通信情報保護學會論文誌
第 8 卷 第 1 號. 1998. 3

Construction of Security MIB for EDI System

Tae-Kyou Park

Abstract

This paper considers the design and management of security **MIB** for **EDI** system. **EDI** system has to establish various security services and mechanisms to protect against security threats. Hence, the **EDI** system requires appropriate security management to monitor and control the security objects for its security services and mechanisms. In this paper, I identify security objects for management of security services defined in the **EDI** system, and propose the design of a security **MIB** and describe the use of **SNMP** network management protocol in its management.

Keywords: *EDI Security Management Information Base, SNMP, Security Management*

1. Introduction

Electronic Data Interchange **(EDI)** is basically the concept of computer-to-computer exchange of messages or information relating to various types of activities in an organization or business. The security in the **EDI** system has a serious impact on the ways in which organizations and companies conduct their business transactions and manage their documents and messages. The basis for security in the **EDI** system is the **OSI** Security Architecture international standard[4]. This document describes a general framework in terms of security services, security mechanisms, security management functions, and some other relevant aspects of security in open systems, and gives some high level recommendations. Key to provision of a security service is its management. An **EDI** system needs to support the management of these security services as well as how changes in policy and its enforcement can take place. For instance, in the case of data confidentiality and integrity services, it is necessary to manage the keys used in the encryption and decryption process. In the case of message security labeling service, we need to manage the security policy regarding the labeling of documents and messages, and their transactions. Thus, there may be several authorities performing different aspects of these security management functions

* *Associate Professor in Department of Computer Science, Hanseo University*
이 논문은 1997년 Univ. of Western Sydney 방문 시 한국과학재단의 지원을 받아 연구되었음.

such as access control authorities, authentication authorities, key management authorities and audit management authorities. In practice, several of these functions may be handled by a single authority. One of the difficulties that the network manager has to face, with regard to security management, involves selecting and using the appropriate security management application to be secure against security attacks. In this paper, I identify a number of security-related managed objects which can be contained in Simple Network Management Protocol **(SNMP)** Management Information Base **(MIB)**, and are important for controlling and configuring security measures in the **EDI** system. In particular, I focus on security management objects and the design of a security **MIB (SMIB)** using the formal Structure of Management Information **(SMI)** encoding rules[6]. I propose a common **SMIB** definition for **EDI** system components such as user agent **(EDI-UA)**, message store **(EDI-MS)**, and message transfer agent **(MTA)**. This **SMIB** definition proposed in this paper is based on Simple Network Management Protocol version 2 **(SNMPv2)** protocol. This paper is organized as follows. Section 2 reviews the security service elements in **EDI** system and considers as an example the **KT-EDI** system[5,14]. Section 3 briefly considers the network security management in **EDI**. In section 4, I identify the security objects, then construct a **SMIB** for **EDI** System and security management using **SNMP** protocol is described in section 5. Network security protocols with the **SMIB** are given in section 6. Finally, section 7 concludes the paper.

2. Security Elements in EDI System

TU-T recommends two kinds of standardization for an **EDI** system : one is a document standard **(EDIFACT)**[9], and the other is the communication standard (**F.435** / **X.435** Recommendations[7]) based on the **X.400** Message Handling System **(MHS)**[8]. That is, the basic activity of the **EDI** system is the conveyance of electronic messages. The **EDI** interchanges can be conveyed in many ways, for example, directly over a telephone line or encapsulated in a file transfer. One method of providing a supporting infrastructure for the **EDI** is to use the **MHS**. The nature of the **MHS** should be borne in mind when considering fourteen security elements of security service defined in **X.402**, and further seven elements appear in **X.435**. For instance, Korea Telecom-**EDI (KT-EDI)** system focus on twenty-seven security elements from **X.402** and **X.435** documents. Also the functional models, communication protocols, potential threats and transfer message types of the **X.435 EDI** system have been applied. As shown in Table 1, **KT-EDI** system has various security services such as origin authentication, **EDI** Message **(EDIM)** responsibility authentication, secure access management, data confidentiality, data integrity, non-repudiation of **EDIM** responsibility, non-repudiation, message security labeling, and security management. The **X.400** recommendation belongs to the application layer of the **OSI** reference model. The originator of the message uses **UA** to compose a message and to submit it to the message transfer system **(MTS)**. A **UA** is also involved when the **MTS** delivers the message to its recipient (the user associated **UA**). After delivery, the recipient uses the services of its **UA** to process the received

Table 1. Relation between security service elements, and MHS components

Security Services	Security Service Elements	* UA/UA	UA/MS	MS/MTA	US/MTA	MTA/MS	MTA/MTA	MTA/UA	MS/UA
Origin Authentication (X.402)	Message Origin Authentication	0	0		0				0
	Probe Origin Authentication			0	0				
	Report Origin Authentication					0	0	0	
	Proof of Submission							0	
	Proof of Delivery	0							N
EDIM Responsibility Authentication (X.435)	Proof of EDI Notification	0							
	Proof of Retrieval		0						
	Proof of Transfer						0		
Secure Access Management (X.402)	Peer Entity Authentication		0	0	0	0	0	0	0
	Security Context		0	0	0	0	0	0	0
Data Confidentiality (X.402)	Connection Confidentiality		0	0	0	0	0	0	0
	Content Confidentiality	0							
	Message Flow Confidentiality	0							
Data Integrity (X.402)	Connection Integrity		0	0	0	0	0	0	0
	Content Integrity	0							
	Message Sequence Integrity	0							
Non-Repudiation of EDIM Responsibility (X.435)	Non-Repudiation of EDI Notification	0							
	Non-Repudiation of EDI Retrieval		0						
	Non-Repudiation of EDI Transfer						0		
	Non-Repudiation of EDI Content	0							
Non-repudiation (X.402)	Non-Repudiation of Origin	0			0				
	Non-Repudiation of Submission							00	
	Non-Repudiation of Delivery	0							0
Message Security Labeling (X.402)	Message Security Labeling	0	0	0	0	0	0	0	0
Security Management (X.402)	Change Credentials		0		0	0	0	0	
	Register		0		0				
	MS-Register		0						

* UA : EDI_UA, N : Receiver MS to Sender UA , 0 : Applicable between two EDI components.
UA : User Agent, MS : Message Store, MTA : Message Transfer Agent

message. Within the **MTS**, a set of **MTAs** cooperate in conveying messages to their recipients. Together with **MS**, the collection of **UAs** and **MTAs** comprise the **MHS**. A principal feature of the **MHS** is its operation in a store-and-forward manner, which is important for security analysis because of the increased risk to the information while stored temporarily in various network nodes.

There are two basic differences between regular network **MHS** and **EDI** system. First, **EDI** information is exchanged in the form of special messages, such as banking transactions, orders, invoices, letters, contracts, and proprietary materials, between companies and business partners. Second, each **EDI** message is transmitted under some special regime or requirements such as the request for confirmation, the receipt of an "equivalent" message, non-repudiation of content, legal binding, and acceptance of special conditions. The security capabilities in the **X.400** system have been achieved using different mechanisms, for example, the inclusion of new elements in the exchanged messages during the association establishment stage or by adding information in the **MHS** envelope. It should be emphasized that security capabilities included in the **MHS** system define only how to transfer and use relevant security parameters. Rules about generation and interpretation of these parameters are not in the **MHS** recommendations. Its aim is to provide security independent of the communication services supplied by other entities of higher or lower levels. Security of the **X.400 MHS** also requires certain management functions and support. Only the authorized entities may change user credentials or security labels. Most of the techniques (mechanisms) used to implement the described security services are based on cryptography. Security services of the **MHS** allow the selection of alternative algorithms. The service elements needed to implement security in the **X.400** system must be supported by the Directory Authentication environment, defined in the **X.509** Directory Service recommendation[12]. The Directory System stores certified copies of the user public keys of the **MHS** that can be used to provide authentication and facilitate the exchange of user credentials. Thus, mechanisms to secure data confidentiality and integrity are provided. The **X.509** recom- mendation defines a framework for the provision of an entity authentication by the Directory service to its users. These users include the Directory itself, as well as other applications and services. The Directory can usefully be involved in meeting their needs for authentication and other security services, because it is a natural place from which communicating parties can obtain authentication information about each other. The **X.509** recommendation describes two levels of authentication: simple authentication, using a password as a verification of claimed identity, and strong authentication, involving credentials formed using cryptographic techniques. The strong authentication method is based on public-key cryptosystems. For communicating with **MHS** components and another Directory Service Agent **(DSA)**, Directory Access Protocol **(DAP : UA, MS** or **MTA-DSA)** and Directory Service Protocol **(DSP : DSA-DSA)** are supported. The generation of user certificates is performed by some off-line Certificate Authority **(CA)** which is separate from the Directory Service Agencies **(DSAs)**.

3. Design of Security Management Model for EDI System

The design of security management model is aimed to enable the **EDI** system to provide the various security services defined in section 2. As shown in Fig 1, **KT-EDI** system architecture basically consists of **MHS** components such as **UA**s (**P3** or **P7**), **MS**s, **MTA**s and **DSA**s for Directory services. The **EDI** system for communications among **MHS** components is supported by some protocols such as **P1 (MTA-MTA)**, **P2 (UA-UA)**, **P3 (UA** or **MS-MTA)**, **P7 (UA-MS)** and Pedi (defining the heading of **EDI**). Each system component has a secure **EDI** subsystem **(SES)** and a secure management subsystem **(SMS)**. The **SES** is composed of secure **UA**s, secure MSs, and secure **MTA**s according to its functional role as **MHS** components. Each component transfers the services and related messages to the **SES** through its **SES** interface. The **SES** can access the various information from the **SMIB** and/or **MIB** through **SNMP** interface, hence plays a main role of **EDI** system. The **SMS** consists of three management agents such as a key management agent **(KMA)**, a audit management agent **(AMA)**, and a **SNMP** management agent **(SMA)**. The **KMA** has two functions, one is a directory service agent **(DSA)** function to keep and manage public key certificates, and the other is a directory user agent **(DUA)** function to get and keep secret keys and public keys to be used in security services. The **AMA** plays the role of storing and retrieving the security relevant events such as a event classifier, audit records, history record, and audit provider. The **SMA** is the "heart" of the security management of **KT-EDI** security services and mechanisms, and controls and manages the security related management information base **SMIB**. Security management has to provide facilities for allowing the network security, manager of the **EDI** system to control the security-relevant managed objects used in **EDI** security services and mechanisms such as security service requests, confidential keys, algorithm identifiers and security labels. For these facilities, each **EDI-UA, UA-MS** and **MTA** must include its local **MIB** and **SMIB** in which it can control its own resources, or grant or restrict access to the entire security manager or selected critical parts of the element security manager.

The security manager should have the facilities for archiving and retrieving the appropriate security information and managing and controlling the security objects. In addition, a network management system should provide the network manager with facilities for monitoring and analyzing the security measures. An example of real time monitoring of security measures could be the generation of an alarm when a single user has made numerous unsuccessful login attempts for a network host or the notification of repeated denials of user attempts on a particular service. In the context of **SNMP**, the real time monitoring can be accomplished in two different ways: the manager polls every agent in the network at frequent time intervals for some key security management information. The agent then notifies the manager of any unusual event concerning the agent' s security by sending a trap message. On the other hand, the analysis of security logs is also important in discovering

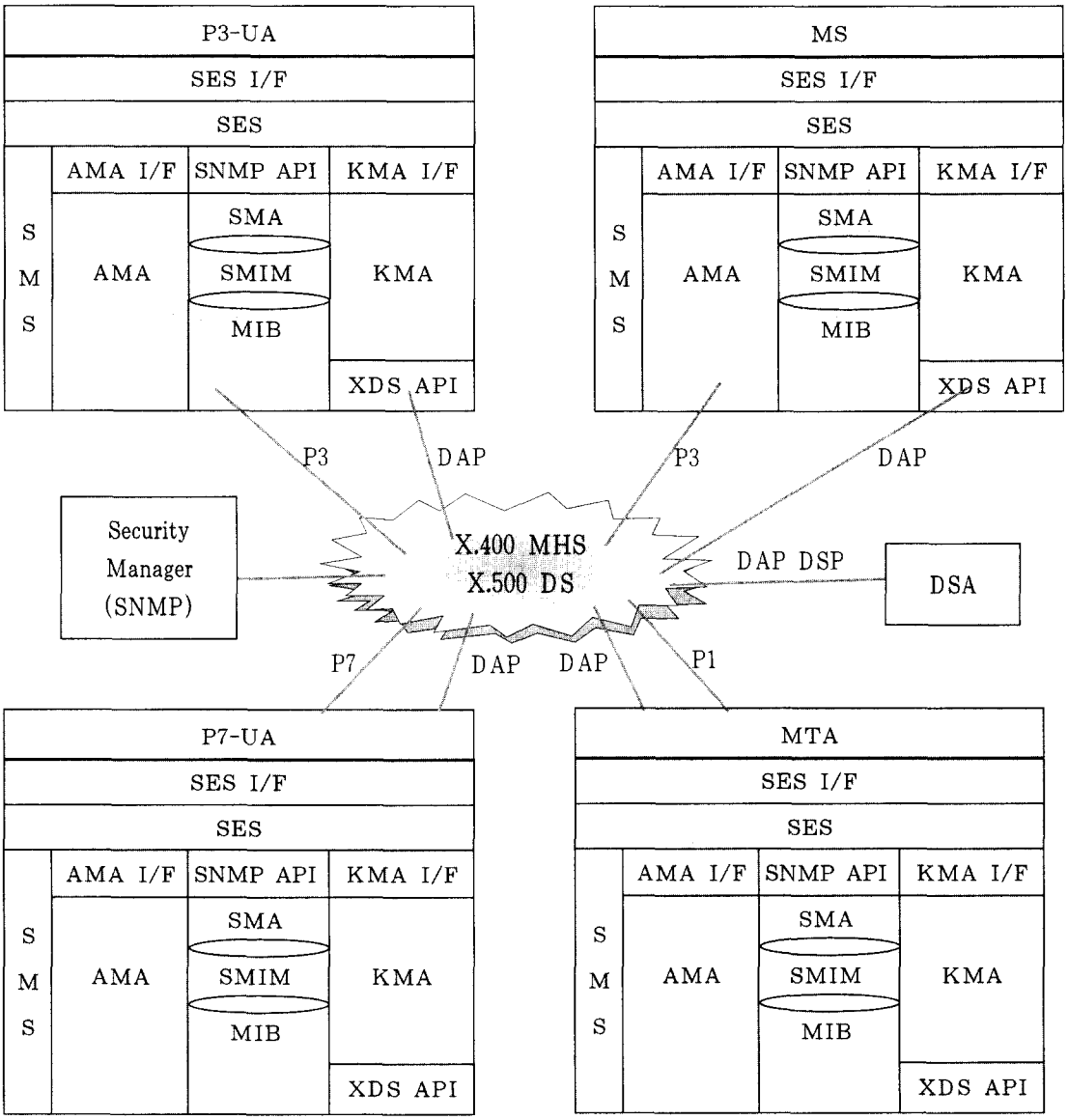

Fig. 1. Secure KT-EDI system architecture

security attacks that are not detectable as they occur. Using a local system **MIB**, this can be accomplished by an **SNMP** management application which polls periodically the agent of the network for security related information and stores the data related in a database. This approach using Temporal database for **SNMP** based network management can be found in[1]. By using the **SMIB** design, various tools that check the network security with the appropriate **SNMP** interface can become specific security management agents or element managers in a

network management architecture.

4. Security Objects and SMIB for EDI System

A network manager can adopt with regard to security management the development of the **SMIB** that will fulfill our special network security needs. Most of the information needed for security management will be stored in the **SMIB**. That is, the **SMIB** is the storage in which the secure network maintains all data pertinent to its security functions such as identities of authorized users, authentication data, user entity capabilities and privileges, security parameters of all network resources, access control privileges and various processing and recovery logs. The individual objects are identified and structured, and their usage for providing all security relevant parameters to various security service elements of the **EDI** system is described. The **SMIB** objects in the **EDI** system must be protected to the highest level of security. The **SMIB** may be implemented as a distributed information base to the extent that is necessary to enforce a consistent security policy in a logical or physical grouping of end-systems (security domain). In practice, parts of the **SMIB** may or may not be integrated with the **MIB** of the open system. There are many realizations of **SMIB** such as a table of data or a single file or a distributed set of data base segments or rules embedded within the software or hardware of the real system. Rules for inserting, maintaining, deleting and using information in the **SMIB** constitute security management protocols. Management protocols, especially security management protocols, and the communication channels carrying the management information, are potentially vulnerable. Particular care must therefore be taken to ensure that the management protocols and information are protected. Security management may require the exchange of security-relevant information between various administrations, in order that the **SMIB** can be established or extended. In some cases, the security-relevant information will be passed through non-**OSI** communication paths, and the local systems administrators will update the **SMIB** through methods not standardized by **OSI**. In other cases, it may be desirable to exchange such information over an **OSI** communication path, in which case the information will be passed between two security management applications running in an open system. The security management applications will use the communicated information to update the **SMIB**. Such updating of the **SMIB** requires prior authorization of appropriate security administrator or access privileges control of other authorized entities. The **SMIB** purposed in this paper stores security attributes for each association maintained within the **EDI** system. The attributes include security keys, request flags and identifiers needed by the **EDI** application and **MHS** protocol in the implementation of the security mechanisms. The **SMIB** can be implemented as a table of entries, one of each communicating pair of hosts. It allows the security management applications to control the operation of the **EDI** system. The steps used in creating a **MIB** requires the followings. 1) Gather the security variables want to control the target **EDI** system. 2) Construct a skeletal **SMIB** modules. 3) Categorize the security objects class and determine whether

there can exist multiple instances of that managed object class. If not, then for each of its attributes, use the **OBJECT-TYPE** macro to make an equivalent definition. Multiple instances are defined as a conceptual table. 4) Begin compiling **SMIB** by using a **MIB** compiler supporting **SNMP**v2. 5) Refine **SMIB** observing compiler output for correct data relations. For constructing **SMIB** according to the above steps, I first analyzed the data structures for implementation of each security services for **EDI** system, then identified security-related variables as security objects of **SMIB**. A number of security management objects can be identified, and these objects are classified into object groups according to the **EDI** security service elements. These **SMIB** objects are divided into the seven groups such as origin authentication, **EDIM**, data confidentiality, data integrity, non-repudiation, message security labeling and secure management group as shown in Table 2. The **EDIM** group has the objects identified from **EDIM** Responsibility Authentication and Non-Repudiation of **EDIM** Responsibility security services in Table 1. The secure management group has the objects identified from Secure Access Management and Security Management security service in Table 1. The other five groups has the objects from the corresponding security services respectively in Table 1. Each of them is represented as an object group in **SNMP**v2 definitions. Table 2 includes the security management objects in each group and their definitions.

5. SNMP in Managing SMIB for EDI System

By far the most widely used network management standard is the **SNMP**[3] protocol. It lets agents or managers set and read parameters and lets systems generate and transmit traps, which are special event notifications. Certain parameters may be security sensitive such as operational status, cryptographic algorithms and keys. The **SNMP** element manager may keep the database of secrets and authorization information for each community which specify

Table 2. Structures of EDI-SMIB objects groupsOrigin

Origin Authentication Group	Definitions
MessageOriginAuthenAlgID	Identifier of the algorithm used for Message Origin Authentication.
ProbeOriginAuthenAlgID	Identifier of the algorithm used for Probe Origin Authentication.
ProofOfDeliveryAlgID	Identifier of the algorithm used for Proof of Delivery.
ProofOfDeliveryRequest	Indicates whether Proof of Delivery is used or not.
ProofOfSubmissionRequest	Indicates whether Proof of Submission is used or not.
ThisRecipientName	The name of EDI-UA receiving EDI messages.

EDIM Group	Definitions
EDINotifRequests	Indicates whether Proof/Non-repudiation of EDI Notification is used or not.
EDINotifSecurity	Indicates whether Proof/Non-repudiation of EDI Notification Security is used or not.
EDIReceptionSecurity	Indicates whether Proof/Non-repudiation of EDI Reception Security is used or not.
NonRepOfEDIContentAlgID	Identifier of the algorithm used for Non-Repudiation of EDI Content.
NonRepOfEDINotifAlgID	Identifier of the algorithm used for Non-Repudiation of EDI Notification.

Data Confidentiality Group	Definitions
ContentConfidAlgID	Identifier of the algorithm used for Content Confidentiality.
ConnectionConfidAlgID	Identifier of the algorithm used for Connection Confidentiality.
ConfidAlgBlockSize	The block size supported by the algorithm which an identifier uniquely identifies.
ConfidAlgDecKeyLength	The length of decryption key of the algorithm which an identifier uniquely identifies.
ConfidAlgDecKey	The decryption key of the algorithm which an identifier uniquely identifies.
ConfidAlgEncKeyLength	The length of encryption key of the algorithm which an identifier uniquely identifies.
ConfidAlgEncKey	The encryption key of the algorithm which an identifier uniquely identifies.
ConfidAlgInitVectorIndicate	Indicates whether an initialization vector is used or not the algorithm identified.
ConfidAlgInitVectorLength	The length of the initialization vector used by the confidentiality algorithm identified.
ConfidAlgInitVector	The initialization vector used by the confidentiality algorithm identified.
ConfidAlgOperateMode	The mode in which the confidentiality algorithm operates.
ConfidAlgSymIndicate	Indicates whether the algorithm is symmetric or asymmetric.
ConfidAlgSyncIndicate	Indicates whether the algorithm is requires availability of synchronization information.
ConfidAlgSyncInfoLength	The size of the information transmitted within the field defined for synchronization.

Data Integrity Group	Definitions
ContentIntegrityAlgID	Identifier of the algorithm used for Content Integrity.
ConnectionIntegrityAlgID	Identifier of the algorithm used for Connection Integrity.
DigestAlgID	Identifier of the algorithm used for Data Integrity.
DigestAlgInitVectorIndicate	Indicates whether the digest algorithm requires an initialization vector or not.
DigestAlgInitVectorLength	The size of the initialization vector needed by the digest algorithm.
DigestAlgInitVector	The initialization vector used by the digest algorithm to process the cryptographic checksum.
DigestAlgInputSize	The size in octets of the input of the digest algorithm.
DigestAlgOutputSize	The size in octets of the output of the digest algorithm.
SigAlgID	Identifier of the signature algorithm used for Data Integrity.
SigAlgCheckKeyLength	The key size used by the signature algorithm to check the integrity checksum.
SigAlgCheckKey	The key used by the signature algorithm to check the checksum.
SigAlgGenKeyLength	The key size used by the signature algorithm to generate the checksum.
SigAlgGenKey	The key used by the signature algorithm to generate the checksum.
SigAlgInitVectorIndicate	Indicates whether or not an initialization vector is required by the signature algorithm.
SigAlgInitVectorLength	The size in octets of the initalization vector used by the signature algorithm.
SigAlgInitVector	The initialization vector used by the signature algorithm to process the integrity checksum.
SigAlgInputSize	The size in octets of the input of the signature algorithm.
SigAlgOutputSize	The size in octets of the output of the signature algorithm.
SigAlgSymIndicate	Indicates whether the signature algorithm is symmetric or asymmetric.

Non-Repudiation Group	Definitions
NonRepOfDeliveryAlgID	The algorithm identifier used for Non-repudiation of Delivery.
NonRepOfOriginAlgID	Identifier of the algorithm used for Non-repudiation of Origin.
NonRepOfSubmissionAlgID	Identifier of the algorithm used for Non-repudiation of Submission.
ProofOfDeliveryRequest	Indicates whether Proof / Non-repudiation of Delivery is used or not.
ProofOfSubmissionRequest	Indicates whether Proof / Non-repudiation of Submission is used or not.

Message Security Labeling Group	Definitions
MinimumSecurityLabel	The maximum security label supported by EDI system.
MaximumSecurityLabel	The minimum security label supported by EDI system.
SecurityPolicyID	Indicates which security policy is supported by EDI system.
SecurityCategories	Indicates which security classification is supported by EDI system, e.g. top secret.
SecurityClassification	Indicates which security category is supported by EDI system, e.g. staff only.
PrivacyMark	Indicates which privacy mark is supported by EDI system, e.g. In confidence.

Secure Management Group	Definitions
PeerEntitySigAlgID	Identifier of the algorithm used for Peer Entity Authentication among EDI-UA, EDI-MS and MTA.
UserName	Username used for Register when MTA and MS-users register to MTA and MS respectively.
UserAddress	User address used for Register when MTA and MS-users register to MTA and MS respectively.
InitiatorPassword	Initiator password used for Peer Entity Authentication when the association is established.
SubjectPublicKeyAlgID	Subject Public Key Identifier currently stored in Directory Service Agent for EDI security services.

what parameters each community is allowed to access. **SNMP** version 1 allows read or write access to different subsets of parameters depending on which group a user is in. The community concept is a local one, defined at the agent. The agent establishes one community for each desired combination of authentication, access control and proxy characteristics. Each community is given a unique community name within this agent, and the managers within that community employ the community name in all "Get" and "Set" operations such as setting passwords. An access mode with "read-only" or "read-write" is defined for each community. However, enhanced security was one of the primary goals behind the design of version 2 of **SNMP**. **SNMP**v2 is designed to provide, in essence, three security-related services such as privacy, authentication, and access control. Privacy is the protection of transmitted data from eavesdropping or wiretapping. Privacy requires that contents of any message be distinguished in such a way that only the intended recipient can recover the original message, and uses **DES** for encrypting the **SNMP** message. The specification mentions the possibility of algorithms such as using other algorithms, including public key algorithms. A message, file, document, or other collection of data is said to be authentic when it is genuine and came from its alleged source. Message authentication is a procedure that allows communicating parties to verify that received message are authentic. The two important aspects are to verify that the contents of the message have not been altered and that the source is authentic.

Each **SNMP**v2 message can be authenticated and integrity-protected using a shared secret configured into the system being managed and the system doing the management. This is done by creating a cryptographic checksum using a protected **MD5** message digest ; the message digest is sent along with the message. We also wish to verify the timeliness of the message, that is, it has not been delayed and replayed, and the sequence relative to other messages flowing between two parties is maintained. In **SNMP**v2, each message includes a message header, which contains security-related information. The message structures in Fig. 2 show the general format as well as the private and authenticated format. The header consists of five fields. The srcParty identifies the party of the manager or the agent sending the message. The desParty identifies the party of the agent or the manager to whom the message is sent. The context may indicate that this exchange relates to an access to a **MIB** local to the agent; in this case, the context value serves to identify a subset of the agent's **MIB**, known as an **MIB** view. The combination of source party, destination party and context value is used to determine the access control privileges for this exchange. The authInfo field contains information relevant to the authentication protocol. The privDst field repeats the identifier of the destination party. Together with the appropriate parameters, the **PDU** field contains one of the commands such as "Get", "GetNext", "GetBulk", "Set", "Trap", "Inform" and "Response".

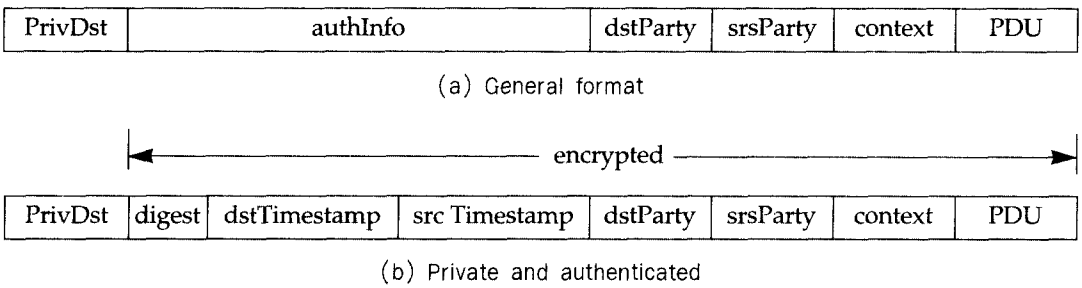

Fig. 2. SNMPv2 massage formats

If the message is authenticated and private, then the authInfo field contains information needed for authentication and the entire message, including header and the **PDU** but excluding the privDst field, is encrypted. The privDst field must remain unencrypted so that the destination **SNMP**v2 entity can determine the destination party and therefore determine the privacy characteristics of the message. In the context of network management, the purpose of access control is to ensure that only authorized users have access to a particular **MIB** and that access to and modification of a particular portion of data is limited to authorized individuals and programs. Thus, the access control policy is determined by three parameters. A source party requests a management operation in a destination party and identifies the context of the request. The context may specify an **MIB** view local to the destination party or may specify a remote proxied entity. For a given pair of source/destination party, there may be multiple access control policies, one for each context. The context is communicated by the source to the destination in the **SNMP**v2 message header. This approach eliminates the necessity of defining a unique source/destination party pair for access control policy, thus enables a single destination party to perform in a variety of contexts for a given source party. The value of the privileges parameter represents the list of **SNMP**v2 **PDU**s that may be sent from the source to the destination. The parameter is encoded by assigning an integer value that is a power of 2 to each **PDU**.

Access control is determined by information in the party **MIB**. This **MIB** consists of four tables: party table, context table, access control table, and **MIB** view table. The best way to describe the function of these tables for access control is to consider their use during message transmission. Consider a message that is sent from a manager to an agent. The message header includes the fields srcParty, dstParty, and context. The party table at the agent contains information about each local and remote party known to the agent. The party information includes authentication parameters that need to be applied to srcParty and privacy parameters that need to be applied to dstParty. The context table contains one entry for each context known to agent. Each entry specifies whether the

Table 3. EDI-SMIB objects defined in SNMPv2

Object Groups	SMIB Objects in SNMPv2	SYNTAX	MAX-ACCESS
Origin Authentication Group (orginAuthGroup)	messageOriginAuthenAlgID	OBJECT IDENTIFIER	read-write
	probeOriginAuthenAlgID	OBJECT IDENTIFIER	read-write
	proofOfDeliveryAlgID	OBJECT IDENTIFIER	read-write
	proofOfDeliveryRequest	TruthValue	read-write
	proofOfSubmissionRequest	TruthValue	read-write
	thisRecipientName	DistinguishedName	read-write
EDIM Group (edimGroup)	eDINotifRequests	TruthValue	read-write
	eDINotifSecurity	TruthValue	read-write
	eDIReceptionSecurity	TruthValue	read-write
	nonRepOfEDIContentAlgID	OBJECT IDENTIFIER	read-write
	nonRepOfEDINotifAlgID	OBJECT IDENTIFIER	read-write
	contentConfidAlgID	OBJECT IDENTIFIER	read-write
Data Confidentiality Group (dataConfidGroup)	connectionConfidAlgID	OBJECT IDENTIFIER	read-write
	confidAlgBlockSize	Integer32	read-create
	confidAlgDecKeyLength	Integer32	read-create
	confidAlgDecKey	OCTET STRING	read-create
	confidAlgEncKeyLength	Integer32	read-create
	confidAlgEncKey	OCTET STRING	read-create
	confidAlgInitVectorIndicate	TruthValue	read-create
	confidAlgInitVectorLength	Integer32	read-create
	confidAlgInitVector	OCTET STRING	read-create
	confidAlgOperateMode	DisplayString	read-create
	confidAlgSymIndicate	TruthValue	read-create
	confidAlgSyncIndicate	TruthValue	read-create
	confidAlgSyncInfoLength	Integer32	read-create
Data Integrity Group (dataIntegrityGroup)	contentIntegrityAlgID	OBJECT IDENTIFIER	read-write
	connectionIntegrityAlgID	OBJECT IDENTIFIER	read-write
	digestAlgID	OBJECT IDENTIFIER	read-write
	digestAlgInitVectorIndicate	TruthValue	read-create
	digestAlgInitVectorLength	Integer32	read-create
	digestAlgInitVector	OCTET STRING	read-create
	digestAlgInputSize	Integer32	read-create
	digestAlgOutputSize	Integer32	read-create
	sigAlgID	OBJECT IDENTIFIER	read-write
	sigAlgCheckKeyLength	Integer32	read-create
	sigAlgCheckKey	OCTET STRING	read-create
	sigAlgGenKeyLength	Integer32	read-create
	sigAlgGenKey	OCTET STRING	read-create
	sigAlgInitVectorIndicate	TruthValue	read-create
	sigAlgInitVectorLength	Integer32	read-create
	sigAlgInitVector	OCTET STRING	read-create
	sigAlgInputSize	Integer32	read-create
	sigAlgOutputSize	Integer32	read-create
Non-Repudiation Group (nonRepudGroup)	*sigAlgSymIndicate*	TruthValue	read-create
	nonRepOfDeliveryAlgID	OBJECT IDENTIFIER	read-write
	nonRepOfOriginAlgID	OBJECT IDENTIFIER	read-write
	nonRepOfSubmissionAlgID	OBJECT IDENTIFIER	read-write
	proofOfDeliveryRequest	TruthValue	read-write
Message Security Labeling Group (msgSecLabGroup)	proofOfSubmissionRequest	TruthValue	read-write
	minimumSecurityLabel	DisplayString	read-write
	maximumSecurityLabel	DisplayString	read-write
	securityPolicyID	DisplayString	read-write
	securityCategories	DisplayString	read-write
	securityClassification	DisplayString	read-write

Security Management Group (secureMgmtGroup)	privacyMark	DisplayString	read-write
	peerEntitySigAlgID	OBJECT IDENTIFIER	read-write
	userName	Taddress	read-create
	userAddress	OCTET STRING	read-create
	initiatorPassword	OCTET STRING	read-create
	subjectPublicKeyAlgID	OBJECT IDENTIFIER	read-create

* *Italic specified objects are defined within the object tables.*

context is local, in which case the proxied device is indicated. The **MIB** view table is referenced by the context table, The appropriate entry defines a subset of the local **MIB** that is accessible through this context. Finally, each entry in the access control table has a unique combination of srcParty, dstParty and context, and this indicates which management operations (which **PDUs**) are allowed for this combination. As a result, **SNMP**v2 provides the protection against threats such as disclosure, masquerading, message content modification, and message sequence and timing modification. Table 3 illustrates how the **SMIB** objects can be defined for **SNMP**v2. **SYNTAX** means object types in **SMI** rules and **MAX-ACCESS** stands for maximum access privileges to the objects. Subsequently after the security object types and access privileges are determined for encoding according to **SMI** syntax rules for **SNMP MIB**, I have used the standard **MIB** compiler (**SMIC**-compiler[13]) for compiling the **SMIB** which supports **SNMP** and **SNMP**v2.

6. Network Security Protocols with SMIB

For providing confidentiality and integrity of messages, the adoption of a lower layer security protocol such as Network Layer Security Protocol **(NLSP)**[10] or Transport Layer Security Protocol **(TLSP)**[11] may be appropriate. In this **EDI** security management model, The **TLSP** is adopted for confidentiality and integrity services because it is easier to access the **SMIB** and/or **MIB** in Transport Layer than in Network Layer. The parameters to be used during the establishment of security associations are also stored as security attributes within **SMIB**. Some instances of object groups such as data confidentiality and data integrity group can be accessed by the **TLSP** protocol[12]. Fig. 3 shows which protocols use the **SMIB** and/or **MIB** objects. Also **EDI** applications and security management application can access the **SMIB** and/or **MIB** through **SNMP** application interface. As an example a security manager can get and set a value of a security object "securityClassification" in Message Security Labeling Group as the following.

$snmpget mgr_commty ms_host private.x.msgSecLabGroup.securityClassification

private.x.msgSecLabGroup.securityClassification : DISPLAY STRING-(ascii) : Confidential

$snmpset mgr_commty ms_host private.x.msgSecLabGroup.securityClassification octetstring

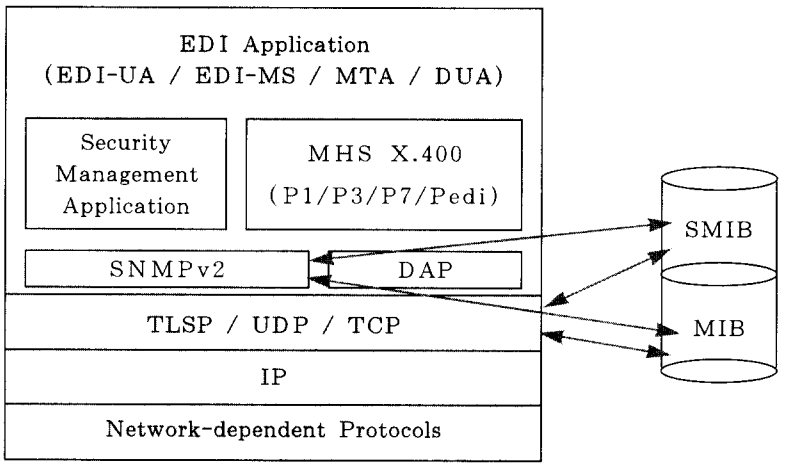

Fig. 3. SMB and Protocols

"Secret"

mgr_commty ms_host private.x.msgSec LabGroup.securityClassification : DISPLAY STRING-(ascii): Secret

7. Conclusion

EDI system needs appropriate security management for controlling the security objects for its security services and mechanisms. So far, I have reviewed the security elements in standard **EDI** system, and designed security management model for **KT-EDI** system. Also, I have identified a number of security management objects in the **EDI** system based on standards, and designed the common **SMIB** for the security management of the **EDI** system components such as **UA**, **MS**, and **MTA** using **SNMPv2 SMI**. By using **SNMPv2** protocol with **SMIB**, we can perform the key management, access control, monitoring and control **EDI** system securely. However **SNMPv2** does not address threats of denial of service and traffic analysis. Nevertheless, **SNMPv2** protocol ensures that the basic security requirements defined in **ISO** 7498-2 security architecture for an **EDI** system.

References

1. T. K. Apostolopoulos, and V.C. Daskalou, **SNMP**-based Network Security Management using a Temporal Database Approach, Information Systems Security edited by S.K.Katsikas, and D.Gritzalis, Chapman & Hall, 1996.
2. Panos Katsavos, and Vijay Varadharajan, A secure Frame Relay service, Computer Networks and **ISDN** Systems, 26 (1994).
3. William Stallings, Network and Internetworking Security, Principles & Practice, IEEE Press, 1995.
4. Information Processing Systems - Open

Systems Interconnection Reference Model - Security Architecture, **ISO** 7498-2, 1988.

5. **ETRI** Research Report, Development of a Security Service Server for Information Security Services, Research report, 1995.12.
6. J. Case, K. McCloghrie, M.Rose & S. Waldbusser, Structure of Management Information for Version 2 of **SNMP**v2 (RFC1902), January 1996.
7. ITU-T, X.435 Message Handling Systems : Electronic Data Interchange Messaging System, 1992.8. ITU-T, X.400 Message Handling Systems : Systems and Service Overview, 1992.
8. ITU-T, X.400 Message Handling Systems : Systems and Service Overview, 1992.
9. **UN / EDIFACT** (Electronic Data Interchange for Administration, Commerce and Transport) Syntax Rules (**ISO** 9735), **UN / ECE, WP**.4., March 1993.
10. **ISO / IEC CD** 11577, Network Layer Security Protocol (**NLSP**), Dec. 1991.
11. **ISO / IEC DIS** 10736, Transport Layer Security Protocol (**TLSP**), Dec. 1991.
12. ITU-T, X.509 The Directory- Authentication Framework, 1988.
13. David Perkins, and Evan McGinnis, Understanding **SNMP MIB**s, Prentice Hall, Inc, 1997.
14. J.H.Lee, Data Structure of Secure **EDI** System, The 2nd Symposium proceedings of Secure **EDI**-related technology, 1996.

□ 著者紹介

박 태 규

1980년 10월 경북대학교 전자공학과(전산전공) 공학사
1981년 2월 ~1982년 12월 한국국방연구원 전산센터 연구원
1982년 12월 ~1992년 2월 한국전자통신연구소 선임연구원
1989년 8월 충남대학교 대학원 전산학과 이학석사
1996년 2월 성균관대학교 대학원 정보공학과 공학박사
1997년 1월 ~1998년 1월 Univ of Westerm Sydney, Research Fellow
1992년 3월 ~현재 한서대학교 컴퓨터과 부교수

※ 주관심분야 : 컴퓨터 네트워트 보안, 병렬암호화