An Extension of Multi Layer IPSec for Supporting Dynamic QoS and Security Requirements

Synopsis of the thesis by: **Arnab Kundu (S R No:** 5511-210-061-04610)

Degree: M.Sc. (Engg.) (ERP), **Department:** SERC.

Research Supervisor: **Professor N. Balakrishnan,** SERC.

Organization Guide: **Dr. G. Athithan**, Sc-'G', CAIR.

Governments, military, corporations, financial institutions and others exchange a great deal of confidential information using Internet these days. Protecting such confidential information and ensuring their integrity and origin authenticity are of paramount importance. There exist protocols and solutions at different layers of the TCP/IP protocol stack to address these security requirements. Application level encryption viz. *PGP* for secure mail transfer, *TLS* based secure TCP communication, *IPSec* for providing IP layer security are among these security solutions. Due to scalability, wide acceptance of the IP protocol, and its application independent character, the IPSec protocol has become a standard for providing Internet security.

The IPSec provides two protocols namely the *Authentication header (AH)* and the *Encapsulating Security Payload (ESP)*. Each protocol can operate in two modes, viz. *transport* and *tunnel* mode. The AH provides data origin authentication, connectionless integrity and anti replay protection. The ESP provides all the security functionalities of AH along with confidentiality. The IPSec protocols provide end-to-end security for an entire IP datagram or the upper layer protocols of IP payload depending on the mode of operation.

However, this end-to-end model of security restricts performance enhancement and security related operations of intermediate networking and security devices, as they can not access or modify transport and upper layer headers and original IP headers in case of tunnel mode. These intermediate devices include routers providing Quality of Service (QoS), TCP Performance Enhancement Proxies (PEP), Application level Proxy devices and packet filtering firewalls.

The interoperability problem between IPSec and intermediate devices has been addressed in literature. *Transport friendly ESP (TF-ESP)*, T*ransport Layer Security (TLS)*, splitting of single IPSec tunnel into multiple tunnels, *Multi Layer IPSec (ML-IPSec)* are a few of the proposed solutions. The ML-IPSec protocol solves this interoperability problem without violating the end-to-end security for the data or exposing some important header fields unlike the other solutions.

The ML-IPSec uses a multilayer protection model in place of the single end-to-end model. Unlike IPSec where the scope of encryption and authentication applies to the entire IP datagram, this scheme divides the IP datagram into *zones*. It applies different protection schemes to different zones. When ML-IPSec protects a traffic stream from its source to its destination, it first partitions the IP datagram into zones and applies zone-specific cryptographic protections. During the flow of the ML-IPSec protected datagram through an authorized intermediate gateway, certain type I zones of the datagram may be decrypted and re-encrypted, but the other zones will remain untouched. When the datagram reaches its destination, the ML-IPSec will reconstruct the entire datagram.

The ML-IPSec protocol, however suffers from the problem of static configuration of zones and zone specific cryptographic parameters before the commencement of the communication. Static configuration requires a priori knowledge of routing infrastructure and manual configuration of all intermediate nodes. While this may not be an issue in a geo-stationary satellite environment using TCP-PEP, it could pose problems in a mobile or distributed environment, where many stations may be in concurrent use. The ML-IPSec endpoints may not be trusted by all intermediate nodes in a mobile environment for manual configuration without any prior arrangement providing the mutual trust. The static zone boundary of the protocol forces one to ignore the presence of TCP/IP datagrams with variable header lengths (in case of TCP or IP headers with OPTION fields). Thus ML-IPSec will not function correctly if the endpoints change the use of IP or TCP options, especially in case of tunnel mode. The zone mapping proposed in ML-IPSec is static in nature. This forces one to configure the zone mapping before the commencement of the communication. It restricts the protocol from dynamically changing the zone mapping for providing access to intermediate nodes without terminating the existing ML-IPSec communication. The ML-IPSec endpoints can off course, configure the zone mapping with maximum number of zones. This will lead to unnecessary overheads that increase with the number of zones. Again, static zone mapping could pose problems in a mobile or distributed environment, where communication paths may change.

Our extension to the ML-IPSec protocol, called *Dynamic Multi Layer IPSec (DML-IPSec)* proposes a multi layer variant with the capabilities of dynamic zone configuration and sharing of cryptographic parameters between IPSec endpoints and intermediate nodes. It also accommodates IP datagrams with variable length headers. The DML-IPSec protocol redefines some of the IPSec and ML-IPSec fundamentals. It proposes significant modifications to the datagram processing stage of ML-IPSec and proposes a new key sharing protocol to provide the above-mentioned capabilities.

The DML-IPSec supports the AH and ESP protocols of the conventional IPSec with some modifications required for providing separate cryptographic protection to different zones of an IP datagram. This extended protocol defines *zone* as a set of non-overlapping and contiguous partitions of an IP datagram, unlike the case of ML-IPSec where a zone may consist of non-contiguous portions. Every zone is provided with cryptographic protection independent of other zones. The DML-IPSec categorizes zones into two separate types depending on the accessibility requirements at the intermediate nodes. The first type of zone, called *type I* zone, is defined on headers of IP datagram and is required for examination and modification by intermediate nodes. One type I zone may span over a single header or over a series of contiguous headers of an IP datagram. The second type of zone, called *type II* zone, is meant for the payload portion and is kept secure between endpoints of IPSec communications. The single type II zone starts immediately after the last type I zone and spans till the end of the IP datagram. If no intermediate processing is required during the entire IPSec session, the single type II zone may cover the whole IP datagram; otherwise the single type II zone follows one or more type I zones of the IP datagram. The DML-IPSec protocol uses a mapping from the octets of the IP datagram to different zones, called *zone map* for partitioning an IP datagram into zones. The zone map contains *logical boundaries* for the zones, unlike physical byte specific boundaries of ML-IPSec. The physical boundaries are derived on-the-fly, using either the implicit header lengths or explicit header length fields of the protocol headers. This property of the DML-IPSec zones, enables it to accommodate datagrams with variable header lengths. Another important feature of DML-IPSec zone is that the zone maps need not remain constant through out the entire lifespan of IPSec communication. The key sharing protocol may modify any existing zone map for providing service to some intermediate node.

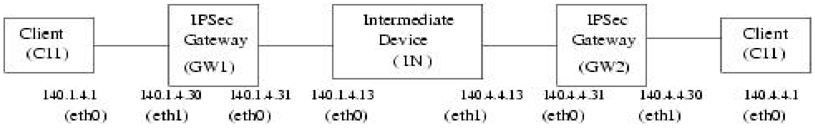
The DML-IPSec also redefines *Security Association (SA)*, a relationship between two endpoints of IPSec communication that describes how the entities will use security services to communicate securely. In the case of DML-IPSec, several intermediate nodes may participate in defining these security protections to the IP datagrams. Moreover, the scope of one particular set of security protection is valid on a single zone only. So a single SA is defined for each zone of an IP datagram. Finally all these individual zonal SA's are combined to represent the security relationship of the entire IP datagram. The intermediate nodes can have the cryptographic information of the relevant type I zones. The cryptographic information related to the type II zone is, however, hidden from any intermediate node. The key sharing protocol is responsible for selectively sharing this zone information with the intermediate nodes.

The DML-IPSec protocol has two basic components. The first one is for processing of datagrams at the endpoints as well as intermediate nodes. The second component is the key sharing protocol. The endpoints of a DML-IPSec communication involves two types of processing. The first one, called *Outbound processing*, is responsible for generating a DML-IPSec datagram from an IP datagram. It first *derives the zone boundaries* using the zone map and individual header field lengths. After this partitioning of IP datagram, *zone wise encryption* is applied (in case of ESP). Finally zone specific *authentication trailers* are calculated and appended after each zone. The other one, *Inbound processing*, is responsible for generating the original IP datagram from a DML-IPSec datagram. The first step in the inbound processing, the *derivation of zone boundary*, is significantly different from that of outbound processing as the length fields of zones remain encrypted. After receiving a DML-IPSec datagram, the receiver starts decrypting type I zones till it decrypts the header length field of the header/s. This is followed by zone-wise *authentication verification* and zone-wise *decryption*. The intermediate nodes processes an incoming DML-IPSec datagram depending on the presence of the security parameters for that particular DML-IPSec communication. In the absence of the security parameters, the key sharing protocol gets executed; otherwise, all the incoming DML-IPSec datagrams get partially decrypted according to the security association and zone mapping at the inbound processing module. After the inbound processing, the partially decrypted IP datagram traverses through the networking stack of the intermediate node . Before the IP datagram leaves the intermediate node, it is processed by the outbound module to reconstruct the DML-IPSec datagram.

The key sharing protocol for sharing zone related cryptographic information among the intermediate nodes is the other important component of the DML-IPSec protocol. This component is responsible for dynamically enabling intermediate nodes to access zonal information as required for performing specific services relating to quality or security. Whenever a DML-IPSec datagram traverses through an intermediate node, that requires access to some of the type I zones, the inbound security database is searched for cryptographic parameters. If no entry is present in the database, the key sharing protocol is invoked. The very first step in this protocol is a *header inaccessible* message from the intermediate node to the source of the DML-IPSec datagram. The intermediate node also mentions the *protocol headers* that it requires to access in the body portion of this message. This first phase of the protocol, called the *Zone reorganization phase*, is responsible for deciding the zone mapping to provide access to intermediate nodes. If the current zone map can not serve the header request, the DML-IPSec endpoint *reorganizes the existing zone map* in this phase. The next phase of the protocol, called the *Authentication Phase* is responsible for verifying the identity of the intermediate node to the source of DML-IPSec session. Upon successful authentication, the third phase, called the *Shared secret establishment phase* commences. This phase is responsible for the establishment of a temporary shared secret between the source and intermediate nodes. This shared secret is to be used as key for encrypting the actual message transfer of the DML-IPSec security parameters at the next phase of the protocol. The final phase of the protocol, called the *Security parameter sharing phase*, is solely responsible for actual transfer of the security parameters from the source to the intermediate nodes. This phase is also responsible for updation of security and policy databases of the intermediate nodes. The successful execution of the four phases of the key sharing protocol enables the DML-IPSec protocol to dynamically modify the zone map for providing access to some header portions for intermediate nodes and also to share the necessary cryptographic parameters required for accessing relevant type I zones without disturbing an existing DML-IPSec communication.

We have implemented the DML-IPSec for ESP protocol according to the definition of zones along with the key sharing algorithm. *RHEL version 4* and Linux kernel version *2.6.23.14* was used for the implementation. We implemented the multi-layer IPSec functionalities inside the native Linux implementation of IPSec protocol. The *SA structure* was updated to hold necessary SA information for multiple zones instead of single SA of the normal IPSec. The zone mapping for different zones was implemented along with the kernel implementation of SA. The inbound and outbound processing modules of the IPSec endpoints were re-implemented to incorporate multi-layer IPSec capability. We also implemented necessary modules for providing partial IPSec processing capabilities at the intermediate nodes. The key sharing protocol consists of some user space utilities and corresponding kernel space components. We use *ICMP* protocol for the communications required for the execution of the protocol. At the kernel level, pseudo character device driver was implemented to update the kernel space data structures and necessary modifications were made to relevant kernel space functions.

User space utilities and corresponding kernel space interface were provided for updating the security databases. As DML-IPSec ESP uses same Security Policy mechanism as IPSec ESP, existing utilities (viz. setkey) are used for the updation of security policy. However, the configuration of the SA is significantly different as it depends on the DML-IPSec zones. The DML-IPSec ESP implementation uses the existing utilities (setkey and racoon) for configuration of the sole type II zone. The type I zones are configured using the DML-IPSec application. The key sharing protocol also uses this application to reorganize the zone mapping and zone-wise cryptographic parameters. The above feature enables one to use default IPSec mechanism for the configuration of the sole type II zone.



For experimental validation of DML-IPSec, we used the testbed as shown in the above figure. An ESP tunnel is configured between the two gateways *GW1* and *GW2*. *IN* acts as an intermediate node and is installed with several intermediate applications. Clients *C11* and *C21* are connected to GW1 and GW2 respectively. We carried out detailed experiments for validating our solution w.r.t firewalling service. We used *stateful packet filtering* using iptables along with *string match extension* at IN. First, we configured the firewall to allow only FTP communication (using port information of TCP header and IP addresses of Inner IP header) between C11 and C21. In the second experiment, we configured the firewall to allow only Web connection between C11 and C21 using the Web address of C11 (using HTTP header, port information of TCP header and IP addresses of Inner IP header).

In both experiments, we initiated the FTP and WEB sessions before the execution of the key sharing protocol. The session could not be established as the access to upper layer headers was denied. After the execution of the key sharing protocol, the sessions could be established, showing the availability of protocol headers to the iptables firewall at IN following the successful key sharing.

We use *record route* option of ping program to validate the claim of handling datagrams with variable header lengths. This option of ping program records the IP addresses of all the nodes traversed during a round trip path in the IP *OPTION* field. As we used ESP in tunnel mode between GW1 and GW2, the IP addresses would be recorded inside the encrypted Inner IP header. We executed ping between C11 and C21 and observed the record route output. Before the execution of the key sharing protocol, the IP addresses of IN were absent in the record route output. After the successful execution of key sharing protocol, the IP addresses for IN were present at the record route output.

The DML-IPSec protocol introduces some processing overhead and also increases the datagram size as compared to IPSec and ML-IPSec. It increases the datagram size compared to the standard IPSec. However, this increase in IP datagram size is present in the case of ML-IPSec as well. The increase in IP datagram length depends on the number of zones. As the number of zone increases this overhead also increases.

We obtain experimental results about the processing delay introduced by DML-IPSec processing. For this purpose, we executed ping program from C11 to C21 in the test bed setup for the following cases: 1.ML-IPSec with one type I and one type II zone and 2. DML-IPSec with one type I and one type II zone. We observe around 10% increase in RTT in DML-IPSec with two dynamic zones over that of ML-IPSec with two static zones. This overhead is due to on-the-fly derivation of the zone length and related processing. The above experiment analyzes the processing delay at the endpoints without intermediate processing. We also analyzed the effect of intermediate processing due to dynamic zones of DML-IPSec. We used iptables firewall in the above mentioned experiment. The RTT value for DML-IPSec with dynamic zones increases by less than 10% over that of ML-IPSec with static zones.

To summarize our work, we have proposed an extension to the multilayer IPSec protocol, called Dynamic Multilayer IPSec (DML-IPSec). It is capable of dynamic modification of zones and sharing of cryptographic parameters between endpoints and intermediate nodes using a key sharing protocol. The DML-IPSec also accommodates datagrams with variable header lengths. The above mentioned features enable any intermediate node to dynamically access required header portions of any DML-IPSec protected datagrams. Consequently they make the DML-IPSec suited for providing IPSec over mobile and distributed networks. We also provide complete implementation of ESP protocol and provide experimental validation of our work. We find that our work provides the dynamic support for QoS and security services without any significant extra overhead compared to that of ML-IPSec.

The thesis begins with an introduction to communication security requirements in TCP/IP networks. Chapter 2 provides an overview of communication security protocols at different layers. It also describes the details of IPSec protocol suite. Chapter 3 provides a study on the interoperability issues between IPSec and intermediate devices and discusses about different solutions. Our proposed extension to the ML-IPSec protocol, called Dynamic ML-IPSec(DML-IPSec) is presented in Chapter 4. The design and implementation details of DML-IPSec in Linux environment is presented in Chapter 5. It also provides experimental validation of the protocol. In Chapter 6, we summarize the research work, highlight the contributions of the work and discuss the directions for further research.