CHAPTER 2

BASIC CONCEPTS OF RESEARCH

2.1 INTRODUCTION

Advantages in communication technology allow us to build networks where large numbers of low-power and inexpensive sensor devices are integrated in the physical environment, operating together over a wireless media. The sensor nodes have special functions to sense and collect information about the environment under monitoring: temperature, humidity, illumination, noise and so on. Application domains of Wireless Sensor Networks are widely developing in various areas. In the near future, everyday objects that surround us will become a part of the Internet that can generate and consume information. Security in WSNs is a challenging task due to inherent limitations of sensors.

All the functionality of the sensor network is provided thanks to the individual capabilities of the sensor nodes. Each sensor node with limited computational capability has several built-in sensors and communicates wirelessly. Therefore, they have the capacity to collect and process the raw information about their surroundings and communicate with neighbours. Nodes are also small in size, and can unobtrusively provide the physical information of any entity. Moreover, nodes are usually battery powered enabling them to act independently and operate autonomously if required. Base station is a powerful and capable device that serves as an interface between the sensor nodes and the user. A well designed sensor network is
built with long term goals in mind. Often a limited opportunity exists to deploy any sort of network and the initial setup must be maintained throughout measurements. For example, a network deployed on the seafloor by a research vessel cannot be easily modified, yet may be expected to collect data from environments over-saturating the test bed with sensors. A WSN must be a self organizing structure, so as the topology of the network changes, connections remain wherever possible. As motes begin to fail, others are expected to step up and fill in. Similarly, some devices may be programmed to wake up late in the life of a network in order to extend its life. An ideal implementation might take into account battery power and expected lifetime of each node to maximize dependability.

A set of new manifold options is arriving to the WSN based on IEEE 802.15.4 LLNs (Low power and Lossy Networks) with connectivity to the IP world, with the solutions defined by the Internet Engineering Task Force (IETF), 6LoWPAN. Similar to WSN, 6LoWPAN’s sensor nodes are also resource constrained with small data rate, low bandwidth and low transmit power. The major issues among them are power consumption and network lifetime extension. Secure routing is one of the key challenges for 6LoWPAN WSN and the challenge becomes more difficult when the network size is growing and the hardware resources are restricted. Also this standard is considered as a suitable candidate to introduce the concept of “Internet of Things” (IoT) to the real world.

These challenges necessitate energy awareness at all layers of networking protocol stack. The physical and link layer issues are generally common for any sensor application, therefore the research concentrates more on system-level power awareness such as dynamic voltage scaling, radio communication hardware, low duty cycle issues, system partitioning, energy-aware MAC protocols. At the network layer, the main aim is to find
ways for energy-efficient route setup and reliable relaying of data from the sensor nodes to the sink so that the lifetime of the network is maximized. In this scenario, any protocol, architecture or application which is not developed with security in mind is hardly useful. In this study, a survey of the “state-of-the-art” security issues and algorithms that a designer must have in mind while working with current scenario of IoT and WSN is presented.

The contribution of this chapter is arranged as follows:

1. Basic concepts and limitations of Wireless Sensor Networks are discussed.
2. Applications of WSN.
3. Security Goals, threat models and the attacks on the Sensor Networks are discussed.
4. Various existing Routing Protocols are classified and analyzed with the advantages and disadvantages of them.
5. Concepts of Key management and its classification
6. Concepts of LLN, 6LoWPAN and Contiki operating system
7. Concepts of Cloud computing

2.2 WIRELESS SENSOR NETWORKS

2.2.1 Overview

A Wireless Sensor Network is constructed by a group of sensor nodes with characteristics of tiny shape, low energy consumption and cheap cost. A Sensor Node typically contains a wireless transmitter/receiver, and power, sensing, processing, and storage units. Sensors can be used to sense
temperature, air pressure, salinity, moisture, movement, biometric or any other phenomena and forward the sensed data to a nearby node. Sensor nodes are either controlled or uncontrolled. To establish a Wireless communication infrastructure for a known environment with control, deployment may be achieved manually. However, manual deployments become infeasible or even impossible as the number of the nodes increases. If the environment is uncontrolled or the WSN is very large, deployment has to be performed by randomly scattering the sensor nodes to the target area. It may be possible to provide denser sensor deployment at certain spots, but exact positions of the sensor nodes cannot be controlled. Thus, the network topology cannot be known precisely prior to deployment. The network self configures by itself to form the network topology of the moment. Sensor Networks find applications in various areas such as military and surveillance, environmental, health, space exploration, vehicular movement, etc. Figure 2.1 shows a sample wireless sensor network with surveillance Video cameras, Infra-red cameras and Microphones used in a military application in defense.

Figure 2.1 Sample Wireless Sensor Network
2.2.2 Limitations in Sensor Networks

The following section lists the inherent limitations in the sensor networks which make the design of the security procedures complicated. Routing in sensor networks is very challenging due to these characteristics:

1. Sensor nodes are tightly constrained in terms of power, on-board energy, storage and processing capacity and thus require careful resource management.

2. Lack of infrastructure leads to insecure wireless communication.

3. Deployment nature in public and hostile environments makes them highly vulnerable to get captured and vandalised. Physically the security of sensor nodes with tamper proof material results in increased cost of a node.

4. Classical IP-based protocols are not suitable due to the lack of global addressing scheme.

5. Compared to communication networks, most applications demand MP2P traffic from multiple regions (sources) to a base station (sink).

6. Optimization of energy and bandwidth utilization has to be exploited by routing protocols to reduce the redundancy of data in the vicinity of sink node.

7. Sensor nodes can easily be compromised.

8. Heterogeneous nature of the sensor nodes used for deployment is an additional limitation.

9. The topology of a WSN changes very frequently due to node failure, joining or mobility.
10. Dense deployment of sensor nodes in several orders of magnitude higher than that in an ad-hoc network.

Due to such differences, many new algorithms have been proposed for the problem of routing data in sensor networks. The routing mechanisms consider the characteristics of sensor nodes regarding path selection, network architecture and protocol operation apart from its application.

The cryptographic techniques devised for the traditional wired networks are not feasible to Wireless Sensor Networks. Encryption leads to extra processing, memory and battery power which are the prime resources for the sensors’ longevity. The security mechanisms could also increase delay, jitter and packet loss in wireless sensor networks. As minimal (or no) human interaction for the sensors is a fundamental feature of WSN, time to time modification of the key for encryption is another issue. The methods, by which the keys are managed, revoked and assigned to a new sensor that is added to the network or renewed for ensuring robust security for the network must be taken under consideration. Adoption of pre-loaded keys or embedded keys could not be an efficient solution.

2.2.3 Applications of Wireless Sensor Networks

There are many areas of the application that can take advantage of all these benefits. The applications can be classified into the following categories:

1. Monitoring space: The sensor network simply monitors the physical features of a certain environment. This category includes applications such as environmental and habitat monitoring, precision agriculture, indoor climate control, surveillance, treaty verification, and intelligent alarms.
2. Monitoring things: The sensor network controls the status of a physical entity. This category includes applications such as structural monitoring, eco-physiology, condition-based equipment maintenance, medical diagnostics, and urban terrain mapping.

3. Monitoring interactions: The sensor network monitors the interactions of things (both inanimate and animate) with each other and the encompassing space. This category includes applications such as wildlife habitats, disaster management, critical (information) infrastructure systems, emergency response, asset tracking, healthcare, and manufacturing process flow.

While all the application areas presented in the previous classification are mere ideas of where WSN could be applied, the research community has already proven the usefulness of WSN in real-world settings.

2.3 SECURITY GOALS

Some of the security goals in sensor networks are:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Confidentiality</td>
<td>Confidentiality is the ability of hiding the message from an unauthorized attacker. It means that if an illegal and unauthorized adversary access to the message, it cannot understand it.</td>
</tr>
<tr>
<td>Authentication</td>
<td>Authentication is ability to identify the reliability of the message origin; whether a legitimate node has sent the information.</td>
</tr>
<tr>
<td>Integrity</td>
<td>This provides a mechanism in order to know whether the message had been tampered or not.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Freshness and availability</td>
<td>Availability guarantees that the network services are on hand as needed. This factor identifies whether the message can move on to the network or not. If the node can use its resource, then the availability is provided to the network for forwarding the message.</td>
</tr>
<tr>
<td>Graceful degradation</td>
<td>It is the ability of the network performance to degrade gracefully when a small portion of nodes are compromised i.e. the designed mechanism must be resilient to node compromise.</td>
</tr>
<tr>
<td>Non-repudiation</td>
<td>Sender of the message shall not be able to deny later about the sending of the message and that the recipients shall not be able to deny the receipt after receiving the message.</td>
</tr>
<tr>
<td>Resiliency</td>
<td>It is the ability of the network to tolerate the attacks and continue offering its services uninterruptedly.</td>
</tr>
<tr>
<td>Self-healing</td>
<td>It is the ability of the network to recover from security problems and even isolate the source of threat so that it stops jeopardizing the availability of the network in future communications.</td>
</tr>
</tbody>
</table>

2.4 THREAT MODELS

Any WSN environment is prone to threats and attacks from adversaries. The major classification of the threat models due to adversary nodes can be modelled as listed in Figure 2.2.
Figure 2.2 Classification of Threat models

- Mote-class attacker vs. Laptop-class attacker: A mote-class attacker has similar capabilities as other motes and hence can access to only a few motes in the network whereas a laptop-class attacker has access to devices with more computational resources resulting in serious attacks.

- Inside attacker vs. an outside attacker: An outside attacker may be a passive listener who has no special access to the sensor network, but an inside attacker has access to the encryption keys or other codes used by the network. For example, an insider attacker could be a compromised node which was originally a legitimate part of the sensor network.

- Passive vs. active attacker: A passive attacker is only interested in collecting sensitive data from the sensor network, leading to compromising of the privacy and confidentiality requirements. In contrast, the objective of an active attacker is to disrupt the total functionality of the network and thereby degrade its performance.
2.5 ATTACKS ON SENSOR NETWORKS

Most of the routing protocols proposed for ad hoc and sensor networks are not designed to handle security related issues. Therefore there is a lot of scope for attacks on them. Different possible attacks on the flow of data and control information can be categorized as follows: Passive information gathering and message corruption, Node Compromise, Node tampering, False node, Node outage, Traffic analysis, Acknowledgement Spoofing attack, Selective forwarding attack, Flooding attack, Sinkhole attack, Sybil attack, Black hole attack, Wormhole attack, DOS attack, Replay attack and many other attacks.

2.5.1 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 the 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, the adversary modifies the contents of a message before it gets to the receiver.

2.5.2 Node Compromise

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

2.5.4 False Node

Malicious node is 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 the shortest route to the base station so that other nodes route their packets through it in order to save energy.

2.5.5 Node Outage

An 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.

2.5.6 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.
2.5.7 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.

2.5.8 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 the working of the target sensor network.

2.5.9 Sinkhole Attacks

In this case, an 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.

2.5.10 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 the effectiveness of fault tolerant schemes like distributed storage. Also, sybil attacks can affect routing algorithms.

2.5.11 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 the malicious node’s neighbours, who might think that the other malicious node, near to the base station, is their neighbour.

2.5.12 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.

2.5.13 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 already has been discussed, is a classical example of a DoS attack.

### 2.6 CLASSIFICATION OF ROUTING PROTOCOLS

The routing protocols designed for WSN can be classified based on path selection, as proactive, reactive and hybrid as shown in Figure 2.3. Based on the network architecture, they can be further classified as flat (data-centric, flat), hierarchical, such as LEACH, TEEN and APTEEN, and geographical information based, such as GAF and GEAR. There are also classifications based on protocol operation, such as multipath-based, query-based, event-driven, negotiation-based and coherent-based.

![Figure 2.3 Classification of Routing Protocols](image)

Routing protocols for wireless networks can be classified into three types based on the underlying routing information update mechanism employed. A routing protocol could be reactive (on demand), proactive (table driven) or hybrid.
2.6.1 Proactive Routing Protocols

In Proactive or Table driven routing protocols, such as DSDV or OLSR, every node maintains the network topology information in the form of routing tables by periodically exchanging routing information. Routing information is generally flooded in the whole network. Whenever a node requires a path to a destination, it runs an appropriate path finding algorithm on the topology information it maintains. Other proactive routing protocols are SEAD, CGSR, STAR, LORA, WRP, GSR, FSR etc.

2.6.2 Reactive Routing Protocols

Reactive or On-demand Routing Protocols obtain the necessary path when it is required, by using a connection establishment process. They do not maintain the network topology information and do not exchange routing information periodically. Reactive routing protocols often outperform proactive ones due to their ability to adjust the amount of network overhead created to track the mobility in the network. In Reactive routing protocols AODV, DSR, OLSR and TORA are considered to be the most popular routing protocols as many secure versions have been derived from their basic implementation. The analysis of the secure versions of these protocols such as SAODV, SAOMDV, SAR, SQoS, Ariadne, CONFIDANT, ACQUIRE, SPREAD has been investigated.

2.6.3 Hybrid Routing Protocols

Hybrid Routing Protocols utilize the advantages of both the Proactive and Reactive routing protocols. Hybrid Routing Protocols such as ZRP and SLSP combine the best features of both reactive and proactive routing protocols. For example, a node communicates with its neighbours using a proactive routing protocol, and uses a reactive protocol to
communicate with nodes farther away. In other words, for each node, nodes within certain geographical area reached using proactive routing protocols. Outside the geographical area, reactive routing protocols will be used.

2.6.4 Network Architecture Based Protocols

Routing protocols for wireless networks can also be classified into three types based on the Network Architecture with which they are built. The above mentioned routing protocol can be data centric (flat), Hierarchical (cluster based) or location based.

2.6.5 Data-Centric Routing Protocols

In data-centric routing, the sink sends queries to certain regions and waits for data from the sensors located in the selected regions. Since data is being requested through queries, attribute-based naming is necessary to specify the properties of data. SPIN is the first data-centric protocol, which considers data negotiation between nodes in order to eliminate redundant data and save energy. Later, Directed Diffusion was developed and has become a breakthrough in data-centric routing. Then, many other protocols have been proposed either based on Directed Diffusion or following a similar concept. All these protocols are developed with routing in mind and does not concentrate on secured routing.

2.6.6 Hierarchical Routing Protocols

Similar to other communication networks, scalability is one of the major design attributes of sensor networks. A single-tier network can cause the gateway to overload with the increase in sensors density. Such overload might cause latency in communication and inadequate tracking of events. In addition, the single-gateway architecture is not scalable for a larger set of sensors covering a wider area of interest since the sensors are typically not
capable of long-haul communication. To allow the system to cope with additional load and be able to cover a large area of interest without degrading the service, networking clustering has been pursued in some routing approaches.

The main aim of hierarchical routing is to efficiently maintain the energy consumption of sensor nodes by involving them in multi-hop communication within a particular cluster and performing data aggregation and fusion in order to decrease the number of transmitted messages to the sink. Cluster formation is typically based on the energy reserve of the sensors and the sensor's proximity to the cluster head. LEACH is one of the first hierarchical routing approaches for sensors networks. The idea proposed in LEACH has been an inspiration for many hierarchical routing protocols such as TEEN APTEEN, PEGASIS, Hierarchical PEGASIS, although some protocols such as self configurable systems, energy aware routing in cluster based sensor networks have been independently developed. Necessary exploration about hierarchical routing protocols has been done. All these protocols do not concentrate towards secure data communication.

Cluster-based routing protocols group sensor nodes to efficiently relay the sensed data to the sink. The cluster heads are sometimes chosen as specialized nodes that are less energy-constrained. A cluster-head performs aggregation of data and sends it to the sink on behalf of the nodes within its cluster. The most interesting research issue regarding such protocols is how to form the clusters so that the energy consumption and contemporary communication metrics such as latency are optimized. The factors affecting cluster formation and cluster-head communication are open issues for future research.

Moreover, the process of data aggregation and fusion among clusters is also an interesting problem to explore. Since the structure of the
DODAG in the RPL resembles the same, it is best to choose a hierarchical architecture and add security primitives with the best key distribution mechanism based on symmetric key cryptography.

2.6.7 Location Based or Geographical Based Protocols

Most of the routing protocols for sensor networks require location information for sensor nodes. In most cases location information is needed to calculate the distance between two particular nodes so that energy consumption can be estimated. Since, there is no addressing scheme for sensor networks like IP-addresses till before the possibility of utilizing IPv6 addressing and they are spatially deployed on a region, location information can be utilized in routing data in an energy efficient way. For instance, if the region to be sensed is known, using the location of sensors, the query can be diffused only to that particular region which will eliminate the number of transmissions significantly. MECN and SMECN, GAF, GEAR, Chang et al (2007), Akkaya et al (2005), SAR, SPEED are some of the Location based protocols. Some of the protocols discussed are designed primarily for mobile ad hoc networks and consider the mobility of nodes during the design. However, they are also well applicable to sensor networks where there is less or no mobility.

Protocols that utilize the location information and topological deployment of sensor nodes are classified as location-based. The number of energy-aware location-based approaches found in the literature is rather sparse. The problem of intelligent utilization of the location information to aid energy efficient routing is the main research issue and are suitable only for less mobility nodes. Although the performance of these protocols is promising in terms of energy efficiency, further research would be needed to address issues such as Multi-path environment posed by real-time applications.
2.6.8 Multipath Based Routing Protocols

In the multipath routing, multiple paths from source to destination are created and packets will send to destination through these paths. Often, however, selecting the shortest path does not result in reduced network lifetime. Hence, routing algorithms which account for multiple sub-optimal paths need to be considered. In-network aggregation, the fusion of data from different sources promises to increase network lifetime. The data from the lower levels is combined at the higher levels. This approach results in reduced overhead of packet exchanges throughout the network while ultimately communicating the same effective information.

2.6.9 Energy Aware or QoS Based Routing Protocols

Energy-aware QoS routing in sensor networks will ensure guaranteed bandwidth (or delay) through the duration of connection as well as providing the use of most energy efficient path. QoS routing in sensor networks have several applications including real time target tracking in battle environments, emergent event triggering in monitoring applications etc. Currently, there is very little research that looks at handling QoS requirements in a very energy constrained environment like sensor networks.

Necessary survey has been done on various routing protocols and it has been found that some of routing protocols classified below can be protected against some attacks and but not all. Listed secure protocols address to certain threats, attacks and basic security requirements and not all. Special attention towards the key management schemes for the selected protocols has to be designed with the available security primitives.

Another interesting issue for routing protocols is the consideration of node mobility. Most of the current protocols assume that the sensor nodes
and the sink are stationary. However, there might be situations such as battle environments where the sink and possibly the sensors need to be mobile. In such cases, the frequent update of the position of the command node and the sensor nodes and the propagation of that information through the network may excessively drain the energy of nodes. New routing algorithms are needed in order to handle the overhead of mobility and topology changes in such energy constrained environment. Other possible future research for routing protocols includes the integration of sensor networks with wired networks (i.e. Internet). Most of the applications in security and environmental monitoring require the data collected from the sensor nodes to be transmitted to a server so that further analysis can be done. On the other hand, the requests from the user should be made to the sink through Internet. Since the routing requirements of each environment are different, further research is necessary for handling these kinds of situations.

### 2.7 IMPLEMENTATION OF SECURITY PRIMITIVES

The Sensor motes can broadly be classified depending on the capabilities of the microcontroller used in manufacturing as shown in Table 2.1. The selection of a microcontroller depends on what services are to be provided to the node in terms of energy consumption, instruction memory and RAM memory, storage, speed, and external IO ports.

The hardware platforms chosen for this project are TelosB and Aurdino WSN motes which come under the category of Class II and Class III type of nodes. TelosB platform was originally developed at UC Berkeley and is now produced by the Crossbow Technology Company and Aurdino is a Texas instrument microcontroller. Both platforms are tiny, low-power motes with restricted resources, equipped with an 802.15.4 RF interface. Other sensor boards with various sensors, such as accelerometers or magnetometers are also available.
The motes have a USB connector and can directly be plugged into a PC for serial communication or reprogramming. Clearly, these motes are suitable for low data rate applications requiring only minimum data processing. Spending most of their time in the sleep mode, the motes can run for several years on 2 AAA batteries.

**Table 2.1 Classification of Wireless Sensor Motes**

<table>
<thead>
<tr>
<th>Capability</th>
<th>Class I</th>
<th>Class II</th>
<th>Class III</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Atmega168</td>
<td>MSP430F16x</td>
<td>PXA271</td>
</tr>
<tr>
<td>Frequency</td>
<td>4MHz</td>
<td>8 MHz</td>
<td>13 - 416 MHz</td>
</tr>
<tr>
<td>Model</td>
<td>ATmega128L</td>
<td>8 MHz</td>
<td>180MHz</td>
</tr>
<tr>
<td>Word size</td>
<td>8bit</td>
<td>16bit</td>
<td>32bit</td>
</tr>
<tr>
<td>RAM memory</td>
<td>1KB</td>
<td>10KB</td>
<td>256KB</td>
</tr>
<tr>
<td>Inst. memory</td>
<td>16KB</td>
<td>48KB</td>
<td>32MB</td>
</tr>
<tr>
<td></td>
<td></td>
<td>128KB</td>
<td>4MB</td>
</tr>
<tr>
<td>Power (awake)</td>
<td>&lt;1mA</td>
<td>1.8 mA</td>
<td>31-44mA</td>
</tr>
<tr>
<td>Power (slept)</td>
<td>0.1μA</td>
<td>5.1 μA</td>
<td>390μA</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8μA</td>
<td>40μA</td>
</tr>
</tbody>
</table>

Most of the applications need not send large volumes of data through the wireless channel, thus sensor networks will not use the entire bandwidth. Also the execution of the cryptographic primitives will not cause a penalty in the communication between nodes. Suffice to say that the high execution time of Public Key primitives, around 2 seconds, limits their use to specific situations such as key negotiation or broadcast authentication. Regarding memory consumption, the class type of a node will determine the amount of memory available to the application logic, including the security primitives. Therefore, this class type will influence over the type of primitives
that the node can run. Class III type of nodes have roughly 256KB of RAM and 4MB of instruction memory. These type of sensor nodes are powerful enough to cope with any kind of cryptographic primitive, either symmetric or asymmetric, via software.

On the other side of the spectrum, class I nodes have roughly 1KB of RAM and 16KB of instruction memory. They can support software implementations of block cipher algorithms such as AES, RC5, and Skipjack. However, the amount of memory left for implementing the application logic is quite small. Therefore, it may be better to use stream cipher algorithms, whose memory requirements (e.g. 428 bytes of inst. memory for RC4) are quite small. Note that, due to their extreme memory constraints, these kinds of nodes are completely unable to execute Public Key Cryptography on software.

Regarding “class II” nodes, they are the most common hardware platform used on wireless sensor networks. With roughly 4-8KB of RAM and 48-128KB of instruction memory, these nodes are powerful enough to support the execution of any cryptographic primitive on software. Still, considering the actual state of the art, a simple application prototype with support for all the primitives in software will consume 34KB of instruction memory.

From this analysis it is clear that, sensor nodes are capable of running cryptographic primitives such as Symmetric Key Cryptography, Public Key Cryptography, and Hash functions on software. However, the inclusion of HW cryptographic modules in both “class I” and “class II’ nodes should be considered, in order to reduce the overhead posed by the implementation of the primitives.
2.8 **KEY MANAGEMENT**

Security primitives, such as Symmetric Key Cryptography (SKC), Public Key Cryptography (PKC) and Hash functions, can provide a secure communication channel between two or more devices with the properties of confidentiality, integrity, and authentication, protecting the information flow against any unintended recipients. This is essential to create secure protocols and a secure infrastructure, because no external entities or unintended recipients should be able to manipulate the contents of the messages exchanged by two peers. However, such security primitives need certain security credentials, i.e. secret keys, in order to work. The task of creating and distributing these keys, hence constructing a secure key infrastructure, is done by the Key Management Module (KMM).

Consequently, it is necessary to define the properties that a KMS protocol may have, and the requirements of the applications (e.g. network size) that influence over the importance of those properties (e.g. scalability). A Key Management System can be classified by the properties such as Memory Footprint, Processing Speed, Communication Overhead, Security (Confidentiality), Network Resilience, Global Connectivity, Local Connectivity, Node Connectivity, Scalability, Extensibility, and Energy. It is also important to note that the requirements of a certain sensor network application influence over the importance of certain properties for that application.

Leaving the basic inefficient methods of key management such as global key establishment and pair-wise key establishment, the other efficient methods in practice can be broadly classified into Symmetric Key Cryptography based and Public Key Cryptography based. The various schemes in literature spanned under SKC based KMS, can be grouped into Key pool based schemes, Mathematical based schemes and Negotiation based
schemes. Each and every scheme under any one of these categories has its own advantages and disadvantages.

![Figure 2.4 Classification of Key Management System](image)

**Figure 2.4 Classification of Key Management System**

The broad classification of Key Management Schemes is shown in Figure 2.4. Mathematical based schemes provide full connectivity inside the sensor network, since every node can calculate by itself the pair-wise key that it shares with another node. However, these designs are often difficult to apply, and they are not very scalable: The Linear Algebra and Algebraic Geometry schemes need a high amount of memory in order to store the mathematical structures and the combinatorial schemes only work as intended for certain network configurations.

In simple Data centric or flat networks, there is no need to use complex protocols that need of “key pools” or complex negotiations: simple mathematical schemes such as the Blom Key Pre-distribution and Polynomial Key Pre-distribution are enough. On Large Hierarchical networks scalability starts to become an issue and it is better to use other kind of protocols, such as
Dynamic Cluster-based protocols or any “Key Pool” protocol. Scenarios with Mobile Base Stations do not pose a problem, since a sensor node may share a pre-installed pair-wise key with that Base Station. As a result, it is possible to use almost any of the protocols utilized on static networks. On the other hand, in networks with mobile nodes, the number of protocols that fulfill their requirements is limited. Nevertheless, Blom and Polynomial Key Pre-distribution may work for small networks, but for bigger networks it may be necessary to use PKC-based protocols.

If speed becomes the primary property, in those networks where mobile nodes must establish a secure channel in those networks as fast as possible (almost immediately) with other peers and if Extensibility is required, then there is only one useful protocol based on Generalized Quadrangles which is highly dependent on knowing in advance the maximum number of sensor nodes included inside the network.

Regarding the security property, if the security of the network during its initial deployment is not important, it can be possible to use some negotiation-based protocols such as Key Infection in all the groups. For large networks, the redundancy of the network allows to have a tiny fraction of the network disconnected, thus global connectivity can be considered as a secondary property. However, there are some situations where there should be no sensor nodes disconnected from the network. In these situations, global connectivity becomes a main property, and most “key pool” based frameworks cease to be useful. Even more, if extensibility becomes of importance, very few KMS protocols (mostly those based on mathematical frameworks) can become useful for the network designer.
2.9 LOW POWER AND LOSSY NETWORKS

Low power and Lossy networks (LLNs) are a new paradigm for future communications which are made up of many tiny embedded devices with limited power, memory and processing resources. LLNs use low power communication standards which are unreliable in nature. There is a wide scope of application areas for LLNs, including target tracking and surveillance in military, lighting, access and fire control in industrial monitoring and building automation, health care system and scientific exploration in civilian operations. The characteristics of LLNs are as follows:

- Resource limitation of embedded devices
- Multiple traffic patterns P2P or P2MP
- Employment of routing protocol for Link layer with restricted frame – sizes
- Lack of fixed infrastructure
- Unknown network topology prior to deployment
- High risk of physical attacks to unattended sensors

After a decade of research and development in the field of Wireless sensor networks, the lacuna of their interoperability with the internet has been enlightened in real world applications. It has been argued that the IP based network architecture was ill suited for the resource constrained devices that are being embedded in this physical world. Furthermore, without the IP based architecture, the emphasis on the need of an Application gateway to communicate with the other systems or the wider internet has to be created. Routing in LLN has to be cognizant about the power consumption and the unreliable nature of communication. Hence the routing protocol must be
robust to deal with these Network characteristics. The advantages of stateless auto-configuration of IPV6 with large addressing spaces make it as an ideal candidate for LLN. The Internet Engineering task force (IETF) ROLL working group has designed a routing protocol, called RPL for LLNs that allows the nodes to organize themselves as one or more Destination Oriented Directed Acyclic Graphs (DODAG).

2.10 6LOWPAN

A new innovation in Internet protocol technology, called 6LoWPAN is making the Internet of Things become a reality. 6LoWPAN is a standard from the Internet Engineering Task Force (IETF) published in 2007, which optimizes IPv6 for use with communication Technologies such as IEEE 802.15.4 radio. This is where the embedded systems meet the Wireless Technologies. A modern embedded communication chip consists of transceivers which combine half-duplex transmission and reception with the same hardware. Transceivers integrate varying functionalities, from a bare analog interface to whole digital baseband and key MAC functions.

The IoT environment consists of a large number of devices with resource constraint characteristics such as short radio range, limited processing capability and short battery life. Therefore, the IoT implementation requires a communication protocol that can efficiently manage these conditions. 6LoWPAN is a promising solution with the idea of adding an adaption layer called 6LoWPAN in the network protocol stack for integrating low-power networks such as IEEE 802.15.4 into IPv6. This solution can allow the use of the existing infrastructure (Internet Protocol (IP) network) to maximize the utilization of available resources while benefiting from the huge address space of IPv6.
6LoWPAN is an acronym of IPv6 over Low power Wireless Personal Area Networks. The 6LoWPAN network consists of one or more local LoWPANs, which are all connected by IPv6 to the Internet through a gateway (or border router). The LoWPAN devices are characterized by short radio range, low data rate, low power and low cost. The network, therefore, deals with small packet size, low bandwidth and requires resource saving for maintaining the life of network nodes. 6LoWPAN supports both star and peer-to-peer topology; however, the topology can be changed frequently because of uncertain radio frequency, mobility and battery drain.

In the typical model, IP is the only protocol used to connect different protocols from the data link and physical layer to multiple upper layer protocols. 6LoWPAN, however, utilises the 6LoWPAN stack, a combination of LoWPAN adaptation layer and IPv6, to connect its WSNs to the Internet as in Figure 2.5.

![Figure 2.5 Bridge between 6LoWPAN and IP world using edge router](image)

Each node in the 6LoWPAN network has a unique IPv6 address. 6LoWPAN works by compressing the 60bytes of headers down to just 7 bytes and optimizing the mechanism for wireless embedded networking. 6LoWPAN radically alters the calculation by introducing an
adaptation layer that enables efficient IPv6 communication over IEEE 802.15.4 LoWPAN links. By installing IPv6 stack to the sensor nodes, the sensor networks have interoperability with the external IPv6 networks.

2.11 CONTIKI OPERATING SYSTEM

Contiki is an open source, highly portable, multi-tasking operating system for memory-efficient networked embedded systems and wireless sensor networks. Contiki is designed for microcontrollers with small amounts of memory. A typical Contiki configuration is 2 kilobytes of RAM and 40 kilobytes of ROM. Contiki provides IP communication, both for IPv4 and IPv6. Many key mechanisms and ideas from Contiki have been widely adopted in the industry. The uIPv6 embedded IP stack, released in 2001, is today used in systems such as freighter ships, satellites and oil drilling equipment by many companies. Contiki and uIP are recognized by the popular nmap network scanning tool. Contiki's proto-threads, first released in 2005, have been used in many different embedded systems, ranging from digital TV decoders to wireless vibration sensors.

Contiki introduced the idea of using IP communication in low-power sensor networks. This subsequently led to an IETF standard and the IPSO Alliance, an international industry alliance. Contiki is developed by a group of developers from industry and academia lead by Adam Dunkels from the Swedish Institute of Computer Science. The Contiki team currently consists of sixteen developers from SICS, SAP AG, Cisco, Atmel, NewAE and TU Munich.

Contiki contains two communication stacks: uIPv6 and Rime. uIPv6 is a small RFC-compliant TCP/IP stack that makes it possible for Contiki to communicate over the Internet. RIME is a lightweight communication stack designed for low-power radios. Rime provides a wide
range of communication primitives, from best-effort local area broadcast, to reliable multi-hop bulk data flooding. Contiki runs on a variety of platform ranging from embedded microcontrollers such as the MSP430 and the AVR to old home computers. Code footprint is on the order of kilobytes and memory usage can be configured to be as low as tens of bytes. Contiki is written in the C programming language and is freely available as open source under a BSD-style license.

2.11.1 System Overview

A running Contiki system consists of the kernel, libraries, the program loader, and a set of processes. A process may be either an application program or a service. A service implements functionality used by more than one application process. All processes, both application programs and services, can be dynamically replaced at run-time. Communication between processes always goes through the kernel. The kernel does not provide a hardware abstraction layer, but let device drivers and applications communicate directly with the hardware. The system architecture of Contiki OS is shown in Figure 2.6.

Figure 2.6 Contiki System Overview
A Contiki system is partitioned into two parts: the core and the loaded program. The partitioning is made at compile time and is specific to the deployment in which Contiki is used.

Typically, the core consists of

- The Contiki kernel
- The program loader
- The most commonly used parts of the language run-time, support libraries
- A communication stack with device drivers for the communication hardware

The core is compiled into a single binary image that is stored in the devices prior to deployment. The core is generally not modified after deployment, even though it should be noted that it is possible to use a special boot loader to overwrite or patch the core. Programs are loaded into the system by the program loader. The program loader may obtain the program binary.

### 2.11.2 Kernel Architecture

The Contiki kernel consists of a lightweight event scheduler that dispatches events to running processes and periodically calls processes’ polling handlers. All program execution is triggered either by events dispatched by the kernel or through the polling mechanism. The kernel does not preempt an event handler once it has been scheduled. Therefore, event handlers must run to completion. However, event handlers may use internal mechanisms to achieve preemption.
The kernel supports two kinds of events: asynchronous and synchronous events. Asynchronous events are a form of deferred procedure call: asynchronous events are queued by the kernel and are dispatched to the target process some time later. Synchronous events are similar to asynchronous but immediately cause the target process to be scheduled.

Control returns to the posting process only after the target has finished processing the event. In addition to the events, the kernel provides a polling mechanism. Polling can be seen as high priority events that are scheduled in-between each asynchronous event. Polling is used by processes that operate near the hardware to check for status updates of hardware devices. When a poll is scheduled, all processes that implement a poll handler are called, in order of their priority.

The Contiki kernel uses a single shared stack for all process execution. The uses of asynchronous events reduce stack space requirements as the stack is rewound between each invocation of event handlers.

2.11.3 Service Layer Architecture

In Contiki, a service is a process that implements functionality that can be used by other processes. A service can be seen as a form of a shared library. Services can be dynamically replaced at run-time and must therefore be dynamically linked. Typical examples of services include communication protocol stacks, sensor device drivers, and higher level functionality such as sensor data handling algorithms. Services are managed by a service layer, situated directly next to the kernel. The service layer keeps track of running services and provides a way to find installed services. A service is identified by a textual string that describes the service. The service layer uses ordinary string matching for querying installed services. A service consists of a service interface and a process that implements the interface.
The service interface consists of a version number and a function table with pointers to the functions that implement the interface. Application programs using the service use a stub library to communicate with the service. The stub library is linked with the application and uses the service layer to find the service process. Once a service has been located, the service stub caches the process ID of the service process and uses this ID for all future requests. Programs call services through the service interface stub and need not be aware of the fact that a particular function is implemented as a service. The first time the service is called, the service interface stub performs a service lookup in the service layer. If the specified service exists in the system, the lookup returns a pointer to the service interface.

The version number in the service interface is checked with the version of the interface stub. In addition to the version number, the service interface contains pointers to the implementation of all service functions. The function implementations are contained in the service process. If the version number of the service stub matches the number in the service interface, the interface stub calls the implementation of the requested function. The loosely coupled communication stack architecture of Contiki RTOS and its service layers is shown in Figure 2.7.
Figure 2.7 Overview of Contiki Services

Figure 2.8 Loosely Coupled Communication Stack
Since the communication stack is loosely coupled, the different routing protocol configurations co-exist, which can be selected during the compile time as shown in Figure 2.8. The objective functions are of two types, ETX and OF0.

The ETX is based on RSSI and link metrics, where as the OF0 is based on hop count. The ETX objective function provides better results than OF0, so it is selected. Based on RPL parameters the overall network metrics will vary. The variation is studied by means of real time simulation.

2.12 CLOUD COMPUTING

2.12.1 Cloud Conceptual Model

The alphabet of Cloud Conceptual Model lies in mastering the three service models, four deployment models and five fundamental characteristics as depicted in Figure 2.9. The different stake holders involved in various functionalities of the model are listed as:

- Service Models
- Deployment Models
- Fundamental Characteristics
The broad classification of service models includes Infrastructure as a Service, Platform as a Service and Software as a Service as shown in Figure 2.9. Infrastructure as a service is the bottom-most layer in the cloud which provides the consumers with computational power, storage, networks, servers and other fundamental resources. Eg. Amazon’s Elastic Compute Cloud (EC2), IBM, VM Ware, HP. Platform as a Service is the middle service layer which offers deployment platform to the consumers to develop and deploy quick and cost effective applications onto cloud infrastructure using different programming languages and tools supported by cloud vendors. Eg. Microsoft Azure, Google Apps Engine, Salesforce.com. Software as a Service is the top-most service layer which allows the consumers with the capability to use the applications hosted by the vendor through a web browser as client interface on-demand. Eg. Google Docs.

According to the possibilities of deployment, cloud environment is broadly divided into four models namely Private Cloud, Public Cloud, Community Cloud and Hybrid Cloud. Private cloud is dedicated exclusively for a specific private organization which is managed by the organization itself
or by a third party and may exist either on premises or off premises. Public cloud is a cloud infrastructure made available to the general public consumers or any organization and is owned by the organization providing cloud services. Community cloud is a cloud infrastructure which is shared by several organizations and supports a specific community which has common concerns. Hybrid cloud is the composition of two or more types of cloud entities that enables data and application portability and can hold the information in a private or public or community cloud depending on the sensitiveness of the data.

Figure 2.10 shows the Secure Cloud Storage as a Service in which the encrypted data is stored in the cloud servers. This scenario magnifies the possibility of IoT using Cloud Computing. Wireless Sensor Networks of various applications such as Environment monitoring, Transportation, Health care (WSN1, WSN2,...) acts as the Data owners and the data can be consumed by any Data Consumer. Usually a Third Party Auditor (TPA) monitors the secure flow of outsourced data on behalf of the users.

**Figure 2.10 Secure Cloud Storage as a Service**
2.13 SUMMARY

This chapter provides explanation about the basic concepts needed to understand the progress of the research. The Threat models, Routing protocols and Key Management Schemes in literature are broadly classified. The advantages and disadvantages of them are studied in detail. Also the basic concepts of LLN, 6LoWPAN, ContikiOS and Cloud Computing are discussed elaborately.