MODELING OF SENSOR NETWORKS - ATTACK SURFACE, QOS SURFACE

By

ARASAVEL RAMARAJ

Bachelor of Engineering in Computer Science

Bharathiar University

India

2002

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
May, 2006

MODELING OF SENSOR NETWORKS - ATTACK

SURFACE, QOS SURFACE

Thesis Approved:

Dr. Johnson P. Thomas

Thesis Advisor

Dr. G. E. Hedrick

Dr. Debao Chen

Dr. A. Gordon Emslie

Dean of the Graduate College

ACKNOWLEDGEMENTS

I sincerely thank my advisor Dr. Johnson P. Thomas, for his guidance, timely support and concern and undying efforts throughout the entire duration to complete the project and my thesis. I, also thank the other members of my committee Dr. Debao Chen and Dr. G. E. Hedrick for their assistance. I'm also indebted to Dr. John P. Chandler for organizing my research work and helping me in my literature search.

The Computer Science Department, staff and faculty alike deserve special mention for the continual assistance in the technical areas that I required throughout the entire period of my education here. Special thanks to Arun Kumar Venkateshwaran, Toni V. Kunnel for helping in literature search, and my other friends and roommates for all their patience.

Words can not express the gratitude that I have for my parents, Mr. S. Ramaraj and Mrs. R. Thamilmullai, for all their love, care, encouragement, and the efforts taken to ensure that I succeed through all walks of life. I also have to thank my sister Mrs. G. Arulmalar and brother Mr. R. Ezhilmuthu for their support through this duration of my higher education.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Security and QoS are the two foremost features of sensor networks. Sensor networks are deployed across various environments where an optimum level of security and QoS is ideally required. Measurement of service, both quantitatively and qualitatively, has been a long-standing challenge to the community.

We propose a novel approach – creation of service (Security, QoS) surfaces to identify vulnerabilities and QoS features present in a sensor network. A sensor network has been modeled along with the communication protocol employed to indicate how we can obtain an optimum level of security and QoS.

In our approach to determine a model for security of a sensor network we want a measure— at a lower abstraction level—that allows us to refer to very specific states (i.e., configurations) of a system. Given this intermediate viewpoint, we can see that there are certain system features that are more likely than others to be opportunities of attack. The counts of these "more likely to be attacked" system features determine a system's attackability (possibility of being attacked - vulnerability).

We extend the metric based on the notion of attack surface proposed for Operating Systems security in [1] to security of sensor network protocols. This metric takes into consideration the parameters present at the design level of a system: above the level of code, but below the level of the entire system.

These attack opportunities are classified into different abstract *dimensions*, which together define a system's *attack surface*. Intuitively, the more exposed the attack surface, the more likely the system could be successfully attacked, and hence the more insecure it is. An attack surface is defined in terms of the system's actions that are externally visible to the users and the system's resources that are accessed or modified by the system's actions. The dimensions across which an attack surface is defined are targets and enablers, channels and protocols, and a set of access rights.

This approach is extended to determine the QoS surface of a sensor network across the QoS dimensions identified. Intuitively, the more exposed the QoS surface, the more likely the system could provide a high degree of QoS. Similar to an attack surface, a QoS surface is defined in terms of the system's actions that are externally visible to the users and the system's resources that are accessed or modified by the system's actions. The dimensions across which a QoS surface is defined are inhibitors, enhancers and bottlenecks.

We further extend this work by proposing an approach to calculate a single measure for security. Finally, we propose an approach to compare the security and QoS in sensor networks that contain multiple independent protocols. This allows us to increase or decrease the security or QoS levels in the different protocols. The dimensions of the service surfaces can be modified to enhance the service provided by the communication protocol.

The rest of the thesis is outlined as follows:

In chapter 2, we provide the details of the metric employed for determining the attack surface of an operating system. The chapter also provides an example of how an attack surface is modeled using the dimensions identified. In chapter 3, we provide a general overview of sensor network architecture, routing protocols, attacks that can be mounted on a sensor network, QoS challenges faced in the field of sensor networks and the network assumptions made for sensor networks.

In chapter 4, we provide the attack surfaces for the wanderer protocol along with the attacks that can be mounted on this protocol. We also provide an attack surface modeled on the wanderer + LEAP protocol to illustrate the feasibility of our metric. This chapter also provides the various attacks possible on the physical layer, transport layer along with suitable defense mechanisms identified.

In chapter 5, we provide the QoS surface for the On-Demand Delay Constrained Unicast Routing Protocol. We provide the QoS features present in ODRP similar to attacks possible in chapter 4. In chapter 6, we provide an example to illustrate the two metrics – attack surface, QoS surface.

CHAPTER II

REVIEW OF LITERATURE

This chapter provides an overview of the key concepts involved in modeling the new metrics – attack surface and QoS surface. The chapter illustrates the previous work undertaken to understand the metric of an attack surface for operating systems. We make use of the key techniques illustrated to model our new metrics. The concepts – attack surface automata, vulnerabilities, attack surface measurement, comparisons are made use of in subsequent chapters to create an attack surface and QoS surface for sensor networks.

The chapter provides a concise description of the previous work attempted in the modeling of attack surfaces for operating systems. The chapter explains the attack surface automata along with the dimensions that have been identified for modeling the attack surface for operating systems. The chapter illustrates two examples – Linux and Windows operating systems, with the respective attack surfaces, methods to decrease the attack surface. It also provides the results obtained when comparing different versions of the operating system based on the attack surface modeled.

Attack Surface Automata – A Preview

State machines, which will be called *System* and *Threat*, are used to represent the system and threat respectively. A state machine has a set of states, a set of initial states, a set of actions, and a state transition relation. An attack is modeled as a sequence of executions of actions that ends in a state that satisfies the adversary's goal, and in which one or more of the actions executed in an attack involves any vulnerability [2].

2.1 State Machines

A *state machine*, $M = (S, I, A, T)$, is a four-tuple where S is a set of states, I is a set of initial states, A is a set of actions, and T is a transition relation. A state $s \varepsilon S$ is a mapping from typed *resources* to their typed *values*:

$$s\text{: } Res_M \rightarrow Val_M$$

A *state transition*, (s, a, s'), is the execution of action 'a' in state s resulting in state s'. A change in state means either a new resource is added to the mapping, a resource was deleted, or a resource changes in value. Each state transition is assumed to be atomic.

An *execution* of a state machine is the alternating sequence of states and action executions:

$$s_0\ a_1\ s_1\ a_2\ s_2 \ldots s_{i-1}\ a_i\ s_i \ldots$$

Such that $s_0 \varepsilon I$ and $i > 0$, $(s_{i-1}, a_i, s_i) \varepsilon T$.

The *behavior* of a state machine, M, is the set of all its executions. This set is denoted as set $Beh\ (M)$. A state s is *reachable* if either $s \varepsilon I$ or there is an execution, $e \varepsilon Beh\ (M)$, such that s appears in e.

Actions are specified by pre- and post-conditions. For an action, $a \varepsilon A$, if $a.\ pre$ and $a.\ post$ denote a's pre- and post-condition specifications, the subset of the transition relation, T, that involves only the action a is defined as follows:

$$a.\ T = \{(s, a, s')\text{: } S \times A \times S \mid a.pre(s) \Rightarrow a.post(s, s')\}$$

Both the system under attack and the threat (adversary) are modeled as state machines:

$$System = (S_S, I_S, A_S, T_S)$$

$$Threat = (S_T, I_T, A_T, T_T)$$

The resources of a state machine, *M*, are partitioned into a set of *local resources* and a set of *global resources*,

$$ResM = Res^L{}_M \cup Res^G{}_M$$

The *combination* of the two state machines, *ST = System* **X** *Threat*, is defined by merging all the corresponding components:

- $I_{ST} = I_S \cup I_T$
- $A_{ST} = A_S \cup A_T$
- $T_{ST} = T_S \cup T_T$

The global resources of *S* and the global resources of *T* have been identified such that

$Res^G{}_{ST} = Res^G{}_S \cup Res^G{}_T$

So, $Res_{ST} = Res^L{}_S \cup Res^L{}_T \cup Res^G{}_{ST}$

Finally, $Val_{ST} = Val_S \cup Val_T$

An adversary targets a system under attack to accomplish a goal:

$$\text{System-Under-Attack} = (\text{System X Threat}) \times \text{Goal}$$

Goal is a predicate over all the states in S_{ST}. The goal of the adversary in the model of a system under attack is made explicit. In our context, i.e. in wireless sensor networks *Threat* is synonymous with the system's "environment." Thus, *Threat* is used to model environmental failures, due to benign or malicious actions, that affect a system's state. It can be inferred that the way to reduce the attack surface is to ensure that the behavior of *System* prohibits *Threat* from achieving its *Goal*.

2.2 Vulnerabilities

Vulnerabilities share the common intuition that something in the actual behavior of the system deviates from the intended behavior. This can be represented formally by comparing the difference between the behaviors of two state machines [1].

Suppose there is a state machine that models the intended behavior and one that models the actual behavior:

$$Intend = (S_{\mathrm{Int}}, I_{\mathrm{Int}}, A_{\mathrm{Int}}, T_{\mathrm{Int}})$$

$$Actual = (S_{\mathrm{Act}}, I_{\mathrm{Act}}, A_{\mathrm{Act}}, T_{\mathrm{Act}})$$

The vulnerability difference set, *Vul*, is defined to be the difference in behaviors of the two machines:

$$Vul = Beh\ (Actual) - Beh\ (Intend)$$

An execution sequence in *Vul* arises from one or more differences between a component of the state machine *Actual* and the corresponding component of *Intend*, i.e. differences between the corresponding sets of (1) states(or more relevantly, reachable states), (2) initial states, (3) actions, or (4) transition relations. Any one of the above types of differences is referred to as vulnerabilities.

Let's consider each of the cases associated with the vulnerabilities:

1. $\boldsymbol{S_{Act} - S_{Int} \neq \varnothing}$ (2.1)

A difference in state sets indicates that there are some states that are defined for *Actual* that are not intended to be defined for *Intend*. The difference may be due to

1. A resource that is in a state in *Actual*, but not in *Intend*
2. A value allowed for resource in *Actual* that is not allowed for that resource in *Intend*. (A resource that is not in a state in *Actual*, but is in *Intend* is ok.)

The difference is not too serious if the states in the difference are not reachable by some transition in T_{Act}. If they are reachable, then the difference in transition relations will pick up on this vulnerability. However, though the states are not reachable initially, there is a possibility that if any of the specifications for actions changes in the future they may become reachable, so it must be ensured that the set of reachable states in *Actual* is a subset of that of *Intend*.

2. $\boldsymbol{I_{Act} - I_{Int} \neq \varnothing}$ (2.2)

A difference in initial state sets shows that there is at least one state in which one can start an execution when one ought not to. This situation can arise if

1. Resources are not initialized when they should be.
2. Resources are given incorrect initial values.
3. When there are resources in an initial actual state but not in any initial intended state.

3. $\boldsymbol{A_{Act} - A_{Int} \neq \varnothing}$ (2.3)

A difference in action sets shows that there are some actions that can be actually done that are not intended. These actions will surely lead to unexpected behavior. The difference will show up in the differences in the state transition relations.

4. $\boldsymbol{T_{Act} - T_{Int} \neq \emptyset}$ (2.4)

A difference in state transition sets indicates that there is at least one state transition allowed in *Actual* that should not be allowed according to *Intend*. This situation can arise if

1. The action sets are different
2. The pre- and post-conditions for an action common to both action sets are different.

Case 2 occurs when $A_{Act} = A_{Int}$,

Let us consider a given action $a \varepsilon AInt$. If $a.\ T_{Act} - a.\ T_{Int} \neq \emptyset$ then there are some states in *Actual* in which *a* can be executed and not in *Intend* or which can be reached as a result of executing the action *a* in *Actual* and not in *Intend*.

Let $a_{Act}.pre$, $a_{Int}.pre$ be the pre-conditions for the action *a* in *Actual* and *Intend*, respectively, and similarly for their post-conditions ($a_{Act}.post$ and $a_{Int}.post$). In terms of pre- and post-conditions, *no difference* can arise for the transition relations if

- $a_{Act}.pre \Rightarrow a_{Int}.pre$
- $a_{Act}.post \Rightarrow a_{Int}.post$

The system is inferred to be safe if the "actual" behavior is stronger than the "intended" behavior of the system i.e. the conditions are stronger for actions in *Actual* when compared to that of the actions in *Intend.*

As per the assumptions made, *Actual* models the actual behavior of the system then the system combined with the *Threat* machine looks like:

System-Under-Attack = (Actual X Threat) × *Goal* as opposed to

System-Under-Attack = (Intend X Threat) × *Goal*

If *Intend* was implemented correctly, *Goal* would not be achievable. If it were implemented correctly then *Actual* and *Intend* will be equivalent and no adversary can break the security features available for the intended behavior of the system.

2.3 Attacks

As described earlier, an attack is the means of exploiting any vulnerability [5]. An attack is modeled to be a sequence of action executions, at least one of which involves any vulnerability. To be precise, an attack, *k*, either starts in an unintended initial state or reaches an unintended state through one of the actions executed in *k*. An attack includes the execution of actions from both state machines, *System* and *Threat*. The difference between an arbitrary sequence of action executions and an attack is that an attack includes any one or both of the following two conditions

(1) The execution of an action whose actual behavior deviates from the intended behavior.

(2) The execution of an action, a ε $A_{Act} - A_{Int}$ ($\neq \emptyset$).

In case (2), the set of unintended behaviors will include behaviors not in the set of intended behaviors since $A_{Act} \neq A_{Int}$.

For any given attack, *k*, the *means of an attack* is a set comprising of all actions in *k* and the set of all process and data resources accessed in performing each action of the attack *k*. The resources include all global and local resources accessed by each action in *k* and all parameters passed as arguments or returned as a result of each action executed for the attack *k*.

2.4 Dimensions of an attack surface

An attack surface comprises of the following dimensions:

- Targets and Enablers

  Targets are processes or data resources that an adversary aims to control, modify or compromise in a system.

  These are the other processes or data resources that are used by the adversary to gain control and carry out its attack successfully in the system.

- Channels and Protocols

  An adversary gains control over the resources through communication channels and protocols.

- Access rights

  Control over processes or data resources is subject to constraints imposed by the system's set of access rights.

2.4.1 Targets and Enablers

Let us consider a simple example of a virus attack to understand the dimensions of an attack surface clearly.

Example:

The general steps involved in such an attack are:

Step 1: Ship an executable (considered to be piece of data) within a carrier to a target machine.

Step 2: Use an enabler, e.g., a browser, to extract the payload (the executable) from the carrier.

Step 3: Make use of an interpreter to execute the executable to cause a state change on the target machine.

The attacker's goal may be to modify the state of the target machine, to use up its resources, or to set it up for future attacks. The goal of the attacker can be achieved on the completion of the three steps mentioned [1].

The existence of this type of a virus attack leads us to understand two special types of data resources.

1. E*xecutables* is a distinguished type of data resource, they can be evaluated.

Executables are associated with one or more eval functions, *eval: executable →unit.*

The same executable can be interpreted differently by different eval functions with distinct effects. The executables can be considered to be targets and controlling such a target involves the ability to call an *eval* function on it. The adversary calls the corresponding eval function for the executable, so that the pre-condition of the next step in the attack can be established as a side effect.

2. *Carriers* are the second distinguished type of data resource. Executables are embedded in carriers. Specifically, carriers have a function *extract payload: carrier → executable*.

2.4.2 Channels and Protocols

A channel is a means of communicating information from a sender to a receiver (in the present scenario from an attacker to a target machine). Associated with each kind of channel is a *protocol*, the rules of exchanging information.

Channels are considered to be data resources. A channel shared between *System* and *Threat* machines is an element of Res^{G}_{ST} in the combination of the two machines.

In an attack sequence, the *Threat* machine may establish a new message-passing channel, e.g., after scanning host machines (other sensor nodes). Future communication between the two machines, which aids in exploitation of the vulnerability present in the host machine.

2.4.3 Access rights

Access rights are associated with all resources. These access rights determine the type of access available on a particular resource. For executables, execution rights are associated. Similarly, rights are associated with channels (since they are considered to be data resources) and channel endpoints (since they are running processes).

Conceptually these rights can be modeled as a relation, suggestive of Lampson's original access control matrix [6]:

$$Access \subset Principals \times Res \times Rights$$

Where $Principals = Users \cup Processes$, $Res = Processes \cup Data$.

When required, to represent conditional access rights, the above relation can be extended with a fourth dimension *Conditions*,

$$Access \subset Principals \times Res \times Rights \times Conditions$$

An interesting subset of the *Access* relation is the Accounts set. Accounts can be data or process targets. An account is viewed as shorthand for a particular principal with a particular set of access rights. In the case of sensor networks, *Accounts* represent principals, i.e., nodes and base stations. There are some special accounts that have default access rights (nodes) and accounts that have admin privileges (base stations).

Reducing the attack surface with respect to access rights is a special case of abiding by the Principle of Least Privilege: Grant only the relevant rights to each of the principals that are allowed access to a given resource.

2.5 Attack Surface Measurement

The attack surface of a system consists of the set of system actions AS and the collective set of resources of each action a ε A_S. A naive but impractical way of measuring the attack surface is to enumerate the set of system actions of a given system and count the number of resources in each of the action's resource set. A meaningful way to measure the attack surface based on the attack classes of the system is provided. Then, given two versions, A and B, of a system their relative attack surface exposure with respect to the attack classes is compared.

Method

Consider a system with a fixed set, AS, of system actions, each specified in terms of pre- and post-conditions. In practice, a system's API serves as the set of system actions.

Step 1: Identify the resources that are potential targets of attack from the given set of system actions A_S. Let Type be the set of types of all these resources.

Step 2: Given a set, Prop, of properties of interest over the resources, induce a type hierarchy over the set, Type, of resource types identified in Step 1. Every leaf node in this type hierarchy is an attack class of the system. Let Attack Class be the set of attack classes.

Step 3: Define a payoff function *F: Attack Class* $\rightarrow$ *[0, 1]* to assign payoffs to each attack class identified in *Step 2*.

Step 4: Choose some k attack classes from the attack classes identified in Step 2.

Step 5: Compare the two versions of the system, A and B, with respect to these k attack classes to obtain their relative attack surface exposure.

Some notes on the steps involved in the method:

In Step 2, the knowledge of the system is used to state the properties of interest. They will differ from system to system and they may change over time based on the evolving experience with a system as it evolves over time.

In Step 3, payoffs represent the likelihoods of attack. An attack class with a high payoff indicates that resources of that class are more likely to be attacked than resources of an attack class with a lower payoff. One naive way of assigning payoffs is to count the number of times a resource appears in the pre- and post-conditions of system actions; assigning higher payoffs to the resources having higher counts. Another way of assigning payoffs is based on a system's reported history and providing higher payoffs to attack classes that appear in a greater number of vulnerability bulletins. A more sophisticated approach for defining a payoff function is to quantify the "damage" the adversary can effect if resources in a given attack class are compromised, e.g., in terms of cost to repair the system.

Step 4 proves that measuring an attack surface in practice need not involve all the attack classes of the system. Choose the top k attack classes of a system based on the payoffs assigned in Step 3.

In Step 5, there are many ways to use the attack classes to do the comparison. One simple way is to count the number of instances of each attack class in both versions and compare the numbers.

The higher the count for a given attack class, the more the attack surface exposure is for that class. Another way is to incorporate the payoffs identified in Step 3 as weights in these counts; e.g., count the weighted sum of all instances in each attack class with the same immediate super type, and compare the versions with respect to the super type, rather than its individual attack class subtypes. In other words, the attack surface contribution of a resource type T with attack classes S_1, S_2, S_k as its subtypes is given by

$$^{k}\Sigma_{i=1} n\ (Si) \times wi$$

Where n (Si) is the number of instances of the attack class Si and wi is the payoff assigned to Si in Step 3. A more sophisticated comparison might take into account the interactions between various attack classes, e.g., compare the number of sockets opened by services running as root and ignore the other sockets in the system.

2.6 Reducing the Attack Surface

The formal model and measurement method has suggested ways in which the exposure of an attack surface can be reduced:

• Reduce the number of system actions.

• Remove known or potential system vulnerability by strengthening the pre- and post-conditions of a system action, e.g., in a way that prevents the Goal of the Threat from ever being achieved.

• Eliminate an entire attack class.

• Reduce the number of instances of an attack class.

2.7 Linux Example

In this section, the results of measuring the attack surface of four versions of the Linux operating system are described.

Step 1 of the method requires that all resources of the system that are potential targets of attacks were identified. Since it is impractical to enumerate the set of system actions for Linux, and then identify all possible resources each action might access or modify, the set of resource types was derived indirectly. All resources that appear in the MITRE CVEs [31] as potential targets of attack were considered and their types were identified accordingly.

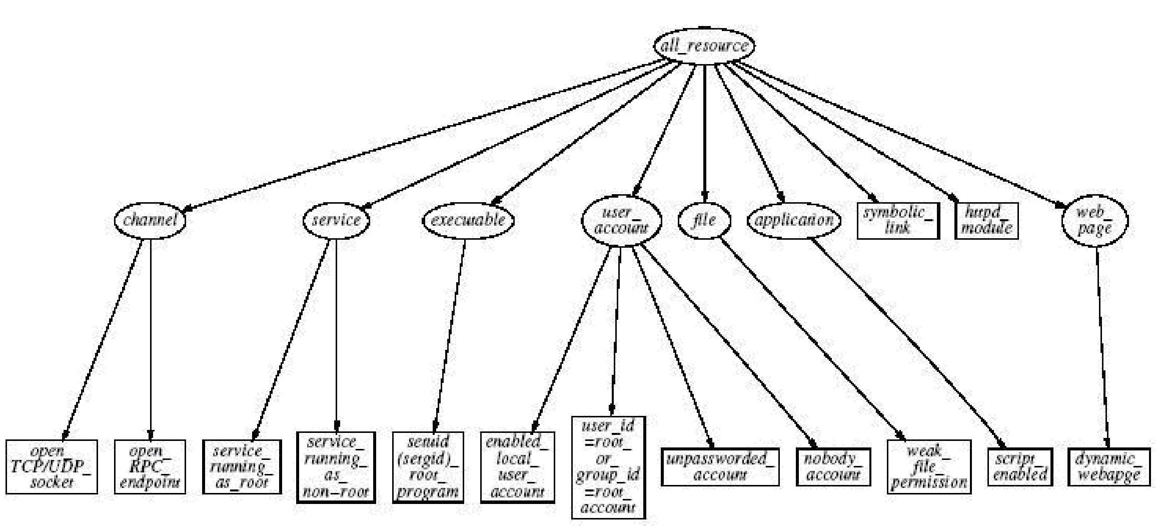

Figure 1: Linux Attack Classes

In Step 2, a set of properties was defined over the resource types identified in Step 1, and a type hierarchy over the extended set of resource types was induced. Every leaf node in this type hierarchy was an attack class, resulting in 14 attack classes for Linux. Fig3. 1 depicts the type hierarchy and the 14 Linux attack classes. For example, in the type hierarchy, *nobody account* ≤ *user account* ≤ *all resource* and *symbolic link* ≤ *all resource*. Descriptive names were used for the types and subtypes to be suggestive of the properties used to induce the hierarchy.

In Step 3, the history of attacks on Linux based on CVEs was used to assign payoffs to the attack classes identified in Step 2. Explicit numeric payoff values were not assigned because there was no plan to use the numeric values in Steps 4 and 5. Instead, a higher payoff for an attack class was assumed if the resources of that attack class appeared a greater number of times in the CVEs. Note that counting the number of times a system appears in the CVEs is different from counting the number of times each attack class of the system appears in the CVEs. For example, consider two versions, A and B, of a system having ten attack classes and each appearing in 50 CVEs. Let two of the attack classes of version A appear in 25 CVEs each, ten of the attack classes of version B appear in 5 CVEs each, and the first two attack classes have lower payoffs compared to the remaining eight. Both A and B appear in same number of CVEs, but the attack surface exposure of B is more than that of A with respect to the ten attack classes.

In Step 4, 11 attack classes were chosen for attack surface measurement out of the 14 identified in Step 2. Three attack classes were not included in the measurement since it was not possible to count the number of instances of each of them for all four versions of Linux.

In Step 5, the number of instances, of each of the 11 attack classes was counted for four versions of the Linux operating system and compared the numbers to get a relative measure of their attack surface.

2.7.1 Attack Classes

The resources appearing in the publicly known vulnerabilities reported in the CVEs and CVE candidates list of MITRE [10] were identified. Further information about the vulnerabilities was obtained from the CERT Advisories [12], Debian Security Advisories [10], and Red Hat Security Advisories [9] referenced in the CVEs. The types of these resources are categorized into 14 attack classes. Each attack class is described with an example (CVE) of a vulnerability of a resource in that attack class.

Linux Attack Classes

- *Open TCP/UDP socket*
- *Open remote procedure call (RPC) endpoint*
- *Service running as root*
- *Service running as non-root*
- The *setuid (setgid) root* program
- *Enabled local user account*
- *The user id=root or group id=root account*
- *The unpassworded account*
- *Nobody account*
- *Weak file permission*
- *Script enabled*

- *Symbolic link*
- *The httpd module*
- *The dynamic web page*

2.7.2 Attack Class Validation

If the 14 attack classes are complete enough, then any vulnerability mentioned in Bugtraq should be covered by one of the mentioned attack classes.

For example, the vulnerability entry in the Bugtraq database with bugtraqid 7769 [9] [10] describes a format string vulnerability, stack overflow, and file corruption in the mod gzip module running in debug mode. The target of the attack involving the vulnerability is the resource mod gzip and it is an instance of the Linux attack class httpd module.

As another example, Bugtraqid 8732 [9] describes ASN.1 parsing vulnerabilities in OpenSSL which a remote attacker can exploit to cause a denial-of-service or to execute arbitrary code on the system. The resources that are the targets of this attack are the applications such as *Ssh* that use OpenSSL. Ssh is an instance of the Linux attack class service running as root.

2.7.3 Attack Surface Measurements

The results of measuring the attack surface of following four versions of the Linux operating system have been provided

• Debian is a Debian GNU/Linux 3.0r1 distribution obtained from Debian's website [27].

• RH Default is a Red Hat 9.0 Linux distribution obtained from Red Hat's website [32].

• RH Facilities is a customized Red Hat 9.0 Linux distribution installed at CMU School of Computer Science.

• RH Used is an instance of RH Facilities after use by a graduate student for three months.

Attack Class	Debian	RH Default	RH Facilities	RH Used
open_TCP/UDP_socket	15	12	40	41
open_remote_procedure_call(RPC)_endpoint	3	3	3	3
service_running_as_root	21	26	29	30
service_running_as_non-root	3	6	8	8
setuid(setgid)_root_program	54	54	72	72
enabled_local_user_account	21	25	33	34
user_id=root_or_group_id=root_account	0	4	3	3
unpassworded_account	0	0	2	2
nobody_account	1	1	1	1
weak_file_permission	7	7	21	37
script_enabled	1	2	2	2
symbolic_link	*	*	*	*
httpd_module	-	-	-	-
dynamic_web_page	-	-	-	-

Table 1: Attack Surface Measurement Results

• ***Default comparison***: Compare the attack surfaces of Debian and RH Default to measure the relative security of different flavors (versions) of the system.

• ***Customized usage-based comparison***: Compare the attack surfaces of RH Default and RH Facilities to observe the change in the security level of a system based on its customization.

• ***Time-based comparison***: Compare the attack surfaces of RH Facilities and RH Used to monitor the security level of a system as it changes over time.

2.7.4 Results

Debian vs. RH Default

As shown in Table 1, RH Default has higher counts in each of five attack classes, Debian has a higher count in one attack class, and both have the same counts in each of five attack classes. Hence the attack surface exposure of Red Hat is greater than that of Debian. Debian is perceived to be a more secure operating system and this is reflected in the measurement. Design choices play an important role in making a system more or less secure.

RH Default vs. RH Facilities

As shown in Table 1, RH Facilities has higher counts in each of seven attack classes, RH Default has higher counts in one attack class, and both have the same counts in each of three attack classes. The attack surface exposure of the facilities distribution is more than that of the default distribution. The facilities distribution is customized to make it more useful compared to the default distribution.

For example, it has the AFS file system installed. It has services such as lclaadmd, opshell, kopshell and terad installed for remote management and network backup. Installing these features increases the counts for the attack classes open TCP/UDP socket, service running as root, enabled local user account, etc. The results show that the attack surface exposure has increased with customization, thereby making the system less secure.

RH Facilities vs. RH Used

As shown in Table 1, RH Used has higher counts in each of four attack classes and both have the same counts in each of seven attack classes. The used version's attack surface exposure is greater than the initially installed version. The three-month use of the system increased the counts of the attack classes open TCP/UDP socket, service running as root, enabled local user account, and weak file permission. The results show that the attack surface exposure has increased over time making the system less secure.

2.8 Windows Example

Howard identified a set of 17 RASQ vectors [1] [11] and defined a simple attack surface function to determine the relative attack surface of seven different versions of Windows.

2.8.1 Attack Vectors for Windows

- *Open sockets*
- *Open RPC endpoints*
- *Open named pipes*
- *Services*
- *Services running by default*
- *Services running as SYSTEM*
- *Active Web handlers*
- *Active ISAPI filters*
- *Dynamic web pages*
- *Executable vdirs*
- *Enabled accounts*
- *Enabled accounts in admin group*

- *Null sessions to pipes and shares*
- *Guest account enabled*
- *Weak ACLs in FS*
- *Weak ACLs in Registry*
- *Weak ACLS on shares*
- *VBScript enabled*
- *Jscript enabled*
- *ActiveX enabled*

2.8.2 Attack Surface Calculation

In Howard's calculation, the attack surface area is the sum of independent contributions from a set of channels types, a set of process target types, a set of data target types, a set of process enablers, all subject to the constraints of the access rights relation, *A*.

$$SURF^{A} = SURF^{A}\,ch + SURF^{A}\,pt + SURF^{A}\,dt + SURF^{A}\,pe$$

This simple approach has a major advantage in that it allows the categories to be measured independently.

Analysis of Attack Surface Calculation

The two main conclusions to draw are that *with respect to the 20 RASQ attack vectors*

(1) The default version of a running Windows Server 2003 system is more secure than the default version of a running Windows 2000 system

(2) A running Windows Server 2003 with IIS installed is only slightly less secure than a running Windows Server 2003 without IIS installed.

While it is too early to draw any conclusions about Windows Server 2003, the RASQ numbers are consistent with observed behavior in several ways:

• Worms such as Code Red and Nimda spread through a variety of mechanisms. In particular, Windows NT 4.0 systems were at far greater risk of being successfully attacked by these worms if the systems were installed with IIS than if they were not. This observation is consistent with the increased RASQ of this less secure configuration.

• Windows 2000 security is generally perceived as being an improvement over Windows NT 4.0 security [10]; the differences in RASQ for the two versions in a similar configuration (i.e., with IIS enabled) reflect this perception.

• Conversely, Windows 2000 (unlike Windows NT 4.0) is shipped with IIS enabled by default, which means that the default system is actually *more* likely to be attacked.

This observation is consistent with anecdotal evidence that many Windows 2000 users affected by Code Red and Nimda had no idea they were actually running IIS.

2.9 Conclusion

This novel approach of creating an attack surface to determine the secure features of a system can be extended to any application. This approach can be verified to be a feasible solution to detect the vulnerabilities in a system, and hence enhance the security associated with the system.

CHAPTER III

SURFACES FOR SENSOR NETWORKS

3.1 Introduction

In the previous chapter we described attack surfaces to measure the security and vulnerabilities associated with operating systems. We extend this to determine the attack surfaces of sensor network communications protocols. This theory is extended to create a QoS surface for sensor networks. The attack surface and the QoS surface are created by considering two features of a sensor network, namely, sensor hardware, and routing (communication) protocols. This chapter describes the major concepts – architecture, routing protocols, attacks and QoS challenges related to sensor networks which will be considered to create the framework for the attack surface and the QoS surface.

We first consider the architecture of a sensor network.

3.2 Architecture

Sensor networks have one or more points of centralized control called base stations. A base station is typically a gateway to another network, a powerful data processing or storage center, or an access point for human interface. They can be used as a nexus to disseminate control information into the network or extract data from it. In some sensor network routing protocols, base stations have also been referred to as sinks.

Base stations are typically many orders of magnitude more powerful than sensor nodes. They might have workstation or laptop-class processors, memory, and storage, AC power, and high-bandwidth links for communication amongst themselves. However, sensor nodes are constrained to use lower-power, lower-bandwidth, shorter-range radios, and so it is envisioned that the sensor nodes would form a multi-hop wireless network to allow sensors to communicate to the nearest base station. Fig. 3.1 illustrates a representative architecture for sensor networks.

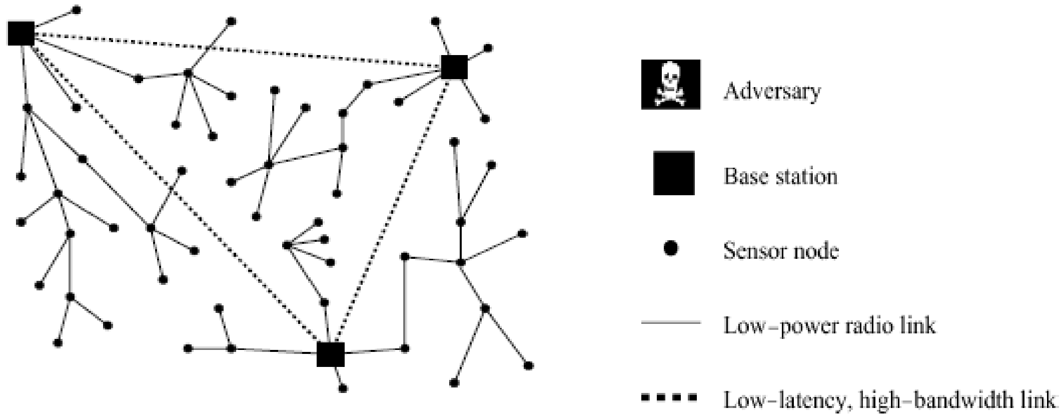

Figure 2: Architecture of a Sensor Network

A sensor network consists of a large number of devices (nodes) that may or may not be mobile and measure an environmental variable such as temperature, pressure, radioactivity, or geographic position. The sensors are equipped with transceivers that enable them to transmit and receive data to and from other sensors. Any change in the environment triggers the need for communication between a regular sensor, and the base station. A communication session is assumed to consist of sending a single packet of data from the sensor node to the powerful sensor (base station); this assumption is possible as the data of an environmental variable only consist of a few bytes. If a node sends out its data packet, chances are that the packet will not reach the destination in a single hop due to the limited transmission range of the source sensor node; in order to obtain a functioning system, other sensors that receive the packet transmitted by the source need to forward the data packet themselves until it reaches the destination, this mode of transmission makes use of the transmission range of the various intermediate nodes to forward the received packets to the destination.

Determining a solution about how to route a data packet has been the subject of intense study and research in ad-hoc networks. The attributes of a sensor network setting are

1. Limited battery power
2. High network node density
3. Frequent network topology changes

Whenever a source and a destination are given, most routing protocols—particularly in ad-hoc networks—first compute a routing path and then route data packets along the computed path. Each node in the network—upon receiving a packet—retransmits the packet to its neighbors with a certain probability according to the probability function [3]. While this basic principle is straight-forward, we must be careful not to overload the probability function: the retransmission probability depends on parameters as diverse as the numbers of copies of the packet already received by the sensor node or the distance to the destination. In order to keep the routing protocol as light-weight as possible, the retransmission probability is kept as simple as possible with respect to the complexity required to compute or estimate the parameters that the probability function depends on.

3.3 Routing Protocols

The routing protocols that can be employed on a sensor network can be broadly classified as multi path routing and single path routing protocols. Each protocol is provided with a payoff value (factor) to indicate its level of security. These pay off values are utilized in chapter 7, while measuring the attack and QoS surfaces.

3.3.1 Single Path Routing Protocols

The single path routing protocols make use of a single path of transmission from a particular sensor node to the base station. An example of such a routing path is shown in Figure 3

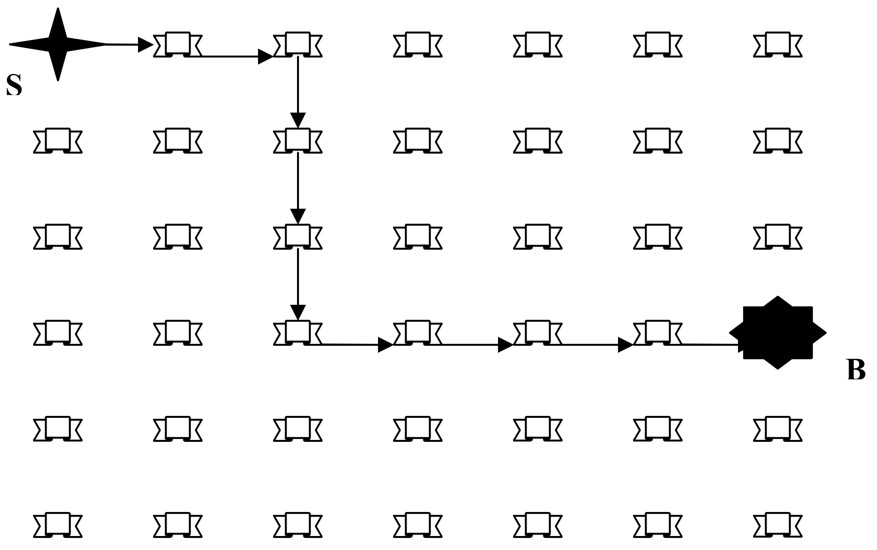

Figure 3: Single Path Routing Protocol

3.3.2 Multi Path Routing Protocols

The multi path routing protocols make use of multiple paths of transmission from a particular sensor node to the base station. The multi path routing protocols can be classified as:

- No link, no node sharing protocols (disjoint)
- Node sharing, no link sharing protocols
- Intertwined or Sharing protocols

Disjoint routing protocols

In this protocol the paths taken will be disjoint (unique). It will not consider the same nodes. Fig 4 shows an example of a disjoint routing protocol.

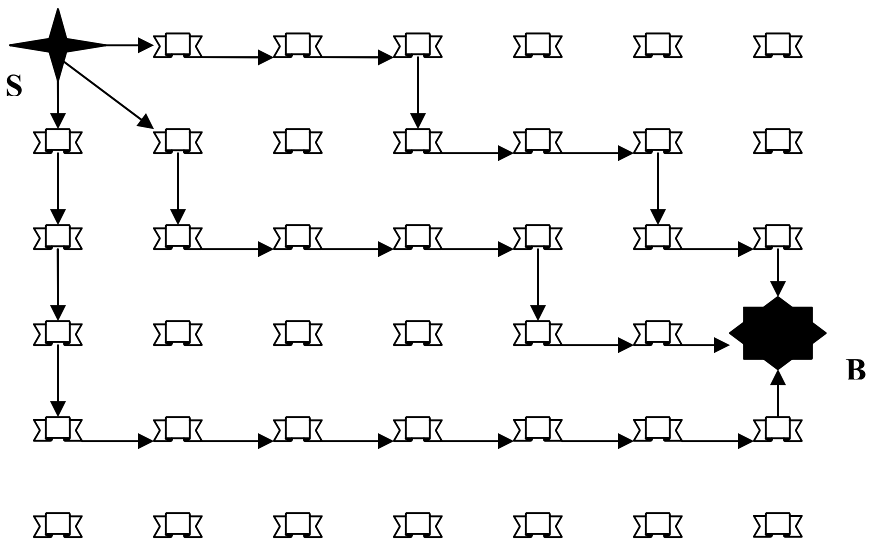

Figure 4: Disjoint Multi Path Routing Protocol

Node sharing, no link sharing protocols

In this protocol, the same nodes present in various paths can be used but the links (the sequence of transmissions is different) for each path. Figs 5 and 6 show an example of such a protocol.

(All previous nodes present as a subset in the new path with reference to the single path mode)

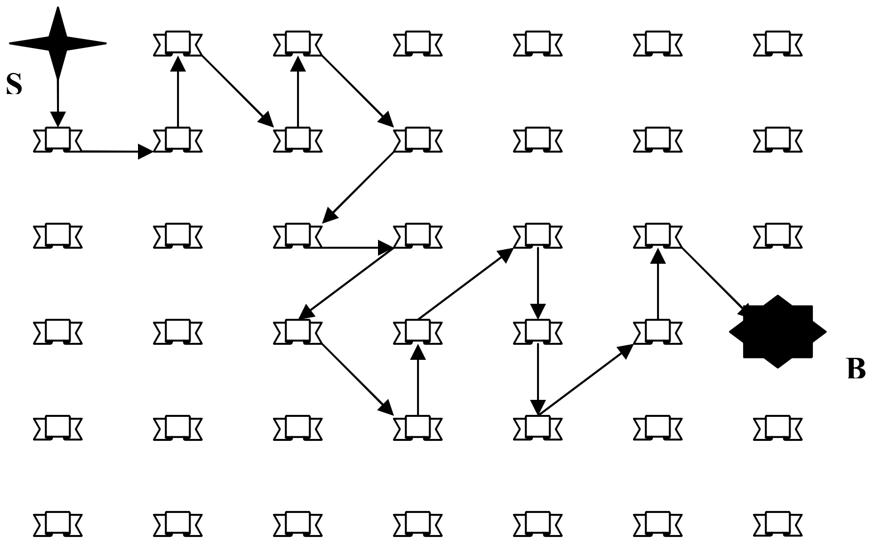

Figure 5: Node Sharing Multi Path Protocol

(A subset of the previous nodes present as a subset in the new path)

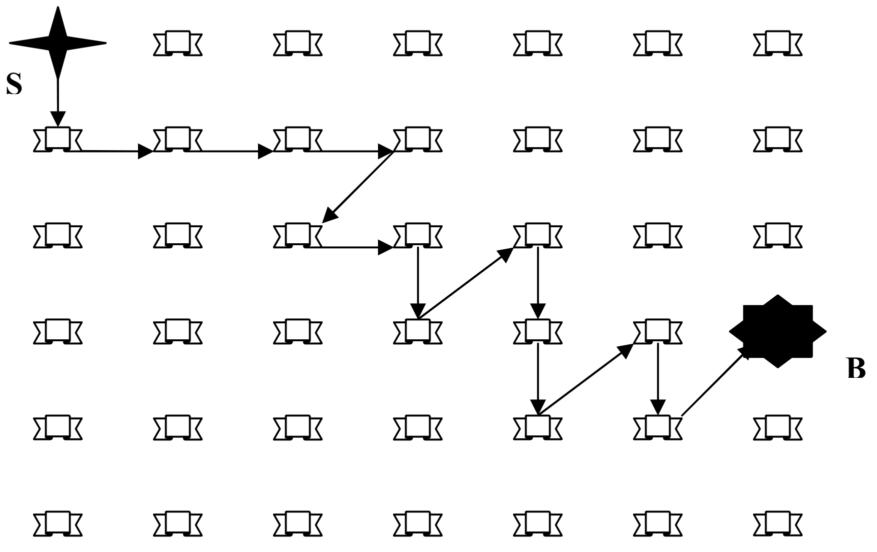

Figure 6: Node Sharing Multi Path Routing Protocol

Shared or intertwined paths

In this type of multi path routing protocols there is a sharing of nodes and links. Fig 7 illustrates such an example.

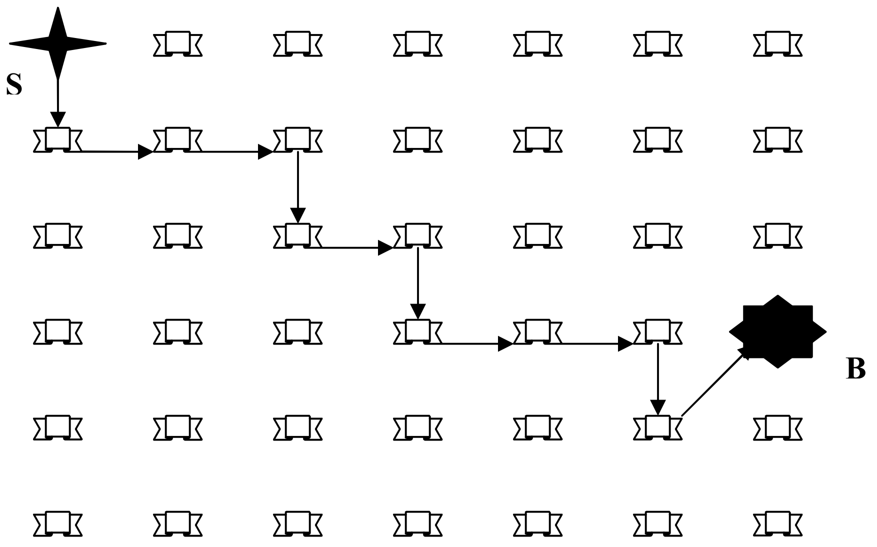

FIG 6: Shared or intertwined multi path routing protocol

Payoff Values

Payoff values for the above routing mechanisms:

- Disjoint paths – ‘a’
- No link, node sharing – ‘b’
- Intertwined – ‘c’

$$\mathbf{a \geq b \geq c}$$

Disjoint paths are most secure because an adversary can never know which node will be taken as the next node and which path will be taken. As the paths are disjoint there is no sharing of resources among all the paths so an adversary can never decide its course of action.

Node sharing paths are the next in line. The adversary can never determine what the next node in the path will be, thus hindering its process of attacking. There is a small probability associated with the scheme that it can be attacked more easily than disjoint paths. Following the traces of the previous path taken, if the nodes of that path are attacked then there is an increased possibility that future transmissions can be tampered with.

Intertwined paths are the most insecure methods. Since an adversary can follow the paths taken by making use of the links from the previous paths taken for transmission, this mode has the maximum susceptibility to attacks. By watching the nodes that were accessed in the previous transmission it is possible for the adversary to compromise those nodes, this increases the possibility of the failure of the whole network.

3.3.3 Types of network layer attacks

Most network layer attacks against sensor networks fall into one of the following categories:

- Spoofed, altered, or replayed routing information
- Selective forwarding
- Sinkhole attacks
- Sybil attacks
- Wormholes
- HELLO flood attacks

- Acknowledgement spoofing

Spoofed, altered, or replayed routing information

This is the most direct attack against a routing protocol. It targets the routing information exchanged between nodes. By spoofing, altering, or replaying routing information, adversaries may be able to attract or repel network traffic, extend or shorten source routes, create routing loops, partition the network, increase end-to-end latency, generate false error messages, etc.

Selective forwarding

Sensor networks (multi-hop networks) are often based on the assumption that participating nodes will faithfully forward received messages. A simple form of this attack is when a malicious node behaves like a black hole and refuses to forward every packet it receives. However, neighboring nodes might conclude that the node (black hole) has failed and seek another route. A more subtle form of this attack is when an adversary selectively forwards packets [4]. An adversary interested in limiting suspicions of it being compromised, can suppress or modify packets originating from a select few nodes and reliably forward the remaining traffic. An adversary launching a selective forwarding attack will follow the path of least resistance and attempt to include itself on the actual path of the data flow.

Sinkhole attacks

In a sinkhole attack, the adversary lures all traffic from a particular area to a compromised node, creating a metaphorical sinkhole with the adversary at the center. Sinkhole attacks can enable many other attacks (selective forwarding, for example).

Sinkhole attacks work by making a compromised node look especially attractive to surrounding nodes with respect to the routing algorithm. For instance, an adversary could spoof a high quality route to a base station. In this case, a laptop-class adversary with a powerful transmitter can actually provide a high-quality route (spoofed) by transmitting with enough power to reach the base station in a single hop.

Sensor networks are susceptible to sinkhole attacks due to their specialized communication pattern. All packets share the same ultimate destination (in networks with only one base station), so a compromised (malicious) node only has to provide a single high-quality route to the base station in order to influence a potentially large number of nodes.

The Sybil attack

In a Sybil attack [2], a single node presents multiple identities to other nodes in the network. The Sybil attack can significantly reduce the effectiveness of fault-tolerant schemes such as distributed storage [17], multi path [18] routing, and topology maintenance [20]. Routes believed to be using disjoint nodes could be actually using a single adversary presenting multiple identities. Sybil attacks also pose a significant threat to geographic routing protocols. Location aware routing often requires nodes to exchange coordinate information with their neighbors to efficiently route geographically addressed packets. It is only reasonable to expect a node to accept a single set of coordinates from each of its neighbors, but by using the Sybil attack an adversary can ‘‘be in more than one place at once’’ [4].

Wormholes

In the wormhole attack [1], an adversary communicates messages received in one section of the network over a low-latency link and replays them in a different section. An adversary situated close to a base station may be able to completely disrupt routing by creating a well-placed wormhole. An adversary could convince nodes who actually are multiple hops from the base station that they are only one or two hops away via the wormhole. This can create a sinkhole: since the adversary on the other side of the wormhole can artificially provide a high quality route to the base station, potentially all traffic in the surrounding area will be drawn through it if alternate routes are significantly less attractive. Fig. 8 shows an example of a wormhole being used to create a sinkhole.

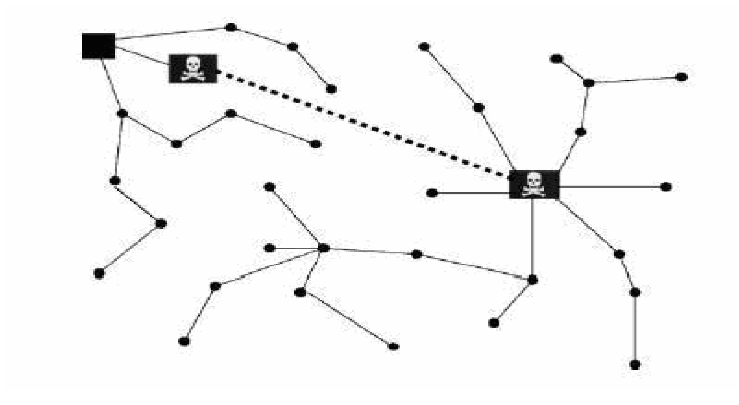

Figure 7: Adversary using a Wormhole to create a Sinkhole

More generally, wormholes can be used to exploit routing race conditions [4]. A routing race condition arises when a node takes some action based on the first instance of a message it receives and ignores later instances of that message. In such a case, an adversary may be able to exert some influence on the resulting topology if it can cause a node to receive certain routing information before it would normally reach them through multi hop routing. Wormholes are a way to do carry out this sort of an attack (routing race conditions). Wormholes can also be used simply to convince two distant nodes that they are neighbors by relaying packets between the two of them.

HELLO flood attack

Many protocols require nodes to broadcast HELLO packets to announce their presence as neighbors in the vicinity, and a node receiving such a packet concludes that it is within (normal) radio range of the sender. This assumption may be rendered false if a laptop-class attacker broadcasts routing or other information, with sufficient transmission power it might result in every node in the network being convinced that the attacker is its neighbor.

When an adversary advertises a very high-quality route to the base station (from its current position) to every other node in the network, the information could cause a large number of nodes attempt to use this high quality route. Thus nodes far away from the adversary would be sending packets into oblivion. The network is left in a state of confusion. Protocols which depend on localized information exchanged between the neighboring nodes of a sensor network for topology maintenance or flow control are subjected to this type of attack.

Acknowledgement spoofing

Several sensor network routing algorithms rely on link layer acknowledgements. An adversary can spoof link layer acknowledgments for ‘‘overheard’’ packets addressed to neighboring nodes. This spoofing of acknowledgements is to convince the sender that a weak link is strong or that a dead or disabled node is alive. For example, a routing protocol may select the next hop in a path using link reliability. Artificially reinforcing a weak or dead link is a subtle way of manipulating such a scheme [4]. Packets sent along weak or dead links are eventually lost; so, an adversary can mount a selective forwarding attack using acknowledgement spoofing by encouraging the target node to transmit packets on those links.

3.3.4 QoS Challenges

- **Bandwidth Limitation:**

  We need to decide upon the tradeoff with quality of service. Depending on this tradeoff we can split the traffic across various paths to meet the bandwidth requirements. The data packets are then aggregated to provide the data that was originally split.

- **Removal of redundancy**

  When data is transferred across the network, data aggregation can be performed to prevent the redundant data from being transmitted across all different paths. This method removing redundant data paves the way for efficient resource utilizations.

- **Energy Vs Delay Tradeoff**

  A 1-hop transfer of data can result in quick transfer of data but it is more energy consuming [14] [15] [16]. A single node usually will not be provided with the required amount of power to perform this 1-hop operation. When multiple hops are the only option in the network, we will be able to conserve energy. Energy is conserved using this method because the power present in a single node is sufficient to transfer data to its neighboring nodes rather than to a distant destination node. Hence, the power provided for a single node will last longer than when using a single hop strategy. Using this multi-hop strategy will result in a greater delay in the transfer of data.

- **Buffer size limitation**

  Buffering of multiple packets will aid in greater conservation of energy. When a node has to switch from receiver mode to sender mode it results in greater usage of power. So before switching modes it is an advantage if the packets that are directed towards it are received first. Though the above inference is advantageous it depends a lot on the buffer size. The buffer present in each individual node has to be sufficient to hold the aggregated data packets. If the buffer size is limited it will result in greater delay and much faster energy loss.

- **Multiple Data Types – support**

  Energy and QOS requirements have to be satisfied for various types of data. This is a necessary component because when there are different types of data that will have to be communicated it results in the need for supporting them all. This last factor is application dependent.

3.3.5 Threat models

There are two classes of attackers:

1. Mote-class attackers
2. Laptop-class attackers.

In the case of mote class attackers, the attacker has access to a few sensor nodes. In contrast, a laptop-class attacker has access to more powerful devices, like laptops. Thus, when a laptop class attacker is present, malicious nodes may have an advantage over legitimate, uncompromised nodes - greater battery power, more efficient CPU, high-power radio transmitter, sensitive antenna.

An ordinary sensor node has limited capabilities. Using its maximum resources it may be able to jam the radio link in its immediate vicinity, while a laptop-class attacker might be able to jam the radio link of the entire sensor network using its stronger transmitter. A single laptop-class attacker might be able to eavesdrop on an entire network, while sensor nodes would ordinarily have a limited range. Also, laptop-class attackers might have a high-bandwidth, low-latency communications channel not available to ordinary sensor nodes, allowing such attackers to coordinate their efforts [4].

A second distinction can be made between outsider attacks and insider attacks. In outsider attacks, the attacker has no special access to the sensor network. In insider attacks, an authorized participant in the sensor network turns bad (malicious). Insider attacks can be carried out from either compromised sensor nodes running malicious code or adversaries who have compromised legitimate nodes (obtain key material, data), then use one or more laptop-class devices to attack the sensor network.

3.3.6 Network assumptions

We assume that radio links are insecure in sensor networks. In the simplest mode of attack, attackers can eavesdrop on the radio transmissions, inject bits in the channel, and replay previously overheard packets. If the defender can deploy many sensor nodes, then the adversary will also be able to deploy a few malicious nodes with similar hardware capabilities as the legitimate nodes. The attacker may gain control by ‘‘turning’’ a few legitimate nodes to malicious nodes by capturing them and physically overwriting their memory [5].

The attacker may gain control of more than one node, and all the malicious nodes might collude to attack the system. In some cases, colluding nodes might have high-quality communications links available for coordinating their attack on the system. If an adversary compromises a sensor node (i.e. exploit the vulnerability present), it can extract all key material, data, and code stored on that compromised node. Providing extreme tamper resistance is a viable defense for physical node compromise in some networks but it cannot be considered as a general purpose solution, because tamper resistance tends to add a significant per-unit cost.

3.3.7 Trust requirements

Base stations interface a sensor network to the outside world. When a significant number of base stations are compromised, the scenario renders the entire network useless. So, we assume that base stations are trustworthy and behave correctly. Most, routing protocols require nodes to trust messages from base stations [4].

Aggregation points may be considered to be trusted components in certain protocols. Nodes relying on routing information from aggregation points trust that messages sent to the aggregation points will be accurately combined with other messages and forwarded to a base station. This assumption is not considered reasonable due to the fact that aggregation points are often regular sensor nodes. So, it is possible for adversaries to try and deploy malicious aggregation points or attempt to turn currently compromised nodes into aggregation points. So, aggregation points may not necessarily be trustworthy.

3.4 Security in sensor networks

A number of secure mechanisms have been proposed for sensor networks [4] [7] [8]. In particular we look at one key management scheme called LEAP [7] as we investigate the attack surfaces resulting from the introduction of LEAP key management into a routing protocol.

LEAP

LEAP (Localized Encryption and Authentication Protocol), is a key management protocol for sensor networks that is designed to support in-network processing. This protocol restricts the security impact of a node compromise to the immediate network neighborhood of the compromised node. LEAP supports the establishment of four types of keys for each sensor node – an individual key shared with the base station, a pair-wise key shared with another sensor node, a cluster key shared with multiple neighboring nodes, and a group key that is shared by all the nodes in the network

The constrained energy budgets and the limited computational and communication capacities of sensor nodes make protocols such as TLS [13] and Kerberos [13] [18] proposed for wired networks impractical for use in large scale sensor networks. At present, the most practical approach for bootstrapping secret keys in sensor networks is to use pre-deployed keying in which keys are loaded into sensor nodes before they are deployed.

The packets exchanged by nodes in a sensor network can be classified into several categories based on different criteria, e.g. control packets versus data packets, broadcast packets versus unicast packets, queries or commands versus sensor readings, etc. The security requirements for a packet will typically depend on the category it falls in. Authentication is required for all types of packets, whereas confidentiality may only be required for some types of packets. For example, routing control information usually does not require confidentiality, whereas (aggregated) readings reported by a sensor node and the queries sent by the base station may require confidentiality. We argue that no single keying mechanism is appropriate for all the secure communications that are needed in sensor networks.

KEYS of LEAP:

- **Individual Key**

  Every node has a unique key that it shares with the base station. This key is used for secure communication between the node and the base station. For example, a node can use its individual key to compute message authentication codes (MACs) for its sensed readings if the readings are to be verified by the base station. A node may also send an alert to the base station if it observes any abnormal or unexpected behavior of a neighboring node. Similarly, the base station can use this key to encrypt any sensitive information, e.g. keying material or special instruction, it sends to an individual node.

Every node has an individual key that is only shared with the base station. This key is generated and pre-loaded into each node prior to its deployment.

- **Group Key**

This is a globally shared key that is used by the base station for encrypting messages that are broadcast to the whole group. For example, the base station issues missions, sends queries and interests. From the confidentiality point of view there is no advantage to separately encrypting a broadcast message using the individual key of each node. However, since the group key is shared among all the nodes in the network, an efficient re-keying mechanism is necessary for updating this key after a compromised node is revoked.

- **Cluster Key**

A cluster key is a key shared by a node and all its neighbors, and it is mainly used for securing locally broadcast messages, e.g., routing control information, or securing sensor messages which can benefit from passive participation. Research has shown that in-network processing techniques, including data aggregation and passive participation are very important for saving energy consumption in sensor networks. For example, a node that overhears a neighboring sensor node transmitting the same reading as its own current reading can elect to not transmit the same. In responding to aggregation operations such as MAX, a node can also suppress its own reading if its reading is not larger than an overheard one. Clearly, for passive participation to be feasible, sensor nodes should be able to decrypt or verify some classes of messages, e.g., sensor readings, transmitted by their neighbors. This requires that such messages be encrypted or authenticated by a locally shared key. As such, LEAP provides each node a unique cluster key shared with all its neighbors for securing its messages. Its neighbors use the same key for decrypting or verifying its messages.

- **Pair-wise Shared Key**

  Every node shares a pair-wise key with each of its immediate neighbors. In LEAP, pair-wise keys are used for securing communications that require privacy or source authentication. For example, a node can use its pair-wise keys to secure the distribution of its cluster key to its neighbors, or to secure the transmission of its sensor readings to an aggregation node. Note that the use of pair-wise keys precludes passive participation.

3.5 Conclusion

This chapter provided the major concepts which will be referred to while creating the framework for the attack surfaces and QoS surfaces. The chapter describes the relationship between creating a security (or service) surface with the major properties of sensor networks. The following chapters illustrate the routing protocol functioning and the definitions for the dimensions of the respective surfaces.

CHAPTER IV

ATTACK SURFACE FOR WIRELESS SENSOR NETWORKS

4.1 Introduction

This chapter provides the parameters and features involved in our approach to model an attack surface for the sensor network system. The attack surface is modeled by determining the dimensions of the sensor network. We study the intrinsic details of the communication protocol by making use of state transition diagrams. This state transition diagram provides us with the surface model details – states, transitions, initial states, actions. The approach takes models of common attacks to indicate the vulnerabilities associated with the protocol.

The chapter deals with Wanderer protocol to describe the method of modeling the attack surface. Once the attack surface has been modeled, we will be able to identify the attacks that can happen in the sensor network. A dimension (rules of access) is enhanced in the Wanderer + LEAP protocol to study the vulnerabilities of the new protocol. A comparison is made between the two versions of the protocol to indicate the usefulness of our attack surface model in increasing the security features of the sensor network.

4.2 Wanderer Protocol

The wanderer protocol has been chosen to illustrate the creation of an attack surface of the sensor network as in section 3.2. The wanderer protocol is a simple protocol that enables us to deploy the idea of an attack surface on a sensor network. The simple features of this protocol are a result of the work carried out in [14] [15] [18].

4.2.1 Salient Features of the Wanderer protocol

1. Each node maintains a list of neighbors available and not the whole network topology.
2. Although packets are broadcasted, one neighbor is chosen from the list as the receiver and only that node retransmits the received packet. A random choice of a sensor node is made from the list of neighbor nodes to be designated as the receiver. This random choice is governed by the random function which is usually application-oriented.
3. The random choice is not always a good choice; number of retransmissions of the same packet will vary with the different choices obtained for the *receiver designated* using the random function. Sometimes, the taken path may be farther away from the destination, in which case the base station requests the packet again from the sensor node (this is actually based on a timeout value for the message request and receive operation).
4. A node may receive the same packet many times; each node keeps track of the number of times it has forwarded a particular packet. A number, called duplicate-forwarding parameter determines the number of times a packet can be forwarded

by a node. This duplicate forwarding parameter is set by the system design specifications.

4.2.2 Working of the Wanderer protocol

The intrinsic working details of the wanderer protocol are represented as a state transition diagram of the protocol. We have provided the protocol in a form similar to an automaton where we can see each stage the sensor network enters when there is transmission of data. Security issues will become apparent from the representation.

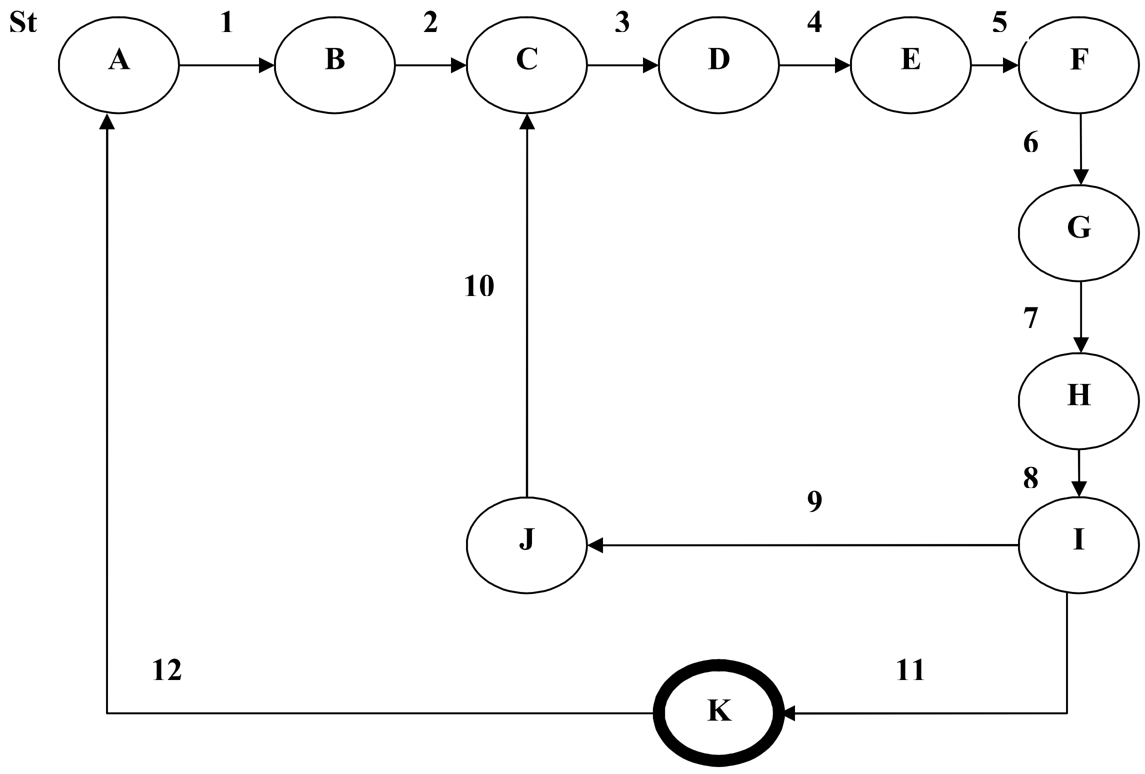

Figure 8: State Transition Diagram for Wanderer Protocol

States:

- **A**

  This is the state which represents the initial state of the sensor network. No transmissions take place in this state. The sensor network just waits for some events to take place so that transmission of data packets is necessary to report the events. The leaf nodes may also be waiting for a request message from a base station. Upon receiving a message from the base station the leaf node enters the next stage to transmit the requested information.

- **B**

  This is the stage which represents that the request message from the base station has been received by the leaf node. This is the stage, which also corresponds to the necessity of leaf nodes having to transmit data packets to report the occurrence of events to the base station. Usually there are intervals of time during which when certain events occur a leaf node will have to transmit data packets to the base station.

- **C**

  This is the stage where the leaf node has the required information for transmission. The information to be transmitted will be suitably wrapped into data packets in the following stages which will be interpreted by the base station as the occurrence of certain events.

- **D**

  This stage corresponds to the availability of the neighbor list in the leaf node. A neighbor list is obtained by the transmission of HELLO packets by each node to its neighbor nodes. A neighbor list is required so that the ‘receiver’ can be chosen to be the next node in the routing path.

- **E**

This stage is responsible for packing the information requested into packets. These packets can be interpreted by the base station to correspond to various scenarios and events.

- **F**

The receiver is determined using the random function from the neighbor list. Care is taken so that the sender is not again chosen to be the receiver else the packet may just loop across the two nodes without traversing in the forward direction. This is explained below.

- **G**

The ID of the source leaf node is also added to the packet to prevent the looping of the packets between the 2 nodes. The ID of the receiver chosen using the random function is stored along with the packet. This ID is added to the packet so that only the receiver can transmit the packets to its neighbors. This is the policy used in the wanderer protocol.

(Source ID + Receiver ID + Packet)

- **H**

The packet is transmitted to all the neighbor nodes present in the neighbor list. Each neighbor node receives the transmitted packet but confirms its identity as the receiver before transmitting the packet again.

- **I**

  The source node ID is stripped from the packet so that the receiver for the next stage of transmission will not be the source node, in order to prevent a loop of the source and neighbor node while making use of the receiver function.

- **J**

  The receiver has the packet ready for further transmission. All the IDs which were initially packed with the data packet are stripped off. Thus the packet is ready for transmission to the next node on the routing path. The packet is now added with the original source ID, the current node ID, the new Receiver node ID.

  (Source ID +Intermediate ID+ new Receiver ID + Packet)

- **K**

  This is the final stage of the routing protocol where in the sensor network, the required packet has been received at the base station. After the packet has been received at the base station the network enters the initial state A.

Actions:

1 – Message transmitted from the base station to the individual sensor node

2 – Information to be sent is collected

3 – Dynamic update of the neighbors list

4 – Attach the source ID

5 – Make use of the random function to determine the receiver

6 – Attach the receiver ID

7 – Transmit the packet to all neighbors

8 – Check if the current node is the receiver node ID

9 – Attach the node's ID as the current intermediate node ID

10 – The information is ready for transmission

11 – If the destination is the base station then the sensor network reaches its final state

12 – Timeout makes the base station, request for information from the sensor node, the base station after a network specific time interval, in the event of failure to receive the requested information, re – initiates the information request process.

4.2.3 Transmissions

- It is preferred to obtain a new node as a receiver; by using the random function i.e. priority is given to the new node rather than the immediate source (the node which transmitted the packet to the current active node in the first place). The immediate source is usually prevented from being the receiver; else there is a possibility of cycles present between the nodes which is not a desired facet of the protocol.
- At any instant of time there can only be one node with an active status. A node becomes the active node when it receives a packet from another node. The information of the immediate source is usually stored at each active node, so that it can be made use of while choosing a receiver.

4.2.4 Attacks on the Wanderer protocol

- Selective forwarding – a compromised node can selectively forward packets.
- Wormhole – between the receiver and the adversary so that sensitive information can be transferred to the adversary.

- HELLO – the adversary makes the receiver become a neighbor to all nodes by transmitting HELLO packets. This is made possible because of the huge amounts of power available for disposal from the adversary.
- Sybil attack – the adversary makes the compromised node a neighbor to all other nodes by posing different suitable identities. This increases the probability of this compromised node being chosen as the receiver using the random function.
- Sinkhole attack – the compromised nodes can be deployed as sinkholes by the adversary. Packets sent to the compromised nodes are lost. When the number of compromised nodes increases in a sensor network, the sinkhole attack can bring down the whole network.

A detailed description of the attacks has been provided in chapter 3.

4.2.5 Vulnerabilities

An attack is the sequence of events initiated by an adversary. The adversary aims to gather sensitive information or affect the normal working of the communication protocol in use. Vulnerability is the state of the sensor network that aids in the adversary mounting an attack. Let us consider the wanderer routing protocol and illustrate some of the vulnerabilities associated with the protocol.

- Each node stores a list of its neighbors. This list of neighbors is updated dynamically, when the node becomes an active node. This dynamic update enables the addition of new neighbor nodes to the list; it also introduces the problem - the adversaries can mount HELLO, Sybil attacks.

- The random function that determines the receiver from the set of neighbors is a possible vulnerability. This random function can be made obsolete by an adversary by mounting the Sybil attack (section 3.3.3). The adversary can also compromise the random function involved. A compromised node can have its own version of the random function which enables only compromised nodes available in the neighbors list to be selected as the receiver.
- There is a vulnerability associated with the trust placed on the receiver to transmit correctly. If the receiver had been compromised then it results in a total failure of the routing protocol in the sensor network.

4.2.6 Measure of vulnerability

Taking into consideration the attack surface automata introduced in Section 2.2, we obtain the following scenarios.

Vul = Behavior (Actual) – Behavior (Intend)

- **$S_a - S_i \neq 0$**

  S is the set of states present in a state transition diagram as presented by equation 2.1

  Any state S is a mapping from resources to their typed values.

  S: Resources → Values

  Resources refer to the physical hardware and its specifications available for each sensor node. Values refer to the actual data contained in each of the sensor node. A state is a representation of the interaction between the resources and values available.

Examples:

1. Message packet sent to adversary through the sinkhole/wormhole.

Route is the resource

The value is the actual route taken (sinkhole/wormhole or normal correct route)

2. A change in the neighbor list

Resource is neighbor list

Values is the neighbors in the list

3. The random function available for selecting the receiver is compromised.

Resource is the random function

Values is the result of the random function

For the above three examples we can see that when the protocol enters a state which requires a generation of a new neighbor list (Sa) when not intended for the correct functioning of the protocol (Si), we will be able to identify the difference in the functioning of the protocols due to the attacks mounted.

Sa is the set of states the protocol enters as a result of an attack being mounted upon and Si is the set of states the protocol will enter as a result of its secure functioning i.e. no attacks are mounted.

For example:

The intended set of neighbors in state D above is N_i

The actual set of neighbors in state D is N_a (this is a state D')

Therefore the set of intended states will be A,B,C,D,E … and actual states will be A’,B’,C’,D’….. The states are therefore different due to the different values of the resources.

Sa $\subseteq$ **Si** is a desired property. An illustration of the states Sa and Si is provided in section 4.5

- **Ia – Ii ≠ 0**

  I is the set of initial states present in a state transition diagram as presented by equation 2.2

  Examples:

  1. A change in the neighbors list makes the random function yield different results for a new action of transmission.
  2. When a node has been compromised it can make sure that it is always chosen as the initial state by mounting a Sybil attack.

  The initial actual neighbor list may be different from the intended list.

In the above examples we can see that when an attack is mounted and the random function has been compromised, only compromised nodes can be chosen as the receiver node (Ii). So the initial states differ for each invocation of the compromised random function. The correct functioning for the protocol will involve the uncompromised receiver nodes in the initial states (Ii). This provides a difference between the actual and the intended functioning of the protocol.

Ia is the set of states the random function chooses receivers as a result of an attack being mounted upon and Ii is the set of states the random function will choose as receivers because of its secure functioning i.e. no attacks are mounted.

- **$Aa – Ai \neq 0$**

  A is the set of actions present in a state transition diagram as presented by equation 2.3

  Examples:

  1. The communication with an adversary is always not intended.
  2. Requirement to change a neighbors list when not intended.
  3. The random selection function becomes obsolete
  4. The attachment of a wrong receiver id

Communication initiated between a receiver node and a compromised node when not intended (Aa) indicates the mounting of an attack, since this actual communication is not intended for the secure functioning of the protocol (Ai). The difference in such actions sets can help us to identify the security breaches made by an adversary on the sensor network. For example, attaching a wrong receiver id is indicative of a spoofing attack. Hence the list of intended versus actual becomes different.

Aa is the set of actions present in the execution of a protocol as a result of an attack being mounted upon and Ai is the set of actions present in the execution of a protocol because of its secure functioning i.e. no attacks are mounted.

- **$Ta – Ti \neq 0$**

  T is the set of transition functions present in a state transition diagram as presented by equation 2.4

  1. Action sets different
  2. Pre conditions / Post conditions for a common action are different.

Examples:

Precondition

Message available, ready for transmission, obtain the receiver information using the random function.

Post condition

Source id is stored, message ready for retransmission.

The above example refers to the state F in figure 8, when the packet has to be transmitted to the receiver chosen by the random function. When a sensor node has been compromised, and the sensor network enters the state F, the precondition is that the compromised node has been chosen as the receiver from the immediate source sensor node. The post condition is other compromised nodes are present with a higher probability of being chosen as the receiver by the random function.

Ti is the set of transitions which were intended for the secure functioning of the protocol and Ta is the set of transitions that resulted from the mounting of an attack. A transition function‘t’ is an element of Ti when it follows the normal functioning of the protocol. Some examples are when the uncompromised random function is used to identify the receiver node (the post conditions and preconditions present indicates the absence of an attack).

Ta is the set of transitions the protocol makes use of as it switches from one state to another as a result of an attack being mounted upon and Ti is the set of transitions; the protocol makes use of as a result of its secure functioning i.e. no attacks are mounted.

Usually the Intend operation of the protocols has lower bounds on the various parameters involved for the correct operation of the protocol. If the conditions/constraints present on the parameters for the Actual operation of the protocol are a subset of the Intend conditions then we can conclude that the working of the protocol is secure against attacks. If the conditions present for the Actual operation is a subset of the conditions present for the Intend operation, it is an ideal scenario where the actual working of the protocol is more secure than the intended working of the protocol. In the case of the wanderer protocol, if the preconditions of the Actual operation of the described example, was that of an uncompromised node being chosen as the receiver using the uncompromised random function, it is a subset of the conditions that will have to be satisfied for the Intended operation of the protocol. The Intended operation of the protocol comprises of all the conditions to be satisfied for the ideal secure working of the protocol – means to always chose an uncompromised node as the receiver.

The measure of vulnerability is the difference between the actual and intended states of the dimensions identified for the protocol in execution. This measure enables us to identify potential breaches of security.

Scenario VS Reality

Let us consider a scenario where ‘n’ attacks are mounted on a sensor network. Once the ‘n’ attacks have been mounted, it results in the total failure of the system. In reality, it indicates that the pre and post conditions involved for an attack ‘I’ to be mounted are available in the sensor network before the attack is mounted as shown below.

Scenario:

Attack 1 → Attack 2 → Attack 3 → ...Attack ‘n’ → Total Failure of the system.

Reality:

Pre A1 → A1 → Post A1, Pre A2 → A2 → Post A2, Pre A3 → ...Post ‘n-1’,Pre ‘ n’ → A ’n’ → Post ‘n’ → Failure.

In the case of the wanderer protocol,

Scenario:

HELLO ➔ Sinkhole ➔ Wormhole ➔ Total Failure of the system

Reality:

$\text{Pre}_{\text{HELLO}}$ ➔ HELLO ➔ $\text{Post}_{\text{HELLO}}$, $\text{Pre}_{\text{Sinkhole}}$ ➔ Sinkhole ➔ $\text{Post}_{\text{Sinkhole}}$, $\text{Pre}_{\text{Wormhole}}$ ➔ Wormhole ➔ $\text{Post}_{\text{Wormhole}}$ ➔ Total Failure of System

4.2.7 Dimensions of an attack surface

The earlier sections of the chapter depict the various features of the wanderer protocol that will have to be considered while creating an attack surface. This chapter illustrates the dimensions of the attack surface for the sensor network employing the Wanderer protocol. We have identified three dimensions to create an attack surface for the sensor network. The features that are vulnerable or secure against attacks have been classified according to the dimensions identified.

Targets and Enablers

If there are n sensor nodes available in the network then we have n targets and enablers.

Any one node can be chosen as the target and the remaining set of nodes ‘en’ contained in the set of the remaining (n-1) nodes can be used as enablers. The enablers are made use of to compromise the target node – change the random function, change the neighbors list, wormhole, sinkhole, Sybil attacks. An adversary can make use of the neighboring nodes (enablers) of the target node to propagate its compromised random function as the new random function to be used for future transmissions.

A compromised node can be used to compromise the other nodes which will result in total failure of the system.

Channels and Protocols

Radio signals are transmitted and received by ordinary message passing mechanisms. The n sensor nodes are considered to be of the same type with similar configurations while the base station has a different configuration.

A new channel can be opened or set up by the adversary under such circumstances then there are chances of wormhole, sinkhole attacks.

Protocols

- The source of the packet is stored along with the packet during transmissions.
- The source id is stripped from the packet.
- The source id is stored to be used in the random function so that a new node can be chosen as the receiver.

Rules of Access (Protection layer):

This is the new dimension that has been added to characterize the attack surface of a sensor network.

- Nodes with the ability to receive and transmit packets on a particular frequency are allowed access to the sensor network.
- Geographical distance between the node and the sensor network places an important access rule on the node.
- Encryption/Decryption of messages acts as a protection layer against attacks. This rule of access is not present in the Wanderer protocol, but it will become clear in the following section of the Wanderer + LEAP (Localized Encryption and Authentication Protocol) [7], a key management protocol for sensor networks was

designed to support in-network processing, while at the same time restricting the security impact of a node compromise to the immediate network neighborhood of the compromised node)

- Ability to perform the required operations (application oriented) is also one access rule. (There is a possibility that an adversary can pose as a component node till an attack has been mounted).

4.3 Threat Models

This section illustrates the attacks possible on a sensor network deploying the Wanderer protocol. Making use of the transition diagrams from a high level perspective, we will be able to identify the vulnerabilities associated with the sensor network. We take into consideration the common attacks possible on a sensor network. The threats that have been illustrated are:

- HELLO and Sybil attacks
- Sinkhole attacks
- Wormhole attacks
- Selective forwarding

4.3.1 Wanderer Protocol

A high level transition diagram is provided for the wanderer protocol. This diagram provides us with a simplistic overview of the Wanderer protocol in action. This high level diagram enables us to understand and identify the vulnerabilities associated with the protocol. The identification is performed by matching the states of the high level diagram with the transition diagrams of the attack models.

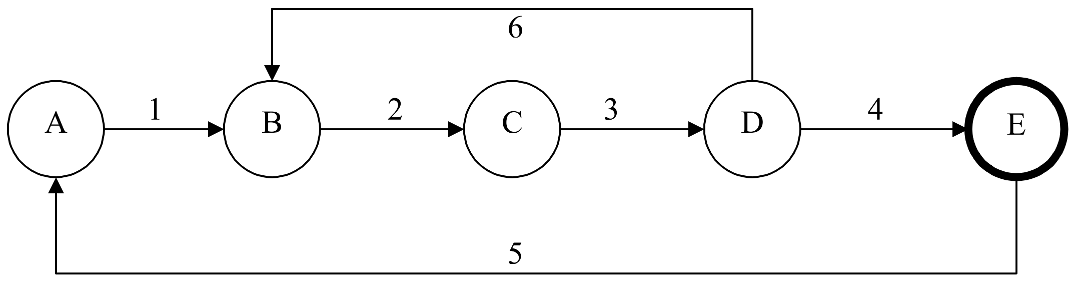

Figure 9: High Level Transition Diagram for Wanderer Protocol

States:

A – The initial state of the sensor network

B – Transmission of packets has to be initiated

C – Neighbor node list is available for choosing the receiver

D – The packet is transmitted to the receiver

E – The packet reaches the destination (base station)

Actions:

1 – The base station has requested information from the sensor nodes

2 – The neighbor list is created

3 – The receiver is chosen from the neighbor list using the random function available at the node

4 – The final transmission of the packet to the base station

5 – After transmission of the packet the sensor network enters the initial stage A

6 – Intermediate transmissions between the sensor nodes, before the packet reaches the base station

4.3.2 HELLO / Sybil attack model

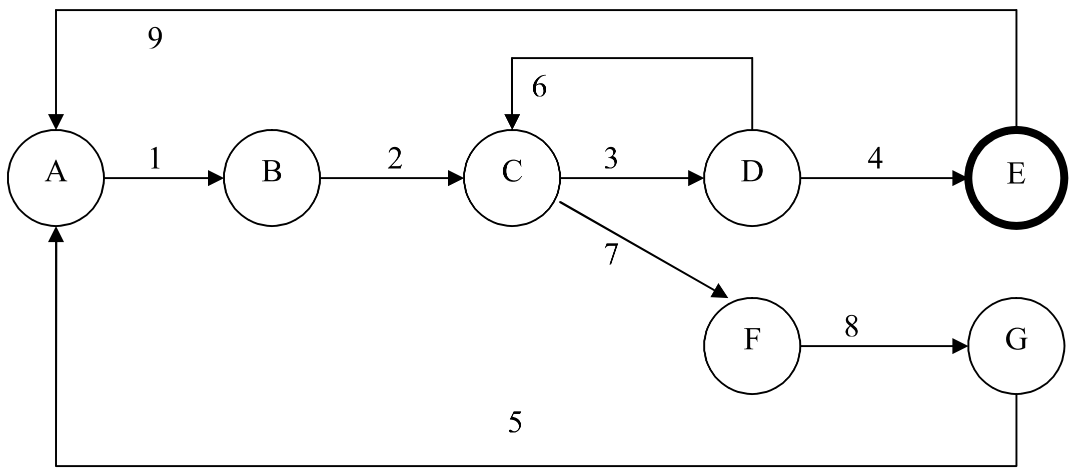

Figure 10: HELLO/ Sybil Attack Model

States:

A – The initial state of the sensor network, packets have to be transmitted

B – HELLO/ Sybil attack so compromised nodes are present to be available on the neighbor list

C – Neighbor node list is available for choosing the receiver, which may have compromised nodes from the Sybil/HELLO attacks

D – The packet is transmitted to the receiver, an uncompromised node has been chosen as the receiver

F – The packet is transmitted to a compromised node that has been chosen as the receiver

G – Selective forwarding or sinkholes, wormholes affect the normal functioning of the network.

E – The packet reaches the destination (base station)

Actions:

1 – The base station has requested information from the sensor nodes

2 – Compromised nodes relay HELLO packets to indicate their false presence as neighbors to the source node.

3 – The neighbor list is created. The receiver is chosen from the neighbor list using the random function available at the node

4 – The final transmission of the packet to the base station

5 – After mounting an attack, the sensor network is allowed to return to its original state so that suspicions are not raised.

6 – Intermediate transmissions between the sensor nodes, before the packet reaches the base station

7 – A compromised node is chosen by the random function to be the receiver

8 – The packet is transmitted to the compromised receiver

9 – After transmission of the packet the sensor network enters the initial stage A

4.3.3 Sinkhole attack model

Sinkholes are mounted by HELLO/Sybil attacks. The compromised nodes are made to act as sink holes so that the data packets are dropped or selectively forwarded to avoid suspicion. Over a period of time all the nodes in the network can be compromised to bring down the whole network. The compromised nodes are present as different nodes in the neighbor list thus affecting the normal functioning of the random function.

States:

A – The initial state of the sensor network, packets have to be transmitted

B – HELLO/ Sybil attack so compromised nodes are present to be available on the neighbor list

C – Neighbor node list is available for choosing the receiver, which may have compromised nodes from the Sybil/HELLO attacks

D – The packet is transmitted to the receiver, an uncompromised node has been chosen as the receiver

F – The packet is transmitted to a compromised node that has been chosen as the receiver

G – Sinkholes thus created affect the normal functioning of the network.

E – The packet reaches the destination (base station)

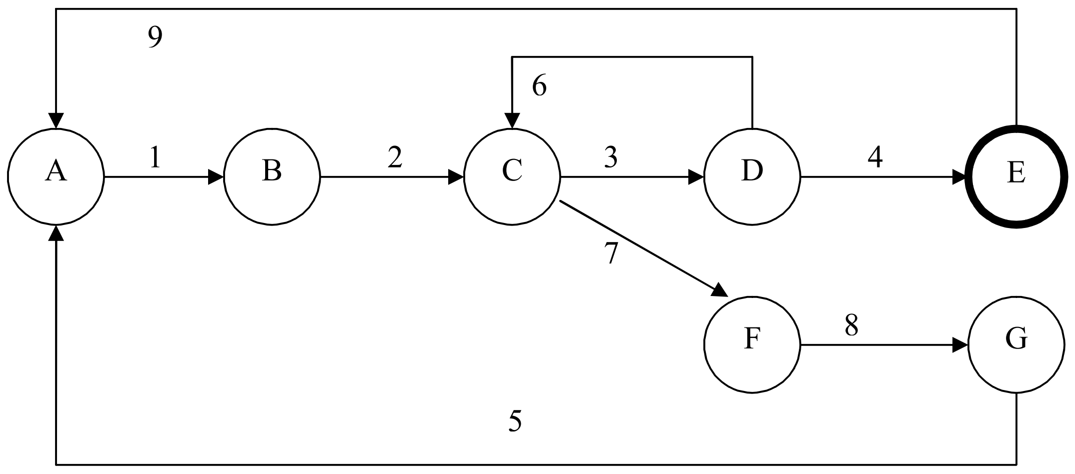

Figure 11: Sinkhole Attack Model

Actions:

1 – The base station has requested information from the sensor nodes

2 – Compromised nodes relay HELLO packets to indicate their false presence as neighbors to the source node.

3 – The neighbor list is created. The receiver is chosen from the neighbor list using the random function available at the node

4 – The final transmission of the packet to the base station

5 – After mounting an attack, the sensor network is allowed to return to its original state so that suspicions are not raised.

6 – Intermediate transmissions between the sensor nodes, before the packet reaches the base station

7 – A compromised node is chosen by the random function to be the receiver

8 – The packet is transmitted to the compromised receiver

9 – After transmission of the packet the sensor network enters the initial stage A

4.3.4 Wormholes threat model

Wormholes are mounted by HELLO/Sybil attacks. The compromised nodes are made to act as ends of the wormhole so that the data packets are routed to adversaries. Over a period of time all the nodes in the network can be compromised to bring down the whole network [19]. The compromised nodes are present as different nodes in the neighbor list thus affecting the normal functioning of the random function.

States:

A – The initial state of the sensor network, packets have to be transmitted

B – HELLO/ Sybil attack so compromised nodes are present to be available on the neighbor list

C – Neighbor node list is available for choosing the receiver, which may have compromised nodes from the Sybil/HELLO attacks

D – The packet is transmitted to the receiver, an uncompromised node has been chosen as the receiver

F – The packet is transmitted to a compromised node that has been chosen as the receiver

G – Sinkholes thus created affect the normal functioning of the network.

E – The packet reaches the destination (base station)

Actions:

1 – The base station has requested information from the sensor nodes

2 – Compromised nodes relay HELLO packets to indicate their false presence as neighbors to the source node.

3 – The neighbor list is created. The receiver is chosen from the neighbor list using the random function available at the node

4 – The final transmission of the packet to the base station

5 – After mounting an attack, the sensor network is not allowed to return to its original state. Any transmission will consider the two ends of the wormhole as neighbors. So finally all transmissions will be routed to an adversary through the wormhole.

6 – Intermediate transmissions between the sensor nodes, before the packet reaches the base station

7 – A compromised node is chosen by the random function to be the receiver

8 – The packet is transmitted to the compromised receiver.

9 – After transmission of the packet the sensor network enters the initial stage A

10 – After a wormhole has been established the sensor network is allowed to reach its initial stage to avoid suspicion thereby getting access to future transmissions.

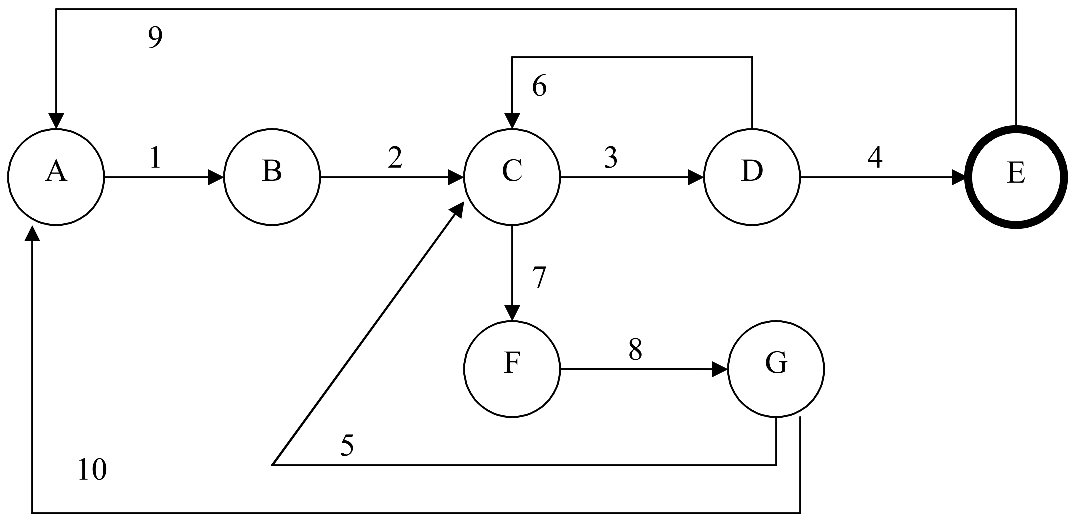

Figure 12: Wormhole Attack Model

In section 4.5, the attack surface model and its feasibility can be easily understood.

Whenever an attack is mounted on the wanderer protocol, the random function present at each compromised node is modified. Any attack on the sensor network deploying the wanderer protocol is aimed at compromising the random function. By compromising the random function, an adversary can gain access to sensitive information being transmitted across the sensor network. Section 4.5 describes the identification process of the attacks possible and vulnerabilities present in the sensor network.

Let us consider the case where a HELLO attack has been mounted on the wanderer protocol. Considering figures 9, 10 we can identify the intended states and the actual states of such an attack.

Intended States – A, B, C, D, E

Actual States – A, B, C, D, E, $(A, B, C, D, E, F, G)_{\text{HELLO}}$

Intended Initial States – B

Actual Initial States – B, A_{HELLO}

Intended Actions – 1, 2, 3, 4, 5, 6

Actual Actions – 1, 2, 3, 4, 5, 6, $(1, 2, 3, 4, 5, 6, 7, 8, 9)_{HELLO}$

Intended Pre Conditions – HELLO packets available from the neighbors

Intended Post Conditions – Neighbor list is created to be used by the random function

Actual Pre Conditions – HELLO packets are sent by the compromised nodes and adversaries

Actual Post Conditions – Compromised node's probability of being chosen as receiver is increased.

In a similar manner we can identify the intended and actual states, actions, conditions for a protocol and the mounting of a sinkholes attack or a wormhole attack by an adversary.

4.4 Attack surface for a wireless sensor network with Wanderer routing protocol utilizing LEAP

We consider the usage of LEAP along with wanderer protocol to illustrate the importance of the theory of attack surfaces for sensor networks. This section follows the same outline as was provided in section 4.2

4.4.1 LEAP

LEAP (Localized Encryption and Authentication Protocol), is a key management protocol for sensor networks that is designed to support in-network processing. This protocol restricts the security impact of a node compromise to the immediate network neighborhood of the compromised node. The design of the protocol is motivated by the observation that different types of messages exchanged between sensor nodes have different security requirements, and that a single keying mechanism is not suitable for meeting these different security requirements. The four encryption / decryption keys are explained in detail in chapter 3.

The pair-wise keys are established by the communication of two nodes taking into consideration that an attack has a lower bound on the time (T_{att}), while the communication time (T_{com}) is lower.

$$\mathbf{T_{com} < T_{att}}$$

4.4.2 Salient Features of the protocol:

The salient features of the wanderer + Leap protocol is similar to that of the wanderer protocol illustrated in section 4.2.1.

This protocol provides a security encryption/decryption mechanism (LEAP) in addition to the features provided by the wanderer protocol.

4.4.3 Working of the Protocol:

The intrinsic working details of the wanderer protocol with LEAP can be seen from a suitable working diagram. From the representation, we will be able to understand the frailties present in the protocol. Security issues will become apparent from the representation. This is similar to the automaton seen in the previous section with additional processing stages.

States:

- **A**

  This is the state which represents the initial state of the sensor network. No transmissions take place in this state. This is the state where the individual keys are pre –deployed for each of the sensor nodes before the sensor nodes are placed in the testing environment. The sensor network just waits for some events to take place so that transmission of data packets is necessary to report the events. The leaf nodes may also be waiting for a request message from a base station.

- **A1**

  Upon receiving a message from the base station the leaf node enters the next stage to transmit the requested information. The message obtained from the base station is decrypted with the individual key already assigned in stage A. This is done as per LEAP.

- **A2**

  Pair wise keys are established so that cluster keys and group keys can be established as per LEAP. This is the stage where all the keys are for available for the entire network.

- **B**

  This is the stage which represents that the request message from the base station has been received by the leaf node. All security constraints have been satisfied as per LEAP usage of the individual key. This is the stage, which also corresponds to the necessity of leaf nodes having to transmit data packets to report the occurrence of events to the base station.

- **C**

This is the stage where the leaf node has the required information for transmission. The information to be transmitted will be suitably wrapped into data packets in the following stages which will be interpreted by the base station as the occurrence of certain events.

- **C1**

This message is encrypted with the use of cluster keys which is common for neighbor nodes. The encrypted message is ready to be transmitted to the 'receiver' chosen from the list of neighbors.

- **D**

This stage corresponds to the availability of the neighbor list in the leaf node. A neighbor list is required so that the 'receiver' can be chosen to be the next node in the routing path.

- **E**

This stage is responsible for packing the encrypted information requested into packets. These packets can be interpreted by the base station to correspond to various scenarios and events.

- **F**

The receiver is determined using the random function from the neighbor list. Care is taken so that the sender is not again chosen to be the receiver else the packet may just loop across the two nodes without traversing in the forward direction. This is explained in the next section.

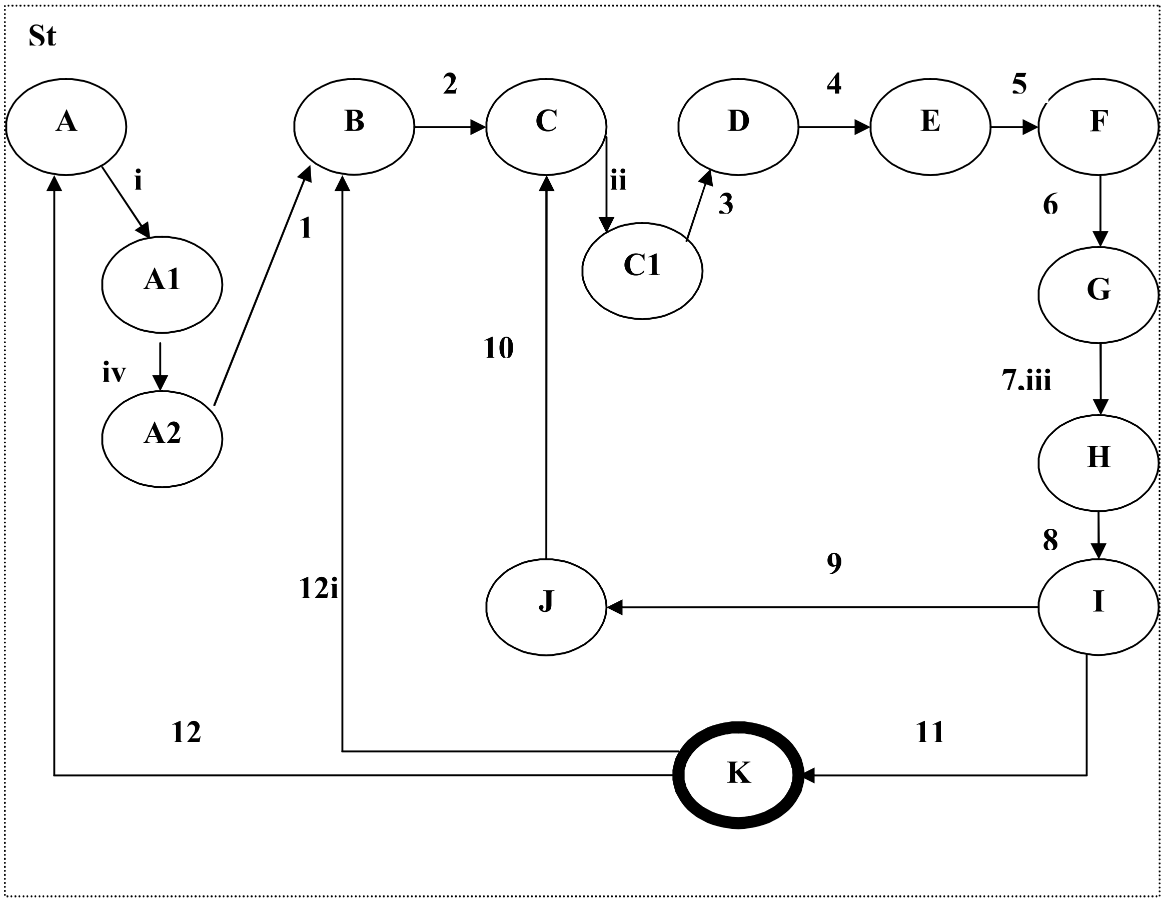

Figure 13: Wanderer Protocol + LEAP Model

- **G**

The ID of the source leaf node is also added to the packet to prevent the looping of the packets between the 2 nodes. The ID of the receiver chosen using the random function is stored along with the packet. This ID is added to the packet so that only the receiver can transmit the packets to its neighbors. This is the policy used in the wanderer protocol.

(Source ID + Receiver ID + Packet)

- **H**

The packet is transmitted to all the neighbor nodes present in the neighbor list. Each neighbor node receives the transmitted packet but confirms its identity as the receiver before transmitting the packet again. The transmitted packet is decrypted using the cluster key mechanism as explained in LEAP.

- **I**

The source node ID is stripped from the packet so that the receiver for the next stage of transmission will not be the source node. The receiver node is taken from the packet so that only the receiver node has the packet ready for the next stage of transmission.

- **J**

The receiver has the packet ready for further transmission. All the IDs which were initially packed with the data packet are stripped off. Thus the packet is ready for transmission to the next node on the routing path. The packet is now added with the original source ID, the current node ID, the new Receiver node ID.

(Source ID +Intermediate ID+ new Receiver ID + Packet)

- **K**

This is the final stage of the routing protocol where in the sensor network, the required packet has been received at the base station. After the packet has been received at the base station the network enters the initial state A or B. It enters A when new sensor nodes have been deployed. It enters B in the case of the same sensor nodes being present in the sensor network.

Actions:

1 – Message transmitted from the base station to the individual sensor node

2 – Information to be sent is collected

3 – Dynamic update of the neighbors list

4 – Attach the source ID

5 – Make use of the random function to determine the receiver

6 – Attach the receiver ID

7 – Transmit the packet to all neighbors

8 – Check if the current node is the receiver node ID

9 – Attach the node's ID as the current intermediate node ID

10 – The information is ready for transmission

11 – If the destination is the base station then the sensor network reaches its final state

12 – Timeout makes the base station, request for information from the sensor node, if new nodes have been deployed then the keys will have to be assigned for the new nodes.

12i – There are no new sensor nodes deployed so the same set will have to be used for the next transmission.

i. Pre-Deployment of Individual keys for direct communication between the base station and the sensor node
ii. Encryption of the requested information to be transmitted.
iii. Decryption of the requested information received.
iv. Pair-wise keys, cluster keys, group keys are established.

4.4.4 Transmissions:

- It is preferred to obtain a new node as a receiver; by using the random function i.e. priority is given to the new node rather than the immediate source (the node which transmitted the packet to the current active node in the first place). The immediate source is usually prevented from being the receiver; else there is a possibility of cycles present between the nodes which is not a desired facet of the protocol.
- At any instant of time there can only be one node with an active status. A node becomes the active node when it receives a packet from another node. The information of the immediate source is usually stored at each active node, so that it can be made use of while choosing a receiver.
- The keys are generated before any transmission takes place. The individual keys are pre deployed for each sensor node. The pair-wise keys are established and the using it the cluster and group keys are established as per LEAP.
- The pair-wise keys that are generated between the nodes are erased after a time limit (a design parameter) so that any compromised node can be removed by simply dropping the keys generated.
- An adversary will be able to access the keying materials but it will not be able to inject or decrypt the messages that are being transmitted in the network. This will be clear when analyzing the threat models.
- HELLO / Sybil attacks which are possible in the wanderer protocol can be prevented by the security mechanism provided by LEAP. The initial attack by any adversary on the wanderer protocol is carried out by the HELLO / Sybil attacks.

Since, the use of LEAP reduces the risk of HELLO attacks being mounted; we will be able to reduce the risks of sinkhole and wormhole.

- Selective forwarding is prevented by the need to synchronize the keys with the base station at frequent intervals. If a node has been compromised, it will not be able to resynchronize its keys with its neighbors, group or the base station. Thus the risk of selective forwarding being mounted is reduced.

This is illustrated by the notion of attack surfaces in the following sections.

4.5 Example of Attack Surface

The previous sections 4.2, 4.3, 4.4 provided us with the transition diagrams for the two variants of the wanderer protocol along with the attacks – hello, sinkhole, wormhole. This example illustrates the property of attack surfaces with reference to the previously explained variants of the wanderer protocol.

We model the system, the attack model as state machines:

$$System = (S_S, I_S, A_S, T_S)$$

$$Threat = (S_T, I_T, A_T, T_T)$$

4.5.1 Wanderer Protocol

From the transition diagram Fig 9, provided in section 4.2, we have

- $S_{wanderer}$, = {set of states}

  = {A,B,C,D,E,F,G,H,I,J,K}

- $I_{wanderer}$ = {set of initial states}

  = {A}

- $A_{wanderer}$ = {set of actions}

  = {1,2,3,4,5,6,7,8,9,10,11,12}

- $T_{wanderer}$ = {set of transition relations}

  = {(A,1,B),(B,2,C),(C,3,D),(D,4,E),(E,5,F),(F,6,G),(G,7,H),(H,8,I),(I,9,J),(I,11,K),(K,12,A)}

4.5.2 Wanderer + LEAP Protocol

From the transition diagram Fig 13 provided in the section 4.4, we have

- $S_{wanderer+LEAP}$ = {set of states}

  = {A,A1,A2,B,C,C1,D,E,F,G,H,I,J,K}

- $I_{wanderer+LEAP}$ = {set of initial states}

  = {A}

- $A_{wanderer+LEAP}$ = {set of actions}

  = {1,2,3,4,5,6,7,8,9,10,11,12,i,ii,iii,iv,12i}

- $T_{wanderer+LEAP}$ = {set of transition relations}

  = {(A,i,A1),(A1,iv,A2),(A2,1,B),(B,2,C),(C,ii,C1),(C1,3,D),(D,4,E),(E,5,F),(F,6,G),(G,7,H),(H,8,I),(I,9,J),(I,11,K),(K,12,A),(K,12i,B)}

4.5.3 Attacks

Let us see the individual sets for each attack model.

- **HELLO/Sybil Attack (Fig 10)**

  - S_{Ti} = {set of states}

    = {A,B,C,D,E,F,G}

  - I_{Ti} = {set of initial states}

    = {A}

- A_{Ti} = {set of actions}

  = {1,2,3,4,5,6,7,8,9}

- T_{Ti} = {set of transition relations}

  = {(A,1,B),(B,2,C),(C,3,D),(D,4,E),(C,7,F),(F,8,G),(D,6,C),(G,5,A), (E, 9, A)}

- **Sinkholes Attack (Fig 11)**
  - S_{Ti} = {set of states}

    = {A,B,C,D,E,F,G}

  - I_{Ti} = {set of initial states}

    = {A}

  - A_{Ti} = {set of actions}

    = {1,2,3,4,5,6,7,8,9}

  - T_{Ti} = {set of transition relations}

    = {(A,1,B),(B,2,C),(C,3,D),(D,4,E),(C,7,F),(F,8,G),(D,6,C),(G,5,A), (E, 9, A)}

- **Wormholes Attack (Fig 12)**
  - S_{Ti} = {set of states}

    = {A,B,C,D,E,F,G}

  - I_{Ti} = {set of initial states}

    = {A}

  - A_{Ti} = {set of actions}

    = {1,2,3,4,5,6,7,8,9,10}

  - T_{Ti} = {set of transition relations}

= {(A,1,B),(B,2,C),(C,3,D),(D,4,E),(C,7,F),(F,8,G),(D,6,C),(G,5,C),

(E, 9, A), (G, 10, A)}

Definition 1:

A protocol P is vulnerable to an attack i if there exist a set of preconditions pre-cond on a set of states $S_s \subseteq S_P$ *which when satisfied leads to states* S_s *where* $S_s = I_{Ti}$ *and the set of post-conditions post-cond required to launch an attack are met.*

S_s is the set of states that the protocol *P* enters as a result of the attacks mounted. S_p is the set of states that the protocol *P* enters in its secure mode of execution. I_{Ti} is the set of initial states from which the protocol starts its new cycle of transmitting information packets.

When a packet transmission makes the sensor network enter one of the states of *Ss*, where that state is an initial state I_{Ti} (the pre conditions and post conditions are satisfied for an attack to be mounted).

Wanderer Protocol:

For example, in the wanderer protocol the pre-conditions that have to be satisfied are the action to request the neighbor list. This leads to state B which becomes vulnerable to an attack only if in the post-condition the presence of compromised nodes is met. If the Pre-conditions are not satisfied, the node will never enter that state. Similarly, if the post-conditions are not satisfied, the node is not vulnerable to an attack.

Using this definition, we see that the state B in the protocol transition diagrams figure 9 is the vulnerable state for the HELLO/Sybil (figure 10), sinkhole (figure 11) and wormhole attacks (figure 12).

B in the wanderer protocol is equivalent to A of the attack models as explained in the attack models section 4.3.

As per the definitions for Ss, Sp, I_{Ti}, we have the following from Fig 5 and the attack models

$$S_s = \{B\}$$

$$S_p = \{B\}$$

$$I_{Ti} = \{A\}$$

Wanderer + Leap Protocol:

Likewise, from Fig 9 and the attack models

$$S_s = \{B\}$$

$$S_p = \{B\}$$

$$I_{Ti} = \{\}$$

The set I_{Ti} is empty because the preconditions and post conditions for a state Ss did not match with any of the states of I_T. Thus, we can see the feasibility of this attack surface model in identifying vulnerabilities and defending against attacks.

With LEAP available, the state B is made robust i.e. less vulnerable to the attacks namely, HELLO/Sybil attack, sinkhole attack, wormhole attack.

Definition 2:

A protocol P is vulnerable to a series of multiple attacks 1, 2, ..., n at the same time if there exists a set of pre-conditions that the lead to states $S_s \subseteq S_P$ such that $S_s = I_{T1}$ and for each attack $i \neq j$, there exists a set of states $S_{si} \subseteq S_{Ti}$ such that $S_{si} = I_{Tj}$ and the set of attack post-conditions are satisfied.

Usually, a protocol is vulnerable to only one attack – a protocol is usually not vulnerable to a series of attacks at the same time.

In the Wanderer protocol it is possible for attacks to take place in sequence. The HELLO/Sybil attack is mounted first which can then be used as pre conditions for the sink hole or the wormhole attack.

If the adversaries are different with unique motives then multiple attacks can be mounted. One adversary can mount a HELLO attack while the other at the same time can indulge in a jamming attack which works on the physical layer of the network.

With the use of LEAP as a third dimension it is not possible for multiple attacks to take place at the same time on the Wanderer protocol. The presence of different keys present for different types of messages reduces the vulnerability associated with the wanderer protocol.

4.6 Conclusion

An approach of creating an attack surface for sensor networks to study the secure and vulnerable features has been proposed in this chapter. Some defense mechanisms have been identified for various attacks possible on sensor networks. The mechanisms can be deployed if they can be implemented. Implementation of these mechanisms depends on the hardware and software details of the sensor network.

Making use of the dimensions of an attack surface created, we are able to measure and increase the security features of the sensor network. This method of fine tuning the dimensions is explained in chapter 6. The chapter illustrates an example where the base station (central processing system) has to consider readings or data from several sensor networks deploying different security service specific communication protocols.

CHAPTER V

QoS SURFACE FOR WIRELESS SENSOR NETWORKS

5.1 Introduction

The notion of QoS is a guarantee by the network to satisfy a set of predetermined service performance constraints for the user in terms of the end-to-end delay statistics, available bandwidth, and probability of packet loss. When mobility grows high, the established routes are susceptible to link failures, diversions, or decrease of throughput. Thus, absolute throughput or delay bounds are hard to guarantee.

Most routing protocols proposed function on a best effort basis with no attempt to provide any Quality of Service (QoS). Routing can inform a source node of the bandwidth and QoS availability of a destination node. Therefore, some researchers have proposed the notion of soft QoS. Soft QoS means that after the connection has been set up, there may exist transient periods of time when QoS specification is not honored. The level of QoS satisfaction is represented as the fraction of total disruption time over the total connection time. This ratio should be higher than a threshold [2] which is required for the smooth functioning of the routing protocol employed in the network.

The essential task is to find a suitable path through the network, or route, between the source and destination(s) that will have the necessary available resources to meet the QoS constraints for the desired service.

The task of resource request, identification, and reservation is the other indispensable ingredient of QoS. QoS routing is a complex and difficult issue because of the dynamic nature of the network topology and generally imprecise network state information [23].

This work proposes QoS surface similar to that of an attack surface created previously. A QoS surface shows the expected QoS that can be delivered by the protocol. There is a tradeoff between the security and the quality of service provided on a sensor network. To facilitate the easy understanding of the proposed QoS surface, we have concentrated on ODRP which provides an optimum level of QoS.

The QoS surface for ODRP protocol is modeled similar to Wanderer Protocol in chapter IV. The dimensions considered for the QoS surface are inhibitors, enhancers, bottlenecks. The chapter provides QoS feature models similar to the threat models. Our approach enables us to identify the QoS features provided by the protocol with the help of state transition diagrams. The state transition diagram of the protocol is reduced to one of the state transition diagrams of the QoS features. If this reduction is successful, then we can conclude that the protocol provides the corresponding QoS feature. We show our approach to be successful in identifying and enhancing the QoS features present in the protocol.

5.2 Difficulty of providing high QoS

- Resource limitations – bandwidth, node energy
- Contention for bandwidth
- Absence of a centralized control/infrastructure

- Time varying links
- Presence of enormous amounts of data

Node count Associated with a base station:

An optimum number of sensor nodes will have to be associated with the base station. There can be a number of base stations present in the network. For data aggregation and analysis, a node is associated with a small subset of base stations only.

In this regard there is a necessity for increasing the lifetime of the individual nodes thereby increasing the life of the entire network as well. The lifetime of the network can be enhanced as follows:

- Providing a power down option with the individual nodes so that they can conserve their energy in the long run.
- A base station has to decide on the number of nodes that has to be active, so that there is sufficient transfer of data for carrying out a particular task.

5.3 QoS Routing Protocol for Ad Hoc Networks ODRP (On-Demand Delay Constrained Unicast Routing Protocol) [23]

The design of this protocol concentrates on the operations at the network layer. The sensor network routing protocol is powered with capabilities like the determining of the resource availability on the neighboring links and the availability of such resource reservation functions in each node.

5.3.1 Routing Information

- Each node maintains a distance vector consisting of |V| - 1 entries for every other node in the sensor network.
- The entry for a node ‘v’ at a node ‘u’ (u <> v) has the following information:
    1. Identifier for the node ‘v’
    2. The shortest distance from ‘u’ to ‘v’ (in hop count)
    3. The next hop of ‘u’ along the above mentioned shortest path to ‘v’
- The vector present at each node is obtained by making use of a suitable proactive wide area distance vector routing protocol in the network.
- The vector is used by ODRP in the route searching phases (to propagate in the promising direction) to prevent pure flooding.

5.3.2 Strategies

ODRP is designed to effectively reduce the communication overhead consumed in acquiring a low-cost delay-constrained path while achieving high route acquisition success probability. The following strategies are employed in the route searching operations:

- Hybrid routing
- Directional search
- Link-Delay based scheduling of packet forwarding

Hybrid Routing

- This strategy first probes the feasibility of the min-hop path connecting the source-destination pair of an incoming QOS request. If the path is feasible the path is returned.
- If the path is not feasible then a destination-initiated route searching process through restricted flooding is initiated.

Directional Search

- This strategy restricts the search range of the route searching process. Thus the rate at which the expensive flooding operations are enforced can be greatly reduced.

Link Delay Based Scheduling

- This strategy is for an intermediate node to decide on when it retransmits a Route Request (RREQ) packet it receives.
- In ODRP this retransmission is scheduled at a speed proportional to the delay of the link over which the packet was received.
- In the presence of multiple RREQ, the RREQ with the least delay value is chosen.
- If the scheduling functions at the various nodes are the same, a low delay path can be acquired as a result of such collaborative operations at intermediate nodes. (A blind flooding process does not lead to a delay-constrained path (though available) due to the potentially chaotic propagations of RREQ.

5.3.3 Operations

The operation of ODRP can be divided into two phases.

- Probe the feasibility of the min-hop path
- If not feasible, perform destination-initiated route discovery.

Phase I: Probing the feasibility of the Min-Hop Path

- The source 's' initiates a process to acquire a constrained path upon receiving a request for a route to a destination 'd' subject to the constraint Δ.
- Source's' sends a probe packet directly to'd' along the min-hop path connecting them and starts a timer.
- The probe packet gathers the accumulated delay information when propagating along the path.
- When destination'd' receives the packet, it checks for the delay constraint, if the constraint is satisfied then a constrained path has been identified. It sends an ACK back to the source along the reverse path.
- This ACK packet makes the necessary entries and reserves the required resources at each intermediate node.
- When the source's' receives the ACK, it can send data packets along the path. Thus, we get a least cost constrained path for the request.

Phase II: Destination-Initiated Route Discovery for a Delay-Constrained Path

- The second phase is initiated if the min-hop path is not feasible.
- Destination 'd' initiates a path discovery process by flooding a RREQ packet which contains the following information:
    1. Identifiers of source's' and destination'd'.

2. A session ID and a sequence number, as copied from the probe message d received.
3. Delay constraint Δ.
4. The accumulated delay on the path probed so far, which is initially zero.
5. Search Scope Limit D = MinHop(d, s) + H, where MinHop(d, s) is the min-hop distance from 'd' to 's' and H is a small positive integer.
6. A flag F when set indicates the destination initiated property of the routing process.

An intermediate node 'x' determines whether it accepts the incoming RREQ or not. The preconditions under which 'x' accepts the RREQ packet are as follows:

- MinHop(x, s) + MinHop(x, d) $\leq$ Δ (to check whether the node 'x' is located inside a predetermined searching range i.e. the directional search strategy).
- Delay (p) $\leq$ d (to ensure the non-violation of the given delay constraint).
- Node 'x' receives the RREQ for the first time or the packet has a lower delay than those received earlier (to ensure that an intermediate node forwards only the RREQ carrying the least delay value among those it receives).

Once the node 'x' accepts the RREQ, it schedules its re-broadcasting of the packet with a delay proportional to the delay of the incoming link over which the packet was received. As the packet is broadcasted, it updates the delay value on the sub path discovered so far. Node 'x' records its next hop as the neighboring node from which it received the RREQ carrying the least delay value and ignores the further duplicate incoming RREQ belonging to the request. So each node in the network forwards the RREQ at most once.

The first sent data packet will reserve resources at intermediate nodes along the path, if timed out without receiving a corresponding acknowledgement, the source ‘s’ sends data packets as best effort traffic along the min-hop path or performs QOS renegotiation.

The above said protocol has the following desirable features:

- Path discovery through restricted flooding is enforced only when the min-hop path does not meet the delay requirements, which reduces the communication overhead.
- Route searching process is directed and restricted by a predetermined searching range limit. This reduces the communication overhead if H is properly set. This routing process degenerates to a network wide flooding if H is set to infinity.

5.3.4 Path Maintenance

When there is a broken link the upstream destination node‘d’ sends a RERR to the downstream source‘s’. Then the source‘s’ send a release message to the intermediate nods to release the resources reserved earlier and remove the entries made previously. The broken state of the link can be determined by the absence of HELLO packets from the upstream node for a certain period of time. Upon receiving the RRER message the source‘s’ enforces a route discovery process if it still has packets to send to the destination node of the session.

Table 2 QoS Routing Protocols - Comparison

Protocols	**Strategy**	**Routing Information**	**Storage Overhead**	**Communication overhead**	**Route Acquisition Latency**	**Metrics**
Location-Aided Routing	Source Routing	Global State	O(|V|)	Zero	Low	Band-width/ delay
Min-hop Routing	Shortest Path Routing	Distance vector	O(|V|)	O(|V|)^0.5	Low	Band-width
Band-width Routing	Flooding	Local State	O(m)	O(|V|)	High	Band-width
Alternate Routing	Shortest Path + Alternate Routing	Distance Vector	O(|V|)	Moderate	Moderate	Band-width/ delay

m: Average number of one-hop neighbors of a node.
|V|: Number of nodes in the network.
|E|: Number of links in the network.

For a QoS route from source s to destination d subject to a path length constraint D. Here D = MinHop (d, s) + H. Here, MinHop (d, s) represents the min-hop distance from d to s and H is a small positive integer. The shaded ellipse area is the intended searching zone. Note that the distance in this figure is measured in hop count.

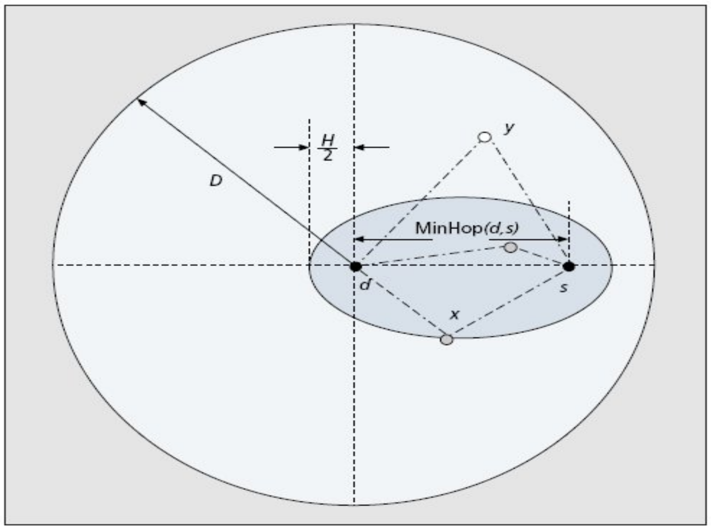

Figure 34: Destination-Initiated Directional Search Process in ODRP

5.4 Working of the protocol

States

A – The initial state of the sensor network. No transmission of packets is necessary. This can be identified as the state where no source and destination have been identified.

B – This is the state of the network when it enters the phase 1 of the ODRP. A min hop path is checked for its feasibility.

C – This is the state of the network when it enters the phase 2 of the ODRP. The min hop path is found to be not feasible in the previous state, so a source initiated route discovery process is started.

D – This is the final state of the network where the transmission of the packets along the least constrained path takes place.

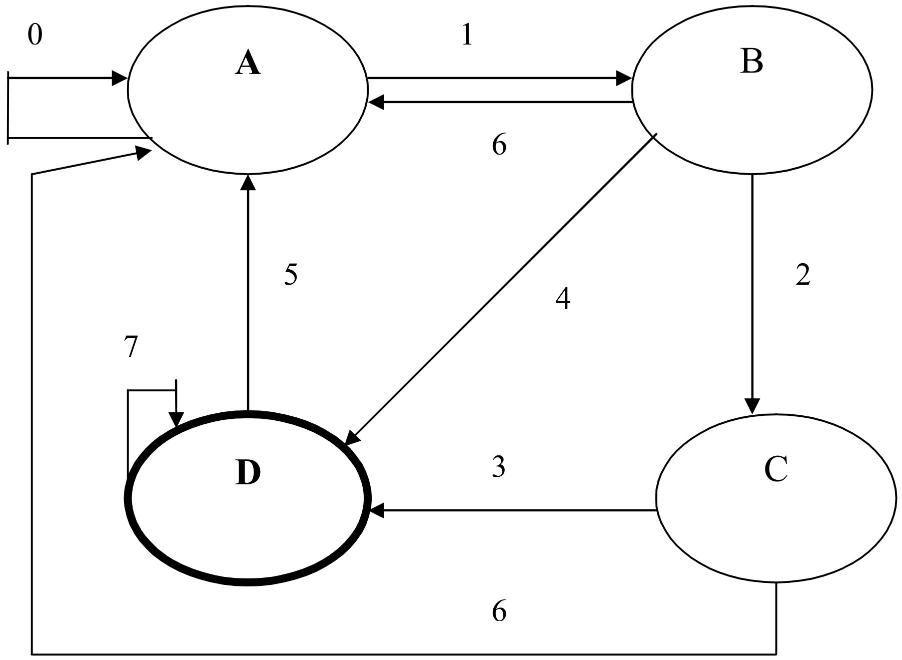

Figure 15: ODRP Model

Events

0 – The event at which the sensor network gets ready for transmission of packets. For example, the power is switched on for the entire network, or the control is transferred to this network to gather information

1 – The need to determine the min hop path or the least cost constrained path.

2 – Phase 1 fails to provide a min hop i.e. the min hop path is found to be not feasible

3 – Phase 2 succeeds in determining the least constrained cost path

4 – Phase 1 succeeds i.e. the min hop path is found to be feasible

5 – The transmission of packets from the source to the destination was successful

6 – Link failure/Network disruptions

7 – Transmission of packets along the path determined

5.4.1 Phase 1

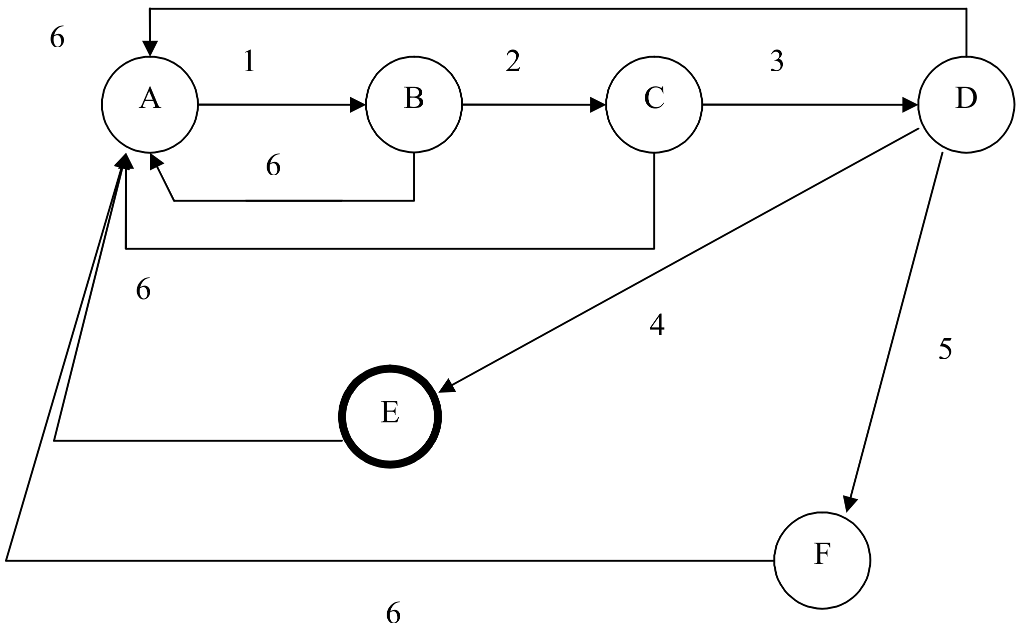

Figure 16: Phase 1 - ODRP Model

States

A – Initial State of the sensor network

B – The network enters phase 1 where the feasibility of the min hop path is determined

C – The source's' sends a probe packet to the destination'd'

D – The probe packet gathers the accumulated delay along the path

E – The accumulated delay is determined to satisfy the delay constraint imposed

F – The min hop path is not feasible so the network enters the phase 2.

Events

1 – The process is initiated by the source node to transmit the probe packet along the min hop path

2 – A timer is started to calculate the accumulated delay along the path

3 – The probe packet is transmitted along the min hop path

4 – The delay constraint checked with the accumulated delay along the path and is found to be a feasible solution

5 – The delay constraint is not satisfied.

6 – Any other external failure of the network – power, control returns the network to the original state.

5.4.2 Phase 2

States

A – The initial state where phase 1 has failed to provide the min hop path i.e. the min hop path was found to be not feasible. So the network enters the phase 2 of the protocol to provide a least cost constrained path.

B – The destination‘d’ initiates a path discovery process.

C – The intermediate node gets ready to accept or reject the incoming RREQ packet based on the conditions outlined in the protocol.

D – The accepted RREQ packet is rebroadcast to the neighboring nodes

E – The RREQ reaches the source‘s’ with the delay constraint being satisfied

F – The least constrained path for the transmission of packets has been identified. The least constrained path is the path that satisfies the delay constraints present in the network. This state refers to the best effort path being taken in situations where, a least constrained path could not be determined successfully.

G – Data packets are sent as best effort traffic when the operations are timed out due to absence of ACK messages.

H – The network enters the path maintenance state when there is a link error (broken links etc.). Phase 2 is responsible for both route discovery and path maintenance [23].

I – The resources reserved at each intermediate node are released by the release message and the network reaches the initial state at which phase 2 started.

Actions

1 – The destination‘d’ initiates a route discovery process.

2 – The incoming RREQ packet is checked to determine if the packet is acceptable i.e. if it is the newest packet, if it is then it is accepted else not accepted

3 – The RREQ packet is transmitted, the delay value is updated

4 – The source‘s’ receives the RREQ packet from the destination through various hops along the network.

5 – The first sent data packet reserves the resources along the path identified in the previous stage.

6 – No acknowledgement message or the operation has been timed out.

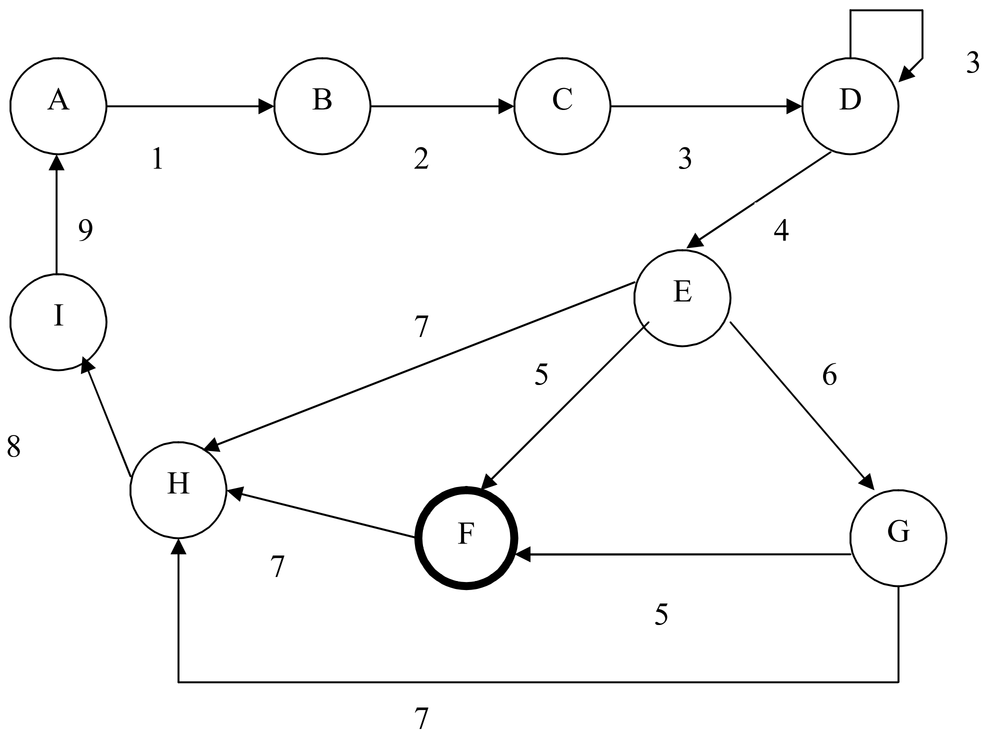

Figure 17: Phase 2 - ODRP Model

7 – In case of a broken link a RERR message is sent from the source to the destination of the session. This broken link can be identified by the absence of HELLO packets or data packets for a certain period of time.

8 – A release message is sent from the source to the destination which releases the resources along the intermediate nodes at which reservations and entries were made.

9 – After RERR has been sent if packets are still pending to be sent to the destination the source initiates a route discovery process which takes the network to the initial state of phase 2.

5.5 Dimensions of a QOS surface

Similar to identifying the dimensions of an attack surface, we have modeled the QoS surface of the sensor network using dimensions. The dimensions have been identified with respect to the factors affecting the QOS of the sensor network as described in QOS Challenges (Section 3.3.4). Also from the state transition diagrams we can identify the various dimensions.

Inhibitors

The inhibitors are the features by which QOS can be affected. In ODRP we can see that the distance between the nodes plays an important role in determining the feasibility of the min hop path. The buffer limitation at each node inhibits the QOS of the network. The physical distance between the nodes affects the QOS because the min hop path depends on it. The effect of inhibitors will have to be kept at a minimum to attain an optimum level of QOS.

These features will be present for the entire lifetime of the network. Only fine tuning the parameters (hardware, software) can help in increasing the QoS of the network.

Enhancers

The features that enhance the QOS of the network are represented by the dimension of enhancers.

When the buffer size present at each node allows for the aggregation of the messages sent from one node to another it reduces the power consumption of the whole network. This is because the transmission of the aggregate is a single transmission rather than individual transmission for each message.

Further, on close study of the ODRP protocol we can see that certain features can be identified as enhancers. One such example is the situation where an error occurs in the transmission of a packet from the source to a destination. As soon as an error occurs a RERR message is sent down along the path so that a new path identification scheme is initiated. This prevents the QOS from going below a threshold value for more than a certain period time (till a new path has been identified). This mode prevents the loss of packets while using a faulty or broken connection path.

Bottlenecks

The features that have a trade off between energy and time belong to the dimension of bottlenecks. These bottlenecks become visible while the system is operating under certain conditions. The buffer size may not be sufficient at each sensor node to aggregate the results of its neighbors if all the neighbor nodes have data to be sent (this occurs only when there is a drastic impact in the environment in which it is currently deployed).

These bottlenecks will affect the QOS though some improvements in certain areas can be achieved by tweaking the parameters. For example when we induce multi hop paths to be taken rather than a single hop path the time taken for the transmission of a packet from source to destination maybe longer but the energy maybe conserved due to the shorter transmissions involved.

5.6 QOS Features

We use the term ‘QoS’ liberally – it not only refers to traditional QOS parameters such as delay, jitter, bandwidth, etc. it also refers to other parameters such as the survivability of the network – which is important for sensor networks.

QOS features for a sensor network can be identified as

- Guaranteed delay
- Error Recovery
- Survivability
- Energy Conservation

We model both the system and the QoS as state machines:

$$System = (S_S, I_S, A_S, T_S)$$

$$QOS\ features = (S_Q, I_Q, A_Q, T_Q)$$

QOS features are a set of QOS parameters. Each feature can be expressed as a transition diagram.

a) Error Recovery

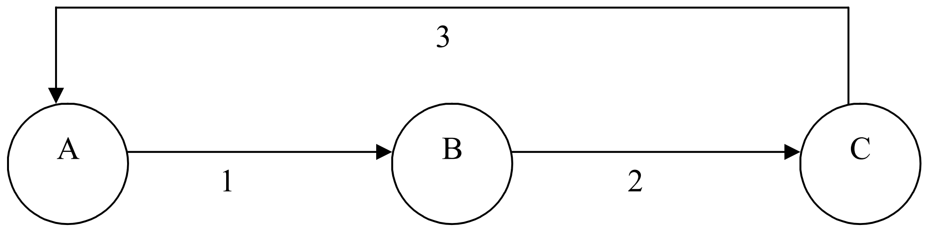

Figure 18: Error Recovery Model

States

A – Normal functioning of the sensor network

B – Error in the transmission of packets. Errors like faulty transmission of packets, corrupt data packets, failure to send the requested data packets within the timeout period, failure in the transmission links thus losing data packets.

C – Recover from error to continue the normal working. Recovery routines comprise of request for retransmission of the corrupt data packets, requesting new data from the remote sensor nodes through new links identified. The receiving node can determine the validity of data packets by using verification procedures similar to the validation of checksums. If this validation process results are unsuccessful then a request for new data packets is sent back to the sender node.

Actions

1 – An error has occurred in the transmission of packets or normal working of the protocol

2 – Recovery routines are decided upon to rectify the errors. As explained in state C, one of the recovery routines is chosen to correct the errors in the data transmission.

3 – The error recovery routines are executed so that the sensor network enters its normal execution mode again.

b) Survivability

States

A – Normal functioning of the sensor network

B – Attack has been mounted on the sensor network. Attacks like HELLO/Sybil, wormhole, sinkhole can be mounted by an adversary. A brief explanation of these attacks has been provided in the previous chapter.

C – Recover from attack. When an attack has been mounted, depending on the time out interval and the unavailability of the requested packets at the base station, the base station requests the same packets again. So recovery from such attacks comprises of determining new transmission paths, avoiding the sensor nodes which have been compromised while determining the new paths and stopping the transfer of sensitive information to those sensor nodes.

D – The network recovers from the attack and normal operation continues

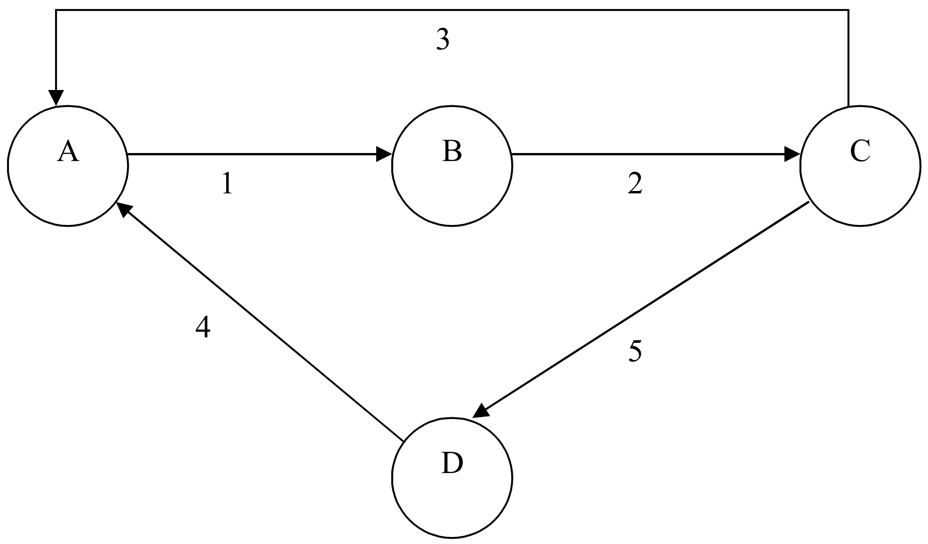

Figure 19: Survivability Model

Actions

1 – An attack made by an adversary on the normal working of the protocol

2 – Recovery routines are decided upon to defend against attacks. From the recovery routines described in state C, the base station initiates the feasible recovery routine so that the extent of damage caused by the attack can be minimized.

3 – The defensive mechanisms are executed so that the sensor network enters its normal working mode again, the attacks are defended against but the transmission will have to start from the source again to avoid any discrepancies in the data packets.

4 – With the attack effects minimized, the network returns to its normal working.

5 – An attack was successful but the effects are reduced by the defensive and recovery mechanisms present in the network. Some defensive routines are creating an inactive area consisting of the compromised nodes and thus preventing any future transmissions reaching those sensor nodes present in the inactive area.

c) Guaranteed Delay (Energy Conservation)

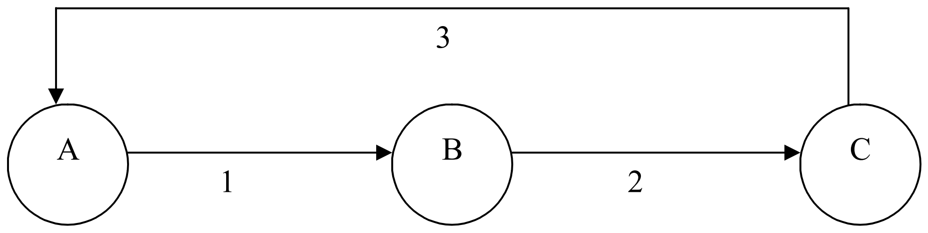

Figure 20: Guaranteed Delay Model

States

A – Normal functioning of the sensor network

B – Intermediate hop transmission of packets to save energy. A single node usually will not be provided with the required amount of power to perform the 1-hop operation of transmitting packets directly to the base station. When multiple hops are the only option in the network, we will be able to conserve energy. Energy is conserved using this method because the power present in a single node is sufficient to transfer data to its neighboring nodes rather than to a distant destination node.

C – Packet reaches the base station

Actions

1 – A packet has to be transmitted from a node to the base station.

2 – The packet is transmitted is across the intermediate nodes till it reaches the base station

3 – The network enters its original state after the packet reaches its destination (base station)

In a similar manner, other QOS features can be expressed as transition diagrams.

Inference:

If there is some Sub-graph in the system state transition diagram of a protocol that matches the state transition diagram for a QoS feature, then we know that this QoS feature is available in the protocol. In other words, if the states present in a protocol can be grouped to effectively produce a state transition diagram of a QoS feature, then the QoS feature is said to exist in the protocol.

There is a difference between the attack surface and the QOS surface. In the attack surface we try to see if there is *any state in the system state transition diagram that matches the initial state of the threat* state transition diagram. In the QOS surface we see if there is *any sub-graph in the system state transition diagram that matches the QOS features* state transition diagram

For example, when we consider the general transition diagram of ODRP figure 15, the states A, B, C, D represent the execution of the protocol. Similarly consider the transition diagram of the Error Recovery Model figure-18; we can see that the states A, B, C of this transition model corresponds to the A, B, and C of figure-15. The transition states of the two diagrams are also similar in their actions. Thus the state transition model of ODRP can be reduced to that of the error recovery model. This indicates that ODRP provides the user with the QoS feature of error recovery. Likewise, we can also see that ODRP provides the QoS features of survivability and Guaranteed delay (to conserve the battery power at each sensor node). Considering the two figures 15, 19, we can see that the states A, B, C and D of figure 15 correspond to the A, B, C and D of figure 19. The transition states are also similar in their actions. This indicates that ODRP provides the user with the QoS feature of survivability. Similarly, considering the figures 15, 20 the states A, B, C of figure 15 correspond to the states A, B, C of figure 20. Also the transition states are similar in their actions. This indicates that ODRP provides the user with the QoS feature of Guaranteed Delay.

5.7 Example for the QoS surface

We model the system, the QoS models as state machines:

$$System = (S_S, I_S, A_S, T_S)$$

$$QoS\_features = (S_Q, I_Q, A_Q, T_Q)$$

Quality of Service Surface

$$System_{ODRP} = (S_{ODRP}, I_{ODRP}, A_{ODRP}, T_{ODRP})$$

$$QoS\_features = (S_Q, I_Q, A_Q, T_Q)$$

ODRP (On-Demand Delay Constrained Unicast Routing Protocol)

The subscript (i) and (ii) are used to indicate the two phases of the protocol. The illustrations can be seen in previous sections Figs 16, 17

- S_{ODRP} = {set of states}

  = {A(i),B(i),C(i),D(i),E(i),F(i),

  A(ii),B(ii),C(ii),D(ii),E(ii),F(ii),G(ii),H(ii),I(ii)}

- I_{ODRP} = {set of initial states}

  = {A(i)}

- A_{ODRP} = {set of actions}

  = {1(i),2(i),3(i),4(i),5(i),6(i),7(i),

  1(ii),2(ii),3(ii),4(ii),5(ii),6(ii),7(ii),8(ii),9(ii)}

- T_{ODRP} = {set of transition relations}

  = {(A(i),1(i),B(i)),(B(i),2(i),C(i)),(C(i),3(i),D(i)),(D(i),4(i),E(i)),

  (D(i),5(i),F(i)),(D(i),6(i),A(i)),(E(i),6(i),A(i)),(C(i),6(i),A(i)),

  (F(i),6(i),A(i)),(B(i),6(i),A(i)),

  (A(ii),1(ii),B(ii)),(B(ii),2(ii),C(ii)),(C(ii),3(ii),D(ii)),

  (D(ii),3(ii),D(ii)),(D(ii),4(ii),E(ii)),(E(ii),5(ii),F(ii)),

  (E(ii),6(ii),G(ii)),(E(ii),7(ii),H(ii)),(F(ii),7(ii),H(ii)),

  (G(ii),5(ii),F(ii)),(G(ii),7(ii),H(ii)),(H(ii),8(ii),I(ii)),(I(ii),9(ii),A(ii))}

$QoS\_features_i = (S_Q, I_Q, A_Q, T_Q)$

Let us see the individual sets for each QOS model. The models have been illustrated in previous sections.

1. **Error Recovery (Figure 18)**
    - S_{Qi} = {set of states}

      = {A, B, C}
    - I_{Qi} = {set of initial states}

      = {A}
    - A_{Qi} = {set of actions}

      = {1, 2, 3}
    - T_{Qi} = {set of transition relations}

      = {(A, 1, B), (B, 2, C), (C, 3, A)}
2. **Survivability(Figure 19)**
    - S_{Qi} = {set of states}

      = {A, B, C, D}
    - I_{Qi} = {set of initial states}

      = {A}
    - A_{Qi} = {set of actions}

      = {1, 2, 3, 4, 5}
    - T_{Qi} = {set of transition relations}

      = {(A, 1, B), (B, 2, C), (C, 3, A), (C, 5, D), (D, 4, A)}

3. **Guaranteed Delay (Energy Conservation) (Figure 20)**

- S_{Qi} = {set of states}

  = {A, B, C}

- I_{Qi} = {set of initial states}

  = {A}

- A_{Qi} = {set of actions}

  = {1, 2, 3}

- T_{Qi} = {set of transition relations}

  = {(A, 1, B), (B, 2, C), (C, 3, A)}

Definition 3:

A protocol P satisfies QoS requirement i if there exists a set of Pre-conditions which results in a set of states represented by a sub-graph S_s of $System_{ODRP}$ such that S_s = $QoS\_features_i$ and a set of post conditions are satisfied to meet the QoS requirements

For example, a node may not receive the required number of packets within a specified time due to another node upstream being out of action – this is the precondition. The protocol than enters a state S_s = *ERROR_RECOVERY.* The post-condition leads to RERR messages being transmitted to find an alternative path.

A sub graph is one which is present as a part of the transition diagram.

From the figures given below the concept of sub graph becomes clear.

A1 = S1, A2 = S3, A3 = S4

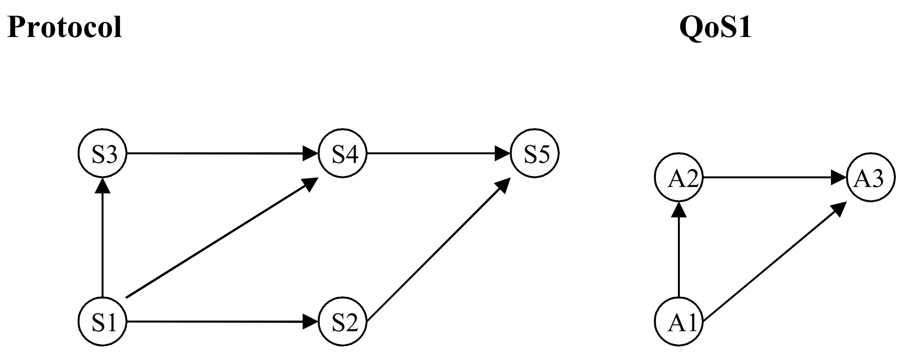

Figure 21: Protocol Schematic Diagram

The QoS graph is a subset of the protocol graph; so we can conclude that the protocol will meet the requirements for QoS type 1. Similarly, when we consider the state transition diagram of ODRP figure 15 and survivability model figure 19 we get the following graphs.

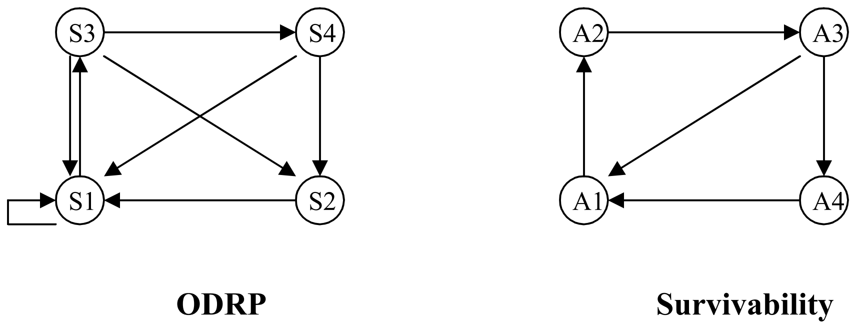

Figure 22: Protocol Schematic Diagram ODRP Vs Survivability model

Comparing the two figures 15, 22, we can see that A = S1, B = S3, C = S4, D = S2. Likewise, comparing the two figures 15, 20, we can see that A = A1, B = A2, C = A3, D = A4. This representation has been provided to indicate the feasibility of determining the QoS features provided by ODRP. From the two graphs shown we can see that S1=A1, S3=A2, S4=A3, S2=A4. When we reduce the graphs of ODRP in not considering some of the transitions, then both the models are similar. This indicates the presence of the QoS feature of survivability in ODRP.

Definition 4:

A protocol P satisfies QoS requirements 1,2,…n if there exist different preconditions that lead to sub-graphs S_{s1}, S_{s2},…, S_{sn} of $System_{ODRP}$ such that S_{si} = $QoS\_features_i$ for all i = 1,2,…n and the post-conditions are satisfied .

It is important to note that the subgraphs S_{s1}, S_{s2},…, S_{sn} may not necessarily be disjoint as some states (nodes) may match between the different graphs. However, and the Pre-conditions and post-conditions will be disjoint.

If the states present in a protocol can be grouped (reduced) to produce a state transition diagram equivalent to that of a QOS feature, then the QOS feature is said to exist in the protocol. From the transition diagrams of the ODRP and the QOS models we can see that error recovery, survivability, guaranteed delay are provided by this ODRP protocol. (When certain states present in the transition diagram are combined as a single unit and the transition diagram is reduced accordingly we get transition diagrams similar to the QOS feature models.)

From the basic transition working diagram provided in the previous section (Figure 11) we can see that the four states A, B, C, D have the three illustrated QOS features as sub-graphs.

5.8 Conclusion

The approach of creating a QoS or service surface has been illustrated in this chapter. In a similar manner, any protocol which has been deployed to provide a specific service can be modeled as a surface across the dimensions. Making use of the dimensions of the QoS surface created, we will be able to increase the QoS of the sensor network.

This method of fine tuning the dimensions is explained in chapter 6. The chapter illustrates an example where the base station (central processing system) has to consider readings or data from several sensor networks deploying different service specific (security, QoS) communication protocols.

CHAPTER VI

QoAS and QoS MEASUREMENT

In this chapter we propose a mapping mechanism which allows us to measure the quality of security (or service) of a particular protocol in relation to that provided by the entire network. In our proposed approach we take the protocol that provides the highest level of security (or service) and compare it with the security or service provided by another part of the same system that is deploying another protocol. If there is a large difference, this indicates that security across the network is not evenly spread. To compensate for this situation, the related hardware and software can be enhanced by modifying the appropriate dimensions. This will result in enhanced security (or service) for the entire system.

We illustrate our approach by considering a situation where a central processing system or base station considers the readings from two sensor networks deploying two different routing protocols. The two different protocols under consideration are wanderer + LEAP protocol and the ODRP protocol. In the following sections mappings are provided between the dimensions of an attack surface and the dimensions of the QOS surface. This illustrates the interaction between the two unique protocols on the readings obtained by the central processing system. The architecture takes into consideration the features provided in [21] [22]. This method of determining the effect of interaction between two protocols can be extended to a number of unique protocols employed in the particular environment. The central processing system takes into consideration the readings from all the networks, deploying the unique protocols.

Under such cases, we will be able to increase the security of the network, which enables the central processing system to receive less tampered data. The security of service of the system is increased by modifying the dimensions of the system. For example, the dimension of channels can be increased by one to provide more channels, thereby increasing the overall system security.

A sensor network that is deployed across a wide area may have several protocols in the network as shown below.

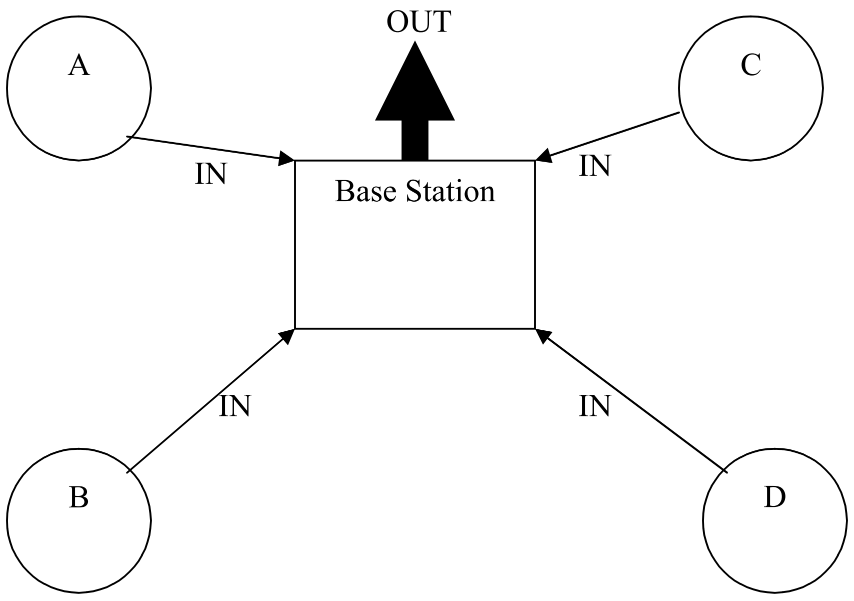

Figure 23: Theoretical Sensor Network Architecture

A – WANDERER + LEAP PROTOCOL	**IN – DATA INPUT**
B – ODRP PROTOCOL	**OUT – DATA OUTPUT**
C, D – OTHER SERVICE SPECIFIC PROTOCOLS	

Different protocols provide different levels of security and quality of service. For example, one protocol may provide a minimum quality of security, whereas another protocol in the same network may provide a very high quality of security.

6.1 Mappings

There have been very few instances of in the literature of research being done in the area of bridging the application-network gap [24]. This reflects the inherent difficulty of making an attempt to link the quality of a protocol in terms of security or service with network parameters. There are three approaches which can be identified as follows:

1. Implicit Mapping
2. Explicit Mapping
3. Restricted Implicit Mapping (probes)

None of these approaches considers security. In this chapter we propose a novel approach to measure the quality of service or security of an individual protocol in relation to the overall quality of service or security of the entire system.

We first define the dimensions of the protocols.

6.2 Attack Surface of a protocol

- Targets and Enablers (TE)
- Channels and Protocols (CP)
- Rules of access (RA)

QOS Surface of a protocol

- Inhibitors (I)

- Enhancers (E)
- Bottlenecks (B)

Let

TQN = Total Quality of the network

QoAS = Quality of Attack Surface

QoQS = Quality of the QOS Surface

So we have

$$\text{TQN} = g\,(\text{QoAS}, \text{QoQS}) \qquad \text{------------------------ (1)}$$

'g' is a function that combines the features of the attack surface with the QOS surface.

Based on our study of the attack surface for a protocol we can conclude that

$$\text{QoAS} = \frac{(\text{TEp}) + \frac{b}{\text{CPp}} + \frac{c}{\text{RAp}}}{(d * \text{TEt}) + (e * \text{CPt}) + (f * \text{RAt})} \qquad \text{------------------------- (2)}$$

As shown in Equation 2, we can make use five constants b, c, d, e, and f to normalize the values of CPp, RAp, TEt, CPt, RAt around the value of TEp.

Where

TEp is the presence factor of Targets and Enablers

TEt is the total presence factor of Targets and Enablers

CPp is the presence factor of Channels and Protocols

CPt is the total presence factor of Channels and Protocols

RAp is the presence factor of Rules of Access

RAt is the total presence factor of Rules of Access

Presence factor of a dimension for an attack surface is explained in Definition 5 below and total presence factor of a dimension for an attack surface is explained in

Definition 6 below. The coefficients b, c, d, e and f can be effectively determined experimentally when studying the behavior and effects of the various dimensions involved in an attack surface, on the security features of the sensor network.

Equation (2) provides us the relation between the security of a sensor network and QoAS (Quality of Attack Surface).

Security $\propto$ 1/QoAS

- As the number of targets and enablers increase, the security of the network decreases thus increasing the quality of the attack surface.
- As the number of channels and protocols increase, the security of the network increases thus decreasing the quality of the attack surface.
- As the number of rules of access increase, the security of the network is enhanced thus decreasing the quality of the attack surface.

Equation (2) provides us with a comparison of the presence factor of the dimensions identified in the attack surface with the total presence factor of the dimensions in the entire system. As the QoAS decreases, the security of the sensor network system is increased.

Definition 5

A presence factor AS^{k}_{i} of a protocol *i* specifies the value of a dimension *k* of an attack surface

Presence factor of a dimension such as targets and enablers specifies the number of different targets and enablers present in the protocol. It is therefore the value of a particular dimension.

Definition 6

The total presence factor AS^k of protocols $P_1,\ldots P_n$ is $\sum_{i=1}^{n} AS^k{}_i$ for a dimension k of an attack surface. The total presence factor of a dimension such as targets and enablers over a number of protocols specifies the total number of different targets and enablers present in the different protocols. It is therefore the value of a particular dimension over all the protocols.

Likewise, we take into consideration the QOS surface for a protocol.

Based on our study of the QOS surface for a protocol we can conclude that

$$\text{QoQS} = \frac{\frac{1}{\text{Ip}} + (\text{b} * \text{Ep}) + \frac{\text{c}}{\text{Bp}}}{(\text{d} * \text{It}) + (\text{e} * \text{Et}) + (\text{f} * \text{Bt})} \qquad \text{------------------------ (4)}$$

As shown in Equation 2, we can make use five constants b, c, d, e, and f to normalize the values of Ep, Bp, It, Et, Bt around the value of Ip.

Where

Ip is the presence factor of inhibitors

It is the total presence factor of inhibitors

Ep is the presence factor of Enhancers

Et is the total presence factor of Enhancers.

Bp is the presence factor of Bottlenecks

Bt is the total presence factor of Bottlenecks

Presence factor of a dimension for a QoS surface is explained in Definition 7 below and the total presence factor of a dimension for a QoS surface is explained in Definition 8 below. The coefficients b, c, d, e and f can be determined experimentally when studying the behavior and effects of the various dimensions involved in a QoS surface, on the QoS features of the sensor network.

Equation (4) provides us the relation between the QoS of a sensor network and QoQS (Quality of Service Surface).

Quality of Service $\propto$ QoQS

- As the number of inhibitors decrease, the QoS of the network increases thus increasing the quality of the QoS surface.
- As the number of enhancers increase, the QoS of the network increases thus increasing the quality of the QoS surface.
- As the number of bottlenecks decrease, the QoS of the network is increased thus increasing the quality of the attack surface.

Equation 4 provides us with a comparison of the presence factor of the dimensions identified in the QoS surface with the total presence factor of the dimensions in the entire system. As the QoQS increases, the QoS of the sensor network system increases.

Definition 7

A presence factor $QS^{k}{}_{\underline{i}}$ of a protocol *i* specifies the value of a dimension *k* of a QoS surface.

Presence factor of a dimension such as inhibitors specifies the number of different inhibitors present in the protocol. It is therefore the value of a particular dimension.

Definition 8

The total presence factor QS^k of protocols $P_1, \ldots P_n$ is $\sum_{i=1}^{n} AS^k{}_i$ for a dimension k of a QoS surface. The total presence factor of a dimension such as inhibitors over a number of protocols specifies the total number of different inhibitors present in the different protocols. It is therefore the value of a particular dimension over all the protocols.

6.3 Example

Consider a simple architecture for two sensor networks A1, A2 deploying the ODRP and the wanderer + LEAP protocol respectively (Fig 6.1). Let there be 10 nodes present in each network which are capable of operating at 2 different frequencies. One frequency will be used for the communication between the sensor nodes and the second frequency for communication between the base station and the sensor node.

We know that,

TQN = g (QoAS, QoQS)

For the Attack surface of the network, from equation 2, where b=c=d=e=f=1

$$\text{QoAS} = \frac{\text{TEp} + \frac{1}{\text{CPp}} + \frac{1}{\text{RAp}}}{\text{TEt} + \text{CPt} + \text{RAt}} \qquad \text{(a)}$$

Let us consider the following values for the attack surface of Wanderer + LEAP (A2):

- TEp = 10 (the number of sensor nodes present)
- CPp = 2 (the number of channels - frequency for communication present)
- RAp = 4 (the number of keys used for encryption and decryption. LEAP makes use of 4 different keys for communication)

Let us consider the following values for the attack surface of ODRP (A1):

- TEp = 10 (the number of sensor nodes present)
- CPp = 2 (the number of channels present)
- RAp = 1 (the factor to represent the absence of key based security mechanisms)

Using equation (a) for A1 and A2

$$\text{QoAS (A1)} = \frac{10 + \frac{1}{2} + \frac{1}{1}}{10 + 2 + 1} = 11.5/13 = 0.88$$

$$\text{QoAS (A2)} = \frac{10 + \frac{1}{2} + \frac{1}{4}}{10 + 2 + 4} = 10.75/16 = 0.67$$

- Let us reduce the number of sensor nodes in network A1 and A2 by 5 - when the number of sensor nodes is reduced, less number of sensor nodes is available as targets for adversaries.

$$\text{QoAS (A1)} = \frac{5 + \frac{1}{2} + \frac{1}{1}}{5 + 2 + 1} = 6.5/8 = 0.81$$

$$\text{QoAS (A2)} = \frac{5 + \frac{1}{2} + \frac{1}{4}}{5 + 2 + 4} = 5.75/11 = 0.52$$

- Let us increase the number of channels in ODRP network A1 and A2 by 1 - more channels increase the security because if one channel is compromised the other one can be used.

$$\text{QoAS (A1)} = \frac{10 + \frac{1}{3} + \frac{1}{1}}{10 + 3 + 1} = 11.33/14 = 0.80$$

$$\text{QoAS (A2)} = \frac{10+\frac{1}{3}+\frac{1}{4}}{10+3+4} = 10.58/17 = 0.62$$

- Let us increase the number of rules of access in ODRP network A1 and A2 by 2 – if the rules of access (constraints to be satisfied) are varied then an adversary will not be able to easily compromise a sensor node.

$$\text{QoAS (A1)} = \frac{10+\frac{1}{2}+\frac{1}{3}}{10+2+3} = 10.83/15 = 0.72$$

$$\text{QoAS (A2)} = \frac{10+\frac{1}{2}+\frac{1}{6}}{10+2+6} = 10.83/15 = 0.59$$

If the number of sensors is reduced, the channels increased and the access rule increased, the overall quality of security improves

$$\text{QoAS (A1)} = \frac{5+\frac{1}{4}+\frac{1}{3}}{5+4+3} = 5.58/12 = 0.46$$

$$\text{QoAS (A2)} = \frac{5+\frac{1}{4}+\frac{1}{6}}{5+4+6} = 5.42/15 = 0.36$$

We know that as QoAS decreases the security of the network is increased. We can see from the above examples that the security of the network can be increased by changing the dimensions of the associated attack surface of the network.

For the QoS surface of the network, from equation 4, where b=c=d=e=f=1

$$\text{QoQS} = \frac{\frac{1}{\text{Ip}} + \text{Ep} + \frac{1}{\text{Bp}}}{\text{It} + \text{Et} + \text{Bt}} \qquad \text{(b)}$$

Let us consider the following values for the QoS surface of Wanderer + LEAP (A2):

- Ip = 2 (factor to indicate the encryption/decryption involved)
- Ep = 0 (factor to indicate single path)
- Bp = 1 (the number of neighbor nodes to be considered while determining the receiver for the next transmission. The buffer size present in each node corresponds to the number of nodes that can be identified as neighbors)

From the above configuration, we can determine the QoQS for A2 as follows:

$$\text{QoQS (A2)} = \frac{\frac{1}{2} + 0 + \frac{1}{1}}{2 + 0 + 1} = 1.5/3 = 0.5$$

Let us consider the following values for the QoS surface of ODRP (A1):

- Ip = 1 (factor to indicate the absence of encryption/decryption)
- Ep = 1 (factor to indicate the single path with a constraint on the distance traveled. The minimum value constraint for the hop count will have to be satisfied before the path is chosen as the transmission path)
- Bp = 1 (A buffer is used to aggregate the data packets from neighboring nodes before transmitting it to the next node in the path, the buffer size is a potential bottleneck compromising between space and conservation of battery power)

From the above configuration, we can determine the QoQS for A1 as follows:

$$\text{QoQS (A1)} = \frac{\frac{1}{1}+1+\frac{1}{1}}{1+1+1} = 3/3 = 1.0$$

We can see that the QoQS for ODRP is greater than that of Wanderer + LEAP (A2). This indicates a higher level of QoS from the sensor network employing ODRP.

Table 3 Encryption - Decryption Factors

Encryption/Decryption	Factor
Absent	1
Partially Present	1.5
Present	2

Using equation (b)

- Let us remove the encryption/decryption of messages across nodes in network A1 and A2 to a factor 1.5 (Partially present is the condition where encryption/decryption is used only for the initial setting up of the neighbor list or the synchronization of with the base station).

$$\text{QoQS (A1)} = \frac{\frac{1}{1.5}+1+\frac{1}{1}}{1.5+1+1} = 2.67/3.5 = 0.77$$

$$\text{QoQS (A2)} = \frac{\frac{1}{1.5}+0+\frac{1}{1}}{1.5+0+1} = 1.67/2.5 = 0.67$$

- Let us use the minimum path constraint on the A1 and A2 protocol also.

$$\text{QoQS (A1)} = \frac{\frac{1}{1} + 2 + \frac{1}{1}}{1 + 2 + 1} = 4/4 = 1.0$$

$$\text{QoQS (A2)} = \frac{\frac{1}{2} + 1 + \frac{1}{1}}{2 + 1 + 1} = 2.5/4 = 0.625$$

- Let us increase the size of the buffer in A1 and A2 protocol by 1, so that the bottleneck dimension is increased while determining the neighbor nodes.

$$\text{QoQS (A1)} = \frac{\frac{1}{1} + 2 + \frac{1}{2}}{1 + 2 + 2} = 3.5/5 = 0.70$$

$$\text{QoQS (A2)} = \frac{\frac{1}{2} + 1 + \frac{1}{2}}{2 + 1 + 2} = 2/5 = 0.40$$

We know that as QoQS increases the QoS of the network is increased. We can see from the above examples that the QoS of the network can be increased by changing the dimensions of the associated QoS surface of the network.

6.4 Theoretical Framework

In the previous section, we presented an approach to measure the relative importance of a protocol. We present a more detailed theoretical framework to our approach in this section.

For the Attack surface of the network,

$$\text{QoAS} = \frac{(\text{TEp}) + \frac{b}{\text{CPp}} + \frac{c}{\text{RAp}}}{(d * \text{TEt}) + (e * \text{CPt}) + (f * \text{RAt})} \qquad \text{-- (2)}$$

Let TE, CP, RA denote the relative importance of the dimensions namely, targets and enablers, channels and protocols, and rules of access. In the context of our mapping we can consider that this total presence factor TEt, CPt, RAt reflects this importance.

So we have,

$TEt \approx q_1 TE, CPt \approx q_2 CP, RAt \approx q_3 RA$ -- (c)

Here q1, q2, q3 are strictly positive such that

$q_1 \approx q_2 \approx q_3$

Three values of q are used to indicate that in practice there might be slight deviations when compared to a perfect mapping.

$\frac{TEp}{TEt} = QoASte$ (QoAS dependence on targets and enablers)

So, $TEp = TEt * QoASte$

QoASte is the dependence of QoAS on the dimension of targets and enablers. It is a fraction of the total value of QoAS for any sensor network system.

$\frac{1/CPp}{CPt} = QoAScp$ (QoAS dependence on channels and protocols)

So, $CPp = \frac{1}{CPt * QoAScp}$

QoAScp is the dependence of QoAS on the dimension of channels and protocols. It is a fraction of the total value of QoAS for any sensor network system.

$\frac{1/RAp}{RAt} = QoASra$ (QoAS dependence on rules of access)

So, $RAp = \frac{1}{RAt * QoASra}$

QoASra is the dependence of QoAS on the dimension of rules of access. It is a fraction of the total value of QoAS for any sensor network system.

Using the above values

We get,

$$QoAS = \frac{TEt * QoASte + \frac{1}{CPt * QoAScp} + \frac{1}{RAt * QoASra}}{TEt + CPt + RAt}$$

Finally using (c)

We have,

$$QoAS \approx \frac{TE * QoASte + \frac{1}{CP * QoAScp} + \frac{1}{RA * QoASra}}{TE + CP + RA}$$

$$\approx \frac{1}{TE + CP + RA}\begin{pmatrix} TE & \frac{1}{CP} & \frac{1}{RA} \end{pmatrix}\begin{bmatrix} QoASte \\ QoAScp \\ QoASra \end{bmatrix}$$

Similarly, for the QOS surface of the sensor network

$$\text{QoQS} = \frac{\frac{1}{\text{Ip}} + (\text{b} * \text{Ep}) + \frac{\text{c}}{\text{Bp}}}{(\text{d} * \text{It}) + (\text{e} * \text{Et}) + (\text{f} * \text{Bt})} \qquad \text{-- (4)}$$

Let I, E, B denote the relative importance of the inhibitors, enhancers, bottlenecks respectively. In the context of our mapping we can consider that the total presence factor It, Et, Bt reflects this importance.

So we have,

$It \approx q_1 I,\ Et \approx q_2 E,\ Bt \approx q_3 B$ -- (d)

Here q1, q2, q3 are strictly positive such that
$q_1 \approx q_2 \approx q_3$

Three values of q are used to indicate that in practice there might be slight deviations when compared to a perfect mapping.

$\frac{1/Ip}{It} = QoQSi$ (QoQS dependence on inhibitors)

So, $Ip = \frac{1}{It * QoQSi}$

QoQSi is the dependence of QoQS on the dimension of inhibitors. It is a fraction of the total value of QoQS for any sensor network system.

$\frac{Ep}{Et} = QoQSe$ (QoQS dependence on enhancers)

So, $Ep = Et * QoQSe$

QoQSe is the dependence of QoQS on the dimension of enhancers. It is a fraction of the total value of QoQS for any sensor network system.

$\frac{1/Bp}{Bt} = QoQSb$ (QoQS dependence on bottlenecks)

So, $Bp = \frac{1}{Bt * QoQSb}$

QoQSb is the dependence of QoQS on the dimension of bottlenecks. It is a fraction of the total value of QoQS for any sensor network system.

Using the above values
We get,

$$QoQS = \frac{\frac{1}{It * QoQSi} + Et * QoQSe + \frac{1}{Bt * QoQSb}}{It + Et + Bt}$$

Finally using (d)

We have,

$$QoQS \approx \frac{\frac{1}{I * QoQSi} + E * QoQSe + \frac{1}{B * QoQSb}}{I + E + B}$$

$$\approx \frac{1}{TE + CP + RA}\begin{pmatrix} \frac{1}{I} & \frac{E}{1} & \frac{1}{B} \end{pmatrix}\begin{bmatrix} QoASte \\ QoAScp \\ QoASra \end{bmatrix}$$

6.5 Theoretical Protocol (An effective Attack surface and an effective QOS surface)

We have identified three dimensions for the attack surface of the 'wanderer protocol' and the quality of service surface of the 'routing protocol with ODRP. As a further study, let us consider the case where we develop a new protocol which combines the strengths of both the above protocols mentioned to achieve an acceptable degree of security and QOS.

The dimensions identified for the attack surface of the wanderer protocol are as follows:

- Targets and Enablers
- Channels and Protocols
- Rules of access (Protection layer)

Likewise, the dimensions identified for the QOS surface of the routing protocol with ODRP are:

- Inhibitors
- Enhancers
- Bottlenecks

Attack surface:

Since we have identified three dimensions for the wanderer protocol we consider the case of a simple cube to express them as shown.

Let us consider the following units of size to express the length of each ‘dimension’:

- ‘a’ for Targets and Enablers
- ‘b’ for Channels and protocols
- ‘c’ for Rules of access

Since we are considering the case of a cube let us assume that all three dimensions play an equal part in determining the attack surface i.e. all dimensions are of the same length ‘z’ units.

So, now we have

$$a \Leftrightarrow b \Leftrightarrow c \Leftrightarrow z$$

With the above equivalence we can calculate the surface area of the cube to be **‘6z^2 square units’**

Thus we have the surface area of a cube which indicates the attack surface area which can be obtained when using the properties of the wanderer protocol. The smaller the attack surface area obtained for a protocol the greater the security associated with its usage.

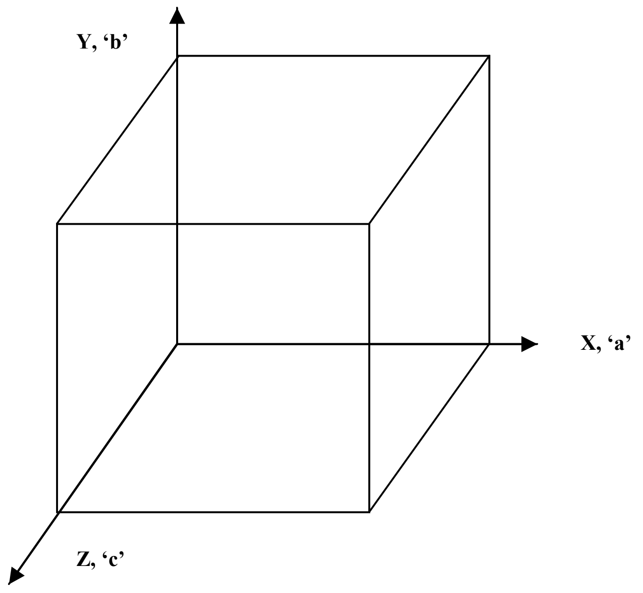

Figure 24: Attack Surface with the Dimensions

Let us do the same for the QOS surface obtained by using the routing protocol with mobile backbones.

QOS surface:

Since we have identified three dimensions for the QOS routing protocol let us consider the case of a simple cube again to express them as shown.

Let us consider the following units of size to express the length of each ‘dimension’:

- ‘d’ for Inhibitors
- ‘e’ for Enhancers
- ‘f’ for Bottlenecks

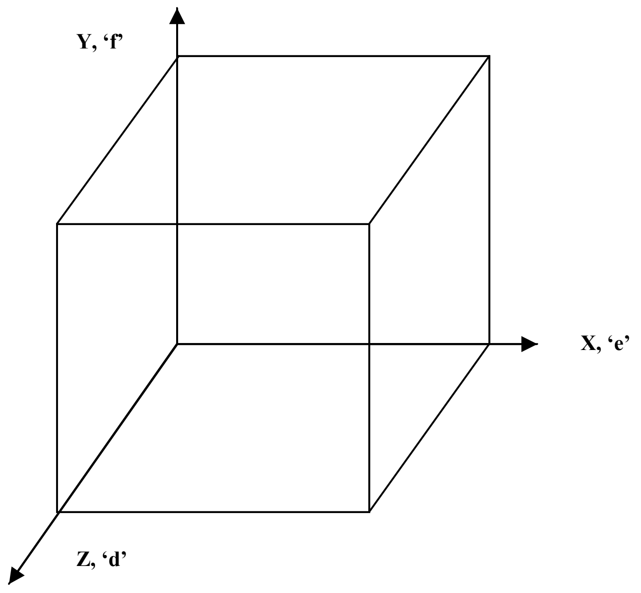

Figure 25: QoS Surface with the Dimensions

Since we are considering the case of a cube let us assume that all three dimensions play an equal part in determining the QOS surface i.e. all dimensions are of the same length 'y' units.

So, now we have

$$d \Leftrightarrow e \Leftrightarrow f \Leftrightarrow y$$

With the above equivalence we can calculate the surface area of the cube to be **'6y^2 square units'**

Thus we have the surface area of a cube which indicates the QOS surface area which can be obtained when using the properties of the QOS routing protocol with mobile backbones. The smaller the QOS surface area obtained for a protocol the greater the QOS associated with its usage. The inverse of equation 4 will be considered since we will be mapping the dimensions of both the attack surface and the QoS surface.

Theoretical Protocol:

Let us finally consider the case of the theoretical protocol. Now, let us assume that the dimensions are paired up as shown:

C == d, b == f, a == e.

So every dimension will have a minimum value of ‘z’ or ‘y’ units. Let us consider ‘x’ to be the length of each dimension of the theoretical protocol.

Case 1: x = 2c

$X = c+d \ (b+f, a+e)$

$= 2z$ or $2y$ units

So now we have the surface area of such a cube to be $6x^2$ square units which is equivalent to $36z^2$ square units or $36y^2$ square units.

Case 2: c < x < 2c

Let p1 (a+e)/2, p2 = (b+f)/2, p3 =(c+d)/2.

So now we have

$X = z + p1$ units

So the surface area of such a cube is $6x^2$ square units which is equivalent to $6(z+p1)^2$ square units.

This will apply to all other cases for the values of ‘X’.

From the above two cases we can see that when we try combining two different protocols which offer a high level of performance in 2 different categories, in our case “security” and “QOS”, the new protocol will not efficiently match them in their performances. There will be a tradeoff between the various dimensions which can result in a surface with larger surface areas in terms of the security and the QOS.

This provides a unifying approach for capturing the attack surface and the QoS surface. More work is required to provide a concrete framework.

CHAPTER VII

CONCLUSION

In this thesis we have proposed a mechanism to compare the relative security and QoS of sensor network protocols using the concept of a surface. The proposed state machine model is general enough to model the behavior of the system, the threats present, the QoS features available, the base station and the sensor nodes on the system. Our service surface measurement method can be applied to any sensor network system. The application of our metric and method to sensor networks give results that confirm perceived beliefs about the relative security of the communication protocols.

In this thesis, we have extended the metric of an attack surface proposed for operating systems to sensor network systems. The protocols are modeled using state transition diagrams and we have defined state transition graphs specifying various attacks on sensor networks. An attack surface models the sensor network system according to the dimensions identified – targets and enablers, channels and protocols, rules of access.

We have also extended this metric to a service surface and have provided the QoS surface for a sensor network to indicate the feasibility of this technique. Similar to creating an attack surface, we have studied the details of the communication protocol and made use of state transition diagrams to model the protocol.

We have defined state transition graphs for various QoS features provided in sensor networks. This QoS surface models the sensor network system according to the dimensions identified – inhibitors, enhancers and bottlenecks.

In this work we have provided an example to illustrate the effectiveness of the proposed approach. The example confirms that we can modify the dimensions involved with the surfaces to provide us a better level of service or security. We view our work as a first step towards a meaningful and practical metric for an integrated framework for security and QoS measurement.

Future work related to our approach will involve identifying more dimensions for the QoS and Security surfaces. Experimental validation of the proposed model is also needed. This will include identifying vulnerabilities, QoS features and classifying them efficiently across the dimensions. Further work on combining and synchronizing the best features of multiple protocols to create an improved protocol would be another avenue for future work.

References

[1] Pratyusa Manadhata, Jeannette M. Wing, “Measuring a System’s Attack Surface”, *13th USENIX Security Symposium*, 2004

[2] Michael Howard, Jon Pincus, and Jeannette M. Wing, “Measuring Relative Attack Surfaces”, *Workshop on Advanced Developments in Software and Systems*, 2003

[3] Christopher L. Barrett, Stephan J. Eidenbenz, Lukas Kroc, Madhav Marathe, James P. Smith, “Parametric Probabilistic Sensor Network Routing”, *Los Alamos National Laboratory Basic and Applied Simulation Science (CCS-5)*, 2003

[4] Chris Karlof, David Wagner, “Secure routing in wireless sensor networks: attacks and countermeasures”, *Ad Hoc Networks 1, pp. 293–315,* 2003

[5] J. Alves-Foss and S. Barbosa, “Assessing Computer Security Vulnerability”, *ACM SIGOPS Operating Systems Review 29, pp. 3-13,* 1995

[6] Barbara H. Liskov and Jeannette M. Wing, “A Behavioral Notion of Sub typing”, *ACM Transactions on Programming Languages and Systems 16, pp. 1811-1840,* 2003

[7] Sencun Zhu, Sanjeev Setia, Sushil Jajodia, “LEAP: Efficient Security Mechanisms for Large Scale Distributed Sensor Networks”, *Conference on Computer Communications and Security (CCS),* 2003

[8] Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler, J. D. Tygar, SPINS: “Security Protocols for Sensor Networks”, *Mobile Computing and Networking,* 2001

[9] Microsoft Security Bulletins,

http:// www.microsoft.com/ technet/ security/ current.asp (last accessed Jan 31, 2006)

[10] MITRE CVEs,

http://www.cve.mitre.org (last accessed Jan 31, 2006)

[11] Michael Howard, "Fending off Future Attacks by Reducing the Attack Surface," 2003 http://msdn.microsoft.com/library/default.asp?url=/library/enus/dncode/html/secure02132003.asp

[12] Butler Lampson, "Protection," *Operating Systems Review", vol. 8, no. 1, pp. 18–24,* 1974

[13] Butler Lampson, Martin Abadi, Michael Burrows, and Edward Wobber, "Authentication in distributed systems: Theory and practice," *ACM TOCS (Transactions on Computer Systems), Vol. 10, No. 4, pp. 265–310,* 1992

[14] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, Erdal Cayirci; "A Survey on Sensor Networks", *IEEE Communications Magazine,* 2002

[15] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva, "Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols". *Proc. 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking,* 1998

[16] Deepak Ganesan, Ramesh Govindan, Scott Shenker, Deborah Estrin; "Highly Resilient, Energy Efficient Multi-path Routing in Wireless Sensor Networks", *Mobile Computing and Communications Review, Vol. 1, No. 2,* 2003

[17] Zygmunt Haas, Joseph Y. Halpern, Li Li; "Gossip-Based Ad Hoc Routing", *Proceedings of INFOCOM 2002,* 2002

[18] Sergio D. Servetto, Guillermo Barranechea; "Constrained Random Walks on Random Graphs: Routing Algorithms for Large Scale Wireless Sensor Networks",

Proceedings of 1st ACM International Workshop and Wireless Sensor Networks and Applications (WSNA 2002), 2002

[19] Y.-C. Hu, A. Perrig, and D. B. Johnson, "Packet leashes: a defense against wormhole attacks in wireless networks", *IEEE Infocom,* 2003

[20] J.R. Douceur, "The Sybil attack", in: *1st International Workshop on Peer-to-Peer Systems (IPTPS _02),* 2002

[21] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, K. Pister, "System architecture directions for networked sensors", *Proceedings of ACM ASPLOS (Architectural Support for Programming Languages and Operating Systems) IX,* 2000

[22] V. D. Park, M. S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks", *IEEE INFOCOM _97, pp. 1405–1413,* 1997

[23] B. Zhang, Hussein T. Mouftah, "QoS Routing for Wireless Ad-Hoc networks: Problems, Algorithms, and Protocols", *IEEE Communications Magazine, pp. 110-117,* 2005

[24] G. Guinea, J. P. Thomas, "Crossing the Man-Machine Divide: A Mapping based on Empirical Results", *Journal of VLSI Signal Processing, pp. 139-147,* 2001

VITA

Arasavel Ramaraj

Candidate for the Degree of

Master of Science

Thesis: MODELING OF SENSOR NETWORKS - ATTACK SURFACE, QOS SURFACE

Major Field: Computer Science

Biographical:

Personal Data: Born in Chennai, Tamil Nadu, India, September 14, 1980, the son of Mr. S. Ramaraj and Mrs. R. Thamilmullai.

Education: Obtained Senior High School Diploma from St. Michael's Academy, Chennai, India, in May 1998; completed Bachelor of Engineering in Computer Science from Bharathiar University, Coimbatore, India, May 2002; fulfilled requirements for the Master of Science Degree at Oklahoma State University in May, 2006.

Name: Arasavel Ramaraj Date of Degree: May, 2006.

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: MODELING OF SENSOR NETWORKS - ATTACK SURFACE, QOS SURFACE

Pages in Study: 139 Candidate for the Degree of Master of Science

Major Field: Computer Science

Scope and Method of Study: The availability of low-power micro-sensors, actuators, radios and embedded processors is enabling the application of distributed wireless sensing to a wide range of applications such as environmental monitoring, condition based maintenance, military, transportation, factory instrumentation and inventory tracking. Security and Quality of Service (QoS) are two essential features of sensor networks. An integrated framework to measure security and QoS has been a long-standing challenge to the community. To determine a model for the security of a sensor network we need a measure— at a lower abstraction level—that allows us to determine the relative security or QoS of two or more systems or protocols. Given this intermediate viewpoint, we can determine if there are certain system features that are more likely than others to be opportunities of attack. The counts of the "more likely to be attacked" system features determine a system's vulnerability. Similarly we can determine if there are certain system features that provide QoS.

Findings and Conclusion: This thesis concentrates on the dimensions involved in modeling a sensor network, deploying a particular routing protocol, to create an attack surface. The effectiveness of this model (the attack surface) is shown by identifying the vulnerabilities present in the network with the aid of this new metric. This model has been found feasible to identify the vulnerabilities present in a sensor network system and thereby provides a relative measurement. The proposed approach also provides us with a means to increase the security of the sensor network along the dimensions identified for the attack surface. The notion of an attack surface has been extended to model the QoS of the sensor network. This QoS surface provides a means to identify the QoS issues and thereby increase the QoS of the sensor network. The two surfaces have been determined in a study of the security and QoS features of a multi – protocol heterogeneous sensor network system.

ADVISER'S APPROVAL: Dr. Johnson P. Thomas