technologies

MDPI

Article

An Effective Security Requirements Engineering Framework for Cyber-Physical Systems

Shafiq ur Rehman * and Volker Gruhn

Institute of Software Technology, University of Duisburg-Essen, 45127 Essen, Germany; volker.gruhn@uni-due.de

* Correspondence: shafiq.rehman@uni-due.de; Tel.: +49-151-710-86963

Received: 24 May 2018; Accepted: 5 July 2018; Published: 12 July 2018

Abstract: Context and motivation: Cyber-Physical Systems (CPSs) are gaining priority over other systems. The heterogeneity of these systems increases the importance of security. Both the developer and the requirement analyst must consider details of not only the software, but also the hardware perspective, including sensor and network security. Several models for secure software engineering processes have been proposed, but they are limited to software; therefore, to support the processes of security requirements, we need a security requirements framework for CPSs. Question/Problem: Do existing security requirements frameworks fulfil the needs of CPS security requirements? The answer is no; existing security requirements frameworks fail to accommodate security concerns outside of software boundaries. Little or even no attention has been given to sensor, hardware, network, and third party elements during security requirements engineering in different existing frameworks. Principal Ideas/results: We have proposed, applied, and assessed an incremental security requirements evolution approach, which configures the heterogeneous nature of components and their threats in order to generate a secure system. Contribution: The most significant contribution of this paper is to propose a security requirements engineering framework for CPSs that overcomes the issue of security requirements elicitation for heterogeneous CPS components. The proposed framework supports the elicitation of security requirements while considering sensor, receiver protocol, network channel issues, along with software aspects. Furthermore, the proposed CPS framework has been evaluated through a case study, and the results are shown in this paper. The results would provide great support in this research direction.

Keywords: security requirements; security requirements engineering; framework; security goal; threat; cyber-physical systems

1. Introduction

We are living in the era of digitization where software, system hardware, and sensors are working together over networks. This combination describes the concept of Cyber-Physical Systems (CPS) [1]. In this situation, the urge to maintain security is of prime importance. Secure system development depends on an extensive focus on the process of requirements engineering towards security. Software engineering gets its developmental supports from tools and techniques, and models that guide them to manage quality development support [2,3]. These techniques provide information on how services are provided. However, when developing a secure system, one has to consider the vulnerabilities of the system as well.

To develop a system with a security focus, several security requirement frameworks have been proposed. Among them, some of the famous ones are SQUARE, SREP, Secure Tropos, CLASP, CORAS, and UMLsec [4–8]. The benefits of these models are limited to the software perspective, and at some point, to supporting the computer hardware. Unfortunately, none focuses on the new trend

that is cyber-physical systems. CPSs get their support from the sensor network, along with other, traditional software. This interaction involves a different form of data processing from and to the outer world [9,10]. Moreover, CPSs require a dedicated communicational channel for secure interaction. The diversity of cyber-physical systems forces the developer to reflect upon details of the security aspect of sensors, receivers, data processors, and communicators, as well as the general software security aspect.

In this paper, we have extended the framework proposed in our previous paper [11]. Here, we are providing details of the implementation of our proposed security requirements engineering framework for CPSs, its activity, and supporting techniques. The proposed framework aims to serve as a complete guide for a number of activities to analyze and identify threats and to determine security requirements of CPSs by taking different aspects of CPS into account. The novelty of such implementation at this scale has not been significantly reported in literature. The findings of this research will be of great benefit to practitioners and researchers who play an important role in the development of Security Requirements Engineering (SRE) for CPSs. The increased demand for security in organizations justifies the need of a security requirements engineering framework. Therefore, organizations that apply the proposed framework derived from the research results can train their requirement analysts and software developers, and this research will help them to explore security requirements in the early phases of software development.

1.1. Why Security Requirements Engineering for CPS

Security requirements engineering is an essential aspect of cyber-physical systems, but there is a lack of methodology defined to develop a secure software system. Though many methodologies and frameworks have been proposed for software, there is still a need to improve them [12]. Many researchers address the requirements engineering best practices and highlight the importance of system functionality, but a small amount of attention has been given to what the system should not do [13].

CPSs are in the initial stages of development, and therefore, face many challenges. Security is one of the challenges of CPSs. Different studies show that cyber threats have increased in the CPS environment, and there is a need to do more research to systematically handle security requirements [14–16]. Recently, many incidents of CPS attacks have been reported in the literature. In 2010, Stuxnet was the first cyber-attack on a CPS. It targeted a Siemens control system 'Supervisory Control and Data Acquisition (SCADA)' through malware to control and destroy Iran's nuclear program [17]. As a consequence, more than 50% of the Iranian nuclear infrastructure was attacked. Certainly, this incident creates an alarm for cyber threats [17]. In 2000, an Australian man was found guilty when he attacked the Maroochy waste management systems and released one million liters of impure sewage into rivers and local parks [18]. In 2006, a hacker infiltrated a US water filtration plant with malware that changed the levels of chemicals being used to treat tap water, and thousands of homes were affected in Illinois, USA [19]. Other famous cyber-attacks are Duqu and Flame, which were used to gain unauthorized access.

It is estimated that the software development budget to fix security flaws is almost 75% of the total cost after handover of the product to the customer. This is an enormous amount of spending that builds untrustworthiness amongst customers [20]. Figure 1 states the number of yearly spending on Information Technology (IT) security, which shows an escalation of approximately 8% annually [21–23].

Software security engineering proposes many tools, techniques, methods, and best practices to develop a secure system [24–26]. There is a lack of understanding of software security that should be clarified and managed in the early phase of the Software Development Life Cycle (SDLC) [27,28]. Therefore, researchers have not been successful in developing a secure software system when applying software engineering best practices [12].

The significance of addressing security is widely accepted from the very beginning of system development [29]. Generally, the security of software is not considered at the very beginning of a SDLC; it is only incorporated in the later stages of software development [30]. As a consequence, there are increased risks of security threats that are introduced in various stages of software development [31,32].

Therefore, integrating security requirements right at the beginning not only ensures secure software, but also saves precious time and reduces the effort of reworking by software development team. Hence, it is evident that to support the process of security requirements, we need a security requirements framework for CPSs.

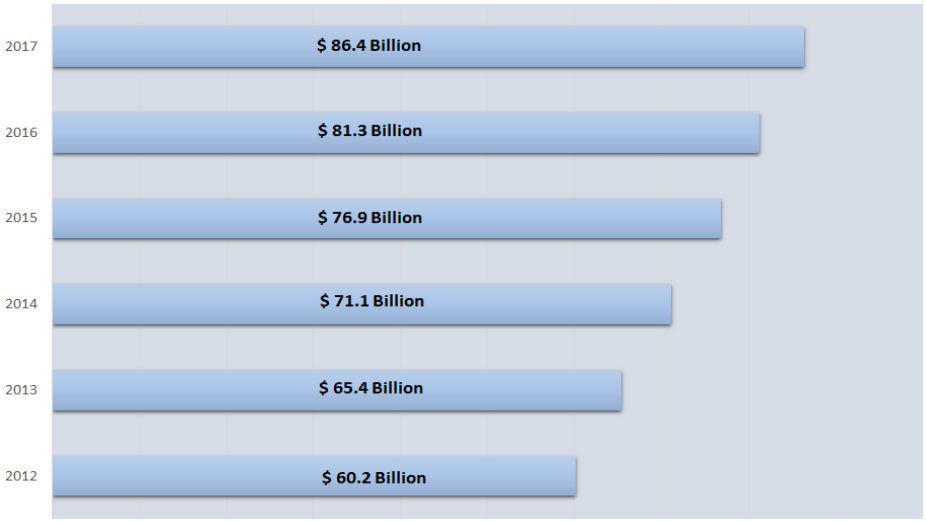

Figure 1. Worldwide spending on information security.

1.2. Security Issues around Sensor Networks

Since many cyber-physical systems depend on sensor networks, their security is an important factor to consider, as a malicious attack could harm or damage the physical part of the system. Figure 2 shows the security issues on the physical, network, and application layers. The application areas of sensor networks are very wide. They range from monitoring machines in production to military applications. The data are processed in networks, which makes their security critical. This is also due to the fact that sensor networks have new security requirements which are not matched by the security techniques of traditional networks [33]. One reason for this is that the sensors are partly located in open, accessible areas. This make them more vulnerable, and they become potential attacks. This is an important aspect of security of sensor networks that should be considered for every component of network [34]. If this is not done, unprotected components are vulnerable to attacks.

Confidentiality is also a subject of major importance within sensor networks. Networks could be used, in the worst case, to spy on individuals [34]. An example would be the long-term monitoring of persons or vehicles on a routine basis. Another aspect that affects the security of sensor networks are attacks on these networks communication between physical environment and gateway to controller/server, as shown in Figure 2. In the simplest form, the attacker sends a high-energy signal to the sensor to prevent communication within the physical layer to the network layer. This can lead to serious consequences, especially in the case of security-critical CPS. Military applications are also threatened by such attacks. A possibility to combat these attacks lies in the nature of the networks themselves. If a part of the network has been compromised, this part can be demarcated and the communication can be routed around it [33,35]. Therefore, it is of utmost importance to address the security of all components (i.e., physical layer, network layer and application layer) of CPSs.

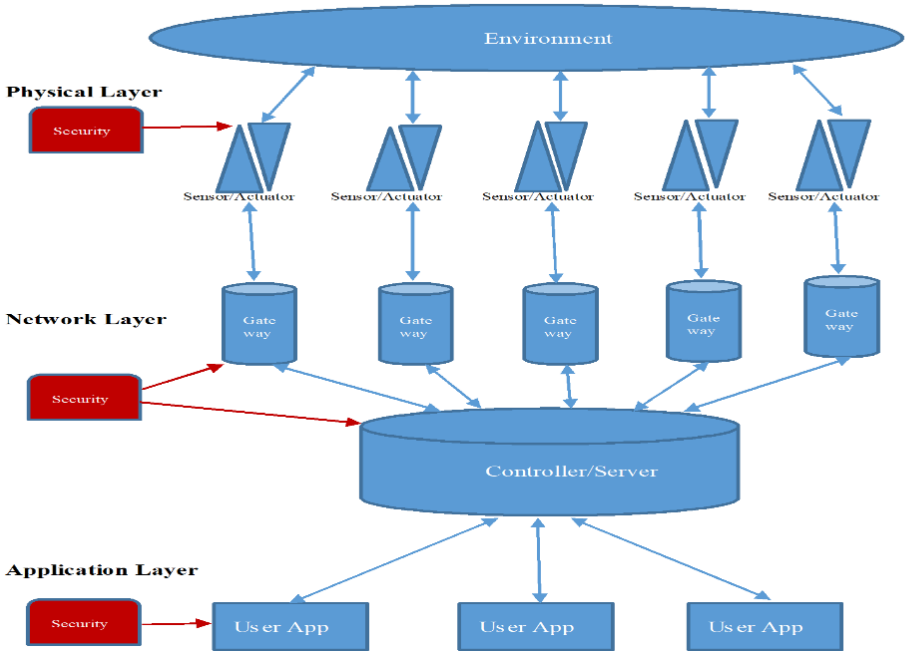

Figure 2. Security issues.

2. Related Work

Security is considered to be the core aspect for the strength of the software. Several types of research have proposed different methodologies, models, and frameworks to support security requirements elicitation and maintenance. The most famous security requirements frameworks are SQUARE, SREP CORAS, MS SDL, UMLsec, and Secure Tropos. Therefore, in this paper, we are providing a review of only major security requirements frameworks that have significant contributions in the literature. A detailed literature review has been conducted to summarize the evidence, and the main contributions of each paper have been examined.

System Quality Requirements Engineering (SQUARE) is a security requirements engineering framework and comprehensive methodology to determine security requirements for software [36]. The framework provides a means to elicit, categorize, and prioritize security requirements for a system. The objective of the framework is to build security concepts in the early stages of the software development lifecycle, and integrate effective security requirements engineering into system development processes. The framework consists of nine activities, and has been implemented in the development of software to elicit security requirements [37].

Security Requirements Engineering Process (SREP) is an asset and risk-driven framework [38]. Authors of this framework combined Common Criteria (CC) with software development life cycle and the requirement repository. They have proposed 9 activities which can be processed in an iterative or an incremental manner. They have suggested that the framework is enhanced with the support of a repository for reusable security requirements. These requirements provided information regarding threats and assets. In another paper [39], the authors have applied the proposed framework on Information System (IS) and suggested computer-based tool development for the support of frameworks.

The authors in the paper [40] have proposed a combination of System Security Engineering Capability Maturity Model (SSE-CMM) with CC. The authors suggested that the combination of two standards not only to provide a product security evaluation, but also a process evolution. The paper [41] proposes a security risk analysis model for securing information systems. They designed Bayesian Network (BN)-based model that presents factors to support of risk assessment and their cause. The authors have further suggested that the risk can be propagated through different mediums

of information systems. To handle this perspective by considering probabilistic-based vulnerability propagation refreshes the previous information at each iteration.

CORAS [42] is another security model that considers asset, risk, and vulnerability. The process works the fusion of model-driven method. The authors have focused on the user and the analyst requirements elicitation approach. CORAS provided support for security assessments of the system with reusable support that is an integrated part of the risk assessment. CORAS can also be applied on a web application for security analysis [43].

The Comprehensive, Lightweight Application Security Process (CLASP) is a framework that connects roles, resources, and their associations [44]. On the basis of relational information, the authors have provided a list of security services. The authors suggest that for each service, there is a need to establish Specific Measurable Acceptable Realistic and Time-bound (SMART) requirements. To illustrate their approach, they have performed a walkthrough on a service-based application for contact storage. However, it lacks the results comparison of cross-project analyses. On the other hand, the paper [45] provided a comparison of Microsoft Secured Development Life Cycle (MS SDL) and CLASP. They have analyzed the activities of both processes, specifically from a security perspective. They have identified that both models have an advantage which is specific for different levels. Even among the two models, neither provided evidence to support communication and physical aspect of security issues.

Secure Tropos is a goal-oriented methodology for eliciting security requirements [46,47]. The modelling concept of secure Tropos focuses on security constraints, security dependencies, secure goals, secure tasks, and secure resources. An actor represents an entity with strategic interests and intentions. An informational entity is referred to as a secure resource. Actors can be linked by dependencies, indicating that one actor depends on the other to attain some goal, execute some tasks, or deliver a resource. International standard common critical [48] provides support for evaluating security requirements for information systems.

The Unified Modelling Langauge for security (UMLsec) is a UML-based modelling method to integrate security features and analysis in the design phase [48]. UMLsec is used to evaluate UML specifications for security threats and vulnerabilities. The extension of UML is provided through the standard UML extension mechanism by the addition of few elements. Security requirements that evolve from modelling the systems design with UMLsec cover the main security goals, specifically confidentiality, integrity, and availability. An approach based on goal perceptive combines UMLsec and secure troops for delivering secure system [8,49]. This approach considers both technical and social aspects. They perceive the support from UMLsec for model translation to ensure the proper application of security requirements at a design level. The authors in the paper [50] have proposed a goal-based security model to handle software product line issues. It does not provide the support for physical security aspects.

These varieties of attempts make difficult the choice for security analysts to select the appropriate security requirements engineering methodology according to their needs and expectations. All of these methodologies provide support for software security, but all of them work in isolation. A model that supports a goal perspective does not consider the activity of software development. Similarly, a model based on the information system does not consider other systems. Furthermore, all of these models are supporting software services or component and their relationships. However, very little attention has been given to hardware, communicators, receivers, or system security issues. Therefore, it is very important to explore security methodologies, especially in the domain of CPS.

3. Risk Assessment, Security Goals and Threats for CPS

Risk assessment does not only identify the risks associated with an individual requirement, but also identifies the risks associated with the interdependencies of various requirements of system development. Lack of risk assessment in implementing security requirements of CPS can lead to unexpected and undesired behavior of the system, as the developers might miss critical requirements

resulting from the interdependency of these security requirements. Therefore, we also aim to posit a roadmap on the risk assessment of security requirements in the CPS framework. The risks in CPSs are based on the requirements of the software and other components of the CPS. Furthermore, we have analyzed and identified the major security goals and threats of a cyber-physical system. This analysis is based on few matrices that were developed during the implementation of a smart car parking system. The identified security goals and threats of CPS are shown in Figure 3.

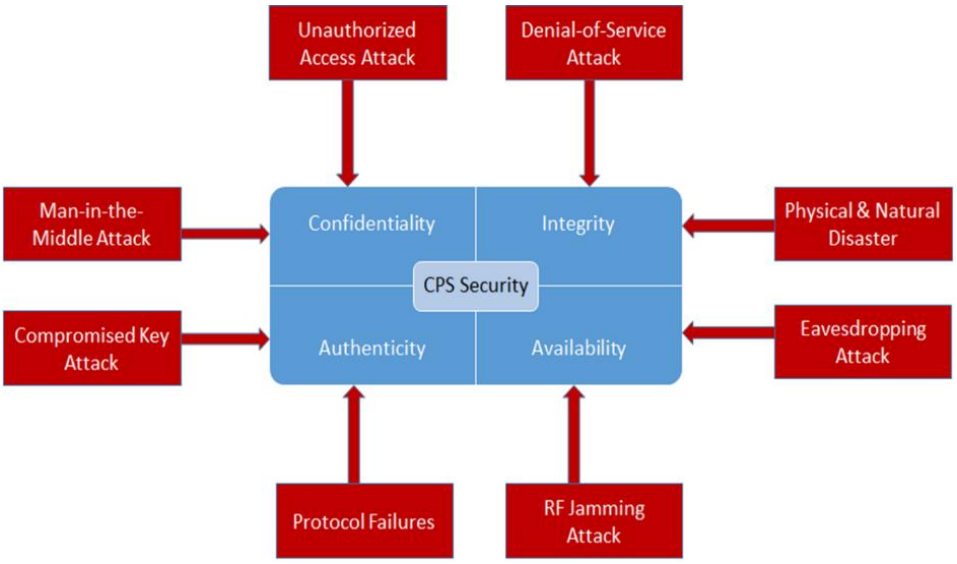

Figure 3. Security goals and threats.

3.1. Security Goals of Cyber-Physical System

The acceptance of cyber-physical systems in society depends on trust from users that must be earned. This trust can only be gained by providing adequate security goals to users. Security goals aim to protect the system from threats and vulnerabilities and reduce risk factors. We aim to extend our understanding of security goals for CPSs. For instance, in the case of sensor data oriented systems with multiple sensor nodes generating data, security issues are crucial to see if the data generated is coming from a trustworthy source. This shows that authentication, availability, integrity, and confidentiality are very important security goals in CPSs. The following are important security goals:

3.1.1. Authentication

Nodes (sensors) should be identified and authenticated before adding them to the network [51]. Authentication in a CPS is considered difficult to achieve as, in some cases, it requires heterogeneous network authentication. The lack of an authentication process can lead to exposure of the network/information to an unauthorized user.

3.1.2. Availability

Availability ensures that the information is available all the time to the authorized user when required. The risk level has to be increased when there is a Denial of Service (DoS) attack or service distractions as a result of a hardware failure, systems updates, or power failure [52].

3.1.3. Integrity

Integrity denotes that the information is accurate and trustworthy to the users. Integrity is disrupted when an information is modified in an unauthorized manner. This ensures that the data cannot be modified in any manner [52].

3.1.4. Confidentiality

The data being transferred within the network is inaccessible to an unauthorized user. The risks associated with confidentiality involve exposure of network information to an unauthorized user [53].

3.2. Threats of Cyber-Physical Systems

Threat could be anything that may harm the cyber-physical systems. The following are important threats of CPS:

3.2.1. Eavesdropping

There are several different ways to attack cyber-physical systems. The first would be eavesdropping, which is used to intercept the exchanged data [54]. In a military context, this is of particularly high relevance as tactical information can be intercepted by the enemy. Also in the case of industrial espionage, this can lead to serious damage to the organization.

3.2.2. Compromised-Key Attack

In a compromised-key attack, the attacker gains access to a key for the system, and thus, can modify data. The attacker can also access other areas of the system. This is done without the actual users of the system being aware of it [55].

3.2.3. Man-in-the-Middle Attack

Man-in-the-middle attacks are sending incorrect information. If it is not recognized as false, it influences the function of the system [56]. In the case of a system which controls the operation of train switches, such attack can lead to malfunctions or collisions of trains.

3.2.4. Denial-of-Service Attack

A denial-of-service attack floods the system with data; thus, the normal operation of the system cannot be continued [57]. For instance, this can lead to users being unable to retrieve their mobile bank data. However, more critical scenarios are also conceivable.

3.2.5. Physical Attack & Natural Disaster

In this type of attack, the attacker could harm physical devices such as hardware, sensors, cameras, terminals etc. Such attack could threaten human lives and this must be prevented at any cost. If these systems were attacked from outside, this could cause great harm. Natural disasters can also lead to a loss of human lives, and the sensors or actuators would be unusable, and the financial damages would be significantly higher [58].

3.2.6. Unauthorized Access

An unauthorized access of data is a real threat that should be handled at the beginning of SDLC. There are many possibilities that an attacker can easily access user information. This information can be injected through network communication or sensor nodes [59].

3.2.7. Radio Frequency Jamming

Radio Frequency (RF) jamming aims to paralyze communication from the physical environment. This may interfere with the interaction of sensors to PLC or any gateways. Usually, the radio frequency jamming occurs with radio signals to detach the tag through electromagnetic waves or high level traffic of signals [60].

3.2.8. Protocol Failures

An adversary can threaten the failure of communication protocols, which may lead to the failure of network communication and physical hardware [61].

Looking at these threats and their consequences, it becomes clear that security for cyber-physical systems has a central role to play in the development of these systems. If this aspect is not taken into account, attackers are free to access data and abuse systems as they wish. In fact, dangers would arise which are hardly imaginable. As a result, the value offered by cyber-physical systems would be almost completely lost.

4. Proposed Security Requirements Engineering Framework for CPS

The purpose of Security Requirements Engineering (SRE) framework is to identify security requirements. Meanwhile, there is no comprehensive security requirements engineering framework available for CPSs, since the nature of CPSs is quite different to classical software systems because of the CPS characteristics of heterogeneity and adaptability. Therefore, we propose a security requirements engineering framework that provides ways to determine security requirements throughout the Requirements Engineering (RE) phase, which consists of a number of activities to elicit and finalize the security requirements for CPSs. These activities identify security to avoid potential consequences of attacks for a CPS. The purpose of this framework is to develop early security concepts in the requirements engineering phase. This quest leads to RE methodologies so that security concerns can be addressed during the early stages of software development. The proposed framework is a systematic approach to incorporate security goals, threats, and risk assessment that are critical to the CPS. We have a set of 8 main activities, and one important technique called *misuse case* as shown in Figure 4. A misuse case is operated like a use case but it is inverse, i.e., a function should not permit the system to operate in a normal condition [62]. This technique is common across all the processes of main activities. Moreover, this CPS framework offers complete guidance to practitioners and researchers to determine security requirements. The framework recognizes the activities that are essential for requirement analysts to follow in order to identify the security requirements for CPS.

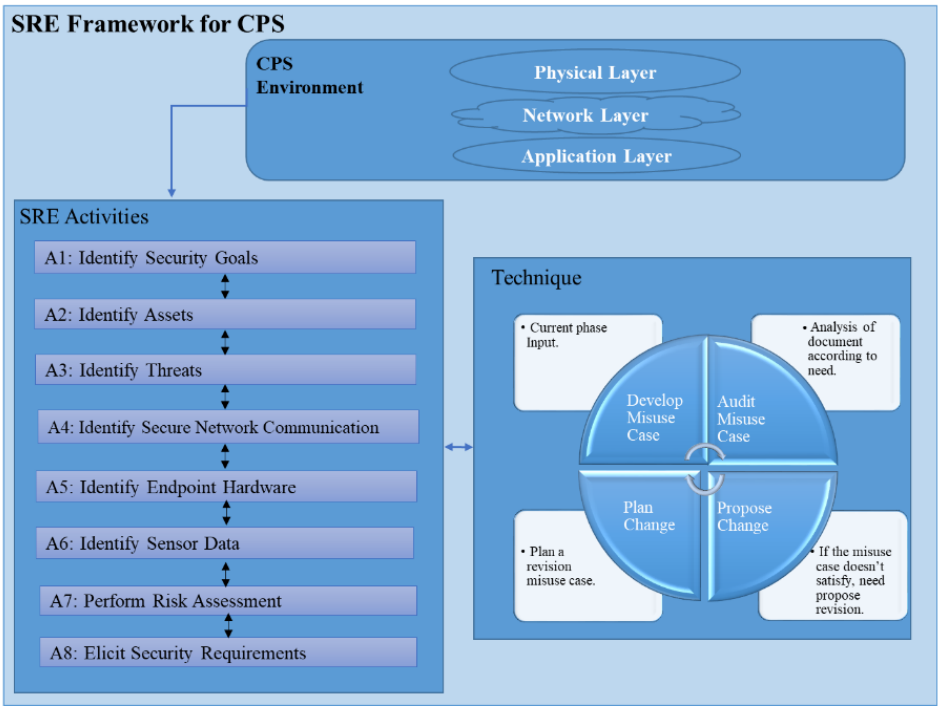

Figure 4. SRE Framework for CPS.

SRE Activities

The CPS framework has the following 8 activities (A1 to A8). These activities are selected based on their importance vis a vis security requirements engineering [63–66], and the characteristics of cyber-physical systems [1,2,58].

A1: Identify Security Goals

Business goals, quality attributes are combined to develop the required security decision. This is achieved to identify security goals. Security goals mainly refer to confidentiality, integrity availability, and authentication.

A2: Identify Assets

Assets could be anything that has value for the organization i.e., people, money, software, hardware, sensors etc. Therefore, the purpose of this activity is to determine all the assets of the CPS components. This activity also involves the environmental and organization asset evaluation. These assets usually involve human resources, data resources, network resources, and sensors and physical components.

A3: Identify Threats

The purpose of this activity is to identify the threats for cyber-physical systems. Threats are then categorized with the help of a misuse case. We have classified threats into 3 categories; software layer, network layer, and physical layer.

A4: Identify Secure Network Communication

The purpose of this activity is to identify a secure network communication protocol. Wireless sensor network devices need to be properly authenticated in the network domain. It is important to deploy standard security protocols like Transport Layer Security (TLS), Datagram Transport Layer Security (DTLS), Internet Key Exchange/Internet Protocol Security (IKE/IPsec) and Host Identity Protocol-Diet Exchange (HIP-DEX). This performs the communication protocol secure for wireless sensor networks.

A5: Identify Endpoint Hardware

It is recommended to use only authenticated endpoint hardware. This activity involves the identification of supporting hardware that may include a sensor, machine, router, reader, point-of-sale terminal, server and smart devices.

A6: Identify Sensor Data Generation/Communication

Sensors and actuators are devices that communicate with the external environment. The sensor generates data regarding the object with which they are in contact. These data are received through an Application Program Interface (API), and pass to a Programming Logic Controller (PLC) and Supervisory Control And Data Acquisition (SCADA). Some of the higher-level sensors provide support for the cloud of centralized data for data broadcasting. These sensors use Machine to Machine (M2M) protocols for communications. Since different sensors depict different mediums, we have proposed to identify all such factors during sensor analysis.

A7: Perform Risk Assessment

The purpose of this activity is to evaluate a risk-related impact. The process elicits the possible risk that might occur in the security requirements. Assets and threats of the system are also identified in this step. The impact of the risk on the asset is organized into groups, ranking the impact from 0 to 4 on the scale of low to high. The ranking impact is then used to calculate the impact factor of each risk. The impact cost sums the value of identified risk cost. Ranking and expert opinion are applied

on impact factor of each risk. The highest value of risk cost should be specified high priority while eliciting security requirements. The impact factor of each risk is calculated by the following formula:

$$\text{The impact factor of risk } = \sum_{n}^{0} \text{Impact cost.}$$

A8: Elicit Security Requirements

The purpose of this activity is to elicit, analyze, and specify the security requirements. Precise and unambiguous requirements are organized and written down.

5. How to Apply CPS Framework

The proposed framework is best worked out by the security advisors, the team of requirements analysts, and other stakeholders who are involved in the project. Table 1 illustrates the way in which the proposed framework is applied to the CPS, which explains the input, technique, output and the name of the activity. The framework is in the form of a workflow process of activities that needs to be followed. The output from each activity represents its completion. The framework proposed an agile methodology to select the required activity. The analyst needs to review each activity and propose changes if required.

Table 1. Framework Workflow Process.

Activity	Input	Technique	Output
A1: Identify security goals	Business goals, quality attributes	Misuse case, workshop session, interview, survey	List of security goals
A2: Identify assets	Entity of properties & their interaction	Misuse case, workshop session, detail analysis	List of CPS assets
A3: Identify threats	General list of CPS, list of security goals & assets	Misuse case, brainstorming session, questionnaire	List of CPS threats
A4: Identify secure network communication	Types of communication, protocol	Misuse case, analysis and comparison	List of secure network communication
A5: Identify endpoints hardware	All possible endpoints hardware, list of hardware vendors	Misuse case, group discussion	List of endpoint hardware and accepted vendors
A6: Identify sensor communication medium	Types of sensor	Misuse case, analysis and comparison	List of sensor types
A7: Perform Risk assessment	Impact of attacker capabilities, list of assets & threats	Risk cost factor calculation based on impact, risk ranking	Risk cost factor, risk rank result
A8: Elicit security requirements	list of security goals, assets & threats, risk result	Analysis, group discussion, interviews	List of security requirements

6. Case Study

The proposed CPS framework has been applied on a smart car parking system. Therefore, a functional prototype was developed for the demonstration of a smart car parking system, consisting of a physical and a software implementation. Figure 5 shows the functional prototype; each parking lot is connected to a Raspberry Pi 3, which collects the evaluated sensor data from the Arduino and sends it to ARTIK cloud. The system maintains the information of the vehicle entering the parking area. The system uses the sensor to determine the identity of the vehicle. The user of the system gets support through a mobile application. Only registered users are able to use the system. In the following section, we have applied 8 activities of the proposed framework and identified its effectiveness.

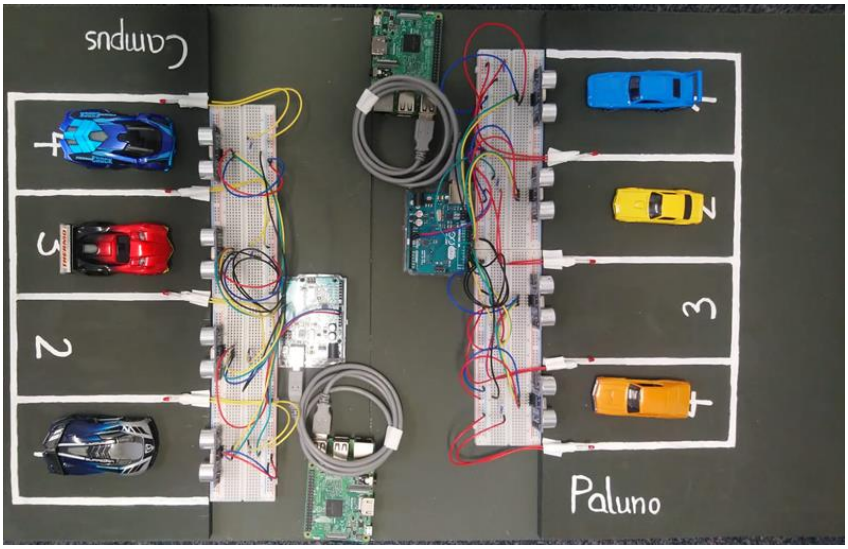

Figure 5. Functional Prototype of Smart Car Parking System.

A1: Identify Security Goals

1. Confidentiality: confidentiality is the major element of security goals. The user and vehicle data shall not disclose personal information to an unauthorized person.
2. Integrity: data is strongly related to the accuracy of the system. Therefore, its integrity is important. Any unwanted update can cause extreme issues, such as two users claiming one parking space, which can cause an issue over vehicle and user ownership, or even accidents by providing improper free lot information.
3. Authorization: the system shall require strong authorization while accessing data, as it manages several user's information simultaneously. Third-party sensor authorization is also needed because it affects the system's accuracy and privacy. Therefore, it is important to ensure the device's good operation due to system criticality.
4. Role-based access control: different authorized users have various roles in the system. These roles define the access limit to a diverse data set, hardware, or screens present in the system. The system maintains the access control list to define the assigned task of reachability in the resources. This can be facilitated by login id, profile detail, port, network channel, and important addresses.
5. Robustness: the system should have a backup or alternate servers, sensors, and electricity sources to provide a timely response if any failure occurs.
6. Availability of data: this means that requested data shall be available when it is needed. Therefore, it is important that the system should be responsive all the time or at a specified time.
7. Contractual integrity: third-party hardware, software, and network providers should properly follow a written contractual definition.

A2: Identify Assets

Assets for CPSs are grouped into 3 categories: software layer, network layer and physical layer. After analysis of the car parking system entities, we have listed 8 assets for the smart car parking system.

1. User data
2. Vehicular data
3. Channel of data
4. Server
5. Database system

6. Arduino
7. Raspberry pi
8. Sensors

A3: Identify Threats

Each asset is then assessed relative to functional requirements, and its negative uses are identified. We have identified 20 major threats. However, because of brevity, we are providing 9 of them, along with their definitions.

1. Data sniffing: over the channel or from the server, a sniffer can get user data over the transfer channel of the user app or database server. This could happen if encryption is absent.
2. Data manipulation without backup.
3. Data tampering: through unauthorized or authorized access. This needs a strong authorization process for critical data.
4. Physical sensor not responding: an attacker could steal or damage the sensor.
5. Data conversion is not working: the digital to analogue conversion after vehicle "id" is processed could face issues. This could happen because its hardware system or software is malfunctioning.
6. Electricity breakdown: intentional or unintentional electricity blackouts cause smooth processing of the process.
7. Data transfer is diverted to unauthorized alternative database: an attacker can transfer stored data to an alternative, unauthorized database.
8. Weak internet connection while updating: poor network bandwidth. Low bandwidth can cause slow updating or data transfer.
9. The central server is not responding: central server does not respond to user or app server because of a malicious action. This could lead to data loss.

A4: Identify Network Communication

We have identified different network protocols, but Transport Layer Security (TLS) is the most secure network communication protocol in smart car parking systems.

A5: Identify Endpoint Hardware

We have determined 9 endpoint hardware components for smart car parking systems:

1. Database storage system hardware.
2. Sensor nodes.
3. Network wires.
4. Routers.
5. Hubs.
6. Receiver.
7. Senses.
8. Computers that provides facility for software manipulation.
9. User tablets, iphone or android devices.

A6: Identify Sensor Communication Medium

The sensor and actuator of car parking lots are situated in a physical environment. Therefore, such an environment needs to be protected against tampering and unauthorized access. Sensor communication uses different channels to receive data from the physical environment. We have used an IEEE 802.15.4 based communication protocol in a smart car parking system to secure communication.

A7: Implement Risk Assessment

In this activity, we analyzed threats, assets, and the expected risks that can affect them. We have identified the impact of each risk on the asset, and categorized them into 4 values. In the following section, we have provided the few important risk analyses. Table 2 shows the impact range from 0 to 4. Four is the highest value and 0 the lowest. The large value of risk cost shows the high impact of risk. Therefore, important attention needs to be paid to the high risk cost.

Table 2. Risk Cost factor.

Risk ID	Event	Threat	Asset	Impact	Risk Cost
Rk1	Data authorized access	Misuse of data	Database User Parking Vehicular data	4: Critical data integrity lost. Information is no longer confidential 4: Data can be misused, Misguidance to the user 4: Misguidance 4: Can cause stealing	4 + 4 + 4 + 4 = 16
Rk2	Vehicular no. reading publication/stolen	Pass to unauthorized channel/data base	Database Communication channel User	4: No record 4: Malicious record 3: No direct impact, however it's been tampered 3: private data	4 + 4 + 3 + 3 = 14
Rk3	Data conversion malfunction	Wrong record updated. Garbage data store	User Database parking	3: Wrong information across valid user 3: no data updated 3: Wrong information across valid user	3 + 3 + 3 = 9
Rk4	Sensor not responding	Sensor stolen	Database Communication channel User Parking location sensor	2: No direct effect, no data update, hence no working process 1: No data transfer 4: No information generated 1: No information generated and no information transfer 4: No functional system	2 + 1 + 4 + 1 + 4 = 12
RK6	Unencrypted channel	Critical data disclosed	Communication channel Database User Parking App and webpage	4: Lost of trust 4: Not updated 4: Lost of trust and user data is disclosed 3: Not function properly 4: Loss of trust and malfunctioning	4 + 4 + 4 + 4 + 3 = 19
Rk5	Internet connection failed	Weak internet connection while updating	Database App and webpage User	4: No data processing 4: No Application usage 4: Dissatisfaction	4 + 4 + 4 = 12
Rk7	Sensor data- not generated/encrypted	Sensor malfunction or stolen	Sensor User Database	4: Not function properly 2: frustration 3: Private data lost 3: Data privacy compromise	4 + 2 + 3 + 3 = 12
Rk8	Data manipulated	Channel diverted	Database User Communication channel Parking location Server	4: Noise data 4: Data integrity lost 4: User data privacy effected 3: Malicious data transfer 4: Malicious information generated and no information transfer 3: May receive error code or useless data	4 + 4 + 4 + 3 + 4 + 3 = 22
Rk9	Overloaded channel	Channel diverted	Database User Car Communication channel Parking location Server	3: Unwanted data generated 4: User data privacy effected 4: User data privacy effected 3: Malicious data transfer 4: data transfer to another server 4: Malicious information generated and no information transfer 3: Not receiving any data or malicious data	3 + 4 + 4 + 3 + 4 + 4 + 3 = 25

A8: Elicit Security Requirements

In order to elicit security requirements for a smart car parking system, the security goals, assets, and threats are analyzed, together with the security risks. All security goals, assets, threats, and risks of the system are subjected to detailed analysis by the stakeholders. On this basis, we have determined 40 security requirements for a smart car parking system, as shown in Table 3.

Table 3. Security Requirements.

No.	Security Requirements (SR)
SR-1	The system shall have strong authentication.
SR-2	The system shall be knowledgeable of spoofing in the car parking database.
SR-3	The system shall monitor unauthorized modification of car parking database.
SR-4	The system shall prevent malicious malware in the car parking database.
SR-5	The system shall control unauthorized access of car parking database.
SR-6	The system shall control unauthorized modification to the communication between a server and user application.
SR-7	The system shall monitor unauthorized access to communication between a server and user application.
SR-8	The system shall monitor Denial-of-Service (DoS) attacks on the car parking database.
SR-9	The system shall require special security that should be maintained for hardware supports.
SR-10	The system shall protect 3rd-party software for sensor support.
SR-11	The system shall provide alternative sensors for smooth processing.
SR-12	The system shall provide alarm notification in case of missing hardware.
SR-13	The system shall require network communications to be protected. Encryption should be established.
SR-14	The system shall require network communication receivers that should be audited on a frequent basis. To ensure its security from viruses and data sniffers.
SR-15	Data channels should be encrypted.
SR-16	A continuous data channel audit mechanism should be present.
SR-17	Channel load should be monitored.
SR-18	A channel-dedicated IP list should be maintained.
SR-19	Data receiving IPs should be checked from the authorized ones to ensure the absence of unsecure recovery.
SR-20	The system database should have different access levels per stakeholders.Critical data should be encrypted and password protected.
SR-21	Car data (user information) should be protected with passwords.
SR-22	Role-based access control over the data should be ensured.
SR-23	Alternative sensors, recovery routers and network cables should be in place to ensure timely recovery and minimal response time.
SR-24	The system shall provide a stable internet connection.
SR-25	The internet connection should be secured by a password. Internet must be protected through a passcode based on car id or user id to ensure use by a dedicated user.
SR-26	The system shall provide automatic connection to an alternative internet that should be ensured in times of failure.
SR-27	In the case of sensors/actuators damage or theft, the system shall be protected with an alarm, and the administrator shall be notified.
SR-28	The data should not be diverted or transferred through the network communication.
SR-29	The sending of data from sensor to gateway cannot be readable by an unauthorized user.
SR-30	The sending of data from gateway to server cannot be readable by an unauthorized user.
SR-31	There should be an alternate solution, in case sensor or actuator is damaged/stolen.
SR-32	The sensor shall not be able to read the incorrect data from the car parking lot.
SR-33	The sensor shall not be able to send the incorrect data to the data acquisition board.
SR-34	The data acquisition board shall not be able to send incorrect data to the server/controller.
SR-35	The system shall monitor unauthorized replacement of sensors/actuators in the parking lot.
SR-36	The system shall monitor interruption of sensor/actuator.
SR-37	The sensor shall not divert the data into another data acquisition board/server.
SR-38	The system shall provide the necessary damage/theft protection of sensors/actuators.
SR-39	The system shall monitor car parking reservation information from authorized access/modification.
SR-40	The system shall monitor parking lot status information from unauthorized access/modification.

7. Comparison of Results

In this work, first we compared the most commonly-used security requirements engineering frameworks. The best papers were selected after a thorough search in the literature. The criteria of comparison were based on particular parameters, and these parameters were decided on their significance of security requirements engineering and characteristics on CPSs. Table 4 shows the comparison of different security requirements engineering frameworks. This comparison helps us to determine the strengths and weaknesses of each framework. Our findings from this comparison survey indicate that none of the frameworks performs all the required activities for a secure software system. This may result in the development of unsecured cyber-physical systems. Furthermore, this comparison helps us to identify the shortcomings in SRE frameworks which have been rectified in our proposed security requirements engineering framework for CPS.

Table 4. Comparison of SRE Frameworks.

Frameworks Activities	SQUARE [36,37]	MS SDL [45,67]	UMLsec [8,48,68]	Secure Tropos [46,47,69]	CLASP [44]	SREP [38,39,70]	CORAS [42,43,71]
A1	x		x	x		x	x
A2	x	x	x		x	x	x
A3	x	x	x	x	x	x	x
A4							
A5							
A6							
A7	x	x			x	x	x
A8	x	x	x	x	x	x	

8. Conclusions

Security requirements are a significant part of cyber-physical systems, but there are a lack of processes to develop secure systems. Many security requirements methodologies have been proposed, but these are limited only to software, and none supports cyber-physical systems. In this paper, our main contribution is to provide a comprehensive security requirements engineering framework for cyber-physical systems that can offer complete guidelines for practitioners and researchers to determine security requirements. The novelty of such an implementation at this scale has not been significantly reported in the literature. The proposed CPS framework identifies security requirements throughout the requirement engineering phase. This not only helps to build a secure CPS, but also to avoid potential consequences in cyber-attacks. Furthermore, our proposed security requirements engineering framework is compared with other existing software security frameworks. The encouraging result shows that none of the software security frameworks implements all the essential activities for the development of secure CPS. To evaluate the CPS framework, we have applied our approach to a smart car parking system, from which we obtained promising results. We have identified 40 security requirements with the help of proposed CPS framework; these major security requirements have contributed to developing a secure smart car parking system. The findings of this research will be of great benefit to practitioners and researchers.

Author Contributions: Supervision, V.G.; Writing—original draft, S.u.R.

Funding: This research was funded by European Community CPS.HUB NRW grant number EFRE Nr. 0-4000-17.

Acknowledgments: This work has been supported by the European Community through project CPS.HUB NRW, EFRE Nr. 0-4000-17.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rehman, S.; Gruhn, V. Recommended Architecture for Car Parking Management System based on Cyber-Physical System. In Proceedings of the International Conference on Engineering & MIS, Monastir, Tunisia, 8–10 May 2017.
2. Kalloniatis, C.; Kavakli, E.; Gritzalis, S. Addressing privacy requirements in system design: The PriS method. *Requir. Eng.* **2008**, *13*, 241–255. [CrossRef]
3. Zhang, L. A framework to specify big data driven complex cyber physical control systems. In Proceedings of the 2014 IEEE International Conference on 2014 Information and Automation (ICIA), Hailar, China, 28–30 July 2014; pp. 548–553.
4. Beckers, K.; Faßbender, S.; Hatebur, D.; Heisel, M.; Côté, I. Common criteria compliant software development (CC-CASD). In Proceedings of the 28th Annual ACM Symposium on Applied Computing, Coimbra, Portugal, 18–22 March 2013; pp. 1298–1304.
5. Lund, M.S.; Solhaug, B.; Stølen, K. *Model-Driven Risk Analysis: The CORAS Approach*; Springer Science & Business Media: New York, NY, USA, 2010.
6. CC: ISO/IEC 15408 Information Technology—Security Technology—Evaluation Criteria for IT Security V2.1. Available online: https://www.iso.org/standard/50341.html (accessed on 1 December 2009).
7. Van Lamsweerde, A. Goal-Oriented Requirements Engineering: A Guided Tour. In Proceedings of the RE'01: 5th International Symposium on Requirements Engineering, Toronto, ON, Canada, 27–31 August 2001.
8. Jürjens, J. UMLsec: Extending UML for secure systems development. In Proceedings of the «UML» 2002—The Unified Modeling Language, Dresden, Germany, 20 September 2002; pp. 1–9.
9. Leitão, P.; Colombo, A.W.; Karnouskos, S. Industrial automation based on cyber-physical systems technologies: Prototype implementations and challenges. *Comput. Ind.* **2016**, *81*, 11–25. [CrossRef]
10. Cleveland, F. Cyber Security Issues for Advanced Metering Infrastructure (AMI). In Proceedings of the IEEE Power and Energy Society General Meeting-Conversion and Delivery of Electrical energy in the 21st Century, Pittsburgh, PA, USA, 20–24 July 2008.
11. Rehman, S.; Gruhn, V. Security Requirements Engineering (SRE) Framework for Cyber-Physical Systems (CPS): SRE for CPS. In Proceedings of the 16th International Conference on Intelligent Software Methodologies, Tools, and Techniques (SOMET_17), Kyushu, Japan, 26–28 September 2017.
12. Ramachandran, M. Software security requirements management as an emerging cloud computing service. *Int. J. Inf. Manag.* **2016**, *36*, 580–590. [CrossRef]
13. Mead, N.R. *How to Compare the Security Quality Requirements Engineering (SQUARE) Method with Other Methods (No. CMU/SEI-2007-TN-021)*; Carnegie Mellon University Software Engineering Institute: Pittsburgh, PA, USA, 2007.
14. Yoo, H.; Shon, T. Challenges and research directions for heterogeneous cyber–physical system based on IEC 61850: Vulnerabilities, security requirements, and security architecture. *Future Gener. Comput. Syst.* **2016**, *61*, 128–136. [CrossRef]
15. Liu, Y.; Peng, Y.; Wang, B.; Yao, S.; Liu, Z. Review on cyber-physical systems. *IEEE/CAA J. Autom. Sinica* **2017**, *4*, 27–40. [CrossRef]
16. Subramanian, N.; Zalewski, J. Quantitative assessment of safety and security of system architectures for cyber-physical systems using the NFR approach. *IEEE Syst. J.* **2016**, *10*, 397–409. [CrossRef]
17. Conforti, R.; de Leoni, M.; La Rosa, M.; van der Aalst, W.M.; ter Hofstede, A.H. A recommendation system for predicting risks across multiple business process instances. *Decis. Support Syst.* **2015**, *69*, 1–19. [CrossRef]
18. Slay, J.; Miller, M. Lessons learned from the maroochy water breach. In Proceedings of the International Conference on Critical Infrastructure Protection, Boston, MA, USA, 19 March 2007; pp. 73–82.
19. Cárdenas, A.A.; Amin, S.; Lin, Z.S.; Huang, Y.L.; Huang, C.Y.; Sastry, S. Attacks Against Process Control Systems: Risk Assessment, Detection, and Response. In Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, Hong Kong, China, 22–24 March 2011; ACM: New York, NY, USA, 2011; pp. 355–366.
20. Ashford, W. On-Demand Service Aims to Cut Cost of Fixing Software Security Flaws. Available online: http://www.computerweekly.com/Articles/2009/07/14/236875/on-demand-service-aims-to-cut-cost-of-fixing-software-security.htm (accessed on 10 July 2018).
21. Gartner. Available online: http://www.gartner.com/newsroom/id/2828722 (accessed on 1 August 2014).

22. CERT-SEI. Available online: www.cert.org (accessed on 1 September 2016).
23. CERT-UK. Available online: https://www.cert.gov.uk/ (accessed on 1 March 2016).
24. Ur Rehman, S.; Khan, M.U. Security and Reliability Requirements for a Virtual Classroom. *Procedia Comput. Sci.* **2016**, *94*, 447–452. [CrossRef]
25. Robles, R.J.; Kim, T.H. Applications, Systems and Methods in Smart Home Technology: A review. *Int. J. Adv. Sci. Technol.* **2010**, *15*, 37–48.
26. Tondel, I.A.; Jaatun, M.G.; Meland, P.H. Security requirements for the rest of us: A survey. *IEEE Softw.* **2008**, *2*, 20–25. [CrossRef]
27. Shahzad, M.; Shafiq, M.Z.; Liu, A.X. A large scale exploratory analysis of software vulnerability life cycles. In Proceedings of the 34th International Conference on Software Engineering, Zurich, Switzerland, 2–9 June 2012; pp. 771–781.
28. Almorsy, M.; Grundy, J.; Müller, I. An analysis of the cloud computing security problem. *arXiv*, 2016.
29. Salini, P.; Kanmani, S. Survey and analysis on security requirements engineering. *Comput. Electr. Eng.* **2012**, *38*, 1785–1797. [CrossRef]
30. Khan, M.U.A.; Zulkernine, M. Quantifying security in secure software development phases. In Proceedings of the 2008 32nd Annual IEEE International Computer Software and Applications Conference, Turku, Finland, 28 July–1 August 2008; pp. 955–960.
31. Khan, M.U.A.; Zulkernine, M. On selecting appropriate development processes and requirements engineering methods for secure software. In Proceedings of the 2009 33rd Annual IEEE International Computer Software and Applications Conference, Seattle, WA, USA, 20–24 July 2009; Volume 2, pp. 353–358.
32. McGraw, G. Software Security: Building Security In, Addison Wesley. In Proceedings of the 2006 17th International Symposium on Software Reliability Engineering, Raleigh, NC, USA, 7–10 November 2006.
33. Perrig, A.; Stankovic, J.; Wagner, D. Security in wireless sensor networks. *Commun. ACM* **2004**, *47*, 53. [CrossRef]
34. Yan, Q.; Yu, F.R.; Gong, Q.; Li, J. Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 602–622. [CrossRef]
35. Martins, D.; Guyennet, H. Wireless sensor network attacks and security mechanisms: A short survey. In Proceedings of the 13th International Conference on 2010 Network-Based Information Systems (NBiS), Gifu, Japan, 14–16 September 2010; pp. 313–320.
36. Mead, N.R.; Stehney, T. *Security Quality Requirements Engineering (SQUARE) Methodology*; ACM: New York, NY, USA, 2005; Volume 30, pp. 1–7.
37. Mead, N.; Viswanathan, V.; Padmanabhan, D. *Incorporating Security Quality Requirements Engineering (SQUARE) into Standard Life-Cycle Models*; Technical Note, CMU/SEI-2008-TN-006; Software Engineering Institute, Carnegie Mellon University: Pittsburgh, PA, USA, 2008.
38. Mellado, D.; Fernández-Medina, E.; Piattini, M. Applying a security requirements engineering process. In *European Symposium on Research in Computer Security*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 192–206.
39. Mellado, D.; Fernández-Medina, E.; Piattini, M. A common criteria based security requirements engineering process for the development of secure information systems. *Comput. Stand. Interfaces* **2007**, *29*, 244–253. [CrossRef]
40. Lee, J.; Lee, J.; Lee, S.; Choi, B. A CC-based security engineering process evaluation model. In Proceedings of the 27th Annual International Computer Software and Applications Conference on COMPAC 2003, Dallas, TX, USA, 3–6 November 2003; pp. 130–135.
41. Feng, N.; Wang, H.J.; Li, M. A security risk analysis model for information systems: Causal relationships of risk factors and vulnerability propagation analysis. *Inf. Sci.* **2014**, *56*, 57–73. [CrossRef]
42. Stolen, K.; den Braber, F.; Dimitrakos, T.; Fredriksen, R.; Gran, B.A.; Houmb, S.H.; Aagedal, J.O. Model-based risk assessment–the CORAS approach. In Proceedings of the NIK (2002) Informatics Conference, Kongsberg, Norway, 25–27 November 2002.
43. Dimitrakos, T.; Ritchie, B.; Raptis, D.; Stølen, K. Model-based security risk analysis for Web applications: The CORAS approach. In Proceedings of the EuroWeb, Oxford, UK, 17–18 December 2002.

44. Viega, J. Building security requirements with CLASP. In Proceedings of the ACM SIGSOFT Software Engineering Notes, St. Louis, Missouri, 15–16 May 2005; ACM: New York, NY, USA, 2005; Volume 30, pp. 1–7.
45. De Win, B.; Scandariato, R.; Buyens, K.; Grégoire, J.; Joosen, W. On the secure software development process: CLASP, SDL and Touchpoints compared. *Inf. Softw. Technol.* **2009**, *51*, 1152–1171. [CrossRef]
46. Mouratidis, H.; Argyropoulos, N.; Shei, S. Security requirements engineering for cloud computing: The secure tropos approach. In *Domain-Specific Conceptual Modeling*; Springer: Cham, Switzerland, 2016; pp. 357–380.
47. Pavlidis, M.; Mouratidis, H.; Panaousis, E.; Argyropoulos, N. Selecting Security Mechanisms in Secure Tropos. In Proceedings of the International Conference on Trust and Privacy in Digital Business, Lyon, France, 28–31 August 2017; Springer: Cham, Switzerland, 2017; pp. 99–114.
48. Hatebur, D.; Heisel, M.; Jürjens, J.; Schmidt, H. Systematic development of UMLsec design models based on security requirements. In Proceedings of the International Conference on Fundamental Approaches to Software Engineering 2011, Saarbrücken, Germany, 26 March–3 April 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 232–246.
49. Mouratidis, H.; Jurjens, J. From goal-driven security requirements engineering to secure design. *Int. J. Intell. Syst.* **2010**, *25*, 813–840. [CrossRef]
50. Mellado, D.; Mouratidis, H.; Fernández-Medina, E. Secure Tropos framework for software product lines requirements engineering. *Comput. Stand. Interfaces* **2014**, *36*, 711–722. [CrossRef]
51. Ericsson, G.N. Cyber security and power system communication essential parts of a smart grid infrastructure. *IEEE Trans. Power Deliv.* **2010**, *25*, 1501–1507. [CrossRef]
52. Sridhar, S.; Manimaran, G. Data integrity attacks and their impacts on SCADA control system. In Proceedings of the IEEE PES General Meeting, Providence, RI, USA, 25–29 July 2010; pp. 1–6.
53. Oh, S.-R.; Kim, Y.-G. Security Requirements Analysis for the IoT. In Proceedings of the 2017 International Conference on Platform Technology and Service (PlatCon), Busan, South Korea, 13–15 February 2017; pp. 1–6.
54. Burmester, M.; Magkos, E.; Chrissikopoulos, V. Modeling security in cyber–physical systems. *Int. J. Crit. Infrastruct. Prot.* **2012**, *5*, 118–126. [CrossRef]
55. Zeng, K. Physical layer key generation in wireless networks: Challenges and opportunities. *IEEE Commun. Mag.* **2015**, *53*, 33–39. [CrossRef]
56. Stojmenovic, I.; Wen, S. The fog computing paradigm: Scenarios and security issues. In Proceedings of the 2014 Federated Conference on Computer Science and Information Systems (FedCSIS), Warsaw, Poland, 7–10 September 2014; pp. 1–8.
57. Zhu, B.; Joseph, A.; Sastry, S. A taxonomy of cyber-attacks on SCADA systems. In Proceedings of the Internet of Things (iThings/CPSCom) 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing, Washington, DC, USA, 9–22 October 2011; pp. 380–388.
58. Rajkumar, R.R.; Lee, I.; Sha, L.; Stankovic, J. Cyber-physical systems: The next computing revolution. In Proceedings of the 47th Design Automation Conference, Anaheim, CA, USA, 13–18 June 2010; ACM: New York, NY, USA, 2010; pp. 731–736.
59. Farooq, M.U.; Waseem, M.; Khairi, A.; Mazhar, S. A critical analysis on the security concerns of internet of things (IoT). *Int. J. Comput. Appl.* **2015**, *111*. Available online: http://www.pcporoje.com/filedata/592496.pdf (accessed on 10 July 2018).
60. Khoo, B. RFID as an enabler of the internet of things: Issues of security and privacy. In Proceedings of the Internet of Things (iThings/CPSCom), 2011 International Conference on and 4th International Conference on Cyber, Physical and Social Computing, Dalian, China, 19–22 October 2011; pp. 709–712.
61. Al-Fuqaha, A.; Guizani, M.; Mohammadi, M.; Aledhari, M.; Ayyash, M. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 2347–2376. [CrossRef]
62. Sindre, G.; Opdahl, A.L. Eliciting security requirements with misuse cases. *Requir. Eng.* **2005**, *10*, 34–44. [CrossRef]
63. Suleiman, H.; Svetinovic, D. Evaluating the effectiveness of the security quality requirements engineering (SQUARE) method: A case study using smart grid advanced metering infrastructure. *Requir. Eng.* **2013**, *18*, 251–279. [CrossRef]

64. Zafar, N.; Arnautovic, E.; Diabat, A.; Svetinovic, D. System security requirements analysis: A smart grid case study. *Syst. Eng.* **2014**, *17*, 77–88. [CrossRef]
65. Gopal, T.; Subbaraju, M.; vivek Joshi, R.; Dey, S. MAR (S) 2: Methodology to articulate the requirements for security in SCADA. In Proceedings of the 2014 Fourth International Conference on Innovative Computing Technology (INTECH), Luton, UK, 13–15 August 2014; pp. 103–108.
66. Souag, A.; Mazo, R.; Salinesi, C.; Comyn-Wattiau, I. Reusable knowledge in security requirements engineering: A systematic mapping study. *Requir. Eng.* **2016**, *21*, 251–283. [CrossRef]
67. Mir, T.M.; Revuru, A.K.V.; Manohar, D.J.; Batta, V. Microsoft Corporation, 2012. Threat Analysis and Modeling during a Software Development Lifecycle of a Software Application. U.S. Patent 8,091,065, 3 January 2012.
68. Jürjens, J. *Secure Systems Development with UML*; Springer Science & Business Media: Berlin, Germany, 2005.
69. Mouratidis, H.; Giorgini, P. Secure tropos: A security-oriented extension of the tropos methodology. *Int. J. Softw. Eng. Knowl. Eng.* **2007**, *17*, 285–309. [CrossRef]
70. Labunets, K.; Massacci, F.; Paci, F. An experimental comparison of two risk-based security methods. In Proceedings of the 2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, Baltimore, MD, USA, 10–11 October 2013; pp. 163–172.
71. Fredriksen, R.; Kristiansen, M.; Gran, B.A.; Stølen, K.; Opperud, T.A.; Dimitrakos, T. The CORAS framework for a model-based risk management process. In Proceedings of the InInternational Conference on Computer Safety, Reliability, and Security, Catania, Italy, 10–13 September 2002; Volume 2, pp. 94–105.