**The Chinese University of Hong Kong**

**Department of Computer Science and Engineering**

# Intrusion Detection for Wireless Sensor Networks

Ph.D. -- Term 2 Paper

written by Edith C.H. Ngai
supervised by Prof. Michael R. Lyu
Spring 2005

# Abstract

Wireless sensor networks (WSNs) are rapidly emerging as an important area in mobile computing research. Applications of WSNs are numerous and growing, some of them are even security critical, like military or safety applications. Security measures must be applied to protect the network from a variety of attacks. Since no intrusion prevention measures is perfect, intrusion detection becomes an important second wall to protect the network. WSN has unique nature which is different from other kind of networks. It contains a large amount of tiny sensing devices which are limited in energy, computation, and communication capabilities. They are designed for specific applications, and they interact closely with their physical environments. Providing adaptive new intrusion detection measures remain a challenging research problem. In this project, we examine the characteristics and vulnerabilities of WSNs and propose a new intrusion detection framework to protect the network security. Furthermore, we plan to investigate new methodologies to detect, locate, and resist the intrusions on WSNs.

# Table of Content

# 1. Introduction

Wireless sensor network (WSN) is a rapidly emerging area as an important research area. The variety and number of applications are growing on wireless sensor networks. They range from general engineering, environment science, health service, military, etc. Wireless sensor network requires large number of sensor collection data from the environments. They are tiny devices with limited energy, memory, transmission range, and computation power. WSN is self-organized with collaboration among the nodes. Base station is present in the network, which receives the aggregated data from the sensors. It is usually a powerful computer with more computational power, energy, memory, and connected to the Internet.

Some applications in wireless sensor network are secure critical. For military applications, WSNs are dispersed into an adversay's territory for detecting and tracking the enemy soldiers and vehicles. For some indoor environment, sensor networks are deployed to detect intruders via a wireless home security system. WSNs are always unattended but physically reachable from the outside world, so they are vulnerable to security attacks. Therefore, WSNs must be secured to prevent an intruder from obstructing the delivery of correct sensor data and from forging sensor data.

Intrusion prevention measures, such as encryption and authentication, can be used in wireless sensor networks to reduce intrusions, but cannot eliminate them. For example, encryption and authentication cannot defend against compromised sensor nodes, which carry the private keys. From the experiences of security research, no matter how many intrusion prevention messages are inserted in a network, there are always some weak links that one could exploit to break in. Intrusion detection presents a second wall of defense and it is a necessity in any high survivability network [1].

Wireless sensor network is vulnerable to security attacks. To provide a secure wireless sensor networks, we need to deploy intrusion detection and response techniques. Further research is necessary to adapt these techniques to this new environment. In this project, we propose our intrusion detection framework for wireless sensor networks. We will provide the background on wireless sensor network, its security issues, and intrusion detection in Section 2. We will define our research direction in Section 3. Then, we will present the details of our intrusion detection framework on WSN and its detection components in Section 4. Afterwards, we will describe our intruder tracing mechanisms in Section 5. Finally, we will present the conclusion and future work.

# 2. Background

## 2.1. Wireless Sensor Networks

### 2.1.1. Characteristics

**Self-organizing capabilities**

A wireless sensor network (WSN) consists of a large number of sensor nodes. They are deployed over an area and form a wireless network. The position of sensor nodes need not be engineered or pre-determined. This allows random deployment in inaccessible terrains or disaster relief operations. On the other hand, this also means that sensor network protocols and algorithms must possess self-organizing capabilities.

**Cooperative effort of sensor nodes**

A unique feature of sensor networks is the cooperative effort of sensor nodes. Sensor nodes are fitted with an on-board processor. Instead of sending the raw data to the nodes responsible for the fusion, sensor nodes use their processing abilities to locally carry out simple computations and transmit only the required and partially processed data.

**Short-range communication and multihop routing**

Since large number of sensor nodes are densely deployed and they are having short communication range. Hence, multihop communication in sensor networks is expected to consume less power than the traditional single hop communication. Furthermore, the transmission power levels can be kept low, which is highly desired in covert operations. Multihop communication can also effectively overcome some of the signal propagation effects experienced in long-distance wireless communication [4].

**Limitations on energy and computation power**

The sensor nodes are autonomous devices with limited battery, computational power, and memory.

**Dynamic Topology**

Dynamic environmental conditions require the system to adapt over time to changing connectivity and system stimuli.

**Operation**

Figure 2.1 shows the complexity of wireless sensor networks, which generally consist of a data acquisition network and a data distribution network, monitored and controlled by a management center. The plethora of available technologies makes even the selection of components difficult, let alone the design of a consistent, reliable, robust overall system [3].

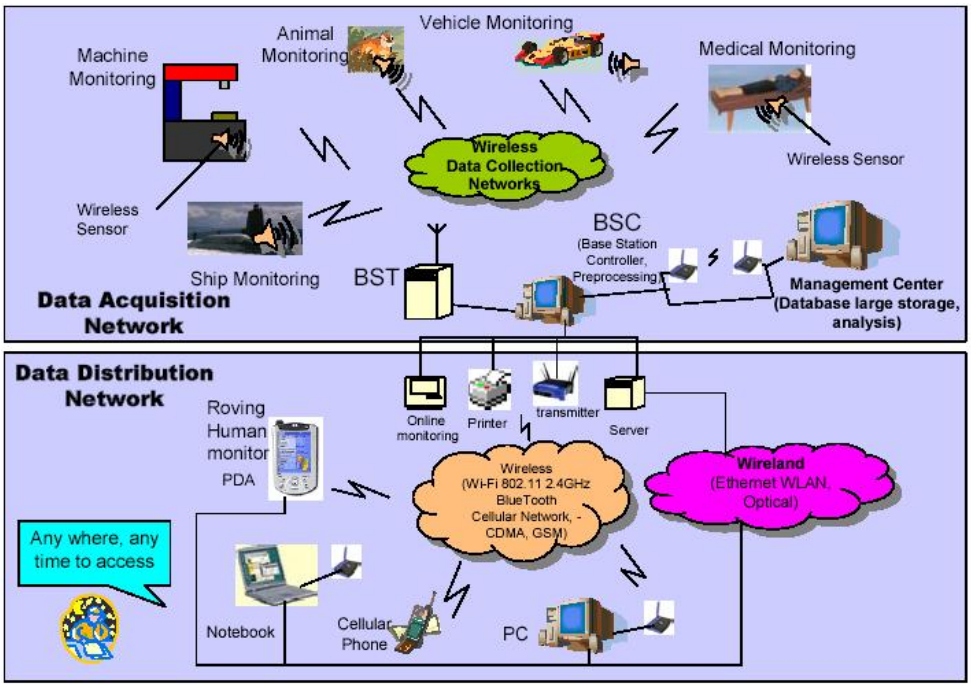



**Figure 2.1 Wireless Sensor Networks**

## 2.1.2. Applications

WSNs encourage several novel and existing applications such as environment monitoring, infrastructure management, public safety, medical and health care, home and office security, transportation, and military applications. The sensor nodes in WSN are very compact and autonomous. They have limited computation and communication capabilities, and limited power supply. The following are some of its applications [2]:

**General engineering**

It can be used for automotive driving, fingertip accelerometer virtual keyboards, sensing and maintenance in industrial plaints, aircraft drag reduction, smart office space management, tracking of goods in retail stores, tracking of containers and boxes in shipping companies, social studies on human behavior, commercial and residential security.

**Agriculture and environmental monitoring**

It can used in crop and livestock management and precision control, planetary exploration, geophysical monitoring, monitoring of freshwater quality, habitat monitoring, disaster detection, can contaminate transport.

**Civil engineering**

It can be used for monitoring of structures, urban planning, and disaster recovery.

**Military applications**

It can be used for asset monitoring and management, surveillance and battle-space monitoring, urban warfare, protection to buildings.

**Health Monitoring and Surgery**

It can be applied in medical sensing, and micro-surgery.

There are a lot opportunities for applying wireless sensor networks. However, a number of challenges must be met before many exciting applications can become realistic and practical. We will then discuss the requirements in building an application on WSN.

### 2.1.3. Requirements

Due the characteristics and limitations of WSN, the following are the requirements in building applications on this network [5]:

**Large number of sensors**

We make use of the cheap small-sized sensors, the summarized new information.

**Low energy use**

In many applications, the sensor nodes will be deployed in a remote area in which case servicing a node may not be possible. Thus, the lifetime of a node may be determined by the battery life, thereby requiring minimal energy expenditure.

**Efficient use of the small memory:**

When building sensor networks, issues such as routing-tables, data replication, security and such should be considered to fit the small size of memory in the sensor nodes.

**Data aggregation**

The huge number of sensing nodes may congest the network with information. To solve this problem, some sensors such as the cluster heads can aggregate the data, do some computation (e.g., average, summation, highest, etc.), and then broadcast the summarized new information.

**Network self-organization**

Given the large number of nodes and their potential placement in hostile locations, it is essential that the network be able to self-organize itself. Moreover, nodes may fail (either from lack of energy or from physical destruction), and new nodes may need to join the network. Therefore, the network must be able to periodically reconfigure itself so that it can continue to function. Individual nodes may become disconnected from the rest of the network, but a high degree of connectivity overall must be maintained.

**Collaborative signal processing**

Yet another factor that distinguishes these networks from Mobile Ad-hoc Networks (MANETs) is that the end goal is the detection/estimation of some event(s) of interest, and not just communication. To improve the detection performance, it is often quite useful to fuse data from multiple sensors. This data fusion requires the transmission of data and control messages. This need may put constraints on the network architecture.

**Querying ability**

There are two types of addressing in sensor network; data-centric, and address-centric according to. In data-centric, a query will be sent to specific region in the network. Whereas, in addressing-centric, the query will be sent to an individual node.

### 2.1.4. Comparison with MANET

**Differences from MANET**

Although many protocols and algorithms have been proposed for traditional wireless ad hoc networks, they are not well suited for the unique features and application requirements of sensor networks. To illustrate this point, the differences between sensor networks and ad hoc networks [6] are outlined below [4]:

- The number of sensor nodes in a sensor network can be several orders of magnitude higher than the nodes in an ad hoc network.
- Sensor nodes are densely deployed.

- Sensor nodes are prone to failures.
- The topology of a sensor network changes very frequently.
- Sensor nodes mainly use broadcast communication paradigm whereas most ad hoc networks are based on point-to-point communications.
- Sensor nodes are limited in power, computational capacities, and memory.
- Sensor nodes may not have global identification (ID) because of the large amount of overhead and large number of sensors.

**Sensor Networks vs Mobile Ad Hoc Networks**

| | **WSN** | **MANET** |
|---|---|---|
| Communication pattern | Specialized to:<br>Many-to-one<br>One-to-many<br>Local communications | Typically support routing between any pair of nodes |
| Energy/resources constrained | More | Less |
| Mobility | Mostly not mobile | Mostly mobile |
| Cooperation among nodes | More cooperative, exhibit trust relationships | Less cooperative |
| Security mechanism | Authentication and routing based on public key cryptography is too expensive | Both public key or asymmetric cryptography can be applied |
| Routing | Distance vector and source routing protocols are generally too expensive | Support different types of routing protocols |

**Advantages over MANET**

Although many protocols and algorithms have been proposed for traditional wireless ad hoc networks, they are not well suited to the unique features and applications requirements of sensor networks. Yes, sensor nodes are prone to failures and may not have global identification (ID). Still, sensor networks have many advantages over the traditional wireless ad hoc network. They are listed as follow [5]:

- Wireless sensor networks improve sensing accuracy by providing distributed processing of vast quantities of sensing information (e.g., seismic data, acoustic data, high-resolution images, etc.). When networked, sensors can aggregate such data to provide a rich, multi-dimensional view of the environment;
- They can provide coverage of a very large area through the scattering of thousands of sensors;
- Networked sensors can continue to function accurately in the face of fail-Network self-organization: Given the large number of nodes and their potential very large area through the scattering of thousands of sensors;
- Networked sensors can continue to function accurately in the face of failure of individual sensors. Thus, allowing greater fault tolerance through a high level of redundancy;

- Wireless sensor networks can also improve remote access to sensor data by providing sink nodes that connect them to other networks, such as the Internet, using wide-area wireless links.
- They can localize discrete phenomenon to save power consumption;
- They can minimize human intervention and management;
- They can work in hostile and unattended environments;
- They can dynamically react to the changing network conditions.

### 2.1.5. Routing

In this section, we survey the state-of-the-art routing protocols for WSNs. In general, routing in WSNs can be divided into flat-based routing, hierarchical-based routing, and location-based routing depending on the network structure. In flat-based routing, all nodes are typically assigned equal roles or functionality. In hierarchical-based routing, however, nodes will play different roles in the network. In location-based routing, sensor nodes' positions are exploited to route data in the network. A routing protocol is considered adaptive if certain system parameters can be controlled in order to adapt to the current network conditions and available energy levels. Furthermore, these protocols can be classified into multipath-based, query-based, negotiation-based, QoS-based, or coherent-based routing techniques depending on the protocol operation. The classification of routing protocols in wireless sensor networks can be referred to the paper written by J. N. Al-Karaki et. al [7] as shown in Figure 2.2. It contains the reference numbers corresponding to that in their paper.

In this project, we mainly focus on the classification of routing protocols by their network structure. We will describe the most representative routing protocols belonging to flat-based routing, hierarchical-based routing, and location-based routing in the following paragraphs.

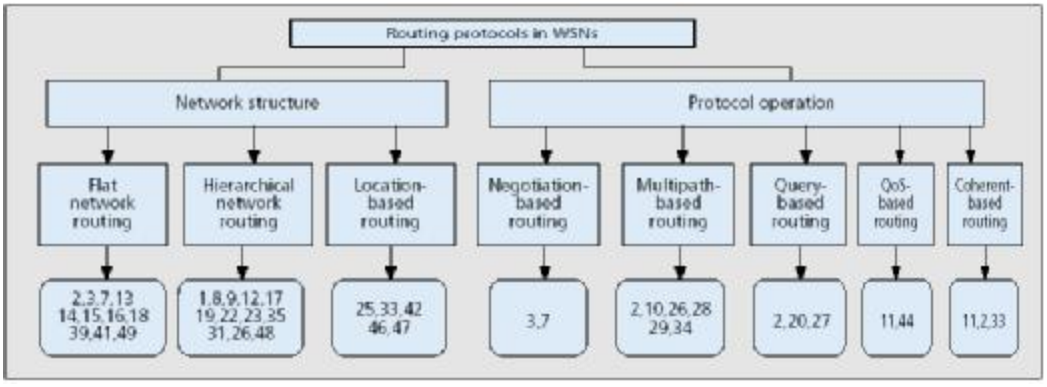



**Figure 2.2 Routing protocols Classification**

## Flat-based routing

In flat-based routing, all nodes are typically assigned equal roles or functionality. They collaborate together to perform the sensing task. Heinzelman et.al. in [8] and [9] proposed a family of adaptive protocols called Sensor Protocols for Information via Negotiation (SPIN) that disseminate all the information at each node to every node in the network assuming that all nodes in the network are potential base-stations. This enables a user to query any node and get the required information immediately. A SPIN node obtains new data and advertises by broadcasting ADV message. An Interested neighbor sends a REQ message for the DATA. The neighbor sensor node repeats this process. It can avoid flooding, localized protocol where only 1-hop neighbors are required. However, it provides no guarantee on data delivery as data far away will not be delivered to destination if intermediates nodes are not interested. In [10], C. Intanagonwiwat et. al. proposed a popular data aggregation paradigm for WSNs, called directed diffusion. Directed diffusion is a data-centric (DC) and application-aware paradigm in the sense that all data generated by sensor nodes is named by attribute-value pairs. In directed diffusion, the sink interests on some application data and sends request. Then, the network builds gradients. Finally, data dissemination takes place. There are 2 models: Event Radius Model and Random Source Model. Unlike SPIN, there is no need to maintain global network topology in directed diffusion. However, directed diffusion may not be applied to applications (e.g., environmental monitoring) that require continuous data delivery to the BS. This is because the query- driven on demand data model may not help in this regard. Moreover, matching data to queries might require some extra overhead at the sensor nodes.

## Hierarchical-based routing

Nodes will play different roles in the network. Hierarchical routing is mainly two-layer routing where one layer is used to select clusterheads and the other layer is used for routing. Heinzelman, et. al. [11] introduced a hierarchical clustering algorithm for sensor networks, called Low Energy Adaptive Clustering Hierarchy (LEACH). LEACH is a cluster-based protocol, which includes distributed cluster formation. LEACH randomly selects a few sensor nodes as clusterheads (CHs) and rotate this role to evenly distribute the energy load among the sensors in the network. In LEACH, the clusterhead (CH) nodes compress data arriving from nodes that belong to the respective cluster, and send an aggregated packet to the base station in order to reduce the amount of information that must be transmitted to the base station. LEACH assumes that all nodes can transmit with enough power to reach the BS if needed. It is not applicable to networks deployed in large area. In [12], Sensor

Aggregation Routing is a set of algorithms for constructing and maintaining sensor aggregates were proposed. The objective is to collectively monitor target activity in a certain environment (target tracking applications). A sensor aggregate comprises those nodes in a network that satisfy a grouping predicate for a collaborative processing task. The parameters of the predicate depend on the task and its resource requirements. Another protocol is the Virtual Grid Architecture Routing (VGA) proposed in [13]. It is an energy-efficient routing paradigm that utilizes data aggregation and in-network processing to maximize the network lifetime. Square clusters were used to obtain a fixed rectilinear virtual topology. Inside each zone, a node is optimally selected to act as clusterhead. Data aggregation is performed at two levels: local and then global. The set of clusterheads, also called Local Aggregators (LAs), perform the local aggregation, while a subset of these LAs are used to perform global aggregation. However, the determination of an optimal selection of global aggregation points, called Master Aggregators (MAs), is NP-hard problem. Figure 2.3 illustrates an example of fixed zoning and the resulting virtual grid architecture (VGA) used to perform two level data aggregation.

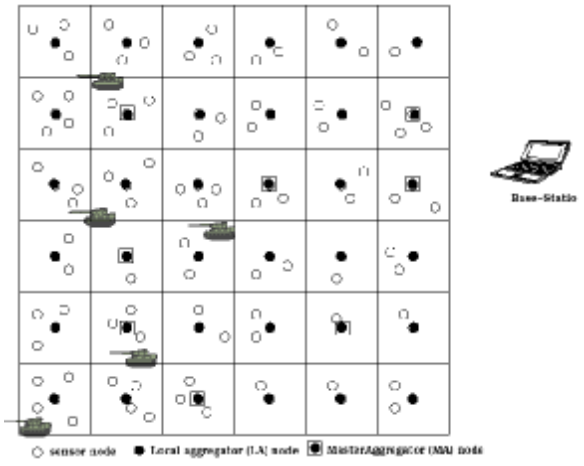



**Figure 2.3 VGA Routing Protocol**

## Location-based routing

In location-based routing, sensor nodes' positions are exploited to route data in the networks. Geographic Adaptive Fidelity (GAF) [14] is an energy-aware location-based routing algorithm designed primarily for mobile ad hoc networks, but may be applicable to sensor networks as well. The network area is first divided into fixed zones and forms a virtual grid. Inside each zone, nodes collaborate with each other to play different roles. GAF conserves energy by turning off unnecessary nodes in the network without affecting the level of routing fidelity. Yu et al. [15] proposed Geographic and Energy Aware Routing (GEAR). It discussed the use of

geographic information while disseminating queries to appropriate regions since data queries often include geographic attributes. The protocol, called Geographic and Energy Aware Routing (GEAR), uses energy aware and geographically-informed neighbor selection heuristics to route a packet towards the destination region. The key idea is to restrict the number of interests in directed diffusion by only considering a certain region rather than sending the interests to the whole network. By doing this, GEAR can conserve more energy than directed diffusion.

## 2.2. Security in WSN

As mentioned in the previous section, sensor networks pose unique challenges. Security techniques used in traditional networks cannot be applied directly. First, we have to make sensor networks economically viable as sensor devices are limited in their energy, computation, and communication capabilities. Second, unlike traditional, sensors are often deployed in accessible area, presenting the added risk of physical attack. Third, sensor networks interact closely with their physical environments and with people, posing new security problems. Consequently, existing security mechanisms are inadequate, and new ideas are needed. The new problems inspire new research and provide an opportunity to properly address sensor network security [16].

### 2.2.1. Threat Models

Some people consider attackers as insider attacks and outsider attacks. In an outsider attack, the attack node is not an authorized participant of the sensor network. As the sensor network communicates over a wireless channel, a passive attacker can easily eavesdrop on the network's radio frequency range, in an attempt to steal private or sensitive information. The adversary can also alter or spoof packets, to infringe on the authenticity of communication or inject interfering wireless signals to jam the network. Another form of outsider attack is to disable sensor nodes. An attacker can inject useless packets to drain the receiver's battery, or he can capture and physically destroy nodes. A failed node is as same as a disabled node [17].

Different from outsider attackers, insider attacks are performed by compromised nodes in the WSN. With node compromise, an adversary can perform an insider attack. In contrast to disabled node, compromised nodes activity seeks to disrupt or paralyze the network. A compromised node may be a subverted sensor node or a more powerful device, like laptop, with more computational power, memory, and powerful radio. It may be running some malicious code and seek to steal secrets from the sensor network or disrupt its normal functions. It may have a

radio compatible with sensor nodes such that it can communicate with the sensor network. A compromised node can exhibit arbitrary behavior, which is well known as the Byzantine model [18].

### 2.2.2. Security Requirements

Authentication is necessary to enable sensor nodes to detect maliciously injected or spoofed packets. It enables a node to verify the origin of a packet and ensure data integrity. Almost all applications require data authentication. On one hand, for military and safety-critical applications, the adversary has clear incentives to inject false data reports or malicious routing information; on the other hand, even for civilian applications such as office/home applications where we expect a relatively non-adversarial environment. Although authentication prevents outsiders from injecting or spoofing packets, it does not solve the problem of compromised nodes. Since a compromised node has the secret keys of a legitimate node, it can authenticate itself to the network. However, we may be able to use intrusion detection techniques to find the compromised nodes and revoke their cryptographic keys network-wide.

Ensuring the secrecy of sensed data is important for protecting data from eavesdroppers. We can use standard encryption functions and a shared secret key between the communicating parties to achieve secrecy. However, encryption itself is not sufficient for protecting the privacy of data, as an eavesdropper can perform traffic analysis on the overheard ciphertext, and this can release sensitive information about the data. In addition to encryption, privacy of sensed data also needs to be enforced through access control policies at the base station to prevent misuse of information. Node compromise complicates the problem of secrecy, for sensitive data may be released when a compromised node is one endpoint of the communication; or if a globally or group shared key is used, the compromised node can successfully eavesdrop and decrypt the communication between other sensor nodes within its radio frequency (RF) range.

Providing availability requires that the sensor network be functional throughout its lifetime. Denial-of-service (DoS) attacks often result in a loss of availability. In practice, loss of availability may have serious impacts. In a manufacturing monitoring application, loss of availability may cause failure to detect a potential accident and result in financial loss; in a battlefield surveillance application, loss of availability may open a back door for enemy invasion. Various attacks can compromise the availability of the sensor network. When considering availability in sensor networks, it is important to achieve graceful degradation in the presence of node compromise or benign node failures.

Service integrity is another security requirement. Above the networking layer, the sensor network usually implements several application-level services. Data aggregation is one of the most important sensor network services. In data aggregation, a sensor node collects readings from neighboring nodes, aggregates them, and sends them to the base station or another data processing node. The goal of secure data aggregation is to obtain a relatively accurate estimate of the real-world quantity being measured, and to be able to detect and reject a reported value that is significantly distorted by corrupted nodes [17].

### 2.2.3. Security Measures

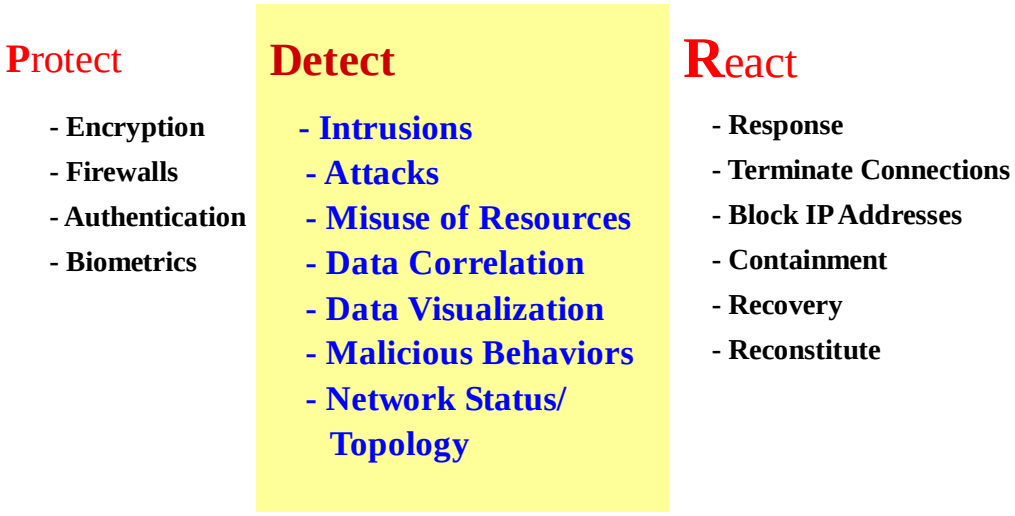
One common measure to protect the network is key establishment and management. The key management protocol must establish a key between all sensor nodes that must exchange data securely. It should support node addition or deletion and be able to work in undefined deployment environment. Unauthorized nodes should not be allowed to establish communication with network nodes.

There are trusted-server schemes for the key establishment and management, but fining trusted servers is difficult in WSN. Public key cryptography is a popular method for key establishment, but it is too expensive and infeasible for sensors. The more practical approaches are the key pre-distribution schemes. Single mission key is obviously unacceptable. It is because the adversary only has to compromise one node will compromise the whole system. Pairwise private key sharing is a straight forward method. However, setting up secret keys between every two nodes is impractical because it requires pre-distribution and storage of n-1 keys in each node which is n(n-1)/2 per WSN. Most of the keys would be unusable since direct communication is possible only in the nodes neighborhood. Also, addition or deletion of the node and re-keying are complex.

A basic Probabilistic Approach is proposed by Eschenauer and Gligor [19]. It relies on probabilistic key sharing among nodes of WSN and uses simple shared-key discovery protocol for key distribution, revocation and node re-keying. Three phases are involved: key pre-distribution, shared-key discovery, path-key establishment. Recently, researchers proposes a class of random key predistribution techniques that address the problem of key establishment [key2,key3]. A deployment-based Scheme proposed by Du, et. al [22]. It improves Random Key Predistribution (Eschenauer and Gligor) by exploiting Location Information. It studies a Gaussian distribution for deployment of Sensor nodes to improve security and memory usage.

Key establishment and management is mainly a prevention approach in network defense. Apart from protection, there exist also some detection and reaction techniques for protecting the network security. The following figure shows the

division of techniques among protection, detection, and reaction. In this term paper, we will focus more in the detection of intrusions in wireless sensor networks.




## 2.3. Intrusion Detection

Intrusion detection is the process of discovering, analyzing, and reporting unauthorized or damaging network or computer activities. It discovers violations of confidentiality, integrity, and availability of information and resources. Intrusion detection demands as much information as the computing resources can possibly collect and store. It requires experienced personnel who can interpret network traffic and computer processes. It needs constant improvement of technologies and processes to match pace of Internet innovation. Intrusion can provide digital forensic data to support post-compromise law enforcement actions. It can identify host and network misconfigurations, improve management and customer understanding of the Internet's inherent hostility. Also, it is able to learn how hosts and networks operate at the operating system and protocol levels.

### 2.3.1. Intrusion Detection for Traditional Network

All computer activity and network traffic falls in one of three categories, including normal, abnormal but not malicious, and malicious. Properly classifying these events are the single most difficult problem, even more difficult than evidence collection. Two primary intrusion detection models are the network-based intrusion detection and the network-based intrusion detection. Network-based intrusion detection monitors network traffic for signs of misuse. Host-based intrusion detection monitors computer processes for signs of misuse. Systems are called "hybrid" systems if they do both. A hybrid IDS on a host may examine network

traffic to or from the host, as well as processes on that host.

Intrusion detection paradigms include the following:

- Ÿ Anomaly Detection - the AI approach
- Ÿ Misuse Detection - simple and easy
- Ÿ Burglar Alarms - policy based detection
- Ÿ Honey Pots - lure the hackers in
- Ÿ Hybrids - a bit of this and that

Among all, anomaly detection and misuse detection are the most common traditional intrusion detection techniques. The following are the details of the two techniques:

- ² Anomaly detection
  - Ÿ Establish a profile of the subject's normal activities
  - Ÿ Consider a deviations of a subject's observed activities from its norm profile
  - Ÿ A subject – a user, file privileged program, host machine, network
  - Ÿ Disadvantages: May treat all anomalies as attacks, false alarms are anticipated
- ² Misuse detection (pattern recognition)
  - Ÿ Identify and store signature patterns of known attacks
  - Ÿ Match observed behavior with known patterns of attack signatures
  - Ÿ Attack signatures -- e.g. Strings, even sequences, activity graphs, attack scenarios (event sequences, preconditions, target compromised states)
  - Ÿ Disadvantages: cannot detect novel or unusual attacks whose signatures are unknown; have to update the attack signature patterns constantly

Intrusion detection model consists of six main components [23]:

- Ÿ Subjects: Initiator of activity on a target system-normally users.
- Ÿ Objects: Resources managed by the system-files, commands, devices, etc.
- Ÿ Audit records: Generated by target systems in responses to actions performed or attempted by subjects on objects-user login, command execution, file access, etc.
- Ÿ Profiles: Structures that characterize the behavior of subjects with respect to objects in terms of statistical metrics and models of observed activity. Profiles are automatically generated and initialized from templates.
- Ÿ Anomaly records: Generated when abnormal behavior is detected.
- Ÿ Activity rules: Actions taken when some condition is satisfied, which update profiles, detect abnormal behavior, relate anomalies to suspected intrusions, and produce reports.

Observed behavior is characterized in terms of a statistical metric and model. A metric is a random variable x presenting a quantitative measure accumulated over a period. The statistical models may be an operational model, mean and standard deviation model, multivariate model, markov process model, time series model, etc [23].

The paper [24] analysis the characteristics of the activity graphs, detects and reports violations of the stated policy. It uses a hierarchical reduction scheme for the graph construction, which allows it to scale to large networks. An early prototype of GrIDS has successfully detected a worm attack.

### 2.3.2. Intrusion Detection for MANET

In this section, we will discuss some related work on intrusion prevention and detection for mobile ad hoc networks.

The paper [1] shows that the architecture for better intrusion detection in wireless ad hoc networks should be distributed and cooperative. A statistical anomaly detection approach should be used. The trace analysis and anomaly detection should be done locally in each node and possibly through cooperation with all nodes in the network. Further, intrusion detection should take place in all networking layers in an integrated crosslayer manner.

In [25], it makes use of geometric random graphs induced by the communication range constraints of nodes and presents the necessary and sufficient conditions for detection and defending against wormholes. Their defense mechanism is based on local broadcast keys.

The paper [26], we introduce a secure routing protocol called JANUS, which focuses on the establishment of secure routes between the base station and mobile devices, and the secure routing of the data.

In [27], it presents the rushing attack as a new attack which results in denial-of-service when used against all previous on-demand ad hoc network routing protocols. It then develops Rushing Attack Prevention (RAP), a generic defense against the rushing attack for on-demand protocols. Also, the same authors present a generic mechanism, called packet leashes, for detecting and thus defending against wormhole attacks in [28].

The paper [29] considers ad hoc networks with multiple, mobile colluding intruders. It investigates the placement of the intrusion detection modules for misuse intrusion detection. It mathematically formulates different detection objectives, and shows that computing the optimal solution is NP-had in each case. Another paper [30] written by the same authors consider the signature detection technique and investigate

the ability of various routing protocols to facilitate intrusion detection when the attack signatures are completely known. They show that reactive ad-hoc routing protocols suffer from a serious problem due to which it might be difficult to detect intrusions even in the absence of mobility. Mobility makes the problem of detecting intruders harder.

### 2.3.3. Intrusion Detection for WSN

There are not many papers working on general intrusion detection techniques for wireless sensor networks, relatively more works on intrusion detection for specific kind of attacks, like wormhole attacks, routing holes, or to particular operations, like routing, localization, etc. One paper proposes an anomaly approach based on self-organized criticality (SOC) and Hidden Markov models to detect data inconsistencies [31]. This approach is developed based on the structure of naturally occurring events. With the acquired knowledge distilled from the self-organized criticality aspect of the deployment region, it applies a hidden Markov model. It lets sensor networks adapt to the norm of the dynamics in its natural surrounding so that any unusual activities can be singled out.

Another paper formulates the attack-defense problem by game theory and use Markov Decision Process to predict the most vulnerable sensor nodes [32]. It formulates attack-defense problem as a two-player, nonzero-sum, non-cooperative game between an attacker and a sensor network. It shows that this game achieves Nash equilibrium and thus leading to a defense strategy fro the network. Then, it uses Markov Decision Process to predict the most vulnerable sensor node. Finally, it uses an intuitive metric (node's traffic) and protects the node with the highest value of this metric.

Indeed, wireless sensor networks are susceptible to many forms of intrusion. In wired networks, traffic and computation are typically monitored and analyzed for anomalies at various concentration points. This is often expensive in terms of network's memory and energy consumption, as well as its inherently limited bandwidth. Wireless sensor networks require a solution that is fully distributed and inexpensive in terms of communication, energy, and memory requirements. In order to look for anomalies, applications and typical threat models must be understood. It is particularly important for researchers and practitioners to understand how cooperating adversaries might attack the system. The use of secure groups may be a promising approach for decentralized intrusion detection [16].

Apart from those more general approaches, some papers provide intrusion detection techniques for particular operations. In [33], it describes a distributed algorithm, BOUNDHOLE, to build routes around the routing holes, which are

connected regions of the network with boundaries consisting of all the stuck nodes. It shows that hole-surrounding routes can be used in geographic routing, path migration, information storage mechanisms and identification of regions of interest. The paper [34] proposes a general scheme to detect localization anomalies that are caused by adversaries. It formulates the problem as an anomaly intrusion detection problem, and proposes a number of ways to detect localization anomalies. In [35], it describes an intrusion detection technique that uses information about both the network topology and the positioning of sensors to determine what can be considered malicious in a particular place of the network. The technique relies on an algorithm that automatically generates the appropriate sensor signatures. It applies that approach to an intra-domain distance-vector protocol and reports the results of its evaluation.

Moreover, there are some papers applying fault-tolerant technologies in providing network security. In [36], secure multi-path routing to multiple destination base stations is designed to provide intrusion tolerance against isolation of a base station. Also, anti-traffic analysis strategies are proposed to help disguise the location of the base station from eavesdroppers. The paper in [37] targets the identification of faulty sensors and detection of the reach of events in sensor networks with faulty sensors. It proposed two algorithms for faulty sensor identification and fault-tolerant event boundary detection. These algorithms are localized and scalable for WSNs.

In this project, we will investigate new intrusion detection technologies for wireless sensor network which adapt to the network features. Also, we will consider possible methodologies by considering on the routing, network topology, and application data mining, etc.

# 3. Research Direction

In wireless sensor networks, sensor nodes have limited battery power, memory, and processing power. They form a distributed and self-organizing wireless network with multihop routing topology. Based stations (sinks) are powerful machines for communication amongst themselves. To reduce the total number of messages sent and save energy, sensor readings from multiple nodes may be processed at one of many possible aggregation points. An aggregation point collects sensor readings from surrounding nodes and forwards a single message representing an aggregate of the values. Typical applications may periodically transmit sensor readings for processing. In this project, we are interested in providing network security by intrusion detection. More than detecting an intrusion, we concern especially about how the attackers in a wireless sensor network can be traced.

## 3.1. Intrusion Detection, Tracing, and Reaction

In this project, we will investigate various solutions for intrusion detection on wireless sensor networks. We define the following intrusion detection framework (Figure 3.1) which adapt to the environment of wireless sensor networks. It is able to detect the intrusions, locate the attackers, and react to the intrusions.

Our intrusion detection framework is composed of three parts, including intrusion detection, intrusion tracing, and intrusion reaction. These three procedures are performed one after another. Firstly, intrusion detection will be performed. If intrusions exist, then intrusion tracing will be executed to locate the attacks. Finally, intrusion reaction will carry out to defend against the attacks. A variety of audit data will be examined to support the processes of intrusion detection and tracing, like the application data from sensing, the routing information, the behaviors of individual nodes, and the network topology. These audit data can be collected by different techniques. Localization can provide the location of nodes, such that it builds a global view on the network topology. Data fusion can analyze the application data and discover inconsistency among them. Understanding the routing activities is important as many attackers invade the networks by manipulating the routing protocol. Sensor nodes equipped with monitoring components are capable to observe the behavior of its neighboring nodes. Apart from the above techniques, the events of intrusions will be recorded in the history. This may help for future analysis and prediction. All the data collected will assist the intrusion detection and intrusion tracing procedures for detecting the attacks and identifying the attackers. After that, the intrusion reaction procedure will provide effective solution to protect the

network against the attacks.

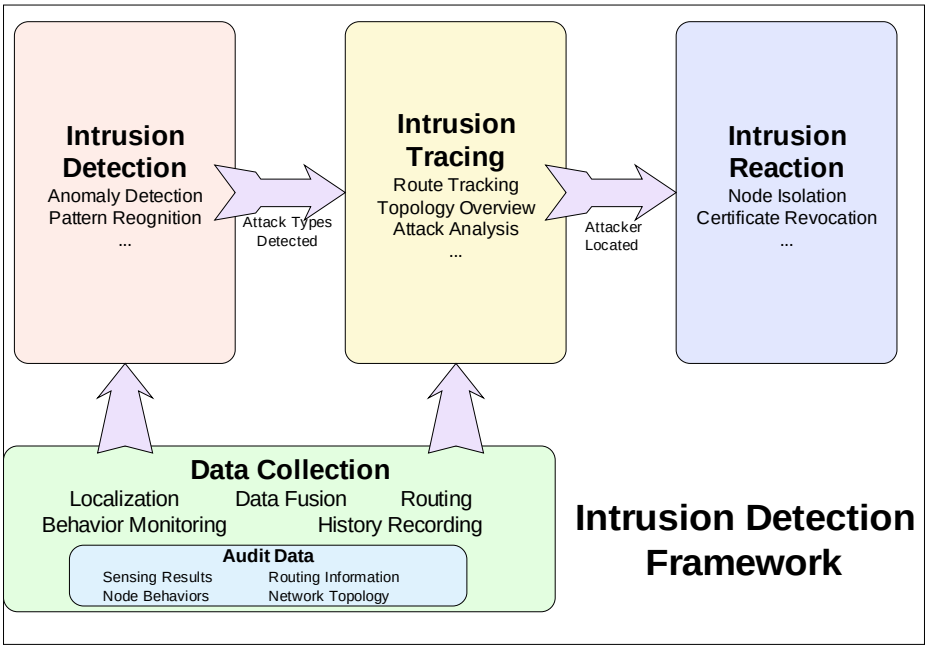



**Figure 3.1 Intrusion Detection Framework**

The intrusion detection process will detect the intrusions in the network by sensing any suspicious phenomenon from the audit data. If it detected an intrusion on the network, it will then classify what kind of attack it belongs to. The traditional mechanisms for detecting an attack are anomaly detection and pattern recognition. We will investigate new methods to collect relevant audit data and detect intrusions especially for wireless sensor networks.

Intrusion tracing is an important process in our intrusion detection framework. In addition to sensing an intrusion, we work further to identify and locate the attack. We believe identifying and locating the attack aids the determination of more effective solutions for defending against the attacks. The architecture, traffic pattern, and behaviors are different from traditional networks and mobile ad hoc networks. We aim at developing novel and effective mechanisms for tracing the attackers on wireless sensor networks.

Intrusion reaction is the final stage in this intrusion detection framework. Its main purpose is to defend against the attacks after discovering, classifying, identifying, and locating the attacks. Traditional methods for this kind of protection include node isolations and certificate revocation. They have the same effect of detaching the malicious devices from the network, such that the network can operate normally and exempt from further attacks. We plan to propose new methods to react to these attacks which can effectively punish the attackers and minimize the interference to normal nodes. For example, we may offer more specific measures in

response to particular type of attacks on the network.

## 3.2. Possible Problems

There are a number of possible research problems about intrusion detection on wireless sensor networks. In this section, we will describe some of them.

We may investigate the suitable system architecture providing intrusion detection for wireless sensor networks. Hierarchical intrusion detection architecture is one possible solution. It is because the sensor nodes, the aggregation nodes, and the sink are responsible for different tasks on detection and analysis under this hierarchical structure.

Moreover, we are especially interested in finding the proper ways to detect the attacks in the network. Different types of intrusions may occur in various layers under the OSI model. We suggest that detection mechanisms can be applied separately in different layers. For example, we may focus on application data collection and encryption/decryption in application layer; routing protocol and routing information in network layer. We can also examine which kind of detection mechanism, like misuse detection and anomaly detection, is more effective for detecting specific types of intrusions.

Intrusion detection depends on audit data collection and data analysis skills. We will study how to collect data for building normal profile and attack pattern. Accurate information on system activities increases the chance of detecting an intrusion successfully. Also, we will explore the techniques in knowledge engineering, like Markov model, for analyzing the audit data and modeling the system behavior, so that intrusions can be detected effectively.

Wireless sensor network may be a large network with many sensor nodes. Adapting to this scalable and distributed environment is important. Promoting the collaboration among the nodes can do so. Similar to the system architecture, we will consider how sensor nodes, aggregation nodes, and sink can cooperate and share their information and opinions together to bring better results on intrusion detection.

In addition, we concern about maintaining an up-to-date system status for better understanding on the network condition. Firstly, the intrusions which have happened should be correctly recorded. They may assist in adding in new attack patterns for pattern recognition. Secondly, the system should collect the most recent behaviors of the network and preserve the normal operations of the system. Thirdly, it is essential to isolate the malicious devices and inform the other nodes. We expect that novel intrusion resistant measures can be carried out after detecting an intrusion.

Furthermore, we may examine other security measures related to intrusion detection, like the intrusion-tolerant technologies, and see if they can enhance they accuracy, performance, or completeness of our intrusion detection framework.

## 3.3. Future Direction

In order to complete the above intrusion detection framework, we will investigate methodologies to collect representative audit data correctly. Also, we will decide on and complete the intrusion detection architecture for our framework. It should define the network architecture, the detection components, and their cooperation. Then, we will investigate methods to analyze the audit data for intrusion detection. Furthermore, we will explore mechanisms to locate and resist to the intrusions effectively. Finally, we will formulate and evaluate our intrusion detection solution

# 4. Intrusion Detection in Wireless Sensor Networks

## 4.1. Security Related Properties

Four properties that are specific for WSNs and require attention are hostile environment, limited resources, in-network processing, and application-specific architecture [2].

**Hostile environment**

WSNs can be deployed in hostile environments such as battlefields. In these cases, it is hard to protect nodes from physical attacks. Security information potentially could be collected from compromised nodes. The development of tamper-proof nodes is one approach to security in hostile environment. However, the development of such systems is far from simple and certainly not cheap in terms of computation and memory requirements [38]. Because of the physical accessibility of sensor nodes, the security mechanisms for WSNs are specifically concerned with situations in which one or more nodes are compromised.

**Limited resources**

Sensor network nodes are designed to be compact and therefore are limited by size, energy, computational power, and storage. The limited resources limit the types of security algorithms and protocols that can be implemented. Security solutions for WSNs operate in a solution space defined by the trade-off between resources spent on security and the achieved protection.

**In-network process**

Communications between the nodes in WSN consumes most of the available energy, much less than sensing and computation do. For that reason, WSNs perform localized processing [39] and data aggregation [40]. An optimal security architecture for this type of communication is one in which a group key is shared among the nodes in an immediate neighborhood. However, in an environment in which the nods can be captured, the confidentiality offered by the shared symmetric keys is easily compromised.

**Application-specific architectures**

WSN system architecture must be designed to be application specific. The flexibility of a general-purpose architecture is traded for the efficient utilization of the resources. Almost every aspect of a WSN can be adjusted to improve performance

and optimize resource consumption in a network for a particular application. This allows a network designer to determine the importance of various security threats and adjust security mechanisms to these threats.

## 4.2. Types of Attacks

Wireless sensor network is vulnerable to different kinds of attacks. We divided them into different categories based on physical, application layer, network layer, MAC layer attacks.

**Physical Attacks**

Since wireless sensor networks can be deployed in hostile environment or densely populated areas, physical access to individual nodes is possible. Even causal passersby may be able to damage, destroy, or tamper with sensor devices. Destruction of the node could cause gaps in sensor or communication coverage. More equipped attacks can interrogate a device's memory, stealing its data or cryptographic keys. Its code can be replaced with a malicious program which is potentially undetectable to neighboring nodes. The capability profile of the subverted node becomes a fully authorized insider.

**Application level attack**

The most common kind of application or service level attack is the denial of services attack. A denial of service attack is any even that diminishes or eliminates a network's capability to perform its expected functions [41]. It is the general result of any action that prevents any part of a WSN from functioning correctly or in a timely manner. Hardware failures, software bugs, resource exhaustion, environmental conditions, or any complicated interaction between these factors can cause a DoS. An adversary may possess a broad range of attack capabilities. A physically damaged or manipulated node used for be less powerful than a normally functioning node. Subverted nodes that interact with the network only through software are as powerful as other nodes.

**Network layer attack**

Most of the attacks on wireless sensor are routing related. It may due to abnormal updates to routing table, malicious packet on a route, false route requests, too many packets, no reply from destination, packets with incorrect encryption, failed to forward a packets, etc [42]. The following paragraphs will describe the attacks of misdirection, selective forwarding, sinkhole attack, sybil attack, wormholes, and hello flood attacks.

Misdirection on routing can be performed by spoofed, altered, replayed routing information. By forwarding messages along wrong paths, an attacker misdirects them, perhaps by advertising false routing updates. An attacker could inflict the attack on a particular sender by diverting only traffic originating from the victim node.

In a Selective forwarding attack, malicious nodes may refuse to forward certain messages and simply drop them, ensuring that they are not propagated any further. A simple form of this attack is when a malicious node behaves like a black hole and refuses to forward every packet she sees. However, such an attacker runs the risk of neighboring nodes concluding that she has failed and deciding to seek another route. A more subtle form of this attack is when an adversary selectively forwards packets. An adversary interested in suppressing or modifying packets originating from a select few nodes can reliably forward the remaining traffic and limit suspicion of her wrongdoing.

In a sinkhole attack, the adversary's goal is to lure nearly all the traffic from a particular area through a compromised node, creating a metaphorical sinkhole with the adversary at the center. Because nodes on, or near, the path that packets follow have many opportunities to tamper with application data, sinkhole attacks can enable many other attacks, like selective forwarding.

In a Sybil attack [43], a single node presents multiple identities to other nodes in the network. Any system whose correct behavior is based on the assumption that most nodes will behave properly may be at risk for Sybil attacks. The Sybil attack is especially threatening to fault-tolerant schemes such as distributed storage, disparity and multipath routing, and topology maintenance. Replicas, storage partitions, or routes believed to be using disjoint nodes could in actuality be using a single adversary presenting multiple identities.

In the wormhole attack, an adversary tunnels messages received in one part of the network over a low latency link and replays them in a different part. The simplest instance of this attack is a single node situated between two other nodes forwarding messages between the two of them. However, wormhole attacks will more commonly involve two distant malicious nodes colluding to understate their distance from each other by relaying packets along an out-of-bound channel available only to the attacker.

Many protocols require nodes to broadcast HELLO packets to announce themselves to their neighbors, and a node receiving such a packet may assume that it is within (normal) radio range of the sender. This assumption may be false: a laptop-class attacker broadcasting routing or other information with large enough transmission power could convince every node in the network that the adversary was

its neighbor.

**MAC layer attack**

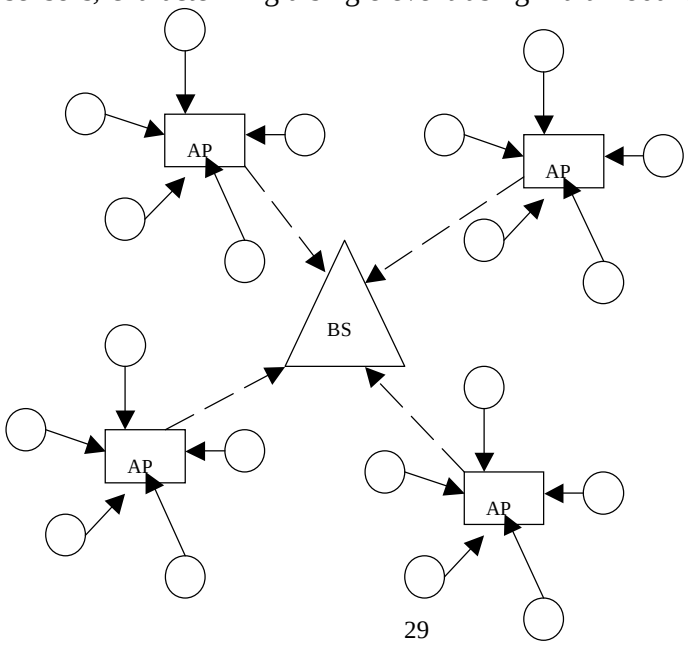
Jamming is deliberate interference with radio reception to deny the target's use of a communication channel. For single-frequency networks, it is simple and effective, rendering the jammed node unable to communicate or coordinate with others in the network.

Similarly to jamming the physical radio channel, an attack can willfully cause collisions or corruption at the link layer. By detecting and parsing radio transmissions near the victim, the attacker can disrupt key elements of packets, such as fields that contribute to checksums or the checksums themselves. With little effort or duration of transmission, the attacker may be able to cause the victim to discard a much longer packet, thus wasting the channel access as well as the transmitter's energy. Such disruption may trigger back-offs in contention-control mechanism that delay other messages.

## 4.3. System Architecture

### 4.3.1. Hierarchical-based Approach

In hierarchical network architecture, the functions of sensing, computation, and data delivery are unequally divided among the nodes (Figure 4.1). These functions may be divided across the tiers, with the lowest tier performing all sensing, the middle tier performing all computation, and the top tier perform all data delivery. Alternatively, each layer can perform a specialized role in computation. For example, lower level sensors might provide a simple band-pass filter to cull interesting data from noise, while nodes at a higher tier might fuse the filtered data received from multiple sensors, characterizing a single event using multimodal sensor data (Figure 4.2).



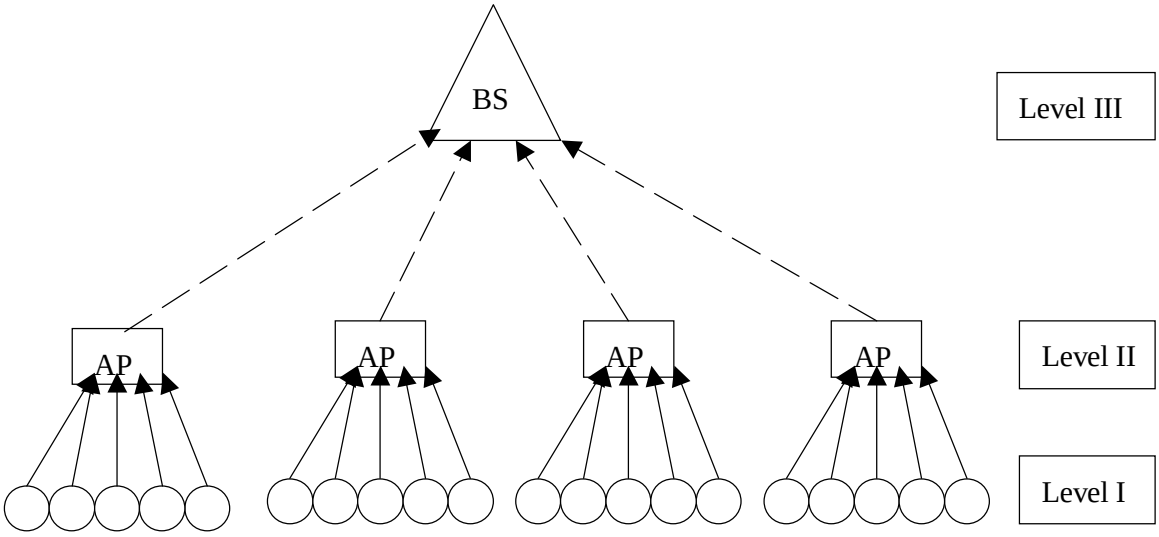



**Figure 4.1 Hierarchical-based Architecture**

**Level I: Sensor nodes (SN)**

1. Data collection – collecting application data
2. Neighbor monitoring facility – monitor behavior of neighboring nodes
3. Intrusion reaction engine – react to intrusions (by isolating the relevant nodes)

**Level II: Aggregation points (AP)**

1. Data aggregation – aggregate the application data from sensor nodes nearby
2. Local detection engine – monitor behavior of the network or individual nodes
3. Intrusion identification – identify the intrusions by analyzing the aggregated data and the network behavior (by chi square distance measure, EWMA forecasting, markov model)
4. Intrusion reaction engine – react to intrusions (by isolating the relevant nodes)

**Level III: Base stations (BS -- assume they will not be compromised)**

1. Application data analysis – analyzing the application data from aggregation points
2. Local detection engine – monitor behavior of the network or individual nodes
3. Intrusion identification – identify the intrusions by analyzing the aggregated data and the network behavior (by chi square distance measure, EWMA forecasting, markov model)
4. Intrusion reaction engine – react to intrusions (by isolating the relevant nodes)

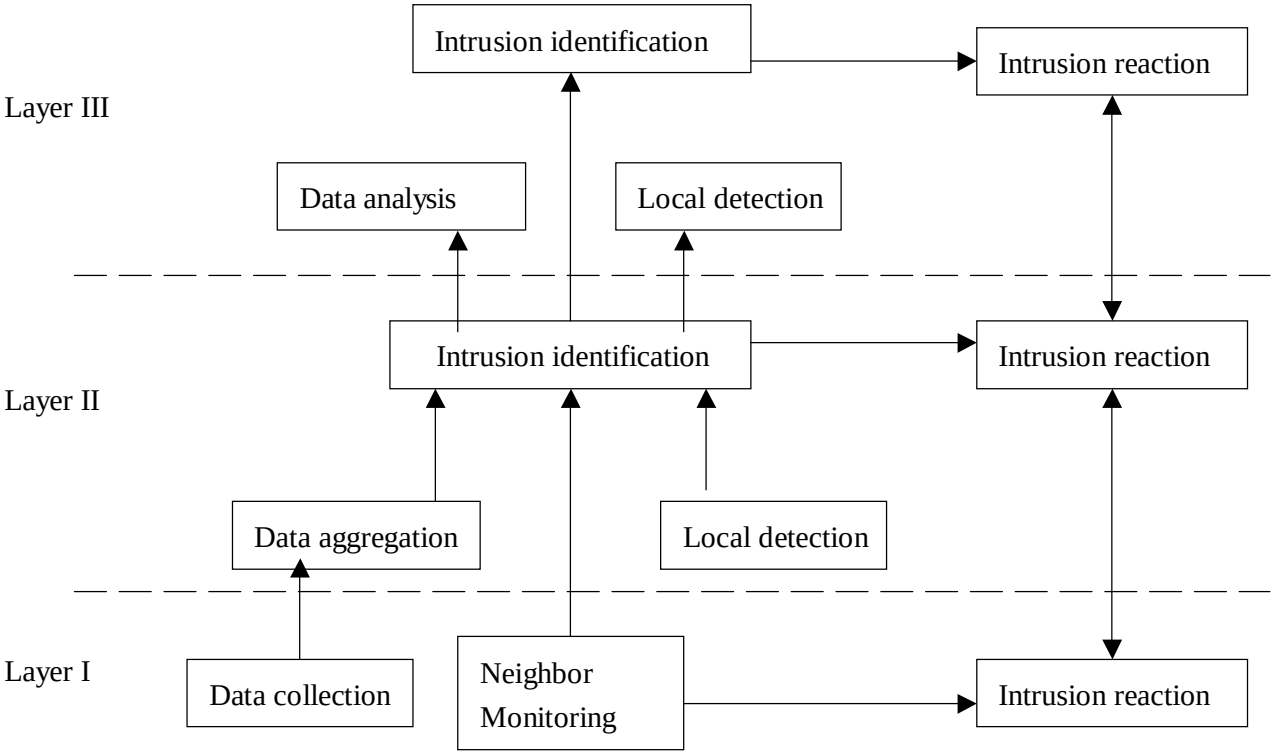



**Figure 4.2 Intrusion Detection in Hierarchical-based Architecture**

## Anomaly Detection

We suggest that different types of attack can be detected in various layer with specific behavior or pattern.

| ***Types of Attacks*** | ***Layer that can detect the attacks*** | ***Layer being attacked*** | ***Pattern/behavior*** |
|---|---|---|---|
| Collection of wrong application data | (a) II<br>(b) III | (a) I<br>(b) II | (a) The set of data being collected by a group of sensor nodes<br>(b) The set of data collected by the aggregation nodes |
| No reply from destination | (a) II<br>(b) III | (a) I<br>(b) II | (a) The pattern of data dissemination from I to II<br>(b) or that from II to |

| | | | |
|---|---|---|---|
| | | | III |
| Selective forwarding | (a) I | (a) I | (a) Listen the network traffic of its neighboring nodes |
| Too many packets | (a) I<br>(b) II<br>(c) III | (a) I<br>(b) I<br>(c) II | (a) Receiving large amount of packets from certain nodes, or by overhearing the traffic of the neighboring nodes<br>(b) |
| False routing information | (a) I<br>(b) II<br>(c) III | (a) I<br>(b) I<br>(c) II | Depends on the routing protocol???<br>(a) ?? depends on routing protocol<br>(b) Not receiving data from certain sensor nodes ,or certain other nodes keep receiving data, or no acknowledgement from the aggregation point<br>(c) Not receiving data from certain aggregation points, or certain other nodes keep receiving data, or no acknowledgement from the base station |

The divided data in the network belong to application or audit data. They are different characteristics:

| | ***Application data*** | ***Audit data*** |
|---|---|---|
| ***II*** | (1) Average the values collected from its group of sensor nodes<br>(2) Then, use chi square distance measure | (1) summarize its own detection result with those from sensor nodes nearby<br>(2) use event frequency from sensor nodes as data vector<br><$SN_{i,1}$, $SN_{i,2}$ ,… , $SN_{i,p}$ ><br>and EWMA forecast with chi square distance measure |
| ***III*** | (1) Use EWMA forecast for each aggregation point<br>(2) Then, use chi square distance measure | (1) summarize its own detection result with those from aggregation points<br>(2) use event frequency from sensor nodes as data vector<br><$AP_{i,1}$, $AP_{i,2}$ ,… , $AP_{i,p}$ ><br>and EWMA forecast with chi square distance measure |

The following approaches describe data vector in different ways:

***Approach 1***

Sequence of the most recent k events <$X_1$, … , $X_T$>

$$\Pr\{X_1,...,X_T\} = q_{x1} * \prod_{t=2}^{T} p_{x-1,x_t} \qquad (1)$$

Normal data as training data to obtain the probability matrix,

$$P = \begin{bmatrix} P_{1.1} & \cdots & \\ P_{1,2} & \cdots & \\ . & \cdots & \\ . & \cdots & \\ P_{s,1} & \cdots & P_{s,s} \end{bmatrix} where P_{i,j} = \Pr\{X_{t+1} = j \mid X_t = i\}$$

The higher the probability in (1), more likely the sequence of states results from

normal activities.

***Approach 2***

The frequency distribution of event types for the recent *i*th audit events <$X_1$, … , $X_T$>,

$$X_{i,k} = h * 1 + (1-h) * X_{i-1,k}$$ , if *i*th event belongs to *k*th event type

$$X_{i,k} = h * 0 + (1-h) * X_{i-1,k}$$ , if *i*th event belongs to *k*th event type

$$\hat{X}_i = \hat{X}_{i-1} + l\,(X_i - \hat{X}_{i-1})$$ (Forecasted value)

(Observed value: $X_i$)

$$X_i^2 = \sum_{k=1}^{p} \frac{(X_{i,k} - \hat{X}_{i,k})^2}{\hat{X}_{i,k}}$$ (Chi Square Distance)

The larger a Chi Square distribution value, the more likely the corresponding audit event is intrusive.

### 4.3.2. Cell-based Approach

Apart from a hierarchical approach, we also think of a cell-based approach for intrusion detection in wireless sensor network (Figure 4.3). Our goal is to perform intrusion detection such as to identify attacks to the network. Also, we plan to apply fault tolerance technology by redundancy to collect and aggregate correct data. The centralized architect is shown in the following figure. It consists of a number of components which are responsible for intrusion detection and intrusion classification. These components, including data collection, routing, neighbor monitoring, data fusion, history, etc will be discussed with details in the following section.

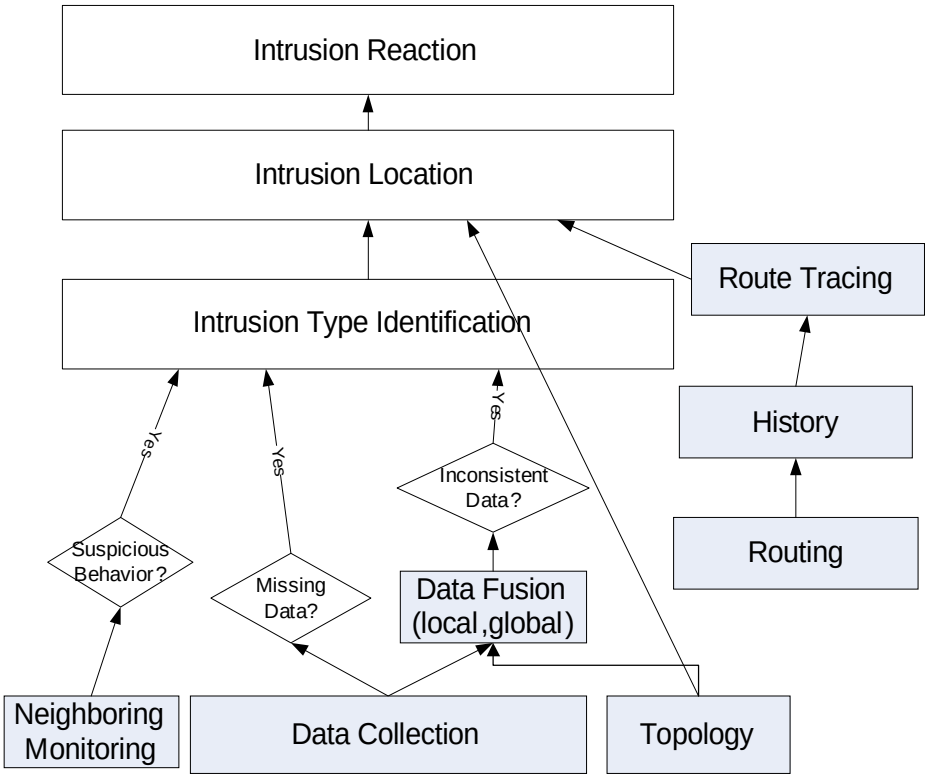



**Figure 4.3 Intrusion Detection in Cell-based Architecture**

We make the following definitions to this architecture.

N: number of BSs in the network

R: transmission range of BS

r: transmission range of sensor node

G: set of good nodes

B: set of bad nodes

d(p->q) : data being sent from p to q

N(p): neighbors of p

- Coverage of a base station:

$$C_i = \{p : |p - BS_i| \le R\}$$

- Number of coverage from base stations:

$$S_k = \{p \in BSi_i \cap BSi_2 ... \cap BSi_k \mid 1 \le i_j \le N\}$$

- p sends data to q successfully (in 1-hop)

$$p \xrightarrow{s} q = |p - q| \le r \wedge p, q \in G$$

- p sends data to q successfully via k hops

$$p \xrightarrow{s}_k q = p_1, ..., p_{k-1} \in G \mid p \xrightarrow{s} p_1 \wedge \left(\wedge_{i=1}^{k-2} p_i \xrightarrow{s} p_{i+1}\right) \wedge p_{k-1} \xrightarrow{s} q \wedge (\forall i, j \in \{1, ..., k\} \mid i \ne j : p_i \ne p_j \wedge p_i \ne p \wedge p_i \ne q)$$

- p fails in sending data from p to q

$$p \xrightarrow{f} q = failure\_on\_transmission\_from\_p\_to\_q$$

| Type of Intrusion | |
|---|---|
| Sinkhole SH(q), HelloFlood HF(q) | |
| Description | A region of nodes will forward packets destined for a BS through an adversary. |
| Formulation | $\exists p \xrightarrow{s}_k q \xrightarrow{s} BS_i \mid \angle p_l BS_i p_r \ge q_m \wedge p > m$ |
| Wormhole WH(q) | |
| Description | An adversary tunnels messages received in one part of the network over a low latency link and replays them in a different part. |
| Formulation | $\exists p \xrightarrow{s}_k q_1 \xrightarrow{s} q_2 \xrightarrow{s} BS_i \mid \angle p_l BS_i p_r \ge q_m \wedge p > m$ |
| Missing Data MD(Ci) | |
| Description | Missing data from p to BSi |
| Formulation | $\exists p \xrightarrow{f} BS_i \mid p \in C_i$ |
| Wrong Data (local) $WD_L(p)$ | |
| Description | Inconsistent data among p and its neighboring nodes |

| | |
|---|---|
| Formulation | $\left\lvert d(p \xrightarrow{w} BS_i) - d(N(p) \xrightarrow{s} BS_i) \right\rvert > d_m$ |
| Wrong Data (global) $WD_G(p)$ | |
| Description | Inconsistent data of p received by BSi and other BSs |
| Formulation | $\forall i : p \in C_i \mid \exists \neq d(p \longrightarrow BS_i)$ |
| Selective Forwaring / Interference | |
| Description | Sensor p does not forward data to its neighboring nodes |
| Formulation | $p \xrightarrow{f} \forall N(p)$ |

## 4.4. Detection Components

### 4.4.1. Data Fusion

We separate data fusion to be local and global data fusion Local data fusion compares the data among a node p and its neighboring nodes. Global data fusion compares the data from node p with its base stations.

A number of data fusion techniques can be applied. Marzullo's method yields the smallest sensor fusion interval guaranteed to contain the correct value [44]. It combines the intervals of sensors by computing local averages. We consider n sensors. Each sensor yields an estimated closed interval of T, denoted as $I_i = [a_i, b_i]$, $a_i <= b_i$, i<=n.

Suppose the fusion center receives a list $\{I_1, I_2, \ldots, I_n\}$ of sensor interval estimates. At most f out of the n (0<=f<n) interval estimates are assumed to be faulty. The fusion center takes the list $\{I_1, I_2, \ldots, I_n\}$ as an input and outputs a closed interval, I = [a,b], a<=b, representing the final estimate of T.

Let

$$I_{\{i_1,i_2,\ldots,i_{(n-f)}\}} = \bigcap_{j=1}^{n-f} I_{i_j} \neq \emptyset$$

and the list

$$\{I_{\{i_1,i_2,\ldots,i_{(n-f)}\}} : \{i_1, i_2, \ldots, i_{(n-f)}\} \subseteq \{1, 2, \ldots, n\}\}$$

represent all the nonempty intersections of (n-f) closed intervals in $\{I_1, I_2, \ldots, I_n\}$.

The final interval I = [1,b] given by Marzullo's method is that which is the shortest close interval containing the union of $I_{\{i_1,i_2,\ldots,i_{(n-f)}\}}$
Marzullo's method is given a functional representation, called Marzullo function, denoted by:

$$I = M(\{I_1, I_2, \ldots, I_n\}) = \overline{\bigcup I_{\{i_1,i_2,\ldots,i_{(n-f)}\}}}$$

Example:
{[8,11], [9,12], [10,13]}

| Fused interval outputs | | | | |
|---|---|---|---|---|
| f | 0 | 1 | 2 | 3 |
| I | [10,11] | [9,12] | [9,13] | NULL |

We can apply this technique to our intrusion detection system (data fusion component) for identifying suspicious nodes by local data fusion and global data fusion for overlapping areas. The malicious nodes may have great chance to provide data inconsistent with the majority.

Apart from the Marzullo's method, the Schimd-Schossmaier function proposed in Cho et al. [45] is a fault-tolerant interval intersection function with the same worst-case behavior as the Marzullo function but satisfying the Lipschitz condition. Li [46] also proposed a new fault-tolerant interval integration function based on the Dempster-Shafer theory evidence. Cho et al. recently proposed a new interval integration method that further narrows the region containing the rue value of the state measured by the sensors. Another important formulation of the data fusion deals with combining information from multiple sensors to obtain results better than the best or best subset of sensors [47, 48, 49].

### 4.4.2. Localization (Network Topology)

Localization is a technique for finding the locations on nodes. It works with the "Route tracing" component to figure out the position of the attackers. It provides the base station a view on the network topology.

The most well-known positioning system for outdoor environments is the global positioning system (GPS) [50]. In addition to the GPS system, positioning can also be done using some wireless networking infrastructure. Several location estimation models, such as angle of arrival (AoA), time of arrival (ToA), and received signal strength (RSS) are widely used in cellular networks and wireless sensor networks. The RSSI technique is more suitable for WSN. It measures the

attenuation in radio signal strength between sender and receiver. The power of the radio signal falls off exponentially with distance, and the receiver can measure this attenuation in order to estimate the distance to the sender. These techniques can estimate the distance between two devices. If an object known its distances to multiple devices at known locations, one may estimate its location. The common techniques are trilateraion and multilateraion.

Trilateration is a well-known technique in which the positioning system has a number of beacons as known locations. These beacons can transmit signals so that other devices can determine their distances to these beacons based on received signals. If a device m can hear at least three beacons (a1, a2, and a3), its location can be estimated (Figure 4.4).

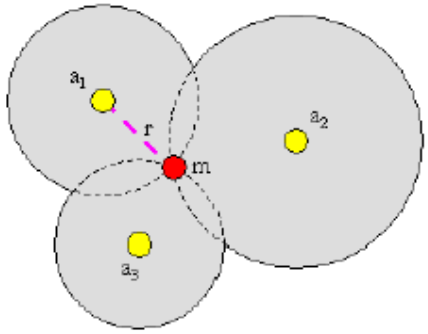



**Figure 4.4. Trilateration Method**

The trilateration method has its limitation in that at least three beacons are needed to determine a device's location. In a sensor network with nodes randomly deployed, this is hard to achieve. Several multilateraion methods are proposed to relieve this limitation [51]. For the estimated distances (di) and known positions (xi, yi) of the anchors we derive the following equations:

$$(x_1 - x)^2 + (y_1 - y)^2 = d_1^2,$$
$$\vdots$$
$$(x_n - x)^2 + (y_n - y)^2 = d_n^2,$$

subtracting the last equation from the first n-1 equations:

$$x_1^2 - x_n^2 - 2(x_1 - x_n)x + y_1^2 - y_n^2 - 2(y_1 - y_n)y = d_1^2 - d_n^2,$$
$$\vdots$$
$$x_{n-1}^2 - x_n^2 - 2(x_{n-1} - x_n)x + y_{n-1}^2 - y_n^2 - 2(y_{n-1} - y_n)y = d_{n-1}^2 - d_n^2.$$

Rearrange to form Ax = b where

$$A = \begin{bmatrix} 2(x_1 - x_n) & 2(y_1 - y_n) \\ \vdots & \vdots \\ 2(x_{n-1} - x_n) & 2(y_{n-1} - y_n) \end{bmatrix},$$

$$b = \begin{bmatrix} x_1^2 - x_n^2 + y_1^2 - y_n^2 + d_n^2 - d_1^2 \\ \vdots \\ x_{n-1}^2 - x_n^2 + y_{n-1}^2 - y_n^2 + d_n^2 - d_{n-1}^2 \end{bmatrix}.$$

This minimum square error (MMSE) problem can be solved by QR decomposition / choleski factorization / using a standard least-squares approach: $\hat{x} = (A^{\mathrm{T}}A)^{-1}\ A^{\mathrm{T}}b$

### 4.4.3. Route Tracing

The route tracing component is to trace the route in data propagation. It helps to trace the identification of the attack. The ID of immediate nodes can be stored in the packet to avoid spoofing with key signature.

### 4.4.4. History

The history on the network behavior is a piece of information helpful for future intrusion detection. The history component stores the past routes, topology, data collected, and the intrusion detected. It keeps updating attack pattern and compares with the past records with current behavior.

### 4.4.5. Neighbor Monitoring

Neighbor monitoring component can detect misbehavior of neighboring nodes, such as selective forwarding and interference. It is a technique proposed in wireless ad hoc network and is feasible for wireless sensor network. This technology enables local inspection of node behaviors.

**Intrusion Detection Components**

| Components\Attack Types | | I | II | III | IV | V |
|---|---|---|---|---|---|---|
| Neighbor Monitoring | BS | **Dominating intermediate node** | **Dominating intermediate node** | **Selective forwarding** | --- | --- |
| | Sensor | --- | --- | **Selective forwarding** | --- | **Interference (jamming with neighbors)** |
| Data Comparison | Global | (may have missing or inconsistent data) | (may have missing or inconsistent data) | **Missing data** | **Inconsistent data (IVa – malicious sensor or intermediate nodes)** | Missing data |
| | Local | (may have missing or inconsistent data) | (may have missing or inconsistent data) | **Missing data** | **Inconsistent data (IVb – sensor failure or being compromised)** | Missing data |
| Routing (with topology info.) | BS | **a region of nodes forward packet through the same adversary** | **An adversary tunnels messages and replays them in a different part** | --- | --- | --- |

Attack Types: I - Sinkhole, Hello Flood II – Wormhole III – Missing Data IV – Wrong Data V - Interference

## 4.5. Grid-based Analysis

### Grid-based deployment

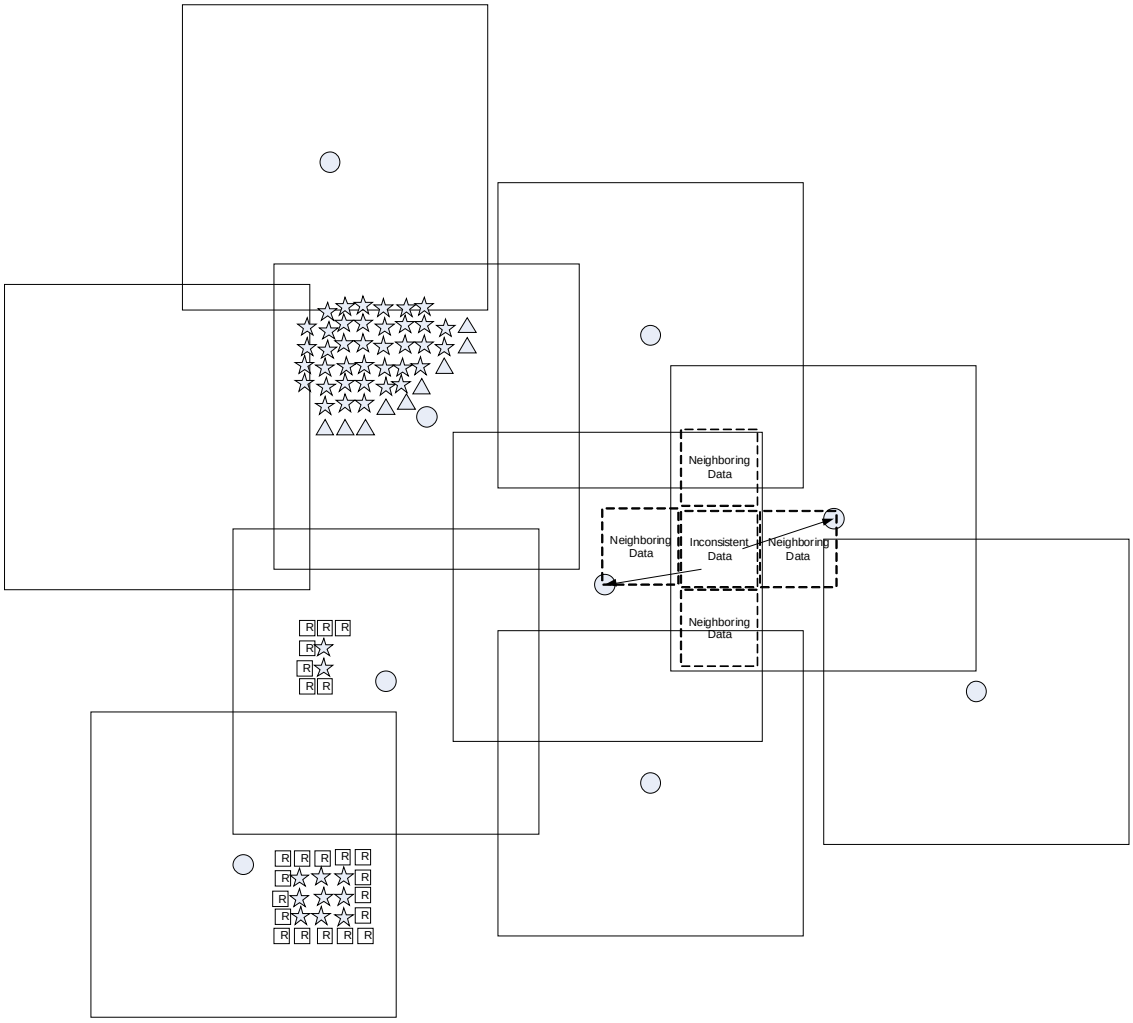



**Figure 4.5 Grid-based Analysis**

Sensor Node:

**(1) Selective Forwarding**

(Faulty node) F: Sa,b, where a = a_1 … a_n and b = b_1 … b_n

(Surrounding node) S: Si,j where (a-1<=i<=a+1, b-1<=j<=b+1) ^ (Si,j !=Sa,b) ^ (|Sa,b-Bx,y|<=|Si,j-Bx,y|)

(reporting node) R: $R \subset S$

No. of surrounding nodes to F (estimation) N(S)

= (a_n – a_1 + 3) + (b_n - b_1 + 3)

Accuracy = N(R) / N(S)
Performance = no. of messages to BS = N(R)

**(2) Interferences**

Faulty / Affected nodes F: Sa,b , where a = a_1 … a_n and b = b_1 … b_n

| | | | | |
|---|---|---|---|---|
| R | R | R | R | R |
| R | F | F | F | R |
| R | F | F | F | R |
| R | F | F | F | R |
| R | R | R | R | R |

(Surrounding node) S: Si,j where (a-1<=i<=a+1, b-1<=j<=b+1) ^ (Si,j !=Sa,b)

(how about irregular shape?)

| | | | | |
|---|---|---|---|---|
| | | R | | |
| R | F | F | F | R |
| | F | F | F | |
| R | F | F | F | |
| | | | R | |

(reporting nodes) R: Rc,d where c = c_min … c_max and d = d_min … d_max

Estimate the affected area = area surrounded by the reporting nodes
= (c_max – c_min + 1) * (d_max – d_min + 1)

Accuracy = N(R) / N(S)
Performance = no. of messages to BS = N(R)

Base Station:

**(1) Missing data**

F: MD(Sa,b) where a = a_1 … a_n and b = b_1 … b_n

No. of surrounding nodes N(S) = (a_n – a_1 + 3) * 2 + (b_n – b_1 + 1) * 2

No. of suspicious nodes N(P) = N(S) – no. of nodes overlapped by another cluster
= N(P) / 2  (estimation by assume overlapped in two sides)
Carry **route tracing** by the suspicious nodes

(i) P sends messages to the faulty neighbors to know their next hop and no. of hops to the BS
(ii) BS compares the received results.
(iii) Using the topology information, it narrows down the final set of suspicious nodes FP.

(graph / tree representation  related?)

Performance
= Cost (no. of messages from P to BS) + Cost(comparison and route tracing)

False positive alarm rate
= N(FP) – N(F) / N(P)

**(2) Inconsistent data**

F: WD(Sa,b) where  a = a_1 … a_n  and  b = b_1 … b_n

Global data fusion
Accuracy (Prob. for discovering the data inconsistency)
= Area (CV>1) / Total Area

Local data fusion
Accuracy (Prob. for discovering the data inconsistency with neighboring data)
= n * P(local) ,  where  n is the no. of neighboring data areas and
P(local) = Prob. for discovering the data inconsistency

Overall accuracy
= w1 * Accuracy (Global) + w2 * Accuracy (local)
= w1 * Area (C>1) / Total Area + w2 * n * P(local)

Performance
=cost (global comparison) + cost (local comparison)
= CV * Area (CV) + n * Area (neighbor data)

# 5. Tracing Network Attacks

## 5.1. Related Work

In traditional network, IP traceback is a common approach for tracing network attacks. Although access-control technologies, such as firewalls, are commonly used to prevent network attacks, they cannot prevent some specific attacks, including TCP SYN flooding. Consequently, more companies are deploying intrusion detection systems (IDS). IDSs detect network attacks; however, they do not let us identify the attack source. This is especially problematic with denial-of-service attacks, for example, thus can remain hidden. Several efforts are in progress to develop source-identification technologies to trace packets even when an attacker forges its IP address [52].

There are two approaches to the problem of determining the route of a packet flow: one can audit the flow as it traverses the network (proactive tracing), or one can attempt to infer the route based upon its impact on the state of the network (reactive routing). Both approaches become increasingly difficult as the size of the flow decreases, but the latter becomes infeasible when flow sizes approach a single packet because small flows generally have no measurable impact on the network state [53].

Two proactive tracing methods which are packet marking and messaging have been proposed. In packet marking (Figure 5.1), a packet travel through the network, they gather and store information about the routers they traverse. A router can write its identifier probabilities in packet's IP header identification field [54]. In Messaging approaches (Figure 5.2), routers create and send messages containing packet information to the packet's destination [55].

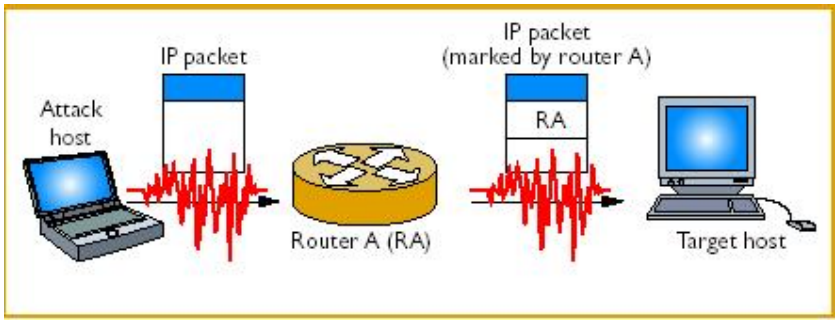



**Figure 5.1. Packet marking**

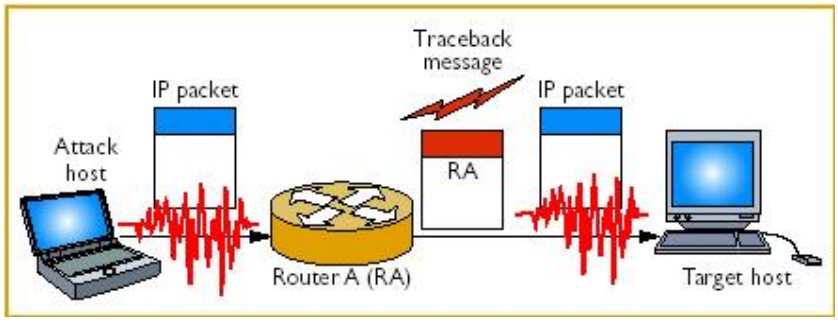



**Figure 5.2 ICMP traceback message**

Reactive tracing starts tracing after an attack is detected. Most of the methods trace the attack path from the target back to its origin. Hop-by-hop tracing (Figure 5.3) starts at the router nearest the target host and follows the attack packet back to its source, hop by hop, during the attack. To decrease the number of hops required for tracing with an overlay network, one approach builds an overlay network by establishing IP tunnels between edge routers and special tracing routers then reroutes IP packets to the tracking routers via the tunnels [56]. Some other reactive tracing techniques in traditional network includes IPsec authentication [57] and traffic pattern matching [58].

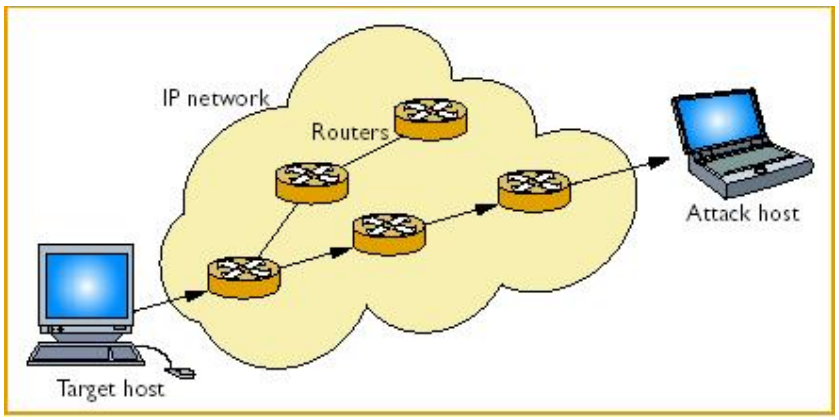



**Figure 5.3 Hop-by-hop tracing**

There are not many approaches proposed to trace the attackers for wireless sensor networks. J. Staddon et. al proposes an approach for tracing of failed nodes in sensor networks [59]. It defines “dead” node as a node ceases sending or routing measurement because it died, and “silent” node as a node is not sending measurements and its status cannot be determined. It proposes algorithms for finding new routes for the “silent” nodes. In Figure 5.4, it shows only one of the sensors is dead (indicated in black) and 7 nodes have ceased sending measurements because of the routing topology. A single route update allows the base station to trace the dead node and begin receiving measurements again.

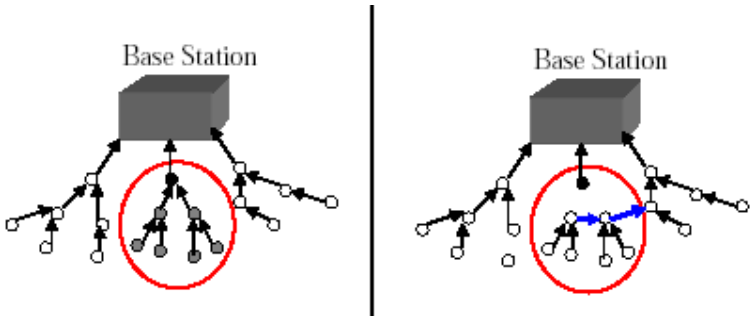



**Figure 5.4 Single Route Update**

First, the silent nodes are subdivided into disjoint sets and a new route is broadcast for each set. Then, it uses two subdividing algorithms, Subdividing by Best R Neighbors and Subdividing by Best Base Station Neighbors, for faster tracing. It also suggests an adaptive Subdivision-based tracing. In Figure 5.5, when 10 nodes become silent the 2 nodes adjacent to the base station (marked in black) are known to be dead. Subdividing the remaining nodes once by best R neighbors requires 2 route updates, each of length 5. Subdividing by best base station neighbors requires two updates (one with a large hop to R) each of length 3.

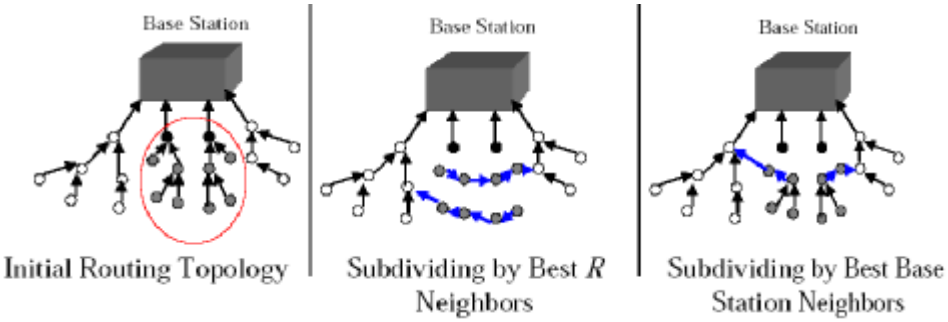



**Figure 5.5 Subdivision-based Tracing**

## 5.2. Tracing Sinkhole Attack on WSN

### 5.2.1. Problem Statement

**Description of sinkhole**

To launch a sinkhole attack, an adversary lures nearly all traffic from a particular area through a compromised node. The adversary usually attracts network traffic by advertising itself as having the shortest path to the base station. For example, a laptop class adversary provides a single-hop link to base station. It can then tamper packets originated from any nodes in the area. Another common kind of attacks is selective forwarding. By ensuring that all traffic in an area flows through a compromised node, an adversary can selectively forward any packets routing through it to the base station (Figure 5.6).

Wireless sensor networks are vulnerable to sinkhole attacks as they have special communication pattern. All sensors are sending packets to the same destination which is the base station. Sensors in the same area are affected even only one compromised node is providing a high quality route to the base station. Sinkhole attack is difficult to detect because simply using user authentication and signed routing information cannot prevent compromised nodes from generating signed routing packet with wrong information.

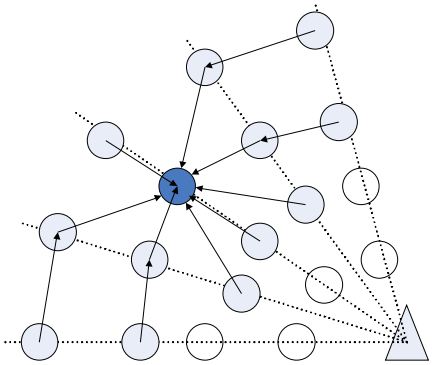


**Figure 5.6 Example of Sinkhole attack**

**Problem Formulation**

Consider a wireless sensor network randomly deployed with any sensor i having a communication range r. Such model can be modeled as a geometric random graph. Let G(V,r) denote the undirected graph with vertex set V of randomly deployed nodes, and with undirected edges connecting pairs of vertices (i,j) with ||i-j||<=r.

**Definition:**

n_i(x) denotes the neighbors of base station i:

n_i(x) = 1 if ||BSi-x|| <= r or = 0 if ||BSi-x|| > r (1)

**Theorem 1:** Given a geometric random graph G(V,N) defined as in (1), and an

arbitrary logical graph G'(V,N'), a mechanism D(G, G') will detect an sinkhole iff the neighbors of base station is not a subset of the set of base station neighbors of the G(V,N), i.e. $N_{G'} ! \subseteq N_G$.

**Proof:** Assume that D(G, G') detects the sinkhole attack. Let $B_X$ denote the base station neighborhood of graph X. If $N_{G'} !\subseteq N_G$, there exists a node x for which: $B_G$ (x) = 0 and $B_{G'}$ (x) = 1. For such node, $n_i(x) = 1$, with ||BSi-x|| > r, violating the communication range constraint. Hence, if D (G, G') detects a sinkhole attack, it follows that: $N_{G'} ! \subseteq N_G$.

The converse follows immediately. If $N_{G'} \subseteq N_G$, then $B_{G'}(x) <= B_G (x), \forall x \in V$ . Hence, there is no $n'_i(x)$ belongs to $B_{G'}$ such that $n'_i(x) = 1$, ||BSi-x|| > r, and hence, the graph G' is void of any sinkhole attack.

We believe that the sinkhole detection mechanism D(G, G') requires the knowledge of the locations of senor nodes and the flows of network traffic. However, sensor nodes normally have limited battery, memory, and computational power, which forbid them to have a global view of the network. Their short transmission range also limits their communication with only direct neighbors. On the other hand, a base station is a more powerful device with sufficient memory and computational power. It is also the destination of most of the network traffic. We view the base station as a management centre for the detection and location of the sinkhole attacks.

### 5.2.2. Network Model and Notations

We assume that the network nodes are randomly deployed within a specific region. We also assume that the adversary who launches the sinkhole attack attracts network traffic by advertising a direct route to the base station. (This assumption can be generalized, so as to model similar problems, like the wormhole attacks.)

The network considered comprises a set N of sensor nodes and a base station. Sensor nodes are associated with unique ids. They collect and send application data to the base station periodically by routing packets hop by hop. Only the base station maintains a global view of the location of nodes by some localization mechanisms. It broadcast authenticated beacons to all the nodes in the network periodically. This prevents nodes from recognizing the base stations wrongly. We use the following notations in this paper:

The base station is denoted by BS and the sensor nodes are denoted by Si. The set of neighboring nodes of BS are denoted by N(BS), where ||x – BS||<=r for x belongs to N(BS). The compromised node in the sinkhole attack is denoted by Cx. It may provide a high quality route by transmitting with enough power to reach the base station in a single hop. Then, neighboring nodes of Cx will forward packets destined for the base station through the adversary, and also propagate the attractiveness of the

route to its neighbors. Eventually, a large number of surrounding nodes of Cx are susceptible to the sinkhole attack. Cx can perform selective forwarding or tampering at this moment. The nodes that are affected in this attack are denoted by Tx.

### 5.2.3. Sinkhole Attack Detection

In this section, we present several mechanisms to detect sinkhole attacks. Sinkhole can be detected by determining whether the packets reaching the base station are forwarded by real neighbors, or some compromised nodes. Furthermore, it can be detected by observing various network misbehaviors, like selecting forwarding or spoofing of application data. Then, we can figure out the nodes affected by the sinkhole and determine the attack region.

**Neighbors of the Base Station**

A sinkhole is usually a compromised node which provides an extreme high quality route to the base station. It is actually not a valid neighbor of the base station as its physical location is out of the normal transmission range r. To detect whether the packets arriving the base station come from valid neighboring nodes, the base station will periodically generating an independent key to each of its neighboring nodes. The neighboring nodes of the base station can then attach a key signature to the packets that it forwards to the base station.

**Definition**: For a base station BS, we define its neighborhood N(BS) as: N(BS) = {i: ||BS-i||<=r}. We assign a unique key Ki to be the secret key between a node i and the base station. Hence, by definition, all one-hop neighbors of the base station possess a secret key with the base station. We follow the convention that messages from node i to the base station are attached with a key signature by Ki. Hence, a link between nodes i and base station iff, $i \in N(BS)$.

**Theorem 2**: Given Ki, N(BS), $\forall i \in V$, where V is the set of vertices defined by network nodes, and an arbitrary local random graph G'(V, r), the neighbor matrix of base station is defined by:

$N_{G'}(i) = 1$ if i process Ki or

$= 0$ if Else (2)

yields the desired sinkhole-free graph G'(V,r) such that $N_{G'} \subseteq N_G$, where G(V,r) is the geometric random graph defined in (1).

**Proof**: By the definition of $N_{G'}$, there exists a link between node i and the base station if they share one secret key. According to the establishment of secret keys at the base station, in turn implies that i, BS satify (1), which defines the neighborhood of base station in the geometric random graph G(V,r). Hence, N(i)=1, iff ||BS-i||<=r. According to theorem 1, if a detection mechanism D(G,G') observes that the base

station only receives packets from its neighboring nodes, such that $N_{G'} \subseteq N_G$, then G' is a sinkhole-free graph.

The packets from neighboring nodes to the base stations are attached with a signature of the corresponding neighboring node. This approach is more lightweight than encrypting every message with a secret key. Also, it applies only to the neighboring nodes of the base station, instead of every node in the network. The base station renews the keys of its neighboring nodes periodically, in order to ensure no keys are compromised. The base station also maintains up-to-date information on its neighboring nodes. However, signatures of neighboring nodes only help to detect the presence of sinkhole, but do not provide any further information on its approximate location.

**Attack Region Detection**

Apart from detecting sinkhole by key signatures of neighboring nodes, the attacks can be detected by analyzing the application data received at the base station. A common kind of violations is selective forwarding. In many sensor network applications, sensor nodes are responsible for collecting local data and sending them to the base station. If a sinkhole performs selective forwarding, the base station may discover them by observing missing application data from some sensor nodes. Since the base station will aggregate the application data from sensor nodes, it can figure out the list of sensor nodes with missing data without difficulties. These sensors are affected by the sinkhole attack. They may change their routes in a way that they forward the packets to the compromised node. Indeed, the base station can detect the node affected by comparing the application data in similar region. This is because the sinkhole may spoof packets which contain application data. This violation leads to data inconsistency even for the sensors in the same region.

After identifying the list of sensors affected, the base station can estimate the attack region. This region contains the sinkhole and the sensor nodes routing towards the sinkhole. The compromised node usually advertises an attractive route to the base station, so the sensor nodes within a few hop counts will be attracted to the routes provided by the compromised node. Since we assumed that the base station has a global view of the locations of the sensor nodes, it can then circle the potential attack region. In Figure 5.7, the nodes in dark are the nodes which were found to have missing data or inconsistent data. The circle shows the estimated result on the attack region. Basically, the circled area contains all the affected nodes. According to the routing pattern, the compromised node has high chance to be located at the center of the detected area.

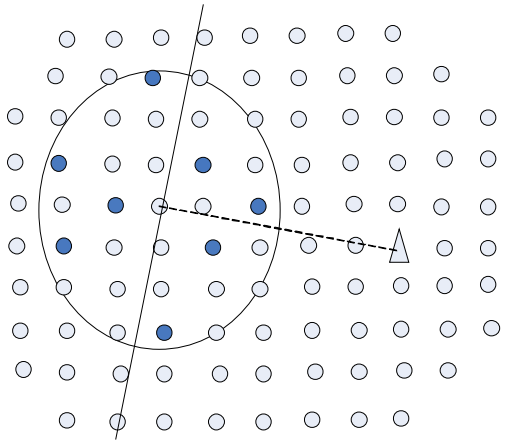


**Figure 5.7 Estimate the Attack Region**

### 5.2.4. Sinkhole Attack Location

Simply detecting the attack region is not effective enough to defend against the sinkhole attack. It is better to locate the attacker and isolate it from the network. In this section, we will present our approach to identify the compromised node in a sinkhole attack by diffusion.

**Sinkhole Identification**

The potential position of the compromised node in a sinkhole attack is estimated by circling the affected nodes in the network. The compromised node is actually the destination point of the network traffic in that circled area. Figure 5.8 shows the flows of network traffic in the detected attack area. The grey node represents the compromised node which will be identified. We perform diffusion from the base station to the nodes inside the detected attack area. These are the nodes being affected in the sinkhole attack. The diffusion aims at probing the affected nodes. They are requested to provide their routing information, including their next hop and number of hop counts to the base station. Whenever they receive the probing messages, they must reply to the base station with the reverse path. The affected nodes should not reply the messages with the shortest path as their shortest path may have already been mislead by the compromised node. By analyzing the routing information provided, the base station can identify the compromised node.

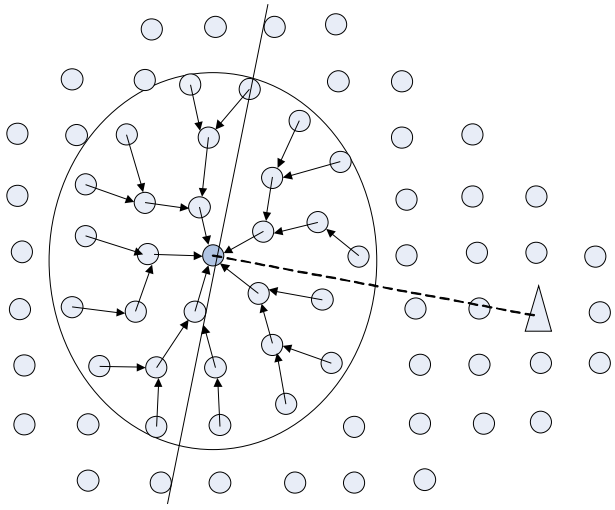


**Figure 5.8 Network Traffic inside the Detected Attack Area**

**Protocol: Probing by diffusion**

The base station BS starts probing by diffusing request message to some of its neighboring nodes. The selected neighboring nodes N(x) are physically closer to the attack region than the unselected neighboring nodes. The base station signs the request message with its secret key. When a probing message is received by the neighboring nodes x, they forward the message to their neighboring nodes y. When nodes receive the request message, they will reply with a message containing their node ID, ID of next hop, and number of hop count to the base station. They will send this reply to the base station by forwarding it in reverse path. Also, they will forward the request message further to their neighboring nodes. This process of replying and forwarding will repeat until it reaches the boundary of the attack region. We can specify the nodes at the boundary, so they will not forward the request message to their neighbors further more. Figure 5.9 shows the reverse paths for passing routing information from the affected nodes to the base station.

The following shows the protocol for probing the affected node in the attack region:

(1) At the beginning of a suspicious sinkhole attack occurs

BS -> N(x): <probing, BSi>

(2) When a probing message is received from N(x)

x -> y (neighbors of x): <probing, x, BSi>

(3) When node y receives a probing message

y -> x: <y, shortest_next_hop, shortest_hop_count> (routing information to BS)

y -> y' (neighbors of y): <probing, y, BSi>

(4) The processes (3) repeats until the request messages reach the boundary of the attack area

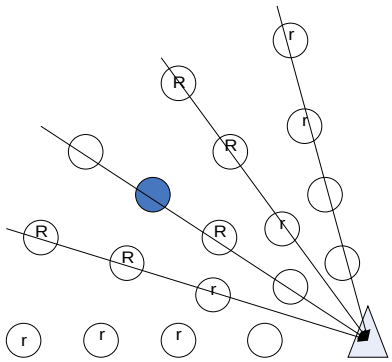



**Figure 5.9 Reverse Path in Probing**

**Analyzing the routing information**

After collecting the routing information from the affected nodes, the base station analyses the received information and locates the sinkhole. The sinkhole is the destination of the network flows. It is normally the compromised node located at the centre of the attack area. It will be identified if the base station knows the network flow in the attack region. The base station holds the information of one-hop routing among the affected nodes and their neighboring nodes after the probing protocol. This information are represented by a number of directed edges, say i **à** j (h), where i denotes an affected node, j denotes the next hop of i, and h denotes the number of hop counts from i to the base station.

We represent the network flow in the attack area by a directed graph G(V,E). The graph G(V,E) consists of an array Adj of |V| lists, one for each vertex in V. For each u∈ V, the adjacency list Adj[u] contains all the vertices v such that there is an edge e(u,v)∈ E. We assume that there is exactly one sinkhole in the attack area in ideal case, but this assumption will be relaxed in the next section. Also, the information provided by the affected nodes is trustworthy. The sinkhole identification procedure below identifies the level of nodes in the graph. It also locates the root node which is the sinkhole at the end of the process. Firstly, the base station will search for nodes u which does not have any incoming edges. These nodes u will be added into the set $S_0$. Then, the base station iterates through the nodes in $S_0$ and finds their next hop v. Nodes v is then added into the set $S_1$. This process repeats until the set $S_n$ contains only one element which is the sinkhole of the attack.

```
1.  S0 = ϕ
2.  n = 0
3.  for each v∈ S
4.      if v has no incoming edge
5.          S0 = S0 + {v}
6.  end for
7.  while (|Sn| >1)
8.      n = n + 1
9.      Sn = ϕ
10.     for each u∈ Sn
11.         if e(u,v)
12.             Sn = Sn + {v}
13.     end for
14. end while
15. Root = Sn
```

**Algorithm 5.1 Identifying the sinkhole**

### 5.2.5. Secure Sinkhole Identification Algorithm

In the above sinkhole identification algorithm (Algorithm 5.1), we assumed all the routing information provided by the affected nodes is correct. However, it is possible that some of the affected nodes are compromised or even collude with each others. They may drop some of the reply packets (Figure 5.10a), such that some routing information are missing. On the other hand, they may provide some incorrect routing information, like wrong edges (Figure 5.10b and 5.10c), and wrong hop counts (Figure 5.10d). Secret keys can be established between individual nodes and the base station to secure messages passing. This avoid s message spoofing by malicious intermediate nodes on the routes.

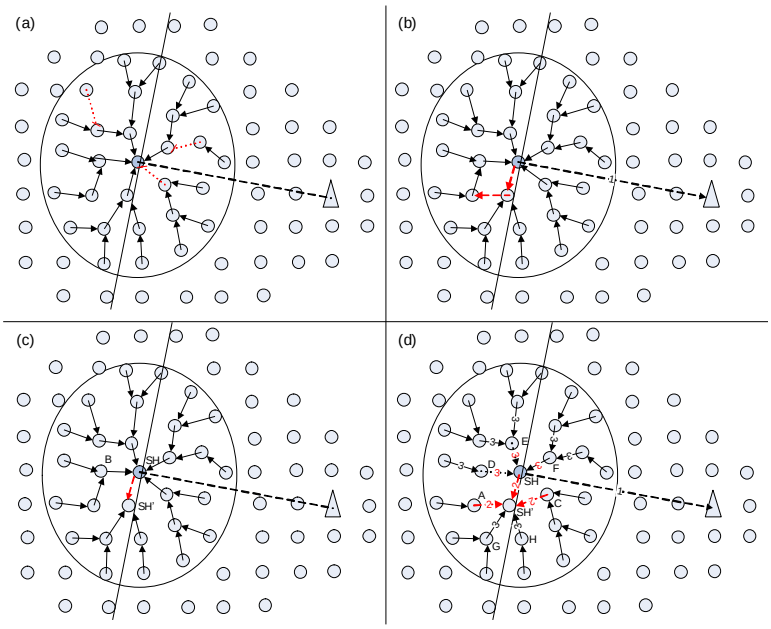



Figure 5.10 Attack area with colluding nodes (a) missing information (b) cycles (c) misleading sinkhole (d) identification sinkhole using hop counts

After the base station probes the affected nodes, reply messages from the affected nodes may be missing to the base station. It is because some malicious nodes may not send the reply message or some reply messages are dropped on the routes. Figure 5.10a shows an example with missing routing information after probing. In this case, more than one tree of network flow will be constructed. The base station must discover all the roots as they are the possible sinkhole in the attack. Algorithm 5.2 shows the extended algorithm which can find all the roots in the affected area. It also avoids the process from entering infinite loops if there exist cycles in the network flow (Figure 5.10b). If there exists more than one roots in the attack area. Algorithm 5.3 calculates the number of subsequent nodes belong to different graphs. The real sinkhole is believed to he be root which attract most network traffic in the attack area.

```
1.  R = ϕ
2.  for each v∈ S
3.      if v has no incoming edge
4.          R = R + findR(v)
5.  end for
6.
7.  findR(node u)
8.      R = ϕ
9.      if u is not yet visited
10.         mark u is visited
11.     else
12.         return (
13.     if u has no outgoing edge
14.         return {u}
15.     for each e(u,v)
16.         R = R + findR(v)
17.     end for
18. end while
```

**Algorithm 5.2 Identifying Multiple Roots**

```
1.  num = numNodeR(R)
2.
3.  numNodeR(node r)
4.      n = 1
5.      if r is not yet visited
6.          mark r is visited
7.      else
8.          return 0
9.      for each e(r, c)
10.         n = n + numNodeR(c)
11.     end for
12.     return n
13. end numNodeR
```

**Algorithm 5.3 Finding Total Number of Nodes in a Tree *T*, given Root *R***

Apart from the above dishonest behavior, a sinkhole may interference the identification process by providing wrong routing information itself (Figure 5.10c). It can provide an outing edge to another node and attempt to convince the base station that its next hop (SH') is the sinkhole. We can identify this attack by checking some unreasonable route. For example, in Figure 5.10c, the route A->B->SH->SH' is less realistic than the route A -> SH'. It is because ||A – SH'|| < r, so there exist e(A, SH') in the network. In more hostile environment, there exist some colluding nodes which provide wrong routing information. They hide the real sinkhole by claiming that they are sending to a false destination intelligently. All of them may provide wrong edges pointing to the false sinkhole SH'. It is very difficult to identify SH as the sinkhole. For example, nodes A and C collude and provide an outgoing route to the false sinkhole SH' in Figure 5.10d. Under this case, a more practical approach is to consider also the hop counts from nodes to the base station. In an ideal case, a sinkhole is a node with no outgoing edge. Its neighboring nodes have same and minimum number of hops to the base station. In extreme case, all nodes surrounding the real sinkhole may provide wrong number of hop counts to the base station (Figure 5.10d). However, we can detect the inconsistency on the number of hop counts. Nodes D, E, and F have same number of hop counts in their incoming and outgoing edges. The false sinkhole SH' has incoming edges with different number of hop counts. By spotting these wrong routing phenomena, we can isolate the sinkhole and other suspicious nodes.

### 5.2.6. Sinkhole Identification Algorithm with Hop Count Information

Nodes store not only next hop, but also number of hop counts to the destinations in their routing protocols. Number of hop counts provides additional routing information for identifying the sinkhole. By considering both types of routing information, we can discover data inconsistency and identify the sinkhole attack more accurately. We summarize a few rules of consistent data:

1. All incoming edges have same number of hop counts to destination.
2. If e(a,b) claims no. of hop count n, then e(b,c) will provide hop count equal to n-1.
3. Sinkhole does not have outgoing edges.
4. Incoming edges to sinkhole should provide minimum number of hop counts to base station.

```
Struct root_info
{
        int nodeID;
        double value1, value2;
        Struct root_info *next;
}
typedef struct root_info info;

TotalNum = total no. of nodes in the attack area
info *result;

for each root r
        result = CheckRoot(r);
end for
for each root_info in the result
        if max(value1, value2) > threshold
                nodeID is suspicious
        end if
end for

*info Checkroot(Node node)
        info rootInfo;
        Node node2;
        if (numNode(node) >= TotalNum/2)
                rootInfo.value1 = formula(node);
                if contradictions on routing information exists
                        node2 = Correct(node);
                        rootInfo.value2 = formula(node2);
                        for each precedent node Pi
                                rootInfo.next = Checkroot(Pi);
                        end for
                end if
        end if
        return rootInfo;
end Checkroot

double Formula (node)
        double value;
        value = w1*numNode(node)/TotalNum
                + w2*num_neighbor_with min_hop_count/TotalNeighbor
                - w3*Boolean_outgoing_edge
end Formula

node2 Correct(node1)
        for each precedent node Pi
                if consistent incoming hop counts
                        outgoing_hop_count = incoming_hop_count – 1;
                else
                        if majority incoming hop counts exists
                                outgoing_hop_count = majority incoming hop count – 1;
                        else
                                return null;
                end if
        end for
        return updatedNode;
end Correct
```

**Algorithm 5.4 Identifying Suspicious Nodes with Information on Hop Counts**

Figure 5.11 shows the flow chart for identifying suspicious nodes with the information of next hop and number of hop counts to base station. Firstly, the base station will find all the possible roots in the attack area by the findR algorithm. These possible roots are the nodes with no outgoing edges. Then, the base station will count the number of nodes in the sub-graphs with the above roots respectively. Only the sub-graph with number of nodes more than half nodes in the attack area will be considered. It is because a sinkhole should be able to attract most of the traffic in the attack area. Afterwards, the sub-graphs with enough number of nodes will be checked if they contain contradictory information. Contradictory information is defined as the inconsistent information and in opposition to the four rules above.

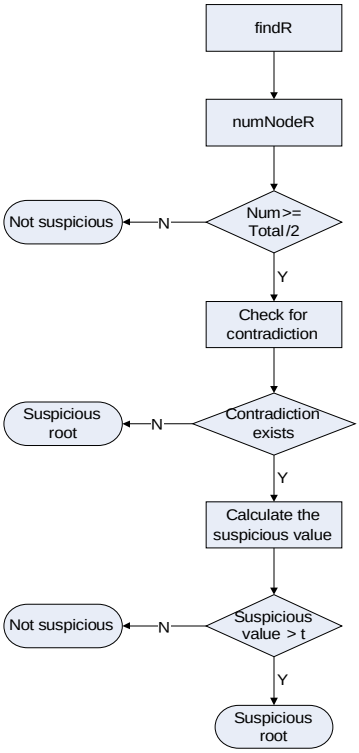



**Figure 5.11 Flow Chart for Identifying Suspicious Sinkhole**

Finally, a suspicious value $V_{suspicous}$ of the root node can be calculated by the following formula:

$$V_{suspicous} = w_1 * numNode(node) / TotalNum + w_2 * num\_neighbor\_with\_min\_hop\_count / TotalNumNeighbor * 2 / min\_hop\_count + w_3 * boolean\_outgoing\_edge,$$

where $w_3$ equals to 1 when no outgoing edge and 0 when there exists an outgoing edge

This value is composed of three parts. The first part is related to the number of nodes involved in the sub-graph with the node as root. The number of nodes is divided by the total number of nodes in the attack area. The second part is the number of neighboring nodes to the root node providing consistent number of hops to the destination. This number of nodes is divided by the total number of neighboring nodes to the root. Then, this ratio is divided by the number of hop counts to the destination. The value 2 is because of the number of hop count from the sinkhole to the base station is in general 1. The neighboring nodes of the sinkhole then provide number of hop count 2 from themselves to the base station. The third part is a boolean showing whether there is an outgoing edge from the suspicious node to another node. A sinkhole hole should be a node attracting network traffic and forward the received

packets to the base station via a high quality route. For that reason, it will not forward packets to a node in the attack area, so it will have no outgoing edge to its neighboring nodes. However, it is possible that the sinkhole provides incorrect routing information to cheat the base station. For example, it may present a non-existing outgoing edge to one of its neighbor, such that its neighbor is suspected by the base station.

In addition, there exist some colluding nodes cooperating with the sinkhole to provide incorrect next hop and number of hop counts. This kind of misbehavior makes the identification of sinkhole even more difficult. There are some limitations on our identification mechanism. Our mechanism is unable to identify the sinkhole correctly if all its neighboring nodes do not provide outgoing edges to itself. It is because our approach works by analyzing the network flow. If none of the nodes show their network traffic to the sinkhole, it is impossible to identify the sinkhole by building a graph on network flow.

### 5.2.7. Enhanced Sinkhole Identification Algorithm

In Algorithm 5.5, we make use of an array to check whether a detected root is a correct sinkhole. We assume array can have negative index, e.g. count[-1], count[-2] and w(p,r) means the hop_count from node p to node r. The index …, -1, 0, 1, 2, … represent the difference between the number of hops provided by a node p and the number of edges from p to the current root. Each node will have its own value. The array count of the root will store the sum of the results among the nodes in its tree.

```
for each root r
        initialize a new array count
        checkRootByCount(r, count, 1);
        if (count[0] => numNode(r)/2)
                r is a correct root.
        end if
end for

checkRootByCount (Node r, Array count, int depth)
        depth = depth +1
        for each precedent node p of r
                increase count[ w(p,r) – depth ] by 1
                calPV(p, count, depth)
        end for
end checkRootByCount
```

**Algorithm 5.5. Finding the array on hop count differences**

An ideal result of the array count of the root in a network with n nodes should have the following result:

| h | -2 | -1 | 0 | 1 | 2 |
|---|---|---|---|---|---|
| Count[h] | 0 | 0 | n | 0 | 0 |

It implies all the nodes in the network agree the current root is the sinkhole. However, the real sinkhole and some colluding nodes in the network may interference the result by providing incorrect information. If the values of the array do not belong to the ideal case, it means the current root may not be the real sinkhole. By analyzing the array count, we may estimate the possible hop counts of the real sinkhole from the fake sinkhole. For example, if we notice that the differences on the hop counts are within the range [-2,2] in majority. Then, we may suspect the real sinkhole is two hops from the fake sinkhole (current root). Afterwards, we can calculate the array counts for the nodes within two hops from the fake sinkhole and conclude the sinkhole as the best result (Algorithm 5.6).

```
for each root r
        initialize a new Array count
        initialize a new Path correctPath

        checkRootByCount(r, count, 1)

        S = {x>0 | forall y>0, count[x]+count[-x]>count[y]+count[-y]}
        x = min (S)

        correctRoot(r, r, x, 0, correctPath , count[0])
        apply correctPath on Network G
end for

checkRootByCount (Node r, Array count, int depth)
        depth = depth +1
        for each precedent Node c of r
                increase count[ w(c,r) – depth ] by 1
                checkRootByCount (c, count, depth)
        end for
end checkRootByCount

correctRoot(Node r, Path p, int totalLevel, int currentLevel, Path correctPath, int bestCount)

        if (currentLevel >= totalLevel)
                return
        end if

        currentLevel= currentLevel+1
        for each precedent node c of r
                initialize a new Array count
                reverse edge (c,r)

                        checkRootByCount (c, count, 1)
                        if (count[0]> bestCount)
                                correctPath = p->c
                        end if
                        correctRoot(c, p->c, totalLevel, currentLevel, correctPath , bestCount)

                reverse edge(c,r)
        end for
end correctRoot
```

**Algorithm 5.6. Find the Correct Sinkhole**

## 5.3. Locating Attackers by Redundant Paths

In this section, we suggest make use of redundant paths to locate the attacker. As a simple example, each node forwards the packet to two of its neighbors (Figure 5.12). Then, a sensor sends measurement to BS with redundant paths. By analyzing the data provided by redundant paths, the base station will locate attackers (Figure 5.13).

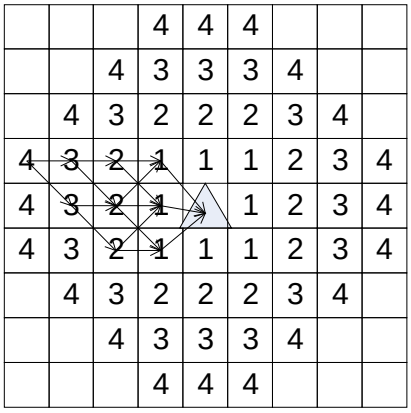



**Figure 5.12 Redundant paths to Base Station**

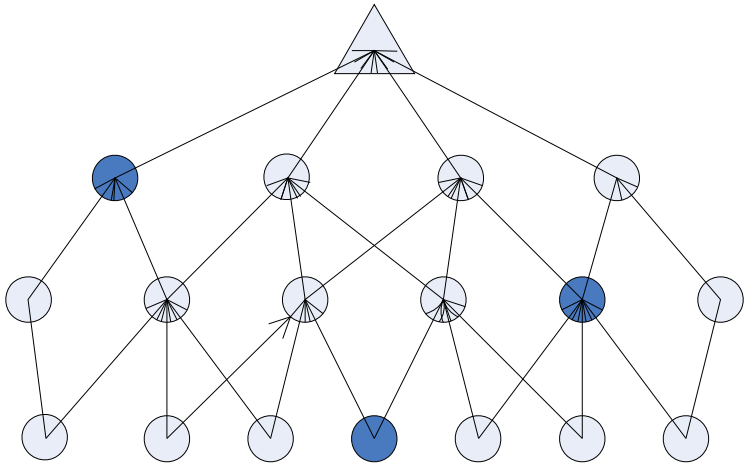


**Figure 5.13 Sub-graph with m=2**

Say, the dark nodes are attackers performing selective forwarding,

**Definition:**

D(x, BS) denotes the number of hops from x to BS

T(x) denotes the sub-graph with x as the root (from x and lower connecting nodes)

T~(x) denotes the sub-graph containing all paths from x to BS

N_T(x) denotes the number of nodes in T(x)

m denotes the number of neighbors that a node will forward its packet to

Given x and y, where D(x, BS) >= D(y, BS), N_T(x) >= N_T(y)
If x is malicious,
(1) all nodes belong to T~(x) performs normally with missing data from x
(2) x fails in providing data of nodes in T(x), but nodes in the same level of x may complete the job.

We will investigate this approach with more details in the future and examine the possibility for applying it in WSNs. Then, we will elaborate this mechanism and measure its performance.

## 6. Conclusion and Future Work

In this paper, we have introduced wireless sensor network and its security issues. To address the security issues, we studied intrusion detection measures and discussed our research direction. We propose an intrusion detection framework which can detect the intrusions, trace the intruders, and resist against the intrusions on wireless sensor networks. Our framework contains various detection components for collecting audit data. A number of techniques are considered for analyzing the data. After detecting and classifying the intrusions, our tracing procedure will attempt to identify and locate the intruders. Finally, resistance measures will be employed to defend against the intrusions. We analyzed the properties of a number of attacks on wireless sensor networks. Then, we proposed two intrusion detection architectures for hierarchical and cell-based wireless sensor networks. Also, we designed a detection and identification algorithm for discovering sinkhole attacks on wireless sensor networks. We plan to explore more identification algorithms for other type of attacks as well.

In the future, we will complete our intrusion detection architecture and develop new methodologies to detect the intrusions effectively. Moreover, we will explore possible measures to identify and locate the intruders for different kind of attacks. Furthermore, intrusion resistance techniques will be investigated to defend against the intrusions. Finally, the performance of our proposed framework will be evaluated.

# Bibliography

[1] “Intrusion Detection in Wireless Ad-Hoc Networks,” Y. Zhang and W. Lee, *ACM MOBICOM*, Boston, MA, USA, 2000.

[2] “Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems,” ed. M. Ilyas and I. Mahgoub, CRC Press, 2005.

[3] “Smart Environments: Technologies, Protocols, and Applications,” ed. D.J. Cook and S.K. Das, John Wiley, New York, 2004.

[4] “A Survey on Sensor Networks,” I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. *IEEE Communication Magazine*, August, 2002.

[5] “Sensor Networks: an Overview,” D. Culler, D. Estrin, and M. Srivastava, *IEEE Computer Magazine*, August 2004.

[6] “Ad Hoc Networks,” C. Perkins, Addison-Wesley, Reading, MA, 2000.

[7] “Routing Techniques in Wireless Sensor Networks: A Survery,” J. N. Al-Karaki and A. E. Kamal, *IEEE Wireless Communications*, December 2004.

[8] “Adaptive Protocols for Information Dissemination in Wireless Sensor Networks,” W. Heinzelman, J. Kulik, and H. Balakrishnan, *Proc. 5th ACM/IEEE Mobicom Conference*, Seattle, WA, August, 1999, pp. 174-85.

[9] “Negotiation-based protocols for disseminating information in wireless sensor networks,” J. Kulik, W. R. Heinzelman, and H. Balakrishnan, *Wireless Networks*, vol. 8, pp. 169-185, 2002.

[10] “Directed diRusion: a scalable and robust communication paradigm for sensor networks,” Intanagonwiwat, R. Govindan, and D. Estrin, *Proc. of ACM MobiCom '00*, Boston, MA, 2000, pp. 56-67.

[11] “Energy-Efficient Communication Protocol for Wireless Microsensor Networks,” W. Heinzelman, A. Chandrakasan and H. Balakrishnan, *Proc. of the 33rd Hawaii International Conference on System Sciences (HICSS '00)*, January 2000.

[12] “Lightweight Sensing and Communication Protocols for Target Enumeration and Aggregation,” Q. Fang, F. Zhao, and L. Guibas, *Proc. of the 4th ACM international symposium on Mobile ad hoc networking and computing*, 2003, pp. 165-176.

[13] “Data Aggregation in Wireless Sensor Networks - Exact and Approximate Algorithms,” Jamal N. Al-Karaki, Raza Ul-Mustafa, Ahmed E. Kamal, *Proc. of IEEE Workshop on High Performance Switching and Routing*, Phoenix, Arizona, USA, April 18-21, 2004.

[14] “Geography-informed Energy Conservation for Ad-hoc Routing,” Y. Xu, J. Heidemann, D. Estrin, *Proc. of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking*, 2001, pp. 70-84.

[15] “Geographical and Energy-Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks,” Y. Yu, D. Estrin, and R. Govindan, *UCLA Computer Science Department Technical Report, UCLA-CSD TR-01-0023*, May 2001.

[16] “Security in Wireless Sensor Networks,” A. Perrig, J. Stankovic, and D. Wagner, *Communications of the ACM*, vol. 47, no. 6, June 2005.

[17] “Designing secure sensor networks,” E. Shi and A. Perrig, *IEEE Wireless Communications*, December 2004.

[18] “The Byzantine Generals Problem,” L. Lamport, R. Shostak, and M. pease, *ACM Transaction Programming Languages and Systems*, vol. 4, no. 3, July 1982, pp. 382-401.

[19] “A Key-Management Scheme for Distributed Sensor Networks,” L. Eschenauer and V. Gligor, *Proc. of ACM CCS'02*, November 2002.

[20] “Random Key Predistribution Schemes for Sensor Networks,” H. Chan, A. Perrig, and D. Song, *IEEE Symposium on Research in Security and Privacy*, May 2003.

[21] “A Pairwise Key Pre-distribution Scheme for Wireless Sensor Networks,” W. Du, J. Deng, Y. S. Han, and P. K. Varshney, *Proc. 10$^{th}$ ACM Conf. Comp. and Commun. Security*, Oct 2003, pp. 42-51.

[22] “A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge,” W. Du, J. Deng, Y. Han, S. Chen, and P. Varshney, *IEEE Infocom*, 2004.

[23] “An Intrusion-Detection Model,” D. E. Denning, *IEEE Transactions on Software Engineering*, vol. SE-13, no. 2, February 1987, pp. 222-232.

[24] “GRIDS – A Graph Based Intrusion Detection System for Large Networks,” S. Staniford-Chen, S. Cheng, R. Crawford, and M. Dilger, *The 19th National Information Systems Security Conference, 1996.*

[25] “Preventing Wormhole Attacks on Wireless Ad Hoc Networks: A Graph Theoretic Approach,” L. Lazos, R. Poovendran, C. Meadows, P. Syverson, and L.W. Chang, *IEEE Wireless Communications and Networking Conference (WCNC)*, 2005.

[26] “JANUS: Towards Robust and Malicious Resilient Routing in Hybrid Wireless Networks,” B. Carbunar, L. Loanidis, and C. Nita-Rotaru, *ACM WiSe*, Philadelphia, Pennsylvania, USA, October 2004.

[27] “Rushing Attacks and Defense in Wireless Ad Hoc Network Routing Protocols,” Y-C. Hu, A. Perrig, and D. B. Johnson, *ACM WiSe*, San Diego, Califonia, USA, September 2003.

[28] “Packet Leashes: A Defense against Wormhole Attacks in Wireless Ad Hoc Networks,” Y-C. Hu, A. Perrig, D.B. Johnson, *IEEE INFOCOM*, 2003.

[29] “Efficacy of Misuse Detection in Adhoc Networks,” D. Subhadrabandhu, S. Sarkar, F. Anjum, *Proc. of the IEEE International Conference on Sensor and Ad hoc Communications and Networks (SECON)*, Santa Clara, CA, October 4-7, 2004.

[30] “Signature based Intrusion Detection for Wireless Ad-Hoc Networks: A Comparative study of various routing protocols,” F. Anjum, D. Subhadrabandhu, S. Sarkar, *Proc. of Vehicular Technology Conference, Wireless Security Symposium*, Orlando, Florida, October 2003.

[31] “Self-organized Critically & Stochastic Learning Based Intrusion Detection System for Wireless Sensor Networks,”S. S. Doumit, D. P. Agrawal, *MILCOM*, October 2003.

[32] “Intrusion Detection in Sensor Networks: A Non-copporative Game Approach,” A. Agah, S. K. Das, K. Basu, *IEEE International Symposium on Network Computing and Applications*, 2004.

[33] “Locating and Bypassing. Routing Holes in Sensor Networks,” Q. Fang, J. Gao, L. J Guibas, *IEEE INFOCOM'04*, March 2004.

[34] “LAD: Localization Anomaly Detection for Wireless Sensor Networks,” W. Du, L. Fang, and P. Ning, *IPDPS*, 2005.

[35] “Sensor-Based Intrusion Detection for Intra-Domain Distance-Vector Routing,” V. Mittal and G. Vigne, *ACM CSS*, Washington, DC, USA, November 2002.

[36] “Intrusion Tolerance and Anti-Traffic Analysis Strategies for Wireless Sensor Networks,” J. Deng, R. Han, S. Mishra, *IEEE International Conference on Dependable Systems and Networks (DSN)*, 2004, pp. 594-603.

[37] “Localized Fault-Tolerant Event Boundary Detection in Sensor Networks,” M. Ding, D. Chen, K. Xing, and X. Cheng, *IEEE INFOCOM*, 2005.

[38] “Tamper resistance – a cautionary note,” R. Anderson and M. Kuhn, *Proc. 2nd USENIX Workshop Electron. Commerce, USENIX*, Berkeley, CA, 1996.

[39] “Localized algorithms in wireless ad-hoc networks: location discovery and sensor exposure,” S. Meguerdichian et al., *Proc. 2nd ACM Symp. Mobile Ad Hoc Networking and Computing (MOBIHOC)*, ACM Press, New York, 2001, pp.106.

[40] “The impact of data aggregation in wireless sensor networks,” B. Krishnamachari, D. Estrin, and S. Wicker, *Proc. of the 22nd International Conference on Distributed Computing Systems* , July 2002, pp. 575 - 578.

[41] “Denial of Service in Sensor Networksm” A. D. Wood and J. A. Stankovic, *IEEE Computer*, 2002.

[42] “Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures,” C. Karlof and D. Wagner, *Communications of the ACM*, June 2004.

[43] “The Sybil Attack,” J. R. Douceur, *1st International Workshop on Peer-to-Peer Systems (IPTPS '02)*, March 2002.

[44] “Tolerating failures of continuous-valued sensors,” K. Marzullo, *ACM Trans. Computer Syst.*, vol. 4, issue 4, November 1990, pp. 287-304.

[45] “How to reconcile fault-tolerant interval intersection with the Lipschitz condition,” U. Schimd and K. Schossmaier, *Distributed Computing*, vol. 14, issue 2, pp. 101-111, May 2001.

[46] “Fault-tolerant interval estimation fusion by Dempster-Shafer theory,” B. Li, Y. Zhu, and X. Li, *Proc. 5th Int. Conf. Inf. Fusion*, Annapolis, MD, July 2002, pp. 1605-1613.

[47] “Distributed Detection,” P.K. Varshney, Spinger-Verlay, Deidelberg, 1996.

[48] “Multisensor Fusion,” A.K. Hyder, E. Shahbazian, and E. Waltz, eds., Academic Publishers, 2002.

[49] “Multisensor Fusion under unknown Distributions: Finite Sample Performance Guarantees, in Multisensor Fusion,” N.S.V. Rao, et al. Kluwer Academic Publishers, 2002.

[50] “The Global Positioning System,” P. Enge and P. Misra, *Proc. IEEE Special Issue on GPS,* vol. 83, 1999, pp. 3-15.

[51] “Dynamic fine-grained location in ad hoc networks of sensors,” A. Savvides, C.C. Han, and M.B.

Srivastava, *ACM/IEEE MOBICOM*, Rome, 2001, pp. 166-179.

[52] “Tracing Network Attacks to Their Sources,” T. Baba and S. Matsuda, *IEEE Internet Computing*, April 2002.

[53] “Single-Packet IP Traceback,” A. C. Snoeren and C. Partridge, *ACM Sigcomm '01*, San Diego, CA, August 2001.

[54] “Practical Network Support fro IP Traceback,” S. Savage et al., *Proc. 2000 ACM SIGCOMM*, vol. 30, no. 4, ACM Press, New York, August 2000, pp. 295-306.

[55] “On Design and Evaluation of "intention-driven" ICMP Traceback,” *Proc. IEEE International Conference on Computer Communications and Networks*, October 2001.

[56] “CenterTrack: An IP Overlay Network for Tracking DoS Floods,” *Proc. 9th Usenix Security Smp.,* Usenix Assoc., Berkeley, Calif., August 2000.

[57] “DecIdUouS: Decentralized Source Identification fro Network-based Intrusions,” *Proc. 6th IFIP/IEEE Int’l Symp. Integrated Network Management*, IEEE Comm. Soc. Press, New York, May 1999, pp. 701-714.

[58] “Detection, Defense, and Tracking of Internet-Wide Illegal Access in a Distribued Manner,” *Proc. INET 2000*, Internet Soc., Reston, Va., July 2000.

[59] “Efficient Tracing of Failed Nodes in Sensor Networks,” J.Staddon, D. Balfanz, G. Durfee, *Proc. of the First ACM International Workshop on Wireless Sensor Networks and Applications*, Atlanta, Georgia, USA, September 28 2002.