating the merchants, and validating the financial credentials of subscribers.

6.2.6 PKI trust model components

In e-commerce, a merchant and his subscriber are unlikely to trust each other. The subscriber would be reluctant to give access to his payment details to his merchant, and similarly the merchant does not want to provide his service to unauthorized subscribers. This problem has motivated us to build a Public Key Infrastructure (PKI) [25] based on trust components generally referred as on-line trusted third party (TTP) components. PKI provides e-commerce security through the use of digital certificates and public key cryptography. PKI supports security features such as privacy, data integrity, access control, confidentiality, and non-repudiation. The trust components of PKI include certification authority, registration authority, certificate revocation list, online certificate status protocol, and certificate policy and certificate practice statement. Each of these trust components is explained in the following section.

- *Certification authority:* In a business, two entities can communicate securely by trusting a third party called a Certification Authority (CA), which validates and guarantees their identity. A CA is an important component of PKI since it is the only component that has authority to create, revoke and manage public key certificates of end users and computing systems. Public key certificates are digitally signed by the CA and distributed to principals only after verifying their identity. Digital certificates are in X.509v3 certificate format containing fields such as serial number, signature, issuer, and validity period. The CA operates using industry accepted standards for managing keys and certificates so that the trust in e-commerce is maintained. More on the role of the CA can be found in [24, 25].

- *Registration authority:* The Registration Authority (RA) is a widely deployed optional component used for registering users through an organization official or by electronic signing. Users request certificates through the RA, which directs them to the CA for signing. Simply stated, an RA is an intermediary that handles administrative tasks of certificate management such as subject identification. The RA verifies the end entity's information by checking the user's ID and other professional credentials. When the RA is satisfied with the user details, they do the policy enforcement by generating end user keys and issuing tokens. This process significantly reduces the risk of approving certificates to unauthorized parties.
- *Certificate revocation list:* When a Public Key Certificate (PKC) is issued, it is expected to be in use for its entire validity period. However, various circumstances may cause a PKC to become invalid prior to the expiration of the validity period. Under such circumstances, the CA may need to revoke the PKC. X.509v3 defines a method for revoking PKC. This method involves each CA periodically issuing a signed data structure called a Certificate Revocation List (CRL). A CRL is a list that identifies the references of revoked PKC. This list contains a date of issue and the signature of CA and is made freely available in a public repository. Each revoked PKC is identified in a CRL by its PKC serial number. A CA issues a new CRL on a regular periodic basis (e.g., hourly, daily, or weekly). More on CRLs can be found in [24].
- *Online certificate status protocol:* The Online Certificate Status Protocol (OCSP) is an optional PKI component. OCSP semantics were designed to enable migration from CRL.

Like CRL, OCSP responses are signed and can be saved along with validated messages to enable revalidation at a future point in time. OCSP queries determine whether one or more individual certificates have been revoked or suspended. It explicitly excludes validation of the certificate's signature, timeliness, or path-level validation.

- *Certificate policy statement:* A Certificate Policy (CP), as defined in X.509, is a named set of rules that indicates the applicability of a certificate to a particular community and/or class of application with common security requirements. In any organization that operates a PKI, a Policy Authority is required to select or develop and maintain a CP.
- *Certification practice statement:* A Certification Practice Statement (CPS) is a statement of the practices that a CA employs in managing the certificates that it issues. A CPS details in specific terms the procedures, formats and processes that are in place to ensure the integrity of the PKI. The Operating Authority is responsible for preparing and maintaining the CPS.

Although CP and a CPS may seem similar, however each addresses different requirements both within an organization and outside. The IETF Networking Group has created an informational draft in RFC 2527 to detail issues related to CP and CPS as in [28]. Although RFC 2527 does not provide a standard statement, it does help promote best practices among PKI members.

6.3 SETMS system operational infrastructure

It is impossible to deploy multicast services in an e-commerce environment unless proper authentication, access control and accounting mechanisms are enforced for multicast services. If any flaw persists in these mechanisms, this could lead to privacy theft, confidentiality violation, network congestion, no flow control and mass losses in terms of financial property. Overall, the scalability and reliability of the multicast services will be limited. Therefore, we describe the SETMS operational infrastructure, which supports authentication, access control, accounting, e-payments, message integrity, confidentiality and non-repudiation. Each of these procedures is described in the following sections.

6.3.1 Authentication processes

Authentication has different meanings in different contexts. For the SETMS framework, authentication can be classified broadly into identity authentication and message content verification.

6.3.1.1 Identity authentication

Identity authentication is a protocol whereby some principals can prove their identities to each other. It ensures that no principal can impersonate another principal. Identity authentication can be further classified into host authentication and user authentication. Host authentication is required to allow only the identified hosts to have access to the multicast traffic while user authentication is required to determine if a principal has the privilege to participate in an e-commerce transaction. In the SETMS framework, there are specific principals involved in each authentication process. There should be some authentication process in each of the following pairs represented simply as (subscriber, AAA server), (subscriber, MCP), (MCP, PS), and (PS, AAA server).

The authentication process in (subscriber, AAA server) could be supported by the Host Identification Protocol (HIP) [2]. The position of HIP in the IP stack is depicted in Figure 10. HIP could be used for supporting the mutual authentication process between subscriber and the AAA server. Firstly, the subscriber processes must request the DNS server to issue a Host Identifier (HI), a cryptographic public key. The DNS server must process the request and issue an HI to the host. Once the host receives the HI, the host processes in its operating system would compute a hash of HI and generate a 128-bit HIT_S. This process allows the host machine to use HIT_S as its host identifier in IPv6 sized address structures (optionally, a 32-bit LSI could be used as the HI in IPv4 sized address structures) and its IP address is used only for locating the host. Initially, the host sends an MLDv2 [3] Start Listening message (optionally, IGMP Join for IPv4 networks) request to the nearest multicast capable router. This router forwards the HIT_S details of the host to the AAA client, NAS. HIP facilitates a 4-way handshake between the NAS and end-host to set up an HIP Security Association (SA), which involves exchange of Diffie-Hellman keys [4].

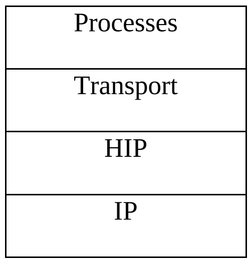
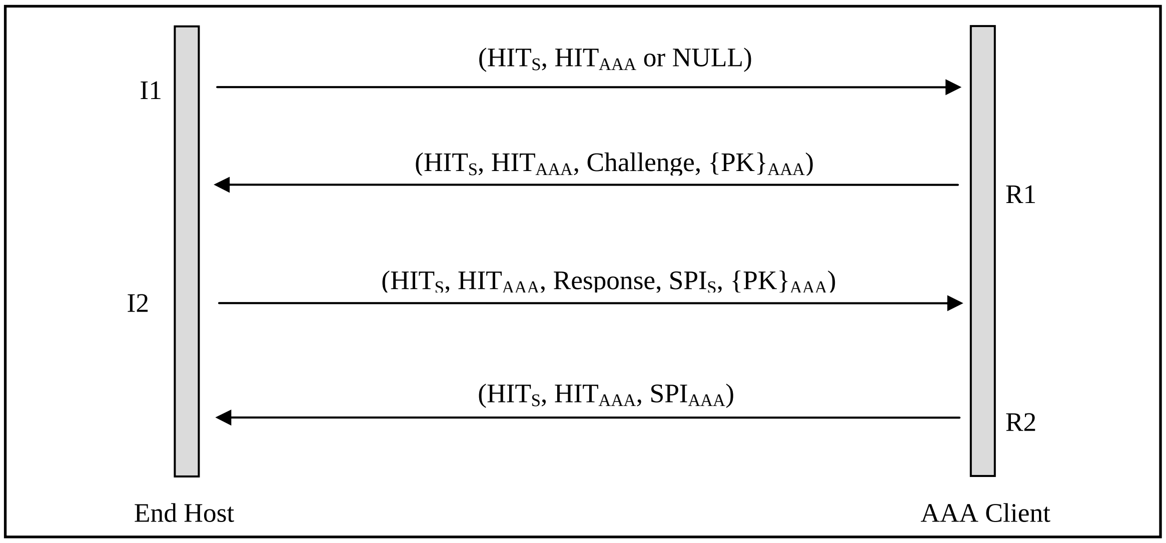


Figure 10: Protocols layer stack

The key exchange process involves mutual authentication of the AAA server and end-host, which is illustrated in Figure 11. The communication link could be protected by IPSec ESP for sending the key details of the data decryption as specified in [9]. The AAA server starts the Diffie-Hellman key exchange process by sending its public key, $\{PK\}_{AAA}$, and its HIT_{AAA} along with a Challenge to the host to prove its authenticity. The Challenge could be in the form of PAP, S/Key, CHAP or EAP. The host solves this Challenge and sends his Response Code along with an IPSec Security Parameter Index (SPI_S) value and HIT_S to the AAA server. The AAA server validates the host's Response, thereby authenticating the end-host and creating an HIP SA. The AAA server replies with its SPI_{AAA} value to the subscriber. Once HIT_S of the host is registered and stored at the AAA server, the host authentication process completes.



I1
(HIT_S, HIT_{AAA} or NULL)
(HIT_S, HIT_{AAA}, Challenge, $\{PK\}_{AAA}$)
R1
I2
(HIT_S, HIT_{AAA}, Response, SPI_S, $\{PK\}_{AAA}$)
(HIT_S, HIT_{AAA}, SPI_{AAA})
R2
End Host
AAA Client

Figure 11: Host authentication process using Diffie-Hellman key exchange

The added advantage of using the HIP protocol is that HIT_S value that is stored at AAA server could be used as a challenge to re-authenticate the host. To support this kind of host authentication, the host's Operating System (OS) and NAS OS must support HIP. The (subscriber, MCP) pair involves in the authentication of the host (HIT_S) of the subscriber at the MCP. The HIT_S identification at the MCP is explained in detail in the "Access Control Processes" section 6.3.2. The (MCP, PS) pair involves in authenticating merchant's data that the PS receives. The MCP and PS could communicate securely using IPSec protocol [7]. IPSec provides security at the network layer. It allows each of the two communicating systems to create a Security Association (SA) at each end for mutual identification. The data transport between both these parties is reliable as the transport protocol is TCP, which guarantees the integrity of the messages. The Authentication Header (AH) [10] and Encapsulating Security Payload (ESP) [13] fields could be inserted after IP header in IPv6 address structure packet for providing message integrity, authentication and confidentiality. AH provides message integrity and source authentication while ESP provides confidentiality of the data. Similarly, the (PS, AAA server) pair can communicate securely using the IPSec protocol.

The subscriber is automatically authenticated in the process of payment authorization. Since the authorization details contain the user identity details such as the subscriber's name, address, and credit card details ($CC\#_S$, optional PIN), MCP can authenticate the user once the payment authorization is validated. More on the subscriber authentication process is discussed in the following section on the payment authorization process.

6.3.1.2 Message content verification

Message content verification is a protocol whereby the content of the message is verified in order to detect if the message has been altered while in transit. This property is generally referred to as the integrity check property. In the SETMS framework, we will discuss the message integrity check procedure for the subscriber, merchant pair, which exchanges the e-payment details. Message integrity can be preserved by using a public key cryptography computation for encrypting the original message. However, encrypting the whole message using a public key will degrade performance as it adds lot of payload to the message header file. Therefore, using digital signatures will reduce the computational payload and offer the necessary integrity of message, which is explained as follows. Digital signature consists of Message Digest (MD) algorithm and Digital Certificate for providing message integrity check. The subscriber's payment details are protected using a symmetric key encryption algorithm of the Message Digest. MD consists of

Hash function, which computes a one-way hash of the message converted into a string of digits. This is followed by signing the hash (instead of signing whole message) using the sender's public key, thus improving computation speed significantly. MD5, SHA-1 and HMAC are some of the popular examples of this category.

6.3.2 Access control processes

In the SETMS framework, we use processes in two steps composed of e-payment authorization process and service delegation process. In first step, we describe the processes involved in authorizing the e-payment details of subscribers (and authentication of host). In second step, we discuss processes involving the description, formatting and sharing of the service parameters between the MCP, PS and AAA server.

6.3.2.1 E-payment authorization processes

We define the triple (subscriber, MCP, PG) to describe the e-payment authorization processes for securing e-commerce payment transactions. We have relied on the concepts of 2KP e-payment protocol [29] for validating the subscriber's e-payments. Specifically, SETMS employs the concepts of the 2KP protocol for enforcing the e-payment authorization processes. 2KP employs a moderate usage of digital signatures to authorize the subscriber's e-payments. A digital signature is an identifying code that can be used to authenticate the identity of the sender of a document, which cannot be easily repudiated or imitated, and can be time-stamped. The 2KP requires the merchant to possess a digital signature in addition to the digital signature of PG.

The e-payment authorization procedure in the SETMS framework is described as follows. At first, the subscriber visits the merchant's website and inspects the terms and conditions of service such as price for service duration, accepted payment types, and support for macro/micro payment. Once the subscriber agrees to the service terms and conditions, the host's software initiates the e-payment process by sending a service request to the merchant. The e-payment authorization process is depicted in Figure 12. The merchant validates the service request and sends a 'Clear' message back to the subscriber with PG's digital certificate (PG_{CERT}) and MCP's certificate (MCP_{CERT}).

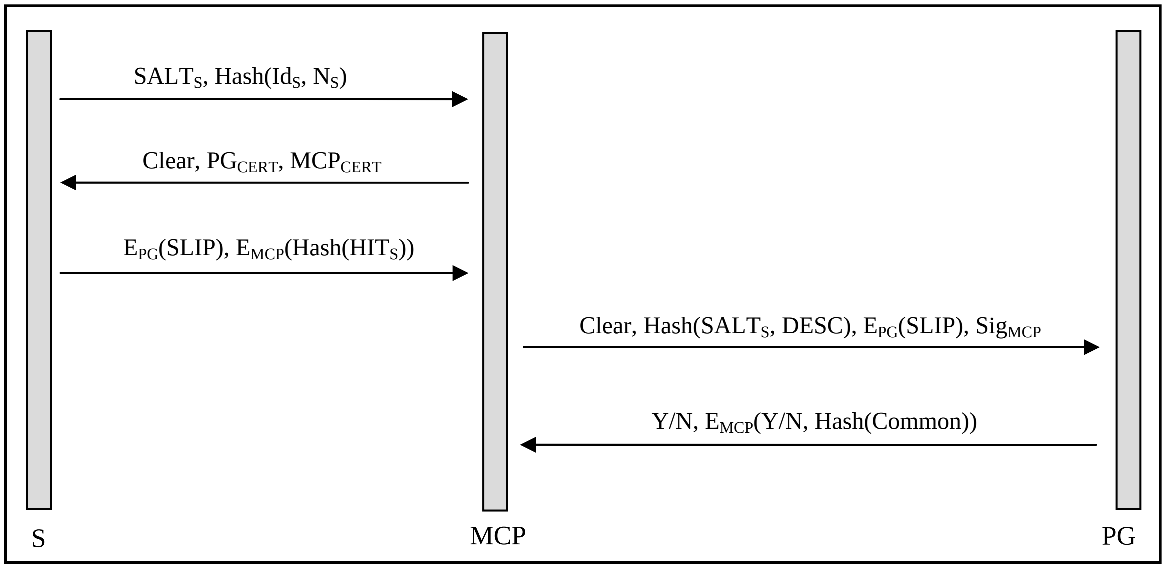

$SALT_S$, Hash(Id_S, N_S)
Clear, PG_{CERT}, MCP_{CERT}
E_{PG}(SLIP), E_{MCP}(Hash(HIT_S))
Clear, Hash($SALT_S$, DESC), E_{PG}(SLIP), Sig_{MCP}
Y/N, E_{MCP}(Y/N, Hash(Common))
S
MCP
PG

Figure 12: E-payment authorization in the SETMS framework

The host's software encrypts the subscriber's payment details using the PG's public key, attaches Hash(HIT_S) encrypted with Sig_{MCP} with the completed service form to it. Then using a hash algorithm such as MD5 and SHA-1, the host's software will generate a Message Digest and send the message back to the merchant. The merchant will verify the message integrity, and the merchant will decrypt the message (containing the Sig_{MCP}) using its private key, get Hash(HIT_S)

details of the subscriber and retrieve the subscriber's completed service form. The merchant then issues a payment Authorization-Request to the PG by forwarding the PG's encrypted payment details of the subscriber along with merchant's certificate, Sig_{CERT}. The PG performs the message integrity check against the merchant's certificate, MCP_{CERT} that it has received. Once the merchant is authenticated, the PG will forward the Authorization-Request to the Issuer via its interface to the secure banking network. The Issuer will respond with a 'Yes/No' message back to the PG. The PG will follow up by generating the Authorization-Response Code and encrypt it using its certificate, followed by sending the Authorization-Response Code to the merchant.

Where,

ID_S is the identification details of the subscriber containing the CC#.

DESC is the description of the multicast service.

$SALT_S$ is a random number generated by the subscriber, which offers protection against the dictionary attacks.

Hash() is a keyed one-way hash function such as the MD-5, SHA-1.

N_S is a nonce generated by the subscriber, which offers protection against replay attacks.

Sig_{MCP} is the signature of the MCP.

Sig_{PG} is the signature of the PG.

PG_{CERT} is the digital certificate of the PG.

E_{PG} is the encryption of the payment details using the PG's public key.

SLIP is concatenation of payment parameters such as price, Hash(Common), $CC\#_S$, R_S, expiry date, etc.

Common contains completed order form and Hash(HIT_S).

Auth-Resp = Y/N, Sig_{PG}(Y/N, Common) is the authorization response given by PG.

6.3.2.2 Service delegation processes

In the SETMS, the Authorization-Response, which is sent to the merchant, will not be forwarded to the subscriber as it will significantly degrade the scalability for multicast services. The services would scale to unlimited subscribers only if there is a mechanism that allows a subscriber to automatically gain access to the multicast traffic once the verification and validation of payment details is complete, without exchange of acknowledgement from the merchant. Therefore, there should be a decentralization of merchant's responsibilities to allow its subscribers to gain access to its services. The service delegation process is depicted in Figure 13 and it is explained as follows. In the SETMS, once the Authorization-Response of the PG is received by the MCP, it is securely forwarded from the MCP to PS and AAA server via the TCP/IP communication. We rely on digital signatures for mutual authentication of communicating principals. The MCP and PS mutually authenticate by exchanging their Digital Certificates, namely MCP_{CERT} and PS_{CERT} respectively.

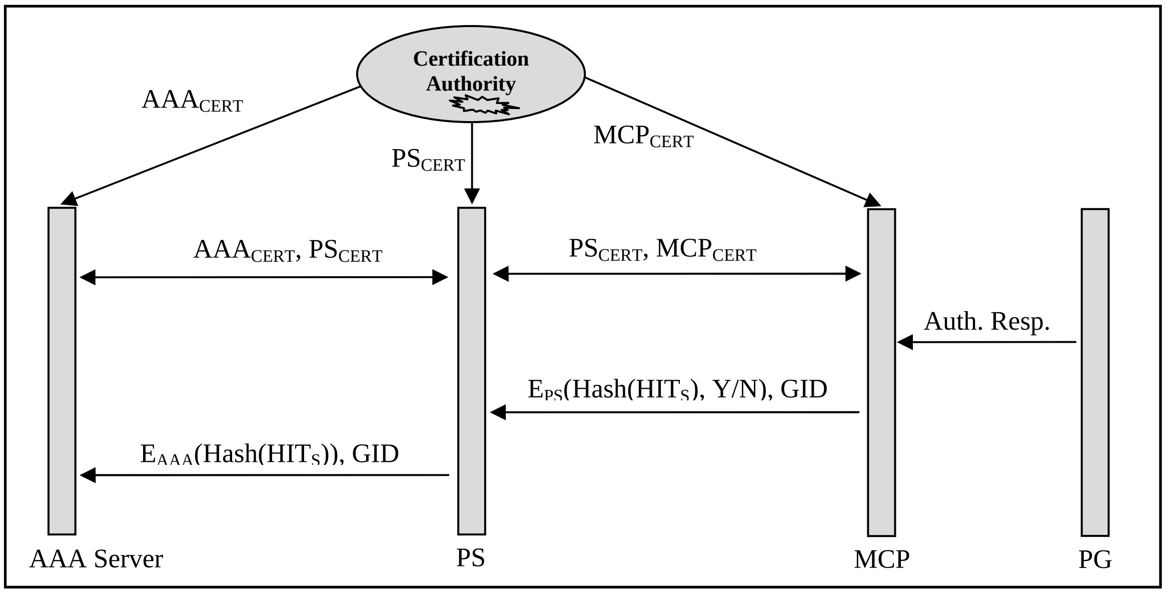


Figure 13: Secure service parameter exchanges through digital signatures

Similarly, the PS and AAA server authenticate each other by exchanging their digital certificates, namely PS_{CERT} and AAA_{CERT} respectively. The MCP has the sole rights to decide whether to authorize the subscriber to have access to its multicast group identified with a Group Identity (GID). The MCP will forward the GID with his Authorization-Response, “Yes/No” and Hash(HIT_S) encrypted using {PK_{PS}}. This process allows total security of the merchant’s authorization response to its subscriber. The PS will decrypt the message using its private key to validate the merchant’s authorization response.

If the authorization-response attribute is “No”, the PS simply ignores the message. If the merchant’s authorization response attribute is “Yes”, the PS will encrypt the Hash(HIT_S) with the {PK_{AAA}} and forward it with the GID to the AAA server. The AAA server can decrypt this message to identify the value of the Hash(HIT_S) and cross-check within its Database to identify if there is any subscriber who has registered at its server with the Hash(HIT_S) value that it received from the PS.

6.3.3 Distributed accounting processes

Accounting for resource consumption involves assigning the multicast service parameters to the subscriber followed by session keys generation for each transaction. Once the payment authorization process is complete, the AAA server has the HIT_S of the subscriber of the multicast group. Accounting is carried out by the AAA server once the session parameters are exchanged by relying on the Diffie-Hellman key exchange procedure to generate session keys. The AAA server takes responsibility for delegating the service parameters to the authorized subscribers. It takes care of accounting and non-repudiating the subscriber by creating session logs for each transaction. The AAA server timestamps each and every session and its activity so that it can non-repudiate the subscriber for the multicast service it receives. The authorized subscribers will receive the decryption key via the IPSec ESP. The rekeying is the responsibility of the multicast routers and is out of the scope of the discussion.

The robustness of the SETMS framework refers to the ability of its accounting process to sustain operation under heavy loads. The framework should reliably account for all use and remain stable under varying situations. For instance, an accounting system that has a very high frequency of signaling with poor communication control may have its various components severed from each other at time of congestion; the various components may not be able to communicate properly and the system may produce unreliable results. The AAA server generates the necessary service parameters and sends them to the NAS to permit the subscriber's access to the multicast service identified by the GID. The AAA protocols must have mechanisms to account for the subscriber's active sessions by monitoring total session time, idle time, and total bytes transmitted and received. The NAS interacts with the router that is nearest to the subscriber. A message will be sent to the PS indicating the Hash(HIT_S) to which the access has been granted. Future service authorization is done through the validation of the subscriber's HIT_S, which is stored in the AAA server's database. The PS should contain the dynamic re-authentication and re-authorization timestamps that must be readily available to the AAA server to re-validate a multicast group subscriber. The AAA server timestamps each transaction to non-repudiate principals involved in the session's lifetime. In the SETMS, the PS has specific control messages that are dedicated to the AAA server to deliver negative authorization code of specific Hash(HIT_S) of a subscriber as shown in Figure 14. This process allows the AAA server to remove the invalid subscriber's host identity details from its database.

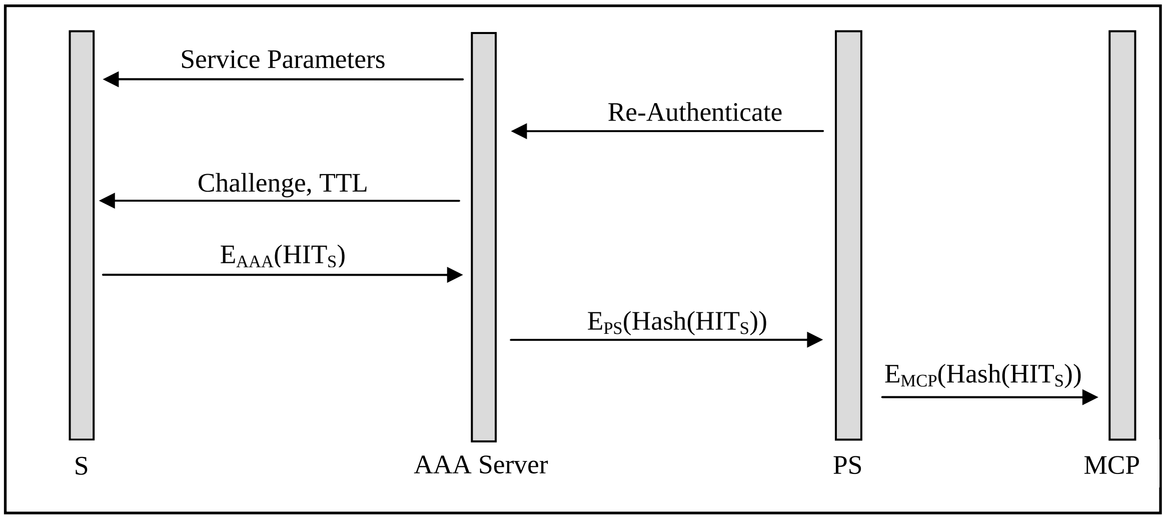


Figure 14: Accounting processes data and control messages

This procedure also helps to keep track of hosts who are misbehaving frequently so that measures could be taken by administering personnel to disable his future request for access to multicast service. Apart from this, the AAA server issues self-certified certificates to all the legitimate hosts when the payment authorization procedure is complete. In the future, when additional authentication is needed, the AAA server can just request the digital certificate it issued to its subscriber. The AAA server mentions the session life time as a service parameter that is passed to the NAS. The NAS will send a Re-Authenticate message (in the form of a Challenge) to the subscriber in order to validate that a host's session is still active. The subscriber sends its Challenge-Response (containing the identity of its host) to the NAS.

Once the NAS receives this response, it then sends an UPDATE message to the AAA server to notify it of the current status of the host. Now, it forms the responsibility of the AAA server to forward the identify details of the subscriber for which the re-authentication was done. Once the user has unsubscribed by sending a LEAVE message, the underlying protocols should release all session resources and terminate the service for the subscriber. The accounted information could be sent to the AAA server in the form of a batch file or as timestamps containing start and stop of the active session. There should be a mechanism for the NAS to send the accounting information at a later point in case the AAA server halts, thus preventing any loss of accounting packets. It should be noted that without the AAA server services, the PKI infrastructure cannot suffice to fulfill many security requirements of e-commerce transactions for multicast environments. The AAA protocols must offer integrity and confidentiality of the subscriber's credentials that are exchanged with the NAS.

Chapter 7

Evaluation and formal validation of SETMS

In this chapter, firstly we evaluate the SETMS framework by providing our arguments in support of it by making use of the support from the multicast and e-commerce literature as well as the MSPP, which is documented in chapter 5. We follow-up in the next section with a formal validation of the proposed SETMS protocol suite using the Automated Validation of Internet Security Protocols and Applications (AVISPA) tool [72], which uses the High Level Protocol Specification Language (HLPSL) [67] for the modeling of security-sensitive protocols.

A formal validation of the SETMS protocol is necessary as it is a well-known fact that any cryptographic design of a security protocol is always prone to errors. Customers trust in the e-commerce infrastructure for the multicast services will deteriorate if these security flaws and breaches are left open in the designed protocol. Also, trying to identify, eliminate or limit these flaws and breaches after deployment would be very costly both in terms of cost expenses as well as the customer's confidence in the modified protocol. Therefore, there is a strong need of a formal validation of the designed e-commerce protocol before its actual deployment to computer networks. The formal validation of the security protocol would identify, eliminate as well as bound the design flaws and breaches, allowing the protocol standardizing body to decide if the protocol is safe enough for the designated infrastructure.

7.1 Evaluation of SETMS framework

It is noted that the HIP protocol is resilient to DoS and man-in-the-middle attacks as stated in [2]. This argument is also supported by the TOE security environment that discusses the possible sources of attacks as discussed in Section 5.3.2.3. HIP also offers anonymity of the subscriber by making use of the unpublished HI as the subscriber's identity in e-commerce. This is also a required security feature of the TOE as discussed in Section 5.2.2. In SETMS, the PS has specific control messages that are dedicated to the AAA server to deliver negative authorization code of specific HIT_S of a subscriber as indicated in Figure 14. This process allows the AAA server to remove the invalid subscriber's host identity details from its Database. This procedure also helps to keep track of hosts that are misbehaving frequently so that measures could be taken by administering personnel to disable its future request for access to multicast service. This argument is supported by the TOE security environment that identifies the threat agents: TA.INSIDER, TA.LEGAL_HOST, and TA.PREV_NEW_INSIDER as described in Section 5.3.2.1.

SETMS provides an opportunity for new merchants to participate in e-commerce due to the fact that the PS can accommodate the distributed content provider's service parameters, even in the case where they have different patterns for representing their service parameters. To monitor the responsibility of the PSs role in pattern transformations, it would definitely require administering personnel. This argument is in compliance with the TOE administrative security policies, which we have defined in Section 5.3.3. Thus, the PS enables wide-scale interoperability, adaptability, acceptability and flexibility of the distributed multicast services. However, they have to be in compliance with the TOE administrative security policies as stated in Section 5.3.3.

The SETMS framework does not use a variant of the 1KP in the payment authorization procedure though the computational cost for establishing PKI infrastructure is observed to be minimal. The 1KP requires only the PG to possess the digital signature and to distribute authentic copies to the merchant and the subscriber. The problem with the usage of the 1KP is it only offers authentication of the subscriber and the merchant is left unauthenticated. This loophole protrudes threat agents such as TA.BADINSIDER as identified in Section 5.3.2.1. This kind of framework is favorable in situations where the subscriber is sure of the merchant's service and does not bother to explicitly identify who the merchant is. Whereas, authentication of messages in the SETMS protocol suite requires the subscriber to possess an authentic copy of the PG's public key certificate as well as merchant's public key certificate as in the 2KP. Thus, it offers non-repudiation of the principal parties involved in the transaction, which is one of the security features of the TOE as stated in Section 5.2.2. SETMS protocol does not rely on the 3KP kind of payment authorization procedure because it will put a huge constraint on the scalability of multicast services in e-commerce. This is because the 3KP requires each and every subscriber to request digital certificates from authorized CAs to participate in e-commerce. Though it non-repudiates all the principal parties, it lacks the essence of scalability in e-commerce as identified by O.MANAGE in Section 5.5 that states the security objectives of the TOE. In contrast, 2KP does not require complex computations and it is computationally faster and efficient, which is also a requirement of the TOE security environment as identified by V.PAY_PROT in Section 5.3.2.2.

In the SETMS framework, the AAA server initiates the Diffie-Hellman key exchange process by sending a Challenge to the subscriber. This process prevents an intruder's attempt to log in as a legitimate subscriber when the attacker fails a specified number of times in providing a correct Response to the Challenge. It could even lock the subscriber's account for a period of time to prevent further exploitation by the intruder. This action requires administering personnel to configure the server as required. This argument is in compliance with the TOE security environment that discusses the vulnerability sources, and administrative security policies as stated in Section 5.3.2.2 and Section 5.3.3 respectively. The AAA server timestamps each subscriber’s transaction as stated in the SETMS accounting process. This allows the framework to collect an irrefutable proof of the principal party’s participation in e-commerce transactions. It also prohibits the faulty claims of principal parties as discussed in the aspects of security features of the TOE in Section 5.2.2.

SETMS framework provides high scalability because once the e-payment authorization procedure completes, the merchant does not send an ACK to the subscriber indicating its authorization response, which would degrade the scalability of the framework when the volume of subscribers is huge. Instead, the MCP sends the details of only the authorized Hash(HIT_S) to the AAA server via PS. This kind of message flow will reduce the control messages between the merchant and the subscriber in the e-payment authorization and subsequent accounting processes. This is also identified as a security objective for the TOE as stated in Section 5.5. The SETMS framework offers flexibility in the e-payment mechanisms because the architectural framework can incorporate several kinds of payment types (not restricted to e-card based e-payments).

7.2 Formal validation tool AVISPA

The Automated Validation of Internet Security Protocols and Applications (AVISPA) tool provides a push-button, industrial-strength technology by making use of an expressive protocol specification language called the High Level Protocol Specification Language (HLPSL) [67]. Basically, the AVISPA tool takes an HLPSL protocol specification as an input and translates it into a corresponding Intermediate Format (IF) specification [75]. This is followed with a thorough analysis of the IF specifications by invoking state-of-the-art back-ends, which are currently On-the-Fly Model Checker (OFMC) [77], Constraint-Logic-based Attack Searcher (CL-AtSe) [76], SAT-based Model Checker (SATMC) [78], and Tree Automata-based Protocol Analyser (TA4SP) [81] as shown in Figure 15. These state-of-art back-ends return attacks (if any) to the protocol analyzer in an intuitive and readable output format. In particular, OFMC performs protocol falsification and bounded verification by exploring the transition system elaborated by an IF specification in a demand-driven way as stated in [34]. CL-AtSe administers constraint solving with powerful simplification heuristics and redundancy elimination techniques as specified in [34]. SATMC models a propositional formula, which is fed to the SAT solver and any model found is translated back into an attack. TA4SP uses the regular tree language and rewriting to approximate the intruder knowledge as specified in [81].

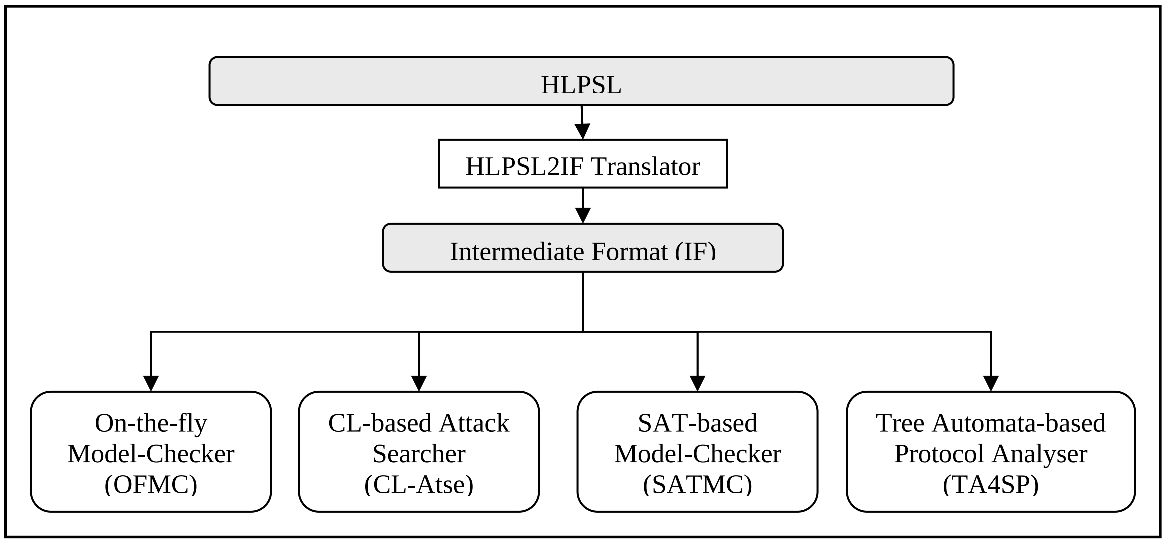
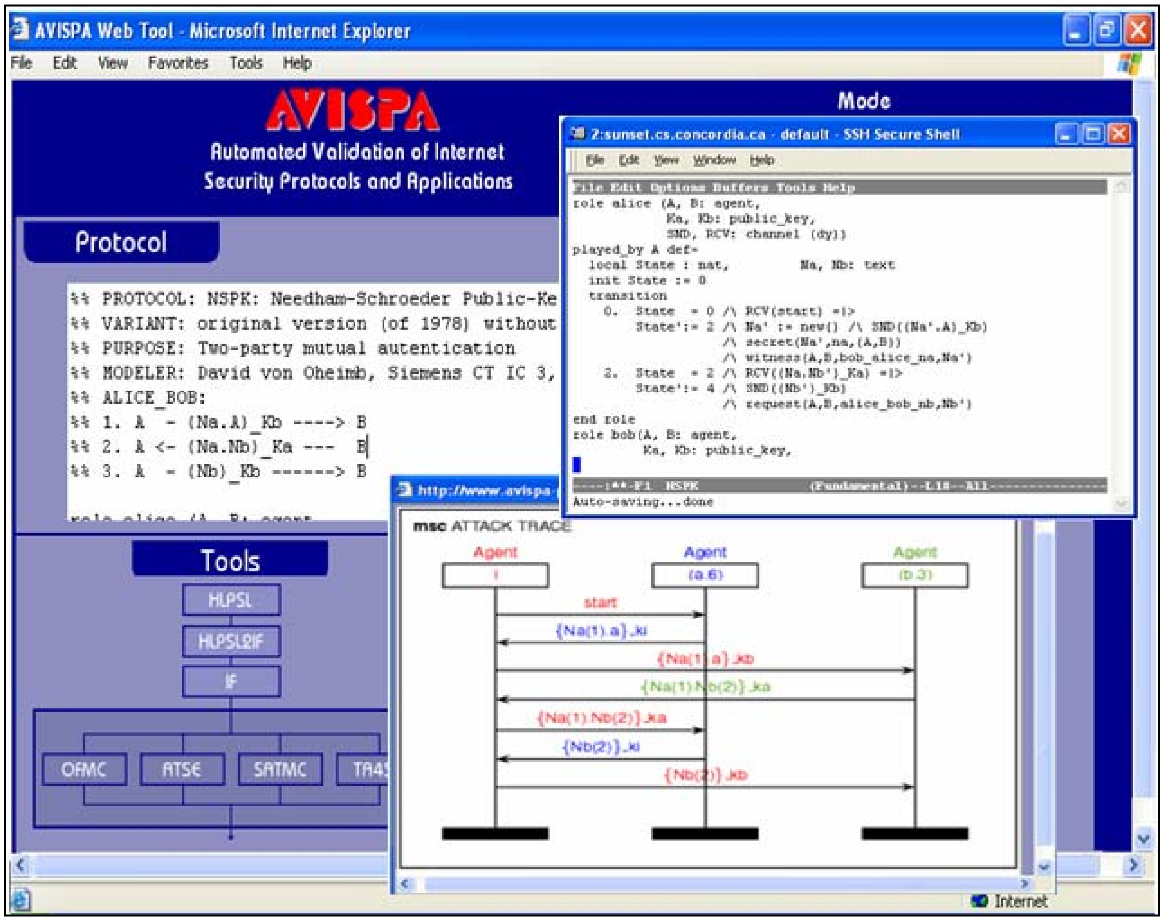

HLPSL
HLPSL2IF Translator
Intermediate Format (IF)
On-the-fly Model-Checker (OFMC)
CL-based Attack Searcher (CL-Atse)
SAT-based Model-Checker (SATMC)
Tree Automata-based Protocol Analyser (TA4SP)

Figure 15: Architecture of the AVISPA tool

HLPSL has a formal semantics based on Lamport’s Temporal Logic of Actions (TLA) [73]. Using HLPSL with TLA semantics to formalize security properties gives the protocol specifier a great generality and expressiveness. Although there are many other specification languages that can be used for protocol analysis such as Language Of Temporal Ordering Specification

(LOTOS) [79] or PROcess MEta Language (PROMELA), the input language to the Simple Promela INterpreter (SPIN) [80] model checker, they are generic modeling languages and lack domain-specific language semantics. For example, LOTOS or PROMELA lack the ability to explicitly specify different channel types (i.e., different intruder models). They also restrict the security goals to secrecy and authentication whereas HLPSL semantic logic facilitates the designer to model very general security goals such as entity and message authentication, replay protection, authorization by a TTP, key authentication, and confidentiality. Thus, HLPSL can be used to formalize protocols that are significantly more complex, both in terms of agent behavior as well as in terms of their intended security goals. The web-based version 1.0 of the AVISPA tool is illustrated in Figure 16. It illustrates the execution of HLPSL of the Needham-Schroeder Public-Key (NSPK) protocol using the AVISPA tool.


AVISPA Web Tool - Microsoft Internet Explorer
File Edit View Favorites Tools Help
AVISPA
Automated Validation of Internet Security Protocols and Applications
Mode
Protocol
Tools
HLPSL
HLPSL2IF
IF
OFMC
ATSE
SATMC
TA4
2:sunset.cs.concordia.ca - default - SSH Secure Shell
File Edit View Window Help
http://www.avispa
msc ATTACK TRACE
Agent
i
Agent
(a,6)
Agent
(b,3)
start
{Na(1).a}_ki
{Na(1).a}_kb
{Na(1).Nb(2)}_ka
{Na(1).Nb(2)}_ka
{Nb(2)}_ki
{Nb(2)}_kb
Internet

```
%% PROTOCOL: NSPK: Needham-Schroeder Public-Ke
%% VARIANT: original version (of 1978) without
%% PURPOSE: Two-party mutual autentication
%% MODELER: David von Oheimb, Siemens CT IC 3,
%% ALICE_BOB:
%% 1. A  - (Na.A)_Kb ----> B
%% 2. A <- (Na.Nb)_Ka ---  B
%% 3. A  - (Nb)_Kb ------> B
```

```
File Edit Options Buffers Tools Help
role alice (A, B: agent,
            Ka, Kb: public_key,
            SND, RCV: channel (dy))
played_by A def=
  local State : nat,         Na, Nb: text
  init State := 0
  transition
    0.  State  = 0 /\ RCV(start) =|>
        State':= 2 /\ Na' := new() /\ SND({Na'.A}_Kb)
                   /\ secret(Na',na,{A,B})
                   /\ witness(A,B,bob_alice_na,Na')
    2.  State  = 2 /\ RCV({Na.Nb'}_Ka) =|>
        State':= 4 /\ SND({Nb'}_Kb)
                   /\ request(A,B,alice_bob_nb,Nb')
end role
role bob(A, B: agent,
         Ka, Kb: public_key,

----:**-F1  NSPK            (Fundamental)--L18--All----------------
Auto-saving...done
```

Figure 16: Web-based version of the AVISPA tool

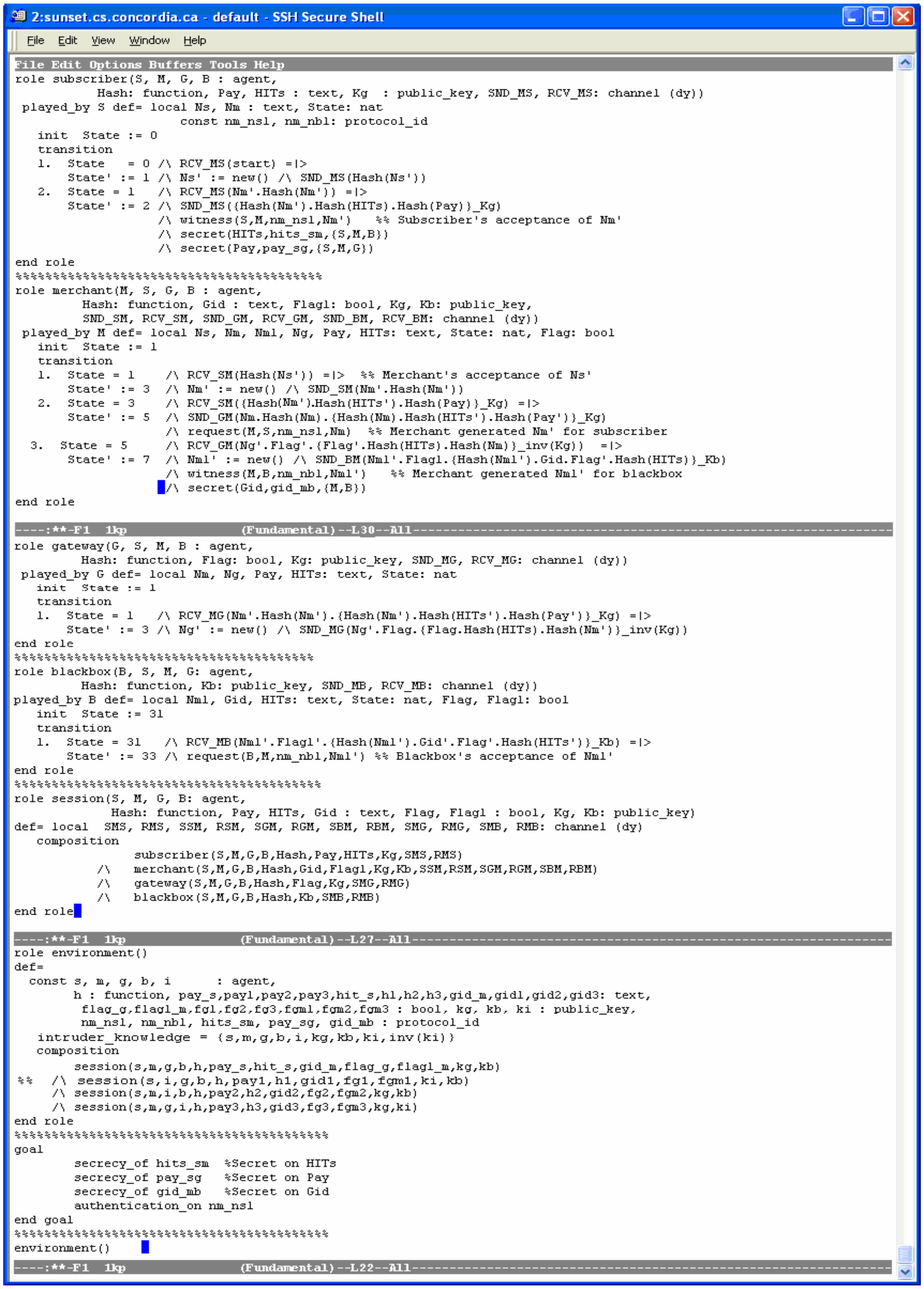
The attack encountered in the NSPK protocol run can also be seen in Figure 16. HLPSL is a well proven expressive and versatile language for modeling security protocols. It has already been widely used to model industrial protocols such as Encrypted Key Exchange (EKE), HIP, TESLA, SSH-transport, TLS, Kerberos, SET-purchase, AAA for Mobile IP as specified in [72].

At first, we will show why the SETMS protocol suite does not rely on the 1KP protocol by validating the protocol run of the 1KP variant for securing e-commerce transactions for multicast environments. It should be noted that this illustration is shown here, just to make sure that the 1KP cannot be used in the SETMS framework as it will lead to several attacks in the protocol run. We begin by representing the 1KP variant for multicast in Alice&Bob-style notation is as below.

- ALICE_BOB:

```
0. S -> M: Hash(Ns)
1. M -> S: Hash(Ns),Hash(Nm)
2. S -> M: {Hash(Nm), Hash(HITs), Hash(Pay)}_Kg
3. M -> G: Nm, {Hash(Nm), Hash(HITs), Hash(Pay)}_Kg
4. G -> M: Ng, Flag, {Flag, Hash(HITs), Hash(Nm)}_inv(Kg)
5. M -> B: Nm1, Flag1, {Hash(Nm1), Gid, Flag, Hash(HITs)}_Kb
```

In the above notation, the agents in the protocol run are S, M, G, and B, which refer to subscriber, merchant, payment gateway and black-box respectively. Black box is a secured module combining together the AAA server and the policy server functionality, and the communication link between the components inside the agent “B” is assumed to be well protected from intruders. The HLPSL for the 1KP variant of the SETMS protocol suite is as shown in Figure 17.



2:sunset.cs.concordia.ca - default - SSH Secure Shell

```
File Edit Options Buffers Tools Help
role subscriber(S, M, G, B : agent,
           Hash: function, Pay, HITs : text, Kg  : public_key, SND_MS, RCV_MS: channel (dy))
 played_by S def= local Ns, Nm : text, State: nat
                      const nm_nsl, nm_nbl: protocol_id
   init  State := 0
   transition
   1.  State   = 0 /\ RCV_MS(start) =|>
       State' := 1 /\ Ns' := new() /\ SND_MS(Hash(Ns'))
   2.  State = 1   /\ RCV_MS(Nm'.Hash(Nm')) =|>
       State' := 2 /\ SND_MS({Hash(Nm').Hash(HITs).Hash(Pay)}_Kg)
                   /\ witness(S,M,nm_nsl,Nm')   %% Subscriber's acceptance of Nm'
                   /\ secret(HITs,hits_sm,{S,M,B})
                   /\ secret(Pay,pay_sg,{S,M,G})
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role merchant(M, S, G, B : agent,
         Hash: function, Gid : text, Flagl: bool, Kg, Kb: public_key,
         SND_SM, RCV_SM, SND_GM, RCV_GM, SND_BM, RCV_BM: channel (dy))
 played_by M def= local Ns, Nm, Nml, Ng, Pay, HITs: text, State: nat, Flag: bool
   init  State := 1
   transition
   1.  State = 1    /\ RCV_SM(Hash(Ns')) =|>  %% Merchant's acceptance of Ns'
       State' := 3  /\ Nm' := new() /\ SND_SM(Nm'.Hash(Nm'))
   2.  State = 3    /\ RCV_SM({Hash(Nm').Hash(HITs').Hash(Pay)}_Kg) =|>
       State' := 5  /\ SND_GM(Nm.Hash(Nm).{Hash(Nm).Hash(HITs').Hash(Pay')}_Kg)
                    /\ request(M,S,nm_nsl,Nm)  %% Merchant generated Nm' for subscriber
  3.  State = 5     /\ RCV_GM(Ng'.Flag'.{Flag'.Hash(HITs).Hash(Nm)}_inv(Kg))  =|>
       State' := 7  /\ Nml' := new() /\ SND_BM(Nml'.Flagl.{Hash(Nml').Gid.Flag'.Hash(HITs)}_Kb)
                    /\ witness(M,B,nm_nbl,Nml')   %% Merchant generated Nml' for blackbox
                    /\ secret(Gid,gid_mb,{M,B})
end role

----:**-F1  1kp                (Fundamental)--L30--All-----------------------------------------------------------
role gateway(G, S, M, B : agent,
         Hash: function, Flag: bool, Kg: public_key, SND_MG, RCV_MG: channel (dy))
 played_by G def= local Nm, Ng, Pay, HITs: text, State: nat
   init  State := 1
   transition
   1.  State = 1   /\ RCV_MG(Nm'.Hash(Nm').{Hash(Nm').Hash(HITs').Hash(Pay')}_Kg) =|>
       State' := 3 /\ Ng' := new() /\ SND_MG(Ng'.Flag.{Flag.Hash(HITs).Hash(Nm')}_inv(Kg))
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role blackbox(B, S, M, G: agent,
         Hash: function, Kb: public_key, SND_MB, RCV_MB: channel (dy))
played_by B def= local Nml, Gid, HITs: text, State: nat, Flag, Flagl: bool
   init  State := 31
   transition
   1.  State = 31   /\ RCV_MB(Nml'.Flagl'.{Hash(Nml').Gid'.Flag'.Hash(HITs')}_Kb) =|>
       State' := 33 /\ request(B,M,nm_nbl,Nml') %% Blackbox's acceptance of Nml'
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role session(S, M, G, B: agent,
             Hash: function, Pay, HITs, Gid : text, Flag, Flagl : bool, Kg, Kb: public_key)
def= local  SMS, RMS, SSM, RSM, SGM, RGM, SBM, RBM, SMG, RMG, SMB, RMB: channel (dy)
   composition
                subscriber(S,M,G,B,Hash,Pay,HITs,Kg,SMS,RMS)
            /\   merchant(S,M,G,B,Hash,Gid,Flagl,Kg,Kb,SSM,RSM,SGM,RGM,SBM,RBM)
            /\   gateway(S,M,G,B,Hash,Flag,Kg,SMG,RMG)
            /\   blackbox(S,M,G,B,Hash,Kb,SMB,RMB)
end role

----:**-F1  1kp                (Fundamental)--L27--All-----------------------------------------------------------
role environment()
def=
  const s, m, g, b, i      : agent,
        h : function, pay_s,payl,pay2,pay3,hit_s,hl,h2,h3,gid_m,gidl,gid2,gid3: text,
         flag_g,flagl_m,fgl,fg2,fg3,fgml,fgm2,fgm3 : bool, kg, kb, ki : public_key,
         nm_nsl, nm_nbl, hits_sm, pay_sg, gid_mb : protocol_id
   intruder_knowledge = {s,m,g,b,i,kg,kb,ki,inv(ki)}
   composition
        session(s,m,g,b,h,pay_s,hit_s,gid_m,flag_g,flagl_m,kg,kb)
%%   /\  session(s,i,g,b,h,pay1,h1,gid1,fg1,fgm1,ki,kb)
     /\ session(s,m,i,b,h,pay2,h2,gid2,fg2,fgm2,kg,kb)
     /\ session(s,m,g,i,h,pay3,h3,gid3,fg3,fgm3,kg,ki)
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
goal
        secrecy_of hits_sm  %Secret on HITs
        secrecy_of pay_sg   %Secret on Pay
        secrecy_of gid_mb   %Secret on Gid
        authentication_on nm_nsl
end goal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
environment()
----:**-F1  1kp                (Fundamental)--L22--All-----------------------------------------------------------
```

Figure 17: HLPSL for the 1KP variant of the SETMS protocol suite

When this protocol was validated using the AVISPA tool, the following OFMC attack was discovered, as shown in Figure 18. The OFMC backend search time was found to be 0.58s and the number of visited nodes was 33 with a depth search of 3 piles. The Message Sequence Chart (MSC) in the Figure 18 illustrates that the intruder compromises the session by intercepting subscriber's messages, which were originally directed towards the merchant.

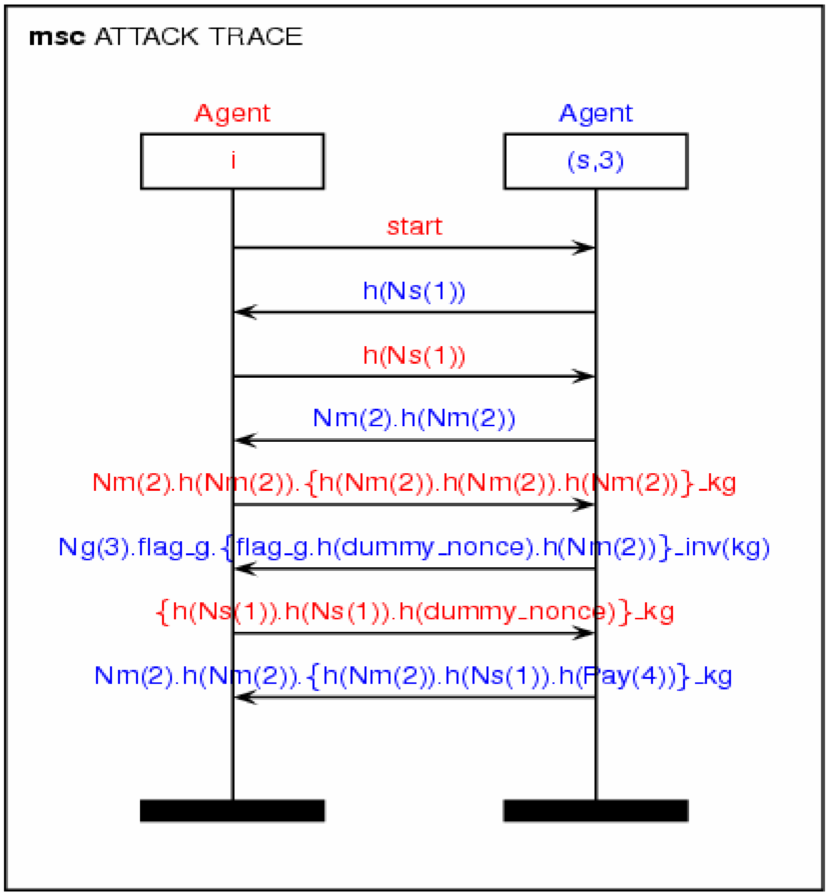

msc ATTACK TRACE
Agent
i
Agent
(s,3)
start
h(Ns(1))
h(Ns(1))
Nm(2).h(Nm(2))
Nm(2).h(Nm(2)).{h(Nm(2)).h(Nm(2)).h(Nm(2))}_kg
Ng(3).flag_g.{flag_g.h(dummy_nonce).h(Nm(2))}_inv(kg)
{h(Ns(1)).h(Ns(1)).h(dummy_nonce)}_kg
Nm(2).h(Nm(2)).{h(Nm(2)).h(Ns(1)).h(Pay(4))}_kg

Figure 18: Interception of subscriber's credentials by an intruder

The MSC attack trace for the above protocol is as follows:

```
----

i -> (s,3): start
(s,3) -> i: h(Ns(1))

i -> (s,3): h(Ns(1))
(s,3) -> i: Nm(2).h(Nm(2))
```

```
i -> (s,3): Nm(2).h(Nm(2)).{h(Nm(2)).h(Nm(2)).h(Nm(2))}_kg
(s,3) -> i: Ng(3).flag_g.{flag_g.h(dummy_nonce).h(Nm(2))}_inv(kg)

i -> (s,3): {h(Ns(1)).h(Ns(1)).h(dummy_nonce)}_kg
(s,3) -> i: Nm(2).h(Nm(2)).{h(Nm(2)).h(Ns(1)).h(Pay(4))}_kg

----
```

Therefore, the 1KP variant of the SETMS protocol is not safe enough for securing the e-commerce sessions for multicast environments.

7.2.1 SETMS specs for the AVISPA tool

The Secure E-commerce Transactions for Multicast Services (SETMS) Protocol Suite is designed to allow for secure e-commerce sessions for multicast environments. The SETMS is a variant of the 2KP protocol. The purpose of the SETMS protocol suite specification is intended to provide secrecy of end-host, authentication of the subscriber, and authorization and secrecy of subscriber’s e-payments. Apart from that, the protocol must securely send the payment credentials to the payment gateway and not authorize the merchant to decrypt these credentials. The subscriber sends its HITs details to the black-box without allowing any other communicating party to know the value of HITs. The merchant authenticates the subscriber based on the challenge that the merchant sends to the subscriber. In a way, it is a one-way authentication protocol where the subscriber doesn’t authenticate the merchant. The non-repudiation is supported as the authorized payments of the subscriber mark as the time stamp for the transaction. It has been assumed that the subscriber and merchant already possess the public key of the payment gateway and that the subscriber is communicating with a genuine merchant. It has to be noted that dishonest merchant in the protocol run would cause attacks as subscriber can no way hold a secure transaction if the merchant acts maliciously. The merchant and the payment gateway are assumed to possess asymmetric keys during the e-commerce transactions.

The SETMS protocol suite run that was illustrated in Figure 12 and Figure 13 in the previous chapter can be depicted in Alice&Bob-style notation as follows:

- ALICE_BOB:

```
0. S -> M: SALTs, Hash(Ids, Ns)
1. M -> S: Clear, PG_cert, M_cert
2. S -> M: Enc_G(Slip), Enc_M(Hash(HITs))
3. M -> G: Clear, Hash(SALTs, DESC), Enc_G(Slip), Sig_M
4. G -> M: Y/N, Enc_M(Y/N, Hash(Common))
5. M -> B: Y/N, Enc_B(gid, Hash(HITS), Y/N)
```

Where, Agent "B" is a Black box that is a secured module combining together the AAA server and the policy server. It should be noted that the communication link between the components inside the agent "B" is assumed to be well protected from intruders. We would now simplify the above SETMS protocol to remove the actual complexities in the protocol run. This simplification has been done to focus on just validating those SETMS protocol's parameters that contribute towards the objective of the protocol validation process. The protocol proceeds among the following agents: subscriber S, merchant server M, payment gateway G and Black Box B. S generates nonces Ns, and M generates nonces Nm and Nm1. M and G are assumed to possess asymmetric key pair (Km, inv_Km) and (Kp, inv_Kp) respectively. S is assumed to possesses digital certificates of the merchant and the payment gateway as {X, Kx}_inv(Kg) and {X, Kx}_inv(Km) respectively. The protocol also makes use of hash function to preserve the freshness of the data in transit. We consider the following goals: the parties shall authenticate each other on (the hash of) the order and payment information. The simplified SETMS protocol specifications in Alice&Bob-style notation is as follows:

- ALICE_BOB:

Step 0. Initial request

```
        S -> M: H(Ns)
```

Step 1. Initial response

```
        M -> S: H(Ns), H(Nm),{H(H(Ns), H(Nm))}_inv(Km)
```

Step 2. Purchase request
```
       S -> M: {H(Ns), H(Nm), H(HITs)}_Km, {H(Nm), H(Pay)}_Kg
```

Step 3. Payment authorization request
```
       M -> G: Nm, {H(Nm), H(Pay)}_Kg
```

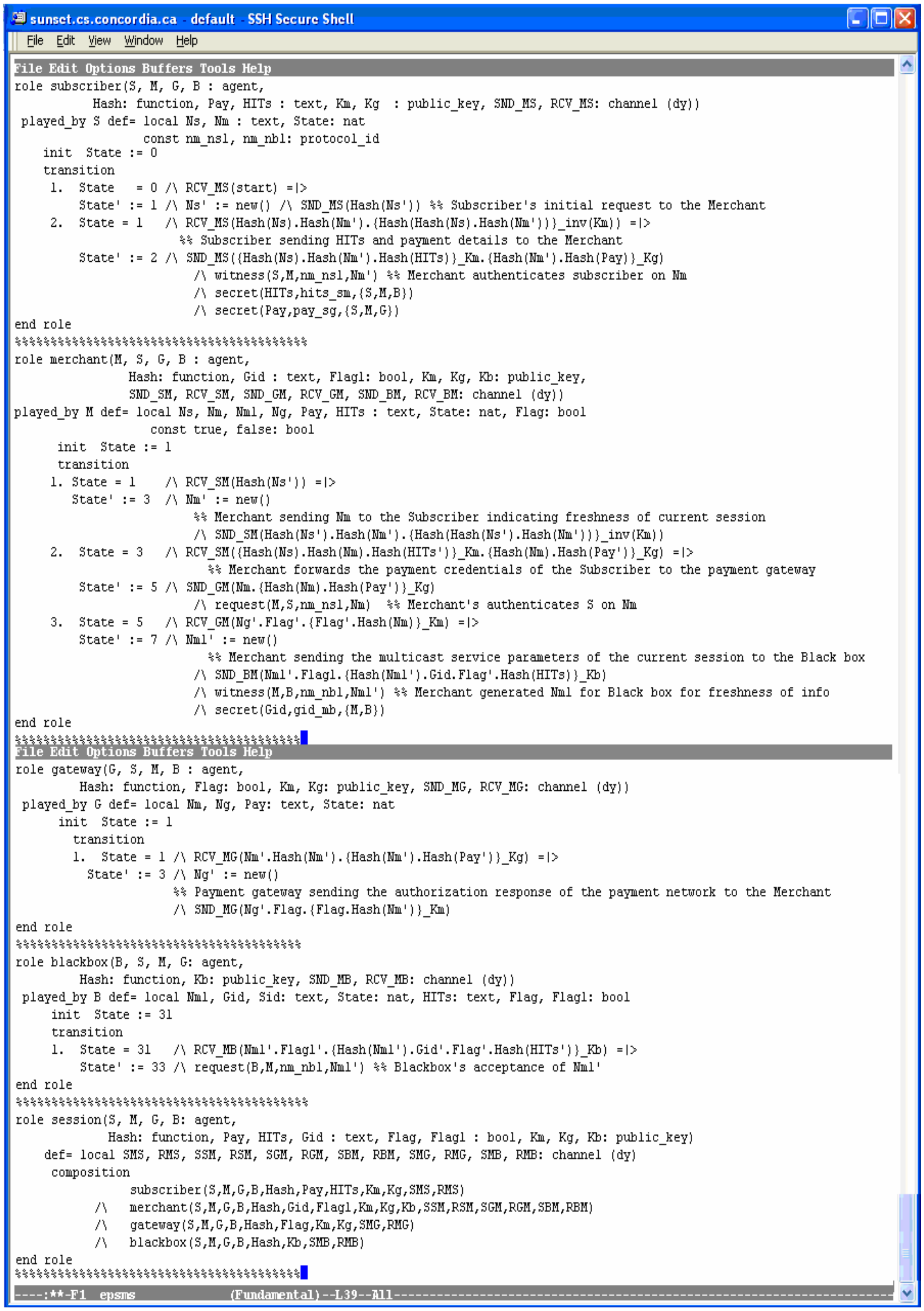
Step 4. Payment authorization response
```
       G -> M: Ng, Flag, {Flag, H(Nm)}_Km
```

Step 5. Service authorization
```
       M -> B: Nm1, Flag1, {H(Nm1), Gid, Flag, H(HITs)}_Kb
```

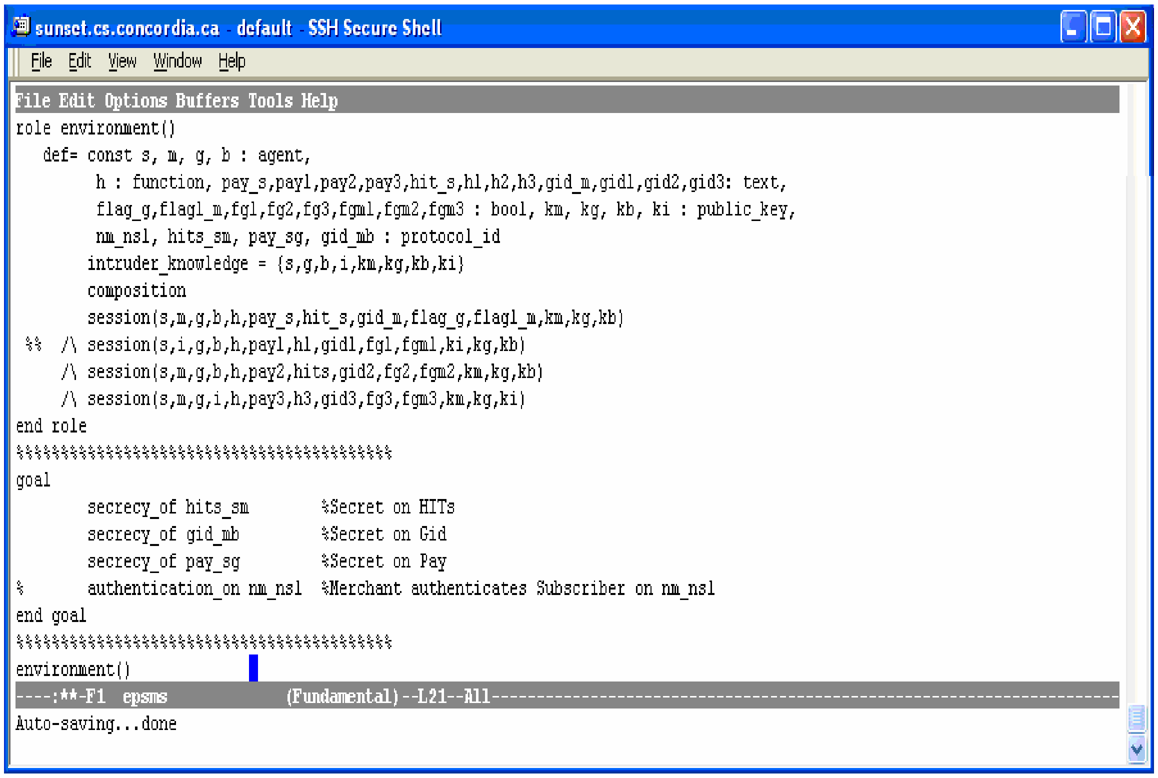
In step 0, the subscriber sends an initial request to the merchant showing its interest to participate in e-commerce. In step 1, the merchant sends its response to show it accepts the subscriber's requesting. In step 2, the subscriber sends its Hash(HIT_S) encrypted with merchant's public key and the order and payment information encrypted using payment gateways public key. In step 3, the merchant forwards the subscriber's order and payment information to the payment gateway (authorization request) to verify if the subscriber has enough funds in his account to be authorized to receive the merchant's service. In step 4, the payment gateway sends its authorization response encrypted with merchant's public key. In step 5, the merchant verifies the authorization response and sends the encrypted details of the multicast group id, authorization response code and the HIT_S details to the black box. The HLPSL for the SETMS protocol suite can also be found in the Appendix section of this thesis work. A sample of the HLPSL code that appears in the Appendix section is disclosed as below:

```
----
role subscriber(S, M, G, B : agent,
                Hash: function, Pay, HITs : text, Km, Kg  : public_key,
                                      SND_MS, RCV_MS: channel (dy))
 played_by S def=
- - - -
```

The HLPSL for the SETMS protocol suite is shown in Figure 19 and it is as follows:


sunset.cs.concordia.ca - default - SSH Secure Shell
File Edit View Window Help
File Edit Options Buffers Tools Help

```
role subscriber(S, M, G, B : agent,
           Hash: function, Pay, HITs : text, Km, Kg  : public_key, SND_MS, RCV_MS: channel (dy))
 played_by S def= local Ns, Nm : text, State: nat
                  const nm_nsl, nm_nbl: protocol_id
    init  State := 0
    transition
     1.  State   = 0 /\ RCV_MS(start) =|>
         State' := 1 /\ Ns' := new() /\ SND_MS(Hash(Ns')) %% Subscriber's initial request to the Merchant
     2.  State = 1   /\ RCV_MS(Hash(Ns).Hash(Nm').{Hash(Hash(Ns).Hash(Nm'))}_inv(Km)) =|>
                       %% Subscriber sending HITs and payment details to the Merchant
         State' := 2 /\ SND_MS({Hash(Ns).Hash(Nm').Hash(HITs)}_Km.{Hash(Nm').Hash(Pay)}_Kg)
                         /\ witness(S,M,nm_nsl,Nm') %% Merchant authenticates subscriber on Nm
                         /\ secret(HITs,hits_sm,{S,M,B})
                         /\ secret(Pay,pay_sg,{S,M,G})
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role merchant(M, S, G, B : agent,
                Hash: function, Gid : text, Flagl: bool, Km, Kg, Kb: public_key,
                SND_SM, RCV_SM, SND_GM, RCV_GM, SND_BM, RCV_BM: channel (dy))
played_by M def= local Ns, Nm, Nml, Ng, Pay, HITs : text, State: nat, Flag: bool
                   const true, false: bool
      init  State := 1
      transition
     1. State = 1    /\ RCV_SM(Hash(Ns')) =|>
        State' := 3  /\ Nm' := new()
                         %% Merchant sending Nm to the Subscriber indicating freshness of current session
                         /\ SND_SM(Hash(Ns').Hash(Nm').{Hash(Hash(Ns').Hash(Nm'))}_inv(Km))
     2.  State = 3   /\ RCV_SM({Hash(Ns).Hash(Nm).Hash(HITs')}_Km.{Hash(Nm).Hash(Pay')}_Kg) =|>
                           %% Merchant forwards the payment credentials of the Subscriber to the payment gateway
         State' := 5 /\ SND_GM(Nm.{Hash(Nm).Hash(Pay')}_Kg)
                         /\ request(M,S,nm_nsl,Nm)  %% Merchant's authenticates S on Nm
     3.  State = 5   /\ RCV_GM(Ng'.Flag'.{Flag'.Hash(Nm)}_Km) =|>
         State' := 7 /\ Nml' := new()
                           %% Merchant sending the multicast service parameters of the current session to the Black box
                         /\ SND_BM(Nml'.Flagl.{Hash(Nml').Gid.Flag'.Hash(HITs)}_Kb)
                         /\ witness(M,B,nm_nbl,Nml') %% Merchant generated Nml for Black box for freshness of info
                         /\ secret(Gid,gid_mb,{M,B})
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role gateway(G, S, M, B : agent,
         Hash: function, Flag: bool, Km, Kg: public_key, SND_MG, RCV_MG: channel (dy))
 played_by G def= local Nm, Ng, Pay: text, State: nat
      init  State := 1
        transition
        1.  State = 1 /\ RCV_MG(Nm'.Hash(Nm').{Hash(Nm').Hash(Pay')}_Kg) =|>
          State' := 3 /\ Ng' := new()
                      %% Payment gateway sending the authorization response of the payment network to the Merchant
                      /\ SND_MG(Ng'.Flag.{Flag.Hash(Nm')}_Km)
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role blackbox(B, S, M, G: agent,
         Hash: function, Kb: public_key, SND_MB, RCV_MB: channel (dy))
 played_by B def= local Nml, Gid, Sid: text, State: nat, HITs: text, Flag, Flagl: bool
     init  State := 31
     transition
     1.  State = 31   /\ RCV_MB(Nml'.Flagl'.{Hash(Nml').Gid'.Flag'.Hash(HITs')}_Kb) =|>
         State' := 33 /\ request(B,M,nm_nbl,Nml') %% Blackbox's acceptance of Nml'
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role session(S, M, G, B: agent,
             Hash: function, Pay, HITs, Gid : text, Flag, Flagl : bool, Km, Kg, Kb: public_key)
    def= local SMS, RMS, SSM, RSM, SGM, RGM, SBM, RBM, SMG, RMG, SMB, RMB: channel (dy)
     composition
                subscriber(S,M,G,B,Hash,Pay,HITs,Km,Kg,SMS,RMS)
          /\    merchant(S,M,G,B,Hash,Gid,Flagl,Km,Kg,Kb,SSM,RSM,SGM,RGM,SBM,RBM)
          /\    gateway(S,M,G,B,Hash,Flag,Km,Kg,SMG,RMG)
          /\    blackbox(S,M,G,B,Hash,Kb,SMB,RMB)
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
----:**-F1  epsms            (Fundamental)--L39--All-------------------------------------------
```



sunset.cs.concordia.ca - default - SSH Secure Shell
File Edit View Window Help
File Edit Options Buffers Tools Help

```
role environment()
   def= const s, m, g, b : agent,
         h : function, pay_s,pay1,pay2,pay3,hit_s,h1,h2,h3,gid_m,gid1,gid2,gid3: text,
         flag_g,flag1_m,fg1,fg2,fg3,fgm1,fgm2,fgm3 : bool, km, kg, kb, ki : public_key,
         nm_ns1, hits_sm, pay_sg, gid_mb : protocol_id
        intruder_knowledge = {s,g,b,i,km,kg,kb,ki}
        composition
        session(s,m,g,b,h,pay_s,hit_s,gid_m,flag_g,flag1_m,km,kg,kb)
 %%   /\ session(s,i,g,b,h,pay1,h1,gid1,fg1,fgm1,ki,kg,kb)
      /\ session(s,m,g,b,h,pay2,hits,gid2,fg2,fgm2,km,kg,kb)
      /\ session(s,m,g,i,h,pay3,h3,gid3,fg3,fgm3,km,kg,ki)
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
goal
        secrecy_of hits_sm       %Secret on HITs
        secrecy_of gid_mb        %Secret on Gid
        secrecy_of pay_sg        %Secret on Pay
%       authentication_on nm_ns1  %Merchant authenticates Subscriber on nm_ns1
end goal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
environment()
----:**-F1  epsms            (Fundamental)--L21--All------------------------------------------------------------------
Auto-saving...done
```

Figure 19: HLPSL for the SETMS protocol suite

7.2.2 SETMS validation results using AVISPA tool

The problems that were intended to be addressed by the SETMS protocol are as follows:

- Secrecy of host identity tag of the subscriber (denoted by "HIT$_S$")
- Secrecy of payment details (denoted by "Pay")
- Secrecy of multicast group id (denoted by "Gid")
- Strong authentication on Nm (merchant authenticates subscriber based on Nm value)
- Weak authentication on Nm (merchant authenticates subscriber based on Nm value)

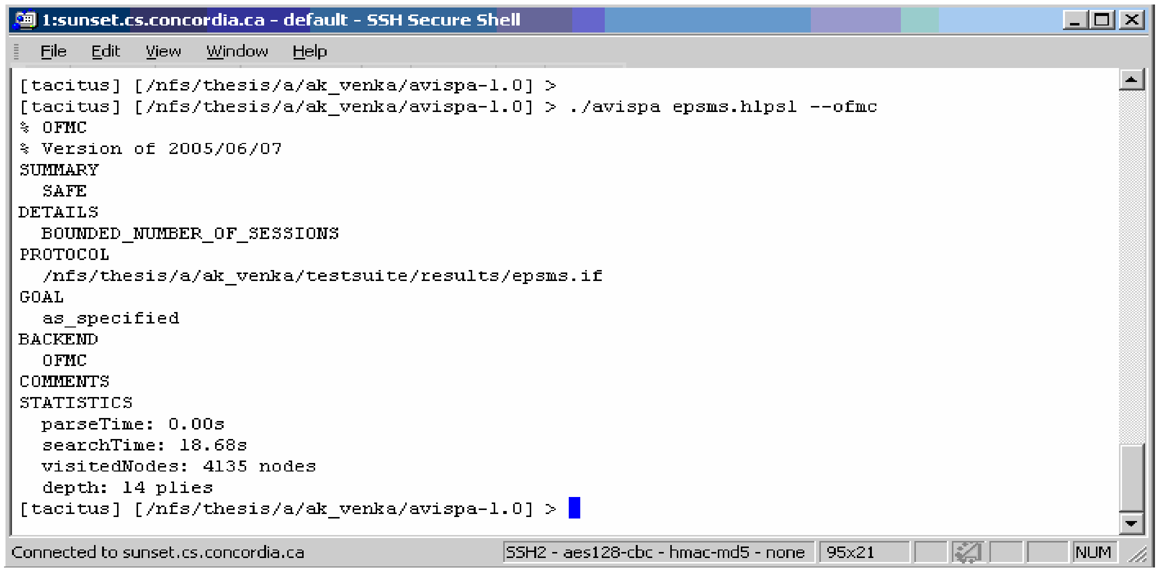
The following assumptions were made in validating the SETMS protocol suite:

- The exchange of digital certificates and signatures procedure between merchant, payment gateway and implicitly trusted CA is assumed to have happened before the protocol run.
- The merchant is assumed to be honest when the subscriber sends its HIT_S.
- The payment process involving description of goods, amount and payment method are simply treated as the parameter “Pay” in the protocol run. Thus, if parameter “Pay” is secured in the protocol run then it should be understood that the subscriber payment credentials are secured.

The validation results of the HLPSL of the SETMS protocol suite using AVISPA is as follows:

- OFMC: SAFE

The SETMS protocol has been found to be safe from OFMC attacks as below in Figure 20. It shows that the protocol run executes a bounded number of sessions with the number of visited nodes being 4135 and the search time being 18.68s with a depth search of 14 piles.

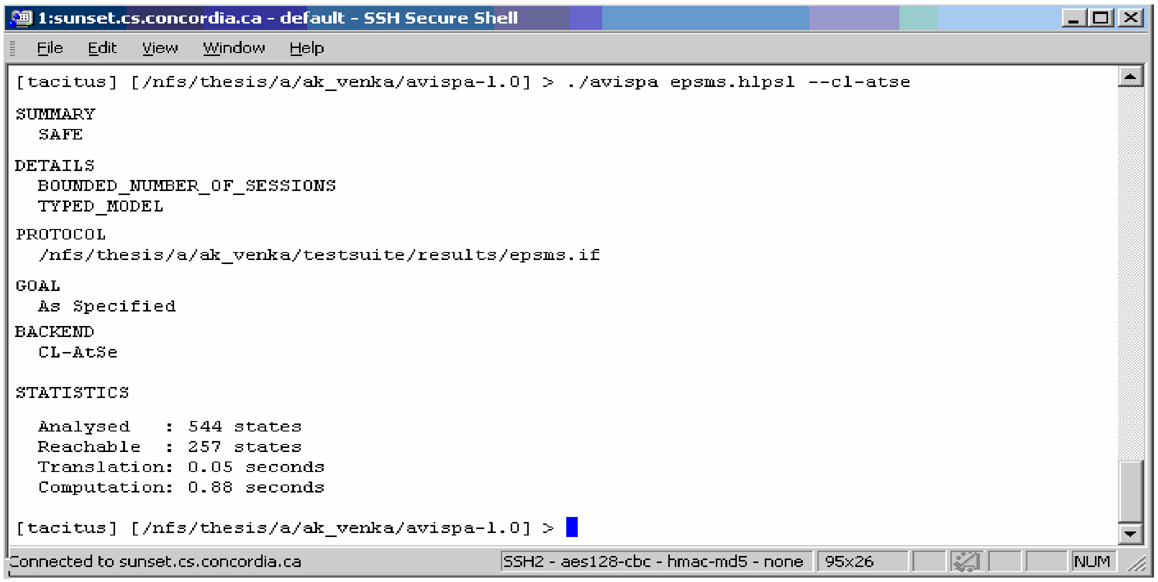

1:sunset.cs.concordia.ca - default - SSH Secure Shell
File Edit View Window Help
Connected to sunset.cs.concordia.ca
SSH2 - aes128-cbc - hmac-md5 - none
95x21
NUM

```
[tacitus] [/nfs/thesis/a/ak_venka/avispa-1.0] >
[tacitus] [/nfs/thesis/a/ak_venka/avispa-1.0] > ./avispa epsms.hlpsl --ofmc
% OFMC
% Version of 2005/06/07
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /nfs/thesis/a/ak_venka/testsuite/results/epsms.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 18.68s
  visitedNodes: 4135 nodes
  depth: 14 plies
[tacitus] [/nfs/thesis/a/ak_venka/avispa-1.0] >
```

Figure 20: OFMC safe SETMS protocol suite

- CL-AtSe: SAFE

The SETMS protocol has been found to be safe from CL-AtSe attacks as below in Figure 21. It shows that the protocol run executes a bounded number of sessions with the number of analysed states being 544 and the reachable states being 257. The translation time for the operation has been noted as 5 seconds and computation time as 88 seconds respectively.


1:sunset.cs.concordia.ca - default - SSH Secure Shell
File Edit View Window Help
Connected to sunset.cs.concordia.ca
SSH2 - aes128-cbc - hmac-md5 - none
95x26
NUM

```
[tacitus] [/nfs/thesis/a/ak_venka/avispa-1.0] > ./avispa epsms.hlpsl --cl-atse
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL
PROTOCOL
  /nfs/thesis/a/ak_venka/testsuite/results/epsms.if
GOAL
  As Specified
BACKEND
  CL-AtSe

STATISTICS

  Analysed   : 544 states
  Reachable  : 257 states
  Translation: 0.05 seconds
  Computation: 0.88 seconds

[tacitus] [/nfs/thesis/a/ak_venka/avispa-1.0] >
```

Figure 21: CL-AtSe safe SETMS protocol suite

- SATMC: INCONCLUSIVE

The SATMC results for the SETMS protocol suite were inconclusive as the search time was taking infinite time without showing any results. This implies that the SETMS protocol suite does not fit into the SATMC state-of-the-art back-end analyzer and therefore, it cannot return attacks (if any) to the protocol analyzer in a readable output format. This kind of inconclusiveness in quite acceptable when performing validation of a protocol suite. For example, SRP (Secure Remote Passwords) protocol [72] that was validated using the AVISPA tool by invoking state-of-art back-ends have returned that the SRP protocol is OFMC: Safe, CL-AtSe: Safe, SATMC: Inconclusive and TA4SP: Inconclusive.

- TA4SP: INCONCLUSIVE

The TA4SP results for the SETMS protocol suite were inconclusive as the search time was taking infinite time without showing any results. This implies that the SETMS protocol suite does not fit into the TA4SP state-of-the-art back-end analyzer and therefore, it cannot show if the protocol is flawed or is safe for any number of sessions.

The following are the theoretical arguments that support that the validated results of the proposed SETMS protocol suite is indeed a secure e-commerce protocol suite for multicast environments. These arguments are necessary to give any reader of the protocol an insight into what these validated results actually mean. These arguments are based on the Alice&Bob-style notation which we described for the SETMS protocol suite.

- Secrecy of (HIT_S): In step 2, the subscriber has sent its Hash(HITs) value encrypted with merchant's public key to the merchant.
    - Arguments: As Hash(HIT_S) is a one-way function, any party (intruder) intercepting this value cannot retrieve the actual HIT_S value. Since the HLPSL assumes merchant is an honest principal, the value of Hash(HIT_S) is assumed to reach the merchant and to be forwarded to the Black box at some point in the future. If an intruder intercepts Hash(HIT_S) value before it reached the merchant, still he cannot pretend as a honest merchant and forward Hash(intruders identity) to the legitimate Black box. This is because the SETMS protocol states that black box and merchant exchange the digital certificates and signatures, hence if the black box receives the HIT_S sent by an intruder, it can easily verify the legitimacy of the merchant and discard the HIT_S value in the case where the packet that it received is not from the legitimate merchant.
- Secrecy of Pay: In step 2, the subscriber has sent the Hash(Pay) value encrypted with the payment gateway's public key.
    - Arguments: As Hash(Pay) is also a one-way function, any intercepting party cannot know the actual value of the Pay. Hence the secrecy of "Pay" value is preserved.

- Secrecy of Gid: In Step 5, the merchant sends the multicast group id (Gid) value encrypted with the black box public key.
    - Arguments: Since the merchant and the black box have exchanged digital certificates and signatures and exchange the messages encrypted with the public key of the receiving party, any intercepting party (intruder) cannot decrypt the message to alter the encrypted contents.
- Strong authentication on Nm: In step 1, the subscriber receives Hash(Nm) value from the merchant. In step 2, the subscriber sends back the value of Hash(Nm) encrypted with the public key of the merchant.
    - Arguments: The use of nonce in the session will offer protection from replay attacks. The merchant authenticates the subscriber once it verifies the actual Hash(Nm) value and the one which it received from the subscriber that is encrypted with merchant's public key. The honest merchant can only decrypt the subscriber's message and authenticate the subscriber respectively.
- Weak authentication on Nm: This condition was verified just to test if this condition holds in case the strong authentication on Nm does not hold.

Chapter 8

Conclusion

This research work is the first attempt by any individual to explore the impact of e-commerce infrastructure for multicast environments. The whole idea of presenting the MSPP is to give the reader a clear understanding of the security requirement specifications and operational controls that are needed to secure multicast sessions in an e-commerce operational environment. The MSPP is written as a high-level specification of security requirements for generic e-commerce transactions. It can be used as a standalone framework for understanding the multicast security related aspects in an e-commerce environment.

The follow-up of MSPP specifications would facilitate any protocol developer and developer to cross check the applicability of the developed protocol with the MSPP to perceive if it meets the multicast security requirements of an e-commerce operational environment. If they follow the MSPP guidelines, the likelihood of building a secure multicast network for e-commerce environment is more. The developer or administrators can also know the constraints in which TOE security environment works. The TOE security environment clearly states the threat agents, sources of vulnerabilities and possible attacks, and administrative security policies that must be in place to secure the operational environment. MSPP also identifies and categorizes the risks to the TOE and provides its direct implication with the associated threats and vulnerabilities. The MSPP concludes by stating the security objectives that are intended to provide shielding against the security threats, attacks, vulnerabilities that arise due to breaches in the TOE or its operational environment. Thus, MSPP security specifications reveal that it is not possible for just one reliable multicast protocol to fit each and every multicast application. Once the MSPP was documented, we have designed the SETMS architectural framework that can be applied at various layers of the IP stack to secure e-commerce transactions for multicast services.

The SETMS framework provides solutions to the problems of end-host authentication, subscriber and his e-payment authorization, and suggests a distributed accounting model for administering e-commerce sessions. The SETMS framework delivers these solutions by relying on a variant of the 2KP protocol and secure service delegating processes namely, MCP and PS, and TTP components that provide support for non-repudiation of principal parties. SETMS supports non-repudiation of principals through its PKI infrastructure and also has the capability to support anonymity of the subscriber through the use of unpublished HI. The SETMS is a highly scalable system, easy to administer and easy to use with a moderate computational overhead and message size that results from the usage of a variant of the 2KP protocol for authorizing e-payments. The SETMS protocol suite, which is a variant of the 2KP secures host identity details and authorizes the subscriber and his e-payments. The framework offers scalability of the accounting infrastructure as it uses minimal control and data messages to provide security to e-commerce sessions, thus avoiding traffic congestion. SETMS is robust as there is no integrity check on the multicast UDP datagrams and explicitly there is no need to send a negative acknowledgement to any principal for the lost packets. It is sole decision of the subscriber to decide whether to participate in e-commerce where the merchant offers a service on “best-effort” delivery. The SETMS framework could be blended in to any developing environment by following MSPP specifications to enforce the policies that fit in the designated developing environment.

The SETMS framework was then evaluated using the support of the documented MSPP and also by the support of multicast and e-commerce literature. The evaluation of the protocol has indeed provided sufficient arguments in support of the designed framework. This was followed with a formal validation of the SETMS protocol suite using the AVISPA tool. The validation of the SETMS protocol suite using the AVISPA tool prove formally that the protocol which was designed is indeed a safe protocol that could be relied on for widespread deployment. However, the SETMS protocol suite was validated based on some assumptions, hence one must be aware of these assumptions before the actual deployment of the protocol.

Finally, we conclude by stating there is no single silver bullet for securing e-commerce transactions for multicast environments. Issues of protocol reliability, simplicity, stability, interoperability, resource consumption, and service cost usually dictate the applicability of multicast applications to an e-commerce environment. A defense in depth at various layers must be employed to secure transactions without causing infrastructure failure. This could be done by deploying components such as properly configured firewalls, server proxies, antivirus software, cryptographic techniques, remote backups, egress filters, and employing IT technical resource managers at various levels of administrative operations.

This initiated research work will lead to enormous scope in the direction of future work for securing e-commerce transactions for multicast services. This paves future directions in the research area of securing both multicast sessions as well as e-commerce sessions. The following research areas are identified as prospects for future work:

- Defining the sets of service parameters (attribute-value pairs) that define the possible multicast services. These parameters are shared between merchant’s content provider and policy server, and between policy server and AAA server with necessary transformations
- Design and development of Micropayments model that could accommodate the multicast services effectively without incurring a lot of overhead on the communication network

- Design of a framework that could customize the service delivery based on the subscriber's service request
- Development of protocols that could detect the lost packets and favor retransmission of those packets without incurring high latency at the receiver
- Development of mechanisms to reduce the interaction of the subscriber with the merchant in the processes of authentication of the subscriber and his host, and authorization of e-payments by relying on the AAA server's infrastructure that directly increases the scalability of the SETMS framework to wider audiences
- Auditing becomes a new requirement to attract new merchants to participate in e-commerce. The administrative personnel or service provider may have to provide proof to its clients that its network and processes are efficient and secure e-commerce is possible

Bibliography

[1] S. Deering, "Host extensions for IP multicasting", IETF RFC 1112, August 1989.

[2] R. Moskowitz, P. Nikander, "Host Identity Protocol Architecture", IETF Internet Draft, draft-ieft-hip-arch-03.txt (work in progress), August 2005.

[3] R. Vida, L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6", IETF RFC 3810, June 2004.

[4] R. Moskowitz et al., "Host Identity Protocol", IETF Internet Draft, draft-ietf-hip-base-04 (work in progress), October 2005.

[5] P. Nikander, J. Laganier, "Host Identity Protocol (HIP) Domain Name System (DNS) Extensions", IETF Internet Draft, draft-ietf-hip-dns-03 (work in progress), October 2005.

[6] J. W. Atwood, "A Classification of Multicast Protocols", IEEE Network, vol. 18, no. 3, May-June 2004, pp. 24-34.

[7] S. Kent, R. Atkinson, "Security Architecture for Internet Protocol", IETF RFC 2401, November 1998.

[8] I. Stoica, et al., "Internet Indirection Infrastructure", ACM SIGCOMM '02, vol. 32, no. 4, August 2002, pp. 73-86.

[9] P. Jokela, "Using ESP transport format with HIP", IETF Internet Draft, draft-ietf-hip-esp-01 (work in progress), October 2005.

[10] S. Kent, R. Atkinson, "IP Authentication Header (AH)", IETF RFC 2402, November 1998.

[11] S. Rafaeli, D. Hutchison, "Survey of key management for secure group communication", ACM Computing Surveys, 2003, vol. 35, no. 3, pp. 309-329.

[12] G. Caronni, K. Waldvogel, D. Sun, B. Plattner, “Efficient Security for Large and Dynamic Multicast Groups”, 7th Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), Proc. of IEEE Computer Society, June 1998, pp. 376-383.

[13] S. Kent, R. Atkinson, “IP Encapsulating Security Payload (ESP)”, IETF RFC 2406, November 1998.

[14] M. Baugher, Carrara, “The use of TESLA in SRTP”, IETF Internet Draft, draft-ietf-msec-srtp-tesla-05 (work in progress), October 2005.

[15] A. Perrig, J.D. Tygar, D. Song, R. Canetti, “Efficient Authentication and Signing of Multicast Streams over Lossy Channels”, Proc. of the 2000 IEEE Symposium on Security and Privacy, IEEE Computer Society Press, May 2000, pp. 56-73.

[16] A. Perrig et al., “Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction”, IETF RFC 4082, June 2005.

[17] S. Mittra, “Iolus: A framework for scalable secure multicasting,” Proc. of ACM SIGCOMM’97, vol. 27, no. 4, October 1997, pp. 277-288.

[18] IETF Authentication, Authorization and Accounting (AAA) working group charter available at: http://www.ietf.org/html.charters/aaa-charter.html.

[19] Randy C. Marchany and Joseph G. Tront, “E-Commerce Security Issues”, 35th Annual Hawaii International Conference on System Sciences (HICSS), January 2002, pp. 2500-2508.

[20] J. McGregor et al., “A Processor Architecture Defense Against Buffer Overflow Attacks”, Proc. of the IEEE International Conference on Information Technology: Research and Education (ITRE), August 2003, pp. 243-250.

[21] S. Xu, R. Sandhu, “Authenticated Multicast Immune to Denial-of-Service Attacks”, ACM SAC, March 2002, pp. 196-200.

[22] N. Brownlee, C. Mills, G. Ruth, “Traffic Flow Measurement: Architecture”, IETF RFC 2722, October 1999.

[23] M. Beadles, D. Mitton, “Criteria for Evaluating Network Access Server Protocols”, IETF RFC 3169, September 2001.

[24] R. Housley et al., “Internet X.509 Public Key Infrastructure Certificates and Certificate Revocation List (CRL) Profile”, IETF RFC 3280, April 2002.

[25] T. Polk, E. Hastings, A. Malpani. “Public Key Infrastructures that Satisfy Security Goals”, IEEE Internet Computing, vol. 7, no. 4, July-August 2003, pp. 60-67.

[26] P. Judge, M. Ammar, “Security Issues and Solutions in Multicast Content Distribution: A Survey”, IEEE Network, January-February 2003.

[27] M. Myers et al., “X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP”, IETF RFC 2560, June 1999.

[28] S. Chokhani, W. Ford, “Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework”, IETF RFC 2527, March 1999.

[29] M. Bellare, J. A. Garay, R. Hauser, A. Herzberg, H. Krawczyk, M. Steiner, G. Tsudik, E. V. Henreweghen, M. Waidner., “Design, implementation and deployment of the iKP secure electronic payment system”, IEEE Journal on Selected Areas in Communications, vol. 18, no. 4, April 2000, pp. 611-627.

[30] SET business description, programmer’s guide, formal protocol definition, and protocol description: http://www.cl.cam.ac.uk/Research/Security/resources/SET/

[31] Stephen Kost, “An Introduction to SQL Injection Attacks for Oracle Developers”, HelpNet-Security, January 2004.

[32] D. M. J. Moyer, J. R. Rao, P. Rohatgi, “A Survey of Security Issues in Multicast Communications”, IEEE Network Magazine, vol. 13, no. 6, November-December 1999, pp. 12-23.

[33] M. Handley, C. Perkins, E. Whelan, "SAP: Session Announcement Protocol", IETF RFC 2974, October 2000.

[34] A. Armando et al., "The Avispa Tool for the automated validation of internet security protocols and applications", Proc. of Computer Aided Verification (CAV), LNCS 3576, Springer Verlag, 2005.

[35] T. Hardjono, G. Tsudik, "IP Multicast Security: Issues and Directions", Annales de Telecom, July-August 2000, pp. 324-340.

[36] M. Banikazemi, "IP Multicasting: Concepts, Algorithms, and Protocols", August 1997, http://www.cse.wustl.edu/~jain/cis788-97/ip_multicast/index.htm

[37] A. Celik et al., "I-DG: A Secure Protocol for Disseminating Data to subscribers via IP Multicast", Proc. of 4th IEEE International workshop on WECWIS, June 2002, pp. 113-120.

[38] P. Chrysanthis, K. Pruhs, V. Liberator, "Middleware support for multicast-based data dissemination: a working reality", Proc. of the 4th International conference on WORDS, September 2003.

[39] M. Kouadio, U. W. Pooch, "A taxonomy and design considerations for Internet accounting", ACM SIGCOMM, Computer Communication Review, vol. 32, no. 5, November 2002, pp. 39-48.

[40] H. Sallay and O. Festor, "A Highly Distributed Dynamic IP Multicast Accounting and Management Framework", Integrated Network Management, March 2003, pp. 45-58.

[41] L. R. Dondeti, B. Haberman, H. J. Sandick, T. Hardjono, H. He, "MBA: A Tool for Multicast Billing and Accounting", Proc. of the 6th IEEE Symposium on Computers and Communications, July 2001, pp.172-177.

[42] L. C. Paulson, "Inductive analysis of the internet protocol TLS", ACM Transactions on Information and System Security, vol. 2, no. 3, August 1999, pp. 332-351.

[43] C. Bolchini, F. Salice, F. Schreiber, L. Tanca, “Logical and Physical Design Issues for Smart Card Databases”, ACM Transactions on Information Systems, vol. 21, no. 3, July 2003, pp. 254-285.

[44] N. Asokan, Ph. Janson, M. Steiner, M. Waidner, “State of the Art in Electronic Payment Systems”, Advances in Computers, Academic Press, vol. 43, Mar 2000, pp. 425-449.

[45] M. Sirbu, “Credits and Debits on the Internet”, IEEE Spectrum, vol. 34, no. 2, February 1997, pp. 23-39.

[46] A. Basu, S. Muylle, “Authentication in e-commerce”, Communications of the ACM, vol. 46, no. 12, December 2003, pp. 159-166.

[47] D. Mitton, M. Beadles, “Network Access Server Requirements Next Generation (NASREQNG) NAS Model”, IETF RFC 2881, Network working group, July 2000.

[48] A. Perrig, R. Canetti, D. Song, J. D. Tygar, “Efficient and secure source authentication for multicast”, Network and Distributed System Security Symposium (NDSS), February 2001, pp. 35-46.

[49] A. Habib, M. Hefeeda, B. K. Bhargava, “Detecting Service Violations and DoS Attacks”, Proc. of Network and Distributed System Security Symposium (NDSS), February 2003, pp. 177-189.

[50] S. Keung, K. Y. Siu, “Efficient protocols secure against guessing and replay attacks”, Proc. of the 4th International Conference on Computer Communications and Networks (ICCCN), September 1995, pp. 105-112.

[51] Sniffing frequently asked questions: http://cs.ecs.baylor.edu/~donahoo/tools/sniffer/sniffingFAQ.htm

[52] E. W. Felten, D. Balfanz, D. Dean, D. S. Wallach, “Web Spoofing: An Internet Con Game”, Proc. of 20th National Information Systems Security Conference, October 1997.

[53] J. T. Trostle, “Timing attacks against trusted path”, Proc. of 1998 IEEE Symposium on Security and Privacy, May 1998, pp. 125-134.

[54] L. Orgill, W. Romney, G. Bailey, M. Orgill, "The urgency for effective user privacy-education to counter social engineering attacks on secure computer systems", Proc. of the 5th conference on Information technology education, October 2004, pp. 177-181.

[55] J. D. Tygar. "Atomicity in Electronic Commerce", Proc. of the 15th Annual ACM Symposium on Principles of Distributed Computing (PODC), November 1996, pp. 8-26.

[56] H. Schulzrinne, S. Casner, R. Frederderick, V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", IETF RFC 3550, July 2003.

[57] M. Baugher et al., "Secure Real-time Transport Protocol (SRTP)", IETF RFC 3711, March 2004.

[58] D. Li, D. R. Cheriton, "Valuating the Utility of FEC with Reliable Multicast", Proc. of 7th IEEE International Conference on Network Protocols (ICNP), October-November 1999, pp. 97-105.

[59] P. Gutman, "Use of Shared Keys in TLS Protocol", IETF Internet draft, draft-ietf-tls-sharedkeys-02 (work in progress), October 2003.

[60] M. Handley and V. Jacobson, "SDP: Session Description Protocol", IETF RFC 2327, April 1998.

[61] Common Criteria for Information Technology Security Evaluation – Part 1: Introduction and general model, version 2.1, August 1999.

[62] Common Criteria for Information Technology Security Evaluation – Part 2: Security functional requirements, version 2.1, August 1999.

[63] Common Criteria for Information Technology Security Evaluation – Part 3: Security assurance requirements, version 2.1, August 1999.

[64] T. Newsham, "Format String Attacks", Guardent, Inc., September 2000.

[65] S. Moskowitz and M. H. Kang, "Covert channels - here to stay?", Proc. of 9th Annual Conference on Computer Assurance (COMPASS), IEEE Press, June-July 1994, pp. 235–243.

[66] P. A. Henry, “Covert channels provided hackers the opportunity and the means for the current distributed denial of service attacks”, Technical report, 2000.

[67] Y. Chevalier et al., “A High Level Protocol Specification Language for Industrial Security-Sensitive Protocols”, Proc. of SAPS’04, Austrian Computer Society, 2004.

[68] Ericsson, “Discussion paper on MBMS data protection for download”, 3GPP TSG SA WG3 Security, S3-040231, May 2004.

[69] C. Rigney et al., “Remote Authentication Dial-In User Service (RADIUS)”, IETF RFC 2138, April 1997.

[70] C. Finseth, “An Access Control Protocol, Sometimes Called TACACS”, IETF RFC 1492, July 1993.

[71] P. Calhoun et al., “Diameter Base Protocol”, IETF RFC 3588, September 2003.

[72] Automated Validation of Internet Security Protocols and Applications (AVISPA): http://www.avispa-project.org/

[73] L. Lamport, “The temporal logic of actions”, ACM Transactions on Programming Languages and Systems, vol. 16, no. 3, May 1994, pp. 872–923.

[74] Y. Chevalier et al., “A High Level Protocol Specification Language for Industrial Security-Sensitive Protocols”, Proc. of Workshop on Specification and Automated Processing of Security Requirements (SAPS), September 2004.

[75] AVISPA. Deliverable 2.3: The Intermediate Format. www.avispa-project.org/delivs/2.3, 2003.

[76] M. Turuani, “Sécurité des Protocoles Cryptographiques: Décidabilité et Complexité”, Phd, Université Henri Poincaré, Nancy, December 2003.

[77] D. Basin, S. Mödersheim, and L. Viganò, “An On-The-Fly Model-Checker for Security Protocol Analysis”, In E. Snekkenes and D. Gollmann, editors, Proc. of ESORICS’03, LNCS 2808, Springer-Verlag 2003, pp. 253-270.

[78] A. Armando, L. Compagna, and P. Ganty, “SAT-based Model-Checking of Security Protocols using Planning Graph Analysis”, In K. Araki, S. Gnesi, and D. Mandrioli, editors, Proc. of the 12th International Symposium of Formal Methods Europe (FME), LNCS 2805, Springer-Verlag 2003, pp. 875-893.

[79] G. Leduc and F. Germeau, “Verification of Security Protocols using LOTOS – Method and Application”, Computer Communications, special issue on Formal Description Techniques in Practice, vol. 23, no. 12, 2000, pp. 1089-1103.

[80] G. J. Holzmann, “The Spin Model Checker”, IEEE Transactions on Software Engineering, vol. 23, no. 5, May 1997, pp. 279-295.

[81] Y. Boichut, P.-C. Heam, O. Kouchnarenko, F. Oehl, “Improvements on the Genet and Klay Technique to Automatically Verify Security Protocols”, Proc. of AVIS’04, ENTCS.

Appendix – HLPSL for SETMS

```
%% PROTOCOL: SETMS

%% VARIANT: 2KP protocol

%% PURPOSE: This document specifies the secure e-commerce transactions for multicast

%% services protocol suite that allows the merchant to authenticate the subscriber and provide

%% secrecy of host identity, e-payment details and multicast group identity.

%% ALICE_BOB:

%% Step 0. Initial request

%%     S -> M: H(Ns)

%% Step 1. Initial response

%%     M -> S: H(Ns),H(Nm),{H(H(Ns).H(Nm))}_inv(Km)

%% Step 2. Purchase request

%%     S -> M: {H(Ns), H(Nm), H(HITs)}_Km, {H(Nm), H(Pay)}_Kg

%% Step 3. Payment authorization request

%%     M -> G: Nm, {H(Nm), H(Pay)}_Kg

%% Step 4. Payment authorization response

%%     G -> M: Ng, Flag, {Flag, H(Nm)}_Km

%% Step 5. Service authorization

%%     M -> B: Nm1, Flag1, {H(Nm1), Gid, Flag, H(HITs)}_Kb

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% HLPSL:

role subscriber(S, M, G, B : agent,

                    Hash: function,
                    Pay, HITs : text, Km, Kg  : public_key,
                        SND_MS, RCV_MS: channel (dy))
 played_by S def=
          local Ns, Nm : text, State: nat
          const nm_ns1, nm_nb1: protocol_id
   init  State := 0
   transition
    1.  State   = 0 /\ RCV_MS(start) =|>
        State' := 1 /\ Ns' := new()
                    /\ SND_MS(Hash(Ns')) %% Subscriber's initial request to the Merchant
    2.  State = 1   /\ RCV_MS(Hash(Ns).Hash(Nm').{Hash(Hash(Ns).Hash(Nm'))}_inv(Km)) =|>
        State' := 2
        %% Subscriber sending HITs and payment details to the Merchant
          /\ SND_MS({Hash(Ns).Hash(Nm').Hash(HITs)}_Km.{Hash(Nm').Hash(Pay)}_Kg)
          /\ witness(S,M,nm_ns1,Nm') %% Merchant authenticates subscriber on Nm
          /\ secret(HITs,hits_sm,{S,M,B})
          /\ secret(Pay,pay_sg,{S,M,G})
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role merchant(M, S, G, B : agent,
                   Hash: function,
                   Gid : text, Flag1: bool, Km, Kg, Kb: public_key,
SND_SM, RCV_SM, SND_GM, RCV_GM, SND_BM, RCV_BM: channel (dy))
   played_by M def=
          local Ns, Nm, Nm1, Ng, Pay, HITs : text, State: nat, Flag: bool
     init  State := 1
```

```
      transition
     1. State = 1  /\ RCV_SM(Hash(Ns')) =|>
        State' := 3 /\ Nm' := new()
         %% Merchant sending Nm to the Subscriber indicating freshness of current session
                      /\ SND_SM(Hash(Ns').Hash(Nm').{Hash(Hash(Ns').Hash(Nm'))}_inv(Km))
     2.  State = 3
            /\ RCV_SM({Hash(Ns).Hash(Nm).Hash(HITs')}_Km.{Hash(Nm).Hash(Pay')}_Kg) =|>
          %% Merchant forwards the payment credentials of the Subscriber to the payment gateway
          State' := 5 /\ SND_GM(Nm.{Hash(Nm).Hash(Pay')}_Kg)
                       /\ request(M,S,nm_ns1,Nm)  %% Merchant's authenticates S on Nm
     3.  State = 5   /\ RCV_GM(Ng'.Flag'.{Flag'.Hash(Nm)}_Km) =|>
         State' := 7  /\ Nm1' := new()
          %% Merchant sending the multicast service parameters of the current session to Black box
                       /\ SND_BM(Nm1'.Flag1.{Hash(Nm1').Gid.Flag'.Hash(HITs)}_Kb)
                       /\ witness(M,B,nm_nb1,Nm1') %% Merchant generated Nm1 for Black box
                       /\ secret(Gid,gid_mb,{M,B})
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role gateway(G, S, M, B : agent,
                  Hash: function,
                  Flag: bool, Km, Kg: public_key,
               SND_MG, RCV_MG: channel (dy))
  played_by G def =
      local Nm, Ng, Pay: text, State: nat
      init  State := 1
      transition
      1.  State = 1   /\ RCV_MG(Nm'.Hash(Nm').{Hash(Nm').Hash(Pay')}_Kg) =|>

          State' := 3 /\ Ng' := new()

          %% Payment gateway sending the Auth-Resp. of the payment network to the Merchant

                          /\ SND_MG(Ng'.Flag.{Flag.Hash(Nm')}_Km)

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role blackbox(B, S, M, G: agent,

                        Hash: function, Kb: public_key,

                     SND_MB, RCV_MB: channel (dy))

   played_by B def= local Nm1, Gid, Sid: text, State: nat, HITs: text, Flag, Flag1: bool

   init  State := 31

   transition

   1.  State = 31   /\ RCV_MB(Nm1'.Flag1'.{Hash(Nm1').Gid'.Flag'.Hash(HITs')}_Kb) =|>

       State' := 33 /\ request(B,M,nm_nb1,Nm1') %% Blackbox's acceptance of Nm1'

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role session(S, M, G, B: agent,

                  Hash: function,

                   Pay, HITs, Gid : text, Flag, Flag1 : bool,

                    Km, Kg, Kb: public_key)

   def=

local SMS, RMS, SSM, RSM, SGM, RGM, SBM, RBM, SMG, RMG, SMB, RMB: channel (dy)

    composition

          subscriber(S,M,G,B,Hash,Pay,HITs,Km,Kg,SMS,RMS)

  /\      merchant(S,M,G,B,Hash,Gid,Flag1,Km,Kg,Kb,SSM,RSM,SGM,RGM,SBM,RBM)

  /\      gateway(S,M,G,B,Hash,Flag,Km,Kg,SMG,RMG)
```

```
   /\        blackbox(S,M,G,B,Hash,Kb,SMB,RMB)
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
role environment()
   def = const s, m, g, b, i : agent,
             h : function, pay_s,pay1,pay2,pay3,hit_s,h1,h2,h3,gid_m,gid1,gid2,gid3: text,
             flag_g,flag1_m,fg1,fg2,fg3,fgm1,fgm2,fgm3 : bool, km, kg, kb, ki : public_key,
             nm_ns1, hits_sm, pay_sg, gid_mb : protocol_id
             intruder_knowledge = {s,g,b,i,km,kg,kb,ki, inv(ki)}
             composition
             session(s,m,g,b,h,pay_s,hit_s,gid_m,flag_g,flag1_m,km,kg,kb)
 %%  /\ session(s,i,g,b,h,pay1,h1,gid1,fg1,fgm1,ki,kg,kb)
         /\  session(s,m,i,b,h,pay2,h2,gid2,fg2,fgm2,km,kg,kb)
         /\  session(s,m,g,i,h,pay3,h3,gid3,fg3,fgm3,km,kg,ki)
end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
goal
             secrecy_of hits_sm          %Secret on HITs
             secrecy_of pay_sg           %Secret on Pay
             secrecy_of gid_mb           %Secret on Gid
%           authentication_on nm_ns1  %Merchant authenticates Subscriber on nm_ns1
end goal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
environment()
```