Int. J. of Computers, Communications & Control, ISSN 1841-9836, E-ISSN 1841-9844
Vol. III (2008), No. 4, pp. 343-352

Analysis and Design on Key Updating Policies for Satellite Networks

Yuxuan Ji, Hengtai Ma, Gang Zheng

Abstract: Satellite networks are becoming increasingly important because of the exciting global communication services they provide. Key management policies have been successfully deployed in terrestrial networks to guarantee the information security. However, long propagation, storage and computation constraints bring new challenges in designing efficient and cost-effective key updating policies for satellite networks. Based on the structure and communication features of satellite networks, a dynamic key management model for satellite networks (DKM-SN) is presented, which includes certificates owned by each satellite, primary keys and session keys both of which are shared between two satellites. Furthermore, a protocol is designed for updating certificates for satellites; different policies for updating primary and session keys are studied and their efficiency and security are analyzed and compared. In addition, simulation environment for satellite networks is built and the key updating processes are implemented in Walker constellation. From the simulation results, further contrasts on key updating time and storage costs between the applications of IBM hybrid key management model (HKMM) and DKM-SN in satellite networks are presented. Finally, important suggestions in designing key updating policies are given.
Keywords: key updating, satellite networks, model, protocol, simulation

1 Introduction

Satellite networks are composed of various kinds of communication satellites, vehicles and constellations. It contains both satellite-to-satellite and satellite-to-ground links. Satellite networks integrate terrestrial systems and all sorts of satellites which are deployed in different orbits with diverse tasks. Nowadays, satellite networks are increasingly used in the long-distance information transmission services. In order to ensure the message confidentiality, integrity and nonrepudiation, as well as efficiency of communication, a key management mechanism should be used to provide data encryption, authentication and key distribution and updating services for satellite communication. Key management model defines the entities in the services, the categories and relationships of the keys, and the key updating protocols and algorithms. In contrast with terrestrial networks, satellite networks are subject to dynamic network topology, long propagation delay, as well as low computing and storage capabilities of satellites. Due to these constraints, efforts should be made to decrease the key updating time in order to reduce the communication cost. Besides, for those keys used to encrypt large amount of information, the key updating protocols should be based on symmetric encryption techniques for the sake of computation cost. The storage cost on satellites should also be reduced by efficiently lowering the number of the keys.

Currently, key management policies for terrestrial networks are comparatively sophisticated. There are generally three kinds of key management policies, including those that use symmetric key encryption techniques, public key encryption techniques and combination of the two. For example, Kerberos [1] uses symmetric key encryption techniques and a KDC (Key Distribution Center); both symmetric and public key encryption techniques are adopted in IBM hybrid key management model (HKMM) in which three kinds of keys including session key, primary key and public key are used. Based on different network features and environments, the number of entities and encryption techniques involved in updating a certain type of key may be very different. For instance, though both for updating the session key, Neuman-Stubblebine protocol [3] includes three parties while Janson-Tsudik protocol [4] involves only

two. Besides, the encryption technique used to update session key can be either symmetric or public. However, all above key management policies are only applicable in terrestrial networks.

Some problems concerning key management policies in satellite networks have been studied. After studying a series of possible security threats, CCSDS showed the urgency and necessity to implement security policies in satellite networks, such as key management, authentication, access control, etc [5]. Ayan Roy-Chowdhury analyzed some problems that occur during encryption and key distribution processes when applying IPSec and SSL into satellite networks [6]. Their discussion was based on a hybrid satellite network with a single satellite component and several ground terminals. Cruickshank designed an authentication and key establishment protocol for two satellite users who need to encrypt data and voice information [7]. Since the public encryption technique was used, the method was applicable in situations where the satellites are only used to relay messages rather than implement public encrypting operations which entail a large amount of computation cost. Tanya Vladimirova et al. introduced some security services required on satellites, proposed an on-board security architecture and AES fault-tolerant mechanism [8]. However, current literature seldom studies the key management policy for satellite networks with communication links between satellites. We focus on this issue and mainly discuss the categories of keys, the protocols for updating keys and the key updating efficiencies of different policies.

HKMM uses session key, primary key and public key. However, the session key updating in HKMM may cause unendurably long propagation if directly applied in satellite networks. In order to solve this problem, a dynamic key management model applicable in satellite networks (DKM-SN)is presented, which also includes session keys, primary keys and public keys. Based on DKM-SN, a protocol is designed for updating public keys in satellite networks. By further studying the protocols such as Neuman-Stubblebine, Janson-Tsudik and improved Beller-Yacobi protocols [9], the efficiency and security issues are analyzed in designing policies for updating primary keys and session keys in satellite networks. Finally, the differences on time and storage costs between HKMM and DKM-SN are shown through simulations under the satellite network environment. The rest of this paper is organized as follows. In section 2, we discuss DKM-SN, detailing the designing principles of all three parts including public key, primary key and session key updates. Section 3 presents the simulation results under satellite network environment. We conclude with a short summary and extensions on future work in Section 4.

2 Key Management Model

In satellite networks, communication process should be kept secret in order to ensure the message confidentiality. Besides, robust and secure protocols are indispensable so that the communication process can resist well-known attacks, such as arrogating, playback, modification and so on. Therefore, we design a DKM-SN to ensure the secrecy, authenticity and integrity of messages, and at the same time decrease the time, storage and computation costs with best efforts when providing key updating services. DKM-SN consists of public, primary and session keys, which are divided based on their functions. Firstly, every satellite has a pair of public and private keys, and a certificate issued by CA (Certificate Authority), all of which are updated through communication between a certain satellite and CA. Secondly, there is a primary key shared between a pair of satellites and it is updated with their public key information (including public and private keys and certificates). Besides, two satellites also need to share a session key when they communicate with each other and the session key is updated by the primary key shared between them. The differences between DKM-SN and HKMM are: (1) HKMM uses KDC while DKM-SN does not; (2) each primary key in HKMM is shared between satellite and KDC while it is shared between two satellites in DKM-SN; (3) in order to lower storage cost, dynamic primary key updating policy is adopted in DKM-SN, which means that for some pairs of satellites their primary keys exist only when they need to establish session keys to communicate.

Figure 1 shows HKMM on left and DKM-SN on right. P denotes primary key, S for session key, T for terminal in terrestrial networks, and V for satellite. We can see that in HKMM the primary keys

are shared between each terminal and KDC while the session keys are shared between every pair of terminals. For instance, in HKMM, session key S1 can be established by terminal 3, terminal 4 and KDC using primary keys P3 and P4. In DKM-SN, both the primary and session keys are shared between two satellites. For instance, session key S21 needs to be established and shared only by two satellites V2 and V3 using their primary key P2.

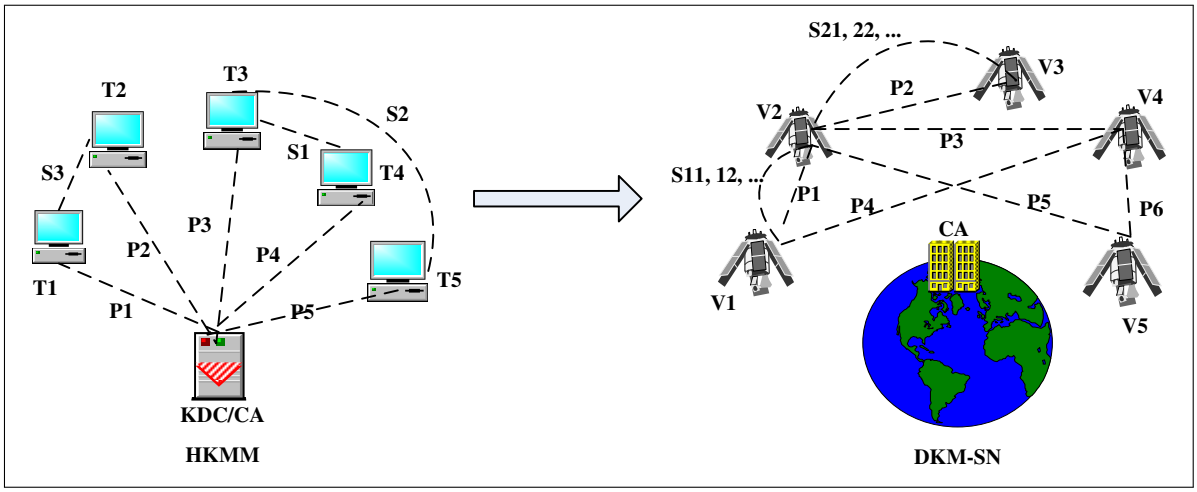

Figure 1: Key management models: HKMM and DKM-SN

2.1 Certificate Updating

Generally, an entity's public key information is updated by itself before it expires. Based on PKIX standards [10], we design a proper protocol for updating public key information.

1. $B \rightarrow M : B, PK_B, Cert(B)$
2. $M \rightarrow B : M, PK_M, [Cert(M)], PK'_M, E(PK_M^{-1}, h(PK'_M, M, B))$
3. $B \rightarrow M : Cert(M)'$

B, M - CA and satellite M; PK_X, PK_X^{-1}, $Cert(X)$ - old public key, private key and certificate of satellite X; PK'_X, $(PK'_X)^{-1}$, $Cert(X)'$ - new public key, private key and certificate of satellite X; $E(K,Y)$ - public key encryption with key K and plaintext Y; $h(Y)$ - hush function; $[Y]$ - Y is optional.

The execution of this protocol between CA and satellite proceeds as follows:

1. CA sends its own identifier, public key and certificate to satellite and informs the satellite to start updating its public key information.
2. Upon receiving message from CA, the satellite does the followings: (1) check validity of CA's certificate; (2) generate a new pair of public and private keys for itself; (3) get a signature with its old private key and send this signature, its identifier, certificate and both old and new public keys to CA.
3. Upon receiving messages from satellite, CA does the followings: (1) check the validity of satellite's old certificate; (2) check satellite's signature; (3) generate a new certificate for the satellite's new public key.
4. Upon receiving new certificate from CA, satellite can check its validity with CA's public key.

Note that the execution can continue only when the checks in the last step are successful. Now we consider some possible attacks on CA and satellite when running the protocol.

1. Attacks on CA

   (1) In the second step of our protocol, it is possible that adversary may arrogate satellite M by sending M's certificate and a public key the adversary generated. However, the adversary has to send a signature at the same time. Since we assume that the adversary doesn't know the satellite's old private key before it expires, then the adversary could not get a correct signature and would be detected by CA.

   (2) Also in the second step, the adversary may try to replay a signature signed by the satellite's private key which has expired. Obviously, only if both satellite and CA use their valid public key information that hasn't expired, our protocol could resist this playback attack.

2. Attacks on satellite

   (1) Since satellite M's new public key is sent without encryption, the adversary may generate a certificate for M by signing M's identifier and the new public key with its own private key, and then sends this false certificate to M in the third step. However, this attack would fail after satellite has received the certificate and checks its validity using CA's public key.

   (2) As an alternative, in the third step the adversary could also replay a certificate that has expired. However, the fact is that in order to keep the freshness of every run of the protocol we require that new public and private keys are different from before. For this reason, the expired certificate would be proved invalid when the satellite checks it.

The above discussion is suitable for a satellite. When updating the public key information for CA, we should first update it before it expires and then update certificates for all the satellites using CA's updated certificate.

2.2 Primary Key Updating

Although all satellites have their own public key information, we avoid using it to encrypt data due to the high complexity and low efficiency of public key encryption algorithms. In most systems, public key information is used to establish symmetric keys so that encryption on data could be faster. We consider two strategies for establishing symmetric keys with public key information: (1) using public key information, two satellites directly establish a shared session key to encrypt data; (2) they first establish a primary key between them with their public key information and then use this primary key to establish their session keys. Both primary key and session keys are symmetric keys shared between the two satellites. In the first strategy, though the cost of data encryption is low, the cost of frequent session key updates remains high. As for the second strategy, since both of frequent data encryption and session key updates adopt symmetric key encryption techniques, it is more efficient than the first one. Note that the primary keys are updated much less frequently than session keys.

HKMM includes primary keys and a KDC in its system. There is a unique primary key shared between KDC and each terminal (or entity). Hence, the number of primary keys in HKMM is n.

In DKM-SN, the second strategy is adopted, but no KDC is deployed in our network and each unique primary key is shared between two satellites. We build DKM-SN in this way for the following reasons: (1) KDC may become a bottleneck since all the session key updates have to pass KDC; (2) satellite networks are typical of long propagation. Since the session key updating process involving three parties (KDC and two satellites) has to be finished within at least 4 steps while that involving two parties (two satellites) in DKM-SN can be achieved within 3 steps, the former would be much slower than the latter. In

satellite networks, even one trip may lead to unendurably long propagation due to the multi-hop routing and long space distance between two satellites.

As for the communication protocol when updating primary keys, improved Beller-Yacobi protocol [9] is suggested. This sophisticated protocol is qualified for updating a symmetric key with public key information and engages only two entities. Besides, this protocol is suitable for the satellites that have limited computing power.

2.3 Session Key Updating

Session keys are distributed in the network to encrypt the large amount of data, such as images, languages, commands and so on. According to the previous analysis, session key in DKM-SN is shared between two satellites and updated by the primary key between the same pair of satellites. Session keys must be updated more frequently than certificate and primary keys because they are used more often. In this way, the possibility of ciphertext-only attacks can be decreased. Obtaining an old session key is not once and for all, since the attacker must intercept and analyze new ciphertexts in order to get new session keys.

As for the protocols for updating session keys, security and efficiency are of most importance. First of all, the protocol should be able to resist well-known attacks, such as replay, modification, typing, reflection and so on. Besides, it should not cost too much time and storage. Based on satellite network features, such as long propagation and limited resources, Janson-Tsudik 2PKDP [4] is suggested for updating session keys. This protocol is illustrated as follows.

1. $A \rightarrow B : A, N_{ab}$
2. $B \rightarrow A : AUTH_{K_{ab}}(N_{ab}, K_{ba}, B), E_{K_{ab}}(N'_{ba}) \oplus K_{ba}$
3. $A \rightarrow B : ACK_{K_{ab}}(N_{ab}, K_{ba}, A)$

This key establishment protocol contains the minimum cost of computation and numbers of messages and steps as proved in [4].

A detailed comparison of the time and computation costs between session key updating protocol with KDC and that without KDC is presented as follows. As for session key updates with KDC, we use Neuman-Stubblebine protocol [3] which also has the minimum number of steps involved.

According to Janson-Tsudik 2PKDP, we get the calculation time C_{jt} in a single entity and overall session key establishment time T_{jt}:

$$\begin{aligned} T_{jt} &= 3 * T_{ab} + 4 * T_{mac} + 2 * T_{es} + 2 * T_{\oplus} \\ C_{jt} &= 2 * T_{mac} + T_{es} + T_{\oplus} \end{aligned} \tag{1}$$

T_{mac} - calculation time of $MAC()$ function; T_{es} - calculation time of symmetric encryption; $T_{\oplus}$ - calculation time of XOR operation; T_{ab} - propagation delay between satellites A and B.

Since $T_{\oplus} \leq$ any other item, so we get:

$$\begin{aligned} T_{jt} &\approx 3 * T_{ab} + 4 * T_{mac} + 2 * T_{es} \\ C_{jt} &\approx 2 * T_{mac} + T_{es} \end{aligned} \tag{2}$$

As for Neuman-Stubblebine protocol, we get the calculation time C_{ns} for a single node and overall session key establishment time T_{ns} similarly:

$$\begin{aligned} T_{ns} &= (2 * T_{ab} + T_{ac} + T_{bc}) + 8 * T_{es} \\ C_{ns} &= 8 * T_{es} \end{aligned} \tag{3}$$

T_{ac} - propagation delay between KDC and satellite A; T_{bc} - propagation delay between KDC and satellite B.

Because propagation delay is much larger than the calculation time of either encryption or $MAC()$ in satellite networks, we conclude from the above estimation that both the calculation time cost of a single node and overall session key updating time cost of Janson-Tsudik 2PKDP are much less than those of Neuman-Stubblebine.

For the three-party session key establishment policy, KDC may become a bottleneck in the system. Fortunately, the two-party session key establishment policy could avoid this drawback.

3 Simulations

In order to verify the effectiveness and suitability of DKM-SN in satellite network environment, a proper simulation environment for satellite networks is very important and necessary. Based on the simulation system for distributed satellite networks [11], we design a simulation scenario and implement the experiments. We set up a Walker constellation consisting of 20 MEO satellites. There are 4 orbital planes on each of which 5 satellites are equally deployed. Each plane is of an inclination of 75° and orbit height 14163km.

In certificate updating simulation, the protocol designed in section 2.1 is applied and 160-bit ECDSA [12] is implemented as signature algorithm. In primary key updating, improved Beller-Yacobi protocol is applied. ECDSA and ECIES [13] are used as signature and encryption algorithms respectively. In session key updating, Janson-Tsudik 2PKDP and 128-bit AES are implemented.

3.1 Certificate Updating Simulation

Dynamic Topology

In this part, we examine the influences of topological changes on the certificate updates for a certain satellite.

Dynamic changes of satellite network topology lead to the variations of routing tables in satellites. Therefore, the time cost of certificate updating for a certain satellite varies with time. Figure 2 shows both the overall and computation time costs of certificate updates for a certain satellite in different periods of time.

We update a satellite's certificate every 10 seconds. Figure 2 shows the time cost during 500 seconds. The longest updating time for the satellite is 834.200ms. Routing information shows that in the longest updating data transmission from the satellite to CA entails 5 hops and thus the three-step interaction between them entails 15 hops all together. The minimum time cost is 167.45ms when there is only 1 hop between satellite and CA and so it costs 3-hop effort to finish the certificate updating process. Based on the result, we suggest update certificates when communication between satellite and CA needs the minimum number of hops.

Static Topology

In this part, we examine the certificate updates of different satellites in Walker constellation during a fixed period of time.

Figure 3 shows both the overall and computation time costs of certificate updates for different satellites during a fixed period of time in Walker constellation. We can see that there are huge gaps of time costs among the satellites since the number of hops between CA and the satellites are very different. In the simulation, for example, there are 5 hops between CA and NO.6 satellite while 1 hop between CA and NO.4.

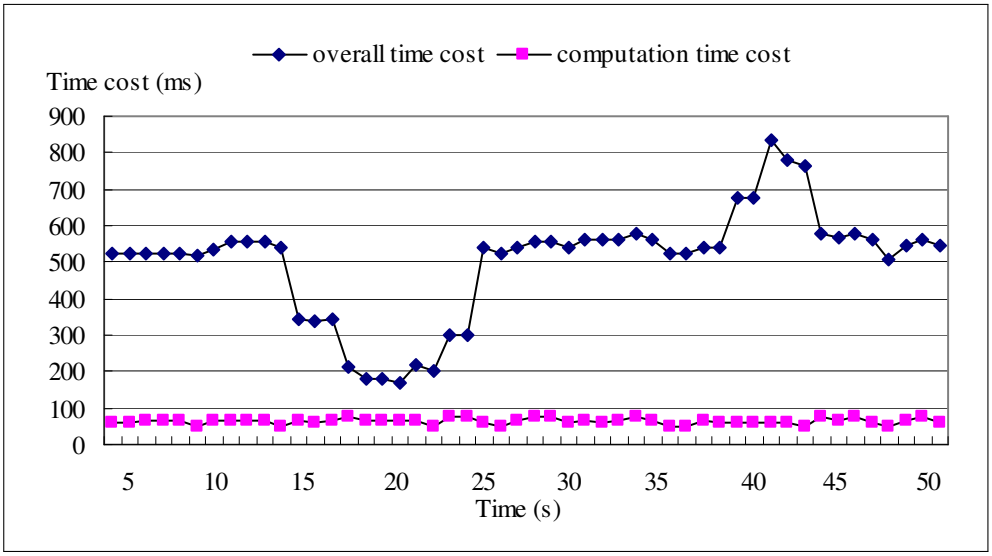

Figure 2: Time cost of certificate updating of a certain satellite in different periods of time

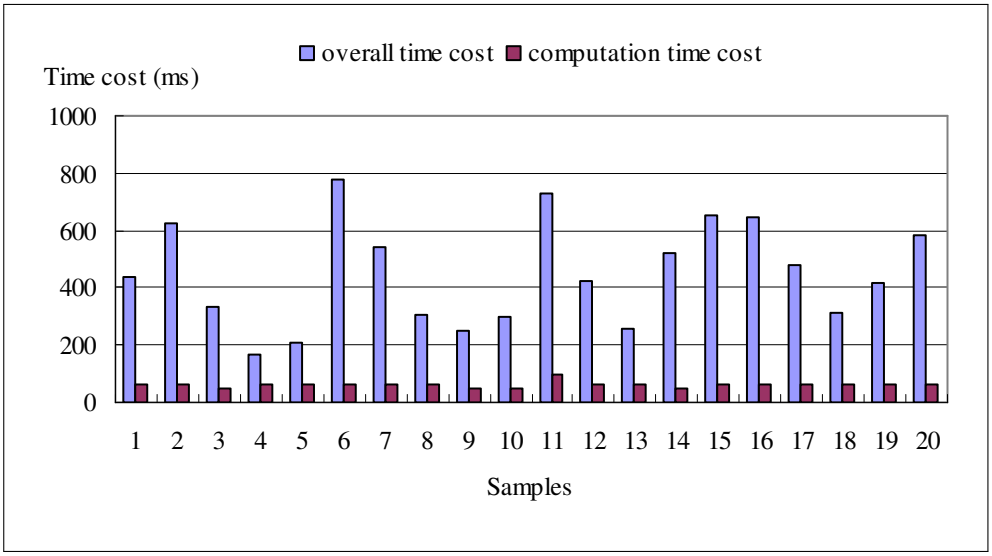

Figure 3: Time costs of certificate updates of all satellites in the same period of time

Besides, from Figure 2 and Figure 3 we can see that the computation cost for different satellites varies little. It is a small fraction of the overall time cost of certificate updates, ranging from 5% to 40%.

3.2 Comparison of Computation Time Costs

Figure 4 shows the computation costs of all three kinds of key updates in Walker Constellation. 20 samples are presented for each kind of updates.

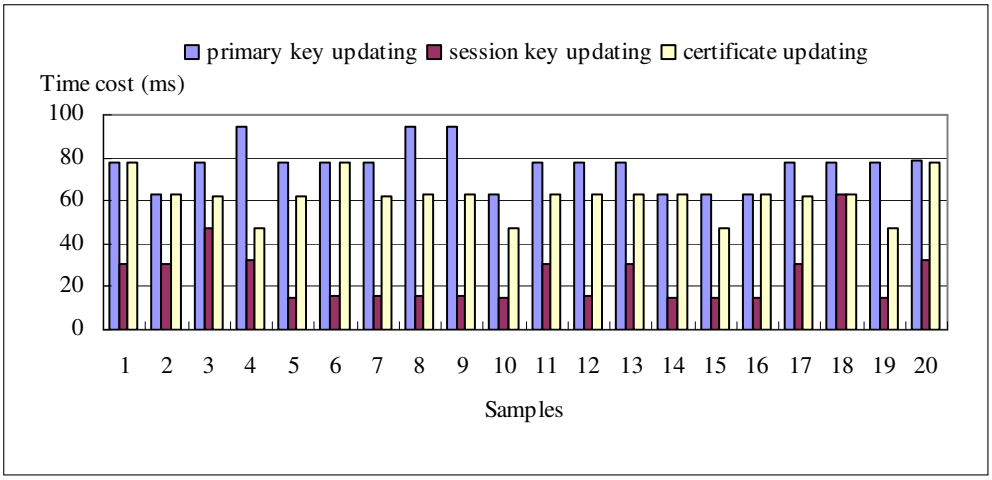

Figure 4: Computation costs of three kinds of key updates

We see from Figure 4 that session key update costs least, which is suitable for the most frequently updated session keys. The average time cost of certificate updates is less than that of primary key updates mainly due to the extra public key encryption in the improved Beller-Yacobi protocol.

3.3 Comparison between Key Management Models

Comparing DKM-SN with HKMM, we mainly have the following conclusions. Firstly, the time costs for session key updates in DKM-SN are much less than that in HKMM. Secondly, DKM-SN will contain more primary keys than HKMM if all primary keys are pre-distributed between all pairs of satellites. Since the storage capacity is limited in satellite, we recommend that for some pairs of satellites their primary keys exist only when necessary. Therefore, if we want to establish a session key between two satellites while they share no primary key, we need to first establish a primary key with their public key information and then establish the session key with their primary key. A much detailed comparison between HKMM and DKM-SN is presented as follows.

Suppose the total number of satellites is n; the average time cost of primary key updates is z.

In DKM-SN, let a denote the maximum number of primary keys, namely $a = \frac{n\times(n-1)}{2}$; h denotes the average time cost of session key updates when there is a shared primary key between every pair of satellites; x denotes the practical number of primary keys; y denotes the average time cost of session key updates when x pairs of satellites share primary keys.

In HKMM, suppose the average time cost of session key updates is $h+\delta$, $\delta > 0$.

Based on DKM-SN, we have,

$$y = \frac{x\times h+(a-x)\times(z+h)}{a} = (z+h) - \frac{z}{a}\times x \tag{4}$$

Obviously, given $x = 0$, we get $y = z+h$. This means there is no primary key pre-shared in DKM-SN and for every two satellites before establishing a session key we should first set up a primary key. Similarly, given $x = a$, we get $y = h$. Under this condition, each pair of satellites owns a primary key.

If $y > h + \delta$, we get $x < \frac{a \times (z-\delta)}{z}$.

So we conclude that when $n < x < \frac{a \times (z-\delta)}{z}$, HKMM is superior to DKM-SN in both average time cost of session key updates and storage cost.

Specifically, based on the simulation data, we have $z \approx 400ms$, $h \approx 360ms$, $\delta \approx 120ms$, $n = 20$.

Under this condition, $y = (z+h) - \frac{z}{a} \times x \approx 760 - 2 \times x$. Figure 5 shows the relationship between average time cost of session key updates and the number of primary keys in DKM-SN.

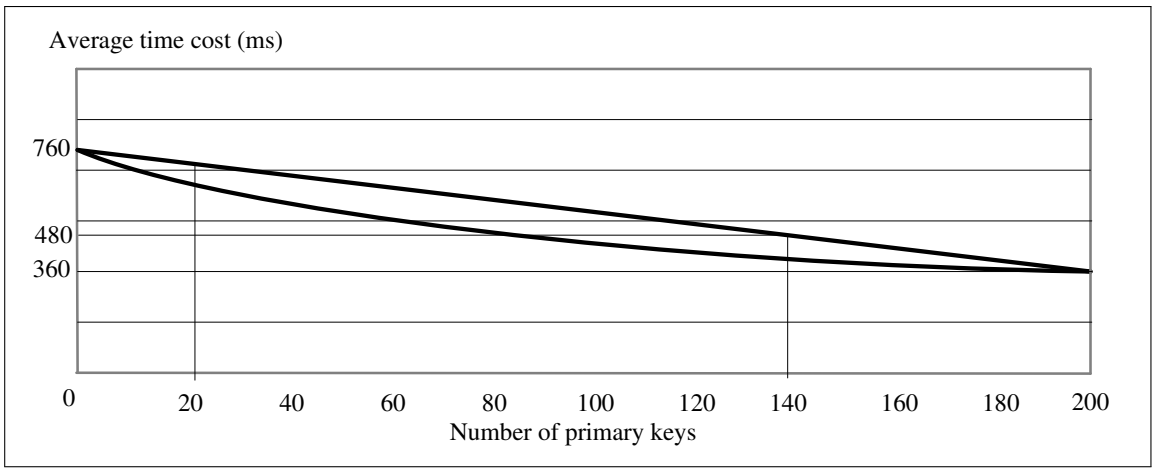

Figure 5: Average time cost of session key updates in DKM-SN

Given $y > h + \delta$, namely $760 - 2 \times x > 480$, we get $x < 140$. We can see from the linear function shown in Figure 5 that if the number of primary keys ranges from 20 to 140, HKMM is better than DKM-SN in both time and storage costs. When $140 < x \leq 200$, average time cost of session key updates in DKM-SN is less than that in HKMM while the number of keys is more than that in HKMM. Furthermore, if we can deploy the primary keys for those satellites that communicate most frequently, the average time cost function of the number of primary keys will be a concave function shown in Figure 5. We can make estimation and decision given real on-board data in satellite networks. For examples, if satellites could store the maximum number of primary keys, pre-allocating a primary key for every pair of satellites is the best choice, which is especially suitable for a network with a small number of satellites.

4 Summary and Conclusions

In this paper, a new key management model called DKM-SN is designed for satellite networks. A protocol for updating satellite certificate is designed and the efficiency and security of different policies for updating primary and session keys are analyzed. The performance is shown when updating three kinds of keys and a further contrast between DKM-SN and HKMM is given in both time cost and storage cost. The efficiency and applicability of different key updating policies are discussed under different on-board storage constraints and time requirements. In the future, we will further discuss the problems in designing efficient, low-cost and secure key management models for satellite networks.

Bibliography

[1] J. Kohl, C. Neuman, The Kerberos Network Authentication Service (V5), *http://www.ietf.org/rfc/rfc1510.txt*, RFC 1510, 1993.

[2] V. Le, S. M. Matyas, D. B. Johnson and J. D. Wilkins, A Public Key Extension to the Common Cryptographic Architecture, *IBM System Journal*, Vol. 32, pp. 461-485, 1993.

[3] B. C. Neuman and S. G. Stubblebine, A Note on the Use of Timestamps as Nonces, *ACM Operating Systems Reviews*, Vol. 27, pp. 10-14, 1993.

[4] Philippe Janson and Gene Tsudik, Secure and Minimal Protocols for Authenticated Key Distribution, *Computer Communications*, Vol. 18, pp. 645-653, 1995.

[5] CCSDS, Security Threats Against Space Missions, *Washington: Informational Report*, CCSDS 350.1-G-1, Green Book, Issue 1, 2006.

[6] A. Roy-Chowdhury *et al.*, Security Issues in Hybrid Networks with a Satellite Component, *IEEE Wireless Communications*, Vol. 12, pp. 50-61, 2005.

[7] H S Cruickshank, A Security System for Satellite Networks, *Fifth International Conference on Satellite Systems for Mobile Communications and Navigation*, London: IEE, pp. 187-190, 1996.

[8] Tanya Vladimirova, Roohi Banu and Martin N. Sweeting, On-Board Security Services in Small Satellites, *MAPLD International Conference*, Washington: NASA Office of Logic Design, 2005.

[9] C. Boyd and A. Mathuria, Key Establishment Protocols for Secure Mobile Communication: a Selective Survey, *Lecture Notes in Computer Science*, Vol. 1438, pp. 344-355, 1998.

[10] J. Schaad, M. Myers, Public-Key Infrastructure (X.509), *www.ietf.org/html.charters/pkix-charter.html*, IETF, PKIX 2797.

[11] X. Ying, Z. Gang, Modeling and Distributed Simulation for Satellite Networks, *Computer Simulation*, Vol. 25, pp. 65-69, 2008.

[12] ANSI X9.62, Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA), American National Standards Institute, 1999.

[13] ANSI. X9.63, Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography, American National Standards Institute, 2001.

Yuxuan Ji
Institute of Software, Chinese Academy of Sciences
National Key Laboratory of Integrated Information System Technology
4# South Fourth Street, Zhong Guan Cun, Beijing 100190, P.R. CHINA
E-mail: jiyuxuan06@gmail.com

Hengtai Ma
Institute of Software, Chinese Academy of Sciences
National Key Laboratory of Integrated Information System Technology
4# South Fourth Street, Zhong Guan Cun, Beijing 100190, P.R. CHINA
E-mail: htma@ios.cn

Gang Zheng
Institute of Software, Chinese Academy of Sciences
National Key Laboratory of Integrated Information System Technology
4# South Fourth Street, Zhong Guan Cun, Beijing 100190, P.R. CHINA
E-mail: gangzhengcn@yahoo.com.cn