Web Applications Security: More Collaboration

Majed Abdullah Albarrk[1], Mohamad Shady Alrahhal[2]

[1,2]King Abdul-Aziz University, Faculty of Computing and Information Technology, Department of Computer Science, Jeddah, Saudi Arabia

---***---

Abstract - *Web usage and technologies are increasing rapidly, but unfortunately, Web application vulnerabilities are increasing as well. Lack of collaboration among IT professionals is considered the main reason applications have a high number of vulnerabilities. However, there are some solutions mentioned in this paper that can help increase the participation among the different IT areas. The solution depends on agent software technology and uses moving vehicles as a case study.*

Key Words: agent, security requirements, moving objects. Web applications, hops, collaboration.

1. INTRODUCTION

Web applications is a client-server software reached by a Web browser (Nations, 2014). Web application vulnerabilities and flaws are the highest among other vulnerabilities. Therefore, the application vulnerabilities are among the top concerns in major countries, and securing applications and services that connect to the Internet or even an intranet is essential (Ranke & Carleton, 2015). Moreover, modern Web applications are getting more complex. Many of these applications consume or provide Web services or APIs. A simple example is when a retail Web site consumes or uses an external credit card processing service using a Web services technology such as RESTful. However, these features in Web applications increase the security potential risks. According to the "(ISC)^2 Global Information Security Workforce Study" (2015), "application vulnerabilities are ranked the number 1 threat to cybersecurity professionals," which encourages organizations to reprioritize their risk management plan tasks (2).

1.1 Reasons for Choosing Web Application Security

There are a few reasons for choosing Web applications security. One reason is that I found the idea of SANS Survey, builders and defenders of Web applications interesting because it talks about the reality of the challenges of securing Web applications (Ranke & Carleton, 2015). Furthermore, I have always wanted to learn more about how to add a security component to the application development life cycle. For example, I have worked as a programmer for seven years, and when I was assigned to tasks in development projects, there was a time limit that made me rush. As a result, the applications had a bunch of flaws. Therefore, I wrote this paper focusing on securing Web applications by increasing the cooperation between developers, builders, and IT security professionals and by applying some solutions that can close this gap, such as using the DevOps approach, supporting security in Software Development Life Cycle (SDLC), using some OWASP tools and projects, and applying OWASP standards.

1.2 Different Perspectives

The focus in this paper is to concentrate on two points of view regarding how to make Web applications more secure: builders and defenders. Builders include developers, development organizations, software department, and all roles involved in building and delivering the Web application. Defenders include security and operations teams who are responsible for running and securing Web applications or securing the entire firm or organization. Each group has their own perspective on Web application security, and unfortunately this difference in perspective affects Web application security and creates a gap between the two. For example, when the SANS Survey asked for who tests the security in the applications, only 21.6% of the answers chose developers, although development teams should get more involved in security testing and merge it with the software life cycle (Ranke & Carleton, 2015). Fortunately, the gap between them starts to close gradually, according to them. However, more effort must be made to reduce the high risk of compromising Web applications by increasing the collaboration between these groups.

1.3 Obstacles

Firms face some challenges when they want builders and defenders to work together. One of the challenges is the lack of shared knowledge. Some information security engineers may not be familiar with some Web application flaws and security issues, while some software developers may lack security knowledge. The other challenge is their priorities. Security can be the highest priority for defenders but not for builders. Developers may have other concerns such as time and functionality (Ranke & Carleton, 2015).

2. RELATED WORK

There are some recommendations to close the gap among IT professionals and reduce the risk of the Web application vulnerabilities. These solutions can be divided into three categories: models and approaches can be applied, environment cultures can be enhanced, and education and knowledge can be increased.

2.1 Models and Approaches

DevOps. The DevOps approach is an operational model designed to resolve the gap between development, quality assurance, and operations. The term "DevOps" obviously comes from combining development and operations. This model aims to decrease the required time and efforts for the change management and increase the awareness among employees who work in these three areas. For example, some systems can be down because of the incompatibility with an upgrade done by operations. This issue is common because of the lack of coordination among IT professionals. Therefore, this model can offer a strategy that helps resolve these kind of issues. But what could Web application security do with the DevOps model? The answer is this model is essential in resolving the security bugs and flaws in Web applications faster, since time matters in public-faced applications to avoid zero-day attacks. As shown in Figure 1, Ullrich and Johnson (2016) show the length of time it takes to patch a vulnerability in respondents' firms in their survey.

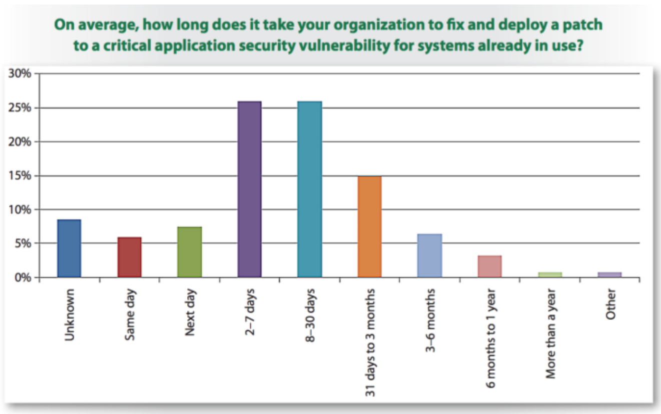

Fig. 1. Source: Ullrich and Johnson, 2016.

DevSecOPs. Brown (2015) describes DevSecOPs as the same as DevOps model but adding a security part to its structure and integrating some security monitoring tools to the model. For example, the need for cloud infrastructure for many firms requires taking secure architecture and design into consideration. One of the ways that can be used in this context is "infrastructure as code," which is a method that makes developers build the infrastructure of an environment automatically by a secure and pretested code to avoid any potential bugs or errors (Brown, 2015).

2.2 Security Development Life Cycle

There are several SDLC models such as Waterfall, spiral, Agile, SCMM, IDEAL, SCRUM, etc. figure 2 shows the general steps used in any SDLC model in terms of agaility.

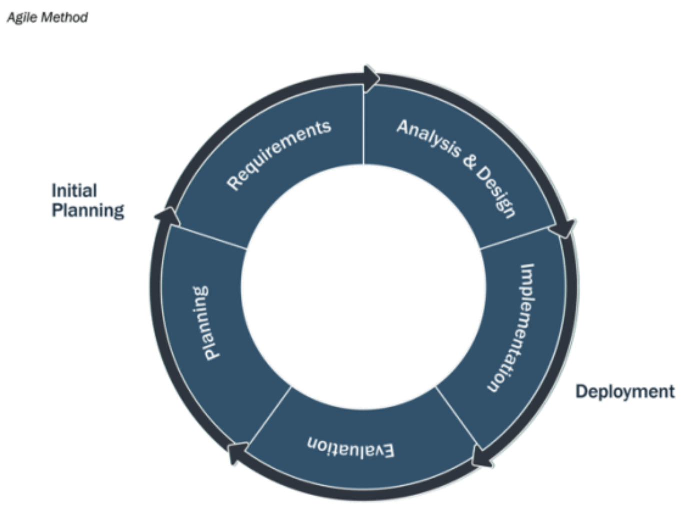

Fig. 2. General steps in SDLC models

After projects mangers choose the right SDLC model for a project, the project should support security in all SDLC model phases. For example, the security specialists should get involved early in the SDLC, such as in the initiation and analysis phases. Involving them early has two benefits. First, they will understand everything about the project from the beginning. Second, they can prevent some unsafe plans and wrong requirements. For example, processing financial transactions that do not fit the policy of the firm would be an unsafe plan, and requiring more information from users than the application needs is a wrong requirement. Moreover, SDLC project managers should review the design phase output before the approval of the design phase. This is essential because some obvious security flaws can be detected in the design phase, such as storing sensitive information like credit card information in a database with no extra security controls. Moreover, information analysts should design security features and controls in the Web application to apply and assure the security in that project. Furthermore, programmers should be aware of how to follow secure coding standards implementing some security organizations recommendations and standards such as OWASP standards. Finally, the main objective of the testing phase is to ensure the quality of the product, which will not happen without testers having security knowledge, who usually reveal many bugs and security flaws. One of the testing approaches is static testing, which analyzes the source code before publishing the Web application in a test server. It uses some predefined logical code flaws and errors and compares them with the code. On the other hand, dynamic testing evaluates the Web application in a runtime environment whether in a test server or production server such as some Web application scanning tools. This testing is very useful for organizations that do not have the source code of the Web applications.

3. Multi Hop based Protocol (MHP)

This section is structured, so that the problem is defined with the objectives. A Threat Model (TM) is defined second.

Then, the agent based system architecture is introduced, in details, according to the predefined threat model. Finally, the architecture details represented by a sequence diagram.

3.1 Problem and Objective

In VCSs, the users (or vehicles) exchange some information about the traffic roads and climate case. Based on the exchanged information, a decision is made to keep continue driving on the current road or moving towards another one. The classical scenario of VCS is illustrated in Figure 3.

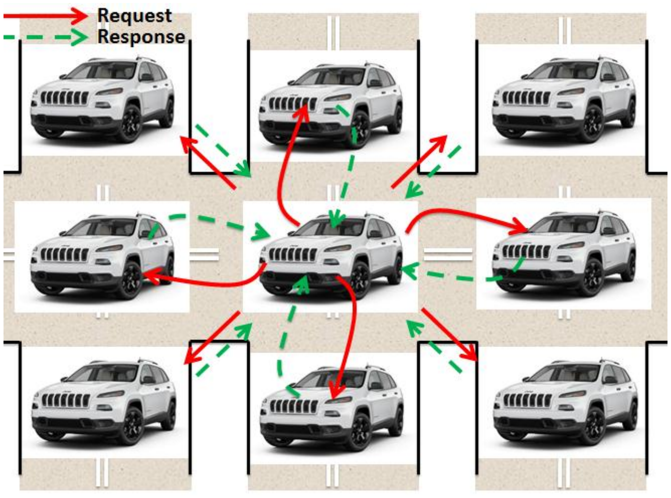

Fig. 3. The classical scenario of VCS.

In Figure 3, a sender (user or vehicle) who wants to explore a road, sends multiple requests to the vicinity. The receivers (i.e., other users or vehicles) manipulate the request, and then feedback the sender with the corresponding response. However, such classical scenario has many security gaps and no security requirements are provided.

To strengthen the previous scenario, the security requirements (mentioned above in introduction section) must be satisfied. The security requirements are confidentiality, integrity, authorization, mutual authentication, accountability, and non-repudiation. Consequently, the first objective is guaranteeing the previous security requirements. On other hand, the classical scenario is vulnerable to attacks, such as unauthorized access, replay, modification, repudiation, eavesdropping, and masquerade attacks. So, the second objective is to ensure robustness against the previous attacks.

3.2 Threat Model (TM)

Figure 4 illustrates the TM.

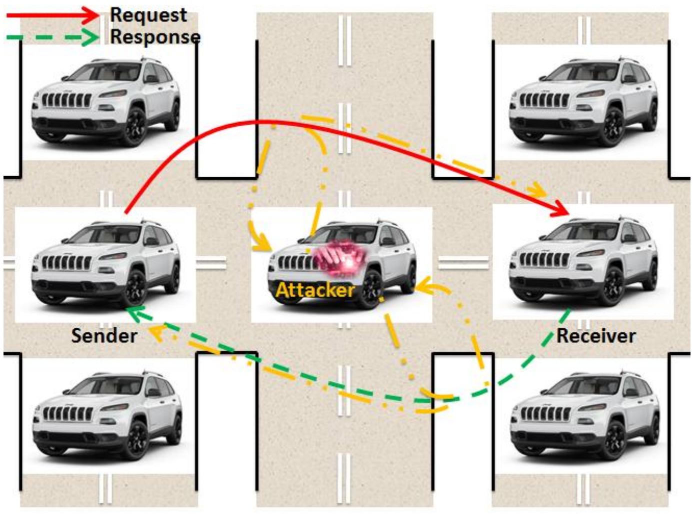

Fig. 4. Threat model.

As shown in Figure above, the attacker can be any network user (vehicle driver) that is located as a man in the middle. The objective of the attacker is to obtain the sent request, and then modify it. In addition, the attacker can obtain and modify the retrieved response. Consequently, the mode of the attack is active. In regarding to the attacks themselves, the attacker has the following capabilities summarized in Table 1.

Table 1. Capabilities of attacker.

Capability NO	Attack name
1	Can eavesdrop the communication channel.
2	Unauthorized access attack.
3	Replay attack.
4	Modification attack.
5	Repudiation attack.
6	Masquerade attack.

3.3 Proposed Agent Based System Architecture

We use agents based software technology to build the system (alluhaybi 2029-2020). Privacy is considered out of scope, where location based services can be deal with in future (Alrahhal 2018-2019). The framework of the proposed system consists of a number of vehicles (N_{ve}), number of RSUs (N_{rsu}), and a Traffic Management Center (TMC) of a smart city, all connected via a network, as shown in Figure 5.

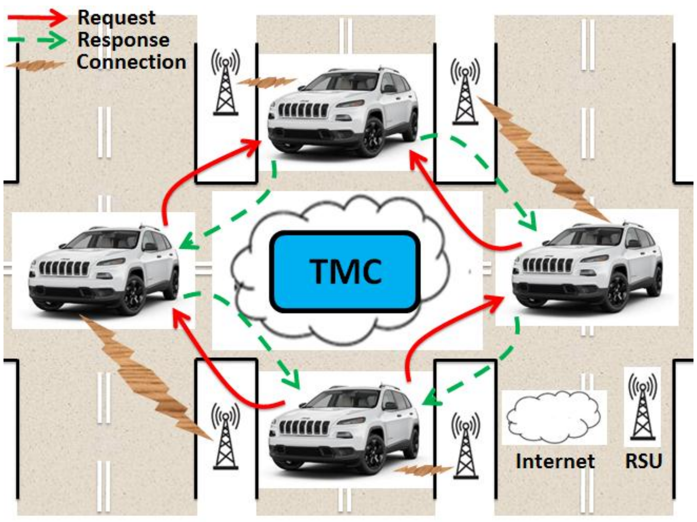

Fig. 5. Framework of the proposed system.

The system is managed by three agents, which are (Custodian_M), (Cloner_{MA}), and (Redirector_{MA}), as shown in Figure 6.

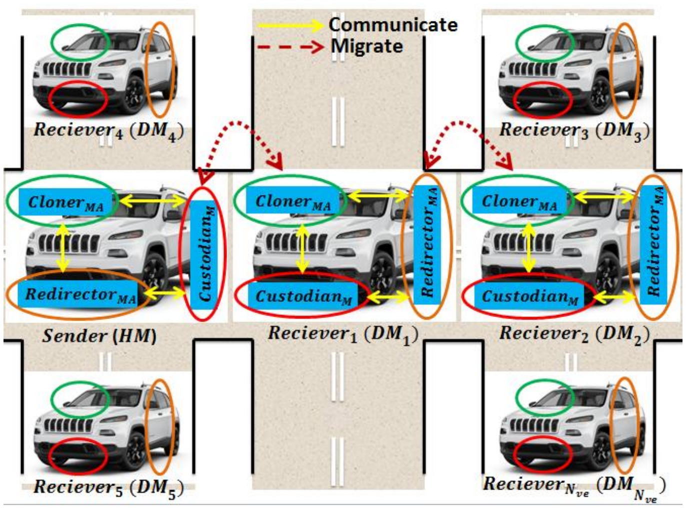

Fig. 6. The proposed agent based architecture.

Table 2 summarizes the used agents, defining the type of the agent, where they are installed, and the major missions assigned to each one.

Table 2:. Agents.

Agent name	Type	Mission	Location
Custodian_M	Mobile	Carries the mission created by the user/driver and migrates to different DMs to execute it there.	Each vehicle
Cloner_{MA}	Stationary	Creates four exact copies of the mobile agent for forwarding.	Each vehicle
Redirector_{MA}	Stationary	Forwards the four copies of the mobile agent to the four directions.	Each vehicle

3.4 Roles of Agents

In this sub section, we provide the roles of each used agent in details, where a novel agent based protocol is proposed. This protocol relies on multi-hopes. Here, hope refers to migration of mobile agent between each pairs of vehicles involved in the system. A three parties contribute to create a single hope, which are the source (i.e., HM), the destination (i.e., DM), and the itinerary details of the mobile agent (Custodian_M) between the HM and the DM. The general scenario of our proposed protocol is illustrated in Figure 7.

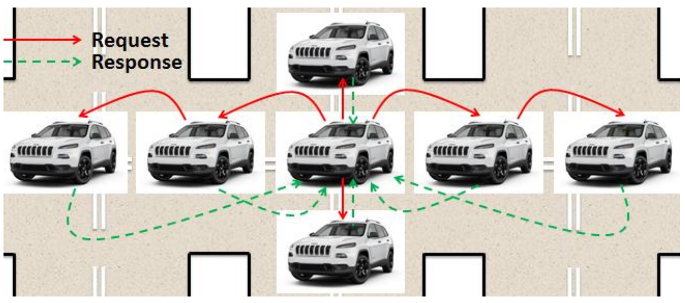

Fig. 7. Hops in the proposed systems.

In Figure 7, the communication starts when the user/driver/sender in the middle vehicle sends a request asking for the traffic road case (for example). This request, then, is received by the nearest vehicles directly located around the sender, manipulated, and the responses are sent back to the sender. In addition, each vehicle (receiver), in turn, forwards the request to its nearest vehicles to be manipulated, and the response generated there are sent back again to the sender. In this environmental context of the VCS, we employ the Public Key Infrastructure (PKI) to satisfy the security requirements and to make defenses against the attacks mentioned in the threat model above. Therefore, we consider the TMC as the Certificate Authority (CA). Then, all the vehicles involved in the system are verified by the TMC, and granted a certificates, which include public keys. The public keys, then, are distributed to other vehicles. The private keys are generated at each verified vehicle (based on the public keys) and are kept secret. Figure 8 summarizes the process of configuration by the PKI.

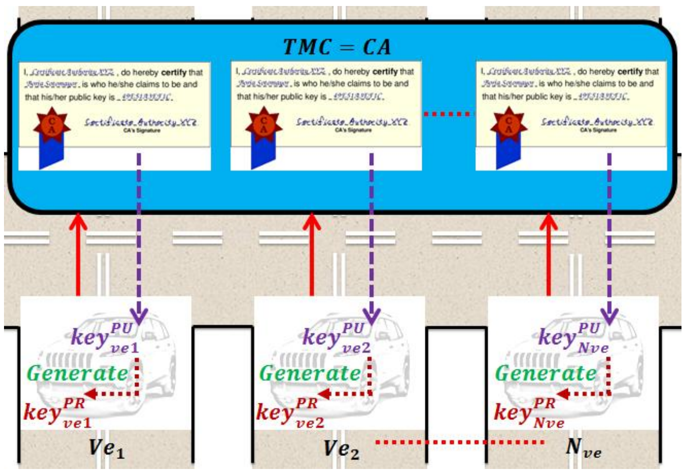

Fig. 8. Granting certificates and generating private keys

Table 3 defines the symbols used in the proposed agent based protocol.

Table3. symbols and definitions.

Symbol	Definition
ve^{i}_{Pla}	Vehicle $platform_i$ (HM or DM)
$MobA_i$	Mobile $Agent_i$
key^{PU}_{i}	Public key of vehicle $platform_i$
key^{PR}_{i}	Private key of vehicle $platform_i$
req	Request
res_i	Result produced by vehicle $platform_i$ (DM_i)
$hash()$	Hash value
ads_i	Address of vehicle $platform_i$
TS_i	Time stamp produced by vehicle $platform_i$
$\rightarrow$	Migrate

The communication starts when the user/driver of the first vehicle (ve^{1}_{Pla} = HM) asks for information from nearby vehicles, as shown in Figure 9.

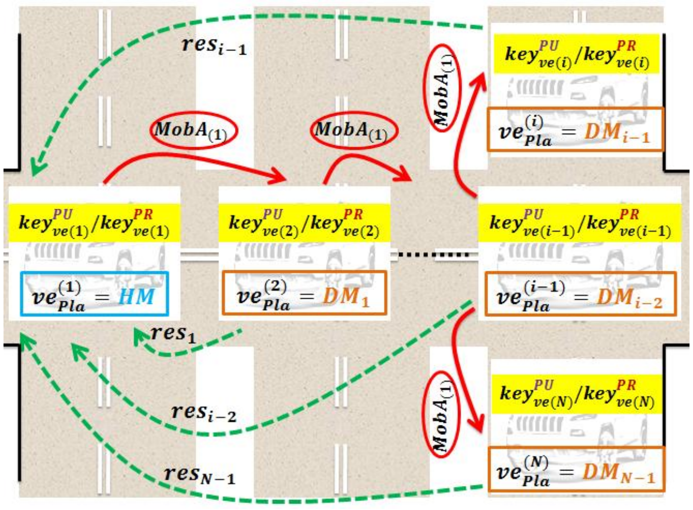
res_{i-1}
$MobA_{(1)}$
$MobA_{(1)}$
$MobA_{(1)}$
$key^{PU}_{ve(i)}/key^{PR}_{ve(i)}$
$ve^{(i)}_{Pla} = DM_{i-1}$
$key^{PU}_{ve(1)}/key^{PR}_{ve(1)}$
$ve^{(1)}_{Pla} = HM$
$key^{PU}_{ve(2)}/key^{PR}_{ve(2)}$
$ve^{(2)}_{Pla} = DM_1$
$key^{PU}_{ve(i-1)}/key^{PR}_{ve(i-1)}$
$ve^{(i-1)}_{Pla} = DM_{i-2}$
res_1
res_{i-2}
$MobA_{(1)}$
$key^{PU}_{ve(N)}/key^{PR}_{ve(N)}$
$ve^{(N)}_{Pla} = DM_{N-1}$
res_{N-1}

Fig. 9. Exchange a request and results in MHP.

4. EVALUATION

we measure the quality of the MHP protocol by verification, validation, and testing stages. This can be done by measuring the Software Quality Factors (SQFs). The SQFs are as follows:

1. Correctness: it determines how well a system fulfills the customer's overall objectives.
2. Reliability: it determines the likelihood with which a system can be expected to perform its intended function.
3. Efficiency: it determines the level of computing resources (including time) required by a system to perform its function.
4. Integrity: it determines how well the data is secured.
5. Usability: it determines how easy the system is to use.
6. Maintainability: it determines how easily bugs can be found and fixed.
7. Flexibility: it determines the effort required to modify an operating system.
8. Testability: it determines how easily the system can be tested to show that the customer's requirements have been met.
9. Portability: it determines how easily the system can be used on another machine should the customer change their platform or should other customers require it.
10. Reusability: it determines how easy it is to reuse some of the software to make future developments more cost-effective.
11. Interoperability: it determines the effort required to couple one system to another.

To infer the values of the SQFs listed above, the following software metrics are defined as a measurable quantity:

1. Accuracy: it refers the precision with which computation and control is carried out.
2. Auditability: it refers the ease with which conformance to standards can be checked.

3. Communication commonality: it refers the degree to which standard interface protocols and bandwidth have been used.
4. Completeness: it refers the degree to which full implementation of the required functionality has been achieved.
5. Complexity: it refers the degree to which a program is difficult to understand.
6. Conciseness: it refers the compactness of the program in terms of line codes.
7. Consistency: it refers the use of uniform design throughout the software development process.
8. Execution efficiency: it refers the run-time performance of the system.
9. Expandability: it refers the degree to which architectural design can be expanded.
10. Hardware independence: it refers the degree to which the software is decoupled from hardware on which it operates.
11. Error tolerance: it refers the damage that occurs when a program encounters an error.
12. Tractability: it refers the ability to trace a design representation or actual program component back to the system requirements.

Table 4 shows the results.

Table 4:The relationship between software metrics and SQFs.

SQF / Metric	Correctness	Reliability	Efficiency	Integrity	Usability	Maintainability	Flexibility	Testability	Portability	Reusability	Interoperability
Accuracy		√									
Auditability				√				√			
Communication commonality											√
Completeness	√										
Complexity		√				√	√	√			
Conciseness			√			√	√				
Consistency	√	√				√	√				
Execution efficiency			√								
Expandability							√				
Hardware independence									√	√	
Error tolerance		√									
Tractability		√									

5. CONCLUSION

In order to reduce the Web applications' potential risks, different IT teams with different perspectives about security should collaborate. The goals of increasing collaboration and understanding each other's language can be reached by applying some sufficient models and approaches, enhancing the work environment, and increasing security training.

REFERENCES

1. Bird, J., Johnson, E., & Kim, F. (2015) 2015 state of application security: Closing the gap: A SANS survey. Retrieved from: https://www.sans.org/readingroom/whitepapers/analyst/2015-state-application-security-closing-gap-35942
2. Brown, J. (2015). DevSecOps: Taking a DevOps approach to security. Database and Network Journal, 45(2), 17.
3. Cois, C.A., Yankel, J., & Connell, A. (2014). Modern DevOps: Optimizing software development through effective system interactions. *IEEE*. Retrieved from: http://ieeexplore.ieee.org.libezproxy2.syr.edu/xpls/icp.jsp?arnumber=7020388
4. Nations, D. (2014). Web applications: What is a Web application? About.com. Retrieved from: http://Webtrends.about.com/od/Webapplications/a/Web_application.htm.
5. Ranke, M., & Carleton, J. (2015). The professionals' perspective: Cyber security in the DACH region. A Frost & Sullivan white paper. Retrieved from https://www.isc2cares.org/uploadedFiles/wwwisc2caresorg/Content/Professional%20Prespective%20DACH%20Region-English.pdf:

6. Swartout, P. (2012). Continuous delivery and DevOps : A quickstart suide. Birmingham: Packt Publishing.
7. Ullrich, J., & Johnson, E. (2016). 2016 State of application security: Skills, configurations and components. A SANS survey. Retrieved from: https://www.sans.org/reading-room/whitepapers/application/2016-state-application-security-skills-configurations-components-36917
8. Alrahhal, Mohamad Shady, et al. "AES-route server model for location based services in road networks." International Journal Of Advanced Computer Science And Applications 8.8 (2017): 361-368.
9. Alrahhal, Mohamad Shady, Maher Khemakhem, and Kamal Jambi. "A SURVEY ON PRIVACY OF LOCATION-BASED SERVICES: CLASSIFICATION, INFERENCE ATTACKS, AND CHALLENGES." Journal of Theoretical & Applied Information Technology 95.24 (2017).
10. Alrahhal, Mohamad Shady, Maher Khemekhem, and Kamal Jambi. "Achieving load balancing between privacy protection level and power consumption in location based services." (2018).
11. Alrahhal, H.; Alrahhal, M.S.; Jamous, R.; Jambi, K. A Symbiotic Relationship Based Leader Approach for Privacy Protection in Location Based Services. ISPRS Int. J. Geo-Inf. 2020, 9, 408.
12. Alrahhal, Mohamad Shady, Maher Khemakhem, and Kamal Jambi. "Agent-Based System for Efficient kNN Query Processing with Comprehensive Privacy Protection." International Journal Of Advanced Computer Science And ApplicationS 9.1 (2018): 52-66.
13. Al-Rahal, M. Shady, Adnan Abi Sen, and Abdullah Ahmad Basuhil. "High level security based steganoraphy in image and audio files." Journal of theoretical and applied information technology 87.1 (2016): 29.
14. Alluhaybi, Bandar, et al. "A Survey: Agent-based Software Technology Under the Eyes of Cyber Security, Security Controls, Attacks and Challenges." International Journal of Advanced Computer Science and Applications (IJACSA) 10.8 (2019).
15. Fouz, Fadi, et al. "Optimizing Communication And Cooling Costs In Hpc Data Center." Journal of Theoretical and Applied Information Technology 85.2 (2016): 112.
16. Alluhaybi, Bandar, et al. "Dummy-Based Approach for Protecting Mobile Agents Against Malicious Destination Machines." IEEE Access 8 (2020): 129320-129337.

AUTHORS

Mohamad Shady Alrahhal: received PHD from king abdulaziz university, KSA, department of computer science, college of computing and information technology (2018). Received master degree from Damascus University, Syria, (2013). Received a degree of computer engineering from Albath University, Homs, Syria, (2011). Currently, he interests in agent based software technology, artificial intelligent, cyber security, imageand video processing, and risk management. Practically, he was the director of the IT department of IoT company, and risk management manager

Majed Abdullah Albarrak: Received a master of Science degree from Information School, Syracuse university, USA, in information management (2016), and A computer science BS degree from Imam University, Riyadh, Saudi Arabia (2006). 14 years of experience in risk management and many areas of data science including data mining, text mining, data warehouse. He has been worked as data management section for 3 years.