A Framework for Automating Security Analysis of the Internet of Things

Mengmeng Ge, Jin B. Hong, Walter Guttmann, Dong Seong Kim

University of Canterbury
Private Bag 4800, Christchurch, New Zealand

Abstract

The Internet of Things (IoT) is enabling innovative applications in various domains. Due to its heterogeneous and wide-scale structure, it introduces many new security issues. To address this problem, we propose a framework for modeling and assessing the security of the IoT and provide a formal definition of the framework. Generally, the framework consists of five phases: 1) data processing, 2) security model generation, 3) security visualization, 4) security analysis, and 5) model updates. Using the framework, we can find potential attack scenarios in the IoT, analyze the security of the IoT through well-defined security metrics, and assess the effectiveness of different defense strategies. The framework is evaluated via three scenarios, which are the smart home, wearable healthcare monitoring and environment monitoring scenarios. We use the analysis results to show the capabilities of the proposed framework for finding potential attack paths and mitigating the impact of attacks.

Keywords: Attack Graphs, Internet of Things, Security Analysis, Security Modeling

1. Introduction

In the Internet of Things (IoT), every physical object becomes locatable, addressable and reachable in the virtual world [1, 2, 3]. As more and more objects in the physical world are expected to connect to the Internet, the IoT is supposed to contain millions or billions of objects which will communicate with each other and with other entities (e.g., human beings). These objects not only include computers and laptops which already exist in traditional networks, but also physical devices (such as home appliances), vehicles, *etc.* The heterogeneity of devices and technologies that are used for providing services has a great impact on the interoperability and management of IoT devices. Besides, many

Email addresses: mge43@uclive.ac.nz (Mengmeng Ge), jho102@uclive.ac.nz (Jin B. Hong), walter.guttmann@canterbury.ac.nz (Walter Guttmann), dongseong.kim@canterbury.ac.nz (Dong Seong Kim)

devices have constrained resources and limited computational capabilities and are deployed in an open environment (e.g., street lights), which makes them prone to being controlled or destroyed by malicious people. With its inherent complexity and heterogeneous structure, the IoT is facing numerous threats and attacks which will negatively affect its normal functionality. Thus protecting the security of the IoT is a difficult yet important task.

The motivation of our work lies within the field of security modeling for the IoT. Vulnerabilities of the IoT reside in different aspects, including devices (hardware, operating systems), communication protocols, service applications, service APIs and the design of the IoT architecture. By exploiting such vulnerabilities, an attacker can launch various attacks including eavesdropping, Denial of Service (DoS) attacks, node capture and node controlling [1]. With the presence of varied and complex attacks, the ability to discover potential attack scenarios (e.g., an attacker's paths to a target IoT device) and to mitigate the impact of malicious attacks becomes a critical issue. Research on modeling the security of the IoT is also very limited due to the pioneering nature of the IoT.

In this paper, we propose a framework for modeling and assessment of the security of the IoT. The framework is used to construct a graphical security model and a security evaluator to automate the security analysis of the IoT. More specifically, the graphical security model is based on the Hierarchical Attack Representation Model (HARM) [4] to capture potential attack paths in the network. We refer to our model as the extended HARM; it adds to the basic HARM another layer describing subnets and their connectivity. The security evaluator uses various security metrics to assess the security and interacts with an analytic modeling and evaluation tool, Symbolic Hierarchical Automated Reliability and Performance Evaluator (SHARPE) [5], to output the analysis results. The driving idea behind the framework is to mitigate the impact of potential attacks in the IoT and increase the IoT security level.

An earlier version of this paper appeared in [6], and we have extended the earlier version with (1) a formal definition of the framework, (2) a three-layer graphical security model (i.e., the extended HARM) for the IoT, (3) detailed calculation steps of security metrics and (4) a comprehensive evaluation using both heterogeneous and homogeneous networks.

To the best of our knowledge, this work is the first approach to use a graphical security model in modeling and assessing security for the IoT. The main contributions of this paper are summarized as follows:

- propose a framework for modeling and assessing security of the IoT (Section 4.1);
- develop a graphical security model to compute attack scenarios (Section 4.1);
- formally define this framework (Section 4.2);
- use various security metrics to carry out the analysis (Section 4.2); and

- evaluate the framework using three scenarios, including a smart home, wearable healthcare monitoring and environment monitoring (Section 5).

The rest of the paper is organized as follows. Section 2 gives background information about the graphical security model, HARM and the evaluation tool (SHARPE). Section 3 presents related work on existing security modeling approaches for the IoT and discusses their constraints. Our framework for modeling and analyzing security of the IoT is described and formally defined in Section 4. The framework is evaluated with three different scenarios in Section 5. Extensions and limitations of the framework are discussed in Section 6. Finally, Section 7 concludes the paper.

2. Background

We introduce the HARM and the extended HARM, which is used as our security model and the SHARPE, which is used as our external evaluator.

Graph-based and tree-based security models (e.g., attack graphs (AGs) [7], attack trees (ATs) [8]) have been widely used in assessing the security of systems. In graph-based attack models, an AG shows all possible sequences of attackers' actions that eventually reach the target. With increasing size of the network, calculation of a complete AG has exponential complexity, thus causing a scalability problem. In tree-based attack models, an AT is a tree with nodes representing attacks and the root representing the goal of attacks. It systematically presents potential attacks in the network. However, an AT does not explicitly reflect the sequences of attackers' actions.

In order to address the above issues, the two-layer HARM [4] was introduced to combine AGs and ATs. In the HARM, the upper layer captures the network reachability information and the lower layer represents the vulnerability information of each node in the network. The layers of the HARM can be constructed independently of each other. This decreases the computational complexity of calculating and evaluating the HARM compared with the calculation and evaluation of an AG. Thus, the HARM addresses the scalability problem of the single layer AG. Besides, by using an AG for the network reachability in the upper layer, the HARM can show the sequences of attackers' actions which cannot be captured by using an AT.

To further improve the scalability, the three-layer HARM was developed in [9] with the subnet reachability at the highest layer. In the three-layer HARM, the complexity of the security evaluation is further decreased because computations are grouped in each layer using a bottom-up approach. The mobility of devices (e.g., node addition or removal) can be easily adjusted in the three-layer HARM without reconstruction of the whole model. Additionally, more layers can be used based on different IoT scenarios. For example, a smart home with several networks (e.g., Wi-Fi, Bluetooth, *etc*) can be modeled using the three-layer HARM; a number of smart homes in an area can be modeled using the four-layer HARM with the home connectivity in the highest layer.

The SHARPE [5] is a software package for performance and reliability analysis of computer systems. It accepts a mathematical model of the system and analyzes it using various algorithms. Several model types are provided, for example, Markov chains, Semi-Markov chains, reliability block diagrams, fault trees and reliability graphs. Each model type supports at least one analysis algorithm; for example, fault trees have five analysis algorithms including reliability, unreliability, mean-time-to-failure, *etc.* Given the behavior of the components of a system in the form of time-dependent functions and the structure of the system in the form of a model type, the SHARPE can compute the behavior of the system as a function of time which is used for performance and reliability analysis.

3. Related Work

We discuss current work on security models for IoT networks and non-IoT networks.

3.1. Security Models for the IoT

Several papers focus on developing security modeling approaches for the IoT. We discuss them in three aspects: security frameworks, game-based security modeling and adaptive security models.

Security frameworks: some papers proposed a high-level description or a theoretical framework of security modeling without any simulation work or with incomplete analysis.

Radomirovic [10] proposed a dense IoT model along with a Dolev-Yao adversary model to address security and privacy issues of communication protocols in the IoT. The dense IoT is defined as an asynchronous communication network with high connectivity and ubiquitous functionality. An attacker model is also introduced in which the adversary has corruption and fingerprinting abilities. The paper pointed out future work towards a formal model limiting the adversary's capabilities.

Yang *et al.* [11] presented a high-level security framework for the IoT. The framework is based on a model encompassing three interlinked elements, which are communication, control and computation. They regarded the IoT as the linkage between control and computation. The computation algorithms have a direct influence on the end devices. As the direct control can be intervened by attackers, they put security control between computation and control. They concluded protecting IoT is not only a technical issue but also a social issue.

Stepanova *et al.* [12] proposed a theoretical framework for modeling the IoT security based on graph theory. By defining the IoT as "net of nets of things", they designed formalized network property indicators to assess the sustainability of nets of things (NoT) and described a method to maintain the sustainability of the NoT entities. Their future work includes the efficiency evaluation of the method with pre-defined indicators.

Atamli *et al.* [13] provided a threat model which consists of three sources of threats and eight types of attack vectors to determine where efforts should be

invested to secure systems. Using the threat model, they analyzed the impact of threats and deduced the security and privacy properties for the IoT based on three use cases: power management, smart car and smart healthcare system. Their future work includes the design of a security package that can be used for any use case.

Huang *et al.* [14] proposed a security framework named SecIoT under the 5th generation wireless system. SecIoT consists of a secure authentication system, which employs the multi-channel security protocol for device authentication, a role-based access control mechanism with fine-grained roles, and a risk indicator interface based on security risk analysis techniques. A prototype IoT was presented with an authentication protocol analysis and user acceptance studies on access control and risk indicator. The user studies indicated that a fine-grained role-based access control should be supported in the IoT and that a risk tree map is the best way to represent risks. Their future work includes the development of availability enhancement and trust management into the framework.

Ashra *et al.* [15] proposed an overview of threat mitigation approaches in the IoT based on "autonomic security". They classified these approaches into self-protecting, self-healing and hybrid of self-protecting and self-healing, and discussed their usage against different threats in three layers: machine-to-machine, network and cloud.

Game-based security modeling: several papers addressed game-based security modeling for the IoT. However, their scope either focused on mitigating the impact of certain attacks [16] or emphasized model solutions for specific domains [17, 18].

Hamdi *et al.* [17] constructed a Markov game-theoretic model to support decision making in the realm of IoT healthcare applications. Specifically, for smart things, the decision of whether or not to authenticate a forwarding packet is based on the assessment of power life, channel bandwidth, memory capacity and compromised nodes through the game-based model. The performance of the model was evaluated through simulation which showed smart things extend their lifetime by adopting the adaptive security policy.

Chen *et al.* [16] proposed a fusion-based defense mechanism to mitigate impacts of intentional attacks in the IoT architecture. They formulated a zero-sum game between the defense strategy and the attacker. In the worst-case scenario, the attacker knows the network topology and is capable of compromising all nodes simultaneously. The results of performance evaluation showed the robustness of the IoT was greatly enhanced by the proposed mechanism.

Rontidis *et al.* [18] developed a decision support method which minimizes security risks in the field of IoT prosumers selection. A prosumer offers applications or services in the IoT service deployment stages. They formulated a non-cooperative and complete information game between the user and the attacker. The worst-case scenario was considered where the attacker knows all security controls of prosumers. Following this scenario, a mixed strategy was proposed to randomize the prosumer selection in an optimal way and compared

with two heuristic solutions through simulation which proved the effectiveness of the strategy in mitigating security risks.

Adaptive security models: since 2012, adaptive security has been utilized in the Adaptive Security for Smart Internet of Things (ASSET) project which aims at developing risk-based adaptive security methods and mechanisms for the IoT. Adaptive security refers to a security solution that learns and adapts to changing environments dynamically, and identifies and responds to unknown threats. As the IoT is a dynamic system, security mechanisms implemented in the IoT should adapt to the dynamic context. There are a number of papers published from the project. However, their solutions were only designed for the eHealth domain.

Savola *et al.* [19] investigated security objectives of the IoT applications in an eHealth scenario and proposed the definition of a high-level adaptive security management mechanism using security metrics. The proposed mechanism is a cyclic process consisting of four critical models, which are adaptive security monitoring, analytics and predictive models, decision-making models, and metrics-based adaptive security models.

Abie *et al.* [20] introduced the adaptive framework with the emphasis on adaptive risk management. Based on the continuous cyclic process, the framework provides security solutions adaptively upon estimations of risk damage and benefits and evaluates solutions through security metrics. A patient-monitoring case study was indicated to be used for validating the framework in the future simulation experiment.

In order to accurately evaluate the adaptive security solutions (e.g., [20]) in real-life scenarios and realistic simulations, Berhanu *et al.* [21] presented a design and implementation of a testbed with heterogeneous biomedical sensors (Shimmer nodes and RaspberryPi Mini-PC with eHealth sensor shields). The testbed was set up using off-the-shelf hardware and open source software and then validated using the impact of antenna orientation on the energy consumption of sensors. The experimental results showed that the testbed functions well, thus being useful in studying the feasibility of the adaptive solutions.

Torjusen *et al.* [22] investigated the run-time adaptive behavior which deviates from the normal activities of the system. It was regarded as a major threat to the sustainability of IoT-enabled eHealth services. Based on the risk-based adaptive security framework in [20], Torjusen *et al.* developed a self-adaptive security framework by introducing run-time verification methods. Four run-time verification enablers were integrated into every phase of the initial feedback loop, which are models at run-time, requirements at run-time, dynamic context monitoring and a runtime verification component. The new framework was instantiated by means of Colored Petri Nets.

Hamdi *et al.*'s work mentioned above (i.e., [17]) is also part of the ASSET project. In their future work, they considered the implementation of the game-based model in the testbed [21].

Habib *et al.* [23] identified assets, vulnerabilities and threats for eHealth applications and proposed a threat detection and prevention mechanism based

on adaptive security. Generally, security events are at first gathered by sensors and monitoring components in devices, then analyzed to determine whether the events are threats or not. From the analysis, a planning function decides actions on the events via a knowledge base or learning mechanism. Thus, based on their analysis, the adaptive security mechanism is able to adjust security levels according to the threat levels.

3.2. Security Models For Non-IoT Networks

Graphical security models have been widely used for security analysis in various types of non-IoT networks. We discuss tree-based models, graph-based models, the HARM and related model generation tools.

Tree-based models: Mauw *et al.* [24] proposed a formal representation of ATs including the definition, transformations between ATs, calculations of attribute values associated with ATs and the projection algorithm applied to answer questions of people's interests (e.g., which attack causes damage over a certain limit).

Ten *et al.* [25] presented an analytical approach to evaluate the vulnerabilities in the supervisory control and data acquisition (SCADA) system using ATs. An AT was constructed according to attack goals and used to evaluate vulnerability indices for each attack leaf, each intrusion scenario and the overall system based on security conditions, countermeasures and password policies. A case study for the power system control network was conducted to identify possible break-in points and to evaluate the vulnerabilities.

Saini *et al.* [8] proposed the idea of threat modeling using ATs. They constructed an AT for an online certificate repository in the Grid Security Infrastructure toolkit and analyzed possible attacks and the impacts caused by the attacks.

Roy *et al.* [26] proposed attack countermeasure trees (ACTs) for the security analysis by taking into account both attacks and defense mechanisms. In the ACT, defense mechanisms can be deployed on any node of the tree instead of only leaf nodes. Qualitative and probabilistic analysis can be performed using the ACT to evaluate the security of the network. Besides, structural and Birnbaum importance measures can be used to prioritize attacks and countermeasures respectively. They implemented the ACT in the SHARPE and showed the usability of their model in three case studies: ACTs for a BGP attack, a SCADA attack and a malicious insider attack.

Graph-based models: Jha *et al.* [27] proposed an algorithm to generate AGs using a model checking technique for vulnerability analysis. Their algorithm can compute all potential attacks and contain only relevant states of the network and the intruder. They also designed minimization analysis approaches on attack graphs to formalize the security analysis and incorporated probabilities into AGs to perform reliability analysis.

Ou *et al.* [28] developed a vulnerability analysis tool to analyze the security impact of software vulnerabilities on networks. The tool automatically processes bug reports from existing vulnerability scanners and generates AGs to perform

the security analysis. The tool was implemented on Red Hat Linux. They tested the tool in a testbed with 500 Linux hosts connected via the Internet. The results showed their tool ran efficiently and identified a policy violation caused by vulnerabilities.

Ingols *et al.* [29] improved the AG generation tool designed in [30]. They considered client-side vulnerabilities, zero-day vulnerabilities and two common countermeasures including personal firewalls and intrusion prevention systems. They also redesigned the methods for computing network reachability to support reverse reachability (i.e., compute reachability from the malicious server backwards to the vulnerable clients). The experiments were carried out using a real network with 85 hosts and larger simulated networks. The results demonstrated that the enhanced tool is as scalable as their tool in [30]. Their future work includes modeling additional countermeasures, attacks and adversaries, and performing field tests.

Albanese *et al.* [31] used AGs to efficiently generate network hardening solutions. They defined a network hardening strategy as a set of atomic defense actions and introduced a cost model which takes into account the cost of interdependent actions. Then they designed an approximation algorithm to compute the minimum-cost hardening solution. The experiments were carried out using synthetic attack graphs and the results validated the performance of their approach. The evaluation of the proposed approach using real attack graphs will be included in their future work.

HARMs: Hong *et al.* [4] developed the two-layer graphical security model called HARM to assess the security of enterprise networks. The HARM is generated using network topology information in the upper layer and host vulnerability information in the lower layer. They performed complexity analysis on the HARM, AG and AT and concluded the HARM has smaller or equal computational complexity in the model construction, evaluation (i.e., calculation of attack paths) and update (e.g., host addition or removal) phases of the security analysis.

Hong *et al.* extended the previous paper and performed the scalability analysis of the multi-layered HARM in [9]. They compared the two-layer and three-layer HARMs with the single layer AGs in terms of model construction and evaluation. The simulation results demonstrated that the HARM is more scalable than the single layer AG. In particular, the three-layer HARM was found to be more scalable than the two-layer HARM.

Jia *et al.* [32] developed a software tool to generate AGs and HARMs from scanning reports, and to convert existing AGs into HARMs. They also designed a visualization tool to visualize AGs and HARMs. The feasibility of the tool was evaluated using an example enterprise network. Their future work includes supporting different types of HARMs (e.g., using ATs in the upper layer and AGs in the lower layer) and improving collection of network reachability information.

3.3. Summary

There is no previous work on constructing a formal graphical security model for analyzing the security of the IoT. There are several benefits of using a

graphical security model. Firstly, all potential attack paths can be captured in the model, whence solutions are no longer limited to defending against specific attacks. Moreover, the formal model can be used to analyze the security of various IoT scenarios. Lastly, it provides an intuitive way to analyze security weaknesses of systems and to evaluate potential countermeasures because sequences of attackers' actions are captured in the model. In our work, we focus on constructing a graphical security model along with the security evaluator and applying them to capture and analyze different attack scenarios for the IoT.

4. The Proposed Framework

The main goals of the framework are to identify all possible attack paths in the IoT, evaluate the security level of the IoT through security metrics, and assess the effectiveness of different defense strategies. The proposed framework is shown in Figure 1.

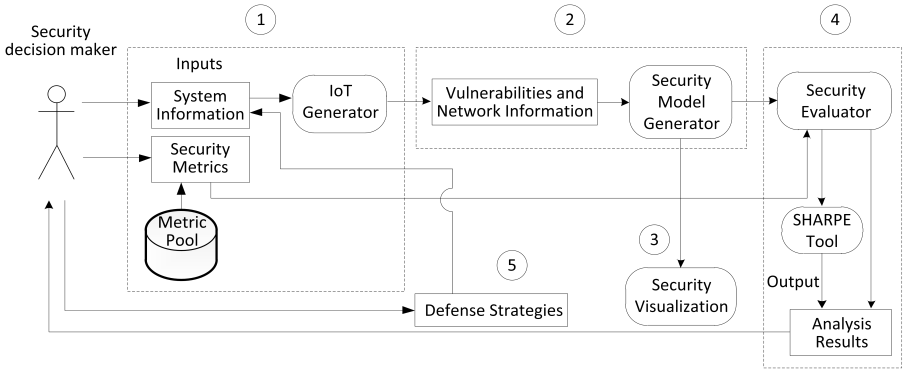

Figure 1: The proposed framework.

4.1. Framework Description

There are five phases in the framework: 1) data processing, 2) security model generation, 3) security visualization, 4) security analysis, and 5) model updates. We explain each phase in the following.

In phase 1, the security decision maker mainly provides two inputs needed to construct an IoT network: system information and security metrics. First, the system information includes the subnets forming the IoT, node information and network topology for each subnet, and the vulnerability information for each node. We use a static IoT network, thus all the inputs are fixed after the generation. Currently, the subnet classification method for IoT nodes is based on the communication protocols that devices use. Thus some devices may belong to several subnets due to the communication protocols they use. Then the system information is fed into the IoT Generator. The IoT Generator creates an IoT network consisting of the specified subnets with network topology

information and node vulnerability information. Second, the security decision maker also selects the security metrics, which will be used as an input into the security analysis phase.

In phase 2, we generate the extended HARM of the IoT network created in phase 1. Specifically, the Security Model Generator takes the constructed network with topology and vulnerability information as inputs and generates the extended HARM of the network. Based on the extended HARM, we compute all possible attack paths in the IoT network. An attack path specifies a sequence of nodes that the attacker can compromise to reach the target node. The model supports multiple attackers and multiple targets (e.g., attackers in different places, a group of devices as the targets).

The extended HARM is based on the HARM [33] and extends the HARM to three layers. They represent the subnet connectivity information in the upper layer, the network reachability information (i.e., nodes connected in the topological structure) in the middle layer and the vulnerability information of nodes in the lower layer, respectively. Compared to the basic HARM, the extended HARM additionally describes the subnets and their connectivity. Apart from generating the extended HARM for the overall IoT, we can choose various sets of subnets, construct the extended HARM for the chosen set of subnets and also combine several extended HARMs based on the subnet connectivity. When the IoT contains a large number of nodes, the subnet division makes the model construction and further security evaluation more flexible. Besides, by grouping devices into different subnets based on their communication protocols, we are able to model any IoT with a wide variety of heterogeneous communication technologies, thus addressing the heterogeneity issue of the IoT. As each layer is constructed independently, the extended HARM improves the scalability of the basic HARM.

In phase 3, the IoT network (including attack paths) is visualized in the form of an AG in the upper layer and middle layer, respectively, and a set of ATs in the lower layer.

In phase 4, the security analysis is carried out for the IoT network. The attack path information or other information (e.g., a set of nodes or vulnerabilities) is taken as an input into the Security Evaluator along with the determined security metrics. Based on the metrics, the Security Evaluator can perform one of two options. One option is to output the analysis results directly and the other option is to generate a textual input file and export the file into the SHARPE which computes the security analysis results. The description and calculation of security metrics are presented in Section 4.2.

In phase 5, any changes caused by the defense strategies are captured to update model inputs. Based on the security analysis results, the security decision maker knows which part of the IoT is the most vulnerable, thus being able to decide proper defense strategies. The deployment of a defense strategy changes either the vulnerability information (e.g., eliminates a specific vulnerability in an IoT node or mitigates the effect caused by the vulnerability) or the topology information (e.g., IoT node removal or addition), which is updated and taken as the input into the IoT Generator. The previous phases are

carried out again to re-analyze the security of the network after the deployment of the defense strategy. When choosing the defense strategies, the security decision maker can also assess the effectiveness of different strategies via the framework by using security metrics, comparing their effects and choosing the best one among them.

4.2. Framework Formulation

We formally describe the framework in terms of the network, subnet, node and vulnerability information, and then define the extended HARM.

4.2.1. General Notations

An IoT network has three major attributes, which are a finite set of subnets S, a finite set of IoT nodes T, and a finite set of vulnerabilities V. We denote a subnet as $s \in S$, a node as $t \in T$ and a vulnerability as $v \in V$. For one target, the attacker may be able to find multiple attack paths to reach it via one or multiple entry points. Thus we consider a set of all attack paths AP for a given target. Each attack path $ap \in AP$ is a sequence of nodes and each node in the path has one or more vulnerabilities. The definitions and notations of security metrics used in the framework are listed in Table 1.

The attributes of an IoT network $IoT = (S, T, V)$ are shown as follows:

- Each subnet $s \in S$ has a name s_{name}, a set of IoT nodes $s_{nodes} \subseteq T$, a topology information $s_{topo} \in \{tree, mesh, ...\}$, and a set of adjacent subnets $s_{adj} \subseteq S$ according to the network structure.
- Each node $t \in T$ has a name t_{name}, a type $t_{type} \in \{sensor, mobile\ device, ...\}$, a mobility information $t_{mobility} \in \{static, mobile\}$, a set of adjacent nodes $t_{adj} \subseteq T$ according to the network structure, a set of vulnerabilities $t_{vuls} \subseteq V$, and a set of security metrics $t_{metrics} \subseteq \{d_t, asp_t, ac_t, aim_t, mttc_t\}$.
- Each vulnerability $v \in V$ has a name v_{name}, a privilege level that is acquired by the attacker after the vulnerability is successfully exploited $v_{privilege} \in \{root, user, ...\}$, and a set of security metrics $v_{metrics} \subseteq \{asp_v, ac_v, aim_v, cr_v\}$.

4.2.2. Security Model Definition

We define the extended HARM based on the HARM [33]. The extended HARM has three layers: upper, middle and lower layers. The upper layer model (an AG) represents the subnet connectivity information and the attackers' entry points, the middle layer model (an AG) captures the network reachability information and the attackers' entry points, and the lower layer model (a set of ATs) depicts the vulnerability information of each node (if the node has vulnerabilities) and an attack goal achieved by the attackers by exploiting one or multiple vulnerabilities.

Definition 1. The extended HARM of an IoT network $IoT = (S, T, V)$ is defined as a 5-tuple $GSM = (U, M, L, C_{U,M}, C_{M,L})$. Here, U is an AG model

Table 1: Notations and definitions of security metrics.

Metrics	Notations	Definitions
Vulnerability level		
• Attack success probability	asp_v	Probability of an attacker to successfully exploit a vulnerability ([0, 1])
• Attack cost	ac_v	Cost spent by an attacker who successfully exploits a vulnerability ([0, 10])
• Attack impact	aim_v	Potential loss caused by an attacker who successfully exploits a vulnerability ([0, 10])
• Compromise rate	cr_v	Frequency of the attacker to successfully exploit a vulnerability per unit of time (one hour)
Node level		
• Node degree	d_t	Degree of a single node t
• Attack success probability	asp_t	Probability of an attacker to successfully compromise a node
• Attack cost	ac_t	Minimum cost spent by an attacker who successfully compromises a node
• Attack impact	aim_t	Maximum potential loss caused by an attacker who successfully compromises a node
• Mean-time-to-compromise	$mttc_t$	Mean time for the attacker to successfully compromise a node
Attack path level		
• Attack success probability	asp_{ap}	Probability of an attacker to successfully compromise the target via an attack path
• Attack cost	ac_{ap}	Cost spent by an attacker who successfully compromises the target via an attack path
• Attack impact	aim_{ap}	Potential loss caused by an attacker who successfully compromises the target via an attack path
• Mean-time-to-compromise	$mttc_{ap}$	Mean time for the attacker to successfully compromise the target via an attack path
Network level		
• Attack success probability	ASP	Probability of an attacker to successfully compromise the target via all potential attack paths
• Attack cost	AC	Minimum cost spent by an attacker who successfully compromises the target
• Attack impact	AIM	Maximum potential loss caused by an attacker who successfully compromises the target
• Mean-time-to-compromise	$MTTC$	Mean time for the attacker to successfully compromise the target in minimal attack sequences
• Average node connectivity	ANC	Average of local node connectivity over all pairs of nodes in the network

for S (the upper layer), M is an AG model for T (the middle layer) and L is a set of AT models for V (the lower layer). The relationship between components in the upper and middle layer is described by $C_{U,M} = \{(s, t) \mid s \in S \text{ and } t \in s_{nodes}\} \subseteq S \times T$. Each node that has one or more vulnerabilities has a corresponding AT in the lower layer; the partial mapping $C_{M,L} : T \rightarrow L$ gives the associated AT.

Definition 2. An AG is defined as a directed graph $ag = (N, E)$ where N is a finite set of components and $E \subseteq N \times N$ is a set of edges between components. Let k be the subnet including one or multiple attackers where $k \notin S$ and $k_{nodes} \cap T = \emptyset$. The representations of U and M are as follows:

- U: $N \subseteq S \cup \{k\}$ and $E \subseteq (S \cup \{k\}) \times S$
- M: $N \subseteq T \cup k_{nodes}$ and $E \subseteq (T \cup k_{nodes}) \times T$.

The restrictions on the edges imply that there are no edges into the attacker subnet or its nodes.

Definition 3. An AT is defined as a 5-tuple $at = (A, B, c, g, root)$. Here, A is a set of components which are the leaves of at and B is a set of gates which are the inner nodes of at. We require $A \cap B = \emptyset$ and $root \in A \cup B$. Let $\mathcal{P}(X)$ denote the power set of X. The function $c : B \rightarrow \mathcal{P}(A \cup B)$ describes the children of each inner node in at (we assume there are no cycles). The function $g : B \rightarrow \{AND, OR\}$ describes the type of each gate. The representation of the attack tree at_t associated to the node $t \in T$ is as follows:

- at_t: $A \subseteq t_{vuls}$.

This means that the vulnerabilities of a node are combined using logical *AND* and *OR* gates.

4.2.3. Security Metrics Calculation

The security metrics, shown in Table 1, are divided into four levels, which are the network, attack path, node and vulnerability levels. The values of some metrics in higher levels are calculated from lower levels in the security analysis phase. This is done for attack success probability, attack cost, attack impact and mean-time-to-compromise [34, 35]. For example, values in the network level are calculated from values in the attack path, node and vulnerability levels. The value of the attack success probability is in the range of zero to one, while the value of ac_v and aim_v is in the range of zero to ten. Take the attack success probability as an example. The larger the value is within the range, the higher the probability is for an attacker to exploit the vulnerability. By introducing the value range, we use standardized metric values as it is not easy to get the exact values of the security metrics from real-world scenarios. The Common Vulnerability Scoring System (CVSS) [36] uses a similar way to assess the severity of vulnerabilities.

We calculate the attack cost and attack impact using the Security Evaluator. For the attack success probability and mean-time-to-compromise, we use the Security Evaluator and the SHARPE. In the following calculations, for each

node $t \in T$ that has an attack tree $at_t = (A, B, c, g, root)$, we assign values to asp_v, ac_v, aim_v and cr_v for each $v \in A$ based on the CVSS and existing research papers that analyze the vulnerabilities.

Attack success probability: attack success probability is used to measure the probability of an attacker to successfully achieve an attack goal. At the node level, the metric is the probability for an attacker to compromise the node. At first, we calculate the attack success probability values for each inner node of an attack tree by Equation (1). Then the attack success probability value of a node $t \in T$ is the attack success probability value of the root of the corresponding attack tree by Equation (2). At the path level, the metric is the probability for an attacker to compromise the target via the attack path. The attack success probability value of an attack path is calculated by Equation (3). At the network level, the metric is the probability for an attacker to compromise the target via all potential paths. The network-level value ASP is calculated in Algorithm 1.

$$asp_b = \begin{cases} \prod\limits_{a \in c(b)} asp_a, & \substack{b \in B \\ g(b)=AND} \\ 1 - \prod\limits_{a \in c(b)} (1 - asp_a), & \substack{b \in B \\ g(b)=OR} \end{cases} \tag{1}$$

$$asp_t = asp_{root} \tag{2}$$

$$asp_{ap} = \prod_{t \in ap} asp_t, \quad ap \in AP \tag{3}$$

Algorithm 1: Calculation of ASP

Data: AP and asp_t $(t \in ap)$

Result: ASP

begin

  $H \leftarrow \{t \mid t \in ap \text{ for some } ap \in AP\}$

  Construct a directed graph $graph$ with node set H

  for *each attack path* $(t_1, ..., t_n) \in AP$ **do**

    for *each* $i \in \{2, ..., n\}$ **do**

      Include edge (t_{i-1}, t_i) with value $1 - asp_{t_i}$ in $graph$

    end

  end

  $ASP \leftarrow 1 - CalculateProbability(graph)$

end

In Algorithm 1, we use the reliability graph model in the SHARPE to calculate the probability that there is no attack path from the attacker to the target and then use 1 minus that probability to calculate ASP. Specifically, the SHARPE analyzes the reliability graph using the factoring algorithm [5]. After

factoring, if the sub-graph becomes series-parallel, its analysis can be done using Equation (4) where F is the distribution function of time variable i and J is the number of nodes included in the structure.

$$F(i) = \begin{cases} 1 - \prod_{j=1}^{J}[1 - F_j(i)], & \text{for a series structure} \\ \prod_{j=1}^{J} F_j(i), & \text{for a parallel structure} \end{cases} \tag{4}$$

Attack cost: attack cost is used to measure the cost spent by an attacker to successfully achieve an attack goal. At the node level, the metric is the cost spent by an attacker to compromise a node. Attack cost values for each inner node of an attack tree and each node $t \in T$ are calculated by Equations (5) and (6). At the path level, the metric is the cost spent by an attacker to compromise the target via the attack path. The attack cost value of an attack path is calculated by Equation (7). At the network level, the metric is the minimum cost spent by an attacker to compromise the target among all potential paths. The network-level value AC is thus given by Equation (8).

$$ac_b = \begin{cases} \sum\limits_{a \in c(b)} ac_a, & \substack{b \in B \\ g(b) = AND} \\ \min\limits_{a \in c(b)} ac_a, & \substack{b \in B \\ g(b) = OR} \end{cases} \tag{5}$$

$$ac_t = ac_{root} \tag{6}$$

$$ac_{ap} = \sum_{t \in ap} ac_t, \quad ap \in AP \tag{7}$$

$$AC = \min_{ap \in AP} ac_{ap} \tag{8}$$

Attack impact: attack impact is used to compute the potential loss caused by an attacker to successfully achieve an attack goal. The potential loss is the loss of confidentiality, integrity and availability. At the node level, the metric is the loss caused by an attacker to compromise a node. Attack impact values for each inner node of an attack tree and each node $t \in T$ are calculated by Equations (9) and (10). At the path level, the metric is the loss caused by an attacker to compromise the target via the attack path. The attack impact value of an attack path is calculated by Equation (11). In the network level, the metric is the maximum loss caused by an attacker to compromise the target among all potential paths. The network-level value AIM is thus given by (12).

$$aim_b = \begin{cases} \sum\limits_{a \in c(b)} aim_a, & \substack{b \in B \\ g(b) = AND} \\ \max\limits_{a \in c(b)} aim_a, & \substack{b \in B \\ g(b) = OR} \end{cases} \tag{9}$$

$$aim_t = aim_{root} \tag{10}$$

$$aim_{ap} = \sum_{t \in ap} aim_t, \quad ap \in AP \tag{11}$$

$$AIM = \max_{ap \in AP} aim_{ap} \tag{12}$$

Mean-time-to-compromise: mean-time-to-compromise is used to measure the mean time for an attacker to successfully achieve an attack goal. At the node level, the metric is the mean time for an attacker to compromise a node. If the node has only one vulnerability, which means the AT contains just one node with a compromise rate cr_{root}, we obtain the mean-time-to-compromise value $mttc_t$ by Equation (13). If the node has more than one vulnerability, which means the AT has more than one node, we use Algorithm 2.

$$mttc_t = 1/cr_{root} \tag{13}$$

Algorithm 2: Calculation of $mttc_t$

Data: at_t and cr_v ($v \in A$)

Result: $mttc_t$

begin

Create a tree *tree* with structure at_t and values cr_v as leaves

$mttc_t \leftarrow CalculateMean(tree)$

end

In Algorithm 2, we use the fault tree model in the SHARPE to calculate the mean-time-to-compromise. Specifically, the SHARPE analyzes the fault tree with repeated components using the factoring algorithm [5]. After factoring, if the sub-tree has no repeated components, its analysis can be done using Equation (14) where F is the distribution function of time variable i and J is the number of nodes included in the structure.

$$F(i) = \begin{cases} \prod_{j=1}^{J} F_j(i), & \text{for } AND \text{ gate} \\ 1 - \prod_{j=1}^{J} [1 - F_j(i)], & \text{for } OR \text{ gate} \end{cases} \tag{14}$$

At the path level, the metric is the mean time for an attacker to compromise the target via the attack path. We calculate $mttc_{ap}$ by Equation (15). At the network level, the metric is the minimum mean time for an attacker to compromise the target among all potential attack paths. We calculate $MTTC$ by Equation (16).

$$mttc_{ap} = \sum_{t \in ap} mttc_t, \quad ap \in AP \tag{15}$$

$$MTTC = \min_{ap \in AP} mttc_{ap} \tag{16}$$

5. Evaluation

The IoT has been widely applied in various fields, including healthcare, transport, environment monitoring, *etc*. We use three example networks in three different scenarios to show the usefulness of the framework. They are the home network in a smart home, the wireless body area network in the wearable healthcare monitoring scenario and the wireless sensor network for environment monitoring.

5.1. Sinkhole Attack in a Smart Home

A smart home is one of the application domains of the emerging IoT. It has come into thousands of families and brought new technologies to people's lives [37]. Unfortunately, it also provides a platform for attackers to hack into the home network and remotely control home systems. As many IoT devices are resource-constrained, standard security solutions may not be implemented. IoT devices can become entry points into the smart home and can then be exploited to leak sensitive information [38]. Thus the smart home environment is exposed to various threats. With an increasing number of Internet-connected devices in the house, vulnerabilities and related threats also increase. We describe the attack scenarios in the smart home and show the benefits of the framework via a use case.

5.1.1. Scenario Description

A smart home is formed by a number of home automation systems, which can autonomously operate devices and thus control the home on behalf of users [39].

ZigBee technology, an IEEE 802.15.4-based specification [40], is designed to be used by applications that require low data rate, low cost, low power consumption and two-way wireless communications. Some examples are home appliances (e.g., air conditioners, refrigerators, and washing machines), lighting control (e.g., light bulbs), environment monitoring (e.g., temperature, humidity) and security (e.g., smart door lock, surveillance camera).

The Wi-Fi standard has been widely established as the wireless home networking technology. It is designed to provide relatively high data rate communications. It can be used for multimedia applications of digital products in the home network (smartphones, smart TVs, tablets, *etc*).

5.1.2. Network Setting

We consider the IoT-enabled home network shown in Figure 2 as an example. The home network is a heterogeneous network with devices using different operating systems, applications and communication protocols. It includes a ZigBee network and a Wi-Fi network. As ZigBee and Wi-Fi can coexist with less interference problems than alternative technologies, the combination of them has the potential to provide comprehensive home network solutions [41]. A smart home automation hub is used to support Wi-Fi, ZigBee and Internet communications. Specifically, the integrated hub is able to establish a ZigBee network that allows home devices to communicate with each other by using

the ZigBee wireless protocol; it provides the Internet connection for the ZigBee network and the Wi-Fi network; it also provides a user interface control panel so that users can connect to the hub through the Internet to get access to ZigBee devices and remotely control them [42].

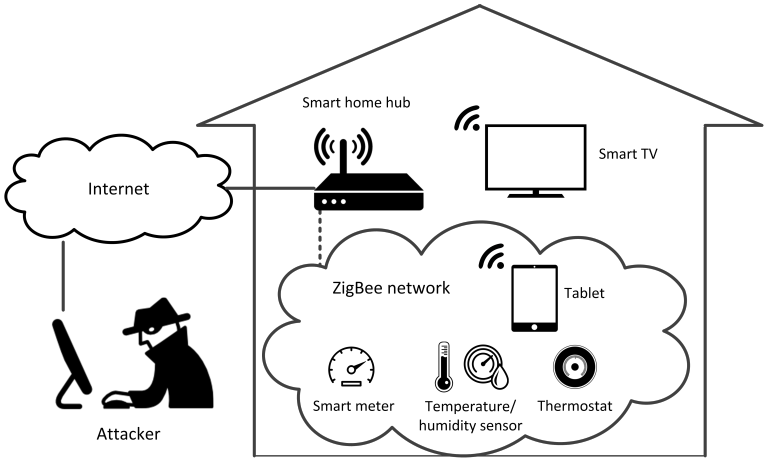

Figure 2: A smart home scenario.

The ZigBee network contains heterogeneous sensors [43, 44]. Our use-case has a number of ZigBee devices presented in the emulated environment in [44], such as electricity meters, thermostats, temperature and humidity measurement sensors. ZigBee devices communicate wirelessly to the hub (acting as the coordinator) in the form of a mesh topology. Some devices act as routers to extend the limited range of the network (e.g., the electricity meter). They can transfer packets to/from other ZigBee devices. Some devices are end devices thus only interacting with a router or the hub (e.g., thermostat, temperature and humidity measurement sensor).

We use an Android tablet equipped with a ZigBee chip. It connects to both the Wi-Fi network and the ZigBee network and acts as a ZigBee router in the ZigBee network. We also use a Smart TV which connects to the Wi-Fi network. Both of them get access to the Internet through the smart home hub.

5.1.3. An Attacker Model

We assume the attacker's goal is to lure the traffic from the smart home hub through a compromised device as the ZigBee routing algorithm is prone to Sinkhole attacks [44]. The assumptions about the attacker's ability are listed in the following.

1. The attacker is able to remotely compromise the Smart TV. In the literature, there are several papers addressing remote attacks on smart

TVs [45, 46]. According to the practical proof-of-concept attacks or experiments introduced in the papers, the attackers can remotely exploit software vulnerabilities on a smart TV without physical proximity to the target and gain control over it. Then they can use it as a gateway to exploit vulnerabilities in any other devices inside the home (e.g., the tablet).

2. The attacker is able to compromise the Android tablet with a specific malware exploiting several bugs in the software and the operating system [44]. After the tablet is compromised, the attacker can use the tablet to launch other attacks targeting the ZigBee network. We use the Sinkhole attack as an example of a further attack.

Specifically, we use the attacks introduced in [46, 44]. For the smart TV, the attacker can construct a malicious media file by using FFmpeg to find exploitable vulnerabilities in supported media formats and upload the file on the Internet. After the victim downloads the malicious file and starts to play back the video file, the TV is compromised. We assume FFmpeg 5.0 is used by the TV. Two vulnerabilities are found in two types of media file formats supported by FFmpeg and the attacker can exploit any one of them to run arbitrary code and gain the root privilege of the TV. The information about the two vulnerabilities in the Common Vulnerabilities and Exposures (CVE) and their CVSS base scores are summarized in Table 2. For the Android tablet, the attacker can write a malware to get the root permission and change the transmission power of the ZigBee chip integrated in the device. The malware was developed based on a malfunctional Trojan, *Backdoor.AndroidOS.Obad.a*. According to [47], it exploits three bugs: firstly, an error in the DEX2JAR software was used to disrupt the conversion of Dalvik bytecode into Java bytecode, which complicates the statistical analysis of the Trojan; secondly, an error in the Android operating system was used to modify the AndroidManifest.xml file, which makes a dynamic analysis of the Trojan extremely hard; thirdly, a previously unknown error in the Android operating system was used to obtain extended Device Administrator privileges without appearing on the list of applications which have such privileges, which makes the detection impossible.

Table 2: Vulnerability information in the TV.

CVE ID	CVSS Base score	Impact	Exploitability
CVE-2008-4866	10.0	10.0	10.0
CVE-2009-0385	9.3	10.0	8.6

5.1.4. Data Processing

As the home network consists of devices using WiFi and/or ZigBee communication protocols, we introduce two subnets to differentiate heterogeneous devices based on our classification method. The subnets are denoted as s_{wifi} and s_{zigbee} respectively. In the ZigBee network, we use 5 devices acting as routers and 3 end devices attached to each router. ZigBee routers are denoted as t_{ri}

($i \in \{1, 2, ..., 5\}$) and ZigBee end devices are denoted as t_{ej} ($j \in \{1, 2, ..., 15\}$). The smart home hub denoted as t_{hub} and an Android tablet denoted as t_{tab} belong to both ZigBee network and Wi-Fi network. The Wi-Fi network also includes a TV denoted as t_{tv}. In the IoT generator, the IoT network is represented as $IoT_1 = (S_1, T_1, V_1)$ where S_1=$\{s_{wifi}, s_{zigbee}\}$, T_1=$\{t_{hub}, t_{tv}, t_{tab}, t_{r1}, ..., t_{r5}, t_{e1}, ..., t_{e15}\}$ and V_1=$\{v_{tv1}, v_{tv2}, v_{tab1}, v_{tab2}, v_{tab3}\}$. Here, v_{tv1} and v_{tv2} refer to two vulnerabilities found in two types of media file formats in the TV, namely CVE-2008-4866 and CVE-2009-0385 in Table 2. Three software bugs in the tablet are denoted as v_{tab1}, v_{tab2} and v_{tab3}.

We show the full list of attributes for a subnet s_{zigbee}, a node t_{tv} and a vulnerability v_{tv1} as examples.

- $s_{zigbee_{name}} = zigbee$
- $s_{zigbee_{nodes}} = \{t_{tab}, t_{r1}, ..., t_{r5}, t_{e1}, ..., t_{e15}\}$
- $s_{zigbee_{topo}} = mesh$
- $s_{zigbee_{adj}} = \{s_{wifi}\}$
- $t_{tv_{name}} = tv$
- $t_{tv_{type}} = smart\ device$
- $t_{tv_{mobility}} = static$
- $t_{tv_{adj}} = \{t_{hub}, t_{tab}\}$
- $t_{tv_{vuls}} = \{v_{tv1}, v_{tv2}\}$
- $t_{tv_{metrics}} = \{d_t, asp_t, ac_t, aim_t, mttc_t\}$
- $v_{tv1_{name}} = CVE\text{-}2008\text{-}4866$
- $v_{tv1_{privilege}} = root$
- $v_{tv1_{metrics}} = \{asp_v, ac_v, aim_v, cr_v\}$

Based on the vulnerabilities described in Section 5.1.3, we make assumptions about the metric values of vulnerabilities in the TV and the Android tablet and show the values in Table 3. For the values of vulnerabilities in the TV, we use the impact values in Table 2 for the values of the attack impact and estimate the values of the other three security metrics from the exploitability values in the same table. Both v_{tv1} and v_{tv2} allow an attacker to exploit from the Internet without any authentication. However, v_{tv1} has low access complexity while v_{tv2} has medium access complexity. Besides, we assume the victim has a probability of 0.5 to download the malicious file after its distribution. Thus combined with the downloading probability of the victim, we use medium attack success probability and low attack cost for v_{tv1} and low attack success probability and medium attack cost for v_{tv2}. The compromise rate indicates the frequency that the vulnerability can be exploited successfully. We estimate the compromise

rate as once per week as the victim might download the video files at weekends and accidentally get a malicious one.

For the metric values of vulnerabilities in the Android tablet, we can estimate the values based on the descriptions as no CVSS scores are available. All three vulnerabilities in the tablet allow an attacker to exploit from the Internet without any authentication. We assume they have low access complexity as the tools used by an attacker to exploit vulnerabilities are easy to obtain. Thus, we use high attack success probability and low attack cost. As people might use their tablets every day and accidentally download the malware, we assume the compromise rate as twice per week. Vulnerabilities v_{tab1} and v_{tab2} can be exploited to complicate the analysis of the Trojan, which are assumed to have low attack impact. Vulnerability v_{tab3} is used to obtain the extended privileges which is assumed to have high attack impact.

Table 3: Metric values for each vulnerability.

Metric / Vulnerability	asp_v	ac_v	aim_v	cr_v
v_{tv1}	0.45	1.0	10.0	0.006
v_{tv2}	0.3	5.0	10.0	0.006
v_{tab1}	0.8	3.0	2.0	0.012
v_{tab2}	0.8	3.0	2.0	0.012
v_{tab3}	0.8	3.0	10.0	0.012

5.1.5. Security Model Generation and Visualization

We use $IoT_1 = (S_1, T_1, V_1)$ as the input into the Security Model Generator and compute the extended HARM. The model is represented as $GSM_1 = (U_1, M_1, L_1, C_{U_1,M_1}, C_{M_1,L_1})$.

As an example of an attack graph, we show U_1=($\{k_1,\ s_{zigbee},\ s_{wifi}\}$, $\{k_1 \rightarrow s_{wifi},\ s_{wifi} \rightarrow s_{zigbee}\}$).

As an example of an attack tree, we show $at_{t_{tv}}$=($\{v_{tv1},\ v_{tv2}\}$, $\{root_{tv}\}$, $c(root_{tv})$=$\{v_{tv1},\ v_{tv2}\}$, $g(root_{tv})$=OR, $root_{tv}$).

We use two subnets in the upper layer of the model to represent s_{zigbee} and s_{wifi}. Figure 3 shows the visualized attack path in the network captured by the model. By exploiting the vulnerabilities, the attacker is able to bypass the smart home hub and break into the home network via the smart TV.

5.1.6. Security Analysis and Model Updates

In the Sinkhole attack, more devices choose the malicious tablet to route their data to the smart home hub as the compromised tablet represents a shorter route to the hub with the increased power and increased probability of successfully delivering the packets [44]. We assume the compromised tablet refuses to deliver any packets to/from the hub. Thus we analyze the impact of the attack using the average node connectivity of the network (ANC_1) and the degree of the hub and the tablet ($d_{t_{hub}}$ and $d_{t_{tab}}$) shown in Table 4. In our

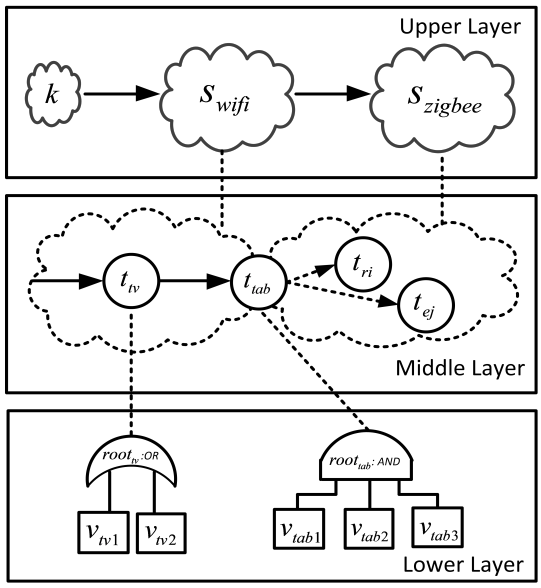

Figure 3: The attack paths in the smart home.

example network, under the Sinkhole attack, the average node connectivity drops as the malicious tablet disconnects with the hub and the TV which partitions the network into two separate parts; the degree of the hub decreases while the degree of the tablet increases as some routers and end devices cut off their initial connections and connect to the tablet because of its higher transmission power.

Table 4: The impact of the Sinkhole attack on the network connectivity.

Metric / Situation	ANC_1	$d_{t_{hub}}$	$d_{t_{tab}}$
Before the attack	1.1146	4	2
After the attack	1.1028	3	10

As there is only one attack path ap=(t_{tv}, t_{tab}) in the network, we calculate values of security metrics in the node and attack path levels.

Attack success probability: we calculate $asp_{t_{tv}}$ and $asp_{t_{tab}}$ by Equations

(1) and (2), asp_{ap} by Equation (3).

$$\begin{aligned} asp_{t_{tv}} &= asp_{root_{tv}} = 1 - (1 - asp_{v_{tv1}}) * (1 - asp_{v_{tv2}}) \\ &= 1 - (1 - 0.45) * (1 - 0.3) = 0.615 \\ asp_{t_{tab}} &= asp_{root_{tab}} = asp_{v_{tab1}} * asp_{v_{tab2}} * asp_{v_{tab3}} \\ &= 0.8 * 0.8 * 0.8 = 0.512 \\ asp_{ap} &= asp_{t_{tv}} * asp_{t_{tab}} = 0.615 * 0.512 \approx 0.314 \end{aligned}$$

Attack cost: we calculate $ac_{t_{tv}}$ and $ac_{t_{tab}}$ by Equations (5) and (6), ac_{ap} by Equation (7).

$$\begin{aligned} ac_{t_{tv}} &= ac_{root_{tv}} = min(ac_{v_{tv1}}, ac_{v_{tv2}}) \\ &= min(1.0, 5.0) = 1.0 \\ ac_{t_{tab}} &= ac_{root_{tab}} = ac_{v_{tab1}} + ac_{v_{tab2}} + ac_{v_{tab3}} \\ &= 3.0 + 3.0 + 3.0 = 9.0 \\ ac_{ap} &= ac_{t_{tv}} + ac_{t_{tab}} = 1.0 + 9.0 = 10.0 \end{aligned}$$

Attack impact: we calculate $aim_{t_{tv}}$ and $aim_{t_{tab}}$ by Equations (9) and (10), aim_{ap} by Equation (11).

$$\begin{aligned} aim_{t_{tv}} &= aim_{root_{tv}} = max(aim_{v_{tv1}}, aim_{v_{tv2}}) \\ &= max(10.0, 10.0) = 10.0 \\ aim_{t_{tab}} &= aim_{root_{tab}} = aim_{v_{tab1}} + aim_{v_{tab2}} + aim_{v_{tab3}} \\ &= 2.0 + 2.0 + 10.0 = 14.0 \\ aim_{ap} &= aim_{t_{tv}} + aim_{t_{tab}} = 10.0 + 14.0 = 24.0 \end{aligned}$$

Mean-time-to-compromise: we use Algorithm 2 to calculate $mttc_{t_{tv}}$ and $mttc_{t_{tab}}$. The SHARPE outputs are shown in the following: $mttc_{t_{tv}} = 83.333$ and $mttc_{t_{tab}} = 152.78$. We also calculate $mttc_{ap}$ by Equation (15).

$$mttc_{ap} = mttc_{t_{tv}} + mttc_{t_{tab}} = 83.333 + 152.78 \approx 236.11$$

From the metric values in the node level, we can see that attacking the TV has higher success probability, lower cost, lower mean-time-to-compromise but lower impact than attacking the tablet. Thus, the attacker is more likely to choose the TV as the entry point. We should protect the TV at first in order to prevent the attacker from breaking into the network.

We assume patching is used to fix the software bug existing in the TV. One strategy is to patch v_{tv1} and denoted as $Defense_{v_{tv1}}$ while another is to patch v_{tv2} and denoted as $Defense_{v_{tv2}}$. We modify the vulnerability information for the TV, reconstruct the IoT network using the IoT Generator, compute the extended HARM and calculate the metric values after patching either v_{tv1} or v_{tv2}. The results calculated in the security analysis phase are shown in Table 5. Due to the limited space, we omit the detailed calculation steps.

Table 5: Security analysis of the attack paths.

Strategy \ Metric	asp_{ap}	ac_{ap}	aim_{ap}	$mttc_{ap}$
No defense	0.314	10.0	24.0	236.113
$Defense_{v_{tv1}}$	**0.15**	**14.0**	24.0	319.44
$Defense_{v_{tv2}}$	0.23	10.0	24.0	319.44

For both $Defense_{v_{tv1}}$ and $Defense_{v_{tv2}}$, asp_{ap} decreases and $mttc_{ap}$ increases, which means both strategies are effective to lower the attack success probability and to extend the mean-time-to-compromise, while aim_{ap} does not change as the impact values of v_{tv1} and v_{tv2} are the same. For ac_{ap}, as exploiting v_{tv1} requires less cost than exploiting v_{tv2}, deploying $Defense_{v_{tv1}}$ incurs more cost for the attacker than deploying $Defense_{v_{tv2}}$. For asp_{ap}, as exploiting v_{tv1} has higher success probability than exploiting v_{tv2}, deploying $Defense_{v_{tv1}}$ decreases the attack success probability more than deploying $Defense_{v_{tv2}}$. If the defender is only able to deploy one strategy to protect the TV, $Defense_{v_{tv1}}$ should be chosen as it causes lower attack success probability and more cost for the attacker.

5.1.7. Summary

Using the framework, one can find potential attack paths, decide which devices included in the paths should be protected at first and compare the effectiveness of different device-level strategies based on the evaluation of various security metrics. As a result, one can choose the most effective device-level security strategies for specific devices.

5.2. Node Controlling in Wearable Healthcare Monitoring

The emerging IoT has provided many benefits to the improvement of e-health applications. One application is the vital sign monitoring in hospitals [48], which uses wireless sensing technology to provide continuous monitoring for patients. As the data collected from the patients is sensitive, security threats may put a patient into a critical condition (e.g., lack of treatment or wrong treatment).

5.2.1. Scenario Description

We consider the wireless body area network (WBAN) which has been widely applied in wearable healthcare monitoring. It allows the vital physiological parameters of patients to be collected by wearable or implantable sensors and transmitted using short-range wireless communication techniques (e.g., IEEE 802.15.4 [49] or ZigBee [40]). In the WBAN, communications can be divided into two parts: intra-body and extra-body [50]. The intra-body communication network transmits data between the monitor sensors placed on the human body and the coordinator device (which is in charge of collecting data from monitor sensors and sending it to the external network). The extra-body communication network transmits data between the coordinator device and an external network (e.g., the hospital network providing local data processing and remote access via the Internet).

5.2.2. Network Setting

We use the intra-body communication in the WBAN in Figure 4 as an example. It shows 9 sensor nodes placed on the human body along with a coordinator device (e.g., PDA). The network is a heterogeneous network as nodes have different applications to measure different health data. For example, sensor node sn_1 measures the heart rate and the electrocardiogram (ECG) and sn_9 senses the blood oxygen.

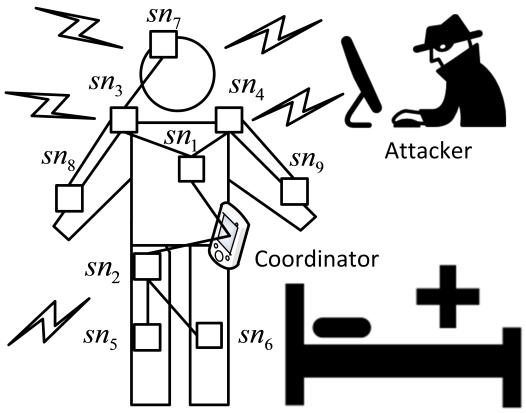

Figure 4: An intra-body communication network in the WBAN.

We assume a tree-based routing protocol is used for the intra-body communication [51] and the network topology does not change. Communications between sensor nodes and the coordinator device are single-hop or multi-hop. Data packets are sent to the coordinator device at pre-determined times or immediately when an emergency event occurs. Each sensor node runs the same operating system (e.g., TinyOS 2.x) with different applications and has a buffer overflow vulnerability in the operating system [52]. The coordinator device receives all information from sensors and provides an interface towards the hospital network. A key management scheme is used to protect data confidentiality, data integrity and data authentication [53].

5.2.3. An Attacker Model

We assume the attacker's goal is to compromise a sensor node that stores critical data on it and manipulate the content of the data packets sent from the node. The attacker model describes the attacker's capabilities as follows:

1. The attacker is able to get into the hospital. However, as sensors are deployed on the human body, it is difficult for an attacker to physically access nodes without being detected. Thus the attacker can only communicate with the sensor nodes in its radio range.

2. The attacker has a laptop-class device. He can exploit the buffer overflow vulnerability targeting the operating system to compromise a sensor node within an accepted time. Once a node is compromised, the attacker has full control (e.g., steal cryptographic keys, obtain routing table, inject and run arbitrary code). He can also reprogram the compromised node into a malicious node and exploit it to compromise other nodes.
3. The coordinator device is assumed to be strongly protected such that the attacker cannot compromise the coordinator.

5.2.4. Data Processing

Based on our subnet classification method (communication protocol), we introduce one subnet for the whole network as all sensor nodes use radio communication. The subnet is denoted as s_{wban}. Each sensor node has the same buffer overflow vulnerability; we denote this vulnerability as v_{sn}. This vulnerability allows an attacker to exploit within the communication range of the sensor node without any authentication and has low access complexity as the tools used by the attacker are easy to obtain. Thus we use high attack success probability and low attack cost. After exploiting the vulnerability, the attacker has full control of the sensor node. Thus we use high attack impact. We also assume the compromise rate as once per week as the attacker needs to be in the hospital to get access to the nodes. Estimated values of security metrics for the vulnerability are shown in Table 6.

Table 6: Metric values for the vulnerability.

Metric / Vulnerability	asp_v	ac_v	aim_v	cr_v
v_{sn}	0.8	3.0	10.0	0.006

5.2.5. Security Model Generation and Visualization

We assume the attacker's goal is to compromise sn_1 which measures the heart rate and the ECG information and manipulate the data sent from it to cause wrong treatment. The attacker is supposed to take either sn_4 or sn_9 as the access point by compromising it and exploiting it to compromise other nodes. We use one subnet in the upper layer and compute the extended HARM. Figure 5 shows the visualized attack paths in the network. As each node has the same vulnerability, we show only one v_{sn} in the lower layer. By exploiting the vulnerability, the attacker is able to compromise a series of nodes and control them for malicious purpose.

5.2.6. Security Analysis and Model Updates

We calculate the values of security metrics in the node, attack path and network levels. Network level metrics are denoted as ASP_2, AC_2, AIM_2 and $MTTC_2$. We define ap_1=(t_{sn_9}, t_{sn_4}, t_{sn_1}) and ap_2=(t_{sn_4}, t_{sn_1}).

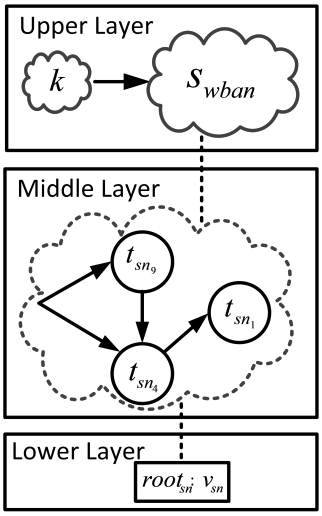

Figure 5: The attack paths in the intra-body communication network.

Attack success probability: as each sensor has only one vulnerability, we calculate $asp_{t_{sn_i}}$ by Equation (2). We also calculate asp_{ap_1} and asp_{ap_2} by Equation (3). ASP_2 is calculated using Algorithm 1 in which the SHARPE output is 0.7424.

$$
\begin{aligned}
asp_{t_{sn_i}} &= asp_{root_{sn_i}} = 0.8 \\
asp_{ap_1} &= asp_{t_{sn_9}} * asp_{t_{sn_4}} * asp_{t_{sn_1}} = 0.8 * 0.8 * 0.8 = 0.512 \\
asp_{ap_2} &= asp_{t_{sn_4}} * asp_{t_{sn_1}} = 0.8 * 0.8 = 0.64
\end{aligned}
$$

Attack cost: we calculate $ac_{t_{sn_i}}$ by Equation (6). We also calculate ac_{ap_1} and ac_{ap_2} by Equation (7), AC_2 by Equation (8).

$$
\begin{aligned}
ac_{t_{sn_i}} &= ac_{root_{sn_i}} = 3.0 \\
ac_{ap_1} &= ac_{t_{sn_9}} + ac_{t_{sn_4}} + ac_{t_{sn_1}} = 3.0 + 3.0 + 3.0 = 9.0 \\
ac_{ap_2} &= ac_{t_{sn_4}} + ac_{t_{sn_1}} = 3.0 + 3.0 = 6.0 \\
AC_2 &= min(ac_{ap_1}, ac_{ap_2}) = min(9.0, 6.0) = 6.0
\end{aligned}
$$

Attack impact: we calculate $aim_{t_{sn_i}}$ by Equation (10). We also calculate

aim_{ap_1} and aim_{ap_2} by Equation (11), AIM_2 by Equation (12).

$$\begin{aligned}
aim_{t_{sn_i}} &= aim_{root_{sn_i}} = 10.0 \\
aim_{ap_1} &= aim_{t_{sn_9}} + aim_{t_{sn_4}} + aim_{t_{sn_1}} = 10.0 + 10.0 + 10.0 = 30.0 \\
aim_{ap_2} &= aim_{t_{sn_4}} + aim_{t_{sn_1}} = 10.0 + 10.0 = 20.0 \\
AIM_2 &= max(aim_{ap_1}, aim_{ap_2}) = max(30.0, 20.0) = 30.0
\end{aligned}$$

Mean-time-to-compromise: we calculate $mttc_{ap_1}$and $mttc_{ap_2}$ by Equations (13) and (15), $MTTC_2$ by Equation (16).

$$\begin{aligned}
mttc_{ap_1} &= mttc_{t_{sn_9}} + mttc_{t_{sn_4}} + mttc_{t_{sn_1}} = 1/cr_{t_{sn_9}} + 1/cr_{t_{sn_4}} + 1/cr_{t_{sn_1}} \\
&= 166.66 + 166.66 + 166.66 \approx 500.0 \\
mttc_{ap_2} &= mttc_{t_{sn_4}} + mttc_{t_{sn_1}} = 1/cr_{t_{sn_4}} + 1/cr_{t_{sn_1}} \\
&= 166.66 + 166.66 \approx 333.3 \\
MTTC_2 &= min(mttc_{ap_1}, mttc_{ap_2}) = min(500.0, 333.33) = 333.3
\end{aligned}$$

From the metric values in the attack path level, we can see that exploiting ap_2 has higher success probability, lower cost, lower mean-time-to-compromise but lower impact than exploiting ap_1 as there are more nodes in ap_1 which need to be compromised by the attacker. Thus protecting nodes in ap_2 is more effective against the attack.

In terms of the defense strategy for the buffer overflow, we can deploy the method of address space layout randomization (ASLR) for the node, denoted as $Defense_{ASLR}$. The ASLR method is based upon the low chance of an attacker guessing locations of randomly placed areas, thus enhancing the security by increasing the search space. We make the assumptions on the metric values of the vulnerability after deploying the ASLR in Table 7. We decrease the attack success probability and compromise rate, and increase the attack cost as the method can only complicate the attack but not eliminate the vulnerability. The method does not affect the attack impact as the impact measures the loss after the vulnerability is exploited.

Table 7: Metric values for the vulnerability.

Metric / Strategy	asp_v	ac_v	aim_v	cr_v
$Defense_{ASLR}$	0.5	5.0	10.0	0.003

We assume the defender wants to deploy the ASLR defense strategy on one device because of the budget limit. We modify the vulnerability information in each sensor node in the attack path, reconstruct the IoT network using the IoT Generator, compute the extended HARM and calculate the metric values. From the metric values in the network level shown in Table 8, we can assess the effectiveness of the strategy deployed on each node.

For $Defense_{ASLR}$ on sn_9, only ASP_2 decreases while all other values remain

Table 8: Security analysis of the network.

Strategy \ Metric	ASP_2	AC_2	AIM_2	$MTTC_2$
No defense	0.74	6.0	30.0	333.33
$Defense_{ASLR}$ on sn_9	0.70	6.0	30.0	333.33
$Defense_{ASLR}$ on sn_4	0.56	8.0	30.0	499.99
$Defense_{ASLR}$ on sn_1	**0.46**	8.0	30.0	499.99

the same. For $Defense_{ASLR}$ on sn_4 and sn_1, ASP_2 decreases and ASP_2 is lower when using the ASLR on sn_1 since sn_4 and sn_1 have different locations in the network; AC_2 and $MTTC_2$ increase; AIM_2 does not change as the metric values do not change before and after the defense strategy. Thus, based on the analysis results, we can see that protecting sn_1 is more effective than protecting either of other two nodes.

5.2.7. Summary

Using the framework, one can compare the severity of multiple potential attack paths and the effectiveness of specific device-level strategies deployed for different devices. This helps to decide which devices should be protected at first.

5.3. Traffic Analysis in Environment Monitoring

Among the IoT application domains, the habitat and environment monitoring has received a growing interest as it is essential for studying and making efficient use of our environment. As the first step of the analysis, sensor networks are used to collect data from the environment. As sensor networks are usually deployed in an open field with little human interaction, they are prone to failures due to extreme climatic conditions or various malicious attacks.

5.3.1. Scenario Description

Wireless Sensor Networks (WSNs) have been widely used in IoT environment monitoring applications as the WSNs are well-suited for long-term environmental sensing for IoT applications. With the WSNs, environmental monitoring includes both indoor and outdoor applications [54]. One outdoor application is the habitat monitoring which requires a large number of low-cost sensor nodes and a gateway node (i.e., the sink) deployed in a given landscape. Sensor nodes are responsible for data acquisition while the gateway node connects to the remote servers via the Internet.

5.3.2. Network Setting

We consider a WSN with 1000 sensor nodes and one sink deployed in an open and unattended field shown in Figure 6. The network is a homogeneous network as each sensor node has the same application for sensing the temperature and humidity of the environment.

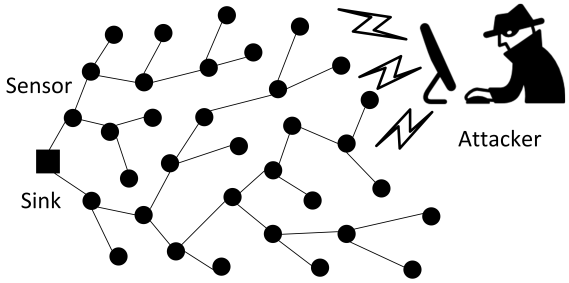

Figure 6: A wireless sensor network.

We assume sensor nodes and the sink are static after deployment. Sensor nodes self-organize and form a routing tree which is rooted at the sink [55]. Each sensor has a transmission range of r meters and uses bidirectional wireless communication. Communications between the sink and sensor nodes are single-hop or multi-hop. Sensor nodes periodically send packets to the sink (e.g., every 10 minutes). Data packets are encrypted by employing a pair-wise key scheme [56]. The sink is connected to the Internet and becomes the gateway between the sensor network and the Internet.

5.3.3. An Attacker Model

We assume the attacker's goal is to destroy the sink physically after finding its location. As the sink is the central point of failure, destroying it will make the whole network unavailable for sending data to the remote servers. The attacker model is based on [57] which describes the attacker capabilities as follows:

1. In the wireless communication, radio links are insecure. We assume an attacker can eavesdrop on radio transmissions by distributing a wireless monitoring device in the area of interest. The transmission range of the monitoring device is larger than the transmission range of a sensor node (e.g., $3r$ meters) but does not cover the entire network.
2. The attacker can physically move from one location to another location in the network but cannot monitor the entire network.
3. Each node routes packets along a fixed path to the sink using wireless communication. Thus the attacker can launch a rate-monitoring attack to deduce the location of the sink by monitoring the packet sending rate of nodes and moving to nodes with higher rates.
4. As the sink is in an open environment, the attacker can physically damage it once he discovers its location.

5.3.4. Data Processing

As all sensor nodes in the network are identical, we introduce one subnet for the whole network, denoted as s_{wsn}. According to the attacker capabilities in Section 5.3.3, we denote the vulnerability of a node described in capability

3 as v_{sn} and the vulnerability of a sink described in capability 4 as v_{sink}. For v_{sn}, sensor nodes are deployed in an open field, thus allowing easy access to them. However, an attacker needs to purchase and distribute a special device to monitor the packet sending rate. Thus we use high attack success probability and high attack cost. By using the vulnerability, the attacker might be able to discover the position of the base station. We assume a medium attack impact. For v_{sink}, as the sink is deployed in an open field, the attacker can easily damage it physically. So we use high attack success probability and medium attack cost respectively. Once the sink is damaged, the data gathered from sensor nodes cannot be delivered to remote servers. Thus we use high attack impact. Besides, for both v_{sn} and v_{sink}, we use a compromise rate of once per week as the sensor network might be deployed in remote areas and it is not easy for an attacker to get there. The estimated values for security metrics for the vulnerabilities are shown in Table 9.

Table 9: Metric values for each vulnerability.

Metric / Vulnerability	asp_v	ac_v	aim_v	cr_v
v_{sn}	0.9	8.0	4.0	0.006
v_{sink}	0.9	5.0	10.0	0.006

5.3.5. Security Model Generation and Visualization

The attacker is assumed to access one sensor node (e.g., sn_{999} deployed at the edge of the network). We use one subnet in the upper layer and compute the extended HARM. From this, Figure 7 shows the visualized attack path in the network. As each sensor has the same vulnerability, we show only one v_{sn} in the lower layer. By exploiting the vulnerabilities, the attacker is able to move along the nodes with a higher packet sending rate and discover the location of the sink.

5.3.6. Security Analysis and Model Updates

As each node (i.e., a sensor or the sink) has only one vulnerability which can be exploited by the attacker, metric values in the vulnerability level equal values in the node level. We only calculate the metric values in the network level, denoted as ASP_3, AC_3, AIM_3 and $MTTC_3$. We list the nodes in the attack path as ap=($t_{sn_{999}}$, $t_{sn_{499}}$, $t_{sn_{249}}$, $t_{sn_{124}}$, $t_{sn_{61}}$, $t_{sn_{30}}$, $t_{sn_{14}}$, t_{sn_6}, t_{sn_2}, t_{sink}).

Attack success probability: we calculate ASP_3 using Algorithm 1 in which the SHARPE output is 0.34868.

Attack cost: we calculate AC_3 by Equations (6), (7) and (8).

$$\begin{aligned} AC_3 &= ac_{ap} = ac_{t_{sn_{999}}} + ac_{t_{sn_{499}}} + ... + ac_{t_{sink}} \\ &= ac_{root_{sn_{999}}} + ac_{root_{sn_{499}}} + ... + ac_{root_{sink}} = 77.0 \end{aligned}$$

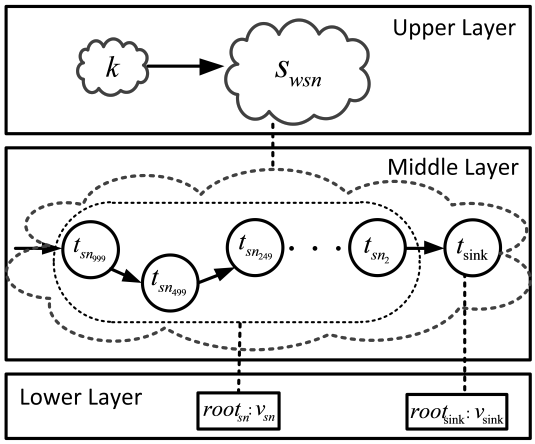

Figure 7: The attack path in the wireless sensor network.

Attack impact: we calculate AIM_3 by Equations (10), (11) and (12).

$$\begin{aligned} AIM_3 &= aim_{ap} = aim_{t_{sn_{999}}} + aim_{t_{sn_{499}}} + ... + aim_{t_{sink}} \\ &= aim_{root_{sn_{999}}} + aim_{root_{sn_{499}}} + ... + aim_{root_{sink}} = 46.0 \end{aligned}$$

Mean-time-to-compromise: we calculate $MTTC_3$ by Equations (13), (15) and (16).

$$\begin{aligned} MTTC_3 &= mttc_{ap} = mttc_{t_{sn_{999}}} + mttc_{t_{sn_{499}}} + ... + mttc_{t_{sink}} \\ &= 1/cr_{t_{sn_{999}}} + 1/cr_{t_{sn_{499}}} + ... + 1/cr_{t_{sink}} = 1666.66 \end{aligned}$$

In terms of the defense strategy, we can deploy the multi-parent routing (MPR) scheme for the sensor node proposed in [57], denoted as $Defense_{MPR}$. When forwarding a packet, the node randomly selects one of its parent nodes to forward the packet. Thus the attacker needs more time to guess which path to follow in order to reach the sink. We make the assumptions on the metric values of the sensor vulnerability after deploying the MPR scheme in Table 10. We decrease the attack success probability and compromise rate, and increase the attack cost as the method complicates the attack but does not eliminate the vulnerability. The method does not affect the attack impact as the impact measures the loss after the vulnerability is exploited.

Table 10: Metric values for the vulnerability v_{sn}.

Metric / Strategy	asp_v	ac_v	aim_v	cr_v
$Defense_{MPR}$	0.6	9.0	4.0	0.003

We use the framework to analyze whether the defense scheme is effective or not based on the network-level security metrics. Figure 8 shows the new attack paths in the network. After deploying the MPR scheme, the extended HARM captures multiple attack paths from a sensor node (i.e., the break-in point) to the sink. We compare values before and after the defense in Table 11.

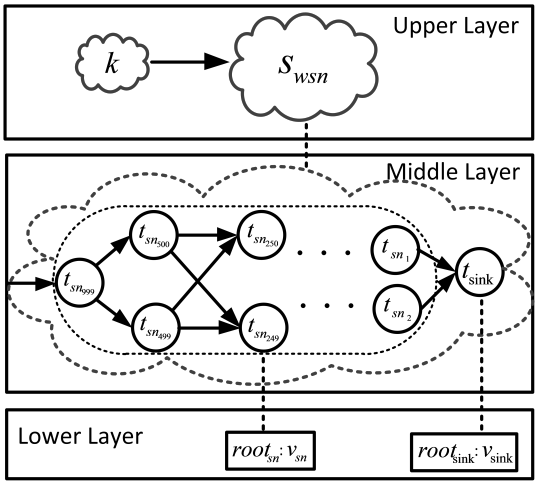

Figure 8: The attack paths in the wireless sensor network.

Table 11: Security analysis of the network.

Strategy \ Metric	ASP_3	AC_3	AIM_3	$MTTC_3$
No defense	0.35	77.0	46.0	1666.66
$Defense_{MPR}$	**0.27**	**86.0**	46.0	**3166.66**

After deploying the MPR scheme, ASP_3 decreases while $MTTC_3$ and AC_3 increase, which indicates the scheme is effective to lower the attack success probability, increase attack cost and extend the mean-time-to-compromise. AIM_3 does not change as the attack impact values do not change before and after the defense strategy. Thus, we can conclude the network-level defense strategy is effective against the traffic analysis attack.

5.3.7. Summary

Using the framework, one can assess the effectiveness of network-level defense strategies deployed for the network based on the security metrics.

6. Extensions and Limitations

In the previous work [6], we designed the framework and used a two-layer graphical security model (i.e., the HARM) to find potential attack scenarios in homogeneous networks. Based on the HARM, in this paper, we developed the extended HARM which is able to capture attack paths in heterogeneous IoT networks. Besides, as long as the node vulnerability information and network reachability information are provided, the attack paths can be captured by the framework. However, more analysis should be made in the following aspects.

Scenarios: a general structure of IoT systems usually consists of local networks, the Internet, back-end services in remote servers or Cloud platforms and remote users who can get access to devices in the local network and control them through the back-end services. In our scenario, we only consider a local network connecting to the Internet via the gateway. Thus other components of IoT systems should be included in order to provide a comprehensive scenario.

Attacker models: other attacker models can be considered. For example, Distributed Denial of Service (DDoS) attacks target a single system using a large number of zombie computers (infected by malwares). It is more disruptive for the IoT as IoT devices can easily be compromised due to limited security protections, and then controlled by attackers as zombie devices [58]. Moreover, such attacks have already been carried out by attackers in the real world. Thus analyzing DDoS attacks and finding protection strategies are essential to mitigate the impact of these attacks.

Defense strategies: we consider device-centric solutions (i.e., software patches) and only one network-level security solution in the analysis of different defense strategies. However, traditional solutions may not work well in securing the IoT as many IoT devices have constrained hardware and poor security mechanisms [38]. Moreover, forever-day vulnerabilities (e.g., vendors and suppliers no longer provide support for their products) are difficult to remove and unknown vulnerabilities (e.g., zero-day vulnerabilities [59]) are impossible to patch. Thus more network-level defense strategies should be considered to secure the IoT deployments (e.g., the addition of monitor devices, software defined networking technologies [60]).

Recently, software-defined networking (SDN) is foreseen as a key enabler for the IoT as SDN is able to manage large-scale networks, establish complex routing topologies and simplify user operations. In particular, it centralizes the network control and provides dynamic, flexible and automated reconfiguration of the networks [61, 62, 63]. There are already several papers about integrating SDN with the IoT. Some papers applied SDN in different parts of the IoT [64], for example, wireless sensor networks [65, 66, 67] and mobile networks [68, 69, 70, 71]; others proposed a software-defined IoT architecture [72, 73, 74]. In order to deal with non-patchable vulnerabilities (e.g., forever-day vulnerabilities), we will change the attack surface of the IoT network to increase the attack efforts of an attacker. With the support of the SDN functions, we will design proactive defence mechanisms that reconfigure the IoT topology. For example, if the network has only nodes with non-patchable vulnerabilities, we could maximize

the number of nodes with "harder-to-exploit" vulnerabilities along the paths to the potential target. In the reconfigured network, the attacker needs to put more effort on compromising the stepping nodes towards the target. We will consider various cases and develop different reconfiguration mechanisms. Then we will analyze how the security and performance of the software-defined IoT change when the solutions are deployed by using our framework. Besides, IoT networks in different domains may have different security requirements and budget limits. So finding the minimal protection strategies for different IoT networks is required in order to provide cost-effective solutions.

There are several limitations we aim to resolve in our future research.

Heterogeneity: to deal with the heterogeneity issue in IoT, we introduce subnets in the upper layer of the extended HARM and classify devices into subnets based on the communication protocols that the devices use. Other classification methods should also be combined and used in the framework as good classification can be beneficial to security modeling, security analysis and deployment of security mechanisms.

Mobility: when analyzing the scenarios, we assume the topology is static. However, one of the key features of the IoT is mobility. The movement of heterogeneous devices has a great influence over the security of the IoT as the attack surface changes with the dynamically changing network. In different scenarios, IoT devices have different movement patterns. Thus, a mobility model needs to be designed to capture node movement in the network (e.g., nodes join or move out) and notify changes to other models in the framework. In wireless communication networks, mobility models have been designed and extensively used in evaluations of network protocols. In particular, a mobility model is used to describe the movement pattern of mobile objects and to represent changes of their location, velocity and acceleration over time [75]. The mobility models have been classified based on their characteristics, which include random-based models [76, 77], models with temporal dependency or spatial dependency [78, 79], models with geographic constraints [80, 81], *etc.* We will examine current mobility models and modify them to capture the movement of IoT devices regarding different realistic scenarios. Our goal is to analyze the security of the IoT consisting of mobile nodes via the framework and investigate the impact of node movements on the IoT security.

Validation/verification: we evaluate the framework using the example networks. Simulations and experiments will be carried out to validate the framework. We will design an IoT testbed, which will be a smart sensing system consisting of various types of sensors (e.g., light sensor, sound sensor, ultrasonic range sensor, and temperature and humidity sensor) to monitor the environment. We will gather data from the experiments and use the data in the simulations. Future work also includes procedures to verify the correctness of the models.

7. Conclusions

Modeling security of the IoT is a difficult task as the IoT is characterized by a large number of heterogeneous and mobile devices and facing numerous threats. In this paper, we have presented a framework for graphically modeling and assessing security for the IoT, which encompasses five phases: 1) data processing, 2) security model generation, 3) security visualization, 4) security analysis, and 5) model updates. In the framework, we have developed an IoT Generator, a Security Model Generator and a Security Evaluator. The IoT Generator creates an IoT network based on the network reachability information and node vulnerability information; the Security Model Generator constructs the extended HARM based on the given IoT network; the Security Evaluator analyzes the security of the network using various security metrics. We have provided a formal definition of the framework. We have also introduced three example networks in three different scenarios, which are smart home, healthcare monitoring and environment sensing, and evaluated the framework via these scenarios. All possible attack paths have been computed by the extended HARM and values of chosen security metrics have been calculated in the security analysis phase. From the analysis results, the security decision maker is able to decide the most vulnerable part of the network, to assess the effectiveness of different defense mechanisms and to choose the most effective way to protect the network, thus mitigating the impact of potential attacks.

References

[1] R. Roman, J. Zhou, J. Lopez, On the features and challenges of security and privacy in distributed internet of things, Computer Networks 57 (10) (2013) 2266–2279. doi:http://dx.doi.org/10.1016/j.comnet.2012.12.018.

[2] R. Roman, P. Najera, J. Lopez, Securing the Internet of Things, Computer 44 (9) (2011) 51–58. doi:10.1109/MC.2011.291.

[3] S. Sicari, A. Rizzardi, L. Grieco, A. Coen-Porisini, Security, privacy and trust in Internet of Things: The road ahead, Computer Networks 76 (2015) 146–164. doi:http://dx.doi.org/10.1016/j.comnet.2014.11.008.

[4] J. Hong, D. Kim, HARMs: Hierarchical Attack Representation Models for Network Security Analysis, in: Proceedings of the 10th Australian Information Security Management Conference (AISM '12), 2012.

[5] R. A. Sahner, K. S. Trivedi, A. Puliafito, Performance and Reliability Analysis of Computer Systems: An Example-Based Approach Using the SHARPE Software Package, Kluwer Academic Publishers, 1996.

[6] M. Ge, D. S. Kim, A Framework for Modeling and Assessing Security of the Internet of Things, in: Proceedings of the 2015 IEEE 21st International Conference on Parallel and Distributed Systems (ICPADS '15), IEEE Computer Society, 2015, pp. 776–781. doi:10.1109/ICPADS.2015.102.

[7] O. Sheyner, J. Haines, S. Jha, R. Lippmann, J. M. Wing, Automated Generation and Analysis of Attack Graphs, in: Proceedings of the 2002 IEEE Symposium on Security and Privacy (SP '02), IEEE Computer Society, 2002, pp. 273–284.

[8] V. Saini, Q. Duan, V. Paruchuri, Threat Modeling using Attack Trees, Journal of Computer Science in Colleges 23 (4) (2008) 124–131.

[9] J. B. Hong, D. S. Kim, Towards scalable security analysis using multi-layered security models, Journal of Network and Computer Applications 75 (2016) 156–168. doi:http://dx.doi.org/10.1016/j.jnca.2016.08.024.

[10] S. Radomirovic, Towards a Model for Security and Privacy in the Internet of Things, in: Proceedings of the 1st International Workshop Security of the Internet of Things (SecIoT '10), 2010.

[11] J. C. Yang, B. X. Fang, Security model and key technologies for the Internet of things, Journal of China Universities of Posts and Telecommunications 18, Supplement 2 (2011) 109–112. doi:http://dx.doi.org/10.1016/S1005-8885(10)60159-8.

[12] T. Stepanova, D. Zegzhda, Applying Large-scale Adaptive Graphs to Modeling Internet of Things Security, in: Proceedings of the 7th International Conference on Security of Information and Networks (SIN '14), ACM, 2014, pp. 479–482. doi:10.1145/2659651.2659696.

[13] A. Atamli, A. Martin, Threat-Based Security Analysis for the Internet of Things, in: Proceedings of the 2014 International Workshop on Secure Internet of Things (SIoT '14), IEEE Computer Society, 2014, pp. 35–43. doi:10.1109/SIoT.2014.10.

[14] X. Huang, P. Craig, H. Lin, Z. Yan, SecIoT: a security framework for the Internet of Things, Security and Communication Networks 9 (16) (2015) 3083–3094. doi:10.1002/sec.1259.

[15] Q. M. Ashraf, M. H. Habaebi, Autonomic Schemes for Threat Mitigation in Internet of Things, J. Netw. Comput. Appl. 49 (C) (2015) 112–127. doi:10.1016/j.jnca.2014.11.011.

[16] P. Y. Chen, S. M. Cheng, K. C. Chen, Information Fusion to Defend Intentional Attack in Internet of Things, IEEE Internet of Things Journal 1 (4) (2014) 337–348. doi:10.1109/JIOT.2014.2337018.

[17] M. Hamdi, H. Abie, Game-Based Adaptive Security in the Internet of Things for eHealth, in: Proceedings of the 2014 IEEE International Conference on Communications (ICC '14), IEEE, 2014, pp. 920–925. doi:10.1109/ICC.2014.6883437.

[18] G. Rontidis, E. Panaousis, A. Laszka, T. Dagiuklas, P. Malacaria, T. Alpcan, A Game-Theoretic Approach for Minimizing Security Risks in the Internet-of-Things, in: Proceedings of the 2015 IEEE International Conference on Communication Workshop (ICCW '15), IEEE, 2015, pp. 2639–2644. doi:10.1109/ICCW.2015.7247577.

[19] R. M. Savola, H. Abie, M. Sihvonen, Towards Metrics-driven Adaptive Security Management in e-Health IoT Applications, in: Proceedings of the 7th International Conference on Body Area Networks (BodyNets '12), IEEE, 2012, pp. 276–281.

[20] H. Abie, I. Balasingham, Risk-based Adaptive Security for Smart IoT in eHealth, in: Proceedings of the 7th International Conference on Body Area Networks (BodyNets '12), IEEE, 2012, pp. 269–275.

[21] Y. Berhanu, H. Abie, M. Hamdi, A Testbed for Adaptive Security for IoT in eHealth, in: Proceedings of the International Workshop on Adaptive Security (ASPI '13), ACM, 2013, pp. 1–8. doi:10.1145/2523501.2523506.

[22] A. B. Torjusen, H. Abie, E. Paintsil, D. Trcek, A. Skomedal, Towards Run-Time Verification of Adaptive Security for IoT in eHealth, in: Proceedings of the 2014 European Conference on Software Architecture Workshops (ECSAW '14), ACM, 2014, pp. 1–8. doi:10.1145/2642803.2642807.

[23] K. Habib, W. Leister, Threats Identification for the Smart Internet of Things in eHealth and Adaptive Security Countermeasures, in: Proceedings of the 2015 7th International Conference on New Technologies, Mobility and Security (NTMS '15), IEEE, 2015, pp. 1–5. doi:10.1109/NTMS.2015.7266525.

[24] S. Mauw, M. Oostdijk, Foundations of Attack Trees, in: Proceedings of the 8th International Conference on Information Security and Cryptology (ICISC '05), Springer-Verlag, 2005, pp. 186–198. doi:10.1007/11734727_17.

[25] C.-W. Ten, C.-C. Liu, M. Govindarasu, Vulnerability Assessment of Cybersecurity for SCADA Systems Using Attack Trees, in: Proceedings of the 2007 IEEE Power Engineering Society General Meeting, IEEE, 2007, pp. 1–8. doi:10.1109/PES.2007.385876.

[26] A. Roy, D. S. Kim, K. S. Trivedi, Attack countermeasure trees (ACT): towards unifying the constructs of attack and defense trees, Security and Communication Networks 5 (8) (2012) 929–943. doi:10.1002/sec.299.

[27] S. Jha, O. Sheyner, J. Wing, Two Formal Analys s of Attack Graphs, in: Proceedings of the 15th IEEE Workshop on Computer Security Foundations (CSFW '02), IEEE Computer Society, 2002, pp. 49–63.

[28] X. Ou, S. Govindavajhala, A. W. Appel, MulVAL: A Logic-based Network Security Analyzer, in: Proceedings of the 14th Conference on USENIX Security Symposium (SSYM '05), USENIX Association, 2005, pp. 8–8.

[29] K. Ingols, M. Chu, R. Lippmann, S. Webster, S. Boyer, Modeling Modern Network Attacks and Countermeasures Using Attack Graphs, in: Proceedings of the 25th Annual Computer Security Applications Conference (ACSAC '09), IEEE Computer Society, 2009, pp. 117–126. doi:10.1109/ACSAC.2009.21.

[30] K. Ingols, R. Lippmann, K. Piwowarski, Practical Attack Graph Generation for Network Defense, in: Proceedings of the 22nd Annual Computer Security Applications Conference (ACSAC '06), IEEE Computer Society, 2006, pp. 121–130. doi:10.1109/ACSAC.2006.39.

[31] M. Albanese, S. Jajodia, S. Noel, Time-efficient and cost-effective network hardening using attack graphs, in: Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks (DSN '12), IEEE, 2012, pp. 1–12. doi:10.1109/DSN.2012.6263942.

[32] F. Jia, J. B. Hong, D. S. Kim, Towards Automated Generation and Visualization of Hierarchical Attack Representation Models, in: Proceedings of the 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), IEEE, 2015, pp. 1689–1696. doi:10.1109/CIT/IUCC/DASC/PICOM.2015.255.

[33] J. B. Hong, D. S. Kim, Assessing the Effectiveness of Moving Target Defenses using Security Models, IEEE Transactions on Dependable and Secure Computing 13 (2) (2015) 163–177. doi:10.1109/TDSC.2015.2443790.

[34] R. M. Blank, P. D. Gallagher, NIST Special Publication 800-30 Revision 1 Guide for Conducting Risk Assessments, Tech. rep., National Institute of Standards and Technology (2012).

[35] D. Leversage, E. James, Estimating a System's Mean Time-to-Compromise, IEEE Security & Privacy 6 (1) (2008) 52–60. doi:10.1109/MSP.2008.9.

[36] L. Gallon, J. Bascou, Using CVSS in Attack Graphs, in: Proceedings of the 6th International Conference on Availability, Reliability and Security (ARES '11), IEEE, 2011, pp. 59–66. doi:10.1109/ARES.2011.18.

[37] J. Gubbi, R. Buyya, S. Marusic, M. Palaniswami, Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions, Future Generation Computer Systems 29 (7) (2013) 1645–1660. doi:10.1016/j.future.2013.01.010.

[38] T. Yu, V. Sekar, S. Seshan, Y. Agarwal, C. Xu, Handling a Trillion (Unfixable) Flaws on a Billion Devices: Rethinking Network Security for the Internet-of-Things, in: Proceedings of the 14th ACM Workshop on Hot Topics in Networks (HotNets-XIV '15), ACM, 2015, pp. 1–7. doi:10.1145/2834050.2834095.

[39] A. Jacobsson, M. Boldt, B. Carlsson, A risk analysis of a smart home automation system, Future Generation Computer Systems 56 (2016) 719–733. doi:http://dx.doi.org/10.1016/j.future.2015.09.003.

[40] P. Kinney, ZigBee Technology: Wireless Control that Simply Works, Tech. rep., ZigBee Alliance (2003).

[41] K. Gill, S. H. Yang, F. Yao, X. Lu, A ZigBee-Based Home Automation System, IEEE Transactions on Consumer Electronics 55 (2) (2009) 422–430. doi:10.1109/TCE.2009.5174403.

[42] C. Y. Chang, C. H. Kuo, J. C. Chen, T. C. Wang, Design and Implementation of an IoT Access Point for Smart Home, Applied Sciences 5 (4) (2015) 1882–1903. doi:10.3390/app5041882.

[43] S. D. T. Kelly, N. K. Suryadevara, S. C. Mukhopadhyay, Towards the Implementation of IoT for Environmental Condition Monitoring in Homes, IEEE Sensors Journal 13 (10) (2013) 3846–3853. doi:10.1109/JSEN.2013.2263379.

[44] L. Coppolino, V. D'Alessandro, S. D'Antonio, L. Lev, L. Romano, My Smart Home is Under Attack, in: Proceedings of the 2015 IEEE 18th International Conference on Computational Science and Engineering (CSE '15), IEEE Computer Society, 2015, pp. 145–151. doi:10.1109/CSE.2015.28.

[45] Y. Bachy, F. Basse, V. Nicomette, E. Alata, M. Kaaniche, J. C. Courrege, P. Lukjanenko, Smart-TV Security Analysis: Practical Experiments, in: Proceedings of the 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN '15), IEEE, 2015, pp. 497–504. doi:10.1109/DSN.2015.41.

[46] B. Michele, A. Karpow, Watch and be watched: Compromising all Smart TV generations, in: Proceedings of the 2014 IEEE 11th Consumer Communications and Networking Conference (CCNC '14), IEEE, 2014, pp. 351–356. doi:10.1109/CCNC.2014.6866594.

[47] R. Unuchek, Obad.a Trojan Now Being Distributed via Mobile Botnets, https://securelist.com/blog/mobile/57453/obad-a-trojan-now-being-distributed-via-mobile-botnets/, Last accessed: 2016-09-14.

[48] J. Ko, C. Lu, M. B. Srivastava, J. A. Stankovic, A. Terzis, M. Welsh, Wireless Sensor Networks for Healthcare, Proceedings of the IEEE 98 (11) (2010) 1947–1960. doi:10.1109/JPROC.2010.2065210.

[49] IEEE Std 802.15.4-2003: Part 15.4: Wireless Medium Access Control and Physical Layer Specifications for Low-Rate Wireless Personal Area Networks, Tech. rep., IEEE (2003).

[50] B. Latré, B. Braem, I. Moerman, C. Blondia, P. Demeester, A Survey on Wireless Body Area Networks, Wireless Networks 17 (1) (2011) 1–18. doi:10.1007/s11276-010-0252-4.

[51] B. Latre, B. Braem, I. Moerman, C. Blondia, E. Reusens, W. Joseph, P. Demeester, A Low-delay Protocol for Multihop Wireless Body Area Networks, in: Proceedings of the 4th Annual International Conference on Mobile and Ubiquitous Systems: Networking Services (MobiQuitous '07), IEEE, 2007, pp. 1–8. doi:10.1109/MOBIQ.2007.4451060.

[52] T. Goodspeed, MSP430 Buffer Overflow Exploit for Wireless Sensor Nodes, http://travisgoodspeed.blogspot.co.nz/2007/08/machine-code-injection-for-wireless.html, Last accessed: 2016-09-14.

[53] D. Singelée, B. Latré, B. Braem, M. Peeters, M. Soete, P. Cleyn, B. Preneel, I. Moerman, C. Blondia, A Secure Cross-Layer Protocol for Multi-hop Wireless Body Area Networks, in: Proceedings of the 7th International Conference on Ad-hoc, Mobile and Wireless Networks (ADHOC-NOW '08), Springer Berlin Heidelberg, 2008, pp. 94–107. doi:10.1007/978-3-540-85209-4_8.

[54] M. T. Lazarescu, Design of a WSN Platform for Long-Term Environmental Monitoring for IoT Applications, IEEE Journal on Emerging and Selected Topics in Circuits and Systems 3 (1) (2013) 45–54. doi:10.1109/JETCAS.2013.2243032.

[55] A. Woo, T. Tong, D. Culler, Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks, in: Proceedings of the 1st International Conference on Embedded Networked Sensor Systems (SenSys '03), ACM, 2003, pp. 14–27. doi:10.1145/958491.958494.

[56] L. Eschenauer, V. D. Gligor, A Key-Management Scheme for Distributed Sensor Networks, in: Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS '02), ACM, 2002, pp. 41–47. doi:10.1145/586110.586117.

[57] J. Deng, R. Han, S. Mishra, Countermeasures Against Traffic Analysis Attacks in Wireless Sensor Networks, in: Proceedings of the 1st International Conference on Security and Privacy for Emerging Areas in Communications Networks (SecureComm '05), IEEE, 2005, pp. 113–126. doi:10.1109/SECURECOMM.2005.16.

[58] C. Zhang, R. Green, Communication Security in Internet of Thing: Preventive Measure and Avoid DDoS Attack over IoT Network, in: Proceedings of the 18th Symposium on Communications & Networking (CNS '15), Society for Computer Simulation International, 2015, pp. 8–15.

[59] L. Bilge, T. Dumitras, Before We Knew It: An Empirical Study of Zero-day Attacks in the Real World, in: Proceedings of the 2012 ACM Conference on Computer and Communications Security (CCS '12), ACM, 2012, pp. 833–844. doi:10.1145/2382196.2382284.

[60] V. Sivaraman, H. Gharakheili, A. Vishwanath, R. Boreli, O. Mehani, Network-Level Security and Privacy Control for Smart-Home IoT Devices, in: Proceedings of the 2015 IEEE 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob '15), IEEE, 2015, pp. 163–167. doi:10.1109/WiMOB.2015.7347956.

[61] D. Kreutz, F. M. V. Ramos, P. E. Verssimo, C. E. Rothenberg, S. Azodolmolky, S. Uhlig, Software-Defined Networking: A Comprehensive Survey, Proceedings of the IEEE 103 (1) (2015) 14–76. doi:10.1109/JPROC.2014.2371999.

[62] B. A. A. Nunes, M. Mendonca, X. N. Nguyen, K. Obraczka, T. Turletti, A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks, IEEE Communications Surveys Tutorials 16 (3) (2014) 1617–1634. doi:10.1109/SURV.2014.012214.00180.

[63] F. Hu, Q. Hao, K. Bao, A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation, IEEE Communications Surveys Tutorials 16 (4) (2014) 2181–2206. doi:10.1109/COMST.2014.2326417.

[64] A. El-Mougy, M. Ibnkahla, L. Hegazy, Software-Defined Wireless Network Architectures for the Internet-of-Things, in: Proceedings of the 2015 40th Annual IEEE Conference on Local Computer networks (LCN '15), IEEE Computer Society, 2015, pp. 804–811. doi:10.1109/LCNW.2015.7365931.

[65] T. Miyazaki, S. Yamaguchi, K. Kobayashi, J. Kitamichi, S. Guo, T. Tsukahara, T. Hayashi, A Software Defined Wireless Sensor Network, in: Proceedings of the 2014 International Conference on Computing, Networking and Communications (ICNC '14), IEEE, 2014, pp. 847–852. doi:10.1109/ICCNC.2014.6785448.

[66] A. D. Gante, M. Aslan, A. Matrawy, Smart Wireless Sensor Network Management Based on Software-Defined Networking, in: Proceedings of the 2014 27th Biennial Symposium on Communications (QBSC '14), IEEE, 2014, pp. 71–75. doi:10.1109/QBSC.2014.6841187.

[67] T. Luo, H. P. Tan, T. Quek, Sensor OpenFlow: Enabling Software-Defined Wireless Sensor Networks, IEEE Communications Letters 16 (11) (2012) 1896–1899.

[68] K. Pentikousis, Y. Wang, W. Hu, MobileFlow: Toward Software-Defined Mobile Networks, IEEE Communications Magazine 51 (7) (2013) 44–53. doi:10.1109/MCOM.2013.6553677.

[69] T. Lei, Z. Lu, X. Wen, X. Zhao, L. Wang, SWAN: An SDN Based Campus WLAN framework, in: Proceedings of the 2014 4th International Conference on Wireless Communications, Vehicular Technology, Information Theory and Aerospace Electronic Systems (VITAE '14), IEEE, 2014, pp. 1–5. doi:10.1109/VITAE.2014.6934489.

[70] M. Amani, T. Mahmoodi, M. Tatipamula, H. Aghvami, Programmable Policies for Data Offloading in LTE Network, in: Proceedings of the 2014 IEEE International Conference on Communications (ICC '14), IEEE, 2014, pp. 3154–3159. doi:10.1109/ICC.2014.6883806.

[71] C. J. Bernardos, A. de la Oliva, P. Serrano, A. Banchs, L. M. Contreras, H. Jin, J. C. Zuniga, An Architecture for Software Defined Wireless Networking, IEEE Wireless Communications 21 (3) (2014) 52–61. doi:10.1109/MWC.2014.6845049.

[72] D. Wu, D. I. Arkhipov, E. Asmare, Z. Qin, J. A. McCann, UbiFlow: Mobility Management in Urban-scale Software Defined IoT, in: Proceedings of the 2015 IEEE Conference on Computer Communications (INFOCOM '15), IEEE, 2015, pp. 208–216. doi:10.1109/INFOCOM.2015.7218384.

[73] A. Hakiri, P. Berthou, A. Gokhale, S. Abdellatif, Publish/subscribe-enabled Software Defined Networking for Efficient and Scalable IoT Communications, IEEE Communications Magazine 53 (9) (2015) 48–54. doi:10.1109/MCOM.2015.7263372.

[74] J. Liu, Y. Li, M. Chen, W. Dong, D. Jin, Software-Defined Internet of Things for Smart Urban Sensing, IEEE Communications Magazine 53 (9) (2015) 55–63. doi:10.1109/MCOM.2015.7263373.

[75] F. Bai, A. Helmy, A Survey of Mobility Modeling and Analysis in Wireless Adhoc Networks, in: Wireless Ad Hoc and Sensor Networks, Springer, 2006, Ch. 1.

[76] T. Camp, J. Boleng, V. Davies, A Survey of Mobility Models for Ad Hoc Network Research, Wireless Communications and Mobile Computing 2 (5) (2002) 483–502. doi:10.1002/wcm.72.

[77] C. Bettstetter, H. Hartenstein, X. Pérez-Costa, Stochastic Properties of the Random Waypoint Mobility Model, Wireless Networks 10 (5) (2004) 555–567. doi:10.1023/B:WINE.0000036458.88990.e5.

[78] C. Bettstetter, Smooth is Better Than Sharp: A Random Mobility Model for Simulation of Wireless Networks, in: Proceedings of the 4th ACM

International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM '01), ACM, 2001, pp. 19–27. doi:10.1145/381591.381600.

[79] X. Hong, M. Gerla, G. Pei, C.-C. Chiang, A Group Mobility Model for Ad Hoc Wireless Networks, in: Proceedings of the 2nd ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM '99), ACM, 1999, pp. 53–60. doi:10.1145/313237.313248.

[80] J. Tian, J. Hahner, C. Becker, I. Stepanov, K. Rothermel, Graph-Based Mobility Model for Mobile Ad Hoc Network Simulation, in: Proceedings of 35th Annual Simulation Symposium, IEEE, 2002, pp. 337–344. doi:10.1109/SIMSYM.2002.1000171.

[81] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, M. Degermark, Scenario-based Performance Analysis of Routing Protocols for Mobile Ad-hoc Networks, in: Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99), ACM, 1999, pp. 195–206. doi:10.1145/313451.313535.