JCSE International Journal of Computer Sciences and Engineering Open Access

Research Paper Vol.-7, Issue-12, Dec 2019 E-ISSN: 2347-2693

Secure Software Architecture and Design: Security Evaluation for Hybrid Approach

Jameel Ahmad Qurashi[1*], Harvir Singh[2], Vijay Nunia[3]

[1]Department of Computer and System Sciences, Jaipur National University, Jaipur, India
[2]Department of Engineering (All Wings), Jaipur National University, Jaipur, India
[3]Department of Computer and System Sciences, Jaipur National University, Jaipur, India

[*]*Corresponding Author: jameelqureshi41@gmail.com, Tel.: +91-8491848803*

DOI: https://doi.org/10.26438/ijcse/v7i12.16 | Available online at: www.ijcseonline.org

Accepted: 10/Dec/2019, Published: 31/Dec/2019

Abstract— Software furnishes administrations that may accompany a few vulnerabilities or risks. Attackers perform activities that break the security of framework through dangers and cause disappointment. To dodge security helplessness, there are numerous security- explicit ideas that ought to be resolved as prerequisites amid software improvement life cycle so as to convey solid and secure software. This paper first, studies various existing procedures, systems required for creating secure software dependent on the related distributed works. It begins by displaying the most important Secure Software Development Lifecycle, a correlation within the primary security highlights for each procedure is proposed. The consequences of the examination will give the software engineer with a rule which will help in choosing the best secured process. Second, the paper lists a lot of the most broadly utilized determination dialects with the points of interest and impediments for each**.**

Keywords— Software, Security, Security evaluation, vulnerabilities, secure architecture etc**.**

I. INTRODUCTION

Secure software is where an unapproved individual can't get to it, change it, or assault it. The level of such security is estimated by the current number of security vulnerabilities. The software without any vulnerability is highly secured software, whereas the software with no less than one helplessness is insecure software [1]. So as to incorporate security in the software designing, the security viewpoints ought to be incorporated from the earliest starting point of the Software Advancement Life Cycle. The secure software designing is the way towards planning, building, and testing software with the goal that it winds up secure, this incorporates Secure Software Advancement Life Cycle (SSDLC) forms and secure software improvement (SSD) techniques. A SSDLC procedure considers security parts of the software amid the advancement life cycle by utilizing SSD strategies. SSD strategies incorporate, among others, security particular dialects, security necessities building procedures, and software security affirmation techniques [2]. The significance of including security in the software creation, originates from the staggering expense of evacuating framework mistakes which may cause security helplessness subsequent to creating software. This paper studies a lot of secure software improvement life cycle forms, recognizes the attributes for everyone, and presents a correlation between them dependent on the security exercises that are incorporated into the advancement lifecycle. This will be valuable for the software designers by helping them in picking the right procedure.

II. RELATED WORK

Herold [11] expounded on the contrasts between security and protection from the business perspective. As per Herold, protection is a set of enactments that decide the direct proportionality to possess the data and allow getting, refreshing, distributing, and passing on such data. Security is a set of strategic procedures, strategies, and methods used to make data mixed up, out of reach, unaltered, and denied for unapproved parties through guaranteeing classification and wellbeing. Herold expressed that security experts are separated into two gatherings in the manner in which they comprehend the connection between the two terms, however neither is valid. On one hand, there is a gathering trust that security and protection are the equivalents. Then again, others trust that the two terms totally restrict each other. As per Herold, numerous associations have two separate divisions: one for protection thinking about lawful issues, and another for security thinking about specialized issues. Subsequently, the two offices don't impart and each works independently. Messenger asserted that such hierarchical structure isn't right; rather, the two offices should cooperate and facilitate with each other. The security division's staff must be associated with protection issues. Similarly, the protection office's faculty must be associated with security issues. As the connection among protection and security is solidified, security is actualized to guarantee protection.

Caloyannides [12] composed another article about the truth of the connection between protection and security, and whether security compels protection or not. The writer began his article by asserting that internet observing can't anticipate fear mongering in light of the fact that it can't screen abroad arrangement and coordination to carry out a wrongdoing. Along these lines, it is inadequate. In addition, individuals who work at delicate positions, similar to managers, may utilize checked data of people in unseemly ways. The third issue is that it is hard to differentiate between satisfactory and unsuitable (damaging) utilization of security. According to Caloyannides [13] there are two ways to deal with privacy: the first is through law enforcement, while the second is through intelligence. Law enforcement is not successful in the long term because it is proven that attacks have been doubled when such enforcement was used and favored over intelligence after 9/11. Rather, the author proposed using enhanced intelligence forces that are professional, effective, and functional in place of using law enforcement.

Rosson and Carroll [14] define the term Usability Engineering as: a set of concepts and techniques that are used to plan, achieve, and verify usability goals and objectives for systems.

Hix and Hartson [15] define usability engineering as,"a process through which usability characteristics are specified, quantitatively and early in the development process."

Good et al. [16] specifies a set of five steps of the usability engineering process as the following: 1) defining measurable usability objectives; 2) setting planned levels to achieve the defined objectives; 3) analyzing the impact of design solutions; 4) incorporating user-derived feedback during and after design stage; and 5) iterating between the design and evaluation until the defined usability objectives are achieved. It is a strategy oriented process model that provides the developers and software engineers from beginners to experts with a guided support that covers most resources and knowledge needed to develop a secure software. The process provides two levels of guidance; the first one is strategic which helps the developers choosing one among several strategies. The second level guidance is tactical helping developers achieve their selection for producing secure software.

III. SOFTWARE SECURITY GOALS

There are three primary measurements to accomplish security in the PC framework i.e., secrecy, trustworthiness and accessibility. These three viewpoints additionally are eluded as CIA. Secrecy intends to reveal data to individuals or projects that are approved to approach that data. Respectability, guarantees that a framework plays out its proposed capacity in a healthy way, free from conscious or incidental unapproved control of the framework. Accessibility, guarantees that frameworks work expeditiously, and benefit isn't denied to approve clients. Dangers to the classification of the framework can uncover data to individuals or projects that are not approved to approach that data. Dangers to the trustworthiness of the framework and its information can harm or degenerate the software or its information. Dangers to the accessibility of the framework and its information can confine access to the software or its information for approved clients. Throughout the years, distinctive security component was utilized to accomplish these objectives like confirmation and approval. Be that as it may be, the rate of assaults to PC framework is expanding, and the circumstances might be basic particularly for vast frameworks. Hence, numerous scientists focus on the software security field so as to deliver highly secured framework [3]. This influence is mentioned recurrently in the literature: NFRs often influence the system architecture more than functional requirements do; "the rationale behind each architecture decision is mostly about achieving certain NFRs", "business goals and their associated quality attribute requirements strongly influence a system's architecture [17].

IV. SOFTWARE SECURITY BASIC CONCEPTS

This section will explain some of security terms that are used in this paper.
Asset: Is whatever has an incentive to the association, its business activities and their congruity, including data assets that help the association's central goal.

Vulnerability: A shortcoming in the plan, task, usage or any procedure in the framework which opens the framework to a risk. [4] Characterized as a shortcoming of a benefit or gathering of advantages that can be misused by at least one assailant.

Threat: A conceivable risk that may result in mischief of frameworks and association.
Attack: A real occasion done by an individual; aggressor to hurt a resource of the software through misusing weakness.
Risk: a potential for misfortune, harm, or devastation of an advantage because of a danger abusing helplessness.

Software Security Requirement: is a non-useful necessity that evokes a control, imperativeness, defends, or

countermeasures to keep away from or expel security vulnerabilities from prerequisites, plans or codes [4-5].
Confidentiality: intends to uncover data to individuals or projects that are approved to approach that data.
Integrity: guarantees that a framework plays out its expected capacity in a healthy way, free from intentional or accidental unapproved control of the framework.

Availability: guarantees that framework works instantly, and benefit isn't denied to approve clients.
Process: is an occasion of a PC program that is being executed.
Secure software process: A lot of exercises used to create and convey a secure software arrangement.

V. DETAILS OF SOME STAGES

Architectural level experience has demonstrated that a decent method to assemble reliable frameworks is to structure them into a lot of various leveled layers. Layers of deliberation uphold descending just utilitarian conditions. Unit A is said to rely upon B at whatever point an activity of B, or a change to B, or all out inaccessibility of B, can influence A [6]. Some essential issues for chains of command are: what capacities to incorporate into each layer and how much security we require in each layer to achieve a planned dimension of security for the entire framework. Not all layers required being similarly secure, the lower levels are increasingly basic and need more grounded insurance. Security and other non-useful prerequisites influence all the structural dimensions of a framework. The Layers design [7] is consequently a decent beginning stage to apply these prerequisites. Utilizing layers we can characterize designs at all dimensions that together actualize a secure or dependable framework. The fundamental thought of the Layered design is the disintegration of a framework into progressive layers of deliberation, where the more elevated amounts utilize the administrations of the lower levels. We have talked about it before, why every one of these dimensions must be composed to guarantee security [2] and how the meaning of non-utilitarian particulars ought to be done at a particular dimension [2]. The calculated undertaking models, both static and dynamic, are characterized at the application level. It is here where the security (and other sort) arrangements of the organization ought to be connected. At this dimension the semantics of the application are surely known and jobs can be utilized to apply the need-to-know approach; i.e., we can characterize the required rights as per the elements of every job [3]. Other non-practical viewpoints are likewise determined here, e.g., the required level of unwavering quality. The lower levels authorize the limitations characterized at the more elevated amounts. Each dimension has its very own security instrument and ought to partake in implementing the security requirements. Respondents of the survey reported six different roles, of which software developer was the most common. Only four respondents reported working in testing[18]. These testers work on higher level testing, as unit testing and acceptance tests are done by the developers working on a feature. For instance, a DBMS Security confirmation and testing Requirements Analysis Design Implementation Secure UCs Authorization governs in calculated model Rule requirement through engineering Language authorization Security test cases upholds the approvals in the application by limiting access to database things; this limitation is proliferated down to control access to the records where this information dwells.

VI. USE CASES AND POSSIBLE ATTACKS

As we showed before, use cases characterize each of the collaboration with the framework we can discover from them the rights required by these jobs to play out their work (need to know). The utilization cases are used to cast a ballot framework that permits casting a ballot in the area or in a region that isn't your very own region, through the Internet. A voter has to directly enroll and to cast a ballot, the area officer keeps rundown of enlisted voters and counts the votes. We would then be able to relate conceivable assaults to utilize cases. For instance, a conceivable assault against voter enrollment compares to an impostor endeavoring to get enlisted with false personality or characteristics. An assault against remote casting a ballot would be an endeavor to send an invalid vote or to block a vote with the reason for evolving it. Relating assaults to utilize cases gives an efficient and generally total rundown of conceivable assaults. Each assault can be examined to perceive how it very well may be cultivated in the particular condition. The rundown would then be able to be utilized to control the plan and to choose security items. It can likewise be utilized to assess the last plan by dissecting, if the framework barriers can stop every one of these assaults.

VII. AUTHORIZED APPLICATIONS

We utilize the entrance grid and RBAC as reference models. Staggered models are likewise conceivable yet when utilized at the application level they are excessively unbending; in any case, they are helpful at lower levels. When we apply the entrance network show, the subsequent stage is to characterize designs that speak to approval principles or arrangements, [6]. This model portrays a passage of the entrance lattice, (s, o, t, p, f), where s is a subject, o is an insurance object, t is an entrance type, p is a predicate obliging the use of the standard , and f is a duplicate banner, demonstrating if the privilege can be exchanged [8]. The classes are theoretical classes and explicit approval models are characterized by solid classes. Reasonable or area models of frameworks can be manufactured utilizing investigation designs [5] and we have built up a gathering of these examples for angles, for example, inventories, arrange preparing, and others. These security examples can be connected to investigation examples to characterize semantic subsystems that join the upsides of examples with the benefits of abnormal state approval definition. For this situation, the client of the example would have a structure to characterize the particular rights his application requires. For instance, if [8] we demonstrated an examination design for a secure stock framework, in that display, the reviewer is approved to check for disparities in stock, while the stock manager is approved to address or modify these inconsistencies. Correspondingly, the stock administrator can include new stockrooms, and so on. The particular rights for every job originate from use cases and are inferred as in [3]. Each utilization case has a lot of performers who cooperate with the framework. In the event that on-screen characters are given rights as indicated by their capacities in the utilization instances of the framework, we are executing a need-to-know approach. Beginning from examples at the application level we have to characterize designs for the lower levels. For instance, we have created examples for working frameworks [8, 7], firewalls [10], and other security instruments. For frameworks that utilize web administrations we have created security designs for application firewalls and statement coordination [9].

VIII. SECURE SYSTEM ARCHITECTURE

Some of the secure mechanisms that can help stop the attacks defined through the use cases. For example, remote users would require certificates for authentication, the voting machine hardware and software needs to be certified for security, the precincts are connected through a VPN, etc.

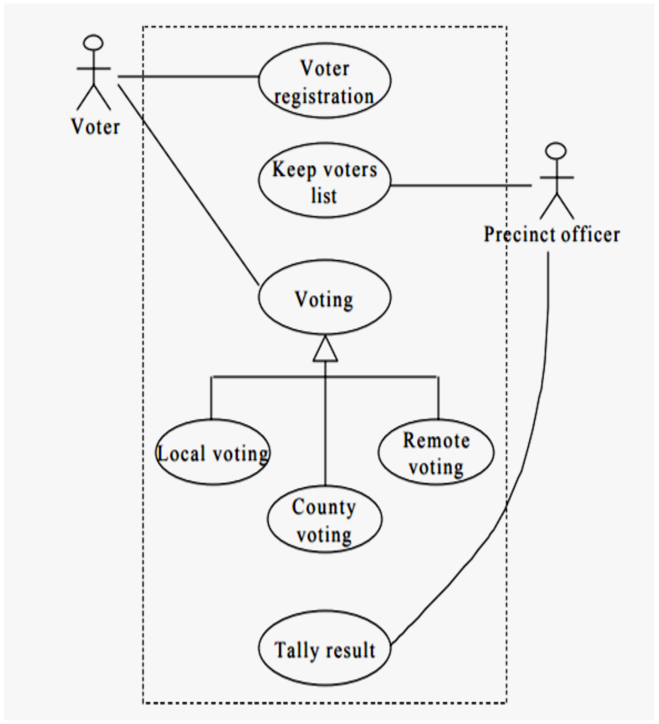

Figure 1: Use for a voting system

IX. PERFORMANCE ANALYSIS

Software engineering assumes a critical job in meeting a software framework's execution. Execution depends to a great extent on the recurrence and nature between segment correspondence and the execution qualities of the segments themselves.

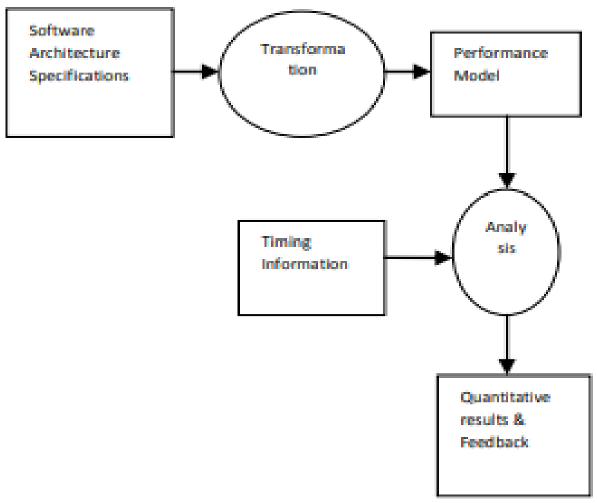
Software Architecture Specifications
Transforma tion
Performance Model
Timing Information
Analy sis
Quantitative results & Feedback

Figure 2: Secure software architecture-based performance analysis

Diverse software engineering based strategies have been created to foresee execution properties, for example, throughput, use of assets, and start to finish inertness. Design based execution investigation procedures change the determination of software engineering into alluring models. At that point, timing data is added to these models. From that point onward, they are broken-down to gauge execution credits quantitatively and to give input about the software engineering. These systems work dependent on accessibility of software curios, for example, necessity and engineering determinations and configuration reports. Since execution is a runtime quality, these approaches require appropriate depiction of the dynamic conduct of a software framework. Regularly, programmed devices are utilized to perform execution investigation once the execution models are made through the general system for dissecting execution at structural dimension. A portion of the benefits of engineering based execution investigation techniques are as per the following: (I) they can help foresee the execution of a framework right off the bat in the software life cycle. (ii)They can be utilized to ensure that execution objectives are met. (iii)They can likewise be utilized to think about the execution of various building decisions. (iv)They can help in discovering bottleneck assets and recognizing potential planning issues before the framework is fabricated.

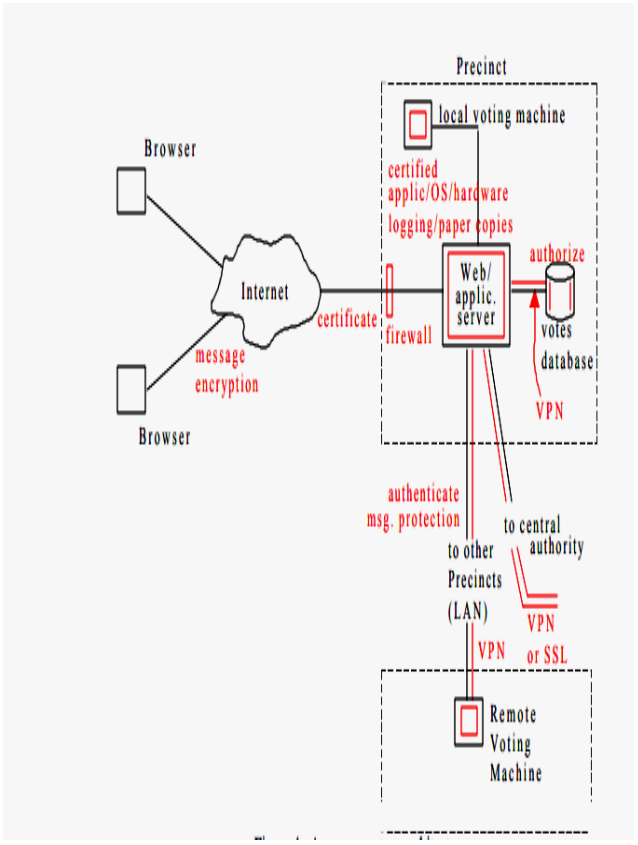

Figure 3: A secure system architecture

X. CONCLUSION AND FUTURE SCOPE

The blend of multilayer designs with examples gives a structure to build up a methodical and reusable way to deal with building frameworks that fulfill explicit non-practical prerequisites. Security designs typify great plan standards and by utilizing them, the creator is certainly applying these standards. Work is expected to include more examples in each dimension and to gather and bring together these examples. We are working now in an inventory of security designs. We likewise need to characterize rules to apply the strategy all the more correctly at each dimension. At long last, we have to assess this system in a genuine situation; for the time being we are applying it to explicit precedents, for example, conveyed therapeutic records, Internet casting a ballot, and appropriated budgetary foundations.

REFERENCES

[1] S. Horing, J. Menard, and R. Staehler, “Stored Program Controlled Network,” Bell System Technical Journal, vol. 61, no. 7, 1982.

[2] D. L. Tennenhouse, J. M. Smith, W. D. Sincoskie, D. J. Wetherall, and G. J. Minden, “A survey of active network research,” IEEE Communications Magazine, vol. 35, no. 1, pp. 80–86, 1997.

[3] M. Casado, T. Garfinkel, A. Akella, M. J. Freedman, D. Boneh, N. McKeown, and S. Shenker, “SANE: A protection architecture for enterprise networks,” in USENIX Security Symposium, 2006.

[4] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker, “Ethane: Taking control of the enterprise,” in ACM SIGCOMM Computer Communication Review, vol. 37, no. 4. ACM, 2007, pp. 1–12.

[5] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “OpenFlow: enabling innovation in campus networks,” ACM SIGCOMM Computer Communication Review, vol. 38, no. 2, pp. 69–74, 2008.

[6] Schneider, T., “Secure Software Engineering Processes: Improving the Software Development Life Cycle to Combat Vulnerability”, SQP VOL. 9, NO. 1, 2006, http://www.asq.org

[7] McGraw, G., Software Security: Building Security In, Addison Wesley, 2006

[8] Verdon, D. and McGraw, G., “Risk Analysis in Software Design,” IEEE Security and Privacy, IEEE CS Press, 2004, volume 2, number 4, pages 79-84.

[9] Lipner, S., “The Trustworthy Computing Security Development Lifecycle,” In Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC ‘04), Tucson, Arizona, USA, 2004, IEEE CS Press, pages 2-13.

[10] Flechais, I., Mascolo, C., and Sasse, M. A., “Integrating Security and Usability into the Requirements and Design Process,” International Journal of Electronic Security and Digital Forensics, Inderscience Publishers, Geneva, Switzerland, 2007, volume 1, number 1, pages 12-26.

[11] Sodiya, A. S., Onashoga, S. A., and Ajayi, O. B., “Towards Building Secure Software Systems,” Issues in Informing Science and Information Technology, Informing Science Institute, California, USA, 2006, volume 3, pages 635-646.

[12] Mead, N. R., Hough, E., and Stehney, T. “Security Quality Requirements Engineering (SQUARE) Methodology,” Technical Report CMU/SEI-2005-TR-009, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, 2005.

[13] L. Futcher and R.v. Solms, “SecSDM: A Model for Integrating Security into the Software Development Life Cycle,” In IFIP International Federation for Information Processing, Volume 237, Proc. of the 5th World Conference on Information Security Education, Springer, 2007, pp. 41-48

[14] I. Flechais, M.A. Sasse, and S.M.V. Hales, “Bringing Security Home: A Process for Developing Secure and Usable Systems,” In Proc. of the New Security Paradigms Workshop (NSPW’07), Ascona, Switzerland, ACM Press, 2003, pp. 49- 57.

[15] J. Gregoire, K. Buyens, B. De Win, R. Scandariato, and W. Joosen, “On the Secure Software Development Process: CLASP and SDL Compared,” In Proc. of the 3rd International Workshop on Software Engineering for Secure Systems (SESS’07), Minneapolis, Minnesota, USA, IEEE CS Press, 2007, pp. 1-1.

[16] M. Graves and M. Zulkernine, “Bridging the Gap: Software Specification Meets Intrusion Detector,” In Proc. of the 4th Annual Conference on Privacy, Security and Trust (PST’06), Ontario, Canada, pp. 265-274.

[17] Jameel Ahmad Qurashi, Sanjay Kumar "*Secure Software Architecture: A Hybrid Approach Based On Non-Functional Security Requirements*." International Journal of Computer Sciences and Engineering 7.1 (2019): 790-794.

[18] *Anooja A, Jameel Ahmad Qurashi, Sanjay Kumar* "A Survey Study of Various Software Cost Effort Estimation in Perspective of India." *International Journal of Computer Sciences and Engineering 7.1 (2019): 928-933.*

Authors Profile

Mr. Jameel Ahmad Qurashi has pursed Bachelor in Computer Applications from University of Kashmir, J&K in 2011 and Master in Computer Applications from University of Kashmir, J&K in year 2014. He is currently pursuing Ph.D. and currently working as Assistant Professor in Department of Computer and System Sciences, Jaipur National University, Jaipur, India since 2016. He has been granted Quarterly Franklin Membership from London Journals United kingdoms in 2019. He has published 06 research papers in reputed international, national and conferences including IEEE and are also available online. His main research work focuses on Software Security, NFR Security, Hybrid Architectural security and Privacy, Algorithmic analysis, IoT and Computational Intelligence based education. He has 5 years of teaching experience and 4 years of Research Experience.

Mr. Harvir Singh has pursed B.Tech and M.Tech in Computer Science from Aligarh Muslim University in 2011 and Ph.D. from Dr. B. R. Ambedkar University, Agra and currently working as Professor and Director in Department of Engineering (All wings), Jaipur National University, Jaipur, India since 2019. He is member of CIA and executed 25 research projects for National and international agencies. He has supervised over 34 thesis and projects and has over 30+ research publications in reputed international, national journals and conferences including IEEE. His main research work focuses on Artificial intelligence, Data Mining, Machine learning, neural networks Software Security, NFR Security, Hybrid Architectural security and Privacy, Algorithmic analysis, IoT and Computational Intelligence based education. He has 26+ years of teaching experience and 30 years of Research Experience.

Mr. Vijay Nunia has pursed B.Tech in Computer Science from University of Rajasthan Technical University, Kota in 2012 and Master in Computer Science from University of Jaipur National University, Jaipur in year 2015. He is currently pursuing Ph.D. and currently working as Assistant Professor in Department of Computer and System Sciences, Jaipur National University, Jaipur, India since 2015. He has been granted Reviewer Membership in ESN Publication in 2019. He has published 02 research papers in reputed international journals and it's also available online. His main research work focuses on GSM, MANET Routing Protocol, Network Security. He has 3 years of teaching experience.