World Journal of Advanced Engineering Technology and Sciences

eISSN: 2582-8266

Cross Ref DOI: 10.30574/wjaets

Journal homepage: https://wjaets.com/

(Review Article)

Software security models and frameworks: an overview and current trends

Fridah C. Korir *

Jomo Kenyatta University of Agriculture and Technology, Juja, Kenya.

World Journal of Advanced Engineering Technology and Sciences, 2023, 08(02), 086–109

Publication history: Received on 03 February 2023; revised on 12 March 2023; accepted on 15 March 2023

Article DOI: https://doi.org/10.30574/wjaets.2023.8.2.0078

Abstract

The continued dependence on information technology applications has led to the adoption of electronic channels and software applications to support businesses, online transactions and communications. In this perspective, both functional and non-functional requirements are critical for the provision of necessary needs at the early phases of the software development process. This is specifically important in the requirement phase of the software development process. Software security is increasingly becoming a necessity concern in this environment. Unfortunately, security concepts such as access control requirements are mostly considered after the functional requirement definition stage. In addition, majority of the developers and organizations regard security as an activity that can be incorporated after the development of a system. This leads to flaws and security defects in the access control mechanism. Therefore, software security issues must be given higher priority in the early stages of the development process. The aim of this paper is to offer a survey of the software security techniques, frameworks and models that have been developed to deal with these issues. The results obtained indicate that software vulnerabilities have made it possible for attackers to exploit them to cause havoc in computerized systems and has become one of the most critical risks facing modern corporate networks. Since most of these vulnerabilities involve exploitable weaknesses introduced through badly written code, the cyber security community has tried to come up with techniques to enhance software products.

Keywords: Frameworks; Software; Attacks; Security; Access control; Models

1. Introduction

Software security entails the engineering of the software such that it can continue to operate correctly even when it is under malicious attack [1]. It is normally associated with the protection of data and products, adversarial skills and resources as well as the costs require for potential assurance remedies [1], [2]. As explained in [3], the software vulnerabilities continue to be the major security threats to software intensive systems. For instance, the authors in [4] point out that privacy and security threats are always major concerns for e-government applications. These security and privacy issues manifest themselves in terms of cyberspace identity thefts and privacy violations. Therefore, in the absence of proper protection, these applications become targets of numerous security attacks which can compromise them at any time, leading to different types of financial, psychological and personal damages. The authors in [5] explain that software security engineering offers the means to define, implement and verify security [6] in software products. It is executed through following a well defined software security development life cycle model or a security capability maturity model. Here, secure software development involves the execution of a set of security engineering activities in conjunction with software development processes [7].

As explained in [8], software security issues must be given higher priority in the early stages of the development process. Unfortunately, majority of the developers and organizations regard security as an activity that can be incorporated after the development of a system [8]. Regarding agile techniques for software development and security, these two elements are present in a software project, the latter should not be compromised by the strive for agile principles and lightweight

*Corresponding author: Fridah C. Korir

processes. Conversely, security procedures must not reduce the effectiveness of the development process. The continued dependence on information technology applications has facilitated the adoption of electronic channels and software applications to support businesses, online transactions and communications. As such, software security is increasingly becoming a necessity concern in this environment [9]. In this perspective, both functional and non-functional requirements are crucial for the provision of necessary needs at the early phases of the software development process. This is specifically important in the requirement phase of the software development process. The contributions of this paper are summarized as follows:

- An extensive analysis of the security issues in software systems is provided
- An elaborate discussion on the various techniques for security assurance in software systems is given
- Various security models and frameworks are described, including the various Software patch management techniques
- Numerous challenges encountered in software security are identified and discussed in detail

The rest of this paper is structured as follows: Section 2 describes significance of software security while Section 3 discusses the various techniques geared towards software security. On the other hand, Section 4 illustrates on the current security models and frameworks while Section 5 discusses software patch management approaches. This is followed by the identification of the challenges in implementing software security in Section 6 while the research gaps and recommendations are detailed in Section 7. Finally, Section 8 concludes the paper and offers some insights into future research work.

2. Significance of software security

Software vulnerabilities have made it possible for attackers to exploit them to cause havoc in computerized systems and has become one of the most critical risks facing modern corporate networks. Since most of these vulnerabilities involve exploitable weaknesses introduced through badly written code [10], the cyber security community has tried to come up with techniques to enhance software products. On the other hand, authors in [11] explain that security incidents and threats are essential issues that determine the cost of creating a software products as well as maintaining them throughout their lifetime. This process entails the creation of security architecture and design, definition of security objectives, elicitation of requirements, and secure programming practices. As explained in [12], software security is an important quality, which has been defined by the ISO/IEC 25010 standard. For instance, authors in [13] have noted that security [14] an essential requirement in most software systems. Considering e-government applications, any weaknesses can have devastating effects [4].

Considering the software development lifecycle, security should be in corporate in the early stages of this process so as to boost efficiency and effectiveness [8]. Failure to do so leads to numerous security issues [15] which can make organizations to spend large amounts of money to buy antivirus and firewall software to protect their systems. Therefore, there has to be some software security measurements [16] that quantify the security of the system directly from its design. The identification of these software security metrics is crucial as it serves to reduce the security risks and weaknesses of the system [17], [18]. Basically, software security engineering aims to address the identified software security risks already in the development phase, which then facilitate efficient creation of effective security solutions [19]. This may involve the identification, mitigation and avoidance of security threats to software and data assets. To attain availability, integrity, and confidentiality of information, security policies are implemented into software security features during software development [20]. The verification of security implementation is normally through appropriate security assurance [21], which is produced and gathered along the development, testing and operations.

One common feature of security faults, such as implementation bugs and design flaws, is their persistence. Failure to address these faults requires constant and costly security engineering activities throughout the software's operational lifetime. Essentially, the elimination of these faults early on requires significantly less resources [22] than having to fix the issues later in the development process [23], [24]. Considering other issues such as problems caused by users, hardware errors, and other issues in the operating environment, their transient nature imply that they be mitigated by means independent of the assets protected. Failure to secure software has led to many cyber attacks that have caused devastating consequences, resulting in huge financial and reputation losses from breach of confidentiality and integrity of company data [26], [27]. Such cyber attacks include the Equifax case, and even human deaths due to the unavailability of software systems.

The authors in [28] have pointed out that many organizations are becoming heavily dependent on information processing systems. This dependency has brought about the need for protecting software systems from threats [29]. Therefore, software security has become an essential issue in modern organizations [30]-[32]. Basically, system security assurance offers confidence that practices, security features, procedures and architecture of software systems mediate and enforce the security policy and are resilient against security failure and attacks [33], [34]. As explained in [35], security assurance entails the demonstration with evidence that a system fulfils established standard security criteria. It offers some confidence that a system meets the security requirements. However, based on the NIST definition, security assurance is the measure of confidence that the security features, practices, procedures, and architecture of an information system accurately mediates and enforces the security policy [36]. On their part, authors in [37] have defined security assurance as the confidence that a system meets its security requirements and is resilient against security vulnerabilities and failures. Here, confidence denotes the level of trust of a system that is safe to use. This means that it has less vulnerability, resulting in the overall reduction of risk in using the related system. The security assurance process and its vital components are depicted in Figure 1below.

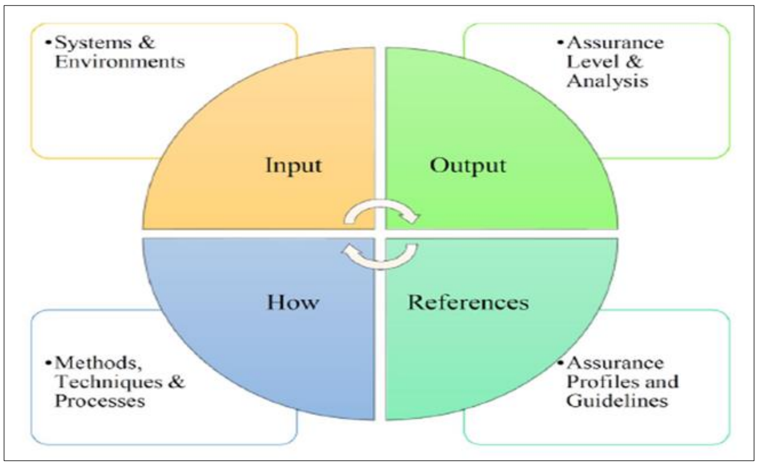
•Systems & Environments
•Assurance Level & Analysis
Input
Output
How
References
•Methods, Techniques & Processes
•Assurance Profiles and Guidelines

Figure 1 Security assurance process [34]

As pointed out in [38], assurance activities are important during the operations and maintenance phase of the software product. This ensures that the assurance level of a system to which it is certified, is maintained. However, the security requirements specified for a system may be violated in the operational phase because of improper implementation of the security measures [39], hazardous environment, or invalidity of the assumptions under which the security requirements were specified. According to [37], security assurance process of a system requires a set of inputs, which include assessment criteria and requirements (assurance profile), target of evaluation (TOE), the operational environment, assurance methods and assurance level. As shown in Figure 2, there are different types of security assurance.

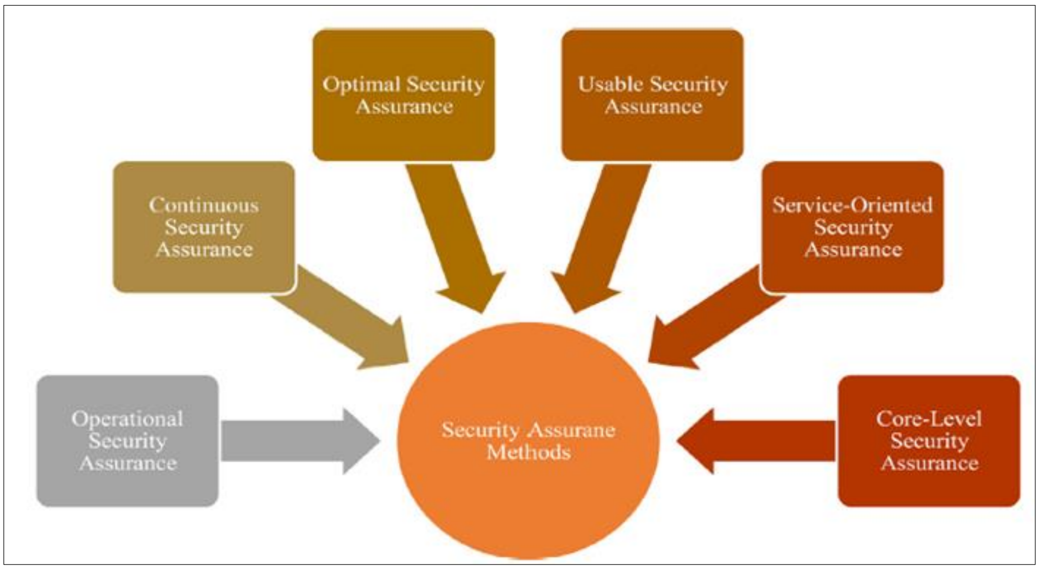
Optimal Security Assurance
Usable Security Assurance
Continuous Security Assurance
Service-Oriented Security Assurance
Operational Security Assurance
Security Assurane Methods
Core-Level Security Assurance

Figure 2 Types of security assurance [34]

The authors in [40] explain that continuous security assurance is a potential means of managing the risks encountered during continuous monitoring, compliance, and security. Based on the evidence collected, continuous security assurance can demonstrate if the security requirements are met throughout system operation. Vulnerabilities have been noted in [41] to affect the system's confidentiality, integrity, and availability. As such, they can cause severe damage to an organization. To mitigate the effects of these vulnerabilities, numerous techniques have been developed for security requirements elicitation, tracking, analysis, correctness and modeling as shown in Figure 3.

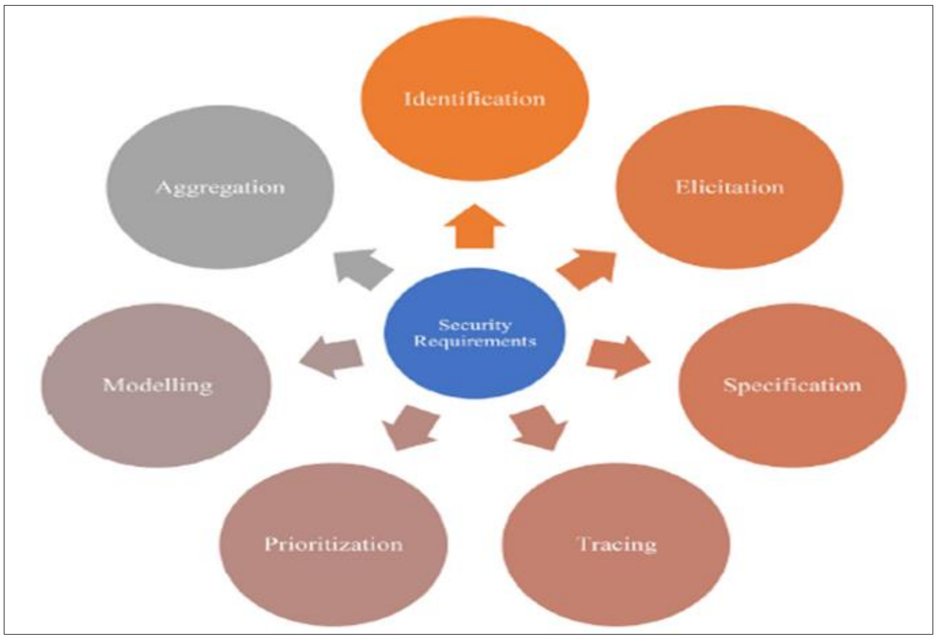

Figure 3 Security requirement elicitation techniques [34]

3. Techniques geared towards software security

Conventionally, approaches to mitigate and manage security are executed by following relatively simple checklists to guide design and implementation practices. However, software security engineering offers more systematic means to address security risks in software development. For instance, software dependability is normally achieved through fault prevention, fault tolerance, fault removal and fault forecasting [42]. In addition, other security verification methods such as security documentation, testing, reviews, and audits may be used [43]. The goal of these techniques is to detect flaws and errors, and to offer some appropriate levels of assurance. Patching is one of the most effective strategies for protecting software systems against such cyber attacks. In this regard, software security patch management involve the application of patches to the security vulnerabilities [44], [45] present in the software products and systems. The procedures included here are the identification of existing vulnerabilities in managed software systems, acquisition, testing, installation and verification of software security patches [46], [47], [48], [49].

After the installation of the software product, point of conjecture (PoC) is utilized to hypothesize security issues which may be faced by the software product in the future as shown in Figure 4. Therefore, the PoC must be identified prior to the implementation phase. The significance of PoCs arise from their importance during security requirements elicitation. Therefore, PoCs need to be validated after the development of the software product. To keep track of existing vulnerabilities in the developed software products, vulnerability databases need to be maintained [50]. These databases serve to record structured [51] instances of vulnerabilities together with their potential consequences. For instance, the Center for Assured Software (CAS) has developed a benchmark test suite with good code and faulted code across different languages. The aim here is offer the evaluation of static analysis tools performance. Similarly, authors in [51] have investigated different SAST tools deployed for the detection of a class of Common Weakness Enumeration (CWE) using the Juliet test suite. To enhance the internal structure of a software system, refactoring has been deployed. This involves the alteration of the software system in such a manner that it does not alter the external behavior of the code but improves its internal structure [53]. It is an effective tool used to enhance the quality of software, such as maintainability, testability, and understandability. In so doing, refactoring can potentially minimize maintenance activities and costs [54], [55]. As explained in [56] and [57], this technique serves to enhance improve the quality of existing software.

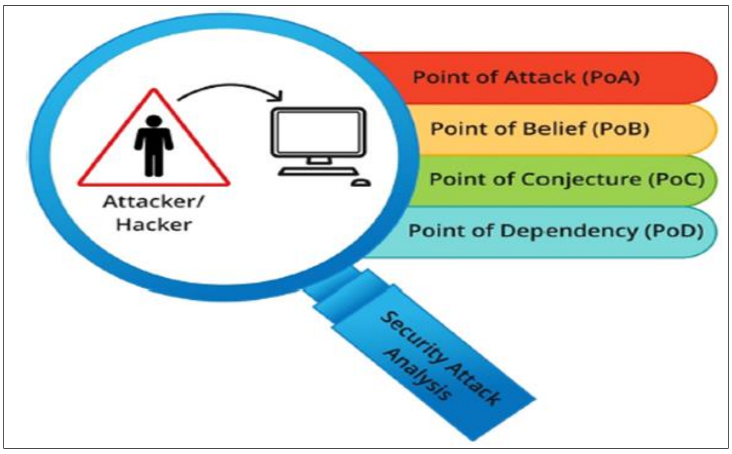

Figure 4 Security analysis PoCs

To detect and address security vulnerabilities [58] during the design phase, authors in [59] have developed security metrics for the object-oriented design. These metrics have been shown to help in the comparison of the security of the different design versions. As such, they have been widely utilized in contemporary literature and practice [54] to measure security in the early stages of software development. In so doing, they reduce the costs paid in order to secure the system at later stages. Information hiding is one of the most important features of object-orientation [60] that can protect information from exposure or loss by preventing unauthorized access [12]. To address some of the weaknesses of the agile methods [61], Extreme Programming (XP) has been proposed to offer security assurance. This offers some guidelines on the creation and gathering of assurance at requirements, design, implementation, and testing phases. According to [62], elicitation of security requirements, whether through misuse cases or by other means, is also possible. On the other hand, a formal analysis and design for engineering security (FADES) has been proposed in [63] as the first goal-oriented software security [64] engineering approach, while Discovering Goals for Security (DIGS) framework has been developed in [65] to model the key entities in information security such as assets and security goals.

To elicit security requirements, authors in [66] have introduced a Model oriented framework to Security Requirement Engineering (MOSRE) framework. This model uses case diagrams and has been applied to E-Health web applications. It has been shown to facilitate the identification, quantification and ranking of the risks associated with various security threats and vulnerabilities. Similarly, the authors in [67] have proposed a methodology based on problem frames and abuse frames for security requirement elicitation. On the other hand, an approach for modeling and reasoning about security requirements has been developed in [68], while a framework that integrates three effective security requirements elicitation techniques (threat modeling [69], misuse case and attack pattern) has been introduced in [70] for the elicitation of security requirements. On their part, authors in [71] have developed the Requirements Engineering Readiness Model (SRERM) to permit organizations to measure their security requirements engineering (SRE) readiness levels, while the security requirements framework has been developed in [72] to overcome the issue of security requirements elicitation for heterogeneous components. Similarly, a system for software assurance is developed in [73] based on the critical security flaws [74] and vulnerabilities related to software installation and software execution.

A taxonomy that provides a comprehensive framework for identifying and analyzing security requirements and potential attacks is introduced in [75]. This taxonomy is based on the survey of various hierarchically ordered and adjacent sciences, notations, and security requirements analyses which are essential for extensive communication security. Similarly, security technique for a class of attacks in open multi-agent systems is introduced in [76]. This is based on the extensive review of security techniques in literature. On their part, the authors in [77] have conducted literature review that enabled them identify the challenges and issues in the development of secure software using the agile approach. On the other hand, a vulnerability analysis test-bed using Linux containers has been developed in [78], which is shown to be portable and easy to deploy. To evaluate the systems security assurance, a risk-based security assurance metric and aggregation technique is developed in [79], while a set of metrics to evaluate the security assurance [80] of the runtime systems is developed in [81]. Based on the security properties of the individual system components, component security assurance methods are introduced in [82], while a quantitative vulnerability detection technique at the source code level is developed in [83] based on code clone detection technique.

Focusing on the optimization of security assurance effort for a specific lifespan of a web application by estimating its security durability, authors in [84] have identified three factors (trustworthiness, dependability, and human trust) that can improve security durability. On the other hand, authors in [85] have developed an anomaly detection component through network traffic monitoring, while a context-based language to express the assurance properties [86] has been proposed in [87] based on the security requirements. On the other hand, authors in [88] have developed a security assessment methodology for security certification. This technique is shown to offer continuous security assessment to protect the cloud provider's intellectual property.

4. Security models and frameworks

Various security models have been developed to express the security status of the system [89]. Normally, the data collected during a system's operation is utilized in expressing the system's current state and validate the security metrics model. In this regard, vulnerability prediction models play critical roles in the identification of the location of source code that require more attention [41], [90]. For instance, authors in [91] have used machine learning [92] techniques to predict the vulnerability of the software from source code before its release. Similarly, a security assurance framework is developed in [93] for connected vehicular technology. On the other hand, an object-oriented security evaluation model has been introduced in [94], while a model-driven approach for cyber range training is proposed in [95]. This model is shown to generate tailor-made training scenarios depending on the organization's requirements and security posture. On their part, authors in [96] have developed a framework to address challenges regarding security transparency of the cloud. In addition, they have introduced a security [97] transparency and audit tool which can help auditors in the evaluation of the evidence produced by the cloud service providers (CSPs).

According to [98], some of the software security maturity models include Open Web Application Security Project (OWASP)'s Software Assurance Maturity Model (OpenSAMM), Building Security In Maturity Model (BSIMM) and BSIMM for vendors (vBSIMM). However, these security-specific maturity models and their checklists are not adequate in all contexts [99]. On the other hand, authors in [37] have presented a quantitative security assurance framework that incorporates both vulnerabilities and security assurance. On the other hand, a framework for conceptualizing security related behavior has been developed in [100]. However, the authors focused on security-related behaviour [101]-[105] and never considered other security perspectives. On their part, the authors in [106] have employed Fuzzy Analytical Hierarchy Process methodology to evaluate useable security [107], [108]. Using a quantitative approach, they were able to assess the impact of security on usability and the impact of usability on security. A security assurance development process (SADP) model has been introduced in [109]. This model comprises of institutional security awareness programme, documentation of security-relevant requirements, application of security principles to design, performing source code level review, implementing and performing security tests, security review as well as risk management. Other Software Security Development Life Cycle (SSDLC) models that have been developed include Microsoft Security Development Lifecycle (SDL) model and Touchpoints for Software Security[110]-[114]. The latter forms the development life cycle model included in BSIMM. To comply with the changing software development practices, Microsoft made an effort to adopt the SDL for agile development. However, the references to agile development have since been removed from the current version of the SDL, and the life cycle model itself integrated into DevOps processes and tools used in Microsoft's public cloud service. On the other hand, a framework to check the authenticity [115] of the software application before its installation is introduced in [116]. Unfortunately, the authors consider only a single aspect, such as a specific operating system or a single-entry point check. A tool for visualization and modeling the hierarchical specification and deployment of security metrics and measurements is presented in [117]. This tool is shown to support security decisions making by managing a large number of metrics and measurements in an efficient way.

The international standard for security development framework has been created by the ISO/IEC [118]-[121]. This comes in the form of Secure Software Engineering Capability Maturity Model (SSE-CMM) [122] which consists of the best practices for security engineering. It basically formalizes security [123] work into an exhaustive set of security processes, by defining 129 security processes divided into 22 process areas. To supplement this high-level standard, multiple international, national and domain-specific security regulations have been crafted [24], [124]. On the other hand, authors in [125] have developed a framework to assist designers, testers, and analysts during the Common Criteria (CC) certification process [126]-[128]. This framework is shown to be model-driven and can be used to analyze, design and evaluate the security properties [129] of information systems. Similarly, a continuous assurance methods for IoT services is developed in [40] which can help in IoT security assurance assessment. In Finland, a comprehensive set of information security instructions has been issued by the Government Information Security Management Board. These instructions are abbreviated as VAHTI model [130]-133] which contains a specific SSDLC model, which is created after the SDL and Touchpoints. It is based on the CC, the ISO standard for software product evaluation and appears to conform with the ISO application security standard. Therefore, VAHTI can be considered to represent the state of the art of software security [134] engineering practices in Finland. To comply with European and North American security regulations, KATAKRI [135]-[139], another Finnish security framework has been developed. Based on the study of the state-of-the-art of security issues [140], challenges, and practices during SDLC in the industries, a software security assurance model is introduced in [141]. This model can help software developers measure their readiness to develop secure software. Similarly, a model-driven framework is developed in [142]. This framework can assist designers to model the security concerns of REST compliant open-source software and to automate software verification and validation process. Similarly, a framework developed in [143] can help show whether the access control requirements that were identified and designed have been correctly implemented at the code level.

The authors in [144] have pointed out that SAFECode is the most notable industrial software security framework outside the SSDLCs and maturity models. This model is based on a rigorous security risk [145] management process and addresses the identified risks through secure design and coding practices, coupled with strict management of third party software components. Here, the security incident response function is utilized to cover the operational part of a software life cycle. Basically, SAFECode [146] supplements security maturity models by offering concrete and actionable instructions for the deployment of security processes. In addition, it contains the essential principles guiding the build-up of an SSDLC-enhanced software development process. On the other hand, a framework that utilizes assurance case methodology for Industrial IoT systems (IIoT) is presented in [147].

5. Software patch management

Software security patch management is a security practice that is designed to proactively prevent the exploitation of security vulnerabilities that exist within the deployed software products and systems. It supports the process of patching known software security vulnerabilities. As pointed out in [148], patching security vulnerabilities in large and complex systems is a challenging process. It involves multiple stakeholders making several interdependent technological and socio-technical decisions. Generally, software security patches are in most cases prioritized over non-security patches by industry practitioners and researchers [149]. This is because these patches are aimed at mitigating software vulnerabilities or security bugs that present exploitable opportunities for malicious entities to gain access to systems. In addition, software security patches are acknowledged to be the most effective strategy to mitigate software vulnerabilities [150]. Therefore, successful software security patch management process is necessary and critical to sustaining the confidentiality, integrity, and availability [151] of IT systems. Figure 1(a) depicts the focal point of software security patch management from a typical software vulnerability life cycle viewpoint. Here, the focal point relates to the process of company X applying security patches to its installed third-party software after the patches are released by the corresponding third-party software vendors (company Y).

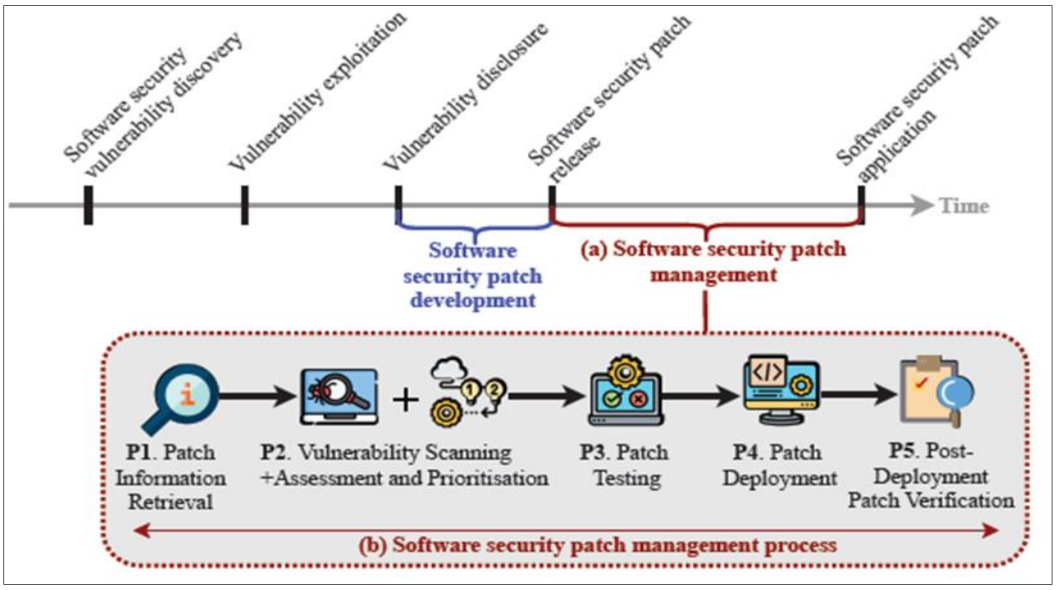
Software security vulnerability discovery
Vulnerability exploitation
Vulnerability disclosure
Software security patch release
Software security patch application
Time
Software security patch development
(a) Software security patch management
P1. Patch Information Retrieval
P2. Vulnerability Scanning +Assessment and Prioritisation
P3. Patch Testing
P4. Patch Deployment
P5. Post-Deployment Patch Verification
(b) Software security patch management process

Figure 5 (a) Focal point for security path management. (b) Software security patch management process [148]

As shown in Figure 1(b), there are five main phases of the software security patch management process [47], [48]. The first phase is the patch information retrieval phase, in which practitioners learn about new patches and acquire them from third-party software vendors such as Microsoft. The next phase is vulnerability scanning, assessment and prioritization, in which the practitioners scan the managed software systems for the newly disclosed vulnerabilities. The goal is to identify the applicability of patches in their organizational context, assess the risk, and correspondingly priorities the patching decisions [152]. The thirst phase is patch testing phase, in which the patches are tested for accuracy and stability. It is also prepared for installation by changing machine configurations, resolving patch dependencies and making backups. In the patch deployment phase, the patches are installed at their target machines. The final phase is patch deployment verification through monitoring of unexpected service interruptions and post-deployment issues are handled in the post-deployment patch verification phase.

6. Challenges in implementing software security

It has been shown that regulative requirements and rigorous security frameworks have led to the introduction of additional security assurance requirements in terms of additional documentation, security reviews, security testing and audits [43]. Unfortunately, despite the rapid releases of security patches addressing newly discovered vulnerabilities in software products, a majority of cyber attacks have been a result of an exploitation of known vulnerabilities, for which patches already exists. Basically, software security vulnerabilities [153] and flaws are as a result of poorly built software that can lead to easy exploitation by the hackers. For instance, inappropriate security requirements engineering has been cited as one of the reasons for developing bad quality software products. To address this problem, security concerns should be taken into account from the early phases of the software development process. For instance, early identification of security vulnerabilities in the source code has been noted to be an essential and challenging task in the software development process [41]. As pointed in [154], it is a challenging task to incorporate access control [155] into the software if most of the requirements related to the access control are identified and implemented after the functional requirements [154].

Techniques such as Static Application Security Testing (SAST) [156] has been developed to boost quality assurance in software engineering. Unfortunately, integrating SAST tools into industry-level product development for security assessment poses various technical and managerial challenges [3]. In addition, SAST has been shown to generate misleading or irrelevant warnings [157], [158]. Many researchers have investigated the effect of refactoring techniques on software quality attributes such as maintainability, understandability, testability, adaptability, modifiability, and reusability [159], [160], [161], [162]. However, it is observed that software refactoring does not always improve all software quality attributes [159], [161], [163], [164]. Recent studies showed that the effects of refactoring techniques on software quality attributes are inconsistent and contradictory [164]. As pointed out in [165] and [53], the inconsistent or contradictory results concerning the effect of refactoring techniques [166] on software quality, is one of the challenges for developers when they use the refactoring techniques to improve software quality. As such, categorization of the refactoring techniques based on their effect on software quality attributes can potentially help developers in achieving their design objectives. This is through the selection of the most beneficial techniques and applying them at the right places with respect to specific software quality attributes [161], [167], [168], [169], [170]. In this regard, a limited number of studies provided the categorization of refactoring techniques based on their effect on desired quality attributes [161], [171], [172]. On the flip side, the security [173] is not considered in these categorizations. On the other hand, the authors in [54] have stated that the effect of refactoring on security is poorly understood and under-studied.

The authors in [174] have pointed out that the application of security engineering methods and models into agile software development was initially seen as challenging. However, as agile development matured, experiences have often been more positive. Despite this, recent empirical research continues to report organizational and technical problems in adaptation. Owing to the perceived mismatch between formal security engineering and software development, many attempts have been made in combining maturity models with agile processes [175], [176]. Through these combinations, it has been shown that agile software development complies even with strict and formal security requirements. However, this is at the unsurprising expense of slower development and higher cost, largely due to non-agile security processes. In light of this, iterative and incremental software development methods have been developed to enable continuous integration and deployment models such as DevOps [177], [178] that considerably shorten maintenance cycles. In terms of security, this shortening manifests itself in terms of accelerated delivery of security improvements [179] and quicker recovery from security incidents. Unfortunately, in regulated environments, DevOps has challenges caused by the introduction of new practices [180]. Despite these shortcomings, many organizations are utilizing automated processes to deploy software into production [181]. The tools, techniques, and methodologies used for the automation incorporate security considerations. To support continuous delivery, authors in [182] have established an acceptable level of continuous software security using agile methods.

According to [5], software security engineering offers the means of defining, implementing and verifying security in software products. This is performed by following some software security development life cycle models or a security capability maturity models. Unfortunately, agile software development methods and processes which are dominant in the software industry are viewed to be in conflict with these security practices and the security requirements [183],

[184]. In this respect, secure software [185] development involves the execution of a set of security engineering activities in conjunction with software development processes [7]. However, in agile software development the execution and progress of software development processes cannot always be determined in advance, nor are they necessarily predictable at the start of a project. This serves to complicate the software development processes in terms of production bottlenecks in terms of production of additional documentation, performing security-related reviews and scans, and time-consuming security testing [186]. This strict process requirements constrain the flexibility gained through the use of agile methods, which goes against the principles of adaptable processes [174], [176], [187].

7. Research gaps and recommendations

The provision of continuous security assurance evaluation and implementation of countermeasures to achieve the security goals has been noted to be a complex task [188]. For instance, the major research challenge revolves around the integration of security engineering practices, maturity models, and activities required to comply with standards into flexible agile work flow [189]. As such, the main goal should not be maintaining agility, but to produce as secure software [191] as necessary and as efficiently as possible. To achieve this, numerous quality improving techniques in agile development such as iterative development, retrospectives, constant refactoring, and continuous integration can be deployed to produce secure software [191], [192]. As discussed in [193], the software security life cycles in an agile context have been a frequent subject in software security research. For instance, numerous challenges in adapting security [194] engineering into agile development have been identified in [195]. To address some of these challenges, security engineering should be integrated with education [196], 197]. Unfortunately, there is a notable lack of industry surveys directly concentrating on the actual development-time activities.

Software security patch management [198]-[200] has been noted to critical in the industry. However, this is still an emerging area of rising interest in research that needs further attention. Although there are many studies aimed at the provision of technical advancements to enhance software security patch management tasks such as an algorithm [201] for optimizing patching policy selection [202], the socio-technical aspects of software security patch management have received relatively limited attention [148]. These socio-technical issues are concerned with the organizational process, policies, skill and resource management, as well as the interaction of people with technical solutions [203]. However, limitations exist since software security patch management process is inherently a socio-technical endeavour where human and technological interactions are tightly coupled such that the success of software security patch management significantly depends on the effective collaboration of humans with the technical systems [204]. Therefore, the understanding of the socio-technical aspects is necessary for identifying the prevailing issues and improving the effectiveness of the software security patch management process [48], [205].

Regarding the development process, there is need shape security properties through the addition of security methodologies and implementing them correctly. This can be achieved through the observation of security principles and defects avoidance. Here, security requirements guide the design, implementation, and verification work [174]. Unfortunately, security requirements are frequently developed independently of other requirements engineering activities [206]. As a result, many vital security requirements [207] are often ignored, with the requirement engineers focusing only on functional requirements. To curb this, numerous security requirement engineering frameworks [65], [70], [72], techniques [68], processes [63], [71] and methodologies [67] have been proposed over the recent past. As explained in [70], software security is not considered by the developers during the early phases of the software development life cycle. In addition, most security requirements engineering approaches generally do not involve all significant stakeholders. Moreover, they fail to use the well-organized techniques for stakeholder identification and prioritization [208].This leads to incomplete security requirements specifications, or confusing, conflicting, uncohesive, disorganized, infeasible, and obsolete security requirements. This results in requirements that are unable to be validated, and not usable by their anticipated parties.

As explained in [34], security assurance methods based on the traditional tools, techniques, and procedures may fail to cater for new challenges. This is attributed to poor requirement specifications, static nature, and poor development processes. For instance, the Common Criteria (CC) frequently used for security evaluation and certification process has many limitations and challenges. Although security testing and evaluation are beneficial to get the desired level of security assurance, there is no single standard process available to measure the security assurance of the software system [209], [210]. In essence, the development of secure software requires considerations beyond the basic security requirements such as authentication, authorization and mandated operational compliance to identify and resolve the risk environment in which the system must operate. Although security assurance methodology [211] has been considered in the different development life cycles of the software, these methods and techniques come with several drawbacks. For instance, these approaches are static [212], time consuming, do not scale well to the extensive, networked, IT-driven system and fails to offer continuous security assurance [213]. Although some efforts have been devoted to resolving these challenges, they are still open issues.

Security requirements identified during the development phase based on the assumption made on the system's operational environment may be invalid if there are any changes in the system environment [214]. As such, evidence collection should be carried out to validate the fulfillment of security requirements of the system in the operational phase. Basically, this implies security assurance of the software after the deployment or implementation phase. Unfortunately, security assurance evaluation of the system in during the operation phase comes with many challenges as well as benefits that cannot be accomplished by an offline assessment [215]. As authors in [216] explain, operation security assurance is complex due to the openness, aggregation, and dynamics nature of IT and cyber–physical systems. It is not technically possible to make the software systems completely secure since some vulnerabilities [217] may be present, which were not fixed during the development process. This may be due to time constraints or other reasons, and hence must be re-examined, prioritized, and fixed. Since security assurance is a very time-consuming and costly process, optimal security assurance should work towards providing optimal security while at the same time reducing these costs [84].

The main focus of security goals is on the user's demand, whereby these demands are changed whenever there is a change in user requirements. According to [218], these security goals are achieved through rigorous testing, establishment, and assessment to offer the much needed defense against malicious attacks [219]. Unfortunately, system users are the weakest link since they may unintentionally invite attacks. Apart from securing the system from threats, it is important to maintain its usability whose focus is on the ease of users keeping simple formula [220].As pointed out in [221], security assurance measurement of a complex software system is necessary but not always feasible. This is because the modern software systems consist of several components such as servers and clients, protocols, and services such that the security weakness or vulnerability in any of these components result in the compromise of the entire system [222]. As such, a process or methodology is required to uphold the security of software components [223] used in a wide range of applications. In this regards and considering the relationship between these entities, a reverse process is required to combine the security values of the decomposed entities to obtain the security of the entire system [224], [225].Unfortunately, the current offline security assurance evaluation approaches for measuring and evaluating cyber security are not effective. In addition, they are not widely accepted approaches since they do not offer continuous security assurance [226] assessments for complex operational software systems. Consequently, a process, method, or tool is needed for the operational security assurance assessment [227].

As explained in [125], security assurance tools helps in improving the system security through the building of security into software systems or determining how secure it is. There is therefore a need for a security assurance tool that measures the system's security level so that it can be improved and maintained overall. Automated information security analysis, validation, evaluation, and testing approaches are required to obtain the evidence regarding security strength or security performance in the software products and telecommunication system. On the other hand, automation of the security assurance process in open source software is also essential. Since open-source software is subjected to frequent updates, therefore automation process should be able to incorporate these updates [142]. In addition, there is need to consider the needs of the entire software development process, starting from the requirement engineering to its final deployment [141]. Similarly, it is important to take into consideration security assurance throughout the system development life cycle [228]. As discussed in [229], poor design practices such as the improper design of security functionality during the development life cycle is a big security concern. As such, a process or a tool is required to design and develop a secure software system. Similarly, verification and certification of designs and codes are also quite significant [230]. In addition, security assurance is an essential property of the application code that has not been addressed before. Therefore, it is important to ensure that code behaviour is in line with the access control [231], [232] policy. Further, a security assurance methodology is needed to obtain firm evidence that the security requirements of the companies are well defined and enforced [233].

According to [234], access control requirements are mostly considered after the functional requirement definition stage. This leads to flaws and security defects in the access control mechanism [235]. As such, a security assurance mechanism is required to ensure that the application code behaves consistently with the access control policy. As explained in [236], security testing is an essential process in security assurance, which should be implemented iteratively on analytical and practical stages. Basically, security assurance offers confidence that system assets are protected. However, this confidence is hinged on the correctness of the security measures, quality of the security policy, and profile of the attackers [237]. However, effectiveness [238] of the assurance techniques is a factor that is difficult to measure in the operational phase [239]. Therefore, a methodology is required to establish whether the security controls [240]-[244] are adequate and appropriate for specific systems. To attain this, it is essential for these factors to be considered while developing a security assurance technique. Another challenge is that direct measurement of security assurance for complex software systems is not always possible. To address this, the system should be decomposed into measurable parts. This is then followed by the measurement of the assurance of decomposed entities. However, this calls for aggregation to obtain the security assurance of the software system by amalgamating the security assurance values [245]-[249]. This calls for a suitable aggregation technique to accomplish this task. The aggregation technique should consider the composing entities that are security assurance relevant and relations between the components. Moreover, it is essential to understand how constituent components of a system under evaluation contribute to the system's security. As such, it is crucial to define and develop some confidence metrics for security assurance level and the selection and aggregation technique for these security metrics. As pointed out in [250], operating system core can be considered instead of the application level service in order to increase the speed and effectiveness of attack detection [251]-[256]. This is because the operating system's core contains every internal attribute and the file system.

8. Conclusion

Many organizations are becoming heavily dependent on information processing systems. This dependency has brought about the need for protecting software systems from threats. Therefore, software security has become an essential issue in modern organizations. System security assurance offers confidence that practices, security features, procedures and architecture of software systems mediate and enforce the security policy and are resilient against security failure and attacks. It has been shown to entail the demonstration with evidence that a system fulfils established standard security criteria, in addition to offering some confidence that a system meets the security requirements. To define, implement and verify security in software products, some software security development life cycle models or a security capability maturity models are followed. One of such famous models is the agile software development methods and processes which are dominant in the software industry. However, it has been shown to be in conflict with these security practices and the security requirements. It has also been shown that effectiveness of the assurance techniques is a factor that is difficult to measure in the operational phase. Owing to these challenges, numerous recommendations are given in this paper, which are thought to be critical for the assurance of software security. The future research will involve the implementations of some of these recommendations to facilitate the measurement of software security achievements.

Compliance with ethical standards

Acknowledgments

I would like to acknowledge colleagues and the university for the support they offered us during the process of developing this article.

References

[1] Potter B, McGraw G. Software security testing. IEEE Security & Privacy. 2004 Oct 8, 2(5):81-5.

[2] Felderer M, Büchler M, Johns M, Brucker AD, Breu R, Pretschner A. Security testing: A survey. InAdvances in Computers 2016 Jan 1 (Vol. 101, pp. 1-51). Elsevier.

[3] Nguyen-Duc A, Do MV, Hong QL, Khac KN, Quang AN. On the adoption of static analysis for software security assessment–A case study of an open-source e-government project. computers & security. 2021 Dec 1, 111:102470.

[4] Twizeyimana JD, Andersson A. The public value of E-Government–A literature review. Government information quarterly. 2019 Apr 1, 36(2):167-78.

[5] Rindell, K., Ruohonen, J., Holvitie, J., Hyrynsalmi, S., & Leppänen, V. (2021). Security in agile software development: A practitioner survey. Information and Software Technology, 131, 106488.

[6] Nyangaresi VO. Privacy Preserving Three-factor Authentication Protocol for Secure Message Forwarding in Wireless Body Area Networks. Ad Hoc Networks. 2023 Feb 8:103117.

[7] McGraw G. Software security: Building security in. Datenschutz und Datensicherheit-DuD. 2012 Sep, 36(9):662-5.

[8] Siddiqui ST. Significance of security metrics in secure software development. Significance. 2017 Aug, 12(6).

[9] Ansari MT, Pandey D, Alenezi M. STORE: security threat oriented requirements engineering methodology. Journal of King Saud University-Computer and Information Sciences. 2022 Feb 1, 34(2):191-203.

[10] Amoroso E. Recent progress in software security. IEEE Software. 2018 Mar 12, 35(2):11-3.

[11] Morrison P, Moye D, Pandita R, Williams L. Mapping the field of software life cycle security metrics. Information and Software Technology. 2018 Oct 1, 102:146-59.

[12] Mumtaz H, Alshayeb M, Mahmood S, Niazi M. An empirical study to improve software security through the application of code refactoring. Information and Software Technology. 2018 Apr 1, 96:112-25.

[13] Mohammed NM, Niazi M, Alshayeb M, Mahmood S. Exploring software security approaches in software development lifecycle: A systematic mapping study. Computer Standards & Interfaces. 2017 Feb 1, 50:107-15.

[14] Nyangaresi VO, Ogundoyin SO. Certificate based authentication scheme for smart homes. In2021 3rd Global Power, Energy and Communication Conference (GPECOM) 2021 Oct 5 (pp. 202-207). IEEE.

[15] Atifi M, Mamouni A, Marzak A. A comparative study of software testing techniques. InNetworked Systems: 5th International Conference, NETYS 2017, Marrakech, Morocco, May 17-19, 2017, Proceedings 5 2017 (pp. 373-390). Springer International Publishing.

[16] Weir C, Becker I, Blair L. A passion for security: Intervening to help software developers. In2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP) 2021 May 25 (pp. 21-30). IEEE.

[17] Alshammari B, Fidge C, Corney D. Assessing the impact of refactoring on security-critical object-oriented designs. In2010 Asia Pacific Software Engineering Conference 2010 Nov 30 (pp. 186-195). IEEE.

[18] Abduljabbar ZA, Nyangaresi VO, Ma J, Al Sibahee MA, Khalefa MS, Honi DG. MAC-Based Symmetric Key Protocol for Secure Traffic Forwarding in Drones. InFuture Access Enablers for Ubiquitous and Intelligent Infrastructures: 6th EAI International Conference, FABULOUS 2022, Virtual Event, May 4, 2022, Proceedings 2022 Sep 18 (pp. 16-36). Cham: Springer International Publishing.

[19] Haney J, Lutters W, Jacobs J. Cybersecurity Advocates: Force Multipliers in Security Behavior Change. IEEE Security & Privacy. 2021 Jul 5, 19(4):54-9.

[20] Fischer F, Grossklags J. Nudging software developers toward secure code. IEEE Security & Privacy. 2022 Mar 1, 20(02):76-9.

[21] Assal H, Chiasson S. 'Think secure from the beginning' A Survey with Software Developers. InProceedings of the 2019 CHI conference on human factors in computing systems 2019 May 2 (pp. 1-13).

[22] Abduljaleel IQ, Abduljabbar ZA, Al Sibahee MA, Ghrabat MJ, Ma J, Nyangaresi VO. A Lightweight Hybrid Scheme for Hiding Text Messages in Colour Images Using LSB, Lah Transform and Chaotic Techniques. Journal of Sensor and Actuator Networks. 2022 Dec, 11(4):66.

[23] Kuhn R, Raunak MS, Kacker R. Can reducing faults prevent vulnerabilities?. Computer. 2018 Jul 1, 51(7):82-5.

[24] Phillips DM, Mazzuchi TA, Sarkani S. An architecture, system engineering, and acquisition approach for space system software resiliency. Information and Software Technology. 2018 Feb 1, 94:150-64.

[25] Hahn A, Tamimi A, Anderson D. Securing your ics software with the attack surface host analyzer (aha). InProceedings of the 4th Annual Industrial Control System Security Workshop 2018 Dec 4 (pp. 33-39).

[26] Munaiah N, Meneely A. Beyond the attack surface: Assessing security risk with random walks on call graphs. InProceedings of the 2016 ACM Workshop on Software PROtection 2016 Oct 28 (pp. 3-14).

[27] Nyangaresi VO. Lightweight anonymous authentication protocol for resource-constrained smart home devices based on elliptic curve cryptography. Journal of Systems Architecture. 2022 Dec 1, 133:102763.

[28] Ansari MT, Pandey D. Risks, security, and privacy for HIV/AIDS data: big data perspective. InBig Data Analytics in HIV/AIDS Research 2018 (pp. 117-139). IGI Global.

[29] Theisen C, Munaiah N, Al-Zyoud M, Carver JC, Meneely A, Williams L. Attack surface definitions: A systematic literature review. Information and Software Technology. 2018 Dec 1, 104:94-103.

[30] Martin W, Sarro F, Jia Y, Zhang Y, Harman M. A survey of app store analysis for software engineering. IEEE transactions on software engineering. 2016 Dec 2, 43(9):817-47.

[31] Almeida DA, Murphy GC, Wilson G, Hoye M. Investigating whether and how software developers understand open source software licensing. Empirical Software Engineering. 2019 Feb 15, 24:211-39.

[32] Kahya MD, Seneler Ç. Geographical distance challenges in distributed agile software development: Case study of a global company. In2018 3rd International Conference on Computer Science and Engineering (UBMK) 2018 Sep 20 (pp. 78-83). IEEE.

[33] Hussain MA, Hussien ZA, Abduljabbar ZA, Ma J, Al Sibahee MA, Hussain SA, Nyangaresi VO, Jiao X. Provably throttling SQLI using an enciphering query and secure matching. Egyptian Informatics Journal. 2022 Dec 1, 23(4):145-62.

[34] Shukla A, Katt B, Nweke LO, Yeng PK, Weldehawaryat GK. System security assurance: A systematic literature review. Computer Science Review. 2022 Aug 1, 45:100496.

[35] Kim SH, Leem CS. A case study in applying common criteria to development process to improve security of software products. InComputational Science and Its Applications–ICCSA 2004: International Conference, Assisi, Italy, May 14-17, 2004, Proceedings, Part I 4 2004 (pp. 1069-1077). Springer Berlin Heidelberg.

[36] Collier ZA, DiMase D, Walters S, Tehranipoor MM, Lambert JH, Linkov I. Cybersecurity standards: Managing risk and creating resilience. Computer. 2014 Jan 2, 47(9):70-6.

[37] Katt B, Prasher N. Quantitative security assurance metrics: REST API case studies. InProceedings of the 12th European Conference on Software Architecture: Companion Proceedings 2018 Sep 24 (pp. 1-7).

[38] Ouedraogo M, Mouratidis H, Dubois E, Khadraoui D. Information systems security criticality and assurance evaluation. InAdvances in Computer Science and Information Technology: AST/UCMA/ISA/ACN 2010 Conferences, Miyazaki, Japan, June 23-25, 2010. Joint Proceedings 2010 (pp. 38-54). Springer Berlin Heidelberg.

[39] Nyangaresi VO. Provably Secure Pseudonyms based Authentication Protocol for Wearable Ubiquitous Computing Environment. In2022 International Conference on Inventive Computation Technologies (ICICT) 2022 Jul 20 (pp. 1-6). IEEE.

[40] Ardagna CA, Damiani E, Schütte J, Stephanow P. A case for IoT security assurance. Internet of Everything: Algorithms, Methodologies, Technologies and Perspectives. 2018:175-92.

[41] Hovsepyan A, Scandariato R, Joosen W. Is newer always better? The case of vulnerability prediction models. InProceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement 2016 Sep 8 (pp. 1-6).

[42] Avizienis A, Laprie JC, Randell B, Landwehr C. Basic concepts and taxonomy of dependable and secure computing. IEEE transactions on dependable and secure computing. 2004 Oct 4, 1(1):11-33.

[43] Such JM, Gouglidis A, Knowles W, Misra G, Rashid A. Information assurance techniques: Perceived cost effectiveness. Computers & Security. 2016 Jul 1, 60:117-33.

[44] Rajapakse RN, Zahedi M, Babar MA, Shen H. Challenges and solutions when adopting DevSecOps: A systematic review. Information and software technology. 2022 Jan 1, 141:106700.

[45] Hussien ZA, Abdulmalik HA, Hussain MA, Nyangaresi VO, Ma J, Abduljabbar ZA, Abduljaleel IQ. Lightweight Integrity Preserving Scheme for Secure Data Exchange in Cloud-Based IoT Systems. Applied Sciences. 2023 Jan, 13(2):691.

[46] Souppaya M, Scarfone K. Guide to enterprise patch management technologies. NIST Special Publication. 2013 Jul, 800:40.

[47] Li F, Rogers L, Mathur A, Malkin N, Chetty M. Keepers of the machines: examining how system administrators manage software updates. InProceedings of the Fifteenth USENIX Conference on Usable Privacy and Security 2019 Nov (pp. 273-288). USENIX Association.

[48] Tiefenau C, Häring M, Krombholz K, Von Zezschwitz E. Security, availability, and multiple information sources: Exploring update behavior of system administrators. arXiv preprint arXiv:2007.08875. 2020 Jul 17.

[49] Gentile U, Serio L. Survey on international standards and best practices for patch management of complex industrial control systems: the critical infrastructure of particle accelerators case study. International Journal of Critical Computer-Based Systems. 2019, 9(1-2):115-32.

[50] Ghaffarian SM, Shahriari HR. Software vulnerability analysis and discovery using machine-learning and data-mining techniques: A survey. ACM Computing Surveys (CSUR). 2017 Aug 25, 50(4):1-36.

[51] Nyangaresi VO. Masked Symmetric Key Encrypted Verification Codes for Secure Authentication in Smart Grid Networks. In 2022 4th Global Power, Energy and Communication Conference (GPECOM) 2022 Jun 14 (pp. 427-432).

[52] Charest T, Rodgers N, Wu Y. Comparison of static analysis tools for Java using the Juliet test suite. In11th International Conference on Cyber Warfare and Security 2016 Feb (pp. 431-438).

[53] Almogahed A. Empirical studies on software refactoring techniques in the industrial setting. Turkish Journal of Computer and Mathematics Education (TURCOMAT). 2021 Apr 11, 12(3):1705-16.

[54] Abid C, Kessentini M, Alizadeh V, Dhaouadi M, Kazman R. How does refactoring impact security when improving quality? a security-aware refactoring approach. IEEE Transactions on Software Engineering. 2020 Jun 30, 48(3):864-78.

[55] Kaya M, Conley S, Othman ZS, Varol A. Effective software refactoring process. In2018 6th International Symposium on Digital Forensic and Security (ISDFS) 2018 Mar 22 (pp. 1-6). IEEE.

[56] Ouni A, Kessentini M, Sahraoui H, Inoue K, Deb K. Multi-criteria code refactoring using search-based software engineering: An industrial case study. ACM Transactions on Software Engineering and Methodology (TOSEM). 2016 Jun 30, 25(3):1-53.

[57] Fernandes E, Chávez A, Garcia A, Ferreira I, Cedrim D, Sousa L, Oizumi W. Refactoring effect on internal quality attributes: What haven't they told you yet?. Information and Software Technology. 2020 Oct 1, 126:106347.

[58] Mutlaq KA, Nyangaresi VO, Omar MA, Abduljabbar ZA. Symmetric Key Based Scheme for Verification Token Generation in Internet of Things Communication Environment. InApplied Cryptography in Computer and Communications: Second EAI International Conference, AC3 2022, Virtual Event, May 14-15, 2022, Proceedings 2022 Oct 6 (pp. 46-64). Cham: Springer Nature Switzerland.

[59] Alshammari B, Fidge C, Corney D. Security metrics for object-oriented class designs. In2009 Ninth International Conference on Quality Software 2009 Aug 24 (pp. 11-20). IEEE.

[60] Kanaki K, Kalogiannakis M. Introducing fundamental object-oriented programming concepts in preschool education within the context of physical science courses. Education and Information Technologies. 2018 Nov, 23(6):2673-98.

[61] Bellomo S, Kruchten P, Nord RL, Ozkaya I. How to agilely architect an agile architecture. Cutter IT Journal. 2014 Feb, 27(2):12-7.

[62] Tondel IA, Jaatun MG, Meland PH. Security requirements for the rest of us: A survey. IEEE software. 2008 Jan 7, 25(1):20-7.

[63] Hassan R, Eltoweissy M, Bohner S, El-Kassas S. Formal analysis and design for engineering security automated derivation of formal software security specifications from goal-oriented security requirements. IET software. 2010 Apr 1, 4(2):149-60.

[64] Nyangaresi VO, Ma J. A Formally Verified Message Validation Protocol for Intelligent IoT E-Health Systems. In2022 IEEE World Conference on Applied Intelligence and Computing (AIC) 2022 Jun 17 (pp. 416-422). IEEE.

[65] Riaz M, Stallings J, Singh MP, Slankas J, Williams L. DIGS: A framework for discovering goals for security requirements engineering. InProceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement 2016 Sep 8 (pp. 1-10).

[66] Salini P, Kanmani S. Model oriented security requirements engineering (MOSRE) framework for web applications. InAdvances in Computing and Information Technology: Proceedings of the Second International Conference on Advances in Computing and Information Technology (ACITY) July 13-15, 2012, Chennai, India-Volume 2 2013 (pp. 341-353). Springer Berlin Heidelberg.

[67] El-Hadary H, El-Kassas S. Capturing security requirements for software systems. Journal of advanced research. 2014 Jul 1, 5(4):463-72.

[68] Paja E, Dalpiaz F, Giorgini P. Modelling and reasoning about security requirements in socio-technical systems. Data & Knowledge Engineering. 2015 Jul 1, 98:123-43.

[69] Abduljabbar ZA, Omollo Nyangaresi V, Al Sibahee MA, Ghrabat MJ, Ma J, Qays Abduljaleel I, Aldarwish AJ. Session-Dependent Token-Based Payload Enciphering Scheme for Integrity Enhancements in Wireless Networks. Journal of Sensor and Actuator Networks. 2022 Sep 19, 11(3):55.

[70] Ansari TJ, Pandey D. An Integration of Threat Modeling with Attack Pattern and Misuse Case for Effective Security Requirement Elicitation. International Journal of Advanced Research in Computer Science. 2017 Mar 15, 8(3).

[71] Mufti Y, Niazi M, Alshayeb M, Mahmood S. A readiness model for security requirements engineering. IEEE Access. 2018 May 24, 6:28611-31.

[72] Rehman SU, Gruhn V. An effective security requirements engineering framework for cyber-physical systems. Technologies. 2018 Jul 12, 6(3):65.

[73] Choi W, Yoo D. Software assurance towards better IT service. Journal of Service Science. 2009 Jun, 1:31-56.

[74] Nyangaresi VO. A Formally Validated Authentication Algorithm for Secure Message Forwarding in Smart Home Networks. SN Computer Science. 2022 Jul 9, 3(5):364.

[75] Brown M. Toward a taxonomy of communications security models. Journal of Cryptographic Engineering. 2013 Sep, 3(3):181-95.

[76] Bijani S, Robertson D. A review of attacks and security approaches in open multi-agent systems. Artificial Intelligence Review. 2014 Dec, 42:607-36.

[77] Oueslati H, Rahman MM, ben Othmane L. Literature review of the challenges of developing secure software using the agile approach. In2015 10th International Conference on Availability, Reliability and Security 2015 Aug 24 (pp. 540-547). IEEE.

[78] Dissanayaka AM, Mengel S, Gittner L, Khan H. Security assurance of MongoDB in singularity LXCs: an elastic and convenient testbed using Linux containers to explore vulnerabilities. Cluster Computing. 2020 Sep, 23:1955-71.

[79] Ouedraogo M, Mouratidis H, Khadraoui D, Dubois E. A risk based approach for security assurance evaluation of IT systems. In2009 Seventh Annual Communication Networks and Services Research Conference 2009 May 11 (pp. 428-430). IEEE.

[80] Al Sibahee MA, Nyangaresi VO, Ma J, Abduljabbar ZA. Stochastic Security Ephemeral Generation Protocol for 5G Enabled Internet of Things. InIoT as a Service: 7th EAI International Conference, IoTaaS 2021, Sydney, Australia, December 13–14, 2021, Proceedings 2022 Jul 8 (pp. 3-18). Cham: Springer International Publishing.

[81] Ouedraogo M, Savola RM, Mouratidis H, Preston D, Khadraoui D, Dubois E. Taxonomy of quality metrics for assessing assurance of security correctness. Software Quality Journal. 2013 Mar, 21:67-97.

[82] Kim TH, Lee SY. Security evaluation targets for enhancement of IT systems assurance. InComputational Science and Its Applications–ICCSA 2005: International Conference, Singapore, May 9-12, 2005, Proceedings, Part II 5 2005 (pp. 491-498). Springer Berlin Heidelberg.

[83] Akram J, Luo P. SQVDT: A scalable quantitative vulnerability detection technique for source code security assessment. Software: Practice and Experience. 2021 Feb, 51(2):294-318.

[84] Kumar R, Khan AI, Abushark YB, Alam MM, Agrawal A, Khan RA. A knowledge-based integrated system of hesitant fuzzy set, AHP and TOPSIS for evaluating security-durability of web applications. IEEE Access. 2020 Mar 3, 8:48870-85.

[85] Bialas A, Michalak M, Flisiuk B. Anomaly detection in network traffic security assurance. InEngineering in Dependability of Computer Systems and Networks: Proceedings of the Fourteenth International Conference on Dependability of Computer Systems DepCoS-RELCOMEX, July 1–5, 2019, Brunów, Poland 2020 (pp. 46-56). Springer International Publishing.

[86] Nyangaresi VO, Ahmad M, Alkhayyat A, Feng W. Artificial neural network and symmetric key cryptography based verification protocol for 5G enabled Internet of Things. Expert Systems. 2022 Dec, 39(10):e13126.

[87] Bobelin L, Bousquet A, Briffaut J. An autonomic cloud management system for enforcing security and assurance properties. InProceedings of the 2015 Workshop on Changing Landscapes in HPC Security 2015 Jun 16 (pp. 1-8).

[88] Hudic A, Smith P, Weippl ER. Security assurance assessment methodology for hybrid clouds. Computers & Security. 2017 Sep 1, 70:723-43.

[89] Kanstrén T, Evesti A. Security metrics, secure elements, and operational measurement trust in cloud environments. InSecurity and Trust Management: 11th International Workshop, STM 2015, Vienna, Austria, September 21-22, 2015, Proceedings 11 2015 (pp. 37-51). Springer International Publishing.

[90] Malhotra R, Jain J. Analysis of refactoring effect on software quality of object-oriented systems. InInternational Conference on Innovative Computing and Communications: Proceedings of ICICC 2018, Volume 2 2019 (pp. 197-212). Springer Singapore.

[91] Bilgin Z, Ersoy MA, Soykan EU, Tomur E, Çomak P, Karaçay L. Vulnerability prediction from source code using machine learning. IEEE Access. 2020 Aug 14, 8:150672-84.

[92] Ghrabat MJ, Hussien ZA, Khalefa MS, Abduljabba ZA, Nyangaresi VO, Al Sibahee MA, Abood EW. Fully automated model on breast cancer classification using deep learning classifiers. Indonesian Journal of Electrical Engineering and Computer Science. 2022 Oct, 28(1):183-91.

[93] Diamantopoulou V, Kalloniatis C, Lyvas C, Maliatsos K, Gay M, Kanatas A, Lambrinoudakis C. Aligning the concepts of risk, security and privacy towards the design of secure intelligent transport systems. InComputer Security: ESORICS 2020 International Workshops, CyberICPS, SECPRE, and ADIoT, Guildford, UK, September 14–18, 2020, Revised Selected Papers 6 2020 (pp. 170-184). Springer International Publishing.

[94] Cuihua X, Jiajun L. An object-oriented information system security evaluation method based on security level distinguishing model. In2009 International Conference on Web Information Systems and Mining 2009 Nov 7 (pp. 497-500). IEEE.

[95] Somarakis I, Smyrlis M, Fysarakis K, Spanoudakis G. Model-driven cyber range training: a cyber security assurance perspective. InComputer Security: ESORICS 2019 International Workshops, IOSec, MSTEC, and FINSEC, Luxembourg City, Luxembourg, September 26–27, 2019, Revised Selected Papers 2 2020 (pp. 172-184). Springer International Publishing.

[96] Ismail UM, Islam S. A unified framework for cloud security transparency and audit. Journal of Information Security and Applications. 2020 Oct 1, 54:102594.

[97] Nyangaresi VO. ECC based authentication scheme for smart homes. In2021 International Symposium ELMAR 2021 Sep 13 (pp. 5-10). IEEE.

[98] Elder S, Zahan N, Kozarev V, Shu R, Menzies T, Williams L. Structuring a comprehensive software security course around the OWASP application security verification standard. In2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET) 2021 May 25 (pp. 95-104). IEEE.

[99] Nurdiani I, Börstler J, Fricker S, Petersen K, Chatzipetrou P. Understanding the order of agile practice introduction: Comparing agile maturity models and practitioners' experience. Journal of Systems and Software. 2019 Oct 1, 156:1-20.

[100] Guo KH. Security-related behavior in using information systems in the workplace: A review and synthesis. Computers & Security. 2013 Feb 1, 32:242-51.

[101] Ameller D, Galster M, Avgeriou P, Franch X. A survey on quality attributes in service-based systems. Software quality journal. 2016 Jun, 24:271-99.

[102] Mahdavi-Hezavehi S, Galster M, Avgeriou P. Variability in quality attributes of service-based software systems: A systematic literature review. Information and Software Technology. 2013 Feb 1, 55(2):320-43.

[103] Nyangaresi VO. Provably secure protocol for 5G HetNets. In2021 IEEE International Conference on Microwaves, Antennas, Communications and Electronic Systems (COMCAS) 2021 Nov 1 (pp. 17-22). IEEE.

[104] Mirandola R, Potena P. Self-adaptation of service based systems based on cost/quality attributes tradeoffs. In2010 12th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing 2010 Sep 23 (pp. 493-501). IEEE.

[105] Sajwan L, Noble J, Anslow C, Biddle R. Why do programmers do what they do? a theory of influences on security practices. InWorkshop on Usable Security and Privacy 2021 May.

[106] Agrawal A, Alenezi M, Khan SA, Kumar R, Khan RA. Multi-level fuzzy system for usable-security assessment. Journal of King Saud University-Computer and Information Sciences. 2022 Mar 1, 34(3):657-65.

[107] Votipka D, Fulton KR, Parker J, Hou M, Mazurek ML, Hicks M. Understanding security mistakes developers make: Qualitative analysis from build it, break it, fix it. In29th USENIX Security Symposium (USENIX Security 20) 2020 (pp. 109-126).

[108] Abood EW, Abdullah AM, Al Sibahe MA, Abduljabbar ZA, Nyangaresi VO, Kalafy SA, Ghrabta MJ. Audio steganography with enhanced LSB method for securing encrypted text with bit cycling. Bulletin of Electrical Engineering and Informatics. 2022 Feb 1, 11(1):185-94.

[109] Lan Y, Han T. SADP: Security assurance development process for building reliable linux-based operating system. In2015 IEEE International Conference on Computer and Communications (ICCC) 2015 Oct 10 (pp. 50-55). IEEE.

[110] Schmittner C, Ma Z, Schoitsch E. Combined safety and security development lifecylce. In2015 IEEE 13th International Conference on Industrial Informatics (INDIN) 2015 Jul 22 (pp. 1408-1415). IEEE.

[111] Kalaimannan E, Gupta JN. The security development lifecycle in the context of accreditation policies and standards. IEEE Security & Privacy. 2017 Feb 14, 15(1):52-7.

[112] Lipner S. Security development lifecycle: Security considerations for client and cloud Applications. Datenschutz und Datensicherheit-DuD. 2010 Mar, 34(3):135-7.

[113] Al-Fedaghi S, Alkandari A. On security development lifecycle: Conceptual description of vulnerabilities, risks, and threats. International Journal of Digital Content Technology and its Applications. 2011 May, 5(5).

[114] Liou JC, Duclervil SR. A survey on the effectiveness of the secure software development life cycle models. Innovations in Cybersecurity Education. 2020:213-29.

[115] Nyangaresi VO. Terminal independent security token derivation scheme for ultra-dense IoT networks. Array. 2022 Sep 1, 15:100210.

[116] Naeem RZ, Abbas H, Shafqat N, Saleem K, Iqbal W. A framework to determine applications' authenticity. Procedia Computer Science. 2019 Jan 1, 155:268-75.

[117] Savola RM, Heinonen P. A visualization and modeling tool for security metrics and measurements management. In2011 Information Security for South Africa 2011 Aug 15 (pp. 1-8). IEEE.

[118] Al-Kilidar H, Cox K, Kitchenham B. The use and usefulness of the ISO/IEC 9126 quality standard. In2005 International Symposium on Empirical Software Engineering, 2005. 2005 Nov 17 (pp. 7-pp). IEEE.

[119] Botella P, Burgués X, Carvallo JP, Franch X, Grau G, Marco J, Quer C. ISO/IEC 9126 in practice: what do we need to know. InSoftware Measurement European Forum 2004 Jan 1 (Vol. 2004).

[120] Jung HW, Kim SG, Chung CS. Measuring software product quality: A survey of ISO/IEC 9126. IEEE software. 2004 Oct 4, 21(5):88-92.

[121] Mesquida AL, Mas A, Amengual E, Calvo-Manzano JA. IT Service Management Process Improvement based on ISO/IEC 15504: A systematic review. Information and Software Technology. 2012 Mar 1, 54(3):239-47.

[122] Kurniawan E, Riadi I. Security level analysis of academic information systems based on standard ISO 27002: 2003 using SSE-CMM. arXiv preprint arXiv:1802.03613. 2018 Feb 10.

[123] Nyangaresi VO. Hardware assisted protocol for attacks prevention in ad hoc networks. InEmerging Technologies in Computing: 4th EAI/IAER International Conference, iCETiC 2021, Virtual Event, August 18–19, 2021, Proceedings 4 2021 (pp. 3-20). Springer International Publishing.

[124] Hamid B, Weber D. Engineering secure systems: Models, patterns and empirical validation. Computers & Security. 2018 Aug 1, 77:315-48.

[125] Deveci E, Caglayan MU. Model driven security framework for software design and verification. Security and Communication Networks. 2015 Nov 10, 8(16):2768-92.

[126] Fekete A. Common criteria for the assessment of critical infrastructures. International Journal of Disaster Risk Science. 2011 Mar, 2:15-24.

[127] Nassar V. Common criteria for usability review. Work. 2012 Jan 1, 41(Supplement 1):1053-7.

[128] Santiago-Delefosse M, Gavin A, Bruchez C, Roux P, Stephen SL. Quality of qualitative research in the health sciences: Analysis of the common criteria present in 58 assessment guidelines by expert users. Social Science & Medicine. 2016 Jan 1, 148:142-51.

[129] Abood EW, Hussien ZA, Kawi HA, Abduljabbar ZA, Nyangaresi VO, Ma J, Al Sibahee MA, Kalafy A, Ahmad S. Provably secure and efficient audio compression based on compressive sensing. International Journal of Electrical & Computer Engineering (2088-8708). 2023 Feb 1, 13(1).

[130] Rindell K, Hyrynsalmi S, Leppänen V. Securing Scrum for VAHTI. InSPLST 2015 Jun (pp. 236-250).

[131] Kelo T, Eronen J, Rousku K. Model for efficient development of security audit criteria. InProceedings of the 17th European Conference on Cyber Warfare and Security: ECCWS 2018 Jun 1 (pp. 244-52).

[132] Rindell K, Ruohonen J, Hyrynsalmi S. Surveying secure software development practices in finland. InProceedings of the 13th International Conference on Availability, Reliability and Security 2018 Aug 27 (pp. 1-7).

[133] Rindell K, Hyrynsalmi S, Leppänen V. A comparison of security assurance support of agile software development methods. InProceedings of the 16th International Conference on Computer Systems and Technologies 2015 Jun 25 (pp. 61-68).

[134] Nyangaresi VO. Lightweight key agreement and authentication protocol for smart homes. In2021 IEEE AFRICON 2021 Sep 13 (pp. 1-6). IEEE.

[135] Rajamäki J. Challenges to a smooth-running data security audits. Case: A Finnish national security auditing criteria KATAKRI. In2014 IEEE Joint Intelligence and Security Informatics Conference 2014 Sep 24 (pp. 240-243). IEEE.

[136] Nykänen R, Hakuli M. Information security management system standards: A gap analysis of the risk management in ISO 27001 and KATAKRI. InProceedings of the 12th European Conference on Information Warfare and Security 2013 Jul 11 (Vol. 1, pp. 344-350).

[137] Rajamäki J, Rajamäki M. National security auditing criteria, KATAKRI: Leading auditor training and auditing process. In12th European Conference on Information Warfare and Security, Academic Conferences and Publishing International Limited, Reading 2013 Jul 11 (pp. 217-223).

[138] Nykänen R, Kärkkäinen T. Comparison of two Specifications to Fulfill Security Control Objectives. InEuropean Conference on Cyber Warfare and Security 2014 Jul 1 (p. 150). Academic Conferences International Limited.

[139] Nykänen R, Kärkkäinen T. Aligning two specifications for controlling information security. International Journal of Cyber Warfare and Terrorism (IJCWT). 2014 Apr 1, 4(2):46-62.

[140] Nyangaresi VO, Petrovic N. Efficient PUF based authentication protocol for internet of drones. In2021 International Telecommunications Conference (ITC-Egypt) 2021 Jul 13 (pp. 1-4). IEEE.

[141] Khan RA, Khan SU. A preliminary structure of software security assurance model. InProceedings of the 13th International Conference on Global Software Engineering 2018 May 27 (pp. 137-140).

[142] Rauf I, Troubitsyna E. Towards a model-driven security assurance of open source components. InSoftware Engineering for Resilient Systems: 9th International Workshop, SERENE 2017, Geneva, Switzerland, September 4–5, 2017, Proceedings 9 2017 (pp. 65-80). Springer International Publishing.

[143] Pavlich-Mariscal JA, Demurjian SA, Michel LD. A framework for security assurance of access control enforcement code. Computers & Security. 2010 Oct 1, 29(7):770-84.

[144] Gasiba TE, Beckers K, Suppan S, Rezabek F. On the requirements for serious games geared towards software developers in the industry. In2019 IEEE 27th International Requirements Engineering Conference (RE) 2019 Sep 23 (pp. 286-296). IEEE.

[145] Abduljabbar ZA, Abduljaleel IQ, Ma J, Al Sibahee MA, Nyangaresi VO, Honi DG, Abdulsada AI, Jiao X. Provably secure and fast color image encryption algorithm based on s-boxes and hyperchaotic map. IEEE Access. 2022 Feb 11, 10:26257-70.

[146] Simpson S. SAFECode whitepaper: Fundamental practices for secure software development 2nd edition. InISSE 2014 Securing Electronic Business Processes: Highlights of the Information Security Solutions Europe 2014 Conference 2014 Oct 18 (pp. 1-32). Wiesbaden: Springer Fachmedien Wiesbaden.

[147] Sklyar V, Kharchenko V. Challenges in assurance case application for industrial IoT. In2017 9th IEEE international conference on intelligent data acquisition and advanced computing systems: technology and applications (IDAACS) 2017 Sep 21 (Vol. 2, pp. 736-739). IEEE.

[148] Dissanayake N, Jayatilaka A, Zahedi M, Babar MA. Software security patch management-A systematic literature review of challenges, approaches, tools and practices. Information and Software Technology. 2022 Apr 1, 144:106771.

[149] Li F, Paxson V. A large-scale empirical study of security patches. InProceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security 2017 Oct 30 (pp. 2201-2215).

[150] Araujo F, Hamlen KW, Biedermann S, Katzenbeisser S. From patches to honey-patches: Lightweight attacker misdirection, deception, and disinformation. InProceedings of the 2014 ACM SIGSAC conference on computer and communications security 2014 Nov 3 (pp. 942-953).

[151] Nyangaresi VO, Mohammad Z. Privacy preservation protocol for smart grid networks. In2021 International Telecommunications Conference (ITC-Egypt) 2021 Jul 13 (pp. 1-4). IEEE.

[152] Alshawish A, De Meer H. Prioritize When Patching Everything is Impossible!. In2019 IEEE 44th Conference on Local Computer Networks (LCN) 2019 Oct 14 (pp. 125-128). IEEE.

[153] Cadariu M, Bouwers E, Visser J, van Deursen A. Tracking known security vulnerabilities in proprietary software systems. In2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER) 2015 Mar 2 (pp. 516-519). IEEE.

[154] Devanbu PT, Stubblebine S. Software engineering for security: a roadmap. InProceedings of the Conference on the Future of Software Engineering 2000 May 1 (pp. 227-239).

[155] Nyakomitta SP, Omollo V. Biometric-Based Authentication Model for E-Card Payment Technology. IOSR Journal of Computer Engineering (IOSRJCE). 2014, 16(5):137-44.

[156] Yang J, Tan L, Peyton J, Duer KA. Towards better utilizing static application security testing. In2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP) 2019 May 25 (pp. 51-60). IEEE.

[157] Aloraini B, Nagappan M, German DM, Hayashi S, Higo Y. An empirical study of security warnings from static application security testing tools. Journal of Systems and Software. 2019 Dec 1, 158:110427.

[158] Pashchenko I. FOSS version differentiation as a benchmark for static analysis security testing tools. InProceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering 2017 Aug 21 (pp. 1056-1058).

[159] Al Dallal J, Abdin A. Empirical evaluation of the impact of object-oriented code refactoring on quality attributes: A systematic literature review. IEEE Transactions on Software Engineering. 2017 Jan 25, 44(1):44-69.

[160] Nyangaresi VO, Moundounga AR. Secure data exchange scheme for smart grids. In2021 IEEE 6th International Forum on Research and Technology for Society and Industry (RTSI) 2021 Sep 6 (pp. 312-316). IEEE.

[161] Kaur S, Singh P. How does object-oriented code refactoring influence software quality? Research landscape and challenges. Journal of Systems and Software. 2019 Nov 1, 157:110394.

[162] Almogahed A, Omar M, Zakaria NH. Impact of software refactoring on software quality in the industrial environment: A review of empirical studies.

[163] Lacerda G, Petrillo F, Pimenta M, Guéhéneuc YG. Code smells and refactoring: A tertiary systematic review of challenges and observations. Journal of Systems and Software. 2020 Sep 1, 167:110610.

[164] Almogahed A, Omar M, Zakaria NH. Categorization refactoring techniques based on their effect on software quality attributes. International Journal of Innovative Technology and Exploring Engineering (IJITEE). 2019 Jun, 8(8S):439-45.

[165] Nyamawe AS, Liu H, Niu Z, Wang W, Niu N. Recommending refactoring solutions based on traceability and code metrics. IEEE Access. 2018 Sep 6, 6:49460-75.

[166] Elish KO, Alshayeb M. Using Software Quality Attributes to Classify Refactoring to Patterns. J. Softw.. 2012 Feb, 7(2):408-19.

[167] Mohammad Z, Nyangaresi V, Abusukhon A. On the Security of the Standardized MQV Protocol and Its Based Evolution Protocols. In2021 International Conference on Information Technology (ICIT) 2021 Jul 14 (pp. 320-325). IEEE.

[168] Alshayeb M, Al-Jamimi H, Elish MO. Empirical taxonomy of refactoring methods for aspect-oriented programming. Journal of Software: Evolution and Process. 2013 Jan, 25(1):1-25.

[169] Abebe M, Yoo CJ. Trends, opportunities and challenges of software refactoring: A systematic literature review. international Journal of software engineering and its Applications. 2014 Jun, 8(6):299-318.

[170] Bashir RS, Lee SP, Yung CC, Alam KA, Ahmad RW. A methodology for impact evaluation of refactoring on external quality attributes of a software design. In2017 International Conference on Frontiers of Information Technology (FIT) 2017 Dec 18 (pp. 183-188). IEEE.

[171] Mkaouer MW, Kessentini M, Bechikh S, Ó Cinnéide M, Deb K. On the use of many quality attributes for software refactoring: a many-objective search-based software engineering approach. Empirical Software Engineering. 2016 Dec, 21:2503-45.

[172] Mens T, Tourwé T. A survey of software refactoring. IEEE Transactions on software engineering. 2004 Feb, 30(2):126-39.

[173] Nyangaresi VO, Morsy MA. Towards privacy preservation in internet of drones. In2021 IEEE 6th International Forum on Research and Technology for Society and Industry (RTSI) 2021 Sep 6 (pp. 306-311). IEEE.

[174] Türpe S, Poller A. Managing Security Work in Scrum: Tensions and Challenges. SecSE@ ESORICS. 2017 Jan, 2017:34-49.

[175] Schweigert T, Vohwinkel D, Korsaa M, Nevalainen R, Biro M. Agile maturity model: analysing agile maturity characteristics from the SPICE perspective. Journal of Software: Evolution and Process. 2014 May, 26(5):513-20.

[176] Silva FS, Soares FS, Peres AL, De Azevedo IM, Vasconcelos AP, Kamei FK, de Lemos Meira SR. Using CMMI together with agile software development: A systematic review. Information and Software Technology. 2015 Feb 1, 58:20-43.

[177] Izurieta C, Prouty M. Leveraging secdevops to tackle the technical debt associated with cybersecurity attack tactics. In2019 IEEE/ACM International Conference on Technical Debt (TechDebt) 2019 May 26 (pp. 33-37). IEEE.

[178] Leite L, Rocha C, Kon F, Milojicic D, Meirelles P. A survey of DevOps concepts and challenges. ACM Computing Surveys (CSUR). 2019 Nov 14, 52(6):1-35.

[179] Nyangaresi VO, Alsamhi SH. Towards secure traffic signaling in smart grids. In2021 3rd Global Power, Energy and Communication Conference (GPECOM) 2021 Oct 5 (pp. 196-201). IEEE.

[180] Laukkarinen T, Kuusinen K, Mikkonen T. Regulated software meets DevOps. Information and Software Technology. 2018 May 1, 97:176-8.

[181] Bell L, Brunton-Spall M, Smith R, Bird J. Agile application security: enabling security in a continuous delivery pipeline. " O'Reilly Media, Inc.",2017 Sep 8.

[182] ben Othmane L, Angin P, Weffers H, Bhargava B. Extending the agile development process to develop acceptably secure software. IEEE Transactions on dependable and secure computing. 2014 Jan 9, 11(6):497-509.

[183] Pereira JC, de FSM Russo R. Design thinking integrated in agile software development: A systematic literature review. Procedia computer science. 2018 Jan 1, 138:775-82.

[184] Stoica M, Ghilic-Micu B, Mircea M, Uscatu C. Analyzing agile development-from waterfall style to scrumban. Informatica Economica. 2016 Oct 1, 20(4):5.

[185] Nyangaresi VO, Abd-Elnaby M, Eid MM, Nabih Zaki Rashed A. Trusted authority based session key agreement and authentication algorithm for smart grid networks. Transactions on Emerging Telecommunications Technologies. 2022 Sep, 33(9):e4528.

[186] Younas M, Jawawi DN, Mahmood AK, Ahmad MN, Sarwar MU, Idris MY. Agile software development using cloud computing: a case study. IEEE Access. 2019 Dec 25, 8:4475-84.

[187] Rindell K, Hyrynsalmi S, Leppänen V. Busting a myth: Review of agile security engineering methods. InProceedings of the 12th International Conference on Availability, Reliability and Security 2017 Aug 29 (pp. 1-10).

[188] Haddad S, Dubus S, Hecker A, Kanstrén T, Marquet B, Savola R. Operational security assurance evaluation in open infrastructures. In2011 6th International Conference on Risks and Security of Internet and Systems (CRiSIS) 2011 Sep 26 (pp. 1-6). IEEE.

[189] Poth A, Sasabe S, Mas A, Mesquida AL. Lean and agile software process improvement in traditional and agile environments. Journal of Software: Evolution and Process. 2019 Jan, 31(1):e1986.

[190] Alsamhi SH, Shvetsov AV, Kumar S, Shvetsova SV, Alhartomi MA, Hawbani A, Rajput NS, Srivastava S, Saif A, Nyangaresi VO. UAV computing-assisted search and rescue mission framework for disaster and harsh environment mitigation. Drones. 2022 Jun 22, 6(7):154.

[191] Bartsch S. Practitioners' perspectives on security in agile development. In2011 Sixth International Conference on Availability, Reliability and Security 2011 Aug 22 (pp. 479-484). IEEE.

[192] Kasauli R, Knauss E, Kanagwa B, Nilsson A, Calikli G. Safety-critical systems and agile development: A mapping study. In2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA) 2018 Aug 29 (pp. 470-477). IEEE.

[193] Ayalew T, Kidane T, Carlsson B. Identification and evaluation of security activities in agile projects. InSecure IT Systems: 18th Nordic Conference, NordSec 2013, Ilulissat, Greenland, October 18-21, 2013, Proceedings 18 2013 (pp. 139-153). Springer Berlin Heidelberg.

[194] Nyangaresi VO, Mohammad Z. Session Key Agreement Protocol for Secure D2D Communication. InThe Fifth International Conference on Safety and Security with IoT: SaSeIoT 2021 2022 Jun 12 (pp. 81-99). Cham: Springer International Publishing.

[195] Oueslati H, Rahman MM, ben Othmane L, Ghani I, Arbain AF. Evaluation of the challenges of developing secure software using the agile approach. International Journal of Secure Software Engineering (IJSSE). 2016 Jan 1, 7(1):17-37.

[196] Oyetoyan TD, Cruzes DS, Jaatun MG. An empirical study on the relationship between software security skills, usage and training needs in agile settings. In2016 11th International Conference on Availability, Reliability and Security (ARES) 2016 Aug 31 (pp. 548-555). IEEE.

[197] Williams L, McGraw G, Migues S. Engineering security vulnerability prevention, detection, and response. IEEE Software. 2018 Jul 11, 35(5):76-80.

[198] Dey D, Lahiri A, Zhang G. Optimal policies for security patch management. INFORMS Journal on Computing. 2015 Aug, 27(3):462-77.

[199] Brykczynski B, Small RA. Reducing internet-based intrusions: Effective security patch management. IEEE software. 2003 Jan 6, 20(1):50-7.

[200] Souppaya M, Scarfone K. Guide to enterprise patch management technologies. NIST Special Publication. 2013 Jul, 800:40.

[201] Nyangaresi VO. A Formally Verified Authentication Scheme for mmWave Heterogeneous Networks. Inthe 6th International Conference on Combinatorics, Cryptography, Computer Science and Computation (605-612) 2021.

[202] Uemura T, Dohi T. Optimal Security Patch Management Policies Maximizing System Availability. J. Commun.. 2010 Jan, 5(1):71-80.

[203] Islam C, Babar MA, Nepal S. A multi-vocal review of security orchestration. ACM Computing Surveys (CSUR). 2019 Apr 30, 52(2):1-45.

[204] Siavvas M, Tsoukalas D, Jankovic M, Kehagias D, Tzovaras D. Technical debt as an indicator of software security risk: a machine learning approach for software development enterprises. Enterprise Information Systems. 2022 May 4, 16(5):1824017.

[205] Dissanayake N, Zahedi M, Jayatilaka A, Babar MA. A grounded theory of the role of coordination in software security patch management. InProceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering 2021 Aug 20 (pp. 793-805).

[206] Demi S. Blockchain-oriented requirements engineering: A framework. In2020 IEEE 28th International Requirements Engineering Conference (RE) 2020 Aug 31 (pp. 428-433). IEEE.

[207] Nyangaresi VO. Target Tracking Area Selection and Handover Security in Cellular Networks: A Machine Learning Approach. InProceedings of Third International Conference on Sustainable Expert Systems: ICSES 2022 2023 Feb 23 (pp. 797-816). Singapore: Springer Nature Singapore.

[208] Faily S. Engaging stakeholders during late stage security design with assumption personas. Information & Computer Security. 2015 Oct 12, 23(4):435-46.

[209] Ribeiro VV, Cruzes DS, Travassos GH. Moderator factors of software security and performance verification. Journal of Systems and Software. 2022 Feb 1, 184:111137.

[210] Farooq MS, Kalim Z, Qureshi JN, Rasheed S, Abid A. A blockchain-based framework for distributed agile software development. IEEE Access. 2022 Jan 27, 10:17977-95.

[211] Zhou E, Hua S, Pi B, Sun J, Nomura Y, Yamashita K, Kurihara H. Security assurance for smart contract. In2018 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS) 2018 Feb 26 (pp. 1-5). IEEE.

[212] Siavvas M, Kehagias D, Tzovaras D, Gelenbe E. A hierarchical model for quantifying software security based on static analysis alerts and software metrics. Software Quality Journal. 2021 Jun, 29(2):431-507.

[213] Al Sibahee MA, Abdulsada AI, Abduljabbar ZA, Ma J, Nyangaresi VO, Umran SM. Lightweight, Secure, Similar-Document Retrieval over Encrypted Data. Applied Sciences. 2021 Jan, 11(24):12040.

[214] Kukreja N. Winbook: a social networking based framework for collaborative requirements elicitation and WinWin negotiations. In2012 34th International Conference on Software Engineering (ICSE) 2012 Jun 2 (pp. 1610-1612). IEEE.

[215] Park KC, Shin DH. Security assessment framework for IoT service. Telecommunication Systems. 2017 Jan, 64:193-209.

[216] Hecker A, Riguidel M. On the operational security assurance evaluation of networked IT systems. InSmart Spaces and Next Generation Wired/Wireless Networking: 9th International Conference, NEW2AN 2009 and Second Conference on Smart Spaces, ruSMART 2009, St. Petersburg, Russia, September 15-18, 2009. Proceedings 2009 (pp. 266-278). Springer Berlin Heidelberg.

[217] Hosmer C, Hosmer C. IoT vulnerabilities. Defending IoT Infrastructures with the Raspberry Pi: Monitoring and Detecting Nefarious Behavior in Real Time. 2018:1-5.

[218] Mao W, Cai Z, Towsley D, Feng Q, Guan X. Security importance assessment for system objects and malware detection. Computers & Security. 2017 Jul 1, 68:47-68.

[219] Nyangaresi VO, Abduljabbar ZA, Ma J, Al Sibahee MA. Verifiable Security and Privacy Provisioning Protocol for High Reliability in Smart Healthcare Communication Environment. In2022 4th Global Power, Energy and Communication Conference (GPECOM) 2022 Jun 14 (pp. 569-574). IEEE.

[220] Liu Y, Gummadi KP, Krishnamurthy B, Mislove A. Analyzing facebook privacy settings: user expectations vs. reality. InProceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference 2011 Nov 2 (pp. 61-70).

[221] Tekeoglu A, Tosun AŞ. An experimental framework for investigating security and privacy of IoT devices. InIntelligent, Secure, and Dependable Systems in Distributed and Cloud Environments: First International Conference, ISDDC 2017, Vancouver, BC, Canada, October 26-28, 2017, Proceedings 1 2017 (pp. 63-83). Springer International Publishing.

[222] Fitzgerald B, Stol KJ. Continuous software engineering: A roadmap and agenda. Journal of Systems and Software. 2017 Jan 1, 123:176-89.

[223] Shahin M, Babar MA, Zhu L. Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices. IEEE access. 2017 Mar 22, 5:3909-43.

[224] Zahedi M, Rajapakse RN, Babar MA. Mining questions asked about continuous software engineering: A case study of stack overflow. InProceedings of the evaluation and assessment in software engineering 2020 Apr 15 (pp. 41-50).

[225] Rauf I, Petre M, Tun T, Lopez T, Lunn P, Van der Linden D, Towse J, Sharp H, Levine M, Rashid A, Nuseibeh B. The case for adaptive security interventions. ACM Transactions on Software Engineering and Methodology (TOSEM). 2021 Sep 28, 31(1):1-52.

[226] Nyangaresi VO, Abduljabbar ZA, Mutlaq KA, Hussain MA, Hussien ZA. Forward and Backward Key Secrecy Preservation Scheme for Medical Internet of Things. InHuman-Centric Smart Computing: Proceedings of ICHCSC 2022 2022 Nov 29 (pp. 15-29). Singapore: Springer Nature Singapore.

[227] Ouedraogo M, Khadraoui D, Mouratidis H, Dubois E. Appraisal and reporting of security assurance at operational systems level. Journal of Systems and Software. 2012 Jan 1, 85(1):193-208.

[228] Vivas JL, Agudo I, López J. A methodology for security assurance-driven system development. Requirements Engineering. 2011 Mar, 16:55-73.

[229] Jones RL, Rastogi A. Secure coding: building security into the software development life cycle. Inf. Secur. J. A Glob. Perspect.. 2004 Nov 1, 13(5):29-39.

[230] Gobinathan B, Mukunthan MA, Surendran S, Somasundaram K, Moeed SA, Niranjan P, Gouthami V, Ashmitha G, Mohammad GB, Shanmuganathan VK, Natarajan Y. A novel method to solve real time security issues in software industry using advanced cryptographic techniques. Scientific Programming. 2021 Dec 28, 2021:1-9.

[231] Pandya V, Saiyed A, Patel K. Recent Advancement in Fine-Grained Access Control and Secure Data Sharing Scheme for Distributed Environment. InEmerging Technologies for Computing, Communication and Smart Cities: Proceedings of ETCCS 2021 2022 Apr 20 (pp. 617-632). Singapore: Springer Nature Singapore.

[232] Nyangaresi VO, Ma J, Al Sibahee MA, Abduljabbar ZA. Packet Replays Prevention Protocol for Secure B5G Networks. InProceedings of Seventh International Congress on Information and Communication Technology: ICICT 2022, London, Volume 2 2022 Jul 27 (pp. 507-522). Singapore: Springer Nature Singapore.

[233] Bousquet A, Briffaut J, Caron E, Dominguez EM, Franco J, Lefray A, López O, Ros S, Rouzaud-Cornabas J, Toinard C, Uriarte M. Enforcing security and assurance properties in cloud environment. In2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC) 2015 Dec 7 (pp. 271-280). IEEE.

[234] Medeiros N, Ivaki N, Costa P, Vieira M. An approach for trustworthiness benchmarking using software metrics. In2018 IEEE 23rd Pacific Rim International Symposium on Dependable Computing (PRDC) 2018 Dec 4 (pp. 84-93). IEEE.

[235] Zhang M, de Carnavalet XD, Wang L, Ragab A. Large-scale empirical study of important features indicative of discovered vulnerabilities to assess application security. IEEE Transactions on Information Forensics and Security. 2019 Jan 29, 14(9):2315-30.

[236] Mohammed NM, Niazi M, Alshayeb M, Mahmood S. Exploring software security approaches in software development lifecycle: A systematic mapping study. Computer Standards & Interfaces. 2017 Feb 1, 50:107-15.

[237] Jimenez M, Papadakis M, Le Traon Y. Vulnerability prediction models: A case study on the linux kernel. In2016 IEEE 16th International Working Conference on Source Code Analysis and Manipulation (SCAM) 2016 Oct 2 (pp. 1-10). IEEE.

[238] Al Sibahee MA, Ma J, Nyangaresi VO, Abduljabbar ZA. Efficient Extreme Gradient Boosting Based Algorithm for QoS Optimization in Inter-Radio Access Technology Handoffs. In2022 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA) 2022 Jun 9 (pp. 1-6). IEEE.

[239] Khan RA, Khan SU, Alzahrani M, Ilyas M. Security assurance model of software development for global software development vendors. IEEE Access. 2022 May 26.

[240] Liu J, Wang Y, Zha L, Xie X, Tian E. An event-triggered approach to security control for networked systems using hybrid attack model. International Journal of Robust and Nonlinear Control. 2021 Aug, 31(12):5796-812.

[241] Schmitz C, Schmid M, Harborth D, Pape S. Maturity level assessments of information security controls: An empirical analysis of practitioners assessment capabilities. Computers & Security. 2021 Sep 1, 108:102306.

[242] Tan Y, Liu Q, Liu J, Xie X, Fei S. Observer-based security control for interconnected semi-Markovian jump systems with unknown transition probabilities. IEEE Transactions on Cybernetics. 2021 Feb 26, 52(9):9013-25.

[243] Ge H, Yue D, Xie X, Deng S, Dou C. A unified modeling of muti-sources cyber-attacks with uncertainties for CPS security control. Journal of the Franklin Institute. 2021 Jan 1, 358(1):89-113.

[244] Nyangaresi VO, Al Sibahee MA, Abduljabbar ZA, Ma J, Khalefa MS. Biometric-Based Packet Validation Scheme for Body Area Network Smart Healthcare Devices. In2022 IEEE 21st Mediterranean Electrotechnical Conference (MELECON) 2022 Jun 14 (pp. 726-731). IEEE.

[245] Zhu HS, Lin C, Liu YD. A programming model for sustainable software. In2015 IEEE/ACM 37th IEEE International Conference on Software Engineering 2015 May 16 (Vol. 1, pp. 767-777). IEEE.

[246] Wen SF, Shukla A, Katt B. Developing security assurance metrics to support quantitative security assurance evaluation. Journal of Cybersecurity and Privacy. 2022 Aug 10, 2(3):587-605.

[247] Shukla A, Katt B, Nweke LO, Yeng PK, Weldehawaryat GK. System security assurance: A systematic literature review. Computer Science Review. 2022 Aug 1, 45:100496.

[248] Ray A, Åkerberg J, Björkman M, Gidlund M. Towards security assurance for heterogeneous industrial networks. InIECON 2015-41st Annual Conference of the IEEE Industrial Electronics Society 2015 Nov 9 (pp. 004488-004493). IEEE.

[249] Weldehawaryat GK, Katt B. Towards a quantitative approach for security assurance metrics. InThe 12th International Conference on Emerging Security Information 2018.

[250] Sakthivel RK, Nagasubramanian G, Al-Turjman F, Sankayya M. Core-level cybersecurity assurance using cloud-based adaptive machine learning techniques for manufacturing industry. Transactions on Emerging Telecommunications Technologies. 2022 Apr, 33(4):e3947.

[251] Nyangaresi VO, El-Omari NK, Nyakina JN. Efficient Feature Selection and ML Algorithm for Accurate Diagnostics. Journal of Computer Science Research. 2022 Jan 25, 4(1):10-9.

[252] Adepu S, Mathur A. Assessing the effectiveness of attack detection at a hackfest on industrial control systems. IEEE Transactions on Sustainable Computing. 2018 Oct 30, 6(2):231-44.

[253] Sugumar G, Mathur A. Testing the effectiveness of attack detection mechanisms in industrial control systems. In2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C) 2017 Jul 25 (pp. 138-145). IEEE.

[254] Yu SY, Malawade AV, Chhetri SR, Al Faruque MA. Sabotage attack detection for additive manufacturing systems. IEEE Access. 2020 Feb 5, 8:27218-31.

[255] Kolberg J, Gläsner D, Breithaupt R, Gomez-Barrero M, Reinhold J, von Twickel A, Busch C. On the effectiveness of impedance-based fingerprint presentation attack detection. Sensors. 2021 Aug 24, 21(17):5686.

[256] Taormina R, Galelli S, Tippenhauer NO, Salomons E, Ostfeld A, Eliades DG, Aghashahi M, Sundararajan R, Pourahmadi M, Banks MK, Brentan BM. Battle of the attack detection algorithms: Disclosing cyber attacks on water distribution networks. Journal of Water Resources Planning and Management. 2018 Aug 1, 144(8):04018048.