ISSN 2347 - 3983
Volume 8. No. 9, September 2020
International Journal of Emerging Trends in Engineering Research
Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter40892020.pdf
https://doi.org/10.30534/ijeter/2020/40892020

Performance Analysis of Automation Monitoring System shifting from DevOps to DevSecOps

Dr.K.V.D.Kiran[1], MR.P.J.R.Shalem Raju[2]
[1]PROFESSOR, DEPARTMENT OF CSE, KONERU LAKSHMAIAH EDUCATION FOUNDATION, VADDESWARAM, AP, INDIA,kiran_cse@kluniversity.in
[2]ASSOCIATE PROFESSOR, DEPARTMENT OF CSE, SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN, BHIMAVARAM,AP,INDIA, jesuratnashalemraju@svecw.edu.in

ABSTRACT

The purpose of this paper is to conduct a study on integrating all necessary security dimensions, which in the DevOps era can easily be left out of the picture, as a sticky part into both application and infrastructure development in Information Sharing Companies. Like any other start-up company, risk panorama towards Information sharing is evolving with each new vulnerabilities find out day by day and absence of protection expertise amongst each. Developer and Operations groups which can be liable for the short tempo of tendencies with inside the company's products. This paper evolves via way of means of introducing right metrics for having a Reliable Monitoring systemin every degree of development, via way of means of recording each earlier than and after states and additionally proposed a method for development with inside the enterprise. To investigate and to reviewed through the four principles of DevSecOps: ethnicity, Automation, dimension and division .As a consequence, it was found that the available research focuses heavily on securing the technologies frequently used in DevSecOps and found the prime challenges to be securing the consumption pipeline, opposite security with fast deliveries.

Key words : DevOps, DevSecOps, secure software engineering, principles.

1. INTRODUCTION

In the rapid pace of software development and deployment in the DevOps era, security is often left out of the picture because typically threatening like people who are slower than process, information division companies weren't an exception. To support an agile process with a security author plan to integrate all the necessary security tools, measures and policies in information exchange companies. This gap between security and business development should not become a bottleneck and slow down the entire cycle. The integration of security with development and operation is called DevSecOps. Adapting DevSecOps is a must for all small organizations if they want to keep their business on the market. Due to the business model of information division companies, moving from DevOps to DevSecOps can provide excellent service to a broad spectrum of companies and their customers. Thanks to the Information Division API, almost all types of operators can connect with retailers globally and sell their data to end customers. The lack of security experts, security measures and security knowledge in the information exchange companies, had exposed many security flaws to the company and had the potential to put the entire business at risk; Most of these security flaws will be addressed in the future course of this study.The paper consists of 4 sections , section 1 deal with problem statement and section 2 deals with methodology of the paper and section 3 contains the concepts of moving from DevOps to DevSecOps and the last section deals with discussion of these automation tools.

1.1 Problem

In this article, a systematic review of the literature was carried out, with the aspire of kind the current state of security research in the circumstance of the DevOps expansion method[1]. The investigate answered two research questions to move to DevSecOps:

RQ1: What safekeeping actions are allied with DevSecOps in the literature?

Some of the main security issues have been identified in information exchange companies consisting of application security and platform security. Also, each of these two categories was separated into some subcategories to better address them. Here is a list of security flaws that have been detected in application development and deployment.

• There was a lack of automated security controls in the codes written by developers who do not have sufficient knowledge in the area of cybersecurity[2].
• Expose all security credentials in plain text in all application jobs
• Lack of a minimum privilege policy for user access.
• Applications used vulnerable libraries, implementing code in unauthorized Docker images.
• Lack of adequate security controls against the code and its respective third-party libraries

RQ2: How are EADD (Ethnicity, Automation, Dimension, and Division) principles reflected in DevSecOps study?
As the EADD principles have beforehand used in studious works, I choose them as a yardstick in my investigate to assess the primary studies' understanding of DevOps as an secretarial happening that changes ethnicity, facilitates division and expects automation and dimension with security[3].

2. METHODOLOGY

As this study is based on improving security in an IT startup and all the findings have been applied to this case study, we believe action research is an appropriate way to conduct this study. Each improvement step in this study has been carried out through the following three steps: [3]

1. Discover those areas that security failed to achieve by establishing a reliable monitoring and alert system.

2. Take measurable actions to mitigate or eliminate the negative impact of the security problems detected in both the applications and the platform.

3. Evaluate the result of each improvement action by comparing the states before and after each phase.

Having lots of sensitive data, services, and applications hosted on any public cloud, it makes sense for the author to break all the corresponding assets into smaller pieces to get a better understanding of the elements that your security should improve. Information division assets have been divided into three different but related components, such as monitoring and alerts, application security and platform security, including application security and platform security, has been in the process of improvement from May 2018 until now is given as a summary of proof of framework in Figure1[4 6].

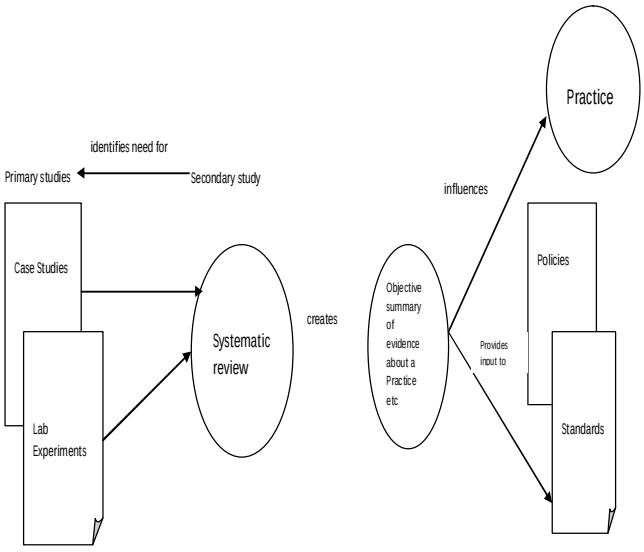

Figure1: Objective summary of evidence about a Practice

2.1. Monitoring and alert

Establishing a reliable monitoring solution to extract different types of metrics from all assets in use, is not only the main cornerstone of this study for having a reliable validation method to record different states of each phase, but it also plays a role. crucial in providing clear transparency in the functionality of all applied security measures that the author plans to apply in the information exchange assets. In this phase, we will install a metric scraper on each asset depending on the type of functional and security metrics we are looking for for those specific assets. Running an accurate alert system and rules based on the metrics that the monitoring system collects from different assets is also another important thing that we must use to establish a successful DevSecOps in information exchange companies.[7 ,8, 9]

2.2 Application protection

As for the security of the application, it is necessary that some security measures intervene in the storage, writing and deployment of codes to have an application that does not put the security of the organization at risk. The author had taken the following list as a step-by-step plan to introduce better security in applications and code that developers write and are implemented using production deployment tools.

a) Hosting codes written in a secure and reliable source code manager, such as GitHub and Git, which can control the version of the codes that are sent to the corresponding repository and allows to quickly roll back to the previous version of the codes in case of code not wanted to be pushed to the repository.

b) Run the static code analyzer against all newly written code and third party libraries they use; These scans should also run automatically on each application deployment. As soon as a vulnerability is detected during application deployment, the deployment work should end immediately and not go to production. The result of this type of failed implementation should announce both the developer and the product manager, immediate feedback.

c) Repetitive feedback to the respective developers and the product manager about the failed implementation can be a great source of truth to understand why an application implementation has failed. Loosely integrating with Jenkins in the job deployment process can meet this goal. Having detailed information about the result of running the static code analyzer with the new application code on each new deployment will increase the visibility in the background of the deployment pipelines to a decent level.

d) No need to put database credentials and other confidential assets in plain text within application deployment jobs or import them into applications

3. MOVING FROM DEVOPS TO DEVSECOPS BASED ON PROTECTION CONTEXT

In the DevSecOps approach, security is everyone's obligation, and when it comes to software security, those obligations fall specifically on DevOps and on the shoulders of the developer. Due to the loss of security specialists in information exchange companies, it is no longer feasible to teach current (and future) builders quickly enough, however, it can be an excellent and especially applicable method of offering them security rules already established to help them. with stable code writing. Also, having some security scanner equipment across all ranges of the Software Program Improvement Life Cycle (SDLC) and offering them with brief comments can offer a great price to increase code security within the employer. In this explore, the essential issues are the research questions. The stakeholders I want to promote from this study are qualified in the protected software engineering and explore community, who may perhaps promote from the investigate results. The role of methodical fictitious analysis as a research method is in fact twofold: [10]

1) it provides new insights by analyzing previous research and

2) it detects and presents research gaps and thus identifies the needs for new primary studies.

3.1 Integrate security as builder code

One of the most crucial steps in integrating security into the continuous integration / continuous delivery (CI / CD) process is to combine it as early as possible within the life cycle. There are some crucial blessings in this, like growing a short comment loop between inventing a security flaw in the developer's code and returning it to the responsible developer for correction. fix something within the code that they just coded instead of solving a security problem in old code that they developed more than a month ago. The integration of security assessments prior to the upgrade of the software suite in manufacturing can store a number of charges that commercial companies have to pay while detecting a security flaw within the manufacturing software. This idea was established through a review conducted through NIST, which is much cheaper and more powerful as we change security at the early level of the enhancement pipeline. As demonstrated in Desktop 1, the value of mitigating security issues increases as we allow issues to flow into the cessation phase of the software program improvement existence cycle (SDLC). [11] [4] The following diagram is taken from the review "Economic Impacts of Inadequate Infrastructure for Software Testing", which was carried out through the means of the National Institute of Standards and Technology in 2010.

By understanding the reasons mentioned above today, in the information exchange companies, the builders knew how to observe the subsequent security rules after they started coding. But before reviewing them, it's really worth saying that the writer has moved all the code already written in the form of an internal provisioning manipulation control repository to GitHub. GitHubvia's form of default media plays a normal security experiment with its rich security database of Common Vulnerabilities and Exposures (CVE), towards all code repositories. Here are the security measures that builders must take into account while writing code at information exchange companies. [5] [2]

· All code repositories are private.
· All incoming gadgets must have integrity test.
· All communications with the code repository are encrypted.
· None of the constructors is using the privilege account to develop.
· Lock sensitive logs going to the challenge repository through git secrets and techniques.
· All grab branches are protected against deletion and direct commits.
· Repositories connected to the dependency vulnerability tester.
· All commitments signed through the means of builders and this mechanism is periodically affirmed.
· Repositories have a proper .gitignore report to stop stat leaking.

3.2 Autonomous security in CI / CD pipeline

However, through a means of security integration while simultaneously encrypting the capture of vulnerabilities both within the customer's device and within the supply tampering control (SCM); however, some security vulnerabilities are constantly escaping these stages and into the deployment segment. For this matter, it seems to have an automatic vulnerability security test in the deployment pipeline with common comments to make us ensure the security of the code before deploying it in manufacturing. A static assessment security vulnerability scanner for Ruby on Rails programs, in Jenkins used as a continuous integration appliance. vulnerability assessments in CI / CD pipeline after upgrade and DevOps groups acted on them.[12]

3.3 Developing a way of life of visibility

Contrary to what top builders think, the upgrade group's obligation no longer ends after their codes call for the upgrade within the manufacturing environment. By editing with DevSecOps, builders remain accountable for capability in their code within the manufacturing environment. No one, like the builders who wrote the code, knows how their code can be compromised, so your collaboration in tracking code behavior in manufacturing is lavishly priced for increased software security. Along with the DevOps group, the improvement group also has to often examine the behavior of your code and assess what kinds of requests are reaching your software endpoints and how programs respond to future requests. In the exchange of information, metrics of the walking programs

3.4 Secure deployment jobs

In addition to software program enhancement, there are some security-related aspects of software program implementations, including the loss of a centralized vault to store secrets and techniques rather than placing them in an undeniable textual content layout. in implementation work. The writer, with the help of the improvement group, implemented the Hashi Corp vault as a centralized secret control device for all programs and erased all Jenkins' work on secrets and plain text techniques. When introducing the vault to the integration appliance, Jenkins reads the required environment variables from the vault server and injects them into the software during deployment[15][16]; Jenkins has the least privilege access, the simplest read, to the vault server.

Table 1: Represents this improvement in protecting the credentials

Before Using Vault	After using Vault as the secret manager
# Application (system) export RAILS_THREADS=7 export RAILS_EN=”generation” export RACK_EN=dbus_rack export GESEVER_USER=g export GEMSERVER_PWD= cat >$WORKSPACE/config/database.yml<<EOL # Managed via Jenkins ProductionDB: adapter: mysql encoding: utf8 host: port: 3305 database: username: password: pool: 14 EOL	# Application (system) export RAILS_THREADS=7 export RAILS_EV=”generation” export RACK_EV=$RAILS_EV export GESERVER_USER=${GEMSERVER_UNAME} export GEMSERVER_PWD=${ GEMSERVER_PWD} cat >$WORKSPACE/config/database.yml<<EOL # Managed via Jenkins ProductionDB: adapter: mysql encoding: utf8 host: ${MYSQL_CORE_DATABASE} port: ${MYSQL_PORT} database: ${MYSQL_CORE_DATABASE} username: ${MYSQL_CORE_UNAME} password: ${MYSQL_CORE_PWD} pool: 14 EOL

Table 1 represents this improvement in protecting the credentials. Before Using Vault After using Vault as the secret manger Table 1: Comparison between the use of variables in deployment jobs before and after utilizing HashiCorp vault for managing secrets. Since this security vulnerability mitigation directly related to reducing the attack surface on deployment pipeline.[14]

3.5 Detect existing security flaws

Containers, like any other process, can face different types of threats and challenges. In information exchange companies, these security risks against the container and its images can potentially happen in the following seven different ways. - Breaking of the container; This can happen due to a lack of proper isolation in namespaces, which can potentially lead to being vulnerable to cross-container attacks. By having this type of vulnerability in our container application, an attacker could exploit your access to other containers; For example, if one of those containers that hosts the operator application is compromised by an attacker, an attacker could take advantage of your access to those containers that host financial applications. [2. 3]

3.6 Run container as non-root users;

Almost all application containers were running with root user privileges and without any restriction on allowed capabilities, which means that the misbehaving application or a potential hacker has root access to the host machine it is running on the container. - Excessive use of resources; This can cause a DOS auto attack from inside the container. This can happen for different reasons such as bad code or third party libraries leaking memory, CPU or even storage problems. This potential problem has been solved by assigning a predefined quantity that each container can use; This has been coded separately for each application within their Docker deployment files. - Manipulation of container images; The unreasonable access privilege to the container's image repository, the registry servers, will increase the risk of an insider hacker introducing malicious code within the container's images, which are a source of truth for all applications running inside. of the containers. This security issue has been resolved by reducing developer access to the image repository and only the integration tool, Jenkins, and the DevOps team have the correct access to communicate with the registry servers.

3.7 Make sure the basics of host and network security are in place

In the articles reviewed, [1], the expansion surroundings is secluded against insider threats by various security practices (eg access control, hardening, logging). In [2], the center was also on the deployment pipeline and recognize all of its entrusted apparatus and performing security activities to make certain that the preferred level of security is accomplished. In [3] runtime security was achieved by monitoring security metrics, identifying vulnerabilities, and using adaptive self-defense mechanisms to protect the system (eg automatic patches and firewall actions).

3.8 Testing and verification

To verify the outcome of the previous mitigation steps that had been applied to the information division application containers and the docker engine, the author uses an official docker security verification tool, called Docker Bench for Security. Docker Bench for Security is a tool to check the most common best practices about deploying Docker containers in production. This tool performs a security analysis through an automated Jenkins job that has been created by the author. Table 2 represents the current and past security status of docker daemon running, docker images, and running application containers in information exchange companies [7]

3.9 Use automated tools with tailored rules

The EADD activity Use automated tools with tailored rules got four mentions ([3], [9], [10], [17]). Using automated tools with tailored rules, means customizing static analysis in a way makes it more efficient. One of the goals for doing so is reducing the number of false positives. This is an important matter, as according to [10], precisely the idea of security slowing development down (e.g., through a large number of false positives) reduces developers' enthusiasm towards security initiatives. the authors discussed the importance of using the right analysis methods for the used development technologies in order to gain a trustworthy outcome.

4. SECURITY PRACTICES/ACTIVITIES

One of my research questions dealt with how the four principles of DevOps, ethnicity, automation, dimension and division (EADD) are represented in the reviewed works. I wanted to know first of all, if the DevOps principles were reflected in the articles and second of all, whether the researchers saw these principles as something worth mentioning in relation to security. The further I got in my research, the more I realized that the way the researchers acknowledged these principles or not seemed to reflect how they understood DevOps. As said before, some see DevOps as a heavily automated process, while others as a management style encouraging autonomy. These divergent views come to light through analysis of the interpretation of the DevOps principles.

4.1 Discussion on Research Questions

RQ1: Security actions that are connected with DevOps To respond the investigate query of security actions that are linked with DevOps in the research literature, the chosen revisions were cautiously interpret and examined using content analysis. References to security activities in the appraisal works were noticeable downhill in the data removal formulation.

"Investing resources in fixing some vulnerabilities during the growth period can be useless, either because a new vulnerability can appear at runtime in an unexpected use case, or because we were unable to properly rank the vulnerabilities and fix the wrong one." When considering moving security to the left or right, it is still wise to recall the wise words of the creators of BSIMM (McGraw et al., 2019): "DevOps is a cultural change that cannot be addressed with any additional procedural changes. "So, in addition to security measures, there is a need to take a closer look at ethnicity and three other DevOps principles.

RQ2:
EADD principles reflected in DevOps research. To answer the last research question, articles were read and content analysis was performed to analyze references to any of the four DevOps principles, ethnicity, automation, dimension, and separation, that were written into the data extraction spreadsheet. All four DevOps principles are mentioned in the literature reviewed. However, the large number of references to these principles seem to indicate that not all researchers understand DevOps in the same way. This finding is consistent with a previous systematic literature review by Jabbari et al. (2016), who found that different authors mean different things by DevOps. Security is an important component of DevOps, as even the name of a development method indicates the

integration of development with operations. Once again, the findings suggest that the roles in the organization need to be clearly defined: who does what and how information is exchanged between these stakeholders. One article [9] emphasizes the need for a common goal, and I think that after analyzing the study, this becomes obvious. Area of security check is done for the attributes of host configuration ,container run time before and after using of DevSecOps which helps in analyzing and moving to automation of the process by having CI/CD followed with dockerization and swarm configuration is shown in Table2.[18]

If an organization wants to create an ethnos in which tasks are shared and people work together, these people need to know what values they are striving for. If the importance of security is not established, implementing security measures easily becomes a tug-of-war between the security team and developers and operations. If developers are not communicated about the importance of security, developers will feel that security actions are unnecessary and will only slow them down.[19] If, on the other hand, the importance of security is established, Sectors, Developers and Operators can work together to achieve a balance that is consistent with the values of the organization.

Table2: The following table represents area of security check for automation process

Area of security check	**Before**	**After**
Host configuration		
There is a disconnect divider for containers	NO	YES
review is configured for the Docker daemon	NO	YES
Auditing is configured for Docker files and directories	NO	YES
Docker daemon configuration		
There is a limit between containers on the default bridge	NO	YES
Insecure registries are not used	YES	YES
User namespace support	NO	YES
Centralized and remote logging is configured	NO	YES
Containers are restricted from acquiring new privileges	NO	YES
Container Images and Build file		
Containers are not running as root	NO	YES
All containers use trusted base images	YES	YES
Unnecessary packages are not installed inside the container	YES	YES
Container runtime		
AppArmor profile is enabled on running containers	NO	NO
SELinux security options are set	NO	YES
The host's network namespace is not shared among containers	NO	YES
None of the containers running with network mode 'host'.	NO	NO
Memory usage for the container is limited.	NO	YES
The CPU usage for the container is limited.	NO	NO
Container's root file system is mounted as read-only.	NO	NO
Incoming container traffic is bound to a specific host interface.	NO	NO
Ensure 'on-failure' container restart policy is set to '5'.	NO	NO
Ensure PIDs C-Group limit is used.	NO	YES
The container is restricted from acquiring additional privileges.	NO	YES
Container health is checked at runtime.	NO	YES
Ensure the Docker socket is not mounted inside any containers.	NO	YES
Docker Swarm configuration		
Data exchanged between containers are encrypted on different nodes on the overlay network.	NO	YES
Swarm manager is run in auto-lock mode.	NO	YES

5. CONCLUSIONS

The literature review has done some research on just one specific aspect of this transition, including container security, infrastructure hardening, and deployments security. However, the full transition from DevOps to DevSecOps in a real company with many subcategories has not been made. Regarding conducting this study in a wide range of areas that need to be addressed in order to improve their security, this study was conducted in three separate phases: establishing a robust monitoring and alerting system, improving security in both development cycles and software deployments, and also security. infrastructure, excluding human intervention in it and applying a deep level of isolation at different levels of its network.

This article conducted a systematic literature review to understand the current state of security research in the context of the DevOps development method. The study answered the research questions for the transition to DevSecOps:

•RQ1: Which security activities are associated with DevSecOps in the literature?

•RQ2: How are the EADD (ethnicity, automation, dimension and sharing) principles reflected in DevSecOps research?

This helps us research on how DevSecOps is adopted in specific areas of IT, although it has been concluded that it is

an artifact that always needs to be adapted to its environment. This research aimed to answer to the main research questions "How to adopt DevSecOps principles, practices and tools?"by conducting a design science research in Identity and Access Management.T he main research question was broken down into smaller subquestions that helped to construct and evaluate the design artifacts for DevSecOps adoption. The design science research was conducted using a case study approach, wherein the artifacts were observed and analyzed in the problem domain. First, theDevSecOps analysis was made in the case company's IT unit to gain deeper understanding on what are the challenges of DevSecOps adoption at an organizational level.

However, the current level of automation on integrating security into all mission-critical processes will make security as an inevitable part for all business developments in the future, but still, there is plenty of room for improvement.

REFERENCES

1.Artac, M., Borovssak, T., Di Nitto, E., Guerriero, M., & Tamburri, D. A. 2017. **DevOps: introducing infrastructure-as-code**. In 2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C).497–498. IEEE.

2.Tony Hsiang-Chih Hsu, "**Security Monitoring in Hands-On Security in DevOps: Ensure continuous security, deployment, and delivery with DevSecOps** Kindle Edition", 2018

3. Tony Hsiang-Chih Hsu, "Security Monitoring in Hands-On Security in DevOps: Ensure continuous security, deployment, and delivery with DevSecOps Kindle Edition", 2018.

4.Dr.K.V.D.Kiran"**An Automated Evaluation of Live Forensic Approach**", International Journal of Engineering and Advanced Technology (IJEAT),Volume-8, Issue-3S, February 2019,ISSN: 2249 – 8958

5.Dr.K.V.D.Kiran,Mr.P.V.VaraPrasad,"**Improved handoff mechanism for infiltrating user equipments in composite networks**", International Journal of Electrical and Computer Engineering (IJECE),Vol. 10, No. 3, June 2020, pp. 2600~2606,ISSN: 2088-8708.

6.Dr.K.V.D.Kiran,**Securing web application by using qualitative research methods for detection of vulnerabilities in any application of DevSecOps** ,International Journal of Emerging Trends in Engineering Research , volume 8,number 3,pages 878-884.

7. Srinivasu N , **On the viability of adaptive paris metro pricing in agent based model for federated clouds** , International Journal of Recent Technology and Engineering (2018)

8.Kiran P. Gaikwad, C. M. Sheela Rani, S. B. Mahajan, P. Sanjeevikumar, "**Dimensionality Reduction of facial features to recognize emotion state"**, Lecture Notes in Electrical Engineering, (LNEE)-Advances in Systems, Control and Automation", Vol.442, pp-719-725, Dec 2017. https://link.springer.com/chapter/10.1007/978-981-10-4762-6_69.

9. Jyoti B. Kulkarni, Dr. Manna Sheela Rani Chetty, **"Depth Map Generation from Stereoscopic Images Using Stereo Matching on GPGPU"**, Journal of Advanced Research in Dynamical and Control Systems, Volume 9, ISSN 1943-023X, Special Issue – 02 / 2017.

10. Nuvvula, Adiraju, Mubin, Shahana Bano, S Valisetty, "**Environmental smart agriculture monitoring system using internet of things**", International Journal of Pure and Applied Mathematics, Volume 115, Issue 6 Special Issue, 2017.

11. T.Sajana, M.R.Narasingarao, " **An Ensemble Framework for classification of Malaria Disease**" , ARPN Journal of Engineering and Applied Sciences", Vol:13,No:9, Pg.No:3299-3307, May, 2018(Scopus)

12.Sajana T, Narasingarao M R, " **Classification of Imbalanced Malaria Disease using Naïve Bayesian Algorithm**", International Journal of Engineering and Technology(UAE), Vol:7, Pg.No:786-790, 2018.

13. Metamaterial inspired quad band circularly polarized antenna for WLAN/ISM/Bluetooth/WiMAX and satellite communication applications, Rao, MV; Madhav, BTP; Anilkumar, T; Nadh, BP, AEU-INTERNATIONAL JOURNAL OF ELECTRONICS AND COMMUNICATIONS,2018.

14. Paule Christina, "**Securing DevOps: detection of vulnerabilities in CD pipelines**", 2018; Available: https://elib.unistuttgart.de/bitstream/11682/10046/1/2018-04-17-masterthesis-final-christinapaule.pdf

15. **Strip secure payment,** Published in stripe.com documentations; Available: https://stripe.com/docs/payments/3d-secure

16. Michael Falk, Oscar Henriksson, "**Static Vulnerability Analysis of Docker Image**s", 2017; available: http://www.divaportal.org/smash/get/diva2:1118087/FULLTEXT02.pdf

17. Stan Wisseman, "**Third-party libraries are one of the most insecure parts of an application**", 2016; Available: https://techbeacon.com/security/third-partylibraries-are-one-most-insecure-parts-application

18. Siva Kumar, Fazal Noorbasha "**Low Power Carry Look-AheadAdder using Transmission Gate Multiplexer**,", ijeter, Volume 8, No. 1 January2020,

19. , M Trinath Basu, JKR Sastry,"**Strengthening Authentication within OpenStack Cloud Computing System through Federation with ADDS System**",, International Journal of Emerging Trends in Engineering Research, Volume 8, No. 1 January 2020.