Software Reliability and Quality Assurance Challenges in Cyber Physical Systems Security

Abel Yeboah-Ofori
School of Architecture, Computing & Engineering
University of East London.
London, E16 2GA, UK

u0118547@uel.ac.uk
yeboah007@hotmail.com

Abstract

Software Reliability is the probability of failure-free software operation for a specified period of time in a specified environment. Cyber threats on software security have been prevailing and have increased exponentially, posing a major challenge on software reliability in the cyber physical systems (CPS) environment. Applying patches after the software has been developed is outdated and a major security flaw. However, this has posed a major software reliability challenge as threat actors are exploiting unpatched and insecure software configuration vulnerabilities that are not identified at the design phase. This paper aims to investigate the SDLC approach to software reliability and quality assurance challenges in CPS security. To demonstrate the applicability of our work, we review existing security requirements engineering concepts and methodologies such as TROPOS, *I**, KAOS, Tropos and Secure Tropos to determine their relevance in software security. We consider how the methodologies and function points are used to implement constraints to improve software reliability. Finally, the function points concepts are implemented into the CPS security components. The results show that software security threats in CPS can be addressed by integrating the SRE approach and function point analysis in the development to improve software reliability.

Keywords: Software Reliability, Secure Tropos, I*, Cyber Physical Systems, Function Point Analysis.

1. INTRODUCTION

Software reliability and quality assurance challenges in CPS security have been a major security flaw leading to software errors and system failures. These software errors and system failures could lead to various discrepancies, including system shut down, abnormal operations, error propagation, manipulations, and data compromises. The cascading impacts on other network nodes in a supply chain environment that are connected to the CPS is unquantifiable. Applying patches after the software has been developed is outdated and a major security flaw. However, this has posed a major software reliability challenge as threat actors are exploiting unpatched and insecure software configuration vulnerabilities that are not identified at the design phase. Security Requirements Engineering (SRE) is relevant in SDLC phases as security controls and standards alone cannot facilitate comprehensive cyber resilience and security solutions in an organisational context [1]. Most organisations define their business requirements more abstractly as they may not have the foreknowledge of what is required in building supply chain security into a large CPS software project. CPS components integrate cyber digital, cyber physical and physical elements for process flow and information dissemination in real-time systems. Hence, the CPS must be resilient. The thought of software security being something that can be added as patches after the software has been developed or deployed and in use is outdated and a major security flaw. Threat actors are targeting unpatched software's and insecure software configuration vulnerabilities that are not identified at the design and implementation phases of the software development lifecycle (SDLC).

Security requirements engineering is a systems development process of establishing services that the customer requires from a system perspective and the constraints under which it operates. From a requirement engineering perspective, detecting software errors at the early stages of requirement capturing, requirement specification and analysis phases are critical in mitigating supply chain compromises. For instance, compromising the supply chain system is the manipulation of product delivery mechanisms before receipt by the final consumer [2]. Hence, detecting CPS compromises from software errors and configuration errors in the requirement specification phase is very important in reducing threats and vulnerabilities that adversaries could exploit to breach the processes within the network components. [1] posit that the information security requirements engineering approach can provide a comprehensive and structured elicitation and understanding of information security requirements. The cyberattacks that could be deployed include manipulation during development, alterations during production and diversions on delivery channels during distributions.

The paper aims to investigate the software development life cycle approach to software reliability and quality assurance challenges in CPS security. The focus of the paper is three-fold: First, we demonstrate the applicability of our work by review existing security requirements engineering concepts and methodologies such as TROPOS, I*, KAOS and Secure Tropos to determine their security relevance in the software development. Secondly, we consider how the methodologies could be used to implement security in the development phases. Finally, we implement the function points concepts into the CPS components security to improve software reliability in CPS security. The results show that software security threats in CPS can be addressed by integrating the SRE approach and function point analysis in the development to improve software reliability.

2. RELATED WORKS

This section discusses the state of the art in security requirement engineering methodologies, CPS security, and related works in software quality, function points analysis and its integration in the SDLC. Integrating cyber threat intelligence concepts in the requirements capturing and requirement specification phases during the requirements engineering process and constraints generated is the best approach. The security requirements must be captured holistically by the software developer by engaging the various departmental managers and staff to capture the systems processes. Software security requirements modelling captures various integrations of organizational systems network, hardware and software in a cyber physical systems environment. Pavlidis et al. (2012) posit that Identifying and analysing the security requirements along with the functional requirements provide a better comprehension of the systems security issues and limits conflicts throughout the development process [3]. There are existing requirements engineering methodologies such as Tropos [4], *I** [5], KOAS (Respect-IT, [6] and Secure Tropos [3] that supports various aspects of the software development lifecycle phases.

Cyber supply chain security requirements capturing encompass functional requirements for securing the supply inbound and outbound chains. These secure interactions require a software that is based on an open architecture that is evolving, changeable, interoperable and adaptable to new requirements. It requires a robust and autonomous software that can provide secure communication and minimal network serves interferences. CPS combines computational and physical processes, components, networks, embedded systems and software using sensors and actuators for information dissemination [7]. We discuss Tropos, *I**, KOAS and secure Tropos methodologies as follows. Alavi et al. (2013) propose steps that would enable the analysis and elicitation of risk, investment and return on investment concepts in the development of the Risk-Driving Investment Model [1]. However, with the unassailable nature of cybercrime, using a general approach may not prevent cyberattacks on CPS without integrating security in the SDLC phase. Pavlidis et al. (2012) propose a case tool called SecTro that supports automation modelling and analysis of security requirements based on Secure Tropos [3].

Function point analysis is a method used to estimate the size of a software cost estimation for software SDLC projects. Albrecht 1979 initially developed the Mark I function point analysis and counting metric-based method for estimating the software size [8]. Symons 1998 considers the Albrecht Mark I function point analysis model and proposed a Mark II approach that has fewer variables, greater ease of calibration against measurements or estimates and included the security into the method [9]. Abdullah et al 2010, proposed an enhanced parametric estimation model for software security characteristics using function point analysis calculations to encounter the security cost estimation [10]. Mukherjee et al. 2013, surveyed various metrics, model, and tools for software cost estimations to assist in determining the understanding of the relationships among scores of discrete parameters that can affect the outcomes of software projects [11]. Dhakad and Rajawat 2016, proposed a novel variant of an expert user programming of FPA for estimating accurate software size by including it in the list of general systems characteristics [12]. Alves et al 2014, carried out an empirical study on the estimation of size and complexity of software applications with FPA to develop web applications [13]. Abdullah et al 2010 proposed potential security factors in software cost estimation by identifying security throughout the SDLC [14].

3. APPROACH

As stated previously, our approach considers security requirements engineering methodologies and their relevance in SDLC for improving software reliability and quality assurance in CPS security. There are existing works of literature on FPA models and software estimation techniques such as COCOMO, SLIM, SEER-SEM and Check Point that have been used for SDLC projects [11]. However, we discuss user requirements and systems requirements, functional and non-functional requirements, and consider how TROPOS, *I**, KAOS and Secure Tropos methodologies are implemented SDLC.

3.1 User Requirements and Systems Requirements

There are two types of security requirements engineering concepts used to determine security constraints: user requirements, and systems requirements. The user requirements include diagrams, constraints, and operational statements of the user’s language of the services that the organisation may provide. The systems requirements include descriptions of systems function, operational constraints and structured documents that define how and what system implementation is required as part of the development. Figure 1 depicts the various requirement necessary for the implementation.

User requirements definitions include systems of customers, suppliers, distributors and staff systems usage activities. For instance, the AMI will generate a monthly statement detailing the amount of electricity used. Payments will be made via direct debit. The system will generate a monthly management report detailing the use of electric power in an area for those who have paid their utility bill regularly and those who don't. Systems requirements specifications will include a summary of all activities and transactions carried out on the systems and the various interactions with external entities, bank, suppliers, distributors, and customers.

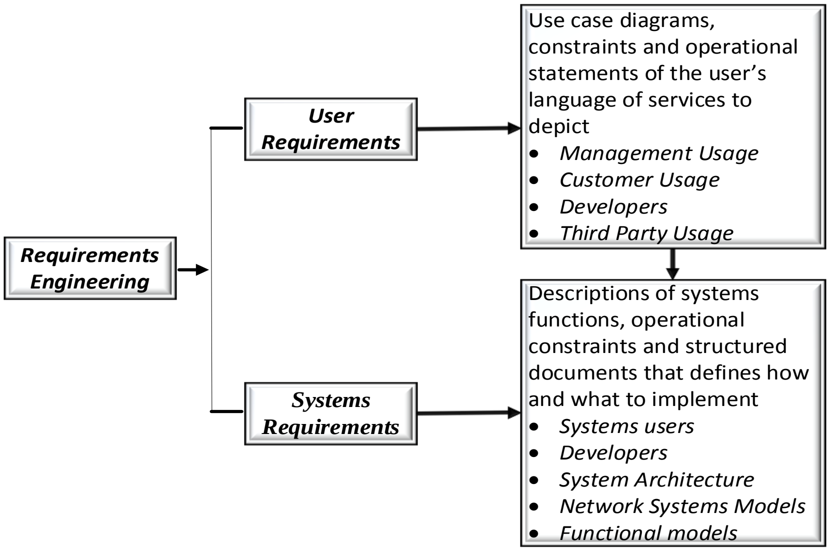
Requirements Engineering
User Requirements
Systems Requirements
Use case diagrams, constraints and operational statements of the user's language of services to depict
• Management Usage
• Customer Usage
• Developers
• Third Party Usage
Descriptions of systems functions, operational constraints and structured documents that defines how and what to implement
• Systems users
• Developers
• System Architecture
• Network Systems Models
• Functional models

FIGURE 1: User Requirements and System Requirements.

3.2 Functional Requirements

Functional Requirements are intended statements for the input parameters of the business services that should provide details of how the system should react to particular inputs and respond in a certain event [3]. Moreover, it should provide details of how the system should react to digital components of the CPS system that is integrated into the physical components and the physical (human) elements. For instance, SCADA systems usually have three components, namely: Field Devices, Services and Clients. The field devices include sensors and controllers PLCs that collect data from various sources [21]. The controllers are used to turn on the pumps and the valves on and off based on sensed data and control algorithms. The servers are responsible for collecting various field inputs such as starting and stopping processes, triggering alarms and implementing the logic required to automate a process. The clients are machines that interact with the servers via terminals, monitors activities and state of the SCADA as well as controlling the start and stop processes in a network. Using cybercrimes vectors to determine the security implications as part of the requirement elicitation process in systems development will ensure that the systems are secure instead of using security configurations that only ensure confidentiality integrity and availability.

3.3 Non-functional Requirements

Non-functional Requirements are constraints on the whole system services or functions that are provided by the system including controls on timing, and on the development procedures, regulations, and policies [3]. The CPS server operates on a distributed platform and gathers various field inputs such as starting and stopping processes, triggering alarms and implementing the logic required to automate a process. The threat actor may attack the system from a remote source or a supplier's systems connected to the distributed system to compromise the CPS. Determining the constraints and functions of the systems is a major challenge looking at the heterogeneity and distributed nature of CPS. For instance, a SCADA system may be connected to over one hundred utility companies locally and globally using the internet and the cyberspace to monitor data usage, power variables, load shedding, bill payments to financial institutions and triggering fire alarms as well as reducing the response timing for dealing with anomalies. These CPS systems are at risk of being hacked into as cyberattacks such as cross site scripting attacks could be initiated from the client-side. Moreover, a cross site requested forgery attack could be initiated from the server-side of the CPS system.

4. IMPLEMENTATION

This section discussed the existing SRE methodologies' including TROPOS, *I**, KAOS and Secure Tropos methodologies. Further, we use the functional requirement and function point approach to implement security goal and constraints in the software.

4.1 Tropos Methodology

Tropos methodology in software development uses the notion of agents and related goals and plans in the requirements engineering [4]. It supports four phases of development including early requirement analysis, late requirement analysis, architecture design and detailed design. The early requirements analysis phases of the development lifecycle identify all stakeholders, goal, actors, functions, and dependencies through to implementation. The late development considers the functional and non-functional requirements of the system under development []. It identifies actors, threat actors, goal and security goal. Tropos is quite different from other agents object-oriented software development methodologies as it captures prescriptively what and how the system should be. Whereas the early requirements approach captures the reasons why software is developed and not what system should be. Tropos prescriptive approach captures and analyses all stakeholder goals for the requirements specification phase for the system under development [5].

4.2 *I** Framework

The *I** framework is a modelling language that supports the notion of distributed intentionality in the early phase of system modelling to understand the problem domain [5]. The modelling language considers the reasoning of the organizational environment and other heterogeneous network systems with different goals and actors that depends on each other. The *I** approach is oriented towards both the actor and goal modelling to answer the questions of WHO and WHY of the system under development whereas the Tropos approach answers the question of what a system should be [5].

4.3. KOAS Methodology

KOAS is a methodology used for requirements engineering method that enables the analyst to build requirement models and derive requirements document from KOAS models [6]. KOAS approach considers goals of all types, although it places less emphasis on the intentionality of actors compared to *I**. The approach can be used to build all types of systems and favours the identification of interesting properties and unexpected alternative designs. KOAS is designed to build a requirement capturing model that illustrates the challenge to be solved in line with the constraints that must be implemented by any provided results. KOAS is designed to tailor challenging descriptions of goals by allowing the manipulation of the concepts that define the relevant descriptions. Further, it enhances the analysis of the process of the challenges by providing a systematic approach for finding and arranging requirements captured. Furthermore, the design explains the responsibilities of stakeholders and allow efficient communication.

4.4 Secure Tropos

Secure Tropos is a security engineering methodology that considers security issues withing a software systems development lifecycle [3]. Secure Tropos is an extension of the Tropos Methodology that considers security requirements alongside functional requirements from the early stages of the systems development process. The Secure Tropos language uses basic concepts such as dependency, goal, task, resources and capabilities and adds security concepts such as security constraints, secure goal, secure plan, secure resource, and threat to capture the security concepts from both social and organizational settings. The concepts are used to model and reason about security from a specific system context. Secure Tropos modelling uses the following activities: Security reference modelling, security constraints modelling, secure entities modelling and secure capabilities modelling [3].

4.5 CPS Security Requirements

Security requirements are the constraints and expectations of how the systems should function to support all stakeholders and business needs [17]. The security requirements concepts include properties such as organisational requirements and business requirements, as well as systems user and operational requirements. The organisational requirements describe the high-level organisational objectives that must be performed to achieve an organisational goal. The organisational requirements contain concepts that specify the overall organisational environment and where integration between the software and the security constraints occurs to attain the organisational goal. The requirements consist of attributes such as user categories, stakeholders, description, user ID, acceptance criteria, time constraints, owners, and sources. The business requirements explain the requirement specifications, and the properties include customer needs and expectations that must be integrated to meet the system requirements. Systems requirements demand specific properties of the application, architecture and the technical requirements need to be able to describe the features and how the system must function. These system requirements properties include the constraints, assumptions and acceptance criteria and the external entities that will be interacting with the system. They include supply chain systems processes and constraints that are generated during the requirements engineering phase that forms the basis for the system. These processes and constraints are statements that support the user requirements and system requirements used to achieve the organisational goal. The user requirements capture operational constraints of the actors (including threat actors), and the system requirements set out the detailed functional and service constraints of the supply chain systems. These detailed descriptions are used to realise the organisational goal and security goal. Requirements engineering gather data about what the customer requires from a system and the constraints under which it operates. For instances, with an organisation’s goal, the user requirement will be to set up customer details on the CMS, distribute electric power to consumers and generate usage bills for each customer. The system requirement will include sending the bills to customers, getting customers to pay through an online banking system or a vending system using a prepaid card. An adversary goal may be to attack the system to compromise the business. The operational requirements determine the system parameters and configurations that are used to establish operational processes between the organisation and the third-party vendors on the supply chain [17].

4.6 Technical Factors in CPS Requirements

Cyber Physical Systems development is a highly abstract system that combines the cyber physical, cyber digital and physical elements to enhance business processes [17]. Although technical advancements do not always produce a more secure environment in an organisation, using cyber threat intelligence and security requirement engineering gathering does provide technical knowledge and understanding. Many attack vectors could impact greatly on software security in an organisational setting. Therefore, the need to analyse CPS threats to gain knowledge and an understanding of the nature of cyberattacks, cybercrimes, threat propagation, TTPs, the extent of manipulations and cascading effects may provide more secure means of protecting the software in the cyber digital system than relying on physical (human) factors. Systems security and software developments require a high-level requirements engineering method to elicit information and gather requirements [21]. Although human factors can be uncontrollable, a better understanding of security requirement engineering will assist in risk assessment and mitigation. Cybersecurity requirements engineering indicates that we need to gather functional requirements and non-functional requirements in system development.

4.7 Challenges in Requirements Engineering Activities

Requirements Engineering activities include requirements elicitation, requirements analysis and negotiation, requirements documentation and requirements validation. Digital components in CPS systems such as the Programmable Logic Control (PLC) and Supervisory Control and Data Acquisition (SCADA) systems are extensively employed in organisations to monitor and control operations in smart grid systems and electrical power distributed environments such as power plants, oil and gas pipelines, water distribution systems, sewage treatment plants. For instance, the Aramco power system attack, Ukraine power system attack and Stuxnet attacks [18] [19] [20] provide a clear understanding of how cyberattacks could have impacted on CPS. Activities required in the requirements elicitation includes gathering user needs, domain information, existing information, regulations, and standards. These are the information required to model the

CSC systems to ensure organisational goal and objectives are being met. However, they are the core areas that are vulnerable to cyberattacks as threat actors could use reconnaissance to gather and exploit these spots. Hence, security requirements cannot be an afterthought in the system development process. These are issues that will address systems development processes to ensure systems security and business continuity processes. The requirements elicitation process includes applying the following requirements capturing techniques as Sampling, Questionnaires, Interviews, Research and Observation. The developer must include:

- Sampling: Developer elicits information from sources, samples the legacy systems and software to understand how the systems were developed.
- Questionnaires: Use structured questionnaires to gather user perceptions of system usage.
- Interviews: Developer engages management on one on one basis. Then meets users one on one to review issues and challenges.
- Research: Use literature reviews and expert opinions to gather system development threats and vulnerabilities spots.
- Observation: Developer sits with users as they interact with the systems to observe system functions, information slows and data structures.

The next phase is where the software developer analyses the legacy systems and negotiates with the client as to the classification of requirements, develops a model for the organisation, allocates requirements to components and then negotiates requirements with top management. On the CPS system, the digital component may be vulnerable to malware and ransomware cyber-attacks. A virus, rootkit and botnet attacks may be attached to the programmable logic control PLCs and SCADA systems and will propagate itself on the system. Information security emphasises more on confidentiality, integrity and availability of data and risk assessment uses the attacks to calculate the risk and impacts.

4.8 CPS Supply Inbound and Outbound Chains

The CPS interconnects supplier inbound and outbound chains. The inbound supply chain environment includes organisational systems that are integrated with the CPS systems and that of the third-party supply chain network system. Figure 2 considers the outbound supply chain environment including the organisations that provide distribution services to other organisations, individuals, and third-party vendors in an interconnected environment. The CPS components include cyber digital, cyber physical and the physical elements. It identifies the various activities including the inbound data inputs, logical application process, and outbound information’s using customer management systems (CMS) and the energy management systems HEMS in a smart grid environment.

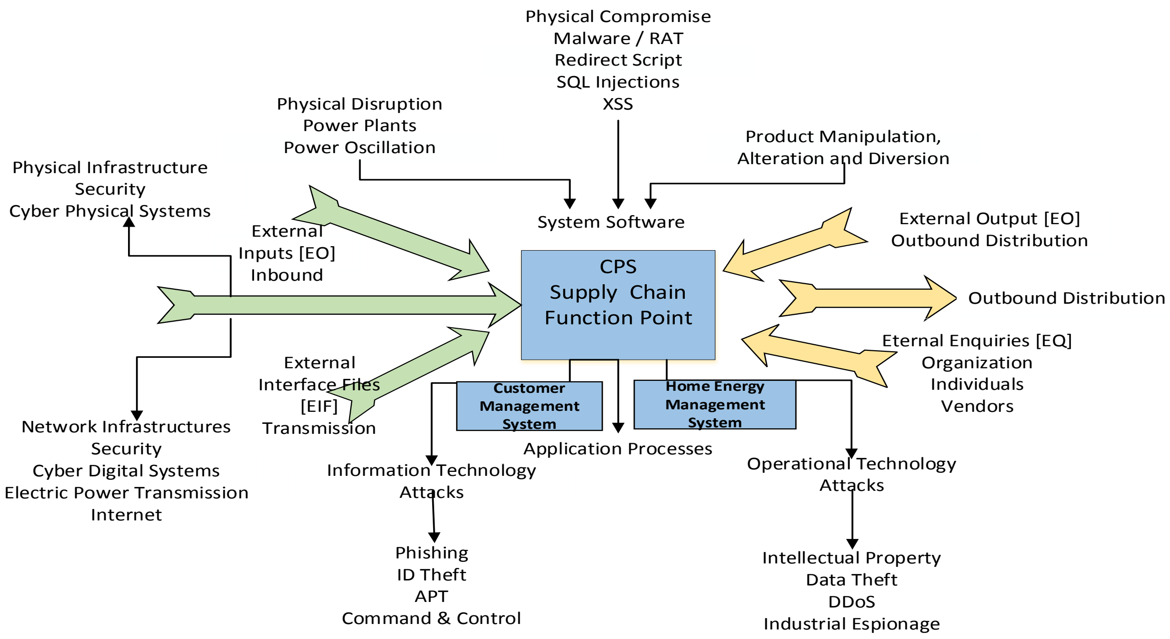

FIGURE 2: CPS Supply Inbound and Outbound Chains.

4.9 CPS Software Security Function Points

CPS software security function points measure the requirements captured and the sum of the software functionality based upon the requirements specifications. Function point (FP) is the subjective measure of software functionality [20]. The nature of the product determines the functionality of the software. Function-oriented metrics are indirect measures of software which focus on functionality and utility [20]. The function point method in software security evaluates the requirements captured, software deliverable and measures its size based on well-defined functional characteristics of the software, the hardware, and physical elements. In the CPS environment, the FP captures and measures the business process requirements and transactions of each stakeholder records such as CMS and HEMS for enquiries. Further, it identifies the varies data groups of each stakeholder file that the software could store and access. The sensitivity of sensed data and the presence of actuation components further increase the requirements of CPS [21].

The CPS security requirements engineering uses concepts of function points [8] [9] external input [EI], eternal out [EO], external interface files [EIF], and external inquiries [EQ] to integrate security in the inbound and outbound chains. The function points are used to integrate the CPS components includes cyber digital, cyber physical and the physical elements in the software design. The rationale is to determine the number of defects per software unit as the basis for quality assurance indicators.

- The CPS takes inbound transaction coming into the system: These are external inputs (EIs) such as logical transaction from external organizations and third parties or system feeds.
- The CPS takes outbound transaction going out of the system: These are external outputs (EOs) such as products, utility bills or external inquiries (EQs) such as online payment systems, monthly reports, or data feeds to other systems.
- The CPS takes internal data that is produced including customer detail, intellectual property, vendor details and stored within the system: These are internal logical files (ILFs) such as logical database groups of user-defined data on the network and system.
- The CPS takes internal data that is interfaced with stakeholders and used to meet process requirement but maintained within a different system: These are external network interfaces

(EIFs), web scripts and constraints such as controls, IDS/IPS, Firewalls that interfaces to other systems.

Figure 3 explains how function points are implemented on external inputs, external outputs, the eternal enquiries and external interfaces files to support the internal logic files and the external interface files on the CPS components. The study used *I** intentionality modelling approach and concepts of secure Tropos security constraints to determine the functions points. The unadjusted function point counts are the core processes used to measure the components as indicated in Figure 3.

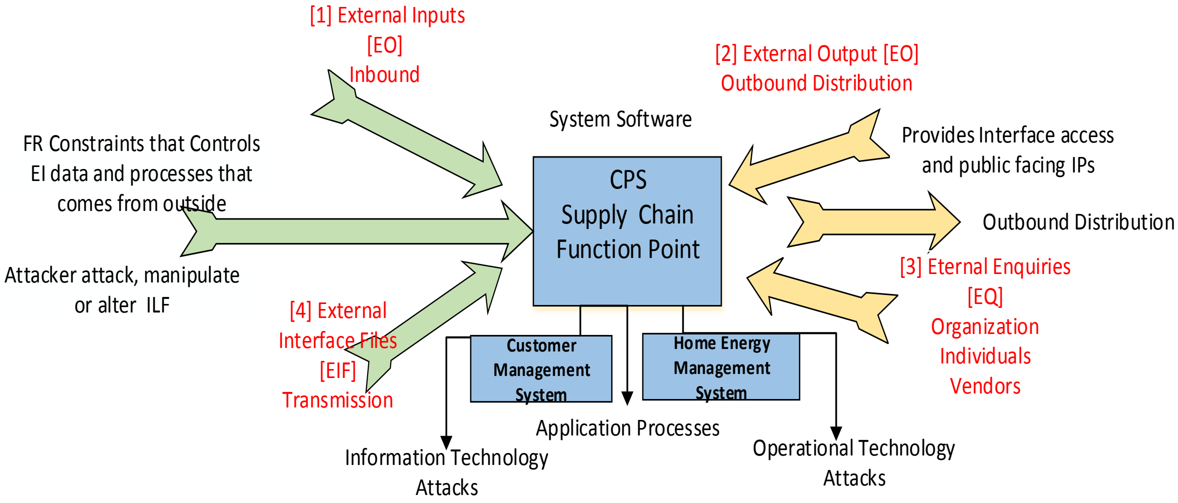

FIGURE 3: Determining the Functional Points.

The external inputs, external outputs, the eternal enquiries and external interfaces are used to support the internal logic files and the external interface files on the CPS five components as explained numbers 1 to 5 as follows:

- **External Input [EI]**
  The external input (EI) is the basic inbound application process (Name, DoB, Address) that processes input data or information that are controlled and comes from external sources of the border of the application of the CMS and HEMS into CPS. The principal intention of an EI by the actor (Actual user or threat actor) is to maintain one or more ILFs and/or to alter the behaviour of the system. For instance, data entered by users or threat actors be determined by the function point. Process data that were not identified by the function point must be identified in file feeds by external applications.
- **External Outputs [EO]**
  The External Output [EO] considers the basic outbound process that transmits data using the public facing IP address to communicate via network nodes or control information outside the application border using the interface in a virtual private network environment. The primary intent of an external output for the outbound chain is to use IPs, and network access to present information to third party vendors through processing logic, and retrieval of information depending on the constraints put on the control data. The processing logic for the deriving processed information must contain a different algorithm that calculates the derived data created as it contains more ILFs. The threat actor may initiate IP spoofing attack, DoS attack on the IP address or use cross site request forgery to attack the website, session hijacking to intercept correspondence and alter the behaviour of the distributed platform. For instance, utility and billing reports may be wrongly generated by the application being counted to generate using the ILF. The

reports generated include derived information that the threat actor may compromise including, ID theft and data theft.

- **External Inquiries [EQ]**
  An external enquiry [EQ] processes data that is coming from external uses such as customers, vendors and the organizations that are information controlled by other application. The purpose of an EQ is to present user data through the retrieval of information being controlled from an internal logic file or external input file. A threat actor could insert a hardcoded password in the system to try to exploit the unchanged defaulted password. The process logic contains no algorithms and generates no derived data for detecting vulnerabilities. Hence, no internal logic file is retained during the inquiry processing. That makes it impossible to detect the actions of the threat actor who may manipulate or alter the system. For instance, information produced by the ILF and EIF application could be intercepted by the threat actor, fabricate it and make it count as authentically information derived from the system.
- **External Interface Files [EIF]**
  External Interface Files [EIF] uses logically related data that are organized into groups from external users and are controlled by referencing it to the system application that is maintained. The main objective to hold the references of the data through the basic processes of the applications within the system that are being counted in the internal logic file. Threat actors (software developer) could exploit the functional requirements and inset a code in the web script in situations where the function pointer is not calculated properly. Therefore, whenever the user accesses the website, they are prompted to clicks on the link and in turn provide access to the threat actor. These attacks are very stealthy and very difficult to detect in software.
- **Internal Logical Files [ILF]**
  The ILF uses logically related data derived from a group of users that are identified as legitimate within the controlled system who may have access to other applications. The main purpose of the ILF is to retain the data that are in process through the various basic input and output processes of the system application being held. For instance, relational database tables, internal user files, business applications and other information that are controlled by user access and privileges. Threat actors could deploy SQL injection attacks to exploit the database and steal personal details, financial records and intellectual property. Functional requirements and constraints are used to control such vulnerabilities. These components are categorised as low, average and high, depending on the data element type (EDT and record element type (RET). The RET provides a non-recursive ILC and EIF field for that are used to analyse the components and the DET provides a subgroup data element type for the ILFs and ELFs component.

4.9.1 Function Point

Function point (FP) is the weighted measure of software functionality [20]. FP is computed in two steps:

a. Calculate the Unadjustable Function Point Count (UFC)
b. Multiply the UFC by a Value Adjustment Factor (VAF). Refer to IFPUG Terminology [9].

The final Adjustable Function Point is calculated as:

$$FP = UFC \times VAF \tag{1}$$

The matrix in Table 1 is used to count the total of all the logical files and transactional functions in the UFC. The UFC is calculated based on the complexity weighting factor. For instance, EI with a low complexity weighting 3 FC, could have an average of 4 FC and high complexity of 6 FC in that order as expressed in Table 1. The RET and DET are determined based on the number of attributes of a table. RET could be set at 0-5 for the number of element and DET could be set at 1-50 or more for the attributes.

Elements of the Components	**Complexity Weighting Factors**		
	Low	**Average**	**High**
External Input [EI]	3	4	6
External Output [EO]	4	5	7
External Inquiries [EQ]	3	4	6
External Interface Files [EIF]	5	7	10
Internal Logic Files [ILF]	7	10	15

TABLE 1: Unadjusted Function Point Count.

4.9.2 Function Point Analysis

Function point analysis method is used for identifying, classifying, and weighting each factor of the FC count for each of the component's business process and data group. It provides a rule set that sizes each function of a software product for the SDLC. The weights are calculated to provide the functional sizes as the unadjusted function point count (UFC) with reference to IFPUG CPM 4.3, ISO/IEC 20926, and ISO/IEC 14143-1standards [9]. Table 2 explains the UFC complexity rating that is associated with each count according to the function point complexity weights using the formula below:

$$UFC = 4_{EI} + 5_{EO} + 4_{EQ} + 7_{EIF} + 10_{ILF} \tag{2}$$

Components	**Complexity Weighting Factors**			
	Low	**Average**	**High**	**Sum**
External Input [EI]	__x3=__	__x4=__	__x6=__	
External Output [EO]	__x4=__	__x5=__	__x7=__	
External Inquiries [EQ]	__x3=__	__x4=__	__x6=__	
External interface Files [EIF]	__x5=__	__x7=__	__x10=__	
Internal Logic Files [ILF]	__x7=__	__x10=__	__x15=__	
Unadjusted Function Point			**=**	

TABLE 2: Unadjusted Function Point Table.

The value adjustment factor (VAF) or technical complexity factors uses the degree of influence (DI) to determine some of the 14 general application characteristics proposed by Albrecht (1979) and other 6 that were proposed to be added to the 14 by Symons (1988). The scale for the DI ranges from 0-5 (Irrelevant to essential) for each component plus the sum of the total scores of the 14 general system characteristics [9] to convert the technical complexity factors using the VAF formula below:

$$VAF = 0.65 + 0.01 \text{ x } DI \tag{3}$$

The VAF varies from 0.65 if all the irrelevant components are set to 0 and also 1.35 of all the relevant are set to 5. The FP analysis technique uses the UFP to assess the unit of measurement for the software product individually or collectively to determine the scope, quality indicators, productivity, and the performance of the software. Hence the FP calculated as FP=UCF x VAF.

4.9.3. Elements Evaluated for VAF/Technical Complexity Assessment

Table 3. explains how the FTR are categorised as file types for transactions for the DET and how the RET are categorised as logical subgroups of data functions for stakeholders. CPS vulnerabilities and threats and included in the categorization of the unit of measure are considered for the software security product individually or collectively to determine VAF.

Complexity	Element	Elements Evaluated for VAF/Technical Complexity Assessment		CPS Vulnerabilities and Threats
Transaction Function Types	[EI]	**File Type Reference:** Files type referenced by a transaction. An FTR must be an Internal Logical File [ILF] or External Logical File [EIF].	**Data Element Types:** A unique user recognizable, a non-recursive field containing dynamic information. Of a DET is recursive, then only the first occurrence of the DET is considered.	Spear phishing attack on input from organizations, third party vendors, and individual customers
	[EO]			Initiate IP spoofing, Man-in-Middle attack on ILF and reports that could go to stakeholders
	[EQ]			Insert hardcoded password to exploit default password, APT and C&C
Data Function Types	[EIF]	**Record Element Type:** User recognizable logical subgroups of data element Stakeholders.		Exploit the functional requirements by inserting a code in the web script or XSS, Session Jacking attack.
	[ILF]			Deploy SQL injection attacks to exploit the DB

TABLE 3: Complexity Assessment of the Function Types.

4.9.4 CPS Software Security for the FPA Enhancement

Further to Table 3, we consider the CSC software security for the analysis of the function points, the elements evaluated for the VAT/Technical complexity assessment and included the CPS vulnerabilities and threats. The paper extends the general system characteristics to include software security estimations to enhance the FPA. To achieve the applicability of the FP, we adopt equation (3) to improve and calibrate the estimation as follows. From the analysis, each component is rated from 0 to 0.5, where 0 means the component is not relevant to the system and 5 means the component is essential. The VAF can be calculated as:

$$VAF = 0.65 + 0.01 \sum_{i=1}^{n-14} F_i \binom{DI}{security} M_i /(100) \tag{4}$$

Where

F_i = indicates the degree of influence for each GSCs
i = the number representing the 14 GSCs used for the VAF
Σ = the summation of the characteristics used to influence the VAF
DI = indicates the degree of Influence of the vulnerabilities
Security = Vulnerability characteristics that may inform the degree of influence on the system
M_i = Minimum and maximum of vulnerability characteristics for (Total Degree of Influence) / (100)

The results show the influence of the 14 GSCs ranging from 0-5 with the standards equations and the results of the enhanced VAF estimation as:

FP = UFP * VAF
UFC = $4_{EI} + 5_{EO} + 4_{EQ} + 7_{EIF} + 10_{ILF}$
VAF = 0.65 + 0.01 (Security) / (100)

Given the DI and the security characteristics that influence the threats and vulnerabilities in table 3, all the stakeholders may agree on the number of characters required during requirement specification and design phase of the SDLC. Further, the nature of cyberattacks that could be deployed on the CPS will also indicate the various scores for the degree of influence required for the VAF as a result.

5. DISCUSSIONS

Software reliability and quality assurance challenges span from software cost estimation, integration of function point counts, quality assurance and risk assessment. These factors determine the extent of security requirements that will be incorporated into the SDLC as well as the technical complexity assessment and security reference model. The software security reference modelling activities consider concepts such as security features of the system under development and the protection objectives. Further, the security constraints modelling considers the constraints imposed on the actors and systems that allow developers to perform analysis by introducing the relationship between a security constraint and its context. Secure entities modelling provides an analysis of the system, and it is considered complementary to the security constraints modelling. Secure capabilities modelling includes the identification of the actors, their intents and security that guarantee the satisfaction of the security constraints. CSP software security requirements capturing are oriented towards I* approach as it captures both the actor and goal modelling to answer the questions of who and why. CSC requirements capturing and analysis supports who should be integrated on the supply chain. The integration of organizational business processes including electronic commerce transactions, financial transactions and mobile device communications answers the questions of why. In essence, all the methodologies meet certain a specific requirement. The Tropos approach answer the question of what a system should be, I* approach is oriented towards modelling the intentionality of actors to answer the questions of who and why, whereas KOAS approach illustrates the challenge to be solved in line with the constraints and places less emphasis on the intentionality of actors. Further, Secure Tropos methodology provides an extension of Tropos and uses conceptual models and reason about security from a specific system context.

5.1 Quality Assurance in Software Security Reliability

Quality assurance in software security reliability uses a systematic approach to identify and integrate security in product development and delivery channels to ensure the product is secure and meet the specified quality required. Quality assurance in software security development systems emphasizes detecting bugs, errors, and defects before they get into the final product. Software Reliability is a significant factor in affecting system security, reliability, and resilience. Software security reliability reflects the design perfection of CPS components as it consists of cyber digital, cyber physical and physical elements. ISO 8402-1986 standard defines quality as: the totality of features and characteristics of a product or service that bears its ability to satisfy stated or implied needs. A measure of excellence or a state of being free from defects, deficiencies, and significant variations. Quality is brought about by strict and consistent commitment to certain standards that achieve uniformity of a product in order to satisfy specific customer or user requirements.

Having technical knowledge and an understanding of the intent and mindset of the threat actors are relevant in SDLC is very important. An understanding about systems security, threat landscape and business processes as part of software development will assist in identifying CPD risks, threats and vulnerability in the early requirements capturing phases. That provides a more secure CPS software than just depending on using risk assessments and controls that emphasise on humans, attacks and impacts. For this purpose, the study identifies challenges in software Requirements Engineering Activities, critical functional and non-functional requirements factors that may impact the CPSs.

The CPS smart grid integrates three entities: Cyber Physical, Cyber Digital and the Physical components that leverage organisational goals, actors, assets, business requirements and supply chain processes in the electric power distribution system. Power Threats could be identifying on the physical plant, cyber digital and cyber physical components. The physical power plants integrate with the cyber digital and provide electric power generation, transmission and distribution to individual homes, industries and organisations. The cyber digital and the cyber physical ingrates to provide intelligence components at the organisational and industrial levels global load balancing, smart generations, and energy savings [21]. Both the cyber digital and physical system components can be attacked by adversaries using malware, SQL injection, APT, and command and control attacks on the infrastructure. The adversary could change configurations remotely on the AMI headend control meter reading, and they may send incorrect utility readings, and they may send commands to connect and disconnect so that such actions can lead to immediate electricity changes. The cyber physical components used for the automation of electric power generation, transmission, and distribution processes. Attacks can take place resulting from their close links with the MDMS and the smart meters that control the data from the third-party vendors. Adversaries can use spyware or malware on the supply chain system to breach the source code and to manipulate the CPS smart grid software components remotely. These interact with smart meters, organizations, and vendors on the mechanism of the distribution. Home appliances that are directly linked to smart meters may be compromised as they are associated through a cyber physical environment.

5.2 Comparing Results with Related Works

There are several existing literatures on the subject of software reliability and function point analysis methods used to calibrate the VAF. For instance, Albrecht (1979) proposed a value adjusted factor also called technical factors that are used as a weighted sum of 14 GSC components in software requirements [8]. Symons (1988) postulates in his analysis of function points [Mark 11] that the technical factors should include security and other components to add up to 20 components [9]. Abdullah et al 2009, proposed a software security characteristic for function point analysis that includes security in the VAF to influence the security characteristics [10].]. Mukherjee et al. 2013, surveyed metrics, model, and tools for determining the relationships among scores of discrete parameters that can affect the outcomes of software projects [11]. Dhakad and Rajawat 2016, proposed an expert user programming for estimating FPA accurately for software size using the GSC [12]. Alves et al 2014, used a survey to determine the size and complexity of software applications with FPA for a web applications development [13].. However, comparing our work with the other works, none of the works considered the function point analysis from cyber physical system software security perspective. Further, none of the works considered the Tropos, *I**, KAOS, and Secure Tropos methodologies for SDLC perspective. The paper has highlighted the security challenges when implementing the methodologies in a CPS environment.

6. CONCLUSION

There is no one size fits all approach to secure software development in the CPS environment due to the changing organization goal, business process, the distribution platform, and the evolving threat landscape. The paper has reviewed existing security requirements engineering concepts and methodologies such as TROPOS, I*, KAOS and Secure Tropos to determine their security relevance software development. The study identified the functional requirements of the system used function points to integrate the constraints in the inbound and outbound chains to improve the reliability of the software in CPS security. The function points determine the subjective measure of software functionality. Hence, the external input, external output, the internal logic files and external interfaces may be relative to the system under development. The results show that software security threats in CPS can be addressed by integrating the SRE approach and function point analysis method in the SDLC to improve the reliability of the software. As software size increases, so are the vulnerabilities and the threat landscape. Future works will consider how software size impacts on cyber supply chain design vulnerabilities, security cost estimations and its implications on cybersecurity.

7. REFERENCES

[1] R. Alavi, S. Islam, D, Jahankhani, H. and A. Al-Nemrat. “Analyzing Human Factors for an Effective Information Security Management System”. International Journal of Secure Software Engineering (IJSSE), 4, 50-74. 2013.

[2] CAPEC-437: “Supply Chain. Common Attack Pattern Enumeration and Classification: Domain of Attack”. https://capec.mitre.org/data/definitions/437.html MITRE. [Assessed on 05/04/2020]

[3] M. Pavlidis. S. Islam. And H. Mouratidis. “A CASE Tool to Support Automated Modelling and Analysis of Security Requirements, Based on Secure Tropos”. 2012.

[4] P. Giorgini, M. Kolp, J. Mylopoulos, M. Pistore. “The Tropos Methodology”. doi: 10.1007/1-4020-8058-1_7. 2004.

[5] E. S. Yu. “Social Modeling and I*.” Faculty of Information, Springer. University of Toronto. http://www.cs.toronto.edu/pub/eric/JMfest09-EY.pdf. [Accessed on 07/03/2020]

[6] Respect-IT. “A KOAS Tutorial”. V1.0. 2007. http://www.objectiver.com/fileadmin/download/documents/KaosTutorial.pdf. [Accesse 14/03/2020]

[7] E. Lee. “Concept Map for Cyber Physical Systems”. http://CyberPhysicalSystems.org/CPSConceptMap.xml. 2012. [Accessed on 14/12/2019]

[8] A. J. Albrecht, “Measuring Application Development Productivity”. Proceedings of the Joint SHARE, GUIDE, and IBM Application Development Symposium, Monterey, California, October 14–17, 1979. IBM Corporation (1979), pp. 83–92.

[9] C. R. Symons, "Function point analysis: difficulties and improvements," in IEEE Xplore Transactions on Software Engineering, vol. 14, no. 1, pp. 2-11, Jan. 1988. DOI: 10.1109/32.4618.

[10] N. A. S. Abdullah, S. Abdullah, M. H. Selamat, A. Jaafar. “Software Security Characteristics for Function Point Analysis” IEEE Conference. Industrial. (2010). DOI: 10.1109/IEEM.2009.5373328

[11] S. Mukherjee, B. Bhattacharya, S. Mandal. “A Survey on Metrics, Models, & Tools of Software Cost Estimation” International Journal of Advanced Research in Computer Engineering & Technology. Vol. 2, Issue 9. 2013.

[12] A. Dhakad, A. S.Rajawat. “A Novel Variant of Function Point Analysis for Accurate Software Size Estimation”. International Journal of Engineering & Scientific Research. Vol 4. Issue 2. 2016.

[13] L. M. Alves, S. Oliveira, P. Ribeiro, R. J. Machado. “An Empirical Study on the Estimation of Size and Complexity of Software Applications with Function Points Analysis” IEEE International Conference on Computational Science and Its Application. 2014. DOI 10.1109/ICCSA.2014.17.

[14] N. A. S. Abdullah, S. Abdullah, M. H. Selamat, A. Jaafar. “Potential Security factors in Software Cost Estimation. IEEE International Symposium on Information Technology 2008. DOI. 10.1109/ITSIM.2008.4631983

[15] A. Yeboah-Ofori, J. D. Abdulai, and F. Katsriku, “Cybercrime and Risk for Cyber Physical Systems: A Review”. IJCSDF. Vol. 8 No.1. Pg. 43-57. 2018. http://dx.doi.org/10.17781/P002556.

[16] A. Susi, A. Perini, and A. Mylopoulos. “The Tropos Methodology and its Use”. Informatica 29. 401-408. 2005.

[17] A. Yeboah-Ofori, and S. Islam, “Cyber Security Threat Modeling for Supply Chain Organizational Environments.” Future Internet, 2019. 11, 63, 2019. doi: 10.3390/611030063

[18] S. Khou, L. O. Mailloux, J. M. Pecarina, and M. Mcevilley. “A Customizable Framework for Prioritizing Systems Security Engineering Processes, Activities, and Tasks”. IEEE Access, Vol.5, pp. 12878- 12894, 2017. doi: 10.1109/ACCESS.2017.2714979.

[19] M. Chenine, J. Ullberg, and G. Ericsson. “A Framework for Wide-Area Monitoring and Control Systems Interoperability and Cybersecurity Analysis”. IEEE Transactions on Power Delivery, 29(2), pp. 633-641. 2014. doi: 10.1109/TPWRD.2013.2279182.

[20] C. Sun, A. Hahn, and C Liu. ”Cyber Security of a Power Grid: State of the Art”. International Journal of Electrical Power and Energy System, 99, Pp. 45-56. 2018..

[21] M. Al Faruque, F. Regazzoni, and M. Pajic. “Design Methodologies for Securing Cyber-Physical Systems”. 2010. doi: 10.1109/CODESISSS.2015.7331365.