A Policy-Based Security Management Architecture Using XML Encryption Mechanism for Improving SNMPv3*

Choong Seon Hong and Joon Heo

School of Electronics and Information, Kyung Hee University
1 Seocheon, Giheung, Yongin, Gyeonggi 449-701 KOREA
cshong@khu.ac.kr, joon@networking.khu.ac.kr

Abstract. Simple Network Management Protocol (SNMP) is the most widely-used network management protocol for TCP/IP-based networks. The functionality of SNMP was enhanced with the publication of SNMPv2. However, both versions of SNMPv1 and SNMPv2 lack security features, notably authentication and privacy. The SNMPv3 solves these deficiencies but it has some inefficiency to deal with the access, service refusal, or unstable action. On the other hand, XML is being used to describe components and applications in a vendor and language neutral. In this paper, we propose a policy-based SNMP security management architecture using XML. We propose a secure network management protocol that adopts the policy-based network management and the XML security features to the existing SNMPv3.

1 Introduction

Simple Network Management Protocol (SNMP)[1] has become the most widely-used network-management tool for TCP/IP-based networks. SNMPv1 defines a protocol for the exchange of management information, but does much more than that. It also defines a format for representing management information and a framework for organizing distributing systems into managing systems and managed agents. In addition, a number of specific data base structures, called management information bases (MIBs), have been defined as part of the SNMP suite; these MIBs specify managed objects for the most common network management subjects, including bridges, routers, and LANs. However, SNMPv1 has the security problem for SNMP message as for such structure. Therefore, working group presented SNMPv2 [2][3], SNMPv3 [2][6] and basic security threat problem solved presenting User-based Security Model (USM) [4] in message processing. But SNMPv3 has some limitations causing unauthenticated access, denial of service, and unstable action [5]. So in this paper, we propose a policy-based SNMP security management architecture to improve SNMP security using XML's security operation. We will discuss the issues and design objectives for policy-based

* This work was supported by University ITRC Project of MIC.

A. Laganà et al. (Eds.): ICCSA 2004, LNCS 3043, pp. 755–764, 2004.

SNMP security management. For proposing the security management architecture, we will define functional structure and describe the detail processing flows in terms of manager and agent, which will be shown how the security policy is managed and enforced. In addition, we describe the functional structure of manager and agent in detail. In addition, we will show the procedure for enforcing XML-based security policy on the proposed functional structure of manager and agent using sequence diagram in depth. We will also define an example of XML policy template and propose the schemes for management of XML encryption and authentication in terms of security management.

2 SNMPv3 Structure and Use of XML in Network Administration

2.1 SNMPv3 Basic Structure

SNMP working group has defined the SNMP structure in RFC 2271[6][7]. Basically, SNMP gives some functions to control various network elements and to monitor status of network elements using Structure of Management Information (SMI), Management Information Base (MIB), and protocol. SNMPv3 basic structure is embodied by discrete SNMP entities' interaction. Each entity is embodied as the module that has single SNMP engine, exchanges message through these engines, or processes encryption, decryption and authentication to access target entities. In SNMP basic structure, the roles of SNMP entities are as follows:

- Dispatcher allows for concurrent support of multiple versions of SNMP messages in the SNMP engine.
- Message Processing Subsystem is responsible for preparing messages to be transmitted and for extracting data from received messages.
- Security Subsystem provides security services such as the authentication and privacy of messages.
- Access Control Subsystem provides a set of authorization services that an application can be used for checking access rights.
- Command Generator initiates SNMP Get, GetNext, GetBulk, and/or Set PDUs and processes the response to a request that it has generated.
- Command Responder performs the appropriate protocol operation using access control and will generate a response message to be sent to the originator.
- Notification Originator monitors the particular events or conditions, and generates Trap and/or Inform messages based on these events or conditions.
- Notification Receiver listens for notification messages and generates response messages when it receives the message containing Inform PDU.
- Proxy Forwarder forwards SNMP messages.

SNMPv3 protocol achieves monitoring function about event or situation, message send-receive through ditto SNMP entities.

2.2 The Use of XML in Network Management

XML[7] that alternates HTML and smooth SGML (Standard Generalized Markup Language) makes ease information transmission using HTTP and document, which was established by standard in 1998 in W3C(World Wide Web Consortium). One characteristic of XML is to separate the contents of document from the expression. Expression can be defined using XSL (eXtensible Stylesheet Language) and expressed identical document by other image changing this style seat. In addition, the expression way can be changed to the document of other form through data conversion capability of XSLT (eXtensible Stylesheet Language Transformations). Also, XML can define structure of document using XML DTD and XML Schema. We can manipulate the XML document using the standard APIs such as access, store and extraction. Therefore, XML has many advantages as a standard that describes management information in network management. If we use XML tool that converts SNMP MIB to XML document to describe information model in network management system, we can easily define and conveniently model the necessary information model. In addition, XML can be used to exchange information without any dependency to hardware or software platform because XML is available in nearly all kinds of hardware and software platform in these days. With these advantages, XML can be a good alternative for development of network management system in terms of scalability, flexibility and efficiency. Considering XML-based network management system, we can easily define the management information using XML instead of new development of complex protocol, which is exchanged between manager and agent. However, there is a controversial scalability problem because the XML-based managed information can not be used in the agent that does not support XML. Because of such a scalability problem, we generally use the SNMP to XML or XML to SNMP translator and gateway that support communications between SNMP-based agent and XML-based manager.

3 Related Works

3.1 COPS-PR and PIB for Policy-Based Network Management

We focus on configuration management based on IETF standards. The IETF has defined a policy framework consisting of management interfaces to define policies, repositories to store policies, policy decision points (PDPs) to evaluate policies, and policy enforcement points (PEPs) to enforce policy decisions Based on this framework, the IETF has standardized a policy core information model (PCIM, PCIMe) that can be used to define policies, to store policies in repositories and evaluate polices at PDPs. In addition, COPS-PR (Common Open Policy Service for Policy Provisioning) [9][10] was standardized for transferring policy decisions between PDP and PEP the protocol. The structure of configuration information carried by COPS-PR is defined in Policy Information Base (PIBs). The PIB is a conceptual tree namespace of Provisioning Classes (PRCs) and Provisioning Instances (PRIs). There may be multiple instances (PRIs) of any PRC. The

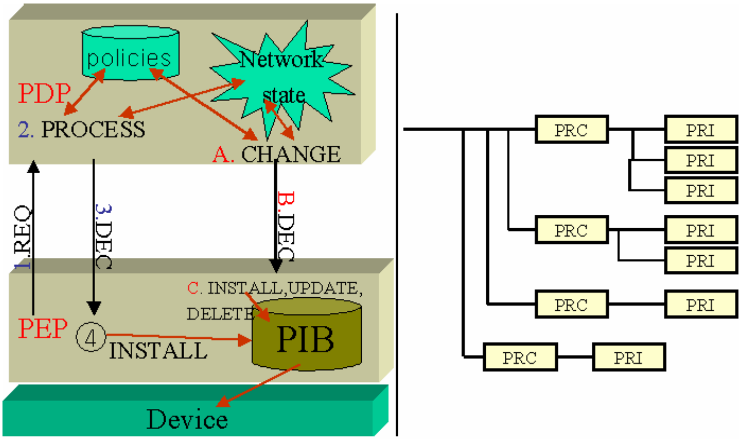


Fig. 1. COPS-PR operation model and PIB tree

language for defining PIBs has been standardized as the Structure of Policy Provisioning Information (SPPI). COPS-PR enforcement process(fig. 1) is as follows:

1. XML PEP requests necessary policy to PDP.
2. XML PDP interprets and analyzes the policy request message from PEP.
3. XML PDP examines the current network state, inquires policy, decides suitable policy and sends the appropriate policy to PDP through Decision (DEC) message.
4. XML PEP installs the received policy from PDP to PIB and PIB controls device according to the policy.
5. XML Also, if network state is changed, PDP inquires necessary policy and send it to PDP through DEC message. On receiving the changed policy from PDP, PEP does same actions as (4).

subsectionConversion of management information and SNMP MIB into XML As described in previous section 2, XML is flexible for expressing the various logical configurations because DTD is not fixed. XML is being used to describe components and applications in a vendor and language neutral way. XML has been widely adapted and has been token an important role in network management. In this section, we describe the way to represent managed data using XML. A new representation for system management data called the Common Information Model (CIM) [11] has been proposed by DMTF. There are two fundamentally different models for mapping CIM to XML. One is a Schema Mapping in which the XML Schema is used to describe the CIM classes and CIM instances are mapped to valid XML Documents for that schema. The other is a Metaschema Mapping in which the XML schema is used to describe the CIM Metaschema and both CIM classes and instance are valid XML documents for that schema.

Similarly, there are two different models for mapping MIB to XML [12]. One is a model-level mapping and the other is a Metamodel-level mapping. Model-level mapping means that each MIB variable generates its own DTD fragment and the XML element name of which are taken directly from the corresponding MIB element names. It is for it to have been good that a person reads a merit of model-level mapping and understands, but it is to have needed a lot of DTD because a disadvantage must write the DTD that MIB is each. Metamodel-level mapping means that the DTD is used to describe in a generic fashion and the notion of MIB variables. MIB element names are mapped to XML attribute or element values, rather than XML element names. Of cause, there is a disadvantage of lack of readability but it makes one to easy development of conversion program between MIB and XML.

4 A Proposed System

4.1 Issues and Design Objectives

In this paper we adopt XML Policy-based security and XML encryption, decryption, and authentication for enhancing the existing SNMP protocol in terms of security. We use COPS-PR to transfer the policy-related information. By adding or removing PRIs, the PDP can implement the desired polices to be enforced at the device. It is important to highlight that the police of each PIB are predefined. Mapping PIB to XML can add PRC dynamically. It supports dynamic adaptability of behavior by changing policy without stopping system and minimizes the rigidity of PIBs. We have three principles for design policy-based SNMP security management architecture as follow:

- Uses Xml Policy-based security and copes on a network attack flexibly.
- Adds XML module to the existing SNMP module for compatibility with the existing protocol.
- Copes with the denial of service which is unfavorable in terms of secure communication over existing SNMP protocol by XML security policy application.

4.2 Proposed Architecture and Operation

Our policy-based SNMP security management architecture composed of several entities as follows:

- XML policy repository: Policy Database server maintaining XML security connection information. PEP determines the appropriate security policy referring to XML policy repository's information. On the other hand, XML policy enforcer in agent side takes appropriate enforcement action for enforcing the designated security policy.
- XML policy decision: It selects the most appropriate security policy among the policies maintained in XML policy repository, composes the PDU embedding the selected security policy and transmits the composed PDU to agent.

- XML encryption: It encrypts the PDU converted to XML at manager.
- XML decryption: It deciphers the PDU encrypted in XML at agent.
- XML parser: It converts SNMP PDU to XML. It is not necessary to convert the SNMP PDU that is generated from XMP policy repository to XML. The XML parser is used to convert the application specific PDU to XML.
- XML interpreter: It converts the XML PDU to SNMP PDU and gives the converted PDU to applications.
- XML Policy enforcer: As an entity at agent application area, it enforces security policy.

With these entities, the five processes at manager to enforcing security policy are as follows:

[MS1] The XML policy decision at SNMP applications as showing in Fig. 1 selects the most reasonable policy from XML policy repository and transfer the selected policy to dispatcher.

[MS2] On receiving the security policy, the Dispatcher sends the message for assignment of SNMP version assignment to Message Processing Subsystem.

[MS3] Message Processing Subsystem determines the appropriate SNMP version of target agent and transfers the message to Security Subsystem.

[MS4] XML parser at Security Subsystem converts PDU to XML and encrypts the XML for enforcing security.

[MS5] After encrypting PDU with XML, Security Subsystem applies the appropriate security models such as USB model and finishes the security related processes in manager.

Figure 2 shows the SNMP's architecture in Manager with XML security function. The security achievement processes at agent shown in Fig. 3 are as follows:

[SA1] Access control subsystem authenticates the request of manager.

[SA2] After having finished the approval process, Security Subsystem decrypts the message received from manager and applies an appropriate SNMP security model such as UBS model.

[SA3] XML decryption at Security Subsystem decrypts the PDU encrypted with XML.

[SA4] XML interpreter at Security Subsystem converts the decrypted PDU to SNMP PDU according to the rules maintained in XML policy repository and transmits the interpreted SNMP PDU Message processing Subsystem.

[SA5] Message processing Subsystem extracts data from PDU and send the extracted data to XML Policy enforcer at SNMP application.

[SA6] XML Policy enforcer applies the security policy to appropriate device and finishes security achievement process at Agent.

4.3 An Example of XML Policy

We proposed a scheme that applies XML to policy definition, which makes it more convenient to share the XML-based policies with difference management

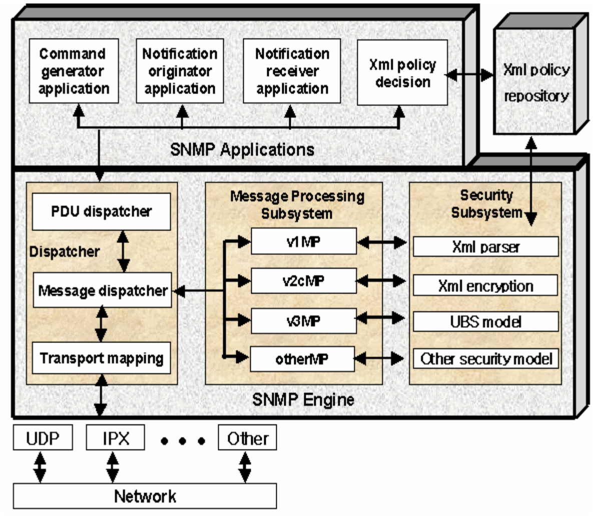


Fig. 2. SNMP manager architecture

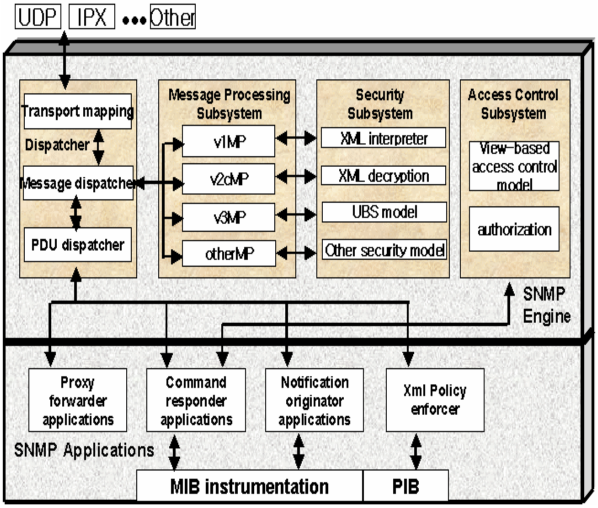
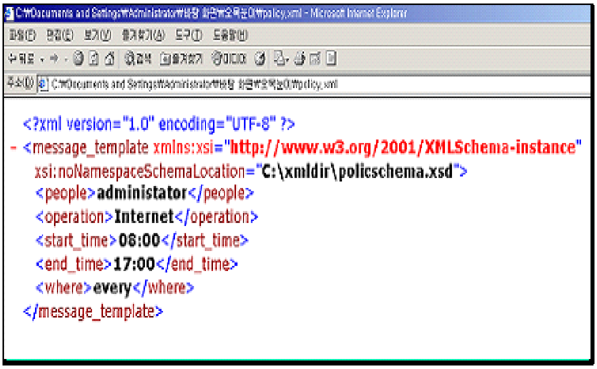


Fig. 3. SNMP agent architecture

systems and to extend the existing policies. In this section, we describe the policy template for defining policy and the PIB model implemented using XML. Fig. 4 shows an example of XML policy template. This template consists of general elements and each element is corresponding to an object in network management domain.

- people: people generally describe a worker.
- operation: operation represents the processing item.
- start time: start time represents the process initiation time.
- end time: end 1time describes the process finish time.
- where: there describes a working place.

The significant of policy shown in Fig. 4 is ”administrator can login at any places and can use the Internet during working hour (from 08:00 to 17:00).”



```
 <?xml version="1.0" encoding="UTF-8" ?>
- <message_template xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="C:\xmldir\policschema.xsd">
   <people>administator</people>
   <operation>Internet</operation>
   <start_time>08:00</start_time>
   <end_time>17:00</end_time>
   <where>every</where>
 </message_template>
```

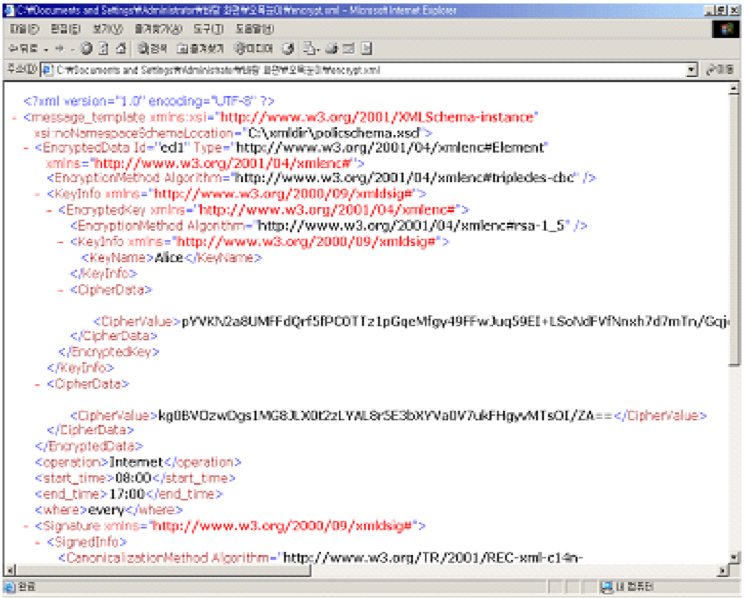
Fig. 4. An example of XML policy template

4.4 XML Encryption and Authentication

Fig. 5. shows the results of encryption and authentication of an XML policy. In the case of encryption, we encrypted the user who was specific element in encryption of XML. An error message can be generated in authentication process if digital signature value is different. If the digital signature is not correct, the data from unauthorized user is disposed. For XML encryption and authentication, we have used XSS4j of IBM alphaworks, SUN J2sdk1.4.1, Xerces 2.3.0 and Xalanj2.4.1 of apache. The operating system was based on Window 2000 professional.

First of all, Encryptor carries out the following process in order to let you encrypt as EncryptedData or EcryptedKey [8].

1. Selects the cryptographic algorithm which is going to apply to Data.
2. Gets key value spent on encryption and selectively marks this.
3. Encrypts data.
4. Designs structure of EncryptedType.
5. Carries out EncryptedData work and finishes an encryption process.



```
  <?xml version="1.0" encoding="UTF-8" ?>
- <message_template xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="C:\xmldir\policschema.xsd">
  - <EncryptedData Id="ed1" Type="http://www.w3.org/2001/04/xmlenc#Element"
      xmlns="http://www.w3.org/2001/04/xmlenc#">
      <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc" />
    - <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
      - <EncryptedKey xmlns="http://www.w3.org/2001/04/xmlenc#">
          <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5" />
        - <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
            <KeyName>Alice</KeyName>
          </KeyInfo>
        - <CipherData>

              <CipherValue>pYVKN2a8UMFFdQrf5fPC0TTz1pGqeMfgy49FFwJuq59EI+LSoNdFVfNnxh7d7mTn/Gqj
          </CipherData>
        </EncryptedKey>
      </KeyInfo>
    - <CipherData>

          <CipherValue>kg0BVOzwDgs1MG8JLX0t2zLYAL8r5E3bXYVa0V7ukFHgyvMTsOI/ZA==</CipherValue>
      </CipherData>
    </EncryptedData>
    <operation>Internet</operation>
    <start_time>08:00</start_time>
    <end_time>17:00</end_time>
    <where>every</where>
  - <Signature xmlns="http://www.w3.org/2000/09/xmldsig#">
    - <SignedInfo>
        <CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-
```

Fig. 5. The encrypted XML

5 Conclusion and Future Works

This paper proposed the policy-based SNMP security management architecture. We adopted the XML policy-based security and an XML encryption function for a security elevation of SNMP protocol. We identified and discussed some issues and guidelines for security management and defined several entities that are parts of security management architecture. In addition, we proposed the security enforcement process from the perspectives of manger and agent in depth. We also described the functional structure of manager and agent for enforcing security policy between XML-based manager and SNMP-based agent in depth. In addition, we defined XML policy template and proposed the schemes for XML encryption and authentication to support the policy-based SNMP security management using XML. Now, we can protect the network from malicious attacks using our security management architecture. However, we need to verify the scalability of the proposed security management architecture and need to study on a security algorithm suitable for or compatible with SNMP protocol sooner or later.

References

1. RFC 1157, "Simple Network Management Protocol", May 1990
2. William Stallings, "SNMP, SNMPv2, SNMPv3, and RMON 1 and 2, 3rd Edition", May 2001
3. RFC 1902, "Structure of Management Information for Version2 of the Simple Network Management Protocol(SNMPv2)", Feb. 1996
4. RFC 2574, "User-based Security Model (USM) for version3 of the Simple Net-work Management Protocol (SNMPv3)", April 1999
5. http://cert.org/, "CERT Advisory CA-2002- 03 Multiple Vulnerabilities in many Implementations of the SNMP", June 2003
6. William Stallings "SNMPv3: A Security Enhancement for SNMP", IEEE Communications Survey, Vol. 1, No.1, 1998
7. W3C, "Extensible Markup Language", http://www.w3.org/XML/
8. Takeshi Imamura, Blair Dillaway, Ed Simon, "XML Encryption Syntax and Processing", W3C, Dec. 2002
9. D.Durham, et al. "The COPS(Common Open Policy Service) Protocol", Jan. 2000
10. K.Chan et al.,"COPS Usage for Policy Provisioning", IETF, RFC3084, Mar. 2001