Key Management Schemes of Wireless Sensor Networks: A Survey

Syed Muhammad Khaliq-ur-Rahman Raazi[1], **Zeeshan Pervez**[1] **and Sungyoung Lee**[1]
[1]*Department of Computer Engineering, Kyung Hee University, Global Campus, Korea*

ABSTRACT

Key Management is the most important issue in the security of Wireless Sensor Networks. It helps in maintaining the confidentiality of secret information from unauthorized users. Sometimes, it is also useful for verifying the integrity of exchanged messages and authenticity of the sender. In this chapter, we will discuss evolution of wireless sensor networks, possible security threats in wireless sensor networks and the evolution of key management strategies for Wireless Sensor Networks. We will also present a detailed critical analysis of the current state-of-the-art key management strategies used in Wireless Sensor Networks. Apart from that, we will also discuss the challenging issues and problems, which still need attention from researchers.

INTRODUCTION

Key management is the most important aspect of security in any type of network. Other aspects of security such as authentication and privacy also depend upon key management. Concept of key in computer science and wireless sensor networks is same as that in our daily lives. Basically, key is a secret that is known to relevant parties only just like in a household, where only insiders have the house's keys. Relevant parties use keys to conceal secret information that they need to exchange with each other. Apart from that, keys are also used to identify parties, which have permission to access certain information.

In real life scenario, physical keys are used to lock a room, a bag or a cupboard. Only authorized persons are given copies of the key, which can open the lock. In this way, things inside lock are kept confidential from other users and only authenticated users can access inside the lock. The same concept is applied in the scenario of computer science and computer networks but in a different way.

In computer networks data and information, which are not something physical, needs to be secured. They are a stream of bits stored on electronic devices and exchanged between two or more parties through a wire or a wireless medium. In wireless medium, it is impossible to prevent unauthorized parties from eavesdropping and impersonating authorized parties. In wired medium also, it is nearly impossible. In order to keep these streams of bits confidential from an outsider, sender performs certain mathematical operations on these bits and sends them to the receiver through a wired or wireless medium. Upon receiving this information, the receiver performs some mathematical

operations, which are reverse of the mathematical operations performed by the sender, to produce the original stream of bits. However, just performing some mathematical operations can't secure important information. If an adversary knows the mathematical operations, it can crack confidential information very easily. Even if the adversary doesn't know the mathematical operations used for decryption in certain scenario, it can guess them by applying already known mathematical operations of cryptography. If a pair of communicating nodes does not employ an already known mathematical operation, then they have to agree upon new mathematical operations and their reverse every time they communicate, which further increases complexity of the problem. In order to simplify this problem, random keys are used.

Keys are fixed length streams of random bits, which are known to only the authorized parties. Sender encrypts data / information in the key i.e. performs mathematical operations on data / information and key collectively. This produces a stream of bits, which does not reveal any information about the original stream of bits. Only authorized parties can decrypt or come to know original data / information. This allows us to use those mathematical functions, which are proven to be secure and have inverse operations, which can reveal original information with the help of key. In order to have keys readily available for every communication, keys need to be managed securely and efficiently. Key management of any computer network depends upon its characteristics, limitations and applications.

Wireless sensor networks consist of a large number of low cost resource constrained sensor nodes. Each node has to sense certain phenomena and forward its readings towards a central node, which is also called the base station. Figure 1 shows an example of a wireless sensor network. Base station uses sensed information according to the application requirements. Wireless sensor network applications include habitat monitoring, military surveillance, border monitoring and health care.

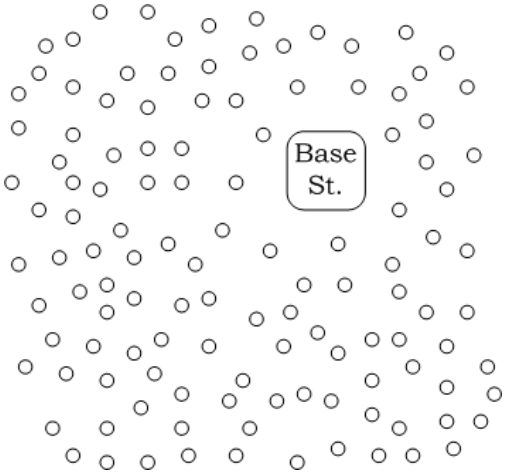

Figure 1. A generic wireless sensor network

Sensor nodes have low memory, computation, communication and energy capabilities. One has to be very careful about resource consumption while proposing a solution for

any problem in this domain. Same thing holds for key management. Apart from making sensor networks as secure as possible, key management should introduce as less overhead as possible. Normally, this is a trade-off, which also depends upon the application scenario. In the next section, we will shed some light on the background of key management and wireless sensor networks so that the readers can understand the motivations behind different key management solutions proposed so far. After that, we will discuss the key management solutions proposed for wireless sensor networks so far. However, before discussing key management solutions, we will shed some light on the possible security threats or attacks that can take place in wireless sensor networks. Also, we will also discuss future research directions in key management before we draw conclusions from this chapter.

BACKGROUND

Key management has been an important research area since the start of computer networks. Before that, research was mainly focused on the security of computing devices. Before the start of computer networks, it was emphasized that computers must have security programs that secure the computer itself and the peripheral devices that were used to transfer information from one point to another.

With the start of computer networking and later on its commercialization, importance of key management grew substantially. Information was shared among different computers through a wired or wireless network and it was not possible to monitor all the links. Also, types of attacks on confidential information grew substantially. All these circumstances invoked research for better security especially key management.

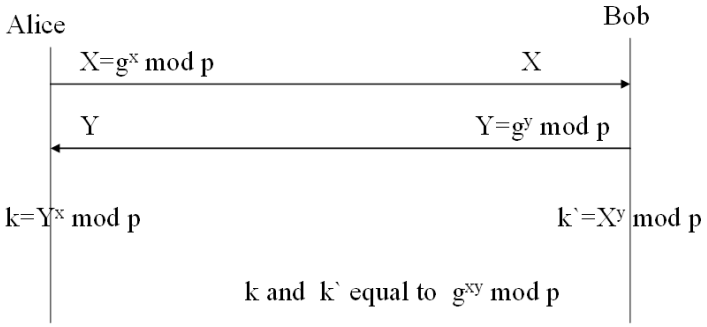

Figure 2. Diffie-Hellman Key Exchange Algorithm

With growth of computer networks, information was also shared among different computers, who had not interacted with each other before. In order to secure such communications, Diffie-Hellman key exchange algorithm was proposed [Diffie et. al., 1976]. In Diffie- Hellman key exchange, two parties agree on two large prime numbers *g* and *p* and choose one random value each. For example, Alice chooses a random value x and Bob chooses a random value y. Then they compute a secret key in the way shown in Figure 2. While this protocol provided a secure way for two unknown parties to communicate, it did not have any authentication mechanism.

Soon after the Diffie-Hellman protocol, concept of Public Key Cryptography was introduced by Rivest, Shamir and Adleman [Rivest et. al., 1978]. Their protocol is normally referred to as RSA protocol. In RSA algorithm, every computer computes its public and private keys as shown in Figure 3. Anything encrypted using a public key, can be decrypted using the corresponding private key. By using this protocol, any two parties can communicate with each other using their public-private key pairs.

1. Randomly select two large prime numbers 'p' and 'q'.
2. Compute N = p.q and ø(N) = (p-1) (q-1).
3. Select any encryption key 'e' such that $1 < e < ø(N)$ and gcd (e,ø(N)) = 1.
4. Solve the following equation to find decryption key 'd': -

   $e \cdot d = 1 \bmod ø(N)$

   where $0 \leq d \leq N$
5. publish the public encryption key: KU={e,N}.
6. keep the secret private decryption key: KR={d,p,q}.

Figure 3. Computation of Public and Private key pairs in RSA Algorithm

Although RSA and Diffie-Hellman secure communication between two parties, who do not know each other, they don't have any mechanism to authenticate the other party. In early 1990s, when internet was growing rapidly, Kerberos [Kohl et. al., 1993] matured. Kerberos is a network authentication protocol based on secret key cryptography. Kerberos is based on the concept of trusted third party, which authenticates the communicating parties and provides keys or communication. Using Kerberos, a communicating party can be sure about the authenticity of the other communicating party. Apart from keeping information confidential, Kerberos also protects against the replay of data packets sent from sender to receiver. Apart from these advantages, Kerberos also has some drawbacks. In Kerberos, the trusted third party is a single point of failure. Also, clocks of communicating parties must be synchronized with the trusted third party for this protocol to work.

With the growth of internet in the era of 1990s many applications, which used the internet, were developed. These applications used some predefined protocols such as Simple Mail Transfer Protocol (SMTP) for e-mail, File Transfer Protocol (FTP) for file sharing and Hypertext Transfer Protocol (HTTP) for sharing information over the internet. In the design of internet, all application level protocols depend upon transport layer protocols for communication. TLS [Dierks et. al. 1999], which encapsulated and secured the transport layer of the internet, was proposed in late 1990s. TLS uses concepts trusted third party and public key cryptography to establish secure connections between the communicating parties.

With the evolution of the internet and growth in the use of personal electronic devices, electronic hardware also evolved rapidly. Size of computation and communication hardware like processor, memory and antenna kept shrinking along with its cost. This goes on until the current day. New models, with more capabilities and reduced size, are introduced in market, previous ones become obsolete and their cost falls to earth very quickly. Reduced size and reduced cost of hardware led to the use of computing devices in monitoring certain activity, phenomena or biometric from human body. This led to the evolution of wireless sensor networks an opened a lot of new research challenges.

Wireless Sensor networks consist of low cost computing devices, which can also sense their environment and forward sensed data to a nearby node. Sensors can be used to sense temperature, air pressure, salinity, moisture, movement, biometric or any other phenomena. Exact number of sensor nodes used in an application depends upon the nature of application. Some applications, like border monitoring or military surveillance applications require hundreds or may be thousands of sensor nodes. If we need to support such a huge number of sensors in a single wireless sensor network, we need to keep the cost of a single node to a minimum.

Reducing the cost of a sensor node has a direct effect on its computation power, communication power and memory. Also, sensor nodes have limited battery life. In certain circumstances, like battlefields, it is impossible to recharge their battery. Limited battery life, lesser memory, low computation power and small range of communication capability of a sensor node makes wireless sensor network a special type of computer network. Wireless sensor networks differ from the internet not only because of the limited capabilities of sensor nodes but also because of ad hoc and data centric nature of wireless sensor networks. In wireless sensor networks, all sensed data is directed towards a central computing device called the base station. Also, old nodes may die down, new nodes may join the network and nodes may change their position during normal network operation of wireless sensor networks.

Research in network security matured with the spread of internet. Many security mechanisms were proposed even after Kerberos and TLS. However, these security mechanisms were best suited for the internet and not for wireless sensor networks. Main reason why these traditional security mechanisms were not viable for wireless sensor networks was that they were too heavy for simple sensor devices. Even these days, either sensor devices do not have enough computation power and / or memory to handle traditional security mechanisms or these mechanisms cause a lot of energy drainage from sensor nodes.

Inapplicability of traditional security mechanisms in wireless sensor networks opened a new research area in network security. A number of researchers have proposed novel and interesting ideas for key management in wireless sensor networks. Some ideas apply to simple sensor networks and some ideas apply to clustered sensor networks [Gupta et. al. 2003], [Younis et. al. 2004] (As wireless sensor networks evolved, some researchers proposed clustering techniques to increase efficiency of wireless sensor networks). In our discussion, we will also identify whether a scheme is applicable to clustered sensor

networks or simple sensor networks or both. However, before proceeding with the discussion of key management schemes, it is better to discuss the types of attacks that can occur in wireless sensor networks. Discussion of attacks against security will make it easy for us to identify strengths and weaknesses of various key management schemes.

SECURITY THREATS IN WIRELESS SENSOR NETWORKS

Main goal of a key management scheme is to ensure confidentiality of information. Also, keys can be helpful in authenticating legitimate nodes. An adversary may try to crack secret key and extract confidential information from the messages exchanged between communicating nodes. If keys are used for authentication purposes, adversary may try to act as a legitimate node and try to extract confidential information from other nodes. While trying to crack a secret key, adversaries try to learn message patterns and guess the secret key. Also, they try to save some encrypted messages, which they can replay later on. In order to prevent adversaries from guessing secret keys, it is important to refresh keys at appropriate time intervals. Time intervals depend upon frequency of communication and frequency of key usage.

Apart from trying to crack secret information, adversary can also harm a sensor network in several other ways. It can try to jam wireless signals of a sensor networks. Also, it can try to create noises and disrupt communication. In other words, adversary can carry out denial-of-service attacks. Apart from that, an adversary can try to drain sensor nodes' energy by initiating bogus messages or replaying old messages. Although many of such attacks, like signal jamming, can not be handled by key management schemes, but we think it is important to list them at least. Readers can refer to the security mechanisms at physical layer to find remedy of jamming attacks. [Zia et. al. 2006] classifies all security attacks in wireless sensor networks in four classes: interruption, interception, modification and fabrication. Interruption is when a communication link in interrupted. Interception takes place when a sensor node or its data is compromised. In modification, adversary gains access and tampers with the data. Finally, fabrication takes place when an adversary injects false data into the network. [Karlof et. al. 2003], [Zhu et. al. 2006] and [Tanya et. al. 2006] are also very helpful resources for studying about security attacks that can take place in wireless sensor networks.

Broadly speaking, we can classify attacks on wireless sensor networks in two categories: outsider attacks and insider attacks. In outsider attack, adversary is not a part of the network. For example, jamming attack is carried out on physical layer of the sensor nodes by a node, which is not part of the sensor network so it is classified as an outside attack. In insider attack, an insider node (a node from within the network) is compromised through node tampering or through a weakness in its system software. Now, we will discuss each attack one by one in a bit detail. We will also identify whether an attack is a type of insider attack or an outside attack and in which category it falls according to [Zia et. al. 2006].

Passive Information Gathering and Message Corruption: Passive information gathering and message corruption are the simplest attacks that can take place in wireless

sensor networks. If information is not encrypted, an adversary can listen to the communication passively. Passive information gathering can be classified as interception carried out by an outsider node. In message corruption, adversary modifies contents of a message before it gets to the receiver. All key management schemes designed for wireless sensor networks can provide protection against these attacks.

Node Compromise: An adversary gets control of a node by compromising it through a weakness in its system software. After compromise, adversary gains access to data, information and cryptographic keys stored in the node. Adversary can also cause the node to malfunction and generate inaccurate data. This attack is carried out by an outsider node but later on, adversary uses the compromised node to carry out an insider attacks. Adversary can interrupt communication, intercept messages, modify data packets transferred from one node to another and fabricate messages.

Node Tampering: An adversary tampers with a sensor node physically and gains access to data, information and cryptographic keys stored on it. Just like node compromise, adversary can also cause the tampered node to malfunction. Classification of this type of attack is same as that of node compromise.

False Node: Malicious node in introduced in the network by an adversary. This malicious node tries to inject malicious data and attract other nodes to send data to it. For example, it can advertise shortest route to the base station so that other nodes route their packets through it in order to save energy.

Node Outage: Adversary removes a node from the sensor network or a node's energy is exhausted. Typically, these types of attacks are not dealt by the key management schemes. Rather, there are other security mechanisms, which help in resolving this issue and finding other more appropriate routes to the base station.

Traffic Analysis: An adversary can analyze the communication patterns of a sensor network and cause harm to the network. For example, if all packets are routed through a single node, it can try to compromise that node first. This is the reason why cluster head nodes should have more security as compared to other nodes in clustered sensor networks. Apart from that, traffic analysis attacks also highlight the need for refreshing keys at regular intervals.

Acknowledgement Spoofing: This type of attack targets routing algorithms of wireless sensor networks. In this case, an adversary can spoof link layer acknowledgement after overhearing packets. Suppose there are two nodes A and B. Node A wants to send some data to the base station through node B but node B is dead. A compromised or outsider node E overhears the initial message sent by node A and spoofs an acknowledgement to node A at link layer. Based on spoofed acknowledgement, node A starts forwarding its packets to the base station through node E. After that, node E drops some or all packets forwarded to it by node A. This attack is classified under interruption and takes place if the forwarding node is not authenticated.

Spoofed, Altered or Replayed Routing Information: An attacker can use compromised nodes or outside malicious nodes to play with routing information in such a way that it creates routing loops, attracts or repels network traffic, alters source routes or generate false error messages. Apart from other hazards, this type of attack cause large network delays and also drain the battery power of sensor nodes very quickly.

Selective Forwarding: In this case, an adversary compromises an insider node or uses an outsider malicious node to create a black hole in the target sensor network. The malicious node deliberately drops data packets in order to disrupt working of the target sensor network.

Sinkhole Attacks: In this case, adversary tries to attract network traffic towards a malicious node. After that, the malicious node can carry out selective forwarding on the traffic. Sinkhole and selective forwarding attacks are most effective if the compromised node or the outsider malicious node is near the base station.

Sybil Attacks: In these types of attacks, a malicious node presents multiple identities in the sensor network. In doing so, it can either steal other nodes' identities or it can try to fabricate new identities itself. Basically, sybil attacks reduce effectiveness of fault tolerant schemes like distributed storage. Also, sybil attacks can affect routing algorithms. For example, it can cause a routing algorithm to determine two disjoint paths, which are not disjoint in reality.

Wormhole Attacks: Wormhole attack is carried out using two distant malicious nodes, which can communicate with each other, through an out-of-band communication channel, which is invisible to the underlying sensor network. One of the malicious nodes is placed near the base station and the other one is placed near the sensor nodes, which generate data. Using this low latency link the malicious node, which is placed near the data generating sensor nodes, convinces data generating nodes that it is just one or two hops away from the base station. This can cause sinkhole in the network. Also, this can create routing confusion especially in malicious node's neighbours, who might think that the other malicious node, near to the base station, is their neighbour.

Hello Flood Attacks: In this case, an adversary sends a HELLO packet itself or replays a routing protocol's HELLO packet with more signal strength. As a result, each of the other sensor nodes thinks that the malicious node is its neighbour. Then the malicious node can advertise a low latency link creating a wormhole. Also, sensor nodes waste their energies in responding to HELLO floods.

DoS (Denial of Service) Attacks: An adversary can carry out DoS attack by disrupting communication between sensor nodes. Typically, DoS attacks occur at the physical layer of wireless sensor networks. Radio Jamming, which we have already discussed, is a classical example of a DoS attack.

While going through the types of attacks in wireless sensor networks, a very important thing to note is that most attacks involve either a malicious outsider node or a

compromised insider node. Therefore, most of the attacks can be avoided by having highly effective node authentication and compromised node eviction scheme. Strength of both node authentication and compromised node eviction depend upon the underlying key management scheme. This highlights the importance of key management in wireless sensor network security once again. In the next section, we will discuss various key management solutions for wireless sensor networks proposed so far in the literature.

KEY MANAGEMENT

After going through the previous section, we learn the importance of key management in wireless sensor networks. Also, we learn what is expected from a key management scheme that is designed for wireless sensor networks. Whenever we think of the points that should be kept in mind while designing a key management scheme for wireless sensor networks, resource constraints (processing and memory capabilities) and energy constraint of the sensor nodes always come first. Otherwise, traditional key management schemes, which we have already discussed in the section of background, are very useful. It was due to the constraints of sensor nodes that always lightweight key management schemes are proposed for wireless sensor networks. However, maintaining required level of security in wireless sensor networks is also very important. Now we will discuss various key management schemes for wireless sensor networks proposed in the literature so far. [Xiao et. al. 2007] presented a very useful survey of key management schemes for wireless sensor networks. We will start from most simple key management solutions for wireless sensor networks and then discuss more complex ones later on. Also, we will include some of the useful findings of [Xiao et. al. 2007].

According to [Xiao et. al. 2007], a key management scheme, which is designed for wireless sensor networks, should support certain characteristics. Strength of any key management scheme for wireless sensor networks depends on how many of those characteristics are present in any scheme. When we talk about the required level of security, it means that any key management scheme should provide authenticity, confidentiality, integrity, scalability and flexibility. Apart from authenticity and confidentiality, maintaining integrity of a secret key is also very important. With integrity, we mean that the adversary should not be able to forge a key or change it altogether. Since the number of nodes may vary in wireless sensor networks and in some cases it may increase substantially, key management scheme should be scalable to cater for this scenario. Finally, wireless sensor networks are dynamic in nature. Old nodes, which run out of energy, die down with time and new ones can be added at any time. Key management scheme for wireless sensor networks should be flexible enough to cater for such scenarios.

Apart from the required level of security, key management schemes designed for wireless sensor networks should also cater for constraints related to sensor nodes. Apart from limited bandwidth, memory and computation capabilities, sensor nodes do not have any prior knowledge regarding their deployment. Limited transmission range and limited battery life also add to the constraints. Limited battery life is the primary reason why asymmetric key management strategies are not considered suitable for wireless sensor

networks. Asymmetric key management schemes perform intense mathematical calculations, drains a lot of energy from sensor nodes. Since sensor nodes can only transmit up to short distances, some sensor network data collection techniques employ in-networking processing [Intanagonwiwat et. al. 2000] and [Karlof et. al. 2004]. In in-network processing, all nodes send their data to a few nodes, which aggregate messages and transmit only processed information towards the command node. In order to avoid unnecessary communication, some schemes in wireless sensor networks require nodes to overhear messages from other nodes [Karlof & Li et. al. 2003] and [Madden et. al. 2002]. It may not be possible for a key management scheme to have all the above characteristics. Also, it is very difficult to design a key management scheme that is optimal for all topologies of sensor networks and their applications. Nonetheless, application developers can choose suitable key management schemes according to their application requirements.

Single Group Key for a Network: It is by far the simplest key management scheme used for wireless sensor networks. In this case, a single key is loaded into every sensor node before deployment. All sensor nodes communicate using that single key. This scheme is very lightweight in terms of memory, computation and communication requirements. It is also flexible and scalable but at the same time, it is also very vulnerable. If a single key is used for a long time, chances of cryptanalytic attack on the key gets higher and it is easier for an adversary to compromise the key. In this scenario, if a node is compromised or the key is revealed in some other way whole network is compromised. There is no way we can refresh the key or revoke the compromised sensor node from network and retain rest of the network.

Pair-wise Key Establishment: Establishing pair-wise key between every pair of nodes in a sensor network is the most secure key management scheme for wireless sensor networks. Every node is preloaded with a key for communication with every other node. For instance, if there are *n* nodes in a network every node will have *n-1* keys stored in its memory. This scheme is possibly the most secure for wireless sensor networks. Pair-wise key establishment not only provides confidentiality and authenticity in a network but also provides efficient mechanism for revocation of a compromised sensor node in the network. However, this scheme is not at all efficient in terms of scalability and memory requirements. If the number *n* becomes too large as in many applications of wireless sensor networks, this scheme becomes impractical. Also, communication between every pair of sensor nodes is not necessary in wireless sensor networks.

Random Pair-wise Key Establishment: [Chan et. al. 2003] argue that all nodes in a sensor network need not share pair-wise keys. In their approach, two nodes share a pair-wise key with some probability *p* and *p* must be chosen carefully in order to keep the network connectivity up to a desired level. Also, node revocation does not need to involve the base station. A node's status in the network depends upon the consensus among nodes, with which it communicates. If a certain number of nodes, with which it communicates, say that node A is compromised, all of them will terminate their communication with node A. Although this scheme works well for small networks, it does not scale well enough if network size becomes too large.

Trusted Key Distribution Center (KDC): In this approach, we mitigate the drawbacks of pair-wise key management by storing all pair-wise keys in a key distribution center. This key distribution center can be the base station or a cluster head node in clustered sensor networks. Although this approach is secure and resilient against node capture and node replication, this approach is also not scalable. This is because every pair of nodes has to obtain keys from the trusted base station for every session. Apart from the communication overhead introduced in this approach, links around the base station may become overloaded. If trusted KDC is a sensor node e.g. a cluster head node in clustered sensor networks, then its memory requirements increase manifold. Also, it must have far better energy and communication capabilities. In addition to all that, the trusted key distribution center becomes a single point of failure especially if it is one of the sensor nodes.

Random Key Pre-distribution Scheme: In wireless sensor networks, it is not necessary that keys are established among every pair of sensor nodes. For a wireless sensor network to work, it is important that every sensor node gets sufficient bandwidth and neighbouring nodes, who can relay its messages to the base station through various paths. For example, if node A has 15 nodes in its neighbourhood, it can establish pair-wise keys with only 4 of them and those 4 neighbouring nodes can provide node A distinct routes to the base station, then node A does not need to establish pair-wise keys with rest of the 11 nodes. Random key pre-distribution scheme was proposed by [Eschenauer et. al. 2002]. In the first phase of their scheme, a key ring of K keys and their identifiers is stored in the memory of each node prior to deployment. Every pair of nodes shares a key with some probability. In discovery phase, every node broadcasts its key identifiers and challenges to find those nodes, with which it shares a key. If some keys are left unused after the discovery phase, they can be used to establish keys between nodes, who do not share a common key. For example, node A shares a key *x* with node B and node B shares another key *y* with node C while nodes A and C do not share a key. If node B has a key *z*, which it does not share with any node, it can send key *z* to both node A and node C so that they can communication with each other using *z*. In this scheme, there are group keys that are shared between the base station and all other nodes. In order to revoke a compromised sensor node, the base station compiles the list of keys known to the compromised node, uses a group key to sign the list and broadcasts it into the network using another one. Upon receiving the list, all nodes delete the keys, which are known to the compromised node, from their memory. Apart from the fact that shortest path to the base station might not be established in this scheme, another drawback is that node revocation might cause many other links, which use one of the deleted keys, to break.

Q-Composite Random Key Pre-distribution scheme: Q-Composite random key pre-distribution scheme, which was an improvement to the random key pre-distribution scheme, was proposed by [Chan et. al. 2003]. In Q-Composite scheme, two sensor nodes must share at least *q* number of keys if they want to establish a link between themselves. In this way, two linked nodes will have other keys for communication if one of the keys is compromised. In this case, size of the random key pool need to be reduced to maintain the probability that two nodes share *q* common keys. This poses another security problem:

adversary will need to compromise only a few sensor nodes to compromise most of the keys.

Multi-path Key Reinforcement Scheme: In basic random key pre-distribution scheme, multiple nodes may share more than one key. In this case, if one node is compromised, there is a chance that links between other non-compromised sensor nodes may also be compromised. In order to solve this problem, [Chan et. al. 2003] proposed that keys, which are used for communication on links between other non-compromised nodes, should be refreshed but not through already established link. For this purpose, they use multiple disjointed paths between two nodes. If two nodes A and B share a common key *k*, and they have *h* disjointed paths between them, node A generates *h* random values and sends each one of them to B through a separate disjointed path. Then both nodes A and B compute a key *k'* using key *k* and all *h* random values. Even if a node in a path is compromised, adversary will not know *k'* and k can be refreshed through *k'*. In order to keep the chances of eavesdropping to a minimum, size of disjointed paths should be kept small. Apart from increased network communication, this scheme also increases the computation overhead of sensor nodes by requiring them to generate random values, which require a lot of energy.

Polynomial Pool-based Key Pre-distribution: In polynomial Pool-based key pre-distribution scheme [Liu et. al. 2003], a setup server generates one t-degree polynomial for each sensor node. These polynomials hold the property $f(x,y)=f(y,x)$. For example, if node *i* receives a polynomial $f(i,y)$ and node *j* receives a polynomial $f(j,y)$, they can compute a common key using identity of the other node. This scheme is scalable. However, whole network is compromised if *t* nodes are compromised. Memory requirement of this scheme is directly proportional to the value of *t*.

Grid-based Key Pre-distribution: This approach is similar to the polynomial based key pre-distribution approach. In this approach, a matrix is stored in each node's memory. If two nodes *i* and *j* want to establish a pair-wise key for communication, they must have a common row or column in the matrix. If none of the rows or columns matches, then they must find alternate path to each other in path key establishment stage. This scheme offers greater probability of key establishment as compared to the random pair-wise key establishment scheme. This scheme reduces communication and computation overhead but increases the storage overhead.

Public Key Cryptography in Wireless Sensor Networks: In previous sections, we discussed that like other traditional key management schemes, public key cryptography can not be used in wireless sensor networks due to highly sophisticated computations involved in it. Contrary to this point of view, many researchers argue that the use of public key cryptography on wireless sensor networks can not be ruled out completely especially the Elliptic Curve Cryptography (ECC), which has been used in wireless sensor networks recently [Malan et. al. 2004], [Gura et. al. 2004], [Wander et. al. 2005]. Also, public key schemes have been used on 8-bit processors [Gura et. al. 2004]. ECC can provide same level of security as that of RSA with much smaller key. According to [Gura et. al. 2004], 160-bit ECC key has the same level of security as that of 1024-bit

RSA. Also, the difference in the number of bits is not constant because 224-bit ECC has the same level of security as that of 2048-bit RSA key. ECC based public keys have been used in TinyOS [Malan et. al. 2004], an operating system developed specifically for wireless sensor networks.

Some schemes provide hybrid approach for key management i.e. they mix both symmetric and asymmetric key management approaches for providing security in wireless sensor networks [Huang er. Al. 2003], [Kotzanikolaou et. al. 2009]. LSec [Shaikh et. al. 2006] also uses hybrid approach for key management in wireless sensor networks. In the first phase, they perform authentication and authorization using symmetric keys. In the second phase, keys are distributed using random secrets. This is performed using asymmetric cryptography.

SHELL: SHELL [Ghumman et. al. 2006] is location-aware combinatorial key management scheme designed for clustered sensor networks. We will discuss SHELL and the rest of the schemes in a bit more detail because they are state-of-the-art solution of key management so far in the literature. SHELL assumes large scale sensor networks with clusters sizes of the order of hundreds of nodes. SHELL uses a small number of keys to manage large sensor networks using combinatory. SHELL employs EBS system of matrices [Eltoweissy et. al. 2004] to use small number of keys for large networks. In addition to using small number of keys for large networks, SHELL also gets rid of single point of failure by using neighbouring cluster heads for key management. SHELL targets sensor networks that are hierarchical i.e. a cluster head node manages a large number of sensor nodes and a base station manages multiple cluster head nodes. In other words, this scheme is suitable for networks, which support in-network processing of information. Also, this scheme supports overhearing of messages as one key is known to a large number of nodes.

SHELL assumes that the cluster head nodes can broadcast messages to all the sensor nodes in its cluster. Also, the cluster head node can reach all nodes in its own cluster. However, if a cluster head node wants to communicate with some node, which is not in its cluster, it has to go through the neighbouring cluster head node. In short, cluster head nodes have more communication, computation, storage and power capabilities as compared to other sensor nodes. Base station or the command node has minimal involvement in this key management protocol. Another important assumption taken in SHELL is that two compromised nodes can't come to know the location of each other i.e. they can not launch a coordinated attack. Also, two compromised nodes can't communicate through an out of band communication channel. Lastly, attacker does not know memory contents of a sensor node before deployment. EBS system of matrices is used by SHELL and another state-of-the-art key management scheme; we think it is important to discuss EBS system of matrices briefly.

Table 1 shows an example of EBS matrix. Size of an EBS matrix depends upon the number of nodes and the number of keys used to manage those nodes. In EBS matrix, number of columns is equal to the number of nodes in a cluster. Number of rows is equal to the number of keys used to manage those nodes. Total number of keys in a network is

k + *m*. Out of these *k* + *m* keys, every sensor node knows a distinct set of *k* keys i.e. set of keys known to one of the sensor nodes can not be exactly identical to the set known to some other sensor node.

	N_0	N_1	N_2	N_3	N_4	N_5	N_6	N_7	N_8	N_9
K_1	1	1	1	1	1	1	0	0	0	0
K_2	1	1	1	0	0	0	1	1	1	0
K_3	1	0	0	1	1	0	1	1	0	1
K_4	0	1	0	1	0	1	1	0	1	1
K_5	0	0	1	0	1	1	0	1	1	1

Table1. Example of an EBS matrix

If a node is compromised, set of *m* keys, which are not known to the compromised node, are used to refresh the *k* keys known to the compromised node. Suppose that in table 1, Node N_1 is compromised. Set of *k* keys known to N_1 is K_1, K_2 and K_4. If the managing node generates new values of K_1, K_2 and K_4, encrypts each one of them in K_3 and K_5 separately, and broadcasts in the cluster, all of the nodes will be able to decrypt the message except the node N_1. Number of nodes that can be supported by *k* + *m* keys can be expressed by the formula: -

$$Number\_of\_nodes = \frac{(k+m)!}{k!m!}$$

It is very important to note that the number of nodes supported by k + m keys grows exponentially with the values of k and m. Values of k and m can be adjusted according to the network and its security requirements. Higher value of m results in higher security but with increased overhead.

Initially, each sensor node is pre-loaded with a discovery key Ksg and two other keys KSCH and KSKey. Ksg is recomputed with one-way hashing function, such as SHA1 [13] or MD5 [14], stored in the node. The one-way hashing function is only known to the sensor node and the command node. Ksg is used to recover the network if the cluster head node is compromised. KSCH and KSKey are used for initial key distribution. In a cluster head node, key Kgc, which is used for communication between the cluster head node and the command node, is pre-loaded along with key Ksg of all nodes that lie in its cluster. Gateways can also communicate between themselves using another type of key provided by the command node. Command node generates the key, used for communication between the cluster head node, and renews them at regular intervals.

In SHELL, cluster head node is responsible for the formation of EBS matrix and generation of communication keys of its own cluster. Also, it is responsible for refreshment of its cluster's data keys. On request, the cluster head nodes generates administrative key of other clusters. In addition to that, the cluster head node is responsible for detecting and evicting compromised sensor nodes in its cluster. Every node is authenticated by the command node right after the initial deployment. After the

initial deployment, gateways form their EBS matrices first. Each EBS matrix, along with the list of sensors in that cluster, is shared between the gateways and the command node. For each cluster, more than one neighbouring cluster head nodes are designated by the command node for managing the administrative keys. For example, if there are *12* keys used in a cluster, command node can designate *4* neighbouring cluster head nodes to manage *3* keys each for that cluster. The cluster head node shares the relevant portions of EBS matrix with each of the neighbouring cluster head node.

Command node shares the key KS_{CH} of each sensor node with its cluster head. It also shares key KS_{Key} of every sensor node with the relevant neighbouring cluster head nodes i.e. the one's responsible for managing any of the administrative keys that will be known to the sensor node. For key distribution, each relevant neighbouring gateway generates one message per individual administrative key in its cluster for each sensor node. The message is first encrypted with the KS_{Key} of the node and then the administrative key of the sensor node's gateway. Gateway decrypts the message, encrypts it with KSCH of the node and sends it to the sensor node. In order to share communication keys, cluster head nodes generate them and send them to their neighbouring cluster head nodes. Neighbouring clusters then send them to sensor nodes in the same way as they send the administrative keys.

If a cluster head node is compromised, either a new cluster head node is deployed or its sensors are redistributed among other cluster head nodes. The new gateway makes a new EBS matrix and repeats the process of initial deployment and initial key distribution. If a sensor node is compromised, keys known to the compromised node are changed with the method mentioned above in the description of EBS matrices. Advantages of SHELL are that it is highly scalable and resilient against node capture attacks. Also, it has very effective node authentication mechanisms. However, it is susceptible to collusion attacks. Collusion attack takes place when two or more compromised nodes collaborate with each other to attack a sensor network. In the same paper, they have also proposed mechanisms to prevent compromised nodes from collusion by assigning the keys strategically.

MUQAMI+: MUQAMI+ [Raazi et. al. 2009], which is an extended version of MUQAMI [Raazi et. al. 2007], is also an EBS based key management scheme for clustered sensor networks. Just like SHELL, MUQAMI+ is highly scalable, resilient against node capture and has effective node authentication mechanisms. Apart from that, it does not have single point of failure in a cluster or in a network. However, its mechanism of avoiding single point of failure is different and more efficient than that of SHELL scheme. Instead of relying on the neighbouring cluster head nodes for key management, responsibility of key management is distributed among few key generating nodes within a cluster. This reduces communication, computation, storage overhead and energy consumption of the sensor nodes in the network. A big advantage of this scheme is that it is very flexible i.e. it allows the responsibility of being cluster head node and being a key generating node to be shifted seamlessly from one node to another. Therefore, if this scheme is employed in a sensor network, responsibilities can be transferred among nodes according to their capabilities and energy levels.

In MUQAMI+, first the cluster head nodes are deployed. After that, sensor nodes are deployed, which report to the nearest cluster head nodes with the help of a pre-loaded key. Cluster head node authenticates every sensor node from the command node before adding it to the cluster. Then the command node sends initial values of administrative keys to all sensor nodes through the cluster head node in such a way that the cluster head node does not get to know the key values. If the cluster head node comes to know about the administrative keys, it will become single point of failure for the cluster. However, the cluster head node does get to know about the key identities known to a particular node and key generating responsibilities assigned to a node. The cluster head node builds EBS matrix based to this information. If a key needs to be refreshed, the cluster head node sends a message to the key generating node and the key generating node refreshes the key.

If a sensor node is compromised, the cluster head asks the key generating nodes, which generate the *k* keys known to the compromised node, to send new values of administrative keys encrypted in the old one, to the cluster head node using pair-wise keys between key generating node and the cluster head node. Cluster head node forwards these k values to the other *m* key generating nodes, which broadcast them after encrypting in their administrative key values. In case a cluster head or a key generating node is compromised, its responsibility is shifted to some other node in the cluster.

One problem with this scheme is that some sensor nodes need to generate keys, which increases computation overhead substantially. Authors of MUQAMI+ propose to solve this issue using one-way hashing functions. In one-way hashing functions, we can't compute the previous value using the current value. In MUQAMI+, one-way hashing functions are stored in sensor nodes. The command node sends initial and final values to the key generating nodes. The key-generating node computes all intermediate values using the first value. Last value is used to confirm the end of a key chain. After this, the key generating node stores the key chain in its memory and uses it in reverse manner i.e. last value first. In this way, adversary can not compute next value of a key even if it knows the current value. However, this solution incurs storage overhead and a small communication overhead. Effectively, we can say that there is a trade off between computation overhead and storage and communication overhead.

LEAP+: LEAP+ [Zhu et. al. 2006] is also a state-of-the-art solution for key management in wireless sensor networks. Its initial version was proposed as LEAP [Zhu et. al. 2003]. Later on, it was proposed as LEAP+ with some extensions. Although it can be used in both homogeneous and heterogeneous (clustered) sensor networks, it is more suitable for homogeneous sensor networks. This scheme is highly scalable and resistant to collusion attacks. Also, compromised node revocation is very simple and sensor node compromise does not affect other parts of the network.

In LEAP+, every sensor node uses a pseudo-random function to compute keys. Pseudo-random function uses node identities and some pre-loaded key values to compute keys. When sensor nodes are deployed, they compute their individual keys, which they share with only the command node. After that, they exchange their identities with their neighbouring nodes and compute pair-wise keys with their neighbours. In order to

broadcast some message, they need a key that is known to all the neighbouring nodes. They compute this key for broadcast purposes and send it to all the neighbouring nodes individually. Lastly, a global key, which is managed by the command node, is used for broadcast in the whole network.

If a sensor node is compromised, all of its neighbouring nodes delete pair-wise keys shared with the compromised node. After that, every neighbouring node computes new value of its key, which is used for broadcast purposes, and sends it to rest of the neighbours individually. Global key is also refreshed in the end. Apart from the increased computation overhead of LEAP+, another drawback of this scheme is that it assumes the network is safe during some initial time period. LEAP+ also has effective mechanisms for authenticated broadcast.

FUTURE DIRECTIONS OF RESEARCH

Until now, research related to key management in wireless sensor networks has been very generic. All the key management schemes, proposed so far, are designed from broad perspective i.e. they are either designed for clustered sensor networks or homogeneous sensor networks. With the passage of time, applications and usage of wireless sensor networks have increased. Still, wireless sensor networks are being employed in newer application areas. For example, wireless sensor networks have been applied to health care scenarios providing ubiquitous health care for patients. Another new idea is to attach sensors to the devices worn by patients, so that their health can be monitored in real time. Temperature, blood pressure sensors can be embedded in wrist watches, necklaces etc. Apart from that, wireless sensor networks have been applied to houses and apartments resulting in smart homes.

With the introduction of sensor networks in new application areas, the characteristics of sensor networks will change according to application scenarios. For example, a small body area network need not be scalable. Also, if it is under constant human observation, adversary can not tamper a sensor node (If a conscious patient is wearing a sensor node, it is under constant human observation). A sensor node in a smart home can have constant source of power and need not use battery. In short, requirements and characteristics of different application areas, employing wireless sensor networks, will differ and would require separate mechanisms for key management. Therefore, future directions of research is that need for those key management schemes will arise that should be specific to different application areas e.g. health care applications and smart home applications.

CONCLUSIONS

We have discussed the security requirements of key management and possible security threats in wireless sensor networks in detail. Also, we have discussed the constraints of sensor nodes and wireless sensor networks. In addition to that, various key management schemes, which were designed according to the requirements and characteristics of wireless sensor networks, were discussed. We learn that we do not have absolute criteria

to rate a key management scheme better than all other schemes. There are different types of wireless sensor networks and different application areas, in which they are used. One key management scheme can be more efficient in one application area or one topology while another scheme can be more efficient in some other application area or some other topology. In future, the number of application areas is expected to increase.

REFERENCES

Chan, H., Perrig, A., and Song, D. 2003. Random Key Predistribution Schemes for Sensor Networks. In Proceedings of the 2003 IEEE Symposium on Security and Privacy (May 11 - 14, 2003). SP. IEEE Computer Society, Washington, DC, 197-213.

Dierks, T., Allen, C. 1999.: The tls protocol version 1.0.

Diffie, W., Hellman, M. 1976. New direction in cryptography. IEEE Trans. on Info. Theory, Vol. IT-22, Nov. 1976, pp. 644-654 (Invited Paper).

Eltoweissy, M., Heydari, H., Morales, L., and Sadborough, H. 2004. Combinatorial Optimization of Group Key Management, J. Network and Systems Management 12(1), 33-50.

Eschenauer, L. and Gligor, V. D. 2002. A key-management scheme for distributed sensor networks. In Proceedings of the 9th ACM Conference on Computer and Communications Security (Washington, DC, USA, November 18 - 22, 2002). V. Atluri, Ed. CCS '02. ACM, New York, NY, 41-47. DOI= http://doi.acm.org/10.1145/586110.586117.

Ghumman, K. 2006. Location-Aware Combinatorial Key Management Scheme for Clustered Sensor Networks. *IEEE Trans. Parallel Distrib. Syst.* 17, 8 (Aug. 2006), 865-882. DOI= http://dx.doi.org/10.1109/TPDS.2006.106

Gupta, G., Younis, M. 2003. Load-balanced clustering of wireless sensor networks. IEEE International Conference on Communications, 2003. ICC '03., vol.3, no., pp. 1848-1852 vol.3, 11-15 May 2003.
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1203919&isnumber=27115.

Gura, N., Patel, A., Wander, A., Eberle, H., Shantz, S.C. 2004 “Comparing elliptic curve cryptography and rsa on 8-bit cpus,” 2004, pp. 119–132. http://www.springerlink.com/content/87aejjlhqn6fuxpy =0pt.

Huang, Q., Cukier, J., Kobayashi, H., Liu, B., and Zhang, J. 2003. Fast authenticated key establishment protocols for self-organizing sensor networks. In Proceedings of the 2nd ACM international Conference on Wireless Sensor Networks and Applications (San Diego, CA, USA, September 19 - 19, 2003). WSNA '03. ACM, New York, NY, 141-150. DOI= http://doi.acm.org/10.1145/941350.941371.

Intanagonwiwat, C., Govindan, R., and Estrin, D. 2000. Directed diffusion: a scalable and robust communication paradigm for sensor networks. In Proceedings of the 6th Annual international Conference on Mobile Computing and Networking (Boston, Massachusetts, United States, August 06 - 11, 2000). MobiCom '00. ACM, New York, NY, 56-67. DOI= http://doi.acm.org/10.1145/345910.345920.

Karlof, C.; Wagner, D. 2003. Secure routing in wireless sensor networks: attacks and countermeasures. Sensor Network Protocols and Applications, 2003. Proceedings of the First IEEE. 2003 IEEE International Workshop on , vol., no., pp. 113-127, 11 May 2003 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1203362&isnumber=27104.

Karlof, C., Li, Y., Polastre, J. 2003. Arrive: An architecture for robust routing in volatile environments. Tech. Rep. CSD-03-1233, University of California at Berkeley (2003).

Karlof, C., Sastry, N., and Wagner, D. 2004. TinySec: a link layer security architecture for wireless sensor networks. In Proceedings of the 2nd international Conference on Embedded Networked Sensor Systems (Baltimore, MD, USA, November 03 - 05, 2004). SenSys '04. ACM, New York, NY, 162-175. DOI=http://doi.acm.org/10.1145/1031495.1031515.

Kohl, J., Neuman, C. 1993.: The kerberos network authentication service (v5).

Kotzanikolaou, P., Magkos, E., Vergados, D., and Stefanidakis, M. 2009. Secure and practical key establishment for distributed sensor networks. In Security and Communication Networks, Wiley InterScience, 2009. DOI=http://dx.doi.org/10.1002/sec.102.

Liu, D. and Ning, P. 2003. Establishing pairwise keys in distributed sensor networks. In Proceedings of the 10th ACM Conference on Computer and Communications Security (Washington D.C., USA, October 27 - 30, 2003). CCS '03. ACM, New York, NY, 52-61. DOI= http://doi.acm.org/10.1145/948109.948119.

Madden, S., Szewczyk, R., Franklin, M. J., and Culler, D. 2002. Supporting Aggregate Queries Over Ad-Hoc Wireless Sensor Networks. In Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications (June 20 - 21, 2002). WMCSA. IEEE Computer Society, Washington, DC, 49.

Malan, D.J., Welsh, M., Smith, M.D. 2004. A public-key infrastructure for key distribution in TinyOS based on elliptic curve cryptography. Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on , vol., no., pp. 71-80, 4-7 Oct. 2004 URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1381904&isnumber=30129.

Raazi, S.M.K., Khan, A.M., Khan, F.I., Lee, S., Song, Y., Lee, Y. 2007. MUQAMI: A Locally Distributed Key Management Scheme for Clustered Sensor Networks. In Trust

Management, vol. Volume 238/2007 of IFIP International Federation for Information Processing, pp. 333 - 348, Springer Boston. DOI = 10.1007/978-0-387-73655-6_22.

Raazi, S.M.K., Lee, H., Lee, S., Lee, Y. 2009. MUQAMI+: a scalable and locally distributed key management scheme for clustered sensor networks. Annals of Telecommunications. DOI = 10.1007/s12243-009-0123-0.

Rivest, R. L., Shamir, A., and Adleman, L. 1978. A method for obtaining digital signatures and public-key cryptosystems. Commun. ACM 21, 2 (Feb. 1978), 120-126. DOI= http://doi.acm.org/10.1145/359340.359342.

Shaikh, R. A., Lee, S., Khan, M. A. U., & Song, Y. J. 2006. LSec: Lightweight security protocol for distributed wireless sensor network. In 11th IFIP Int. Conf. on Personal Wireless Comm., LNCS 4217, (pp. 367–377), Albacete, Spain: Springer-Verlag.

Tanya Roosta, Shiuhpyng Winston Shieh, S. Shankar Sastry. 2006. Taxonomy of Security Attacks in Sensor Networks and Countermeasures. The First IEEE International Conference on System Integration and Reliability Improvements, December, 2006.

Wander, A. S., Gura, N., Eberle, H., Gupta, V., and Shantz, S. C. 2005. Energy Analysis of Public-Key Cryptography for Wireless Sensor Networks. In Proceedings of the Third IEEE international Conference on Pervasive Computing and Communications (March 08 - 12, 2005). PERCOM. IEEE Computer Society, Washington, DC, 324-328. DOI= http://dx.doi.org/10.1109/PERCOM.2005.18.

Xiao Y., Rayi V. K., Sun B., Du X., Hu F. and Galloway, M. 2007. A survey of key management schemes in wireless sensor networks. Comput. Commun. 30, 11-12 (Sep. 2007), 2314-2341. DOI= http://dx.doi.org/10.1016/j.comcom.2007.04.009.

Younis, O. and Fahmy, S. 2004. Heed: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. IEEE Transactions on Mobile Computing 3(4), 366-379. DOI= http://dx.doi.org/10.1109/TMC.2004.41. Student Member-Ossama Younis and Member- Sonia Fahmy.

Zhu, S., Setia, S., and Jajodia, S. 2003. LEAP: efficient security mechanisms for large-scale distributed sensor networks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security* (Washington D.C., USA, October 27 - 30, 2003). CCS '03. ACM, New York, NY, 62-72. DOI= http://doi.acm.org/10.1145/948109.948120.

Zhu, S., Setia, S., and Jajodia, S. 2006. LEAP+: Efficient security mechanisms for large-scale distributed sensor networks. ACM Trans. Sen. Netw. 2, 4 (Nov. 2006), 500-528. DOI= http://doi.acm.org/10.1145/1218556.1218559.

Zia, T. and Zomaya, A. 2006. Security Issues in Wireless Sensor Networks. In Proceedings of the international Conference on Systems and Networks Communication (October 29 - November 03, 2006). ICSNC. IEEE Computer Society, Washington, DC, 40. DOI= http://dx.doi.org/10.1109/ICSNC.2006.66.

KEYWORDS

Key Management, Sensor Networks, Security Threats, Node Compromise, Node Revocation, Communication Overhead, Computation Overhead, Storage Overhead, State-of-the-art key management, Symmetric Key Cryptography.

TERMINOLOGIES

Adversary: Any party, person or a device, which tries to reveal secret information to unauthorized users or tries to disrupt the network operation or causes harm to the network in any other way.

Cryptography: Mechanisms, which are used to hide secret information from unauthorized users. Keys are used to encrypt (conceal) and decrypt (reveal) the secret information.

Symmetric Key Cryptography: Cryptography mechanisms, in which same key is used for both encryption and decryption.

Asymmetric Key Cryptography: Cryptography mechanisms, in which different keys are used for encryption and decryption.

Cryptanalytic attacks: Cryptanalytic attacks are attempts made by an adversary to crack cryptographic information such as secret keys shared between communicating parties.

Denial-of-Service: Denial-of-Service, abbreviated as DoS, is a type of attack on wireless sensor network. In Dos attack, the adversary tries to disrupt the normal network operation by not allowing the sensor nodes to properly communicate with each other.

Single Point of Failure: This is a term used to express a scenario, in which an attack at a single place can bring down the whole system or a unit. In case of wireless sensor networks, single point of failure can be a node, whose compromise can compromise the whole network or a whole cluster.

QUESTIONS / ANSWERES

Question 1: Why is it thought that it is not viable to use asymmetric or public key cryptography in wireless sensor networks?

Answer: Normally, processors are required to perform complex mathematical computations if asymmetric or public key cryptography is used. Wireless Sensor Networks are resource constrained devices having limited battery power, limited processing capability and limited memory capacity. If we use public key cryptography in wireless sensor networks, it uses up a lot of memory and takes a lot of time on sensor nodes to execute. In trying to perform tough mathematical calculations, sensor nodes lose

their battery power very quickly. Therefore, it is thought that public key cryptography is not viable for wireless sensor networks.

Question 2: Can RSA and Diffie-Hellman algorithms perform authentication? Why?

Answer: RSA and Diffie Hellman algorithms are not capable of authenticating the other party. The reason is that both algorithms were designed to facilitate communication between two parties, which do not know each other in advance. In Diffie-Hellman key exchange, a node agrees on a secure secret key with an unknown party through an insecure medium. In RSA, public key of a node is published. Anyone can send messages to the node using its public key. The node uses its private key to decrypt the message.

Question 3: What is the concept of a trusted third party?

Answer: If two parties do not know each other in advance and they want to communicate with each other, then there should be a mechanism, using which they can authenticate each other. Concept of trusted third party is all parties trust a central server and register themselves on it. Whenever two parties try to establish a connection, they authenticate each other from the trusted central server, which is the third party.

Question 4: What are the main goals of a key management scheme?

Answer: Main goal of a key management scheme is to maintain confidentiality of secret information and help in authenticating legitimate parties or nodes in a network. Apart from that, a key management scheme for wireless sensor networks should also be able to deal with the issue of node compromise.

Question 5: Why is it important to refresh secret keys after some time interval? How do we determine the time interval?

Answer: Adversaries always try to crack or guess the keys used to conceal important information by launching cryptanalytic attacks. If same key is used to conceal secret information for a long time period, an adversary may become successful in guessing the secret key. Therefore, it is important to refresh secret keys at regular time intervals. These time intervals depend upon the frequency of information exchange and the time required by an adversary to find success in cracking a secret key.

Question 6: What is the difference between an outsider attack and an insider attack? Elaborate with examples.

Answer: In outsider attack, the adversary is not part of the network. On the other hand, adversary is part of the network in case of an insider attack. Example of an outsider attack is a malicious sensor node placed within the sensor network. The malicious node manages to get it authorized and listens to secret information or injects false information in the network. Example of an insider attack is that a legitimate sensor node from the network is compromised through software or through node tampering and then the

compromised node listens to secret information or injects false information in the network.

Question 7: What are the two most important requirements of a key management scheme designed for wireless sensor networks? Why?

Answer: The two most important requirements of a key management scheme designed for wireless sensor networks are: 1) It should have highly effective sensor node authentication mechanisms; 2) It should also have effective mechanisms to deal with sensor node compromise. These are the two main requirements because most of the attacks on security of wireless sensor networks involve either an unauthorized outsider node or a compromised insider node.

Question 8: What attacks on wireless sensor networks can not be handled by a key management scheme? Give an example.

Answer: Attacks, which are carried out on physical layer of wireless sensor networks, can not be handled by a key management scheme. Example of such an attack is jamming attack. This is a type of Denial-of-Service attack that is launched by disrupting radio communication, through which sensor nodes communicate with each other.

Question 9: Why do we require other key management schemes when we have pair-wise key distribution scheme, which has all security features required a wireless sensor network?

Answer: Although pair-wise key distribution scheme provides high level of security, it is impractical to use it in wireless sensor networks because it is not scalable and its storage overhead is too high. Also, it is not required that every sensor node in a wireless sensor network shares a key with every other sensor node in the network.

Question 10: How does an EBS based key management scheme manage a large number of sensor nodes using a small number of keys?

Answer: When using EBS matrix for key management, each sensor node must have a distinct key combination stored in it i.e. no other node knows the same set of keys. EBS matrix has a property that as the number of keys grow linearly, the number of available distinct key combinations grow exponentially. This allows the management of large number of sensor nodes with a small number of keys.

Question 11: How does SHELL avoid a single point of failure in a cluster of wireless sensor networks?

Answer: SHELL scheme avoids single point of failure in a wireless sensor network by allocating the responsibility of key management to cluster head nodes of the neighbouring clusters while cluster head node of the subject cluster does not get to know those keys. When this technique is used for key management, there is no single node in

the network whose compromise can result in the compromise of the whole cluster or network.

Question 12: How does MUQAMI+ avoid single point of failure after bringing key management responsibility within the subject cluster? Doesn't it add to the burden to sensor nodes?

Answer: When the responsibility of key management is brought within the subject cluster, it is divided among a few key generating nodes, which manage one key each. Also, cluster head node of the subject cluster does not get to know the keys managed by key generating nodes. This way, compromise of a single sensor node can not result in the compromise of entire cluster or entire network. It does not add significant burden to the sensor nodes because a very small number of nodes are required to manage keys. Also, MUQAMI+ allows responsibility of key management to be shifted from one node to another seamlessly.

Question 13: Why don't we consider the command node a single point of failure?

Answer: In a wireless sensor networks, command node in the node which receives all the data from the network. Normally, it is not a sensor node. Rather, it is a computer or a laptop class device, which has more capabilities than a simple sensor node. In some application scenarios of wireless sensor networks, there can be more than one base station collecting data from the network.