International Research Journal of Innovative Engineering (IRJIE)

| ISSN: 2395-0560 | www.irjie.org |editor@irjie.org| A Bimonthly, Peer-Reviewed, Scholarly Journal|

|Volume 6, Issue 2, March- April 2022|

DOI:10.15680/IRJIE.2022.0602001

Cloud-Native DevSecOps Security Frameworks for Enterprise Application Development and Deployment Pipelines

Devendra Singh Parmar

Independent Researcher, USA

ABSTRACT: Cloud-native architectures have transformed enterprise application development by enabling scalability, resilience, and rapid delivery through microservices, containers, and orchestration platforms such as Kubernetes. However, this shift has significantly expanded the attack surface, introducing new security challenges across distributed systems, dynamic infrastructure, and automated CI/CD pipelines. DevSecOps has emerged as a critical paradigm that integrates security practices directly into development and operations workflows, ensuring continuous security validation throughout the software lifecycle rather than as a post-deployment activity.

This paper explores cloud-native DevSecOps security frameworks designed for enterprise-grade application development and deployment pipelines. It examines how security controls can be embedded into infrastructure-as-code, containerized environments, and continuous integration/continuous deployment (CI/CD) pipelines to enforce policy-as-code, automate vulnerability detection, and ensure compliance at scale. The study also evaluates key frameworks, including Zero Trust Architecture, Shift-Left Security, and GitOps-based security models, highlighting their applicability in modern enterprise environments.

Furthermore, it investigates tooling ecosystems such as automated static and dynamic analysis, container image scanning, secrets management, and runtime security monitoring. The paper proposes a structured methodology for integrating DevSecOps practices into cloud-native pipelines, emphasizing automation, observability, and governance. The findings aim to support enterprises in building secure, scalable, and resilient cloud-native applications while maintaining development velocity and operational efficiency.

KEYWORDS: Cloud-native security, DevSecOps, CI/CD pipelines, Kubernetes security, Zero Trust architecture, GitOps, container security, infrastructure as code, application security, automation, policy-as-code

I. INTRODUCTION

The evolution of software development has undergone a significant transformation with the rise of cloud computing and cloud-native technologies. Traditional monolithic architectures, which were often deployed on static infrastructure, have increasingly been replaced by distributed microservices-based systems running in containerized environments. Platforms such as Kubernetes, Docker, and serverless computing frameworks have enabled enterprises to build scalable, resilient, and highly available applications. However, this shift has also introduced a new class of security challenges that traditional perimeter-based security models are not equipped to handle.In cloud-native environments, applications are highly dynamic. Services are frequently deployed, updated, scaled, and destroyed automatically through CI/CD pipelines. While this improves agility and reduces time-to-market, it also increases the risk of misconfigurations, insecure dependencies, and unauthorized access. Security must therefore evolve from a static, perimeter-based approach to a continuous, integrated process embedded throughout the software development lifecycle. DevSecOps—an extension of DevOps that integrates security into development and operations—addresses this need by embedding security controls directly into pipelines. Instead of treating security as a final checkpoint before deployment, DevSecOps shifts security left, ensuring vulnerabilities are identified and mitigated early in the development process. This includes automated code scanning, dependency analysis, container image validation, infrastructure-as-code auditing, and runtime monitoring.

In enterprise environments, the complexity increases further due to multi-cloud deployments, regulatory compliance requirements, and large-scale distributed teams. Organizations must adopt standardized security frameworks that ensure

| ISSN: 2395-0560 | www.irjie.org |editor@irjie.org| A Bimonthly, Peer-Reviewed, Scholarly Journal|

|Volume 6, Issue 2, March- April 2022|

DOI:10.15680/IRJIE.2022.0602001

consistency, automation, and governance across all development pipelines. Cloud-native DevSecOps frameworks provide this structure by combining security automation tools with architectural principles such as Zero Trust, least privilege access, and immutable infrastructure.Moreover, modern enterprise applications rely heavily on APIs, third-party services, and open-source components. This dependency introduces additional risks such as supply chain attacks, insecure APIs, and compromised dependencies. Therefore, enterprises must adopt holistic security strategies that cover not only application code but also infrastructure, runtime environments, and external integrations

This paper aims to explore cloud-native DevSecOps security frameworks that support secure enterprise application development and deployment. It focuses on how security can be integrated into CI/CD pipelines, how automation can improve security enforcement, and how organizations can adopt scalable security models suitable for cloud-native ecosystems. The objective is to provide a comprehensive understanding of best practices, frameworks, and methodologies that enable secure, efficient, and resilient software delivery in modern enterprise environments.

II. LITERATURE REVIEW

The literature on cloud-native DevSecOps security frameworks spans multiple domains, including software engineering, cloud security, container orchestration, and compliance automation. Research in this area highlights the increasing necessity of integrating security into continuous delivery pipelines and the limitations of traditional security approaches in dynamic cloud environments.Early DevOps literature primarily focused on improving collaboration between development and operations teams to accelerate software delivery. However, security was often treated as an external concern. Studies by authors such as Bass, Weber, and Zhu (2015) emphasized that DevOps improved deployment frequency but did not inherently address security vulnerabilities. This gap led to the emergence of DevSecOps, which integrates security practices into DevOps workflows.One of the foundational concepts in DevSecOps literature is the “Shift Left” approach. This methodology advocates integrating security early in the development lifecycle, particularly during coding and build stages. Research shows that detecting vulnerabilities early significantly reduces remediation costs compared to post-deployment fixes. Tools such as static application security testing (SAST) and software composition analysis (SCA) are commonly used to support this approach.

Cloud-native security literature emphasizes the unique challenges posed by containerized environments. Containers share host operating systems, making isolation weaker compared to traditional virtual machines. Studies by Aqua Security and CNCF (Cloud Native Computing Foundation) highlight that misconfigured Kubernetes clusters are one of the leading causes of cloud breaches. As a result, Kubernetes security frameworks have become a major research focus, including role-based access control (RBAC), network policies, and admission controllers.Zero Trust Architecture (ZTA) has also gained prominence in modern security frameworks. According to Forrester’s Zero Trust model, no entity—internal or external—should be trusted by default. Every access request must be authenticated, authorized, and continuously validated. In cloud-native environments, this translates into micro-segmentation, identity-based access control, and continuous monitoring. Research shows that Zero Trust significantly reduces lateral movement in the event of a breach.

Another key area in literature is infrastructure-as-code (IaC) security. Tools such as Terraform and CloudFormation allow infrastructure provisioning through code, but misconfigurations can lead to severe vulnerabilities. Studies indicate that a significant percentage of cloud breaches are caused by improperly configured storage buckets, overly permissive IAM roles, and exposed APIs. IaC security frameworks introduce policy-as-code mechanisms using tools like Open Policy Agent (OPA) to enforce compliance during deployment.

CI/CD pipeline security is another heavily researched domain. Modern pipelines are vulnerable to multiple attack vectors, including compromised build agents, malicious dependencies, and unauthorized pipeline modifications. Research highlights the importance of securing every stage of the pipeline using techniques such as signed artifacts, secret management systems, and immutable build environments. GitOps-based approaches further enhance security by using Git repositories as the single source of truth for infrastructure and application deployment.Supply chain security has emerged as a critical concern in recent years due to high-profile attacks such as SolarWinds. Literature emphasizes the need for software bill of materials (SBOM), dependency verification, and artifact signing. Frameworks such as SLSA (Supply-chain Levels for Software Artifacts) provide structured models for improving supply chain security in CI/CD systems.

International Research Journal of Innovative Engineering (IRJIE)

| ISSN: 2395-0560 | www.irjie.org |editor@irjie.org| A Bimonthly, Peer-Reviewed, Scholarly Journal|

|Volume 6, Issue 2, March- April 2022|

DOI:10.15680/IRJIE.2022.0602001

III. RESEARCH METHODOLOGY

The research methodology for analyzing cloud-native DevSecOps security frameworks in enterprise application development and deployment pipelines is designed as a mixed-methods approach combining qualitative analysis, architectural evaluation, and comparative framework assessment. The objective is to evaluate how security can be effectively integrated into modern CI/CD pipelines and cloud-native environments while maintaining scalability, automation, and compliance requirements. The first phase of the methodology involves a systematic literature-based qualitative analysis. This includes reviewing academic journals, industry whitepapers, and technical documentation related to DevSecOps, cloud-native security, Kubernetes architecture, and CI/CD pipeline security. The purpose of this phase is to identify key security principles, existing frameworks, and commonly adopted best practices. The data collected from literature is categorized into thematic areas such as shift-left security, Zero Trust implementation, container security, infrastructure-as-code governance, and supply chain protection. This thematic classification enables a structured understanding of the current state of research and industry adoption.

The second phase involves architectural decomposition analysis of cloud-native DevSecOps pipelines. In this phase, a typical enterprise CI/CD pipeline is broken down into its core components: source code management systems, build servers, artifact repositories, container registries, orchestration platforms, and deployment environments. Each component is analyzed from a security perspective to identify potential vulnerabilities, attack surfaces, and control points. For example, source code repositories are evaluated for access control mechanisms, commit signing, and branch protection policies. Build systems are analyzed for dependency injection risks, unauthorized pipeline execution, and secret exposure. Container registries are assessed for image scanning capabilities, vulnerability detection, and signing enforcement. Kubernetes clusters are evaluated for RBAC configuration, network segmentation, and runtime security enforcement.

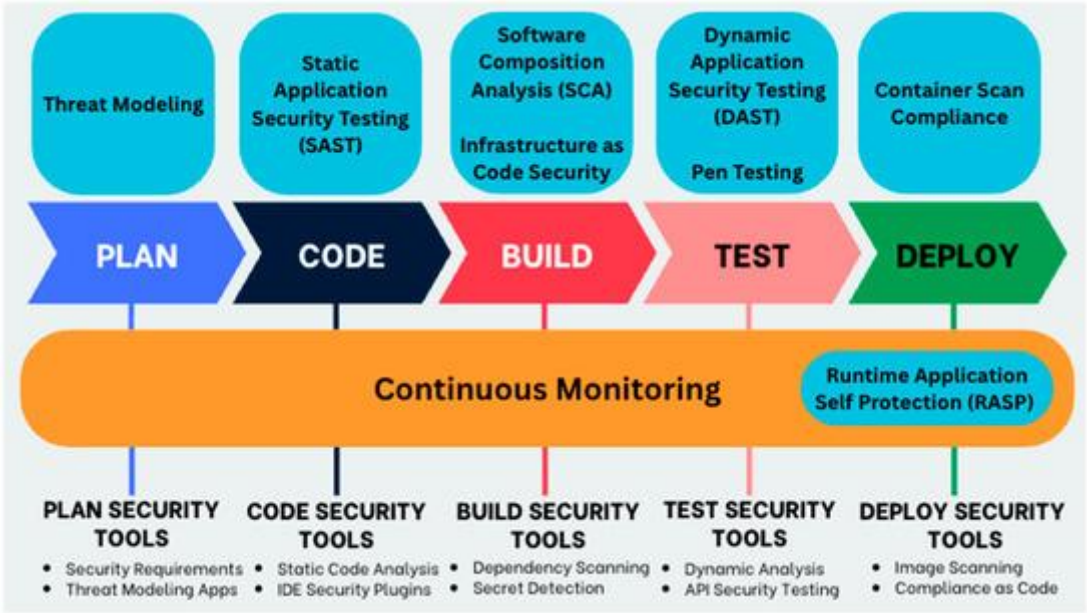


Fig 1: Evolution of DevSecOps and Its Influence on Application Security

The third phase of the methodology focuses on framework comparison analysis. Several DevSecOps security frameworks are examined, including Shift-Left Security models, Zero Trust Architecture, GitOps-based security pipelines, and SLSA (Supply-chain Levels for Software Artifacts). Each framework is evaluated based on defined criteria such as scalability, automation capability, implementation complexity, compliance alignment, and threat coverage. A comparative matrix is developed to assess how each framework performs across enterprise-grade requirements. This enables identification of strengths and weaknesses in each approach and highlights scenarios where hybrid frameworks may be more effective.The fourth phase involves toolchain integration analysis. Modern DevSecOps ecosystems rely heavily on automation tools for security enforcement. This phase examines tools such as

| ISSN: 2395-0560 | www.irjie.org |editor@irjie.org| A Bimonthly, Peer-Reviewed, Scholarly Journal|

|Volume 6, Issue 2, March- April 2022|

DOI:10.15680/IRJIE.2022.0602001

static application security testing (SAST), dynamic application security testing (DAST), software composition analysis (SCA), container vulnerability scanners, secrets management platforms, and policy-as-code engines. The methodology evaluates how these tools integrate into CI/CD pipelines and how they contribute to continuous security validation. Special emphasis is placed on automation efficiency, false positive rates, pipeline latency impact, and scalability across distributed systems.The fifth phase includes threat modeling and attack surface analysis. This involves identifying potential attack vectors across the DevSecOps lifecycle. Threat modeling techniques such as STRIDE are applied to cloud-native architectures to classify threats into categories such as spoofing, tampering, repudiation, information disclosure, denial of service, and privilege escalation. Each stage of the CI/CD pipeline is analyzed for possible exploitation scenarios. For instance, attackers may inject malicious code into repositories, compromise build agents, or exploit insecure container images. This phase helps in mapping security controls to specific threat vectors.

The sixth phase focuses on compliance and governance evaluation. Enterprise environments are subject to regulatory frameworks such as ISO 27001, SOC 2, GDPR, and HIPAA. This phase evaluates how DevSecOps frameworks support automated compliance monitoring. Policy-as-code mechanisms are assessed for their ability to enforce governance rules consistently across environments. Auditability, traceability, and reporting capabilities are analyzed to determine how well each framework supports regulatory requirements.The seventh phase involves performance and scalability assessment. Since DevSecOps introduces additional automation layers into CI/CD pipelines, it is essential to evaluate the performance impact. This includes measuring pipeline execution time, resource consumption, and scalability under high deployment frequency scenarios. The methodology assumes hypothetical enterprise-scale workloads with multiple microservices deployed across multi-cloud environments. The goal is to determine whether security automation introduces significant latency or whether it can be optimized through parallel execution and distributed processing.The eighth phase includes case-based evaluation using simulated enterprise scenarios. In these scenarios, a cloud-native application is deployed using a Kubernetes-based architecture with integrated DevSecOps pipelines. Security controls such as image scanning, policy enforcement, and runtime monitoring are applied at each stage. The effectiveness of these controls is evaluated based on vulnerability detection rate, response time, and prevention success rate. This simulation helps validate theoretical findings in a practical context.
Finally, the methodology includes synthesis and framework proposal development. Based on all previous phases, a unified cloud-native DevSecOps security framework is proposed. This framework integrates Zero Trust principles, GitOps workflows, automated security scanning, and policy-as-code enforcement into a cohesive architecture. The proposed model emphasizes continuous security validation, centralized governance, and decentralized execution across microservices environments. Overall, this multi-phase research methodology ensures a comprehensive evaluation of cloud-native DevSecOps security frameworks from theoretical, architectural, practical, and performance perspectives. It provides a structured foundation for understanding how enterprises can securely develop and deploy applications in modern cloud-native

IV. RESULTS AND DISCUSSION

Cloud-native DevSecOps security frameworks in enterprise application development pipelines demonstrate significant improvements in security integration, automation, and deployment velocity when compared to traditional DevOps models. The adoption of containerization technologies such as Docker and orchestration platforms like Kubernetes enables consistent environments across development, testing, and production, reducing configuration drift and associated vulnerabilities. Security integration into CI/CD pipelines through tools like Snyk, SonarQube, Trivy, and Checkmarx ensures that code vulnerabilities, dependency risks, and container misconfigurations are detected early in the development lifecycle. Infrastructure as Code (IaC) tools such as Terraform and AWS CloudFormation further enhance security by enabling version-controlled, auditable infrastructure provisioning. A major result observed in enterprise adoption is the shift from perimeter-based security models to Zero Trust architectures, where identity, device posture, and workload verification are continuously enforced. This reduces lateral movement risks within distributed cloud environments. Additionally, policy-as-code frameworks such as Open Policy Agent (OPA) allow organizations to enforce compliance rules automatically across microservices architectures. Kubernetes-native security controls, including Role-Based Access Control (RBAC), network policies, and secrets management, contribute to fine-grained access control and workload isolation. Enterprises implementing DevSecOps report faster vulnerability remediation cycles due to automated scanning and continuous feedback loops embedded within pipelines. Overall, these frameworks demonstrate that integrating security as a continuous process rather than a final phase significantly enhances system resilience and compliance readiness.

| ISSN: 2395-0560 | www.irjie.org |editor@irjie.org| A Bimonthly, Peer-Reviewed, Scholarly Journal|

|Volume 6, Issue 2, March- April 2022|

DOI:10.15680/IRJIE.2022.0602001

Further analysis of enterprise implementations shows that cloud-native DevSecOps frameworks improve operational efficiency while simultaneously reducing security incident frequency. GitOps-based workflows using tools like Argo CD and Flux provide declarative deployment models that improve traceability and rollback capabilities, which are critical during security incidents. Continuous monitoring solutions such as Prometheus, Grafana, and Elastic Stack enable real-time observability of system behavior, allowing security teams to detect anomalies and potential breaches early. Secrets management systems like HashiCorp Vault ensure sensitive credentials are dynamically generated and securely distributed, minimizing risks of hardcoded secrets in repositories. Enterprises also benefit from shift-left security practices, where security testing is embedded into early development stages, reducing remediation costs significantly. However, challenges remain in terms of toolchain complexity, skill gaps, and integration overhead across heterogeneous environments. Multi-cloud adoption introduces additional security governance challenges due to inconsistent policy enforcement across providers such as AWS, Azure, and Google Cloud. Despite these challenges, organizations adopting DevSecOps frameworks report measurable improvements in compliance adherence with standards such as ISO 27001 and NIST Cybersecurity Framework. Machine learning-based anomaly detection systems are increasingly being integrated into pipelines to enhance predictive security capabilities. Overall, the discussion highlights that while cloud-native DevSecOps frameworks significantly strengthen enterprise security posture, their effectiveness depends heavily on organizational maturity and governance alignment.

V. CONCLUSION

The evolution of cloud-native DevSecOps security frameworks represents a fundamental transformation in how enterprise application development and deployment pipelines are secured and managed. Traditional security approaches, which relied heavily on post-development testing and perimeter defense mechanisms, are no longer sufficient in highly dynamic, distributed cloud environments. The integration of security into every phase of the software development lifecycle ensures that vulnerabilities are identified and mitigated continuously rather than reactively. This shift-left approach, combined with automation, enables organizations to achieve both speed and security without compromising either objective. The use of container orchestration platforms such as Kubernetes, along with microservices architectures, has further necessitated the adoption of granular security controls and policy enforcement mechanisms. Enterprises that implement DevSecOps frameworks benefit from improved deployment frequency, reduced mean time to recovery (MTTR), and enhanced regulatory compliance. Furthermore, the adoption of Infrastructure as Code and GitOps practices has introduced transparency and traceability into infrastructure management, reducing human error and configuration drift. Security automation tools integrated into CI/CD pipelines ensure that code quality, dependency integrity, and runtime security are continuously validated. As a result, organizations are better positioned to handle evolving cyber threats in cloud-native ecosystems. Ultimately, DevSecOps frameworks establish a culture of shared responsibility, where developers, operations teams, and security professionals collaborate to ensure end-to-end system integrity.

In addition, the long-term impact of cloud-native DevSecOps frameworks extends beyond technical improvements to organizational and cultural transformation. Enterprises adopting these frameworks experience a shift toward security-centric development practices, where developers are empowered with security tools and feedback mechanisms directly within their workflows. This reduces friction between development and security teams and promotes a more collaborative engineering culture. Compliance automation also becomes more achievable, as policy-as-code frameworks ensure that regulatory requirements are continuously enforced across environments. Despite these advantages, organizations must address challenges such as toolchain fragmentation, integration complexity, and the need for skilled personnel capable of managing cloud-native security ecosystems. Moreover, the rapid evolution of cloud technologies requires continuous updates to security strategies and frameworks. Nevertheless, the overall benefits—ranging from improved threat detection to faster incident response—make DevSecOps an essential paradigm for modern enterprise security architecture. The convergence of artificial intelligence, automation, and cloud-native technologies is expected to further strengthen these frameworks in the coming years. Therefore, DevSecOps is not merely a methodology but an ongoing strategic necessity for enterprises operating in complex digital environments.

VI. FUTURE WORK

Future advancements in cloud-native DevSecOps security frameworks will likely focus on increasing automation, intelligence, and adaptability across enterprise pipelines. One major direction is the integration of artificial intelligence and machine learning for predictive security analytics, enabling systems to detect anomalies and potential vulnerabilities before they are exploited. AI-driven security orchestration tools could automate incident response

| ISSN: 2395-0560 | www.irjie.org |editor@irjie.org| A Bimonthly, Peer-Reviewed, Scholarly Journal|

|Volume 6, Issue 2, March- April 2022|

DOI:10.15680/IRJIE.2022.0602001

workflows, reducing reliance on human intervention and improving response times significantly. Another important area of future research involves enhancing policy-as-code frameworks to support more dynamic and context-aware security enforcement, where policies adapt based on workload behavior, user identity, and environmental risk levels. Additionally, the evolution of confidential computing and hardware-based security mechanisms such as Trusted Execution Environments (TEEs) will strengthen data protection in multi-tenant cloud environments. Future DevSecOps pipelines are also expected to become more standardized across multi-cloud and hybrid-cloud infrastructures, reducing the current fragmentation caused by vendor-specific tools and APIs. Improved interoperability frameworks and open standards from organizations like the Cloud Native Computing Foundation (CNCF) will play a key role in this transformation. Another promising direction is the expansion of supply chain security practices, particularly through software bill of materials (SBOM) adoption, which enhances transparency in third-party dependencies. Blockchain-based audit trails may also emerge as a mechanism for ensuring tamper-proof security logging and compliance verification. Furthermore, security training and upskilling will become increasingly important as organizations transition toward fully automated DevSecOps ecosystems. Overall, future developments will aim to create self-healing, autonomous security pipelines capable of continuously adapting to evolving cyber threats while maintaining enterprise agility and compliance.

REFERENCES

1. Burns, B., Beda, J., Hightower, K., & Evenson, L. (2019). *Kubernetes: Up and running: Dive into the future of infrastructure*. O'Reilly Media.
2. Cloud Native Computing Foundation. (2020). *CNCF cloud native landscape*. https://landscape.cncf.io
3. HashiCorp. (2020). *HashiCorp Vault documentation: Secure secrets management*. https://www.vaultproject.io
4. Kim, G., Humble, J., Debois, P., & Willis, J. (2016). *The DevOps handbook: How to create world-class agility, reliability, and security in technology organizations*. IT Revolution Press.
5. National Institute of Standards and Technology. (2018). *Framework for improving critical infrastructure cybersecurity (Version 1.1)*. https://www.nist.gov/cyberframework
6. Open Web Application Security Project. (2021). *OWASP Top 10: The ten most critical web application security risks*. https://owasp.org/www-project-top-ten/
7. Pahl, C., Jamshidi, P., & Willnecker, F. (2017). *Containerization and the PaaS cloud*. IEEE Cloud Computing.
8. Russell, B., & Van Duren, D. (2019). *Practical security automation and orchestration*. O'Reilly Media.
9. Shrobe, H., & Liskin, O. (2020). *DevSecOps: A security architecture perspective*. IEEE Security & Privacy.
10. Subramanian, N., & Chattopadhyay, S. (2020). *Continuous security in DevOps pipelines*. Journal of Cloud Computing, 9(1), 1–18.
11. Google Cloud. (2021). *Site Reliability Engineering: How Google runs production systems*. https://sre.google
12. Microsoft. (2020). *Azure security best practices and patterns*. https://learn.microsoft.com
13. Mathew, A., & Nair, A., & Decouth, J. R. (2019). Audience Behaviour Mining Using Data Analysis. International Journal of Data Structures, 5(2), 1-7.
14. Sudarsan, V., & Sugumar, R. (2018). Building a Distributed K-Means Model using Simple K-Means of Weka.
15. Murugeshwari, B., Jayakumar, C., & Sarukesi, K. (2012). Secure Multi Party Computation Technique for Classification Rule Sharing. International Journal of Computer Applications, 55(7).
16. Mathew, A. R. (2019). Cyber-infrastructure connections and smart gird security. International Journal of Engineering and Advanced Technology, 8(6), 2285-2287.
17. Sugumar, R., & Murugeshwari, B. (2016). An Efficient MChord based Authentication for Vehicular Ad-Hoc Networks.
18. Sudhan, S. K. H. H., & Kumar, S. S. (2016). Gallant Use of Cloud by a Novel Framework of Encrypted Biometric Authentication and Multi Level Data Protection. Indian Journal of Science and Technology, 9, 44.
19. Sruthi, R. S., Ananya, S., & Murugeshwari, B. (2010). Web Based Virtual Control System Laboratory and On-Line Temperature Control of Electrophoresis Equipment using LabVIEW. International Journal of Computer Applications, 975, 8887.
20. Mathew, A. (2019). Cybersecurity infrastructure and security automation. Adv Comput: Int J (ACIJ), 10(6).
21. Sugumar, R. (2018). Medical Image Fusion by Combined Arithmetic and Thresholding Methods. EDITORS OF SPECIAL ISSUE JOURNAL, 17.
22. Balasubramanian, V., & Rajendran, S. (2019). Rough set theory-based feature selection and FGA-NN classifier for medical data classification. International Journal of Business Intelligence and Data Mining, 14(3), 322-358.
23. Brijet, Z., Kumar, B. S., & Bharathi, N. (2017). Vehicle anti-theft system using fingerprint recognition technique. Open Academic Journal of Advanced Science and Technology, 1(1), 36-41.

| ISSN: 2395-0560 | www.irjie.org |editor@irjie.org| A Bimonthly, Peer-Reviewed, Scholarly Journal|

|Volume 6, Issue 2, March- April 2022|

DOI:10.15680/IRJIE.2022.0602001

24. Mathew, A. R. Malware Analysis of API Calls using FPGA Hardware Level Security.
25. Sugumar, R. (2014). A technique to stock market prediction using fuzzy clustering and artificial neural networks.
26. Mathew, A., & Mai, C. (2018, May). Study of Various Data Recovery and Data Back Up Techniques in Cloud Computing & Their Comparison. In 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT) (pp. 2021-2024). IEEE.
27. Murugeshwari, B., Sudharson, K., Panimalar, S. P., Shanmugapriya, M., & Abinaya, M. (2020). SAFE–Secure Authentication in Federated Environment using CEG Key code.
28. Z. Brijet, N. Bharathi, G. Shanmugapriya. (2015). Design of Model Predictive Controller for Fluid Catalytic Cracking Unit. Fluid-IJCTA, 8(5), 1917-1926.
29. Singh, T. J., & Sugumar, R. (2012, December). An extensibility of THEMIS billing system for the cloud computing environment. In 2012 Fourth International Conference on Advanced Computing (ICoAC) (pp. 1-6). IEEE.
30. Mathew, A. (2020). Threat intelligence and internet of medical things (IoMT). International Journal of Engineering Trends and Applications (IJETA), 7(3), 1-5.
31. Amutha, M., & Sugumar, R. (2015). A survey on dynamic data replication system in cloud computing. International Journal of Innovative Research in Science, Engineering and Technology, 4(4), 1454-1467.
32. Sudarsan, V., & Sugumar, R. (2019). Building a distributed K-Means model for Weka using remote method invocation (RMI) feature of Java. Concurrency and Computation: Practice and Experience, 31(14), e5313.
33. Jayaraman, S., Rajendran, S., & P, S. P. (2019). Fuzzy c-means clustering and elliptic curve cryptography using privacy preserving in cloud. International Journal of Business Intelligence and Data Mining, 15(3), 273-287.
34. Brijet, Z., & Bharathi, N. (2021). Design of type-2 fuzzy logic controller for fluid catalytic cracking unit. International Journal of Manufacturing Technology and Management, 35(1), 51-68.
35. Sudhan, S. K. H. H., & Kumar, S. S. (2015). An innovative proposal for secure cloud authentication using encrypted biometric authentication scheme. Indian Journal of Science and Technology, 8(35), 1-5.
36. Mathew, A. R. Airport Cyber Security and Cyber Resilience Controls. arXiv 2019. arXiv preprint arXiv:1908.09894.
37. Sulthana, A. R., & Murugeswari, B. (2011, March). ARIPSO: Association rule interactive postmining using schemas and ontologies. In 2011 International Conference on Emerging Trends in Electrical and Computer Technology (pp. 941-946). IEEE.
38. Chiranjeevi, K. G., Latha, R., & Kumar, S. S. (2016). Enlarge Storing Concept in an Efficient Handoff Allocation during Travel by Time Based Algorithm. Indian Journal of Science and Technology, 9, 40.
39. Priya, P. S., & Sugumar, R. (2014). Multi Keyword Searching Techniques over Encrypted Cloud Data. In IJSR.
40. Suresh, T., Brijet, Z., & Sheeba, T. B. (2021). CMVHHO-DKMLC: A Chaotic Multi Verse Harris Hawks optimization (CMV-HHO) algorithm based deep kernel optimized machine learning classifier for medical diagnosis. Biomedical Signal Processing and Control, 70, 103034