Authenticated Key Management Protocols for Internet of Things

Celia Li
Department of Electrical and Computer Engineering
Ryerson University
Toronto, Canada

Cungang Yang
Department of Electrical and Computer Engineering
Ryerson University
Toronto, Canada
e-mail: cungang@ee.ryerson.ca

Abstract—**The Internet of Things (IoT) provides transparent and seamless incorporation of heterogeneous and different end systems. It has been widely used in many applications such as smart homes. However, people may resist the IOT as long as there is no public confidence that it will not cause any serious threats to their privacy. Effective secure key management for things authentication is the prerequisite of security operations. In this paper, we present an interactive key management protocol and a non-interactive key management protocol to minimize the communication cost of the things. The security analysis show that the proposed schemes are resilient to various types of attacks.**

Keywords- Internet of things; Authentication; Key management; Self-certified keys

I. INTRODUCTION

The Internet of Things (IoT) comprises of billions of devices that can sense, communicate, compute and potentially actuate[1][2][3]. IoT involves accessing, monitoring and controlling various sensors and devices over the internet. A great example of the IoT application is smart homes. Household systems like smart smoke-alarms, air quality sensors, smart doorbells, and home monitoring devices can now communicate with smart watches, and activity trackers. After an activity tracker assessed your sleep – determining when you are in light sleep – it can tell your alarm clock to go off. Your alarm clock in union with your phone will check the weather – just before you wake up (based on your preference and sleep cycle) and tell air conditioners in your car and your home to change the temperature accordingly. Navigation apps on your smart phone – after gathering information from your weather app – can predict how the weather will affect traffic congestion, and plan a route to your work. As the communication between IOT devices may include sensitive and critical data, the security requirements for any IoT-based system are high. To set up a security channel between different devices such as an air quality sensor and a smart watch, a number of security operations (authentication, authorisation, and data integrity) are needed. Since key management is the prerequisite of these security operations, the motivation of this research is thus to develop pairwise key generation and rekeying schemes for IoT devices.

So far, the research on the secure key issues of the IoT is focused on homogenous and heterogeneous wireless sensor networks. Perrig [4] presented a suite of security protocols optimized for sensor networks that they called 'SPINS'. The suite is built upon two secure building blocks, each performing individual required work: SNEP and TESLA. SNEP offers data confidentiality, authentication, integrity, and freshness, while TESLA offers broadcast data authentication. The TESLA protocol, used on regular networks, is modified as a SPINS for use in resource-constrained WSNs. Disadvantages of this scheme include TESLA overhead from releasing keys after a certain delay and possible message delay. A non-interactive key management approach is introduced in the article "self-certified keys - concepts and applications" [5]. This scheme allows the computation of a session key in a non-interactive manner. Non-interactive key management protocol involve minimal interaction among the nodes of the network which requires global clock. In a key pre-distribution scheme[6][7][8], some keys are preloaded into each sensor before sensor deployment. After deployment, sensor nodes undergo a discovery process to set up shared keys for secure communications. This scheme ensures to some probability that any two sensor nodes can communicate using a pairwise key. This scheme does not, however, ensure that two nodes always are able to compute a pairwise key to use for secure communication.

The contribution of this work is developing pairwise key generation and rekey schemes for IoT devices. In particular, we bring in a novel interactive key management protocol which is resilient to attacks and save communication cost. Moreover, we propose a secure non-interactive key management protocol which further reduces the communication cost close to zero.

II. AN INTERACTIVE KEY MANAGEMENT SCHEME

The interactive key management scheme between devices such as device A and S is comprised of two phases that is shown in Figure 1. In phase 1, A requests to communicate with S. They mutually authenticate each other with a Ticket-based authentication protocol and generate a Pairwise Master Secret (PMK). In phase 2, following the establishment of the PMK, a session key rekey protocol is executed to confirm the existence of the PMK and the liveliness of the peers; the session key rekey protocol supports Perfect Forward Secrecy (denoted PFS) which refers to the property that disclosure of long-term PMK does not comprise the session keys from earlier runs.

978-1-5386-7244-0/18/$31.00 ©2018 IEEE
DOI 10.1109/ICIOT.2018.00024

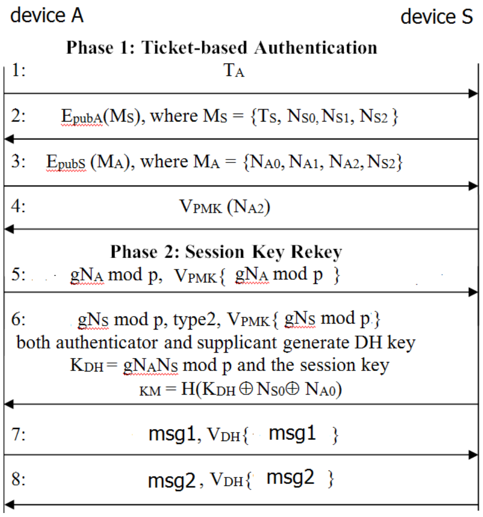
device A
device S
Phase 1: Ticket-based Authentication
1: T_A
2: $E_{pubA}(M_S)$, where $M_S = \{T_S, N_{S0}, N_{S1}, N_{S2}\}$
3: $E_{pubS}(M_A)$, where $M_A = \{N_{A0}, N_{A1}, N_{A2}, N_{S2}\}$
4: $V_{PMK}(N_{A2})$
Phase 2: Session Key Rekey
5: gN_A mod p, $V_{PMK}\{gN_A$ mod p $\}$
6: gN_S mod p, type2, $V_{PMK}\{gN_S$ mod p$\}$
both authenticator and supplicant generate DH key
$K_{DH} = gN_AN_S$ mod p and the session key
$K_M = H(K_{DH} \oplus N_{S0} \oplus N_{A0})$
7: msg1, $V_{DH}\{$ msg1 $\}$
8: msg2, $V_{DH}\{$ msg2 $\}$

Figure 1: Interactive Key Management Protocol

1) Phase 1: Authentication and PMK Generation

Tickets are used to establish the trust relationships among entities. For example, devise A will trust devise S if the ticket of S is valid and issued by the ticket agent it trusts. A ticket agent is defined as an authority who issues and manages various types of tickets and can be trusted by various entities in IoT. Before deployment of IoT devices, the network operator, denoted by *OP*, requests tickets from a ticket agent, one per device, and preinstall the ticket for each node. The *OP* is also responsible for requesting and distributing new tickets before the current tickets expire.

With the design of tickets in the design of the key management protocol, the key generation and negotiation of IoT devises do not need the involvement of the third party, such as the key distributed center or authentication server. The messages exchange only between the pair of devices dramatically reduce the communication cost of the network.

In phase 1, device A and S exchange their tickets and verify the validity of each other's tickets. The trust relationship between A and S from the same network is based on their exchanged tickets which should be issued by a same ticket agent. The results of the protocol are mutual authentication of the pair and the generation of a shared PMK key which is the basis for the following process to create the session key for data confidentiality.

2) *S*ession key rekey*:*

The session key rekey protocol is shown in phase 2 of Figure 1. Here, we assume g and p are public information known by both A and S.

(1) In the first message, gN^A mod p, $V_{PMK}\{gN^A$ mod p$\}$. Device A generates a random number N_A and calculate the MAC value of gN^A mod p with the PMK key. Device S authenticates A.

(2) S generates a random number N_S, gN_S mod p and calculate the MAC value of gN_S mod p with the PMK key. With this step, A authenticates S.

Both A and S then calculate DH key $K_{DH} = gN_AN_S$ mod p and their shared session Key K_M by applying a hash function H to the message $\{K_{DH} \oplus N_{S0} \oplus N_{A0}\}$ where N_{S0} and N_{A0} are the random numbers generated in steps (1) and (2). That is, $K_M = H(K_{DH} \oplus N_{S0} \oplus N_{A0})$.

(3) A sends an acknowledgement message, msg1, $V_{KM}\{AA$, msg1$\}$, to S. S authenticates A.

(4) S sends an acknowledgement message, msg2, $V_{KM}\{$msg2$\}$, to A. A authenticates S.

III. THE NON-INTERACTIVE KEY MANAGEMENT PROTOCOL (NON-INT)

A. Overview

The authenticity of public keys in a public cryptosystem is gained in two different ways: either it is verified by its certificate, or it is verified implicitly during the use of the keys. The latter is introduced by Girault as self-certified keys[9]. Self-certified keys are not verified until it is used for cryptographic function such as signature verification. Public keys of each node are verified without the aid of its public key certificate or an online Certificate Authority (CA)[5]. The concept of self-certified keys is employed in this paper due to its simple non-interactive rekey mechanism. In this section, by coupling the ticket-based technique with the self-certified keys, we obtain a fully non-interactive key management protocol for IoT. In contrast with prior work [5], our techniques for session key update do not require any interaction and do not involve any reliable broadcast communications among devices. Here, we present a new scheme that offers both device A and S to compute or rekey a session key in a non-interactive manner. We achieve this result by using the user-controlled key progression. Compare with interactive key management schemes, the new non-interactive approach further reduce the communication cost of the session key generation and rekey to zero or close to zero.

B. Bootstrapping

The network is initialized by the network operator *OP*. *OP* chooses large primes *p* and *q* with *q|(p-1)* (q is a prime actor of p-1). *OP* chooses a random number $K_A \in Z_q^*$ with order q and generates its (public, private) key pair (y_Z, x_Z). We assume that the public key y_Z, p, q and g are preinstalled to every node of the network. To issue the private key for a device A with identifier ID_A, *OP* computes the signature parameter $r_A = g^{k}{}_{A}$ (mod p) and $s_A = x_Z * h(ID_A, r_A) + k_A$ (*mod q*). r_A is called the guarantee and $x_A = s_A$ is its private key. The public key of A can be computed by any node that has y_Z, ID_A and r_A using the following equation $y_A = y_Z^{h(ID_A, r_A)} * r_A$ (*mod p*). We denote this initial key pair as $(x_{A,0}, y_{A,0})$. We assume that each node has installed the initial pair of public and private key issued by the *OP*.

C. Self-Certification

The non-interactive key management protocol is comprised of two phases. phase 1 in Figure 2 is in charge of the PMK key generation and rekey which is interactive. Phase 2 discuss the session key generation and rekey which is non-interactive.

For the original non-interactive scheme, for each PMK update, the device A and S need to exchange $r_{A,\ t} = g^{KA,t}$ mod p and $r_{S,\ t} = g^{Ks,t}$ mod p where $1 \le t \le n$. This scheme waste valuable bandwidth because each $r_{A,\ t}$ or $r_{S,\ t}$ could be as large as 2048 bits or 3072 bits and number n is uncertain since the number of session keys update within a PMK rekey interval is unknown.

1) Phase 1: Ticket-based Authentication and PMK Generation

In phase 1 of the non-interactive key management protocol.

(1) First message T_A includes R and $V_{PMK}\{R\}$. Device A generates a random number R and calculate the MAC value of R with the PMK key. Device S authenticates A because only A has the shared PMK to generate the MAC value.

(2) Upon receiving the second message, A decrypts it using its private key, and verifies the digital signature of the ticket agent who issued the ticket T_S using the ticket agent's public key. A receives three random numbers N_{S0}, N_{S1}, N_{S2} and gN_S mod p where N_S is the secret value generated and hold by S, A verifies other information of ticket T_S such as the ID of the ticket agent who issued T_S and the ticket expiry date.

(3) If the above verifications succeed, A retrieves S's public key from ticket T_S, and generates a message M_A containing gN_A mod p, *l, ΔT, F* and three random numbers N_{A0}, N_{A1} and N_{A2}. N_A is the secret value generated and hold by A. A then encrypts message M_A using S's public key, and sends the encrypted message to S. S will decrypt the message using its private key and retrieve gN_A mod p, the length of the one-way hash chain *l,* session key progression interval *ΔT,* lifetime of the PMK *F* and three random numbers N_{A0}, N_{A1} and N_{A2}. Again, S authenticates A in this message.

(4) In message 4, S verified A's authenticity. Finally, both A and S calculate the DH key as $K_{DH} = gN_AN_S$ mod p and derive the initial $V_{A,1}$ and $V_{S,1}$ value as $H(K_{DH} \oplus N_{A0} \oplus N_{S0})$. In phase 1, whenever generate or rekey the PMK, A and S generate their new secret values N_A and N_S which are the basis to derive new session keys in the second phase. After phase 1, both A and S know their common secret value V as well.

2) Phase 2: Session Key Generation and Rekey

$x_{A,0}$, $x_{S,0}$, $y_{S,0}$ and $y_{A,0}$ are assigned by the OP. $y_{S,0}$ and $y_{A,0}$ are exchanged by A and S with the second and third messages of phase 1.

We define that $K_{A,t} = K_{A,t\text{-}1} * V \bmod p = K_{A,t\text{-}1} * H(K_{DH} \oplus N_{A0} \oplus N_{S0}) \bmod p = K_{A,1} * (H(K_{DH} \oplus N_{A0} \oplus N_{S0}))^{t\text{-}1} \bmod p$ and $K_{A,1} = N_{A.}$

Thus, $K_{A,t} = N_A * (H(K_{DH} \oplus N_{A0} \oplus N_{S0}))^{t\text{-}1} \bmod p = N_A * V^{t\text{-}1} \bmod p$.

$r_{A,\ 1} = g^{KA,1} \bmod p = gN_A \bmod p$

$r_{A,\ t} = g^{KA,t} \bmod p = g^{NA * (H(K_{DH} \oplus N_{A0} \oplus N_{S0}))t\text{-}1 \bmod p} \bmod p = gN_A * V^{t\text{-}1} \bmod p$.

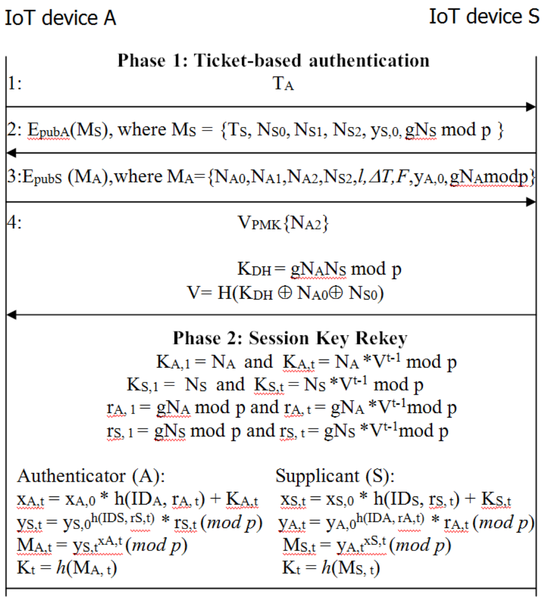
IoT device A
IoT device S
Phase 1: Ticket-based authentication
1:
T_A
2: $E_{pubA}(M_S)$, where $M_S = \{T_S, N_{S0}, N_{S1}, N_{S2}, y_{S,0}, gN_S \bmod p\}$
3:E_{pubS} (M_A),where $M_A = \{N_{A0}, N_{A1}, N_{A2}, N_{S2}, l, \Delta T, F, y_{A,0}, gN_A \bmod p\}$
4:
$V_{PMK}\{N_{A2}\}$
$K_{DH} = gN_AN_S \bmod p$
$V = H(K_{DH} \oplus N_{A0} \oplus N_{S0})$
Phase 2: Session Key Rekey
$K_{A,1} = N_A$ and $K_{A,t} = N_A * V^{t\text{-}1} \bmod p$
$K_{S,1} = N_S$ and $K_{S,t} = N_S * V^{t\text{-}1} \bmod p$
$r_{A,\ 1} = gN_A \bmod p$ and $r_{A,\ t} = gN_A * V^{t\text{-}1} \bmod p$
$r_{S,\ 1} = gN_S \bmod p$ and $r_{S,\ t} = gN_S * V^{t\text{-}1} \bmod p$
Authenticator (A):
$x_{A,t} = x_{A,0} * h(ID_A, r_{A,\ t}) + K_{A,t}$
$y_{S,t} = y_{S,0}^{h(IDS,\ rS,t)} * r_{S,t}\ (mod\ p)$
$M_{A,t} = y_{S,t}^{xA,t}\ (mod\ p)$
$K_t = h(M_{A,\ t})$
Supplicant (S):
$x_{S,t} = x_{S,0} * h(ID_S, r_{S,\ t}) + K_{S,t}$
$y_{A,t} = y_{A,0}^{h(IDA,\ rA,t)} * r_{A,t}\ (mod\ p)$
$M_{S,t} = y_{A,t}^{xS,t}\ (mod\ p)$
$K_t = h(M_{S,\ t})$

Figure 2 Non-interactive Key Management Protocol

For devise S, $K_{S,1} = N_S$ and $K_{S,t} = N_S * V^{t\text{-}1} \bmod p$

$r_{S,\ 1} = gN_S \bmod p$ and $r_{S,\ t} = gN_S * V^{t\text{-}1} \bmod p$

In phase 2, A keeps its secret value $K_{A,1} = N_A$ and derives $K_{A,t} = N_A * V^{t\text{-}1}$ mod p for the following sessions. S keeps $K_{S,1} = N_S$ and derives $K_{S,t} = N_S * V^{t\text{-}1}$ mod p for the following sessions.

On the other hand, to derive the public key of the S, A needs to know $r_{S,\ 1}$ and $r_{S,\ t}$. $r_{S,\ 1} = gN_S$ mod p is transferred to A in message 2 of layer 1 while $r_{S,\ t} = gN_S * V^{t\text{-}1}$mod p can be derived for each session because A know gN_S and V. Each r value we derived will be $\in Z_q^*$ because q is a prime and all r value are modular p and its value must be in Z_q^*.

The initial scheme [5] is not a pure non-interactive key management scheme because in their approach the set of $r_{A,\ t} = g^{Vt}$ mod p is shared through message exchange. Compare with the scheme, our protocol allows A and S to generate the $r_{A,t}$ by themselves, and thus no message exchange are involved.

D. Security Analysis:

For our proposed scheme, the security of the $V_{A,t}$ values depends on the public key algorithm we used in phase 1 which is safe.

The non-interactive has no PFS problem because the PMK has no relationship with the values of $V_{A,t}$ and $V_{S,t}$. If the PMK exposed, it will not compromise the session key.

1) Key Security

In the non-interactive key management protocol, the security of the session rekey procedure of phase 2 depends on the Schnorr signature scheme whose security is based on the intractability of discrete logarithm problems. The Schnorr signature scheme has been provably secure in a random oracle model[10][11].

To derive the value of the session key, the attacker has to figure out $x_{A,t}$ and $y_{S,t}$.

$x_{A,t} = x_{A,0} * h(ID_A, r_{A,t}) + K_{A,t}$
$\quad = x_{A,0} * h(ID_A, gN_A *V^{t-1} \text{mod } p) + K_{A,t} \ \text{mod } p$
$\quad = x_{A,0} * h(ID_A, gN_A *V^{t-1} \text{mod } p) + N_A *V^{t-1} \text{ mod } p$

$y_{S,t} = y_{S,0}^{h(IDS,\, r_{S,t})} * r_{S,t}\ (\textit{mod } p)$
$\quad = y_{S,0}^{h(IDS,}\, gN_S *V^{t-1} \text{mod } p^{)} * r_{S,t}\ (\textit{mod } p)$
$\quad = y_{S,0}^{h(IDS,}\, gN_S *V^{t-1} \text{mod } p^{)} * gN_S *V^{t-1} \text{mod } p\ (\textit{mod } p)$

where only the ID of A and S, p and g are public known. Other parameters are hiding from the attackers. Thus the session keys cannot be disclosed to attackers.

2) Key Refreshment

For the non-interactive key management protocol, the update of PMK is carried out in phase 1 while the session key rekey is automatically implemented by device A and S. Whenever the session key needs rekeying, the phase 2 of each protocol will be carried out.

3) Perfect Forward Secrecy

The only value in phase 1 relating to the generation of session key is V. $V = H(K_{DH} \oplus N_{A0} \oplus N_{S0})$. If the PMK is exposed, it cannot derive DH key. Thus, we can say that the attacker cannot compromise the session key if PMK is exposed.

4) Key Separation:

a. PMK and Session key:

The PFS analysis shows that PMK is independent from the session key. That is, if PMK is exposed, the session key will not be compromised. Due to the same reason, if a session key is exposed, the PMK cannot be compromised either.

b. PMK and DH key:

In the non-interactive key management protocol, DH key $K_{DH} = gN^AN^S$ mod p, the N^A and N^S are secret random numbers that only known by the authenticator and supplicant. The PMK and session key are independent: if PMK is exposed, it does not help to figure out the DH key. On the other hand, if DH key is exposed, the PMK will not be compromised.

c. DH and Session key:

The session key $K_t = h(M_{A,t}) = y_{S,t}^{xA,t}\ (\textit{mod } p)$. To derive the session key, we have to know $x_{A,t}$ and $y_{S,t}$

$x_{A,t} = x_{A,0} * h(ID_A, r_{A,t}) + K_{A,t}$
$\quad = x_{A,0} * h(ID_A, gN_A *V^{t-1} \text{mod } p) + K_{A,t} \ \text{mod } p$
$\quad = x_{A,0} * h(ID_A, gN_A *V^{t-1} \text{mod } p) + N_A *V^{t-1} \text{ mod } p$
$\quad = x_{A,0} * h(ID_A, gN_A * (H(K_{DH} \oplus N_{A0} \oplus N_{S0}))^{t-1} \text{mod } p) +$
$N_A *(H(K_{DH} \oplus N_{A0} \oplus N_{S0}))^{t-1} \text{ mod } p$

$y_{S,t} = y_{S,0}^{h(IDS,\, r_{S,t})} * r_{S,t}\ (\textit{mod } p)$
$\quad = y_{S,0}^{h(IDS,}\, gN_S *V^{t-1} \text{mod } p^{)} * r_{S,t}\ \textit{mod } p$
$\quad = y_{S,0}^{h(IDS,}\, gN_S *V^{t-1} \text{mod } p^{)} * gN_S *V^{t-1} \text{mod } p$
$\quad = y_{S,0}^{h(IDS,}\, gN_S *(H(K_{DH} \oplus N_{A0} \oplus N_{S0}))^{t-1} \text{mod } p^{)} * gN_S$
$*(H(K_{DH} \oplus N_{A0} \oplus N_{S0}))^{t-1} \text{mod } p$

If DH key is exposed, the session key of non-interactive protocol cannot be compromised since only g, p, K_{DH} and IDs of authenticator and supplicant are know. Other parameters are hiding from the attackers. Due to the same reason, if the session key is exposed, the attacker still cannot derive the DH key.

IV. Conclusion

Security has become the central issue for IoT and key management plays a critical role to ensure data confidentiality and integrity. A new design of ticket-based authentication protocol, an interactive key management protocols and a non-interactive key management enhanced the security of 4-way handshake of 802;11i. Security analysis shows that our proposed key management protocols satisfies the principles of PFS, key refreshment and are resilience to attacks.

References

[1] Michelle S. Henriques and Nagaraj K. Vernekar, "Using symmetric and asymmetric cryptography to secure communication between devices in IoT", International Conference on IoT and Application, May 2017.

[2] B.Vinayaga Sundaram; Ramnath.M ;Prasanth.M ;Varsha Sundaram.J, "Encryption and Hash based Security in Internet of Things" 3rd Interational Conference on Signal Processing, Communication and Networking (ICSCN), 2015.

[3]J.Hermans,R.Peeters,andB.Preneel,"ProperRFIDprivacy:Modelandprotocols, "IEEETrans.MobileComput., vol.13, no.12, pp.2888–2902, Dec.2014.

[4] D. Liu and P. Ning, "Establishing Pairwise Keys in Distributed Sensor Networks," Proc. 10th ACM Conf. Computer and Comm. Security (CCS). pp. 52-61, 2003.

[5] Holger Petersen, Patric Horster, "Self-certified keys - Concepts and Applications.", Proc. of conference on Communication and Multimedia Security, Athens, September 22-23, 1997.

[6] A. Perrig et al., "SPINS: security protocols for sensor networks", Proceedings of ACM MOBICOM (2001).

[7] L. Eschenauer, V.D. Gligor, "A key management scheme for distributed sensor networks", Proceedings of the 9th ACM Conference on Computer and Communication Security.

[8] H. Chan, A. Perrig, D. Song, "Random key predistribution schemes for sensor networks", Proceedings of the 2003 IEEE Symposium on Security and Privacy, May 11–14, pp. 197– 213.

[9] M. Girault, "Self-Certified Public Keys", LNCS547, Advances in Cryptology: Proc. Eurocrypt' 91, Springer, pp. 490-497.

[10] P. Horster, M. Michels, H. Peterson, "Meta-ElGamal signature schemes", Proc. 2. ACM Conferences on Computer and Communication Security, pp. 96-107.

[11] Schnorr C.P., "Efficient signature generation by smart cards", *Journal of Cryptology,Vol. 4, No.* 3, pp.161-174, 1994. http://en.wikipedia.org/wiki/Schnorr_signature

[12] M. Long, "Energy-efficient and Intrusion Resilient Authentication for Ubiquitous Access to Factory Floor Information," IEEE Transaction on Industrial Informatics, Vol. 2, No. 1, pp. 40-47, 2006.

[13] David Manz, Jim Alves-Foss and Shanyu Zheng, "Network Simulation of Group Key Management Protocols", *Journal of Information Assurance and Security,* pp. 67-79, January, 2008.

[14]http://www.cacr.math.uwaterloo.ca/conferences/2005/ecc2005/vanstone.pdf