Timestamped Key Management for Mission-Critical WAHNETs

Sheeja. A

Anna University, Tirunelveli,
Nagarcoil.K.K Dist, Tamilnadu,India

sheeja.cse@gmail.com

***Abstract*—Cryptographic key management is challenging due to the dynamicity and the unreliable communications. Moreover, in wireless ad-hoc the number of ad-hoc wireless devices deployed at an incident scene depends on the specific nature of the incident. It is computationally infeasible to design a key management scheme in current mission-critical networks to fulfil the required attributes to satisfy the needed attributes like data integrity, authentication, confidentiality, no repudiation and service availability. In a scalable method of cryptographic key management (SMOCK) communication overhead is less, and it offers high service availability. Once the wireless devices are dispatched from the incident area, the centralized trusted server loses control of these devices. SMOCK will operate securely when small numbers of nodes are involved. To operate more number of nodes we introduced timestamp scheme in this paper. It enables the centralized server to have a control of all the current nodes.**

Keywords*— *certificate graph, predistributed key, combinatorial key management, keypool, timestamp

I. INTRODUCTION

Rapid developments have been achieved in current mission critical systems and they composed of mobile and wireless devices. First responder systems (emergency rescue and disaster recovery), military applications, and Health-care are the examples. Wireless mobile ad hoc network is a relative newly developed network architecture for wireless mobile terminals. Instead of making use of a centralized routing service, mobile devices cooperate with each other to route network packets from source to destination in a multi-hop manner. There is no infrastructure in mobile ad hoc network, each user only knows his neighbors one hop away.

Security is one of the major issue in transferring information and storage of data. Any small hole in the encryption/decryption mechanism may lead to the benefit of the intruders and the hackers. Several algorithms have been evolving for the encryption of data with some keys. Both symmetric and asymmetric encryption algorithms use keys in transaction. Symmetric algorithms use a single key at both sides, sender and the receiver side, whereas, asymmetric algorithms use different keys. Asymmetric algorithms [2] generate two keys, one is kept private and another one is public.

Key management is the provisions made in a cryptography system design that are related to generation, exchange, storage, safeguarding, and replacement of keys. It includes cryptographic protocol design, key servers, user procedures, and other relevant protocols. Key management concerns keys at the user level, either between users or systems. Successful key management is critical to the security of a cryptosystem. In practice it is arguably the most difficult aspect of cryptography because it involves system policy, user training, organizational and departmental interactions, and coordination between all of these elements.

Cryptographic Key Management (CKM) is the complete set of operations necessary to nurture and sustain encrypted data and its associated keys during the key and data lifecycles. It includes Key Store, Key Serving, and Key Administration. There are several fundamental differences between mobile ad hoc network and traditional wired network:

A. Dynamic topology

Network terminals can move freely at certain speed, therefore network topology is always changing and hardly to be predicted;

B. Resource constraints

Mobile terminals can be notebooks, PDAs or mobile phones. They all have limited computation power and short battery life;

C. No infrastructure

Ad hoc network is meant to be deployed swiftly. Mobile users cooperate to route packets, and thus there is no need of centralized services; and

D. Limited physical security

Mobile devices such as notebooks, PDAs and mobile phones don't have strong secure systems due to cost and limited power.

A realistic assumption about mission-critical applications is that before mobile devices are dispatched to an incident area, they are able to communicate securely with the trusted authentication server in their

domain center, and get prepared before their deployment. Once the wireless devices are dispatched into the incident area, the centralized trusted server loses control of these devices and the mobile devices cannot trust anybody if local information cannot authenticate it.

II. Literature Survey

A. *Certificate issuing:*

The protocol begins with issuing of certificates. During this phrase, users issue certificates for their trusted neighbors. The issuing of certificates is bidirectional,[3] which means if sender issues a certificate for receiver, receiver will also issue a certificate for sender, such that they two becomes friends.

B. *Certificate exchanging:*

These certificates can be exchanged among their friends. This is done by exchanging of certificate chain packets with friends.

C. *Maximum clique searching on certificate graph:*

The purpose of issuing and exchanging certificates is to build a certificate graph[4]. With this graph, user can get a rough knowledge of his neighbours. By running maximum clique searching on this certificate graph, user can find a subset of nodes, which are the maximum clique members. These maximum clique members are announced as Certificate Authorities.

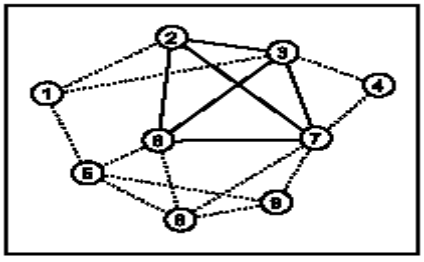

Fig-1

D. *Existing System:*

- In secure communication, wireless sensor networks use symmetric key techniques. In symmetric key techniques, secret keys are predistributed among nodes before their deployment.
- A challenge of the key distribution scheme is to use small memory size to establish secure communication among a large number of nodes and achieve good communication.
- Public-key (corticated)-based approaches were originally proposed to provide solutions to secure communications for the Internet, where security services rely on a centralized certification server.
- The certificate-based approaches to ad-hoc networks and present a distributed public-key-management scheme for ad-hoc networks, where multiple distributed certificate authorities are used.
- To sign a certificate, each authority generates a partial signature for the certificate submits the partial signature to a coordinator that calculates the signature from the partial signatures

E. *Proposed System:*

- In this system we propose to support secure communications with the attributes of data integrity, authentication, confidentiality, nonrepudiation, and service availability.
- To build a secure communication system, usually the first attempt is to employ cryptographic keys. In SMOCK, let us assume a group of people in an incident area, who want to exchange correspondence securely among each other in a pair-wise manner.
- The keypool of such a group consists of a set of private–public key pairs, and is maintained by an offline trusted server.
- Each key pair consists of two mathematically related keys. The *i*th key pair in the key pool is represented by (K^i_{priv}, K^i_{pub}) . To support secure communication in the group, each member is loaded with all public keys of the group and assigned a distinct subset of private keys.
- Each person keeps a predetermined subset of private keys, and no one else has all of the private keys in that subset. For a public–private key pair, multiple copies of the private key can be held by different users.
- A message is encrypted by multiple public keys, and it can only be read by a user who has the corresponding private keys.
- To operate more number of nodes we introduced timestamp scheme is to be introduced. It enables the centralized server to have a control of all the current nodes.

III. Timestamps

To operate more number of nodes in an ad-hoc environment timestamp scheme is introduced in this paper. It enables the centralized server to have a control of all the current nodes. For extending the validity of a cryptographic timestamp for a Time Stamping Service contains security shortcomings. A time-stamping service (TSS) has been identified as a potentially important part of a Public Key Infrastructure (PKI), and draft standards have been produced by both bodies. Conventionally a TSS will take as input a hash-code of a data string supplied by a client, and will return a digital signature

computed on a concatenation of this hash-code and a time-stamp (the cryptographic timestamp). This cryptographic timestamp can then be used as evidence that the original data string existed at the time indicated, without revealing the data string to the TSS. The hash-code should be computed using a collision-resistant one-way hash-function. One particularly important application of a TSS is to prolong the lifetime of a digital signature. Without use of a TSS, when a public key certificate expires or is revoked, all signatures computed using the corresponding private key potentially lose their validity. This is because, if the private key becomes known, it is possible to forge signed documents that are indistinguishable from documents produced prior to the point at which the private key was compromised.

IV. The Haber-Stornetta Renewal Protocol

The basic timestamping protocol has two steps.

- The client requests a timestamp from the TSS.
- The TSS responds with the cryptographic timestamp C.

On receipt of C, the client A stores it as evidence that M existed at time T. where A denotes the client of the TSS, M is the 'message' to be timestamped.

Haber and Stornetta actually propose the use of a cryptographic timestamp function F, which is used to compute C, i.e. C = F(R). Here, X is other data of unspecified form as chosen by the client, and h is a hash-function. The concatenation of M and X is also written as R (for receipt). We have chosen the simplest interpretation of F, namely that F involves concatenation with a timestamp and applying the TSS's digital signature function. In fact, F could also involve the concatenation of data in addition to the timestamp, to support more complex variants of the scheme.

Let T be a timestamping service that is unable to backdate and it requires no record keeping. Absolute timestamp is provided by the inclusion of the time t_r. Relative timestamp is provided by the inclusion of the linking information L_r. Therefore a hybrid timestamp is provided.

Let s_r the stamp for round r.
1) u sends y_r and $ID_r = ID_u$ where ID_u is the unique identification for user u, to T.
2) T computes the timestamp $s_r = {}_{sigT}(C_r)$, where $C_r = (r, t_r, ID_r, y_r; L_r)$ $L_r = (t_{r-1}, ID_{r-1}, y_{r-1}, H(L_{r-1}))$
3) For next request from user v, T sends $(s_r, ID_{r+1}=ID_v)$ to u.

V. Advantages

In SMOCK, a few key pairs can support secure communication of a very large network. According to Algorithm 1, 18 key pairs in the network can support end-to-end secure communication among up to 1000 nodes without resilience consideration.

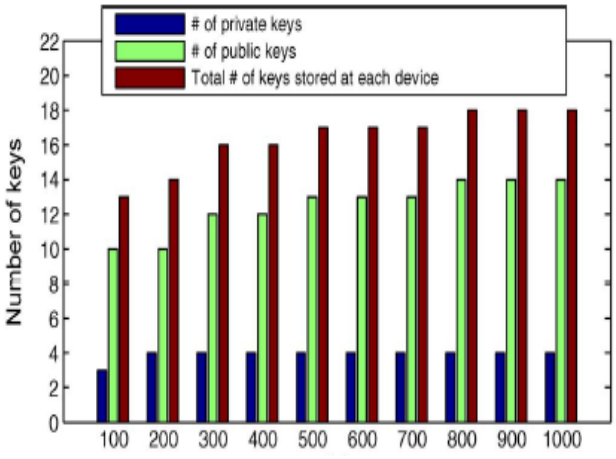

Fig-2

The minimum number of keys needed at each node for typical mission-critical network sizes is shown in above figure. Therefore, we can achieve a very small memory footprint under the scheme with $n \leq 1000$.

VI. Conclusion

The SMOCK scheme, which adopts the combinatorial design of cryptographic keys achieve lightweight key management in an effective manner. In order to measure the resilience of key management protocols, we derived four different types of adversaries varying in their capability with regard to seamless monitoring and software manipulation. In short, our scheme provides a very strong resilience; both past and future key secrecies against node capture by all the adversary types except the strongest one.

VII. Future Enhancement

The idea of SMOCK can be extended to other applications, such as broadcast authentication. Previous works on broadcast authentication by using a one-way hash chain cannot scale to a large number of senders. In addition, authentication delay is increased under packet losses and probabilistic broadcast since authentication in TESLA relies on continuous packet arrival from the source to delivery of the authentication key

Based on the SMOCK idea, a combinatorial hash-chain sharing scheme can be designed: A hash chain pool HC is constructed for the whole network and nodes store the commitment information for all of the hash

chains in HC. To operate more number of nodes timestamp scheme can be introduced. It enables the centralized server to have a control of all the current nodes.

REFERENCES

[1] "A Group-Based Key Management Protocol for Mobile Ad Hoc Networks", Qing Chen., Xiaodong Lin, Sherman Shen, Kazuo Hashimoto., and Nei Kato.

[2] "A Forward & Backward Secure Key Management in Wireless Sensor Networks for PCS/SCADA", Hani Alzaid, DongGook Park, Juan González Nieto, Colin Boyd, and Ernest Foo

[3] " Renewing Cryptographic Timestamps", Sattam S. Al-Riyami and Chris J. Mitchell

[4] "Secure key exchange and encryption Mechanism for Group communication in Wireless ad-hoc Networks", S. Sumathy and B.Upendra

[5] "Cryptography and Network Security", William Stallings, Pearson Education Asia.