ISSN NO: 2249-3034

SURVEYOF REST WEB SERVICES API AND SUSTAINABLE SECURITY OF WEB APPLICATION

Dr.RAMACHANDRA C G
Computer Engineering Department
SIES Graduate School of Technology, Nerul, Navi Mumbai

Abstract-Businesses are increasingly deploying their services on the web, in the form of web applications, SOAP services, message-based services, and, more recently, REST services. In this paper, author has discussed the survey of various REST Web services API and study of sustainable security in web application. Ensuring sustainable security of web applications for minimizing security breaches and enhancing users' trust and satisfaction is the foremost priority of all security experts and web developers. However, sustainable security is multidimensional, emergent, and an irreducible concept. Also, designing sustainable security bof web application is a complex process because it is a multi-attribute approach which is based on the users' needs and organization's policies.

Keywords–REST Web Services API, Web Security.

1. INTRODUCTION

The web Application Programming Interfaces (APIs) has been growing rapidly. Several studies point out that developers have moved from Simple Object Access Protocol (SOAP) or Remote Procedure Call (RPC) to deploying Representational State Transfer (REST) web services, as the means for consumers to use their services. This is corroborated by major websites like Google, Facebook, or Twitter, which are now deploying REST services to provide easy access to their valuable data resources, while promoting their businesses [1]. The REST architecture was introduced in the year 2000, by Thomas Fielding, and is based on the principles that support the World Wide Web [16]. In summary, according to the REST principles [16], REST interfaces rely exclusively on Uniform Resource Identifiers (URI) for resource detection and interaction, and usually on the Hypertext Transfer Protocol (HTTP) for message transfer [3], [6]. A REST service URI only provides location and name of the resource, which serves as a unique resource identifier. The predefined HTTP verbs are used to define the type of operation that should be performed on the selected resource (e.g., GET to retrieve, DELETE to remove a resource). Possibly due to HTTP's features (which fit the REST architecture rather well), long-term presence, and general understandability, REST has become a de facto standard way for offering a service on the Web [3], [8]. Despite this, REST is merely an architectural style, provided without standard specifications. This implies that several decisions have to be made by developers when exposing service APIs, which may result in diverse APIs and, in some cases, in poor design decisions (e.g., using a single HTTP verb for retrieving or deleting a resource). These decisions will impact the client-side developer, that must adapt to the specific style being used, and may even affect the provider (e.g., when a poorly maintainable service is deployed). Server-side developers must define how an API should be exposed (e.g., which URI design schema to use), which characteristics it should possess (e.g., which output formats are supported) or how the documentation is provided (e.g., created with documentation generation software). For instance, this last item is problematic because without standard means for documenting APIs, the tendency will be to use text to describe the API (many times in natural language), which may be diverse in form, structure, or depth and is obviously a problem for client developers [3]. The literature also suggests some heterogeneity in how HTTP features are used, with some being widely adopted (e.g., HTTP verbs, status codes) and others tending to be ignored (e.g., HTTP headers) [3], [4]. Also highly discussed is the adoption of the Hypermedia As The Engine Of Application State (HATEOAS) principle of the REST architecture, which is rarely used. Due to the abovementioned reasons, it is a common belief that most online services available online claim to be REST services without truly following the REST principles, and present severe technical inconsistencies, diverse designs, and operating modes. Thus, some web services expose Web APIs very close to the original style described by Fielding [5], whereas others simply change their existing RPC-style API to being directly accessible via HTTP [2]. To what extent this currently happens is not very clear, and there is the need of using new data as a means to provide an additional data point that supports or disputes findings of previous work.

In the nest section author has discussed sustainable security in web application.Sustainable security is about building sustainable and secure web application while ensuring that every developer who is involved in the design process understands the importance of security [17], [18]. For achieving sustainable security, it is imperative to ensure that best practices are enlisted in considering the security assessment at the very beginning of the web application development life cycle [19]. The last few

ISSN NO: 2249-3034

decades of the 20th century saw the growth of sustainability in web application of all institutional activities [20], [21]. According to a report by Microsoft, there has been a noticeable increase in the practitioners' efforts to reform their whole security models to integrate sustainability during the early stages of web application development [22]. However, designing such web applications is a complex task and has led to failures. In fact, the authors of the present study reviewed few reports of security failures of web applications for data breaches [23]. According to these reports, important data such as phone number, names and addresses of 5001 students were stolen. Few reports cited that web applications such as the entrance examination web application had been attacked for more than 50 times during the examination [24]. These statistics show a marked need for sustainable security, more particularly, in a web application. Constructing sustainable-security design of web application for an institution is a decision making process that involves several decision makers. Multi-criteria decision-making process plays an important role in collating different decisions of the practitioners in one frame [25]. A. Mardani et al. [26] states that Multi Criteria Decision Making Methods (MCDM) is a real approach for solving difficulties related to sustainable issues with multiple resources. Moreover, these methods cover a wide collection of reasonably distinct methods, in which Fuzzy MCDM is a widespread methodology. Further, Fuzzy MCDM also removes the inadequacies of MCDM approaches. Garg et al. [27] documented the barriers for adoption of sustainable approaches through AHP method. However, most of the research did not discuss the attributes involved in sustainable-security. Therefore, it is imperative to apply a more effective approach for estimating sustainable-security and enabling the practitioners to know the preferences of factors while guaranteeing sustainable-security of the web applications developed by them.

2. RELATED WORK

We identified the following two main groups of research work that provide information regarding web APIs: i) Works that empirically analyze a set of web services for specific API characteristics or features; and ii) Works that provide information about quality attributes of web services, including open issues and challenging aspects. In what concerns empirical analyses of web services, the work in [1] is an interesting case that analyses 222 publicly available web API documentations, selected from programmableweb.com, against 20 features. Results show that RESTful web services had been relatively widely adopted by 2010, although the authors emphasize that developers tend to disregard the REST principles, which we confirm in this work in a larger-scale study and almost a decade later.

In [2], twenty REST web services, selected among the APIs with higher number of mashups from programmableweb.com, are analyzed against 17 features. Results show that almost none of the services is actually truly RESTful, despite the small dataset.

The authors in [3] analyzed the 45 most popular web APIs of the programmableweb.com repository using the number of mashups as the sorting metric and against 17 features. Results show diversity in the adoption of REST principles or in common design decisions, which is attributed to missing standards regarding REST. The rather small dataset and the time passed since then make it necessary to take another up-to-date look on the current state of web APIs.

Authors in [4] aimed to understand the meaning of micro services for practitioners by questioning 42 companies. The focus of the work is on micro services, how they are technically implemented, and which difficulties are present. Main findings show that two thirds of the companies use external services, but also about two thirds of the services used are internally developed and operated. REST and HTTP are quite present, while SOAP rarely appears. JSON is more common than XML and the most often used programming languages for implementing the companies' internal REST services are Java, JavaScript, C#, Python and Ruby. The study analyzes a certain services niche, while we aim for a broader study.

The authors in [5] analyze the degree of compliance with the REST architectural principles, from a point of view of mobile applications. 78GB of data logs of HTTP requests collected from a large mobile operators are used to identify patterns and matched against 26 best practices for web services and the Richardson Maturity Model [6]. The largest part of their analyzed dataset complies with level 2 of this model, whereas only a few hosts reach level 3. This study is only based on the analysis of live HTTP requests and does not examine the corresponding API itself and respective documentation.

The services of three popular cloud providers are compared against a catalog of 73 best practices for REST APIs design in [7]. Results show that these cloud providers have reached an acceptable level of maturity, even though they follow only half to two thirds of the catalog guidelines.

The work in [8] analyzes 286 Swagger API description documents and provide a framework for a structural analysis of REST APIs to identify the main characteristics and deficits. With roughly a third of the 286 APIs being hosted by Google and by analyzing only Swagger documents, the authors focus on a particular niche. In [9] 500 popular apps and 15 popular services for the android ecosystem are analyzed, with the results showing that application developers prefer official SDKs for accessing these services over simple HTTP clients. It is a small set of 15 services (and with focus on android applications), thus it is beneficial to have a broader view on this matter.

The authors in [10] analyze 18 popular web services according to 12 linguistic patterns and anti-patterns by

ISSN NO: 2249-3034

semantically analyzing the URIs of web services. Results include the detection of syntactical URI design issues, although the API designers tend to use adequate resource names and no verbs in URIs (a REST best practice). The authors do not analyze other REST-relevant features, such as the use of HTTP headers.

Non-functional aspects (e.g., payment plans and business models) of 69 web services extracted from market.mashape.com and programmableweb.com are analyzed in [11] with the goal of identifying requirements of a governance model of realistic RESTful APIs. The authors observe a wider expression of API limitations (e.g., the client is limited to certain parts of the functionality, in a certain payment plan) if the API is not regulated by an API Gateway (e.g., market.mashape.com). We also go through non-functional aspects, such as existence of payment plans and call limit rates, but intend to provide a wider focus, by analyzing a broader set of technical features, including REST principles compliance and best practices. Regarding the works that provide information on quality attributes of web services, including open issues and challenging aspects.

It is relevant to mention the proposal of an approach for benchmarking the quality of web services considering geo-mobility of the service end-user presented in [12]. High-level architectural and engineering options (e.g., client-side caching, the use of a backup API that could leverage machine-readable API descriptions) are discussed as means to handle variable Quality of Service (QoS), observed during long-term benchmarking experiments. The authors do not present detailed insights on the features of the services under benchmark.

Software engineering research opportunities for the consumption of web APIs are discussed in [13]. The challenges discussed are: i) service consumers have no control over the web service (the provider may change the API or the service); ii) clients may not be sure of the validity of calls to the API at run-time; iii) SDKs out of synchronization with the actual service; iv) and QoS issues. The authors identify a few research lines (e.g., static analysis for checking requests, documenting API signatures, coding practices and patterns for dealing with varying QoS, impact of web API usage on non-functional aspects). Our work can help complementing this kind of study, by providing further detailed data for research to take place.

A language for modeling REST client-server conversations is presented in [14]. Authors refer that most APIs are simply exposing low-level HTTP details without hypermedia controls (which creates obstacles for conversation), which are two features that we analyze in this work for an up-to-date view.

In [15], the authors analyze the evolution of the interface of client-server technologies (including web APIs). The authors overview technologies used for web services and point out reasons for the misuse of REST principles. The authors emphasize that some designs and mechanisms (e.g., REST design, OAuth) are much more successful at web API reuse and discuss the usefulness of an interface description for REST web APIs.

The Sustainability and Innovation Global Executive Study conducted a survey that included 4000 practitioners. This survey specifies that 48% of the respondents agreed that the concept of sustainability compelled them to adapt easily available web application security models whether it secures the application or not [28].

Coral Calero and Mario Piattini [29] presented an article that focused on security sustainability of web applications with three core areas being: human, economic and environmental sustainable-security. According to the authors' research, perdurability as security attribute and cohesion and coupling of design properties are important attributes that highly affect sustainability.

Oyedeji et al. [30] proposed a web application sustainability design catalogue which aimed to help developers in producing a sustainable web application according to the users' needs. This catalogue is made by using the reviews of current and past research on sustainable web application. A framework has also been proposed in this work which includes a set of sustainability goals with respect to security and quality perspective. According to the study, availability and perdurability of sustainable-security attributes and encapsulation, inheritance and abstraction of design properties are important attributes that highly affect sustainability.

Dawood et al. [31] in 2018 presented an article on sustainable design of web applications. The authors provided the principles and viewpoints on sustainability with respect to web application security for providing a situation and methodology on web application design and sustainability. The study also presented the recent research trends in perspective of sustainability in the circumstances of design for software applications. According to the study, design-size, polymorphism inheritance and abstraction are important attributes of design that highly affect sustainability.

Li et al. [32] in 2017 proposed a fuzzy theory based services of security for sustainable mobile-edge computing. Authors of this paper presented a security proxy to support compatibility to outdated security functions. Also, to catch the best direction of the mandatory functions of security, authors established a Fuzzy Inference System (FIS) based tool to reach optimum goals. The results for the proposed model describes that FIS achieved good performance. According to the study, availability, confidentiality and perdurability of security attributes are important and highly affect sustainability.

Robillard et al. [33] in 2016 familiarized the concept of integrating design attributes into the sustainability

ISSN NO: 2249-3034

of software. This was a vision paper which proposed that the concept of sustainable software design should be integrated with sustainable software development. The paper also discussed research challenges like divergent opinions and longevity of software design. According to that study, availability and perdurability of security attributes and cohesion, coupling and abstraction of design properties are important that affect sustainability, highly.

3. SUSTAINABLE SECURITY OF WEB APPLICATION

Security of the users' data is at risk and securing the web applications in all areas including institutional activities is the foremost priority of the security practitioners [34]. However, maintaining security is a difficult task because practitioners are facing various issues including sustainability. The ISO and IEEE series of software engineering standards deliver the supervision of sustainability in web application development perspective [35][43]. The Microsoft describes sustainability as an amount of how stable a design is to secure a web application to do suggested responsibilities [30]. Furthermore, the impact of sustainable web application on economy, society, human beings, and environment is very high [33]. Sustainability of a web application can be defined as the capacity of developing a software product in a sustainable manner [36]. Further, sustainability shows an essential character for high security design of web applications. Design properties of web applications also play a significant role during its development. Hence, achieving the relationship between security, sustainability and design is a very crucial task but very important for effective yet sustainable web application [34]. Sustainable-security may be cleared if the design assures the stable-security structure against security threats during the use of web application. In the wake of rising costs, future calculated uncertainty, and resource constraint, it is essential that the web applications provide high security and that too at decreasing cost. Sustainable security will support in developing web application that will be capable of protecting itself from attacks apart from being dependent upon web application security for its safety in case of threats. Practitioners are trying their best to enlist higher sustainable security of web application design. However, sustainable security is still not at its efficacious best. Furthermore, educational institutions are demanding stable maintenance of security during the use of web applications and attributes play a crucial role in sustainable-security design of web applications. Web application security is the awareness applied to defend web application against malicious attack and other hacker risks so that the website stays to function correctly under such prospective risks. Also, security is essential to provide integrity, authentication and availability. Web application security protects it from attacks such SQL injection, cross scripting and other vulnerabilities [37]. Sustainable software is software which is easy to evolve and maintain, fulfills its intent over time and survives uncertainty [36].

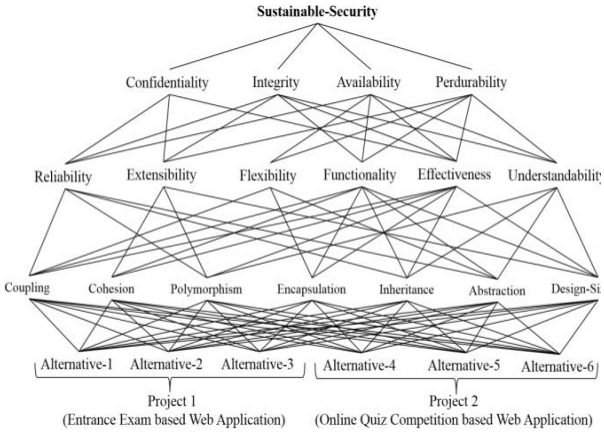

Figure 1: Pertinent sustainable-security characteristics

Relationship between the attributes has been presented in figure 1. Figure 1 shows the order of sustainable-security which is further classified in three levels of sustainable-security affecting attributes. For example, confidentiality affects reliability, extensibility and effectiveness; integrity affects reliability, extensibility, functionality, effectiveness and understandability, etc. Figure 1 also shows that a factor at one level affects one or additional attribute of the higher level but its impact is not the same on them. It may differ. For example, reliability has an influence on confidentiality, integrity and availability as well, but its influence values are not same [35]. The relationship of attributes helps to distinguish among the influences of the same factors with the others factor at a higher level. For the purpose of assessment, there are four factors at level 1 that are defined as follows:

- Confidentiality: Confidentiality in terms of sustainable security refers to protecting information from being accessed by unauthorized parties while ensuring the maintenance of sustainability for human. In other words, only the people who are authorized to do so can gain access to sensitive data [38].
- Integrity: Integrity in terms of security refers to ensuring the authenticity of information with respect to sustainability—that information is not altered, and that the source of the information is genuine [38].
- Availability: Availability in terms of sustainable security means that data is manageable by authorized practitioners in a sustainable environment. If an attacker is not able to compromise the confidentiality and integrity, then the attacker may try to execute attacks like denial of service that would bring

ISSN NO: 2249-3034

down the server, making the website unavailable to legitimate and sustainable users due to lack of availability [25].

- Perdurability: Perdurability may be defined as an extent to which a software may be modified and reused in order to perform stated functions under stated conditions for a specified period. It affects the extensibility, effectiveness, functionality, understandability and flexibility for sustainable security [39].

Factors of sustainable security at level 1 are represented as F1, F2, F3 and F4. There are six factors at level 2 that are shown as follows:

- Reliability: Reliability refers to the dynamic performance of a software as well as sustainable security. Reliability is the extent to which a work product operates without failure under given conditions during a given time period. Reliability means to prevent sustainable web application security from failures and to make strategies to maintain its trustworthiness [40].

- Extensibility: Extensibility is defined as the simplicity with which sustainable security can be improved by making changes in security requirements and goals. Sustainability depends on extensibility for improving the environmental sustainability of web application so that the cost and time incurred in web application security can be lessened [41].

- Flexibility: In the field of sustainable secure design, flexibility refers to the sustainable designs that can adapt external changes when it occurs. It affects design attributes of coupling, encapsulation and abstraction [41].

- Functionality: The quality or state of being functional for a sustainable design that is admired both for its beauty and for its functionality is known as the attribute of functionality of sustainability [41].

- Effectiveness: Effectiveness is a degree to which something is effective in making a preferred outcome; success. In sustainability terms effectiveness is maintaining the degree of effectiveness of sustainable security [41].

- Understandability: This term in sustainability is defined as ability to being understood in sustainable conditions. The information that should be able to comprehend itself in a sustainable environment [41].

The descriptions of seven design attributes with their influence on sustainable security attributes are as follows:

- Coupling: Coupling is the degree of interdependence between software modules. For a good design of software, low coupling is required. Data coupling is considered as the best type of coupling. Coupling of sustainable software impacts reliability, flexibility, functionality and effectiveness [42].

- Cohesion: Cohesion is the measure of the degree to which the elements of the module functionally relate to each other. For a good design high cohesion is required. Cohesion is well affected by sustainable security factors, i.e., extensibility, effectiveness and functionality as extensibility helps in gaining functionality, hence, the support process of cohesion [42].

- Polymorphism: It is a design idea that refers to the capability of a function to take on multiple forms. It is best for code reuse and hence increases the sustainability of developed software. As per the concept, polymorphism does affect the reliability, extensibility, effectiveness and understandability [42].

- Encapsulation: Encapsulation is the enclosure within an object of all the resources essential for the object to function or method of the data. Encapsulation provides the code security, flexibility and maintainability which is further stimulated into sustainable security. Encapsulation directly helps in maintaining sustainable security and hence it increases efficiency, functionality and flexibility of web application [42].

- Inheritance: The capability of a class to inherit properties and characteristics from another class is referred to as the inheritance property of design. This provides the capability of code reusability in design which further supports the concept of sustainability and security. Inheritance works on the concept of code reusability, hence it has an impact on reliability, effectiveness and functionality. It also improves the understandability of code using inherited classes and functions [41].

- Abstraction: Abstraction is the process of signifying vital features and hiding unused data from the user. This reduces complexity of code and improves the overall security of software design. Abstraction works on the concept of security and, hence, improves reliability and functionality of web application. Using abstract methods and classes hides the not useful data from the designer, thus improving the understandability and flexibility of code [42].

ISSN NO: 2249-3034

- Design-Size: Design size is the feature of design which is used to estimate the size of software design. A constrain design size reduces complexity, sustains the design for longer period. Improved design size improves extensibility and effectiveness of design. Estimation of design size does help in improving functionality and understandability of code and thus affects the sustainable security of web application [43]. Design plays a very influential role in software development in sustainable environment, still it is left for end consideration. While developing sustainable software, design should be considered as a team player with other attributes of sustainable security. The loss of design knowledge is a well-known problem that has received numerous attention of academicians in this field. As design and sustainability both are the important attributes for quality web application, hence the integration of both will empower the overall quality of sustainable and secure software.

4. CONCLUSION

In this paper, we went through the REST services, with the REST principlesand adherence to REST service development best practices. We also went through various research which is in the field of REST Web services API. The knowledge brought in this paper, namely sustainable security in web application also playa important role in the field of REST Web services API. Understanding the state of the practice, can provide useful research directions for the academy (e.g., understanding how REST development tools affect compliance with principles or best practices). As future work, we intend to focus on the security aspects in the field of Google App Development with REST API services.

REFERENCES

[1] M. Maleshkova, C. Pedrinaci, and J. Domingue, “Investigating Web APIs on the World Wide Web,” in 2010 Eighth IEEE European Conference on Web Services, 2010, pp. 107–114.

[2] D. Renzel, P. Schlebusch, and R. Klamma, “Today’s Top ‘RESTful’ Services and Why They Are Not RESTful,” in Web Information Systems Engineering - WISE 2012, 2012, pp. 354–367.

[3] F. Bülthoff and M. Maleshkova, “RESTful or RESTless – Current State of Today’s Top Web APIs,” in The Semantic Web: ESWC 2014 Satellite Events, 2014, pp. 64–74.

[4] G. Schermann, J. Cito, and P. Leitner, “All the Services Large and Micro: Revisiting Industrial Practice in Services Computing,” in Service-Oriented Computing – ICSOC 2015 Workshops, 2015, pp. 36–47.

[5] Rodríguez et al., “REST APIs: A Large-Scale Analysis of Compliance with Principles and Best Practices,” in Web Engineering, vol. 9671, A. Bozzon, P. Cudre-Maroux, and C. Pautasso, Eds. Cham: Springer International Publishing, 2016, pp. 21–39.

[6] M. Fowler, “Richardson Maturity Model,” martinfowler.com. [Online]. Available: https://martinfowler.com/articles/richardsonMaturityModel.html. [Accessed: 27-Jun-2017.

[7] F. Petrillo, P. Merle, N. Moha, and Y.-G. Guéhéneuc, “Are REST APIs for Cloud Computing Well-Designed? An Exploratory Study,” in Service-Oriented Computing, 2016, pp. 157–170.

[8] F. Haupt, F. Leymann, A. Scherer, and K. Vukojevic-Haupt, “A Framework for the Structural Analysis of REST APIs,” in Software Architecture (ICSA), 2017 IEEE International Conference on, 2017, pp. 55–58.

[9] M. A. Oumaziz et al., “Empirical Study on REST APIs Usage in Android Mobile Applications,” in Service-Oriented Computing, 2017, pp. 614–622.

[10] F. Palma, J. Gonzalez-Huerta, M. Founi, N. Moha, G. Tremblay, and Y.-G. Guéhéneuc, “Semantic Analysis of RESTful APIs for the Detection of Linguistic Patterns and Antipatterns,” Int. J. Coop. Inf. Syst., vol. 26, no. 02, p. 1742001, May 2017.

[11] A. Gamez-Diaz, P. Fernandez, and A. Ruiz-Cortes, “An Analysis of RESTful APIs Offerings in the Industry,” in Service-Oriented Computing, 2017, pp. 589–604.

[12] D. Bermbach and E. Wittern, “Benchmarking Web API Quality,” in Web Engineering, 2016, pp. 188–206.

[13] E. Wittern et al., “Opportunities in software engineering research for web API consumption,” in Proceedings of the 1st International Workshop on API Usage and Evolution, 2017, pp. 7–10.

[14] A. Ivanchikj, C. Pautasso, and S. Schreier, “Visual modeling of RESTful conversations with RESTalk,” Softw. Syst. Model., pp. 1–21, May 2016.

[15] J. Kopecký, P. Fremantle, and R. Boakes, “A history and future of Web APIs,” It - Inf. Technol., vol. 56, 2014

[16] R. T. Fielding, “Architectural Styles and the Design of Networkbased Software Architectures,”

ISSN NO: 2249-3034

Univ. of California, Irvine, 200.

[17] E. Lebanidze. Securing Enterprise Web Applications at the Source: An Application Security Perspective. Accessed: Sep. 5, 2019. [Online]. Available: https://www.owasp.org/images/8/83/Securing_Enterprise_Web_ Applications_at_the_Source.pdf.

[18] S. Ardi. (2008). A model and Implementation of a Security Plug-in for the Software Life Cycle. [Online]. Available: https://epdf.pub/a-model-andimplementation-of-a-security-plug-in-for-the-software-life-cycle.html.

[19] Y.-W. Huang, S.-K. Huang, T.-P. Lin, and C.-H. Tsai, ''Web application security assessment by fault injection and behavior monitoring,'' in Proc. 12th Int. Conf. World Wide Web, May 2013, pp. 148–159.

[20] N. Cai, J. Wang, and X. Yu, ''SCADA system security: Complexity, history and new developments,'' in Proc. 6th IEEE Int. Conf. Ind. Inform., Jul. 2008, pp. 569–574.

[21] C. Haipeng, ''A preliminary study on the sustainability of Android malware detection,'' Jul. 2018. arXiv:1807.08221. [Online]. Available: https://arxiv.org/abs/1807.08221.

[22] (2016). Microsoft 2016 Corporate Social Responsibility. [Online]. Available: https://www.microsoft.com/en-us/CMSFiles/Microsoft_2016_ Corporate_Social_Responsibility.pdf?version=889768cf-2300-6a48- 33e1-5fca73a1836e&CollectionId=df8dab12-dbf6-441f-a2db5996225f2c6a.

[23] M. Stifel. (2018). Securing the Modern Economy: Transforming Cyber Security Through Sustainability. [Online]. Available: https://www.publicknowledge.org/assets/uploads/documents/Securing_the_Modern_ Economy–Transforming_Cybersecurity_Through_Sustainability_FINAL_4.18.18_PK.pdf.

[24] M. Hackett, B. Johnson, and H. Q. Nguyen, Testing applications on the web: Test planning for internet-based systems. Hoboken, NJ, USA: Wiley, 2011. [Online]. Available: https://www.oreilly.com/library/ view/testingapplications-on/9780471201007/

[25] A. Kumar and R. C. Bansal, ''A review of multi criteria decision making (MCDM) towards sustainable renewable energy development,'' Renew. Sustain. Energy Rev., vol. 69, pp. 596–609, Mar. 2017.

[26] A. Mardani, A. Jusoh, K. Nor, Z. Khalifah, N. Zakwan, and A. Valipour, ''Multiple criteria decision-making techniques and their applications–a review of the literature from 2000 to 2014,'' Econ. Res.-EkonomskaIstraživanja, vol. 28, no. 1, pp. 516–571, Dec. 2015.

[27] D. Garg, S. Luthra, and A. Haleem, ''Ranking of performance measures of GSCM towards sustainability: Using analytic hierarchy process,'' Int. J. SocManag. Econ. Bus Eng., vol. 8, no. 3, pp. 764–770, Feb. 2014.

[28] R. Paradis and B. Tran. (2016). Balancing Security/Safety and Sustainability Objectives. [Online]. Available: https://www.wbdg.org/resources/ balancing-securitysafety-and-sustainability-objectives.

[29] C. Calero and M. Piattini, Green in Software Engineering. Springer, 2015. Accessed: Sep. 13, 2019. [Online]. Available: https://www.springer.com/ gp/book/9783319085807.

[30] S. Oyedeji, A. Seffah, and B. Penzenstadler, ''A catalogue supporting software sustainability design,'' Sustainability, vol. 10, no. 7, p. 2296, Jul. 2018.

[31] K. A. Dawood, K. Y. Sharif, A. A. Zaidan, A. A. Ghani, H. B. Zulzalil, and B. B. Zaidan, ''Mapping and analysis of open source software (OSS) usability for sustainable OSS product,'' IEEE Access, vol. 7, pp. 65913–65933, 2019.

[32] G. Li, H. Zhou, B. Feng, G. Li, T. Li, Q. Xu, and W. Quan, ''Fuzzy theory based security service chaining for sustainable mobile-edge computing,'' Mobile Inf. Syst., vol. 2017, 2017, Art. no. 8098394.

[33] M. P. Robillard, ''Sustainable software design,'' in Proc. 24th ACM SIGSOFT Int. Symp. Found. Softw. Eng., Nov. 2016, pp. 920–923.

[34] N. Selwyn, ''Web 2.0 applications as alternative environments for informal learning—A critical review,'' in Proc. CERI-KERIS Int. Expert Meeting ICT Educ. Perform., vol. 16, Oct. 2017, pp. 1–17.

[35] C. N. Murphy and J. Yates, The International Organization for Standardization (ISO): Global Governance Through Voluntary Consensus. London, U.K.: Routledge, 2009. [Online]. Available: https://www.routledge.com/The-International-Organization-for-Standardization-ISO-GlobalGovernance /Murphy-Yates/p/book/9780415774284.

[36] T. Butler, ''Compliance with institutional imperatives on environmental sustainability: Building theory on the role of Green IS,'' J. Strategic Inf. Syst., vol. 20, no. 1, pp. 6–26, Mar. 2011.

[37] C. Calero, I. G.-R. De Guzmán, M. A. Moraga, and F. García, ''Is software sustainability considered

ISSN NO: 2249-3034

in the CSR of software industry?'' Int. J. Sustain. Develop. World Ecol., vol. 26, no. 5, pp. 439–459, 2019. doi:10.1080/13504509.2019.1590746.

[38] Confidentiality, Integrity, Availability: The three components of the CIA Triad. [Online]. Available: https://security.blogoverflow.com/2012/08/confidentiality-integrity-availability-the-three-componentsof-the-cia-tria (2012).

[39] C. Calero, M. Moraga, and M. F. Bertoa, ''Towards a software product sustainability model,'' Sep. 2013, arXiv:1309.1640. [Online]. Available: https://arxiv.org/abs/1309.1640.

[40] D. Jodarski. Success and Sustainability—How to Ensure Lasting Results With Security Process Improvement. Accessed: Sep. 10, 2019. [Online]. Available: https://www.bigskyassociates.com/blog/successand-sustainability-how-to-ensure-lasting-results-with-Multidisciplinary Open Access Journal, Volume 7,2019.DOI 10.1109/ACCESS.2019.2946776.

security-processimprovement.

[41] R. Caralli, J. F. Stevens, C. F. Wallen, D. W. White, W. R. Wilson, and L. R. Young. (2007). Introducing the CERT Resiliency Engineering Framework: Improving the Security and Sustainability Processes. [Online]. Available: https://resources.sei.cmu.edu/library/asset-view.cfm?assetid= 8389.

[42] J. Carter. (2012). Coupling and Cohesion: A View of Software Design from the Inside Out. [Online]. Available: https://www.ehrscience.com/2012/11/12/coupling-and-cohesion-a-view-of-software-designfrom-the-inside-out-2/

[43] Alka Agrawal, Mamdouh Alenezi, Rajeev Kumar, Raees Ahmad Khan, "Measuring the Sustainable-Security of Web Applications Through a Fuzzy-Based Integrated Approach of AHP and TOPSIS",