Global Scientific JOURNALS

GSJ: Volume 12, Issue 12, December 2024, Online: ISSN 2320-9186
www.globalscientificjournal.com

SOFTWARE DEVELOPMENT AND THE SECURITY CHALLENGES IN DEVELOPING COUNTRIES

[1]Emmanuel Eturpa SALAMI (Department of Software Engineering, Confluence University of Science and Technology Osara, Kogi State Nigeria),
salami.emmanuel@iuokada.edu.ng

[2]Clement NWABUOKEI (Department of Software Engineering, Igbinedion University Okada, Edo State Nigeria)
nwabuokei.clement@iuokada.edu.ng

[3]Kayode Onimisi EKUNDAYO (Department of Computer Science. Confluence University of Science and Technology Osara, Kogi State Nigeria)
kekundayo6835@stu.ui.edu.ng

[4]Metiboba Precious BAMIYO (Department of Computer Science. Confluence University of Science and Technology Osara, Kogi State Nigeria)
Metibobapb@custech.edu.ng

© GSJ

ABSTRACT

Software development in developing countries provides opportunities for growth and innovation, as well as creating employment opportunities but it also comes with a unique set of challenges and these challenges that are often fueled by some factors which are either tangible or intangible. This study investigated some of the most important software development challenges faced by developing countries. Qualitative analysis based on Systematic Literature Review (SLR) was conducted to investigate some of the major challenges. The lack of technological infrastructure which makes it extremely difficult for software developers to create high-quality software products. Also, limited access to skilled workforce, Lack of funding is also a significant challenge that developers face, and language and cultural barriers also pose challenges to software development in developing countries. This is because software often needs to be localized to meet the needs of the local population, and this requires a deep understanding of the local language and culture. However, with the right investment in infrastructure, education, and training, local software industries can thrive and contribute to the growth of the African economy.

(**Keyword: Security, Threats, Compromise, Privacy)**

Introduction

Software development has become a crucial aspect of economic development in developing countries. It provides opportunities for growth and innovation, as well as creating employment

opportunities. However, with the increased use of software and technology comes security challenges. These challenges can be even more severe in developing countries, where resources and infrastructure may be limited. The Nigerian government and private sector have both played pivotal roles in fostering a conducive environment for tech innovation. Initiatives such as the National Information Technology Development Agency (NITDA) and the establishment of tech hubs like Co-Creation Hub (CcHub) in Lagos have provided essential support and resources for software developers. According to a report by the National Bureau of Statistics (NBS), the Information and Communication Technology (ICT) sector contributed 15% to Nigeria's GDP in 2021, underlining its importance to the economy (NBS, 2021). The rise of fintech companies has highlighted the potential of software development in Nigeria. Companies like Pay stack and Flutter wave have revolutionized online payments, attracting significant foreign investment. Paystack, for instance, was acquired by Stripe for $200 million in 2020, marking one of the largest startup acquisitions in Africa (TechCrunch, 2020). These success stories have inspired a new generation of Nigerian software developers to pursue careers in tech. Developing countries have made significant progress in software development over the past decade. However, they still lag behind developed countries, mainly due to limited resources and a shortage of skilled professionals. Despite these challenges, developing countries have taken steps to encourage software development and bridge the gap. The use of technology and software in developing countries has also brought about security challenges. These challenges include the threat of cyber-attacks, data breaches, and privacy violations. Developing countries are particularly vulnerable due to their limited infrastructure and resources to address these challenges. The impact of security challenges in developing countries can be significant. It can lead to loss of data, reputational damage, and financial loss. The consequences can be even more severe for small and medium-sized enterprises, which are the backbone of most developing economies.

© GSJ

Literature review:

Security Considerations in Software Development

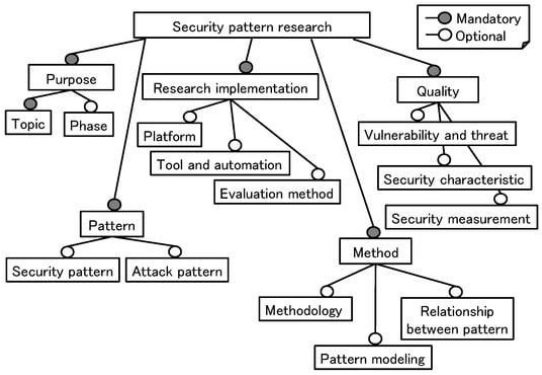
As software development continues to grow, so does the need for secure applications. Security should be a priority from the beginning of software development to minimize risks and prevent costly security breaches IBM (2019). The International Business Machine (IBM) corporation identify some important security considerations software developers need to keep in mind while developing software:

i. Implementation of authentication and authorization protocols to ensure only authorized users access the software.
ii. Encrypting sensitive data and user passwords to protect them from being accessed in plain text.
iii. Conduct regular vulnerability scans and penetration tests to identify vulnerabilities and weaknesses.
iv. Apply secure coding practices and guidelines to minimize the risk of attacks such as SQL injection and cross-site scripting (XSS).
v. Use up-to-date APIs and libraries that are less likely to contain security vulnerabilities.
vi. Integrate secure communication protocols to prevent any man-in-the-middle attacks that may occur during data transmission.

Developers should also be continuously learning about the latest security best practices and keeping up-to-date with current security trends. According to a study by IBM (2019), the average cost of a data breach is $3.92 million, making security an integral part of software development.

One of the key trends in software development security in recent years has been the shift towards DevSecOps, which involves integrating security into every stage of the software development lifecycle. This approach emphasizes collaboration between developers, operations teams, and security professionals, with the goal of building more secure software that can be deployed quickly and efficiently.

Hironori Washizaki et. al., (2021) observed that though there are different research techniques and approaches for security patterns in software development and these techniques are limited to some specific security patterns and it is not enough to accomplish the complexity of secure software systems properly. Therefore, they proposed a taxonomy for characterizing and classifying security pattern research Using an SLR to identify necessary facets, and the categorize and analyze 240 papers to clarify state-of-the-art and future directions of security pattern research in terms of 13 facets including topics and security characteristics.



Feature diagram of the taxonomy with reduction of several features for simplification (Hironori Washizaki et. al.,, 2021)

Five key features in the taxonomy proposed by (Hironori et. Al., 2021) stands out which are; *"Purpose",* which includes topics addressed by security pattern research and phases of the systems and software life-cycle. These are particularly important to help practitioners choose appropriate methods and tools against their needs, such as necessary supports and phases necessary to be secured by utilizing security pattern research. These are also helpful for researchers to identify each topic's and phase's maturity and envision necessary future efforts.
The second is *"Research Implementation"*, which consists of the platform to realize the pattern research results, whether the results are encapsulated or (semi-)automated as a tool and whether experiments or case studies are performed to evaluate the results relevant to the original research purpose. These are important to help practitioners choose easy-to-use or empirically validated methods on targeted platforms. These are also helpful for researchers to identify each research area's maturity and envision necessary future efforts.
The third is *"Quality",* which consists of items related to quality characteristics: threats and vulnerabilities toward a specific security problem; security characteristics in detail such as privacy, integrity, and availability; and whether a security measurement system is incorporated in order to detect changes in security by introducing or applying the research results. These are useful for choosing and carefully using specific methods and tools by understanding their impact on quality characteristics. These are also helpful for researchers to envision necessary future efforts concerning quality aspects, including security and privacy.
The fourth is *"Pattern",* which includes the types of patterns employed or addressed in the pattern research. Patterns that address security concerns can be classified into two types: security patterns and attack patterns. The former addresses both recurring security problems

and corresponding solutions from the viewpoint of defenders to security risks, while the latter addresses only security problems from the viewpoint of malicious attackers by detailing security risks. These are useful for choosing specific methods and tools against intended patterns, especially when practitioners examine specific patterns for use. These are also helpful for researchers to envision necessary future efforts in terms of each specific pattern.
The fifth is *"Method",* which includes the methodology, pattern modeling notations, and pattern relationships. These are useful for considering the adaptability of specific methods and tools to ongoing or intended development contexts, including development methodologies, modeling notations, and patterns considered to be used. These are also helpful for researchers to envision necessary future efforts concerning development contexts and security pattern combinations.

Software Design Security Issues

Software design security is an essential component of any software development process, as it ensures that the software complies with security standards and protects it from potential threats (OWASP, 2019). However, numerous software design security issues can arise during the development lifecycle that may compromise the software's security. One of the most common software design security issues is the lack of input validation. Failing to validate user input can allow attackers to inject malicious code into the software, leading to code execution and data theft (Fujitsu, 2019). Another common software design security issue is the use of weak encryption algorithms. Weak encryption algorithms can be cracked by sophisticated attackers, allowing them to gain unauthorized access to sensitive data (Fujitsu, 2019). Additionally, software design security issues can arise from the use of third-party libraries and frameworks. If these libraries or frameworks are outdated or have vulnerabilities, they can pose a significant threat to the software.

Schoenfield (2021) observed that, integrating security tools and processes into Agile workflows without disrupting the efficiency and collaboration of the team can be complex. In his recommendation stated that Developers and other stakeholders must stay updated on the latest security practices and threats, which requires an ongoing training on the culture of developing with security in mind. Agile methodology lays emphasis on rapid delivery and this can make some developer to take the shortcut in security measures if teams prioritize speed over security (Ambler, 2012). In addition, the iterative nature of Agile means that security must be reassessed with each new iteration, which can be resource-intensive. While Scrum focuses on embedding security practices within the Scrum framework to ensure that security is maintained throughout the development life cycle. Because of its iterative and collaborative nature, it facilitates continuous improvement and adaptability, making it well-suited for integrating security measures. A key aspect in Scrum is the incorporation of security requirements into the product backlog. These requirements can be treated as user stories, ensuring they are addressed in each sprint. Regular sprint reviews and retrospectives provide opportunities to assess security features and address any vulnerabilities promptly (Schwaber & Sutherland, 2019). But one significant challenge is balancing security with the rapid pace of development. Since Scrum focus on delivering functional increments quickly, it can sometimes lead to security being overlooked or deprioritized. Teams must be vigilant to ensure security does not become a secondary concern in the rush to complete sprints (Kerzner, 2022). Integrating security testing into the Scrum cycle without disrupting the workflow can be complete but automated security testing tools must be carefully selected and implemented to fit seamlessly into the continuous integration and delivery pipeline.

Lean methodology focuses on delivering value to the customer, which includes ensuring the security and reliability of the software. An approach that is used in Lean is to incorporate

security into the value stream mapping process and this is done by identifying potential security risks at each stage, which helps team members to proactively address vulnerabilities and integrate security controls efficiently. One significant challenge with Lean is aligning the goal of minimizing waste with the often resource-intensive nature of comprehensive security practices. Ensuring robust security without adding excessive overhead requires a delicate balance and careful planning (Kersten, 2018).

Abdul, et al., (2022) identified thirteen (13) critical cyber security challenges that also effect software development through the use of Systematic Literature Review (SLR) which are; Security issues/Access of Cyberattacks, Lack of Right Knowledge, Framework, Lack of Technical Support, Disaster Issues, Cost Security issues, Lack of Confidentiality and Trust, Lack of Management, Unauthorized Access issues, Lack of Resources, Lack of Metrics, Administrative Mistakes during Development and Lack of Quality, Liability, and Reliability. They recommend that vendor organizations need to give proper attention to these critical challenges to avoid any risk of failure by addressing these challenges.

Rafiq et al., (2022) proposes a new Security Assurance Model (SAM) for Software Development that is adaptable to all contemporary scenarios, emphasizing global software development (GSD) vendor companies. The model was built based on seven security assurance levels which are; Governance and Security Threat Analysis, Secure Requirement Analysis, Secure Design, Secure Coding, Secure Testing and Review, Secure Deployment, and Security Improvement and the results of the case studies indicate that the proposed SAM of Software Development helps measure the security assurance level of an organization. In addition, it can potentially serve as a framework for researchers to develop new software security measures. However, the practicality of adoption is key issues since these model was built using only 3 case studies on software development companies. An industrial empirical study was conducted to determine the impact of software security threats against each phase of SDLC by (Khan et al., 2022) fuzzy analytical hierarchy process (FAHP) was used in the study to prioritize the list of software security risks against the Software Development Life Cycle (SDLC) and the results and analysis of the study provide a ranked-based decision-making framework, which assists the practitioners in considering the most critical security risks on priority. It was discovered from the results that some factors were responsible for software which include; improper plan for secure requirement identification, inception, authentication, authorization, and privacy as well as lack of threat models updating, output validation, and lack of certification in the final release and archive, and spoofing ranked top in security risks of SDLC in GSD.

Threats Modeling Methodologies

Different threats models are used to identify potential threats to data, software applications, or system infrastructure and to also to detect and mitigate vulnerabilities. Threat modeling is essential for identifying and mitigating potential security risks in software applications therefore understanding the different models and techniques can help developers and security professionals improve the security of their systems. These threats modeling techniques all have their areas of strength and depending on the type of system that is being developed they can be used to test the system security against an attack.

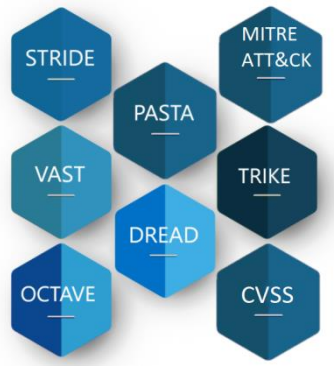


Figure 1.0 Threat Modeling Methodologies

These threats modelling methodologies as shown in figure 1.0 addresses different types of threats that a system might possibly encounter therefore developers could use any of the models to test their system for security.

1. STRIDE model: The STRIDE model is one of the most common approaches for identifying threats. It was developed by Loren Kohnfelder and Praerit Garg in 1999 for identifying potential vulnerabilities in systems. It is an acronym that stands for:

- **S**poofing
- **T**ampering
- **R**epudiation
- **I**nformation disclosure
- **D**enial of service
- **E**levation of privilege

© GSJ

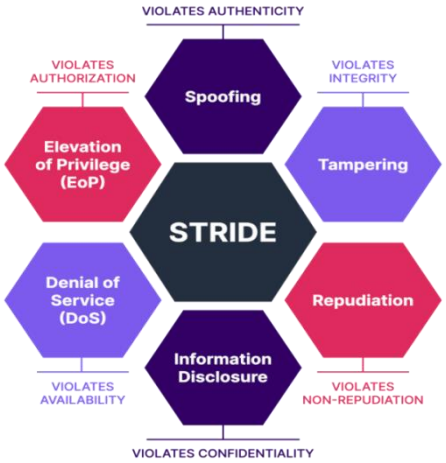


Figure 2.0: Stride Model (Ariel Jacob,2023)

Figure 2.0 shows the various types of threats that an application can be faced with and each of the components has specific area of focus. Spoofing has to do with violation of authentication security, Tampering is a violation of Data integrity, Repudiation violates Non-repudiation, Information Disclosure has to do with violation of Confidentiality, Denial of service (DoS) is a violation of availability. According to Microsoft, this model can be used to identify threats for any software application and can be used in a wide range of industries and scenarios (Microsoft, 2021). The model is also used in information security and software development

to identify and categorize potential threats to a system (Jacob, 2023). However, the complexity and the steep learning curve it presents to developers is a challenge also, mastering the nuances of each threat category requires significant training and expertise. This can be particularly daunting for teams that lack specialized security knowledge. For instance, understanding the intricacies of how tampering might be executed or identifying subtle information disclosure threats requires deep technical and domain-specific insights (Shostack, 2014). The potential for over-reliance on the model, leading to a checklist mentality. While Stride offers a structured approach, it may not cover every possible threat, especially in highly dynamic and complex systems. This can result in a false sense of security, where teams believe they have covered all bases simply by addressing the six categories. The model also tends to focus heavily on traditional software systems and might not be as effective for newer paradigms like microservices or cloud-native applications, where the threat landscape can differ significantly (Myagmar, Lee, & Yurcik, 2005).

2. PASTA model

PASTA stands for "Process for Attack Simulation and Threat Analysis." It is a risk-centric threat modeling technique that involves seven phases. These phases are:

- Define objective
- Define Technical scope
- Application decomposition
- Analyze threats
- Vulnerability and weakness analysis
- Model attacks
- Risk impact analysis

Figure 3.0 PASTA Threat Model (Jain, 2021)

The PASTA Threats Model in figure 3.0 is use in identifying threats and prioritizing them based on risk set by the assessment of cost, impact and feasibility.

Defining objectives makes the end goal clearer and shifts the entire focus to only the relevant assets to be modeled for threats. While the Technical scope marks the boundaries for the application and list out the application dependencies from the environment. Application decomposition define and evaluate all the possible assets along with their data which could be in either transit or be at rest. to identify which applications are susceptible to which threat vectors resulting in threat enumeration. One primary challenge with this model is it complexity and resource-intensive nature. It involves multiple stages, each requiring detailed analysis and extensive documentation. This complexity can be daunting for teams with limited experience

in threat modeling, leading to potential inaccuracies or incomplete threat assessments (Scandariato, et al., 2016). Also, the need for specialized knowledge and skills. Also integrating PASTA into Agile or Lean development processes can be problematic. These methodologies prioritize rapid delivery and iterative progress, this threat model thorough, stage-based approach can slow down development cycles. This misalignment can lead to resistance from development teams and potential neglect of the threat modeling process (Kalle, et, al., 2020). While PASTA offers a robust framework for threat modeling, its complexity is that it requires having a specialized knowledge which makes the integration with Agile or Lean, and maintenance to be a significant hurdles for effective implementation.

3. VAST model

VAST (Visual, Agile and Simple Threat modeling) is an approach that emphasizes visual communications to help users better visually understand the architecture. This approach addresses the traditional lack of clarity in other models as shown in figure 4.0.

Figure 4.0 VAST Threat Model (EC-Council.org)

VAST modeling method includes visualizing the Application, categorizing Threats, and assigning Countermeasures. It aims to apply security at the early development stages of any application. This helps to save costs and efforts when fixing possible security problems later on. A major challenge is the potential oversimplification. While VAST aims to make threat modeling accessible and straightforward, this simplicity can sometimes lead to an underestimation of complex threats. Important security nuances may be overlooked, resulting in incomplete threat assessments (Shostack, 2014). It's heavy reliance on visual representations can be problematic. Even though visual tools are beneficial for understanding and communication, they may not capture all details required for comprehensive threat analysis. This can lead to gaps in security measures if teams rely solely on visual models without delving into more detailed, textual analysis. Integrating VAST into Agile workflows also presents challenges. Because Agile focuses on rapid iterations, this can make it difficult to conduct thorough threat modeling regularly. Ensuring that threat modeling keeps pace with fast development cycles requires significant effort and coordination, potentially slowing down the Agile process (Kersten, 2018). Development team requires a certain level of expertise understanding of both the visual modeling tools and the underlying security principles to be effective in using this threat model when developing software.

4. TRIKE model

The TRIKE model is used when security auditing from a risk management perspective is required. It is a fusion of two models which are the Requirement model and Implementation model. While the Requirement explains the security characteristics and assigned acceptable level of risk, the Implementation aspect uses a Data Flow Diagram to illustrate how data flow and user actions within the system. The complexity of this framework involves detailed modeling of both assets and threats, which requires a significant amount of time and expertise

and it can be overwhelming for teams lacking extensive security knowledge, leading to incomplete or inaccurate threat assessments (Shostack, 2014). Integration with Agile or fast-paced development processes poses another challenge. Also, because it requires a thorough and detailed approach, it can be time-consuming, potentially slowing down development cycles and causing friction within Agile teams that prioritize rapid delivery and iterative progress (Kersten, 2018).

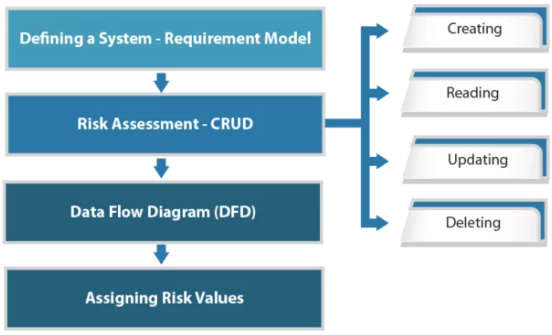


Figure 4.0 TRIKE Threat Model (EC-Council.org)

5. DREAD model

Dread (Damage, Reproducibility, Exploitability, Affected users, Discoverability) model is used to assess, analyze and find the probability of risk by rating the threats as described in figure 5.0. One significant challenge is subjectivity in scoring and it relies on qualitative assessments for each category, which can vary greatly between evaluators. This subjectivity can lead to inconsistent threat prioritization, making it difficult to achieve a uniform security posture and the issue of the potential for oversimplification (Shostack, 2014). Integrating this model into Agile where prioritization on rapid delivery is the concern of the team can slow down the development process, leading to potential resistance from team members (Kersten, 2018).

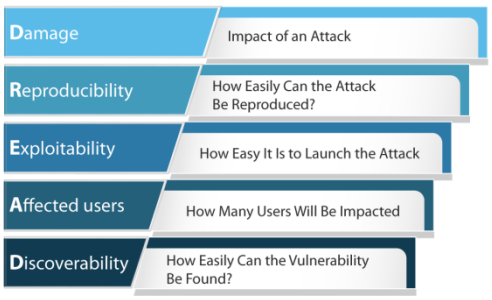


Figure 5.0 DREAD Threat model (EC-Council.org)

6. OCTAVE model

OCTAVE (Operationally Critical Threat, Asset, and Vulnerability Evaluation) model. It is an model that is used in managing risk to IT assets. It provides a more clearer information on assets that are risk and help the organization to design strategy that can protect, reduce or eliminate the risk to the assets. The aim of the model as shown in figure 6.0 is to Identify risk to crucial assets, Evaluate the impact it will have on the assets, Understand the operational Risk Tolerance of the organization, Identify the types of threat and vulnerabilities that the organizations assets are exposed to and then initiate actions to Mitigate those risks. A prominent challenge with OCTAVE is its complexity and resource intensity. The process can be time-consuming and may strain the resources of smaller organizations or teams with limited

experience in threat modeling and it also requires significant organizational commitment. Integrating OCTAVE with Agile or other rapid development methodologies is also problematic. OCTAVE's thorough and methodical approach can slow down development cycles, conflicting with Agile's emphasis on speed and flexibility (Kersten, 2018).

Figure 6.0 OCTAVE Threat Model (EC-Council.org)

These models are essential for identifying and mitigating potential security risks in software applications and understanding the different models and techniques can help developers and security professionals improve the security of their systems.

Security Issues in Software Development in Africa

Software development is a critical sector in Africa, and many countries are leveraging technology to innovate and enhance their economies. In Africa is faced with numerous challenges, which hinder its growth and development. According to Yaw Anokwa (2018) software developer do not have enough competent programmers to carry out complex projects, and this limits the potential in that field. Lack of adequate infrastructure, such as poor internet connectivity and power outages. This affects the delivery of services and exchange of information, causing delays and downtime (Anokwa, 2018). Despite some of the positive trends that the African continents have recorded in the areas of rapid growth, driven by increasing internet penetration, a burgeoning tech-savvy youth population, and rising investments in the technology sector, the region faces significant security challenges in software development. These challenges include inadequate cybersecurity infrastructure, lack of skilled professionals, regulatory weaknesses, and evolving cyber threats. One primary security issues in software development in Africa is the inadequate cybersecurity infrastructure. Many African countries lack the necessary infrastructure to protect against cyber threats effectively. This inadequacy stems from limited investments in cybersecurity technologies and insufficient awareness of the importance of cybersecurity among stakeholders (Adelola, Dawson, & Batmaz, 2020). Consequently, software developed in such environments is often vulnerable to attacks, as basic security measures may be overlooked or inadequately implemented. The shortage of skilled cybersecurity professionals is another critical challenge which might not be limited to the African countries only. There is a significant skills gap in the cybersecurity domain, with a dearth of trained experts to address the growing security needs of the software development industry (Kshetri, 2019). This skills gap affects software applications being developed in the continent where many software is developed without adequate security considerations, thereby increasing the risk of vulnerabilities and exploitation. The regulatory frameworks in most African countries are weak and also there is an inconsistent enforcement

of cybersecurity laws which contribute to the security challenges in software development. It is either the regulatory frameworks are outdated or insufficiently creating an environment where cybercriminals can operate with relative impunity (Kshetri, 2019). This regulatory gap implies that software developers may not prioritize security, knowing that the consequences of security breaches are minimal. According to a report by Check Point Software Technologies (2021), Africa experiences a higher average of cyber attacks per week compared to the global average. These attacks include ransomware, phishing, and malware, which target both individuals and organizations. The evolving nature of these threats makes it challenging for software developers to keep up with the latest security measures and protect their applications effectively. Economic constraints also play a significant role in the security issues faced by software developers in Africa. Many startups and small businesses operating in the technology sector may lack the financial resources to invest in robust security measures (OECD, 2020). As a result, these companies may prioritize functionality and speed to market over security, leaving their software applications vulnerable to attacks. Additionally, the high cost of cybersecurity solutions and services further exacerbates this issue, making it difficult for businesses to implement comprehensive security measures. The lack of awareness about the importance of security in the software development lifecycle leads to the creation of applications that are not built with security in mind which is further compounded by the absence of standardized security training programs and certifications for developers in the region (Grobler et al., 2019).

Recent Incidents of Security Breach

A notable case highlighting the security issues in software development in Africa is the data breach incident at MTN Nigeria, one of the largest telecommunications companies on the continent. In 2019, MTN Nigeria suffered a significant data breach that exposed sensitive customer information, including personal identification details and financial data (Business Day, 2019). This incident underscored the vulnerabilities in software systems and the need for robust security measures in protecting customer data.

In 2020, several Kenyan banks were targeted by a sophisticated cyber-attack that resulted in the theft of millions of shillings. The attackers exploited vulnerabilities in the banks' software systems, gaining unauthorized access to customer accounts and siphoning off funds (Kshetri, 2020). This incident highlighted the critical need for secure software development practices in the financial sector, where the stakes are particularly high.

Security Threats in Software Development: Case Studies from Five African Countries

An exploration of security threats in software development faced by five African countries in recent times: Nigeria, Kenya, South Africa, Ghana, and Egypt. Each case study highlights the unique challenges and incidents that have impacted these nations. Nigeria as one of Africa's largest economies and tech hubs, has faced numerous security threats in software development particularly with the rise of fintech, mobile banking, and e-commerce has made the country a target for cybercriminals. In 2020, several Nigerian fintech startups faced Cyber-attacks that exploited weaknesses in their software applications. These attacks resulted in financial losses and raised concerns about the security of digital financial services (Kshetri, 2020). The incidents prompted regulators and companies to enhance their security protocols and invest in better cybersecurity infrastructure. Kenya, known for its innovation in mobile payments and tech startups, has also grappled with security threats in software development. In 2020, Kenyan banks were targeted by a sophisticated Cyber-attack that led to the theft of millions of shillings. Attackers exploited software vulnerabilities to gain unauthorized access to customer accounts, siphoning off funds. The incident exposed significant weaknesses in the banking software systems and emphasized the critical need for secure software development practices in the

financial sector (Daily Nation, 2020). Kenya's mobile payment systems, such as M-Pesa, have revolutionized financial transactions in the region but have also become prime targets for cybercriminals. In recent years, there have been multiple instances where vulnerabilities in mobile payment software were exploited, leading to unauthorized transactions and financial losses. These incidents highlight the continuous need for rigorous security measures and regular software updates to protect user data (Gikandi & Bloor, 2021). South Africa, a leading tech hub in Africa, has faced numerous cyber threats affecting software development across various sectors. In 2021, several South African companies, including major retailers and healthcare providers, were hit by ransomware attacks. These attacks encrypted critical data and demanded ransom payments for decryption keys. The attacks exploited vulnerabilities in the companies' software systems, causing significant disruptions and financial losses (Govender, 2021). The incidents highlighted the importance of robust cybersecurity measures, including regular software patching and employee training. In 2020, the South African government's IT infrastructure faced a major breach that compromised sensitive data. Attackers exploited vulnerabilities in the government's software systems to gain unauthorized access to personal information of citizens, including identification numbers and addresses (Africanews, 2020). In 2021, several financial institutions in Ghana were targeted by Cyber-attacks that exploited software vulnerabilities. These attacks aimed at accessing customer data and siphoning off funds. The breaches exposed weaknesses in the security measures of these institutions, prompting regulatory authorities to mandate stricter cybersecurity protocols and regular audits (Mensah, 2021). Ghana's e-government initiatives, aimed at digitizing public services, have also faced security threats. In 2020, the e-government platform experienced a Cyber-attack that compromised personal information of citizens. The attackers exploited vulnerabilities in the software systems used for e-government services, highlighting the need for enhanced security measures in government projects (Addo, 2020). In 2020, Egypt's healthcare sector faced a significant Cyber-attack that compromised patient data. Attackers exploited software vulnerabilities in hospital management systems to gain access to sensitive medical records. The breach highlighted the critical need for secure software development practices in the healthcare sector to protect patient information (El-Sayed, 2020). Egypt's banking sector has also been targeted by cybercriminals. In 2021, several banks experienced Cyber-attacks that exploited vulnerabilities in their online banking software. These attacks resulted in unauthorized transactions and financial losses, prompting banks to invest in better cybersecurity infrastructure and regular security assessments (Farouk, 2021).

Recently, there have been some concerning issues with security in software development. One major concern is the prevalence of vulnerabilities in popular programming languages. A recent report found that almost two-thirds of websites contain vulnerabilities that can be exploited by hackers (veracode, 2020). Another issue is the increasing complexity of software systems. As software grows in complexity, it becomes harder to identify vulnerabilities and security flaws. This can lead to serious security breaches if not addressed properly. There has been criticism of the lack of attention paid to security in the software development process. Many developers focus on adding features and improving functionality, while security is often an afterthought. This can lead to vulnerabilities being introduced into software during the development process, which can then be exploited by attackers (OWASP, 2017). It is essential that software developers take security seriously and prioritize it in the development process. This includes using secure coding practices, testing code for vulnerabilities, and staying up to date with the latest security threats and best practices (OWASP, 2017).

The Way Forward

To overcome these security challenges, developing countries must invest in cybersecurity infrastructure and capacity building. This would involve training professionals on the latest security technologies and practices and developing policies and regulations to govern their use.

Additionally, strengthening partnerships with developed countries can also help in addressing the security challenges faced. Developers must prioritize software design security during the development life cycle to ensure that the software meets security standards and is resistant to various attacks. Software developers in Africa must ensure that they apply global best practice in area of security in software development such as Secure coding, Code review, Testing, Access control, and Encryption. They must implement appropriate security measures such as input validation, encryption, and secure coding practices. Additionally, utilizing up-to-date third-party libraries and frameworks can help reduce the likelihood of security vulnerabilities. Implementing a security testing regime during the development life-cycle can also help identify potential security issues before the software is released. This approach can provide insight into how the software behaves in various attack scenarios.

Conclusion

Software design security is crucial for protecting the integrity and confidentiality of sensitive data. By implementing appropriate security measures, utilizing up-to-date libraries and frameworks, and conducting thorough security testing, developers can reduce the likelihood of software design security issues. The challenges facing software development in Africa are numerous and complex, but it is essential to address them if the goal of bridging the technological divide and accelerating development must be achieved. By investing in infrastructure, education, and funding, we can foster a conducive environment for software development in Africa.

© GSJ

References

Abdul. W. Khan, S. Zaib, F. Khan, I. Tarimer, J. T. Seo and J. Shin, (2022). "Analyzing and Evaluating Critical Cyber Security Challenges Faced by Vendor Organizations in Software Development: SLR Based Approach. *Institute of Electrical Electronic Engineering journal* Access, vol. 10, pp. 65044-65054, 2022, doi: 10.1109/ACCESS.2022.3179822.

Adelola, T., Dawson, R., & Batmaz, F. (2020). An exploratory study into the cybersecurity practices of micro-finance institutions in Nigeria. *International Journal of Information Management*, *50*, 142–148.

Alao-Owuuna, I., & Adediwura, O. M. (2023). Information Communication Technology and Its Impact on the Economic Growth of Nigeria. *Asian Journal of Economics, Business and Accounting*, *23*(8), 52–63. https://doi.org/10.9734/ajeba/2023/v23i8955

Ambler, M., & Scott, W. (2012). *Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise* (p. 544). Pearson Education, IBM Press.

Anokwa, Y. (2018). The challenges of software development in Africa. Retrieved May 14, 2024, from https://www.computerworld.com/article/3228151/the-challenges-of-software-development-in-africa.htm

Asongu, S. (2014). Software piracy, inequality and the poor: evidence from Africa. *Journal of Economic Studies*, *41*(4), 526–553. https://doi.org/10.1108/jes-10-2012-0141

Bank of Ghana. (2020). Bank of Ghana cyber & information Security directive. Retrieved June 24, 2024, from bog.gov.gh website: https://www.bog.gov.gh/reg_directives/bank-of-ghana-cyber-information-security-directive/

Business DAY. (2019). MTN Nigeria suffers data breach exposing customers' personal details. Retrieved June 15, 2024, from businessday.ng/MTN Suffers data breach exposing customer personal Data website: https://businessday.ng

Business Tech. (2019). South African banks hit by online attacks. Retrieved July 28, 2024, from https://businesstech.co.za/ website: https://businesstech.co.za/news/banking/348989/south-african-banks-hit-by-online-attacks/

Cao, Y., Li, X., & Q, W. (2017). Software Development Security Challenges under the Agile and DevOps Mechanisms. *Frontiers in Science and Engineering.*

Check Point Software technologies. (2021). Ransomware Exploits and Supply Chain Attacks Lead the Cyber Trends in the First Half of 2021. Retrieved from https://pages.checkpoint.com/cyber-attack-2021-trends.html website: https://pages.checkpoint.com

Fin24. (2017). SARS confirms taxpayer data breach. Retrieved March 16, 2024, from https://www.news24.com/fin24 website: https://www.fin24.com

GhanaWeb. (2019). Cyber-attacks on Ghana government websites rising. Retrieved June 28, 2024, from ghanaweb website: https://www.ghanaweb.com/GhanaHomePage/NewsArchive/Cyber-attacks-likely-to-rise-786047

Grobler, M., Jansen van Vuuren, J., Zaaiman, J., & Jansen van Vuuren, H. (2019). Cybersecurity in Africa: Challenges and solutions. *Journal of Information Warfare*, *18*(4), 45–56.

Hironori, W., Tian , X., Natsumi , K., Yoshiaki , F., Hideyuki, K., & Takehisa, K. (2021). Systematic Literature Review of Security Pattern Research. *Information*, *12*(1), 36. https://doi.org/10.3390/info12010036

IBM. (2020). Cost of a Data Breach Report 2020. Retrieved October 27, 2024, from https://www.ibm.com website: https://www.ibm.com/security/digital-assets/cost-data-breach-report/1Cost%20of%20a%20Data%20Breach%20Report%202020.pdf

IBM. (2021). Cost of a Data Breach Study. Retrieved from IBM website: https://www.ibm.com/security/data-breach

Kalle , R., Ruohonen, J., Holvitie, J., & Hyrynsalmi, S. (2020). Security in agile software development: A practitioner survey. *Science Driect*, *2020*(7), 6–9. https://doi.org/10.1016/S1353-4858(20)30078-7

Kene-Okafor, T. (2021). Paystack expands to South Africa seven months after Stripe acquisition. Retrieved from TechCrunch website: https://techcrunch.com/2021/05/06/paystack-expands-to-south-africa-seven-months-after-stripe-acquisition/

Kersten, M. (2018). *Project to Product How to Survive and Thrive in the Age of Digital Disruption with the Flow Framework* (p. 272). Portland OR 97210, Canada: IT Revolution Press. Retrieved from https://www.google.com.ng/books/edition/Project_to_Product/Hax-DwAAQBAJ?hl=en&gbpv=0

Kerzner, H. (2022). *Innovation Project Management Methods, Case Studies, and Tools for Managing Innovation Projects* (p. 624). Wiley. Retrieved from https://www.google.com.ng/books/edition/Innovation_Project_Management/fL6bEAAAQBAJ?hl=en&gbpv=0 (Original work published 2019)

Khan, R. A., Khan, S. U., Akbar, M. A., & Alzahrani, M. (2022). Security risks of global software development life cycle: Industry practitioner's perspective. *Journal of Software: Evolution and Process*. https://doi.org/10.1002/smr.2521

Kshetri, N. (2019). Cybercrime and Cybersecurity in Africa. *Journal of Global Information Technology Management*, *22*(2), 77–97. https://doi.org/10.1080/1097198X.2019.1603527

Ludovic, L. (2019). Financing software development in Africa. Retrieved March 24, 2024, from https://www.findevgateway.org website: https://www.findevgateway.org/blog/financing-software-development-africa

Mike Da Silva, Maxime Puys, Pierre-Henri Thevenon, Mocanu, S., & Nkawa, N. (2023). Automated ICS template for STRIDE Microsoft Threat Modeling Tool. *HAL (Le Centre Pour La Communication Scientifique Directe)*. https://doi.org/10.1145/3600160.3605068

Mousannif, H., & Al Moatassime, H. (2017). Security Challenges in Software Defined Networking. *Journal of Cyber Security and Mobility*.

OECD. (2020). *Financing SMEs and Entrepreneurs 2020: An OECD Scoreboard* (p. 224). OECD Publishing.

OWASP. (2017). OWASP Top 10 Application Security Risks - 2017. Retrieved June 10, 2024, from https://owasp.org/ website: https://owasp.org/www-project-top-ten/2017/Top_10

R. A. Khan, S. U. Khan, M. Alzahrani and M. Ilyas, (2022). Security Assurance Model of Software Development for Global Software Development Vendors. *Institute of Electrical Electronic Engineering IEEE Xplore* Access, vol. 10, pp. 58458-58487, 2022, doi:10.1109/ACCESS.2022.3178301.

Scandariato, R., Walden, J., & Stuckman, J. (2016). The Effect of Dimensionality Reduction on Software Vulnerability Prediction Models. *IEEE Transactions on Reliability*, *66*(1), 17–37. https://doi.org/10.1109/TR.2016.2630503

Schoenfield, B. S. E. (2021). *Applied Security Architecture and Threat Models* (p. 440). Taylor & Francis Group. (Original work published 2015)

Schwaber, K., & Sutherland, J. (2019). *A Scrum Book: The Spirit of the Game* (p. 572). Pragmatic Bookshelf. Retrieved from https://www.google.com.ng/books/edition/A_Scrum_Book/GWbcDwAAQBAJ?hl=en

Shaoqiu, L. (2024). Research on After-Sales Service Strategies for Cross-Border E-Commerce Based on Digital Technology . *Journal of Emerging Trends in Computing and Information Sciences*. https://doi.org/10.12677/ECL.2024.131053

Shohreh , H., Sampsa, R., Samuel, L., & Jari-Matti, M. (2018). Diversification and obfuscation techniques for software security: A systematic literature review. *Information and Software Technology*, *104*, 72–93. https://doi.org/10.1016/j.infsof.2018.07.007

Shostack, A. (2014). *Threat Modeling: Designing for Security* (pp. 1–33). ieeexplore. Retrieved from https://ieeexplore.ieee.org/servlet/opac?bknumber=9932141

Sneha Leela Jacob, & Parveen Sultana Habibullah. (2024). A Systematic Analysis and Review on Intrusion Detection Systems Using Machine Learning and Deep Learning Algorithms. *Journal of Computational and Cognitive Engineering*. https://doi.org/10.47852/bonviewjcce42023249

Subhrendu, G. N., & Ohri, P. (2022). Software-Defined Networking Security Challenges and Solutions: A Comprehensive Survey. *International Journal of Computing and Digital Systems*, *12*(2210-142X). https://doi.org/10.12785/ijcds/120131

Veracode. (2020). *State of Software Security Flaw Frequency by Language*. Retrieved from https://www.veracode.com/sites/default/files/pdf/resources/infosheets/state-of-software-security-volume-11-flaw-frequency-by-language.pdf