

A Secure Communication Protocol for Telecommunications Management Networks

Lin, Chern-Tang* (林宸堂) and Shieh, Shih-Pyng (謝續平)

Department of Computer Science and Information Engineering
National Chiao Tung University, Taiwan, ROC 30050
Email: ssp@csie.nctu.edu.tw

Abstract

Telecommunications Management Network (TMN) is a standardized concept for constructing network management systems for telecommunication networks and services. To simplify the development of TMN-based network management applications, a TMN platform is needed to execute the common management tasks and provide application programming interfaces (APIs) to programmers. In this research, we investigate the security requirements of communications between TMN-based systems and propose a secure message exchange protocol embedded in the TMN platform. The proposed protocol can establish secure communication channels between TMN platforms, so that the applications built on the platforms do not need to worry the security attacks from hostile outsiders. The protocol needs only one message combined with the synchronized nonce scheme to authenticate the validity of messages. And, the privacy of management information carried within transmitted messages is protected by a private-key cryptosystem. Thus, under limited additional cost, our proposed protocol can support high-quality security services for TMN platforms. And, in addition to applying this protocol to telecommunications management networks, it is also able to apply to the Internet management systems that need high security and low cost for message exchanges.

Keywords: Network Management, TMN platform, Network security, Authentication protocol

1. Introduction

Telecommunications Management Network (TMN) is a standardized concept for constructing network management systems for telecommunication networks and services [1]. It also offers the standardized exchange of management information between various types of network management systems and/or telecommunications equipments. Although the international standards supporting TMN have been established, it is difficult to implement a system based on the TMN concept due to its complicate components. It is therefore desirable to develop a TMN platform that helps programmers implement TMN-based management applications.

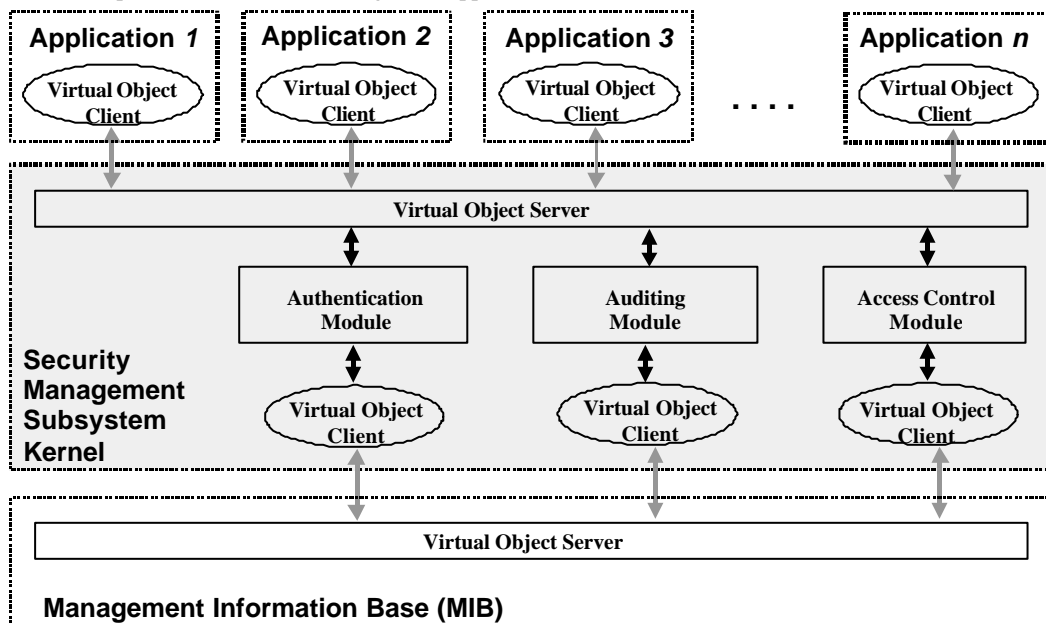


Fig. 1 The diagram of modules in the security management subsystem

A TMN platform has been developed in the “Platform and Tools for Developing an Integrated Network Management System” project supported by Telecommunication Laboratory, Ministry of Communications, Taiwan [2, 3, 4]. The platform consists of five subsystems: accounting, configuration, fault, performance, and security management subsystems [5]. These subsystems provide all management-related and integrated APIs (application programming interfaces) for programmers to develop TMN-based applications. APIs are composed with virtual objects that are similar to RPCs (remote procedure calls) implemented as objects. For example, in this project, the security management subsystem provides 29 APIs to support the management of authorization facilities, access control, encryption and key

1. This work was supported in part by the Telecommunication Laboratory, Ministry of Communications, Taiwan, under contact TL-84-3301.

management, authentication, and security logs (see figure 1 for the diagram of modules in the security management subsystem) [6].

Figure 2 illustrates the relationship between the platform and other components of the entire management network. A TMN-based management system (TMS) can communicate with other management systems, or manage network elements (NEs), such as transmission systems, switching systems, multiplexers, via the data communication network (DCN). DCN is an open and general-purpose communication network. Obviously, all messages transmitted on DCN take the risk of eavesdropping, personating, or manipulating. Therefore, TMSs need a secure communication schemes to authenticate the validity of the received management information and guarantee its confidentiality while the information is being transmitted over DCN. The requirements for the secure scheme of TMN-platforms are:

- the distribution and management of secret information must be efficient since the whole DCN crosses multiple network-domains, and
- due to the long distance between the source and destination of messages, the strength of adopted cryptosystems must be strong enough to prevent attacks from outsiders. Especially, the chosen-plaintext-attack can not be ignored because some messages in TMN are regular and their contents are predictable with monitoring for a long time.

In order to eliminate the cost of developing applications against outsiders' hostile attacks, there are some solutions that are designed for the security of datagram oriented protocols like IP, such as IPsec for IPv6 [7-10] or SKIP [11]. Based on these schemes implemented in the IP layer, the programmers of applications will not worry about the security of messages transmitted through TCP/UDP. But we did not consider adopting these technologies because the popular and standardized solutions for IPv6 or SKIP were still unavailable while we were designing the TMN-based platform at 1994. And, most importantly, to promote the portability of TMN-platforms for various operating systems, it is suggested to develop all features, including secure communication schemes, of platforms on the application layer.

A well-known network management protocol which provides the security features on the application layer is the version 2 of the simple network-management protocol (SNMPv2) [12]. SNMP was originally developed to provide a basic, easily implemented network-management tool for TCP/IP-based environment [13]. SNMPv2 adopts the secret keys previously sharing between the sender and receiver to authenticate the origin of a message and guarantee its privacy, but the security of the mechanism is doubted. Because, whenever SNMPv2 would invoke the procedure of renewing the current secret keys, the distribution of new secrets is protected by the current key. That is, if a secret key is compromised, all following secrets will be easily disclosed. Although there are other secure communication schemes, such as those based on private-key cryptosystems [14-16], which are more secure than those mentioned above, these schemes are not efficient because, generally, they need more messages to authenticate each other and exchange their secret information.

In this paper, we propose a secure TMN communication protocol (STCP) which uses different encryption keys, based on a private-key cryptosystem, for each message to guarantee the privacy of management information. The message-oriented encryption key is distributed within the same message and protected by a public-key cryptosystem. The public-key cryptosystem is also used to help the message receiver authenticate the origin and validity of this messages. In fact, our protocol is similar to the X.509 one-way authentication protocol. In both protocols, a user can verify the origin of a received message and disclose the secret contents carried within the same message. However, in the X.509 one-way authentication protocol, the sender cannot determine whether the valid user has received the message or not. Furthermore, unless a precise and reliable clock-synchronization is used, this protocol is vulnerable to the message replay attacks since it uses the timestamp to verify the freshness of a message. The problems of the X.509

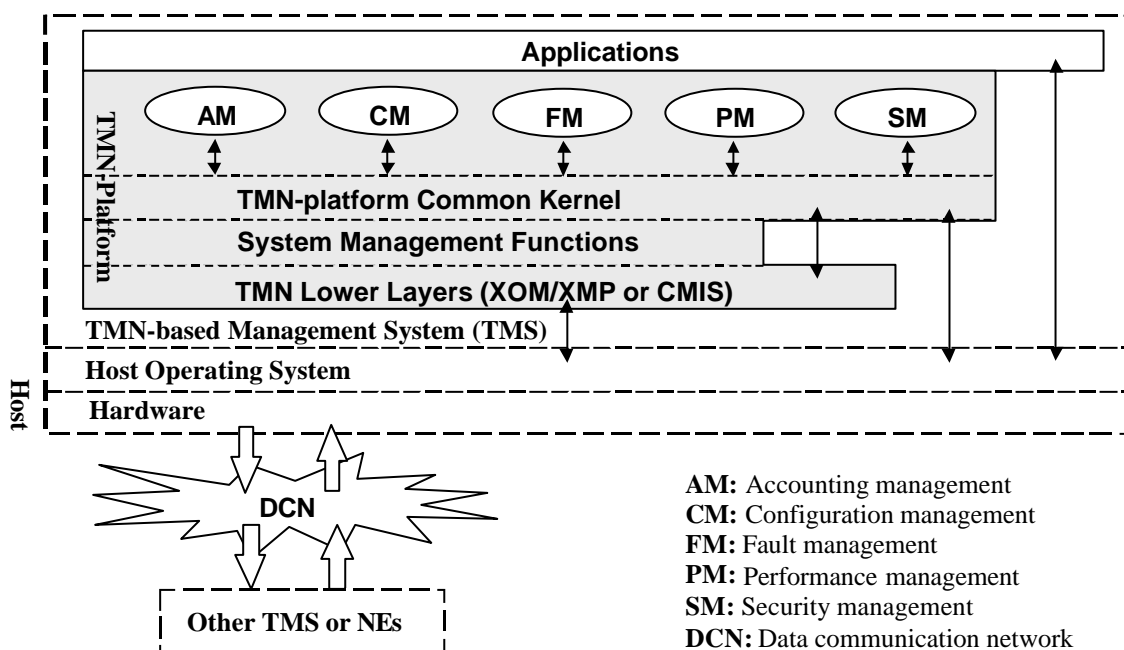


Fig. 2 The architecture of TMN-based management system.

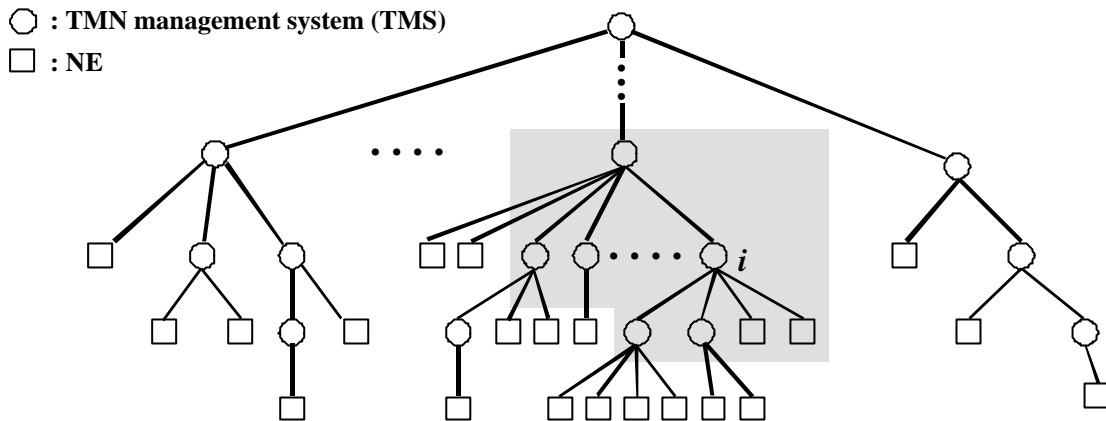


Fig. 3 Logical management-relationship of Taiwan telecommunication network infrastructure

one-way protocol are overcome in STCP by adopting a synchronized-nonce scheme. In addition to applying this proposed protocol to telecommunications management networks, it is also able to be applied to the internet management systems which need high security and low cost for each message exchange.

In section 2, we will describe our TMN environment and the design issues of the proposed protocol. Then, the detailed description of STCP and the synchronized-nonce scheme are presented in sections 3 and 4, respectively. Finally, the last two sections state the security analysis of STCP and the conclusion.

2. The Environment and Design Issues

The overview of Taiwan TMN is illustrated in Figure 3. Each circle denotes a TMN management system (TMS) described in Figure 2. Each TMS must manage at least a TMS/NE (called an agent). Note that the logical connection is different from the topology of the physical DCN which may consist of various network architectures and even cross a number of network domains. Thus both TMSs which have a same parent TMS may belong to different network domains. However, the difference will not affect the design of the communication protocol because the entire TMS is implemented at the application layer of the OSI network architecture, that is, only the logical connection needs to be considered.

For each TMS, such as node i in Figure 3, it can only connect directly to its children TMSs/NEs, parent TMS, and sibling TMSs, shown in the shadow area. For simplicity of presentation, in the rest of the article, the environment of management networks is illustrated in Figure 4. For each TMS, called *local system* (LS), its parent TMS and sibling ones are called *manager system* (MS) and *peer systems* (PSs), respectively. On the other hand, LS's all child TMSs and NEs are called *agent systems* (ASs). The line denotes that both end nodes of the line can directly exchange management information. Note that, although LS and PS_i have no management-relationship, there exists a line between them due to the practical need of our system.

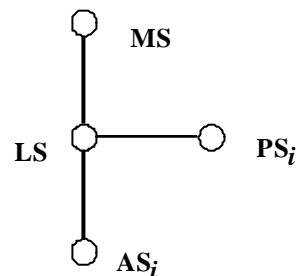


Fig. 4 The simplified environment of the TMN

We summarize five security-related issues for the design of the secure message exchange protocol for the TMN-based network management system.

- The authenticated entities are TMSs instead of the users who invoke management applications in TMSs. And the users are properly authorized privileges to access the specified functions by the security management subsystem of TMS. Thus, we will not discuss the management of users, but will focus on the design of the authentication protocol which can identify TMSs and guarantee the security of message exchange between TMSs.
- The number of TMSs that LS can connect to is limited. They include LS's parent system, sibling TMSs, and child TMSs. Thus it is possible to adopt a simpler security-mechanism instead of a complicated centralized scheme, such as the conventional trusted third-party mechanism [14, 15].
- Some messages transmitted between systems are well-known. These messages are transmitted periodically in the predefined format given by TMN standard for management purposes. If the secret-key is used as a long-term key to encrypt messages, the TMS will take the risk of chosen plain-text attacks.
- The number of transmitted messages for a management task is different. In most cases, a regular management task only needs two/three messages exchanged. In some cases, only one message is needed, for example, the alarm reporting from NE to TMS. In other cases, more messages may be needed. For example, some accounting management tasks may need many messages that contain a large amount of data. If each management task needs two or more messages to complete the

authentication of the participates and the distribution of shared secrets, it will impose a significant overhead (the cost will be heavier while the physical connection crosses multiple domains).

- There is no a precisely synchronized clock to be supported as the timestamp of the secure communication protocol in our environment. The clock-synchronization is one of important services in the configuration management subsystem of TMN-platforms. However, a precise and reliable clock-synchronization protocol must entirely depend upon the secure exchange of synchronization messages. Thus many clock-synchronization protocols support additional authentication schemes to guarantee the security of message exchange [17]. In our systems, the attack-free environment for synchronizing the system clock is provided by the proposed STCP. That is, in our TMN-platforms, it is impractical to verify the freshness of a received message based on the timestamp which is usually used in conventional secure communication protocols.

Based on the five design issues described above, we survey conventional secure communication protocols [3] and conclude that, among all the well-known protocols, the X.509 one-way protocol of ISO authentication framework is a better choice [18]. As shown in Fig. 5, the protocol needs only one message and uses the public-key cryptosystem to authenticate both the sender and receiver and guarantee the confidentiality of data. Therefore, by a single message, the receiver can safely get the sensitive information (denoted as *Data* in Fig. 5) without an additional authentication phase before this message exchange. However, it still has some drawbacks. First, the sender cannot determine whether the receiver really has received the message because no acknowledgment is replied from the receiver (this problem is resolved in the X.509 two-way authentication protocol). Second, since the protocol uses the timestamp to prove the freshness of a message, it is vulnerable to replay attacks [19]. If the nonce scheme (the challenge-response scheme) is applied to replace the timestamp, the protocol needs, at least, two messages (the X.509 three-way authentication is a similar solution). This provides better security, but conflicts with the requirement of minimal communication cost. Furthermore, if the amount of sensitive data for the receiver is large (usually this is the case), it is impractical for the sender to encrypt a big chunk of data with the receiver's public key (K_{PB}), and then encrypt the resulting big ciphertext with the sender's secret key K_{SA} .

In this paper, we propose a new secure TMN communication protocol (STCP), which overcomes the three drawbacks described above, and has the advantages of easy management and low communication cost.

3. The Secure TMN Communication Protocol

The message structure of STCP looks similar to X.509. STCP includes a new verification scheme to verify the freshness of messages, namely, the synchronized nonce (SN) scheme. The SN scheme is used to replace the timestamp of X.509 (see the next section for the details). The basic message of STCP is presented in Figure 6. The message consists of three segments: certificate, authentication, and data segments.

Certificate segment $Cert_A$:

The structure of the certificate segment is similar to X.509. It contains the public-key-related information about the message sender, *A*, such as *A*'s identity, *A*'s public key, the valid period of the key, the version of the protocol, the algorithm and the version of the cryptosystem, and so on. TMS's certificate data is generated and managed by its parent TMS, namely MS. LS can get its certificate with off-line methods, such as embedding the information within the setup disk as LS is created. At run time, if two sibling TMSs want to communicate with each other, they can get the other's certificate from MS with STCP. Since the management of certificates is widely known, we will not describe herein how to distribute/repel these certificates [20].

Authentication segment $\{T_A, h(SN_i), ID_B, \{MsgType, M\}_{K_{PB}}\}_{K_{SA}}$:

The authentication segment in a message is to identify and proof the origin and the validity of the message with a public-key cryptosystem (512-bit RSA is adopted in our system). In addition, this segment also carries a "short" sensitive information. (The length of the information must be short since the computation cost of 512-bit RSA is high.) Since the segment is encrypted with the sender *A*'s secret key, K_{SA} , it is clear that only *A* can produce the correct segment (the purpose is similar to the digital signature). SN_i is a synchronized nonce that is different in each message, and can be predicted only by *A* and *B*. With $h(SN_i)$, the receiver *B* can verify the freshness of the segment, and determine whether the segment is a replay or not. Although every body can decrypt this segment with the sender's public key, no body can disclosure SN_i because $h()$ is an irreversible one-way function (we adopt MD5 in our system). On the other hand, T_A , the local time at *A*, indicates roughly when the segment was issued by *A*. In practice, although T_A is not required, it can help the receiver *B* determine whether the message has been maliciously delayed or not. Pre-synchronized clocks between TMSs is not needed in the scheme.

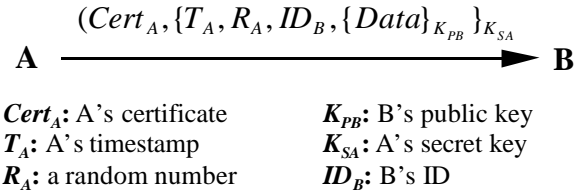


Fig. 5 X.509 one-way protocol

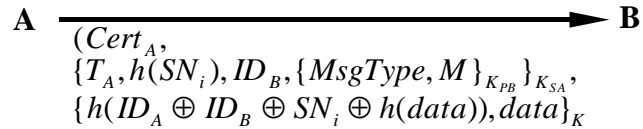


Fig. 6 The secure TMN communication protocol (STCP)

To transmit secret information to the receiver B , $\{MsgType, M\}$ is encrypted with B 's public key K_{PB} , where the variable $MsgType$ indicates the message type and M denotes the secret information. There are three message types: (1) the message is a control message for maintaining STCP (e.g., resynchronizing SN_i . See section 4(E)), (2) the message is for a regular management task, or (3) the message is an urgent message which cannot be lost (See section 4(D)).

Since the public-key cryptosystem (RSA) is strong, it is secure to use the public-key pair as a long-term key. However, it is impractical to encrypt/decrypt lots of data with 512-bit RSA, thus we cascade a data segment behind the authentication segment to reduce the cost of encrypting/decrypting huge data. And the secret information M contains the related information of encrypting the following segment, such as the encryption algorithm, the key length, and the encryption key.

Data segment $\{h(ID_A \ ID_B \ SN_i \ h(data)), data\}_K$:

The real management information $data$ exchanged between TMSs is encapsulated in this data segment. The segment is encrypted with K which is one private key of a predefined symmetric cryptosystem, such as DES [21] or IDEA [22]. The private key K is generated by the sender and different for each message exchange. The key K is distributed and protected within the secret information M of message's authentication segment. The scheme thus can securely distribute the private key. In addition, since the computation overhead of the private-key cryptosystem is much smaller than that of the public-key cryptosystem, the scheme can significantly improve the efficiency of encrypting/decrypting a large amount of data.

Since the certificate, authentication, and data segments are simply concatenated together within a message, to prevent a segment being replaced during transmission, a verifiable header $h(ID_A \ ID_B \ SN_i \ h(data))$ is needed to establish the link between this data segment and the authentication segment. Because SN_i is unpredictable by other outsiders, nobody can forge the header. Thus a replaced data segment can be detected by comparing both SN_i 's in the data segment and the authentication segment. And, if an intruder compromises the data encryption key K , a replaced or modified data can also be detected through the consistence between $data$ and $h(data)$.

Although our protocol uses only one message, its security, compared with the X.509 one-way protocol, is significantly improved. The public-key-based authentication segment with the synchronized-nonce can verify the origin of a message and prevent message replay without a precise clock-synchronization. In addition, the encryption key K carried within the secure authentication segment can be used to efficiently protect the confidentiality of the regular management data. However, as mentioned in the previous section, the X.509 one-way authentication scheme must face the challenge that the sender cannot determine whether or not the message has been delivered to the destination. Fortunately, our STCP can reduce the risk caused by this problem with the proposed synchronized-nonce scheme (see Section 5 for the discussion of message loss/delay). In the next section, we will propose the synchronized-nonce scheme.

4. The Synchronized-Nonce Schemes

The conventional nonce scheme uses the challenge-response protocol, which needs two message exchanges to verify the freshness of the received message. In the TMN environment (refer to Figure 3), if LS adopts the conventional nonce scheme to report an alarm to its manager MS, the following three steps will be performed: (1) LS must first issue a request to MS, (2) MS sends a "challenge" to LS, and (3) LS returns the "response" to the challenge with its alarm to MS. Clearly, for a common alarm report in TMN, three messages are needed to guarantee the freshness of the alarm report.

To reduce the cost of message exchanges, many authentication protocols, such as the X.509 one-way protocol, adopts the timestamp scheme. In the above alarm-reporting case, only one message with a timestamp attached is needed. However, the timestamp scheme requires an accurate clock-synchronization mechanism between LS and MS. To ensure correct and synchronous clocks among TMSs, a "secure" clock-synchronization protocol is needed, which is a contradiction. Therefore, we propose a new synchronized-nonce scheme in our protocol (STCP) to reduce the cost of message exchanges. And the attack-free environment for the clock-synchronization is based on the STCP.

(A) SN Scheme (I)

The "synchronized" nonce means that the value of the nonce can be predicted only by the two communicating parties involved, and cannot be predicted by others. In this way, the "challenge" message of the conventional nonce scheme can be skipped, and hence it is possible to guarantee the freshness of the message with a single message exchange. Before the authentication between the two communicating parties A and B , they must share a secret SN_i in advance. As A sends a message to B with the proposed STCP protocol, $h(SN_i)$ is inserted into the authentication segment of the message. Upon receiving the message, B uses his current nonce to calculate $h(SN_i)$ and compares it with the received one. If they are equal, B believes the received message is fresh. Afterwards A and B concurrently calculate the next nonce, $SN_{i+1} = SN_i + 1$, for the next message exchange. Because both A and B use the same series of SN 's, A can ensure that his previous message has been safely delivered while he receives a message, from B , with a correctly synchronized nonce. By using this way, STCP can reduce the risk of unknowing the loss of messages (The problem also annoys the X.509 one-way protocol and can be avoided only by adopting the two-way protocol which needs double messages).

```

SN_Check(R, SN, sentA B)
    /* R          h(SN) in the received message from B          */
    /* SN         the current nonce in A                          */
    /* sentA B the flag denoting whether A sent a message in the last message exchange with B */
begin
    if (R==h(SN)) || (sentA B==true && R==h(SN-1))
        sentA B := false;
        SN := SN + 1;
        accept the message;
    else
        Error_Handler();
end.

```

Fig. 7 The SN verification procedure invoked by A with the new SN scheme

To share the same initial nonce SN_0 , the parent TMS generates SN_0 and transmits this secret to his new child TMS with off-line methods, such as embedding the information within the setup disk.

Note that the nonce carried within the authentication segment must be protected by an irreversible one-way function. This is due to the fact that the data is visible in this segment which is encrypted with the secret key of the sender.

(B) SN Scheme (II)

Unfortunately, the SN scheme (I) will fail if *A* and *B* want to send messages to each other at the same time, called the concurrent message sending problem. Consider the following example with current nonce SN_5 . *A* sends the message with the nonce $h(SN_5)$ to *B* and then generates the next nonce SN_6 . Before the message arrives at *B*, *B* also sends a message with the nonce $h(SN_5)$ to *A* and generates the next nonce SN_6 . When *B* receives *A*'s message and checks its nonce, *B* will reject this message since the current nonce has been changed to SN_6 , instead of SN_5 .

To resolve the concurrent message sending problem, we modify the SN verification procedure of the previous scheme (shown in Figure 7) to detect the occurrences of concurrent message sendings in the STCP. In the new scheme, both participants of the communication channel share one synchronized nonce and respectively maintain a flag *sent*. After sending *B* a message, the sender *A* will generate the next new nonce and set his flag *sent*_{*A* *B*} to be *true*. If *A* receives a message from *B*, the new scheme will be invoked to verify the freshness of the message. First, *A* calculates $h(SN_i)$ with his local nonce SN_i and verifies whether the result is equal to the received $h(SN_{from\ B})$ carried within the received message. If it is a match, the message is fresh and accepted. *A* then sets his flag *sent*_{*A* *B*} to be *false* and generates the next nonce SN_{i+1} . Otherwise, there are two conditions that must be checked. They are:

- whether *A*'s flag *sent*_{*A* *B*} for the channel is *true*, and
- whether the received $SN_{from\ B}$ is equal to *A*'s preceding nonce (i.e. $h(SN_{from\ B}) = h(SN_{i-1}) = h(SN_i - 1)$).

If both conditions are satisfied, the concurrent message sending problem occurs. Therefore, *A* should still accept the message and update the current nonce. On the other hand, if the two conditions are not satisfied, *A* should invoke the error-handler function to process the event.

(C) SN Scheme (III)

Note that the SN scheme (II) does not work well in an environment where the frequency of sending out messages is large and the message delay is long. That is because the probability of continuous and concurrent message sendings will be large. Consider the following case. Before receiving *B*'s concurrent message with the nonce $h(SN_i)$, *A* has continually sent many messages and his current nonce has become SN_{i+k} , where $k \geq 2$. In this case, the valid message from *B* will be misjudged with the SN scheme (II) as a replay attack and thus will be rejected by *A*. The intuitive solution to the misjudged problem is to use two different nonces for both directions of the communication channel between *A* and *B*. For example, $SN(A,B)_i$ and $SN(B,A)_j$ are used in messages from *A* to *B* and from *B* to *A*, respectively.

This scheme is very simple and can entirely overcome the concurrent sending message problem. However, this solution aggravates the cost for maintaining synchronized nonces since the participants must always keep two nonces for each possible pair of communicating parties. Another disadvantage is that sender *A* can not ensure whether the message has been safely delivered to *B* (like the X.509 one-way protocol) because both nonces, $SN(A,B)_i$ and $SN(B,A)_j$, have no correlation.

In our system, both SN schemes (II) and (III) are implemented. Each communication channel depends on the frequency of transmitting messages and the requirement of the system performance to select one of them in conjunction with STCP.

(D) Error Handler

Although the SN schemes (II) and (III) can resolve the problems of concurrent sending or misjudging messages, other

problems may also cause the failure of SN verification. They include (1) hostile replay attacks, (2) the loss of previous messages, and (3) the failure of message ordering. The occurrence possibility of the last two problems is low because our TMN platform is built on TCP/IP, which supports ordered and reliable communications. Since it is very difficult to identify the real reason, the receiver must follow some error handling policy to reduce the effect of “asynchronized” nonces.

In our system, all exchanged messages can be divided into two types: query-and-response and event-reporting messages. For the query-and-response messages, the upper layer of TMS will follow a predefined process to prevent message loss, such as resubmitting the message while the time expires. The lost message thus can be detected and retransmitted by the sender when the misjudging problem occurs in STCP. For the event-reporting messages, it is usually acceptable to ignore message loss except for some urgent events. Because the event-reporting task contains only one message, if an urgent event-reporting message is misjudged and rejected by the receiver, it is difficult to detect the misjudgment and retransmit the lost message by the upper layer of sender's TMS, as the previous case. Thus, to handle the unpredictable and important event-reporting messages, we define a new message type *Em_Event* for the variable *MsgType* of the authentication segment. With the special message type, the receiver can still accept the urgent message while the misjudging problem occurs.

Thus, whenever the receiver judges a message as not fresh, the error-handler is invoked. First, the SN asynchronization problem is audited and a variable *SN_ERROR_NO* is added by 1. Second, if *SN_ERROR_NO* is larger than a predefined maximum value, a resynchronization process of SN will be invoked and a security alarm will be reported to the upper layer of the security management subsystem. Finally, if *MsgType* is *Em_Event* and the timestamp in the authentication segment of the message is close to the local clock within a limit, the message is accepted. Otherwise, the message is rejected.

(E) Resynchronizing SN

Although the whole TMN system, including SMF and CMIS, is implemented at the application layer of OSI seven-layer model and the possibility of message loss is very low, it is possible that the two communicating parties hold different values of the same SN (asynchronized) due to the physical errors of hosts/networks. The two copies of the same SN need be resynchronized under two conditions. First, the period predefined in STCP for the SN expires. Second, the number of rejected messages is beyond a predefined threshold (The rejection is due to the incorrect SN in the incoming message which may be caused by hostile replays, the misjudging problem, or physical network faults). As the SN resynchronization is performed, an SN_RESET message is sent. Its format is similar to the message of STCP presented in the previous section, except that $h(SN_i)$ is replaced by a predefined number *SN_RESET_NO*; *MsgType* is set to *SN_RESET*; the header of the data segment is $h(ID_A \ ID_B \ SN_0 \ h(New_SN))$; and the new initial value of SN is within *data*. The whole SN_RESET message is: $(Cert_A, \{T_A, SN_RESET_NO, ID_B, \{MsgType=SN_RESET, K\}_{K_{PB}}\}_{K_{SA}}, \{h(ID_A \ ID_B \ SN_0 \ h(New_SN)), New_SN\}_K)$. Since the old initial nonce *SN₀* is used within the header of the data segment, the intruders cannot replay this message to confuse the receiver to change his initial nonce again. Note that the new nonce *New_SN* is randomly selected by the sender of the resynchronization message. Since the initial value of *SN* is a random number and invisible by outsiders, a hostile third-party is unable to predict the following SNs. Due to the unknown *SN*, it will significantly reduce the risk of chosen-plaintext-attacks.

5. Security Analysis of STCP

In the following, we will discuss how STCP can satisfy the security requirements of the TMN system.

(A) Validity and confidentiality. The message of STCP consists of three components. they are the certificate segment, the authentication segment, and the data segment. The data segment is protected by a symmetric cryptosystem, and the message-oriented encryption key is protected by the receiver's public key and encapsulated in the authentication segment. The whole authentication segment is encrypted with the sender's secret-key of the public-key cryptosystem so that no one can forge a valid authentication segment (the purpose is similar to the digital signature). Although the computation complexity of the public-key cryptosystem is higher than the symmetric cryptosystem, the encryption/decryption cost is limited because the authentication segment is short.

(B) Chosen plain-text attack. According to the TMN standard, some management messages are transmitted with predefined contents in fixed periods. Thus, the encryption keys of these messages may be compromised with the chosen plain-text attack. If the session key scheme is adopted in this environment, intruders can compromise the key with the weak messages and use the key to decrypt other messages in the same session. In STCP, however, the data segment is encrypted by a different private key in each message exchange, and other messages are still secure even if a private key is compromised. Therefore, STCP can minimize the effect of the chosen plain-text attack.

(C) Message replay attack. As mentioned in the previous section, STCP used the synchronized nonce to guarantee the freshness of messages. Since the nonces are unpredictable for outsiders, a replayed message will be detected by the receiver.

(D) Data substitution attack. Because the three segments are simply concatenated together within a single message, it may suffer from the data substitution attack, that is, replacing one of the three segments. Fortunately, replacing the certificate segment gains nothing except confusing the communications, and the substitution of the authentication segment is difficult since the forgery or replay of the segment is inhibited. The only candidate of substitution attacks is the data segment. Thus, we design a verifiable header $h(ID_A \ ID_B \ SN_i \ h(data))$ within the segment. Because *SN_i* is unpredictable by others, the header can prove the relationship between the data and authentication segments.

(E) Message loss/delay. The message-loss/delay may be caused by physical faults or by an intruder's hostile control. Therefore, the main problem of using a single message for authentication in STCP is that the sender cannot immediately

determine whether the message has been delivered to the destination in time. Fortunately, with the *SN* sequence, message loss can be easily detected in the succeeding message, that is, an “asynchronized” *SN* will be present in the next received message. In addition, since the messages of STCP contain the senders' timestamp T_A , the receivers can detect whether the messages are delayed. Thus the effect of message loss/delay will be light and tolerable.

6. Conclusions

To manage the complicated telecommunication network, a TMN platform is developed to simplify the implementation of TMN-based applications. All management applications are connected with other TMSs or NEs through the platform and the data communication network. To ensure the security of management message exchanges, we propose a secure message exchange protocol (STCP), which is constructed under the TMN platform. The proposed protocol supports the secure communication of TMSs, so that all management applications built on the platform do not need to worry about the security problems caused by hostile outsiders. STCP uses the synchronized nonce scheme to guarantee the freshness of messages and thus needs only one message exchange for authentication. By this way, STCP minimizes the cost of authenticating the validity of received messages and distributing the encryption keys of data. Furthermore, for each TMS, he only needs to generate and maintain the public keys of his agent systems. Other public keys (his sibling TMSs) can be gotten through his parent TMS with STCP. Thus the proposed protocol is suitable for large network environments. Since STCP provides a stronger security for communications and more manageable for secret information than other schemes, such as SNMPv2 and X.509 one-way authentication protocol. Although the price of a stronger security is more computation overhead (due to the combination of a private-key and public-key cryptosystems), we design the data segment to reduce the computation cost. Thus, in addition to applying this protocol to telecommunications management networks, it is also applicable to the internet management systems which need high security and low cost for message exchanges.

Reference

- [1] Principles for a Telecommunications Management Network. CCITT Draft Recommendation M.3010.
- [2] C. T. Lin and C. W. Cheng, “*Platform and Tools for Developing an Integrated Network Management System (I)*,” Telecommunication Lab., Ministry of Communications, Taiwan, Technical Report TL-83-3301, 1995.
- [3] W. C. Liu, “*Platform and Tools for Developing an Integrated Network Management System (II)*,” Telecommunication Lab., Ministry of Communications, Taiwan, , Technical Report TL-84-3301, 1996.
- [4] C. T. Lin and B. F. Chen “*Platform and Tools for Developing an Integrated Network Management System (III)*,” Telecommunication Lab., Ministry of Communications, Taiwan, , Technical Report TL-85-3301, 1997.
- [5] TMN Management Functions. CCITT Draft Recommendation M.3400.
- [6] K. C. Lee, C. T. Lin, and S. P. Shieh, “*Platform and Tools for Developing an Integrated Network Management System (III) -- Security Management*,” Telecommunication Lab., Ministry of Communications, Taiwan, , Technical Report TL-85-3304, 1997.
- [7] R. Atkinson, “*IP Security Architecture*,” RFC-1825, Aug. 1995.
- [8] R. Atkinson, “*IP Authentication Header*,” RFC-1826, Aug. 1995.
- [9] R. Atkinson, “*IP Encapsulating Security Payload (ESP)*,” RFC-1827, Aug. 1995.
- [10] R. M. Hinden, “*IP Next Generation Overview*,” Communications of the ACM, Vol. 39, No. 6, 1996.
- [11] A. Aziz and M. Patterson, “*Simple Key Management for Internet Protocols (SKIP)*,” Proceedings of the INET'95 Conference, Jun. 1995.
- [12] J. Galvin and K. McCloghrie, “Security Protocols for version 2 of the Simple Network Management Protocol (SNMPv2),” RFC-1446, Apr. 1993.
- [13] J. Case, M. Fedor, M. Schoffstall, and J. Davin, “*A Simple Network Management Protocol (SNMP)*,” RFC-1157, May 1990.
- [14] J. G. Steiner, B. C. Neuman, J. I. Schiller, “*Kerberos: An Authentication Service for Open Network Systems*,” Proceedings of the Winter 1988 Usenix Conference, Feb. 1988.
- [15] S. P. Shieh and W. H. Yang, “*An Authentication and Key Distribution System for Open Network Systems*,” ACM Operating Systems Review, Vol. 30, No.2, 1996.
- [16] S.P. Shieh, W.H. Yang, and H.M. Sun, “*An Authentication Protocol without Trusted Third Party*,” IEEE Communication Letters, May 1997.
- [17] D. L. Mills, “*Network Time Protocol (Version 3) Specification, Implementation and Analysis*,” RFC-1305, Mar. 1992.
- [18] “*Recommendation X.509 and ISO 9594-8, Information Processing Systems - Open Systems Interconnection - The Directory - Authentication Framework*,” CCITT Technical report, Mar. 1988.
- [19] L. Gong, “*Security Risk of Depending on Synchronized Clocks*,” ACM Operating System Review, Vol. 26, No. 1, 1992.
- [20] S. Chokhani, “*Toward a National Public Key Infrastructure*,” IEEE Communications Magazine, Sep. 1994.
- [21] ANSI X3.92, “*American National Standard for Data Encryption Algorithm*,” American National Standards Institute, 1981.
- [22] A. Curiger, and B. Stuber, “*Specification for the IDEA Chip*,” Technical Report No. 92/03, ETH Zurich: Institute for Integrate System, Feb. 1992.