Int. J. of Advanced Networking and Applications
Volume: 01, Issue: 05, Pages: 282-289 (2010)

Customized PKI for SCADA System

Anupam Saxena
Centre for Development of Advanced Computing, Mumbai, India
Email: anupam@cdacmumbai.in
Om Pal
Centre for Development of Advanced Computing, Mumbai, India
Email:ompal@cdacmumbai.in
Zia Saquib
Centre for Development of Advanced Computing, Mumbai, India
Email: saquib@cdacmumbai.in
Dhiren Patel
Indian Institute of Technology Gandhinagar, Ahmedabad, India
Email: dhiren@iitgn.ac.in

ABSTRACT

Security of SCADA (supervisory Control and Data Acquisition) has become a challenging issue today because of its connectivity with the outside world and remote access to the system. One major challenge in the SCADA systems is securing the data over the communication channel. PKI (public key infrastructure) is a well known framework for securing the communication. In SCADA system, due to limited bandwidth and rare communications among some RTUs (Remote Terminal Units), there is a need of customization of general PKI which can reduce the openness of Public Key, frequent transfer of certificates and reduction in DOS (Denial of Service) attacks at MTUs (Master Terminal Units) and RTUs. This paper intends to address the issues of securing data over communication channel in the constrained environment and presents the novel solutions pivoted on key distribution and key management schemes. This paper also presents a set of innovative methods of multicast and broadcast of messages in SCADA system.

Keywords: **Broadcasting, Key Distribution and Management, Multicasting, Public Key Infrastructure, SCADA security**

Date of Submission: February 01, 2010 Date of Acceptance: March 29, 2010

I INTRODUCTION

Supervisory Control and Data Acquisition (SCADA) systems provide means for management, supervisory control, and monitoring of process control and automation systems via collecting and analysing the real time data. Initially these systems were not intended to operate within the enterprise environment, this lead to inability within SCADA components to deal with the exposure to viruses, worms, malware etc. that are commonplace today within the enterprise network.

Due to connectivity of SCADA systems with Internet and the increased risk of cyber attacks, security of such systems have become a challenging issue today. Technology become vulnerable to attacks and technological vulnerability can cause a sever damage on critical infrastructures like electric power grid, oil gas plant and water management system. Protection of such Internet connected SCADA systems from intruders is a new challenge for researchers and therefore, it necessary to apply information security principles and processes to these systems.

SCADA system consists of a human-machine interface (HMI), a supervisory system (controller or MTU), remote terminal units (RTUs), programmable logic controllers (PLCs) and a communication infrastructure connecting the supervisory system to the RTUs.

As the SCADA industry developed, vendors began to adopt open standards and the total number of SCADA protocols commonly in use was reduced to smaller number

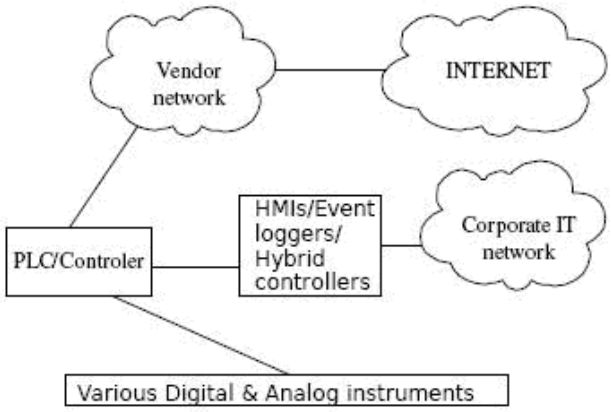

Fig. 1. SCADA System Architecture

of protocols that were popular and were being promoted by industry, including MODBUS, Ethernet/IP, PROFIBUS, ControlNet, InfiNET, Fieldbus, Distributed Network Protocol (DNP), Inter-Control Center Communications Protocol (ICCP), Telecontrol Application Service Element (TASE) etc. The most widely used communication protocols in SCADA system are DNP3 (Distributed

Volume: 01, Issue: 05, Pages: 282-289 (2010)

Network Protocol version 3.0), IEC 62351 and Modbus. In the beginning due to isolation of SCADA system from rest of the world, cyber security was not an issue when these protocols were designed. As the system is becoming more interconnected to the outside world, the necessity of securing the system is increased.

Cryptographic techniques are widely used for providing many security features like higher security, reliability, and availability of control systems etc. to the SCADA systems. There is a need of establishment of secure keys before application of cryptographic techniques. In this paper first we discuss key distribution and key management issues and then we present our PKI based approach for securing the SCADA system in an efficient way with the facility of message broadcast and multicast as an additional feature; scheme is designed such that it also fulfils the essential requirement of availability along with integrity in the SCADA systems.
In next section, we discuss key challenges and related work, In Section 3 we present our proposed (CPKI) key distribution and key management technique with Conclusions in Section 4 and References at the end.

II. KEY CHALLENGES AND RELATED WORK

Along with the connectivity of SCADA system to the Internet, many security threats have emerged, like unauthorized access of devices, capturing and decoding network packets and malicious packet injection in the network.
For securing the SCADA system from these threats, there are certain security requirements, which can be classified as:

1. Authentication: It is very important to ensure that the origin of an object is what it claims to be.
2. Integrity: The manipulation of messages between nodes and insertion of new nodes can be hazardous. A malicious attacker could cause physical damage if they have the ability to alter or create messages.
3. Confidentiality: Ensuring that no one can read the message except the intended receiver.
4. Availability of resources: Insuring that resources are available for legitimate users. Insuring that the information is there when needed by those having authorization to access or view it.

For securing the system, these challenges along with installation and configuration limitations of the system need to be considered. Ludovic Piètre-Cambacédès[3] has pointed out the some constraints of SCADA system:

1. Limited computational capacity: The most of the RTUs are having low computational capabilities.
2. Limited Space Capacity: Memory available in the most of the RTU is quite low.
3. Real-time processing: If transmission and processing of data in SCADA systems not become timely, then it may cause of latency problems.
4. Key freshness: In the absence of key freshness entities would keep re using an ‘old’ key for longer time, which might have been compromised, so there is a need of key freshness for eliminating the possibility of such new security hole.
5. Small number of messages: Due to low bandwidth, number of messages exchanged between nodes need to be minimum and also length of messages need to be also small.
6. Multicasting: Though multicasting is not an essential requirement for SCADA system, but it might be required in certain cases where the facility of common message announcement to a selected group of RTUs is required.
7. Broadcasting: There should be facility of common message announcement to all devices

There is a need to keep these constraints in mind before building a security mechanism for the system. Many efforts have been made in the area of key distribution and key management for securing the System but still there is a scope for improvements.

Sandia National Laboratories [1] proposed a cryptographic key management and Key Establishment approach for SCADA (SKE) in 2002. This technique, divides the communication into two categories: first is 'controller to subordinate (C-S) communication' and second is 'subordinate to subordinate (S-S) communication'. The C-S is a master-slave kind of communication and is ideal for symmetric key technique. The C-C is a peer-to-peer communication and it can use asymmetric key approach. In C-S communication, each controller has a Long Term Key (LTK) shared with its subordinate. The controller also has its own General Seed Key (GSK), which it sends to each of its subordinates. The General Key (GK) is a 128 bit hash of GSK. For communication, the sender obtains a Session Key (SK) from GK. And this SK is used for encryption/decryption. All keys used are of 128 bit in length.

Information Security Institute, Queensland University of Technology, Australia [2] proposed Key Management Architecture for SCADA systems (SKMA). In this scheme a new entity 'Key Distribution Center (KDC)' came into picture, which is used to maintain long term keys for every node. Whenever a new node joins the system, a node-KDC key is manually installed in it. When two nodes want to communicate then with help of node-KDC key, a long term 'node-node key' is generated. Again using the node-node key, a session key is generated for data communication.

In 2002, Mingyan Li [5] proposed a key management approach with multicast and broadcast facility. This approach specifies the shared keys to be stored in the database of MTU (2n-1 keys) and RTU (1+log 2n keys) and these keys are used at run time, where 'n' is number of RTUs. However, this approach provides multicasting in a limited fashion.

Donghyun Choi[6] also proposed a multicast and broadcast scheme with additional computation at run time at MTU side, by doing so the number of keys at MTU is 'n-1' lesser than Mingyan approach. Like Mingyan's approach, this approach also provides multicasting in a limited fashion.

Simple Public Key Infrastructure (SPKI) was developed starting in 1995. Simple Distributed Security Infrastructure (SDSI) is a new design for a public-key infrastructure, designed by Professors Ronald L. Rivest and Butler Lampson of MIT's Laboratory for Computer Science,

members of LCS's Cryptography and Information Security research group [18]. The SPKI/SDSI facilitates to build a secure distributed computing system which may be scalable. SPKI/SDSI builds public keys as principals and each public key as a certificate authority itself [17]. Each principal can issue certificates. SPKI/SDSI provides two types of certificates; these are “name certificates” and “authorization certificates”. Name certificate defines a local name in the local name space of certificate issuer. Authorization certificate grants authorization to the subject of the certificate. A single certificate cannot define both i.e. a name and granting an authorization; so a certificate is either a name certificate or an authorization certificate, but can not be both.

SCADA system is an interconnected infrastructure, where smooth, reliable and continuous operations are desired. Protecting such infrastructures includes a number of challenges, such as secure interaction among nodes, resilience and robustness of entire system. The Wireless Sensor Networks (WSN) have intelligent distributed control capabilities, and the capability to work under severe conditions, so some of the schemes of this area may be useful for securing SCADA systems, as µPKI.

In the paper “Lightweight PKI for WSN µPKI”, Benamar Kadri , Mohammed Feham , and Abdallah M’hamed proposed a lightweight implementation of Public Key Infrastructure (PKI) [16]. Their proposed protocol called µPKI uses public key encryption only for some specific tasks as session key setup between the base station and sensors giving the network an acceptable threshold of confidentiality and authentication. µPKI only implements a subset of a PKI services . Here all sensors are connected to a Base station, which is having more computational and energy power compared to sensors; and each sensor is capable to use both symmetric and asymmetric encryption. The public key of the base station is installed at sensor node with the help of an off-line dealer. It ensures that only legitimate sensors can authenticate base station trough its public key. The public key is used to authenticate the base station by the sensors in the network, and private key is used by the base station to the decrypt data sent by sensors, which ensures confidentiality. For secure end to end transmission between nodes and Base station, µPKI uses two types of handshakes. The first handshake is between the base station and sensors where a sensor generates a random key, encrypts it with the public key of the base station and sends to Base station, by decrypting it , the base station saves the session key in a global table where are saved all the session keys corresponding to each sensor in the network. The second handshake is for securing sensor to sensor communication; where one of the two sensors sends a request (which contains the identifier of the corresponding node) to the base station to establish a secure tunnel with the other sensor. When base station receives this request, it decrypts this and generates a random key, then encrypts a copy of this key for each sensor using the corresponding session keys, and sends it to each sensor [16].

In this paper, we concentrate on accomplishment of fundamental security goals of communications, where secure communication is needed with limited resources, along with broadcast and multicast capabilities. We are using the PKI approach in a customized manner, which basically reduces the openness of Public Key in such a way that it provides broadcasting and multicasting (for any group of RTUs) in the limited environment of SCADA system.

III. PROPOSED METHODOLOGY

Our scheme assumes that messaging takes place among three entities CA (Controlling Authority), MTU and RTU. Scheme uses a high computing capable entity (CA) as an additional element to provide security in SCADA systems. Scheme uses asymmetric key approach with putting restriction on accessibility of each key. Scheme uses 'Key

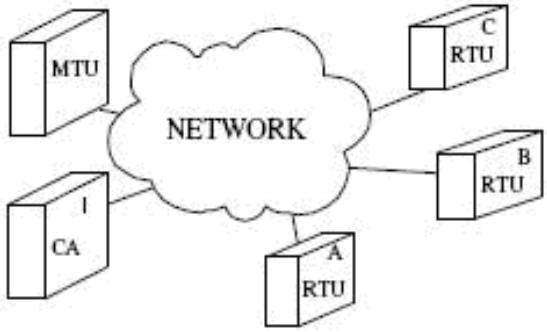

Fig. 2. SCADA System with CA

and CounterKey pair'. Though these keys are analogues to Private-Public keys ('Key' corresponds to Private Key, and 'CounterKey' corresponds to Public key), but are not same in the true sense, because in the scheme'CounterKeys' are not publicly accessible to all.

In actual SCADA network, there are Sub-MTUs associated to MTU which takes care of a particular section of RTUs.

For showing the clear working of our scheme in a simple way, we are showing MTU in place of Sub-MTU, which takes care of their corresponding subsection of RTUs with the help of CA of that subsection.

In turn, these various subsections communicate with each other with the help of their representative CAs which can communicate with each other by establishment of trust among peer CAs.

Scheme assumes that for a sub section, there is one MTU (with moderate computational power), one CA (with very high computational power) and n number of RTUs (with low computational power). MTU and RTUs are attached to CA.

Initially long term keys are stored manually at each node, 'n+1' unique keys stored at CA (corresponding to each MTU/RTU), and one at each RTU/MTU which belongs to that particular RTU/MTU.

RTU Ri has key Li where i = 1,2,...n and and similarly the MTU has key, named “Ln+1”. The CA passes Key and CounterKeys (K and CK) pair to corresponding RTUs and to

Volume: 01, Issue: 05, Pages: 282-289 (2010)

MTU by using the pre shared-keys (Li, where i = 1,2,...'n+1'). Also it passes MTU's CounterKey (CKMTU) to each RTU, and CA's CounterKey (CKCA) to each RTU and to MTU.

For maintaining key freshness, there is a provision of re-distribution of "Key-CounterKey pair" after certain period of time using long term keys, and also these long term keys

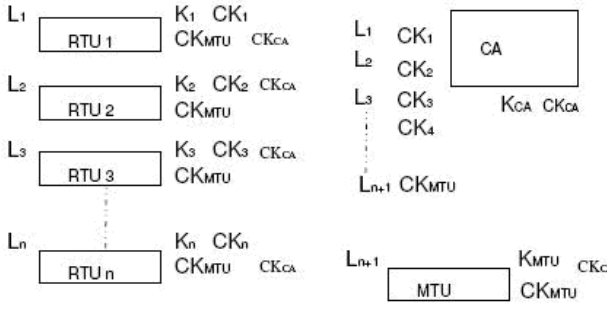

Fig. 3. Initial Key Setup

would be replaced manually after a long time period. This fixed time is can be adjusted depending on the requirement of system.

Each node (RTU/MTU) maintains a database of counter keys of other nodes to which it wants to communicate and if counter key is not available in it's database then it requests to CA for obtaining required CounterKey before initiation of communication. All keys and CounterKeys are confidential, any node can request to CA for getting CounterKey of other node only if it wants to communicate with that node. After initial communication both party (sender and receiver) store the CounterKeys of each other in their database for future use.

In general CA authenticates any node on the network by issuing certificate to that node but in this case extra computational overhead of certificate might be an issue for some RTUs because of their low processing power. In such cases we have two options, first one is to reduce certificate as much as possible by removing unnecessary extra fields from it [3], and second is to replace certificate value with a single unique value like "MAC Address" of the corresponding entity.

In proposed scheme, node encrypts hash of its own MAC address with its own key and forwards this value to other node, to prove its authenticity.

For load balancing, and to avoid CA to become a single point of failure, scheme also recommends the deployment of distributed CA.

A. Proposed scheme categorizes the communication into three categories as follows

1. RTU to RTU communication.
2. MTU to RTU communication.
3. RTU to MTU communication.

1. RTU to RTU communication

In rare cases, an RTU may be interested in communicate to another RTU, in this case RTU will not store CounterKeys of all RTUs but it will only store CounterKey of other RTU at run time if it is needed, which it takes from CA.

If RTU A wants to communicate with RTU B then it checks the CounterKey of RTU B in its database if it is there then RTU A encrypts the message with CounterKey of RTU B and sends to RTU B. If CounterKey of RTU B is not available in database of RTU A then it calculates Hash of its own MAC address, signs it by its own Key, and then encrypts the resulting block (along with address of RTU B) by CA's CounterKey, and sends to CA. When CA gets the request, it decrypts it by its own Key, and then checks the signature of RTU A with the help of Hash Function and by using CounterKey of RTU A. After checking the validity of the RTU B, the CA prepares a response. If both RTUs are

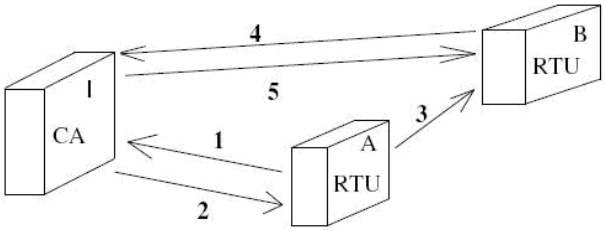

Fig. 4. RTU to RTU Communication

genuine, then CA sends a response (which contains CounterKey of corresponding RTU B) to RTU A by signing the Hash of MAC address of CA by its own key and encrypting the resulting block by the CounterKey of RTU A.

After getting the response of CA, RTU A decrypts the Block

Table 1. CounterKey storage in databases of communicating RTUs

States	RTU A	RTU B	Encryption
Initial State	CK_{MTU}, CK_{CA}, CK_A	CK_{MTU} CK_{CA}, CK_B	
State after Message 1	"	"	Message encrypted with CK_{CA}
State after Message 2	CK_{MTU}, CK_{CA}, CK_A, CK_B	"	Message encrypted with CK_A
State after Message 3	"	"	Message encrypted with CK_{CA}
State after Message 4	"	"	Message encrypted with CK_{CA}
State after Message 5	"	CK_{MTU}, CK_{CA}, CK_B, CK_A	Message encrypted with CK_B

by its own Key and after decrypting hash value by CounterKey of CA, compares hash of MAC address of CA with received hash value from CA, if values match, then RTU A stores the CounterKey of RTU B in its database.

Now RTU A sends message to RTU B by encrypting the message with CounterKey of RTU B. After receiving it, the RTU B starts communication if it has the CounterKey of RTU A in its database, otherwise it does not respond.

If the RTU A does not get reply from RTU B then it waits for a fixed time (this time duration may vary from system to system and will depend on requirements), after that it sends a communication initiation request to RTU B, by calculating hash of its own MAC address, signing it by its Key, and then encrypting the resulting block (along with address of RTU A) by CA's CounterKey, and sends to RTU B. After receiving the request from RTU A, RTU B passes this request to CA. When CA gets the request, it decrypts it by its own Key, and then checks the signature of RTU A with the help of Hash function and by using CounterKey of RTU A. After checking the validity of the sender (who initiated the request), the CA prepares a response. If initial sender A is genuine, then CA sends a response (which contains CounterKey of RTU A) to RTU B by signing the hash of MAC address of CA by its own key and with encrypting the resulting block by the CounterKey of RTU B.

After getting the response of CA, RTU B decrypts the Block by its own Key and after decrypting hash value by CounterKey of CA, compares the hash, if values match then RTU B stores the CounterKey of RTU A in its database. Now the communication can start.

2. MTU to RTU communication

When an MTU wants to communicate to an RTU then it checks the CounterKey of that RTU in its database, if it is there then MTU encrypts the message with CounterKey of RTU and sends to RTU. If CounterKey of RTU is not present there in database of MTU, then it (MTU) calculates Hash of its own MAC address, signs it by its Key, and then encrypts the resulting block (along with the address of RTU) by

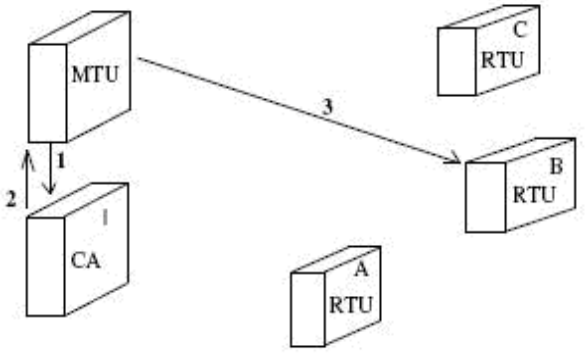

Fig. 5. MTU to RTU Communication

CounterKey of CA and sends to CA. When CA gets the request, it decrypts it by its own Key, and then checks the signature of MTU with the help of Hash Function and by using CounterKey of MTU.

After checking the validity of the MTU, the CA prepares a response. If MTU is genuine, then CA sends a response (which contains CounterKey of corresponding RTU) to MTU by signing the Hash of MAC address of CA by its own key and encrypting the resulting block by the CounterKey of MTU.

After getting the response of CA, MTU decrypts the block by its own Key and after decrypting hash value by CounterKey of CA, compares hash of MAC address of CA

Table 2. CounterKey storage in databases of communicating MTU and RTU

States	MTU	RTU B	Encryption
Initial state	CK_{MTU} CK_{CA}	CK_{MTU} CK_{CA}, CK_B	
State after Message 1	"	"	Message encrypted with CK_{CA}
State after Message 2	CK_{MTU} CK_{CA} CK_B	"	Message encrypted with CK_{MTU}
State after Message 3	"	"	Message encrypted with CK_B

with received hash value from CA, if values match, then MTU stores the CounterKey of RTU in its database and MTU sends the message to RTU by encrypting the message with CounterKey of RTU.

Now both the MTU and RTU B contain CounterKeys of each other in their databases, and they can communicate.

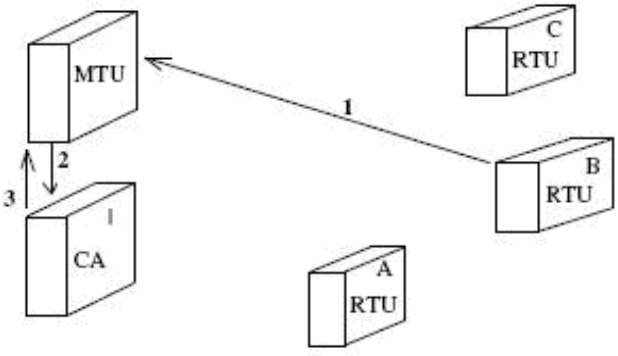

Fig. 6. RTU to MTU Communication

3. RTU to MTU communication

When an RTU wants to communicate to MTU then it sends a message to MTU, by encrypting it with CounterKey of MTU (which is known to every RTU). After receiving it, the MTU starts communication if it has the CounterKey of that RTU in its database, otherwise it does not respond.

If the RTU does not get reply from MTU then it waits for a fixed time (this time duration may vary from system to system and will depend on requirements), after that it sends a communication initiation request to MTU, by calculating

hash of its own MAC address, signing it by its Key, and then encrypting the resulting block (along with the address of RTU) by CounterKey of CA.

After receiving the request from RTU, MTU passes this request to CA. When CA gets the request, it decrypts it by its own Key, and then checks the signature of RTU with the help of Hash Function and by using CounterKey of corresponding RTU. After checking the validity of the RTU, the CA prepares a response. If RTU is genuine, then CA sends a response (response contains CounterKey of RTU) to MTU by signing the hash of MAC address of CA by its own key and then encrypts the resulting block by the CounterKey of MTU.

Table 3. CounterKey storage in databases of communicating RTU and MTU

States	MTU	RTU B	Encryption
Initial state	CK_{MTU}, CK_{CA}	CK_{MTU} CK_{CA}, CK_{B}	
State after Message 1	"	"	Message encrypted with CK_{CA}
State after Message 2	"	"	"
State after Message 3	CK_{MTU}, CK_{CA}, CK_{B}	"	Message encrypted with CK_{MTU}

After getting the response of CA, MTU decrypts the block by its own Key and after decrypting hash value by CounterKey of CA, it compares the hash of MAC address of CA with received hash address, if values match then MTU stores the CounterKey of RTU in its database. Now both the MTU and the RTU are having the CounterKeys of each other in their databases, and they can communicate.

B. Broadcast and Multicast Support

Proposed scheme also provides support for broadcasting; though the multicasting is not an essential requirement of SCADA systems, it provides multicasting as an additional feature. Its implementation uses specific flag bits.

MTU broadcasts message to all entities by encrypting the message with its Key and the message can be decrypted at each entity by using CounterKey of MTU because CounterKey of MTU is available at each RTU.

When any RTU or MTU wants to multicast a message, then it sends its multicast initiator request to CA (containing addresses of the entities, for which multicast is desired), after encrypting it with CounterKey of CA. CA keeps CounterKey of every entity (RTUs/MTU) with it. CA fetches the addresses of end entities (to which multicast is desired) from the multicast request of RTU/MTU, takes the multicast data and encrypts it with its own Key and then sends the resulting block (which contains the address of original initiator of the multicast) to the fetched addresses along with the address of original initiator of the multicast.

CounterKey of CA and MTU are available only at genuine RTUs and these CounterKeys are not open to all, so any fraud entity can not get CounterKey of CA or MTU hence it can not decrypt the broadcast/multicast message. For multicasting, addressing mechanism is used, so scheme assume that only RTUs decrypt the multicast message whose addresses are available in address block of message.

In all kind of communications, the communicating entities specify that whether they are sending a normal message (which any RTU or MTU sends to either by encrypting the data with the CounterKey of corresponding entity), or it is some other kind of message like broadcast/multicast or request/response.

Normal messages are identified by setting F1 to 0, and flag bit F2 need not to be checked in this case. For covering other conditions, it sets flag bit F2 to '0' for Broadcast/Multicast and '1' for request/response, along with setting F1 to 1

Table 4. Table of flag bits

F1	F2	Direction	Description
0	0 or 1	RTU/MTU to RTU/MTU	Normal message
1	0	RTU/MTU to RTU/MTU	Request to start communication
		RTU/MTU to CA	Request to CA to fetch CounterKey of other entity
		CA to RTU/MTU	Response from CA with CounterKey of desired entity
1	1	RTU/MTU to CA	Multicast initiator request
		MTU to RTU/MTU	Broadcast
		CA to RTU/MTU	Multicast

C. Dynamic arranged database for optimal key storage

Due to low memory, MTUs and RTUs can store a limited number of counter keys in their databases. This limited storage of keys can cause an extra overhead at run time, if required key is not available in database of MTU/RTU.

To overcome this problem, scheme uses 'dynamic arranged database for optimal key storage'. Each RTU/MTU stores CounterKey of CA in first row of database and counter key of MTU in second row of database. Always new Counter Key will be stored in third row of database and all keys will shift downward by one row. Key at the bottom row is removed if database is already full. If any CounterKey is used by the node from its database then this used key will be shifted to third row and all CounterKeys (which were above in database from used counter key) will shifted downward by

Volume: 01, Issue: 05, Pages: 282-289 (2010)

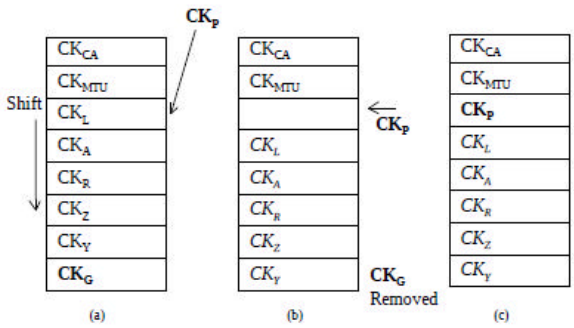

Fig. 7. Insertion of new CounterKey, when database is fully filled

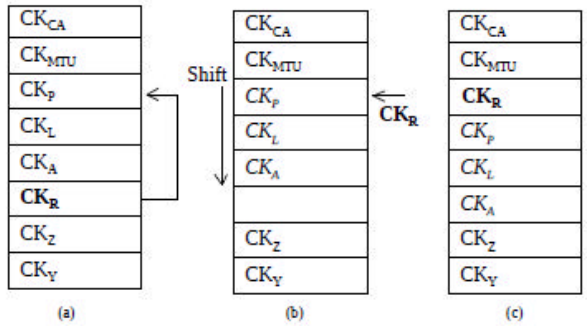

Fig. 8. Shifting of existing CounterKey within the database, with its use.

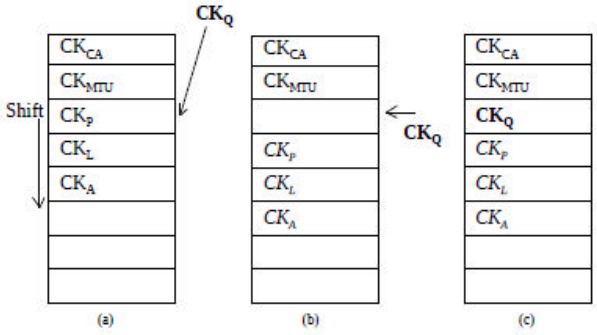

Fig. 9. Insertion of new CounterKey, when database is partially filled

one row. The place of CounterKeys those are at lower position from used counter keys will be unchanged.

IV. CONCLUSION

In SCADA system, due to limited bandwidth and rare communications among some RTUs (Remote Terminal Units), there is a need of customization of general PKI which can reduce the openness of Public Key, frequent transfer of certificates and reduction in DOS (Denial of Service) attacks at MTUs (Master Terminal Units) and RTUs.

We have discussed various existing issues, challenges, and schemes on cryptographic key distribution and key management for SCADA systems. We have devised and proposed a new scheme emphasizing on key distribution and key management in constrained environment using the strength of PKI.

Our attempt is to address the issues of securin g the data over the communication channel in the constrained environment and presented a novel solution pivoted on key distribution and key management schemes. It supports broadcasting of messages and also provides multicasting as an additional feature for SCADA system.

REFERENCES

[1] C. L. Beaver, D.R. Gallup, W. D. NeuMann, and M.D. Torgerson "Key Management for SCADA (SKE)", printed at Sandia Lab March 2002.

[2] Robert Dawson, Colin Boyd, Ed Dawson, Juan Manuel, * RQ] 2DH 1 LHR 36. 0 $ – A Key Management Architecture for SCADA Systems", Fourth Australasian Information Security Workshop (AISW-NetSec 2006).

[3] Ludovic Piètre-Cambacédès, Pascal Sitbon "Cryptographic Key Management for SCADA Systems, Issues and Perspectives", Proceedings of the 2008 International Conference on Information Security and Assurance (isa 2008) Pages 156-161.

[4] Mariana Hentea, "Improving Security for SCADA Control Systems", Interdisciplinary Journal of Information, Knowledge, and Management Volume 3, 2008.

[5] Mingyan Li, R. Poovendran and C. Berenstein "Design of Secure Multicast Key Management Schemes With Communication Budget Constraint", IEEE Communications Letters, Vol. 6, No. 3, March 2002.

[6] Sungjin Lee, Donghyun Choi, Choonsik Park, and Seungjoo Kim" An Efficient Key Management Scheme for Secure SCADA Communication", Proceedings Of World Academy Of Science, Engineering And Technology Volume 35 November 2008.

[7] Yongge Wang and Bei-Tseng Chu "sSCADA: Securing SCADA Infrastructure Communications", August 2004.

[8] http://www.ncs.gov/library/tech_bulletins/2004/tib_04-1.pdf.

[9] Tanveer Ahmad Zia "A SECURITY FRAMEWORK FOR WIRELESS SENSOR NETWORKS" PhD Thesis, University of Sydney, February 2008.

[10] T. Paukatong "SCADA Security: A New Concerning Issue of an Inhouse EGAT-SCADA", Electricity Generating Authority of Thailand, 53 Charan Sanit Wong Rd., Bang Kruai, Nonthaburi 11130, Thailand.

[11]Barry Charles Ezell "Infrastructure Vulnerability Assessment Model (IVAM) ", Risk Analysis, Vol. 27, No. 3, 2007.

[12]http://www.digitalbond.com/index.php/category/scada-protocols.

[13] Joe Weiss PE, CISM "Assuring Industrial Control System (ICS) Cyber Security"

http://www.cooperpower.com/products/protective/idea/pdf/080827_JW_Cybersecurity.pdf.

[14] American Gas Association-Assuring Industrial Control System (ICS) Cyber Security; www.waterresearchfoundation.org/research/../2969/AGAPart1.pdf

[15] Peter Gutmann "PKI: It's Not Dead, Just Resting" IEEE Computer, vol. 35, no. 8, pp. 41-49, Aug. 2002.

[16] Benamar Kadri, Mohammed Feham, and Abdallah M'hamed "Lightweight PKI for WSN µPKI" accepted for International Journal of Network Security, Vol.10, No.3, PP.194–200, May 2010.

[17] Carl Ellison "SPKI / SDSI " October 2004.

[18] http://groups.csail.mit.edu/cis/sdsi.html.

Authors Biography

Anupam Saxena received a B. Tech. degree in Information Technology from UP Technical University, Lucknow, India. He started working as Lecturer in the Institute of Engineering & Rural Technology, Allahabad, India from the year 2006; and in 2007 joined G-DAC as a Project Engineer. He is presently working as Staff Scientist in the Computer Networks & Internet Engineering (CNIE) division of G-DAC Mumbai. He is actively involved in the Network Administration and network security related research projects and also involved in providing corporate training in the field of Information security. His present research interests are in areas of Computer Network, Protocol Development and Information Security.

Om Pal received a Bachelors Degree (B.E) in Computer Science and Engineering from Dr. B. R .Ambedkar University Agra (India), MBA in Operation Management from Indira Gandhi National Open University, Maidan Garhi, New Delhi (India) and pursuing PhD in area of network security from Indian Institute of Technology (IIT) Bombay, Mumbai (India). He Joined NTPC (National Thermal Power Corporation) in 2005 as IT Resource Person. He joined C-DAC (Centre for Development of Advanced Computing) in 2006 and presently working as Staff Scientist.
His present research interests are in areas of network security. He is interested in cryptography, key management schemes and in area of intrusion detection and prevention system. He has published papers in International Journals and International Conferences.

Zia Saquib received a Bachelors Degree (B.E) in Electrical Engineering from Regional Engineering College- Rourkela, India (Now NIT-Rourkela) and a M.S. Degree in Electrical Engineering (Communication Engineering Stream) from Florida Institute of Technology, USA. He joined G-DAC in 2002 as Member Technical Staff and was designated as Group Coordinator and Chairman of the Management Committee of Advanced Computing Training School (ACTS) in 2003 and was given responsibility as Head & Program Coordinator, ACTS and e-Governance Software Group in 2004.
He is presently Executive Director of G-DAC Mumbai and Bangalore (Electronic City) Centers. He is also Head of Computer Networks & Internet Engineering (CNIE), Biometrics, and Software Engineering (SENG) Research Groups.
His present research interests are in areas of network security & biometrics. He is interested in cryptography & key management schemes as applied to constrained environments (such as SCADA and Mobile Ad hoc Networks). In biometrics, his interests are in novel algorithms, performance evaluation and biometric cryptosystems. In addition, he is also involved in development of Network Intrusion Prevention Systems, Storage Networks and Messaging Middleware for performance critical e-Government systems.

Dr. Dhiren Patel is currently a Professor of Computer Science & Engineering at IIT Gandhinagar, Ahmedabad, India (on leave from NIT Surat, India). He carries 20 years of experience in Academics, Research & Development and Secure ICT Infrastructure Design. His research interests cover Security and Encryption Systems, Web Services & Programming, SOA and Cloud Computing, Digital Identity Management, e-Voting, Advanced Computer Architecture etc. Besides numerous journal and conference articles, Prof. Dhiren has authored a book "Information Security: Theory & Practice" published by Prentice Hall of India (PHI) in 2008. He is active in Indo-UK, Indo-French, and Indo-US security research collaborations.