Securing Devops Pipelines

Yash Khemani[1], Saurabh Saw[2], Satyam Agrawal[3], Siddhant Verma[4]

Mr. Omprakash Dewangan[5] Assistant Professor

Department of Computer Science and Information Technology1,2,3,4,5,

Kalinga University, Raipur, Chhattisgarh, India

Abstract— **In recent times, the popularity of the DevOps methodology in software development and deployment has grown significantly due to its ability to enhance speed and efficiency. However, this approach brings forth new security risks and presents challenges. This research paper aims to delve into the significance of securing DevOps pipelines by providing an overview of the risks and threats involved. Additionally, it outlines recommended best practices and tools for ensuring the security of DevOps pipelines. The paper also includes case studies that exemplify successful implementations of DevOps pipeline security. Furthermore, it discusses the limitations and challenges encountered when securing DevOps pipelines and offers recommendations for organizations seeking to enhance the security of their own pipelines. By adopting the recommended best practices and utilizing the suggested tools from this paper, organizations can bolster the security of their DevOps pipelines while mitigating the associated risks and challenges.**

Keywords—DevOps; Pipeline security; Continuous security testing; Infrastructure security; Compliance and governance; Security information and event management (SIEM); Secure software development lifecycle (SDLC); Configuration management; Deployment and release management; Testing and quality assurance; Cultural barriers; Skilled security personnel; Integration with existing security systems

I. INTRODUCTION

DevOps encompasses a collection of methodologies that integrate software development (Dev) and IT operations (Ops) to enhance the speed and efficiency of the software development lifecycle. By automating and optimizing development processes, DevOps facilitates accelerated software releases, transforming the timeline from weeks or months to mere days or even hours.

The significance of DevOps stems from its capacity to expedite time-to-market, ensure superior software quality, and foster seamless collaboration among teams. Through DevOps, organizations gain the ability to promptly adapt to evolving market dynamics and meet customer requirements while simultaneously enhancing the overall excellence of their software products.

A. Explaination of the need for securing DevOps pipelines

However, as DevOps pipelines become increasingly complex and automated, they also become more vulnerable to security threats. DevOps pipelines involve multiple tools, technologies, and processes, all of which need to be secured to prevent security breaches and protect sensitive data.

The need for securing DevOps pipelines is critical, as a single security breach can have significant consequences for an organization. A security breach can result in the theft of intellectual property, financial loss, damage to reputation, and legal consequences.

B. Overview of the paper's structure

The primary objective of this research paper is to thoroughly examine the subject of enhancing the security of DevOps pipelines. The paper will start by providing an overview of common risks and threats faced by DevOps pipelines, followed by an explanation of the essential security practices for DevOps pipelines.

The paper will then discuss the available tools and technologies for securing DevOps pipelines, as well as the challenges and limitations in securing DevOps pipelines. The paper will conclude with case studies of successful implementations of DevOps pipeline security, along with recommendations for organizations looking to improve the security of their DevOps pipelines.

Overall, this research paper will provide a comprehensive understanding of the importance of securing DevOps pipelines and provide practical guidance for organizations looking to secure their DevOps pipelines.

II. RISKS AND THREATS TO DEVOPS PIPELINES

DevOps pipelines involve multiple tools, technologies, and processes, all of which need to be secured to prevent security breaches and protect sensitive data. DevOps pipelines encounter various risks and threats that are commonly observed in their operation:

1. *Insider Threats:* Insider threats are the most significant risk to DevOps pipelines. Employees, contractors, or third-party vendors who have access to the system can steal sensitive information or sabotage the pipeline.

2. *Misconfiguration:* Misconfigurations in the pipeline can leave it vulnerable to attacks. For instance, not setting up proper access controls can allow unauthorized access to the pipeline.

3. *Vulnerabilities in Third-Party Components:* Many DevOps pipelines use third-party components such as libraries or APIs, which can introduce vulnerabilities into the pipeline. Attackers can leverage these vulnerabilities to gain unauthorized access to the pipeline.

4. *Inadequate Testing:* Inadequate testing of the pipeline can leave it vulnerable to attacks. Testing should be done at all stages of the pipeline to ensure that there are no vulnerabilities.

Examples of attacks on DevOps pipelines

There have been several high-profile attacks on DevOps pipelines in recent years. Some of the notable attacks include:

1. *Codecov Attack:* In April 2021, a supply chain attack was carried out on the Codecov platform, which provides code coverage tools for DevOps teams. The attack exposed sensitive information, including credentials, API keys, and tokens.

2. *SolarWinds Attack:* In December 2020, involved a supply chain breach targeted at SolarWinds, a renowned provider of IT management software. In this incident, malicious actors successfully implanted harmful code within the software, enabling unauthorized access to the networks of numerous prominent organizations.

3. *Tesla Cloud Attack:* In 2018, attackers gained access to Tesla's Amazon Web Services (AWS) cloud account and used it to mine cryptocurrency. The attack was caused by an unprotected Kubernetes console in the Tesla cloud infrastructure.

Impact of a security breach on the DevOps pipeline

A security incident occurring within a DevOps pipeline can lead to significant ramifications for an organization. The effects of such an incident can be broadly classified into three key areas:

1. *Financial Implications*: A security breach carries the potential for substantial financial losses. These losses may arise from the expenses involved in rectifying the breach and addressing any resulting damages. Additionally, the organization may experience a decline in business opportunities, leading to further financial setbacks.
2. *Reputational Harm*: A security breach has the capability to tarnish an organization's reputation, undermining the trust placed in it by customers and stakeholders. This erosion of trust can have long-lasting effects on the organization's brand image and future prospects.
3. *Legal Ramifications*: A security breach may also result in legal consequences, encompassing the imposition of fines, litigation, and investigations by regulatory bodies. These legal proceedings can amplify the financial impact and further damage the organization's standing.

In conclusion, understanding the risks and threats faced by DevOps pipelines is essential for organizations looking to secure their pipelines. The examples of attacks on DevOps pipelines and their impact highlight the importance of implementing effective security measures in DevOps pipelines. The next section will discuss the best practices for securing DevOps pipelines.

III. BEST PRACTICES FOR SECURING DEVOPS PIPELINES

A. *Overview of the essentials secrity practices for DevOps pipelines*

Securing DevOps pipelines requires a comprehensive approach that involves securing all the components of the pipeline, including software, configurations, infrastructure, deployment, and testing. Some of the essential security practices for DevOps pipelines are:

1. Secure software development lifecycle (SDLC) practices

2. Secure configuration management practices

3. Secure infrastructure management practices

4. Secure deployment and release management practices

5. Secure testing and quality assurance practices

In this section, we will discuss each of these practices in detail.

B. *Secure software development lifecycle (SDLC) practices*

Securing the software development lifecycle (SDLC) is a critical aspect of securing DevOps pipelines. Some of the essential SDLC practices for securing DevOps pipelines are:

Code Review: Code review is the process of reviewing the code for potential security vulnerabilities before it is integrated into the pipeline.

Static Application Security Testing (SAST): SAST is a type of testing that checks the source code for potential security vulnerabilities.

Dynamic Application Security Testing (DAST): DAST is a type of testing that checks the running application for potential security vulnerabilities.

Open-Source Software (OSS) Management: OSS components should be managed properly, and their versions should be tracked to ensure that any known vulnerabilities are patched.

C. *Secure configuration management practices*

Configuration management involves managing the configuration of the pipeline components, including servers, databases, and other infrastructure. Some of the essential configuration management practices for securing DevOps pipelines are:

1. Configuration Auditing: Regular audits of the configuration of the pipeline components can help identify any misconfigurations or vulnerabilities.

2. Configuration Backups: Backups of the configuration should be taken regularly to ensure that the pipeline can be restored in the event of a failure.

3. Least Privilege: The principle of least privilege should be applied to the configuration of the pipeline components to ensure that only authorized personnel have access to them.

D. *Secure infrastructure management practices*

The Infrastructure management involves managing the physical and virtual infrastructure that supports the pipeline. Some of the essential infrastructure management practices for securing DevOps pipelines are:

1. Identity and Access Management (IAM): To maintain secure infrastructure access, it is crucial to enforce Identity and Access Management (IAM) policies. These policies play a significant role in permitting only authorized personnel to have the appropriate level of access.
2. Network Security: Network security measures such as firewalls, intrusion detection systems, and network segmentation should be implemented to protect the infrastructure.
3. Monitoring and Logging: By implementing monitoring and logging, organizations can effectively detect and respond to suspicious incidents, ensuring the integrity and security of their systems.

E. *Secure deployment and release management practices*

Deployment and release management involve deploying and releasing new code into the pipeline. Some of the essential deployment and release management practices for securing DevOps pipelines are:

1. Continuous Integration and Continuous Deployment (CI/CD): CI/CD practices should be implemented to ensure that new code is continuously integrated and tested before being deployed.
2. Release Management: Release management practices should be implemented to ensure that only authorized personnel can release code into the pipeline.
3. Change Management: Change management practices should be implemented to ensure that any changes to the pipeline are properly documented and tested.

F. *Secure testing and quality assurance practices*

Testing and quality assurance involve testing the pipeline components for potential security vulnerabilities and ensuring that they meet the required quality standards. Some of the essential testing and quality assurance practices for securing DevOps pipelines are:

1. Penetration Testing: Penetration testing should be conducted to identify any potential security vulnerabilities in the pipeline.

2. Security Information and Event Management (SIEM): SIEM tools should be implemented to detect any potential security breaches in the pipeline.

3. Automated Testing: To ensure the adherence to necessary quality standards, it is recommended to employ automated testing tools for evaluating the components of the pipeline.

IV. TOOLS AND TECHNOLOGIES FOR SECURING DEVOPS PIPELINES

As the popularity of DevOps has increased, the range of tools and technologies for enhancing security in DevOps pipelines has also expanded. This section aims to present a comprehensive overview of the available tools and technologies, which can be categorized into four main groups: continuous security testing, infrastructure security, compliance and governance, and security information and event management (SIEM).

A. *Overview of the availabel tools and technologies for securing DevOps pipelines*

Numerous tools and technologies are accessible to enhance the security of DevOps pipelines, each possessing unique advantages and drawbacks. Among the frequently employed options are:

A. Continuous security testing tools: These tools are used to continuously monitor and test the security of applications and infrastructure. Examples include static code analysis tools, dynamic application security testing (DAST) tools, and software composition analysis (SCA) tools.

B. Infrastructure security tools: These tools are used to secure the underlying infrastructure that supports DevOps pipelines. Examples include network security tools, identity and access management (IAM) tools, and vulnerability scanners.

C. Compliance and governance tools: These tools are used to ensure that DevOps pipelines are in compliance with industry standards and regulations. Examples include configuration management tools, audit and logging tools, and policy enforcement tools.

D. Security information and event management (SIEM) tools: The mentioned tools are employed for the purpose of gathering, examining, and connecting security-related information from diverse origins, aiming to identify and address security incidents. Instances of such tools encompass security analytics tools, threat intelligence platforms, and security orchestration and automation tools.

B. Continuous security testing tools

Continuous security testing tools play a vital role in ensuring the security of DevOps pipelines. These tools enable developers to assess the security of their applications and infrastructure at every stage of the software development lifecycle, spanning from development to deployment. Here are some instances of continuous security testing tools:

a. Static code analysis tools: Static code analysis tools thoroughly examine source code to pinpoint security vulnerabilities and coding errors. Renowned examples include Veracode and Checkmarx.

b. Dynamic application security testing (DAST) tools: DAST tools simulate attacks on applications to detect vulnerabilities that may remain undetectable through static analysis. Prominent examples encompass OWASP ZAP and Burp Suite.

c. Software composition analysis (SCA) tools: SCA tools scrutinize the software components and libraries utilized in an application to identify known vulnerabilities and licensing issues. Noteworthy examples encompass WhiteSource and Black Duck.

C. Infrastructure security tools

Infrastructure security tools are used to secure the underlying infrastructure that supports DevOps pipelines. These tools help ensure that the infrastructure is configured securely, that access to resources is properly managed, and that vulnerabilities are detected and remediated in a timely manner. Some examples of infrastructure security tools include:

Network security tools: These tools are used to secure network traffic and prevent unauthorized access. Examples include firewalls, intrusion detection and prevention systems (IDPS), and virtual private networks (VPNs).

Identity and access management (IAM) tools: These tools are employed for the purpose of overseeing user identities and regulating resource access. Notable examples encompass Active Directory, Okta, and Duo Security.

Vulnerability scanners: These tools scan infrastructure and applications for known vulnerabilities and misconfigurations. Examples include Nessus and Qualys.

D. Complaince and governanace tools

Compliance and governance tools are essential for ensuring that DevOps pipelines are meeting regulatory requirements and organizational policies. These tools help in automating compliance checks and in identifying areas where the pipeline may not be compliant. They also help in generating reports for audits and assessments.

a) Chef Compliance: Chef Compliance is a tool that helps in automating compliance checks for infrastructure and applications. It provides out-of-the-box profiles for various regulatory standards such as CIS, HIPAA, PCI DSS, and NIST. Chef Compliance integrates with Chef Automate, which provides a dashboard for viewing compliance status and generating reports.

b) Sonatype Nexus Lifecycle: Sonatype Nexus Lifecycle is a tool that helps in identifying and remediating security vulnerabilities in open-source libraries used in the software development process. It integrates with various DevOps tools such as Jenkins, GitHub, and JIRA. It also provides policy enforcement capabilities for ensuring that only approved components are used in the software development process.

c) IBM OpenPages: IBM OpenPages is a tool that provides a comprehensive governance, risk, and compliance (GRC) solution. It helps in automating compliance assessments and in identifying and mitigating risks. It provides a centralized repository for policies, regulations, and control frameworks. It also provides reporting capabilities for audits and assessments.

E. Security information and event management (SIEM) tools

SIEM tools are used for collecting and analyzing security events from various sources in the DevOps pipeline. These tools help in detecting and responding to security incidents in a timely manner.

1. Splunk Enterprise Security: Splunk Enterprise Security is a robust security information and event management (SIEM) solution designed to gather and evaluate security events from diverse sources, including network devices, servers, and applications. This powerful tool offers a live dashboard that enables real-time monitoring of security events, facilitating the detection of potential security incidents. Furthermore, it incorporates correlation and analysis functionalities, empowering organizations to identify and address complex threats effectively.

2. LogRhythm NextGen SIEM Platform: LogRhythm NextGen SIEM Platform is a tool that provides real-time monitoring and analysis of security events. It provides a centralized console for monitoring security events across the DevOps pipeline. It also provides automated response capabilities for responding to security incidents in a timely manner.

Overall, these tools and technologies can greatly enhance the security posture of DevOps pipelines. However, it's important to choose the right tools and technologies based on the specific requirements and constraints of the organization.

TABLE I. DIFFERENT TYPE OF TOOLS AND TECHNOLOGIES FOR SECURING PIPELINES

Tools/Technology	**Examples of Tools/ Technologies**
Continuous security testing	SonarQube, WhiteSource, Veracode
Infrastructure security	HashiCorp Vault, Chef Automate, Puppet
Compliance and Goverance	Chief Compliance, ReedLock,

	Twistlock
Security information and event management (SIEM)	Splunk ENterprice Security, IBM Qradar, ArcSight

V. CHALLENGES AND LIMITATIONS OF SECURING DEVOPS PIPELINES

A. *Understanding the Complexities and Hurdles in Securing DevOps Pipelines*

Securing DevOps pipelines encompasses a multitude of intricate and demanding aspects, incorporating technological, cultural, and organizational factors. This undertaking poses a range of challenges and limitations that need to be carefully addressed and navigated.

B. *Cultural and organizational barriers to security*

One of the main challenges of securing DevOps pipelines is the cultural and organizational barriers that exist between development and security teams. DevOps is built on the principle of collaboration and communication between different teams, but the security team is often seen as a bottleneck that slows down the development process. This cultural barrier can lead to a lack of understanding and cooperation between teams, which in turn can result in security issues being overlooked or ignored.

To overcome this challenge, organizations need to foster a culture of collaboration and communication between teams. This can be achieved by creating cross-functional teams that include security experts, providing security training for development teams, and encouraging open and transparent communication between teams.

C. *Lack of skilled security personnel*

Another challenge in securing DevOps pipelines is the shortage of skilled security personnel who are familiar with both security and DevOps practices. DevOps requires a unique set of skills and knowledge that is not commonly found in traditional security roles. This shortage of skilled personnel can result in security being treated as an afterthought, rather than an integral part of the development process.

To address this challenge, organizations can invest in training and development programs for their security and DevOps teams. This can include providing security training for developers, cross-training security personnel in DevOps practices, and hiring security personnel with a strong understanding of DevOps.

D. *Integration with existing security systems*

Integrating security into the DevOps pipeline can be challenging due to the need to work with existing security systems and processes. Many organizations have legacy security systems that are not designed to work with DevOps practices, which can create a barrier to integration. Additionally, the integration of new security tools and technologies into the DevOps pipeline can be difficult due to the complexity of the pipeline and the need to ensure that the tools are compatible with existing systems.

To overcome this challenge, organizations need to carefully evaluate their existing security systems and processes and identify areas where they can be integrated with the DevOps pipeline. They can also invest in tools and technologies that are specifically designed to work with DevOps practices, such as container security tools and automated compliance frameworks.

Examples and Case Studies:

1. The Capital One Data Breach of 2019 served as a stark reminder of the security risks associated with misconfigured firewalls in DevOps pipelines. As a result of a misconfiguration in their AWS environment, the breach compromised the personal and financial data of more than 100 million customers. This incident underscores the critical importance of robust configuration management practices, alongside continuous monitoring and testing, to identify and prevent security breaches.
2. In 2017, Equifax experienced a significant data breach, impacting over 143 million customers. The breach was attributed to a vulnerability present in the organization's web application, which allowed unauthorized access to sensitive information. This case emphasizes the vital role of secure software development lifecycle (SDLC) practices, such as thorough threat modeling, secure coding techniques, and regular vulnerability testing, in safeguarding against breaches.
3. The Target Data Breach of 2013 stands as one of the most notable incidents involving the compromise of personal and financial data. The breach affected more than 110 million customers and resulted from a vulnerability in Target's payment system. Exploiting this weakness, the attacker successfully stole credit and debit card information. This scenario highlights the critical need for robust security measures in payment systems, including comprehensive testing, intrusion detection systems, and secure transaction protocols

VI. CASE STUDIES

A. *Examples of successful implementatios of DevOps Pipeline security*

Organizations across different sectors have successfully implemented secure DevOps pipelines to mitigate security risks and streamline software development processes. In this segment, we will explore several case studies that exemplify the advantages gained from the implementation of secure DevOps pipelines.

1. Netflix

Netflix, a prominent streaming platform, has embraced a DevOps culture to efficiently provide its services on a large scale. To ensure the security of its DevOps pipelines, the company employs a diverse array of tools and technologies, encompassing the following:

• Netflix Security Monkey: A tool for monitoring and managing security risks in the cloud environment.

a. Netflix Stethoscope: A security information and event management (SIEM) tool that provides visibility into security events across the organization.

b. Netflix Chaos Monkey: A tool for testing the resilience of the system by randomly shutting down services and instances.

The implementation of secure DevOps pipelines at Netflix has resulted in significant benefits, including faster time-to-market, improved software quality, and enhanced security.

2. Capital One

Capital One, a prominent financial services company, has adopted the DevOps methodology to effectively deliver software solutions on a large scale while prioritizing security and compliance.

The company has implemented a range of security measures to secure its DevOps pipelines, including:

a. Automated security testing: The company uses a range of automated security testing tools to detect vulnerabilities and ensure compliance.
b. Secure configuration management: Capital One has implemented a secure configuration management system that ensures that all software components are up-to-date and secure.
c. Continuous monitoring: The company uses a range of monitoring tools to detect and respond to security incidents in real-time.

The implementation of secure DevOps pipelines has enabled Capital One to deliver software faster and more securely while complying with industry regulations.

B. *Description of the security measures implemented*

The case studies discussed above highlight the importance of implementing a range of security measures to secure DevOps pipelines. These measures include:

a. Secure software development lifecycle (SDLC) practices: Organizations must implement secure SDLC practices to ensure that security is built into the software development process from the start. This includes threat modeling, secure coding, and automated security testing.

a. Secure configuration management practices: Organizations must implement secure configuration management practices to ensure that all software components are up-to-date and secure. This includes implementing configuration management tools, enforcing least privilege access, and implementing secure network segmentation.

b. Secure infrastructure management practices: Organizations must implement secure infrastructure management practices to ensure that their infrastructure is secure and compliant. This includes implementing network security controls, securing data at rest and in transit, and implementing secure access controls.

c. Secure deployment and release management practices: Organizations must implement secure deployment and release management practices to ensure that their software is deployed and released securely. This includes implementing automated deployment pipelines, implementing least privilege access controls, and enforcing secure coding practices.

d. Secure testing and quality assurance practices: Organizations must implement secure testing and quality assurance practices to ensure that their software is tested and validated for security vulnerabilities. This includes implementing automated security testing tools, enforcing secure coding practices, and conducting regular security assessments.

C. Benefits and outcomes of implementing secure DevOps pipelines

Implementing secure DevOps pipelines can provide various benefits and outcomes, including:

a. Improved Security Posture: Implementing secure DevOps pipelines can significantly improve the security posture of an organization. By following secure SDLC practices, organizations can detect and mitigate security vulnerabilities at an early stage in the development cycle.
b. Reduced Security Risks: By implementing a secure DevOps pipeline, the potential for security breaches can be greatly minimized. It is crucial to conduct regular security testing and maintain continuous monitoring of the pipeline to promptly identify and address any security vulnerabilities, mitigating the risk of exploitation.
c. Faster Time to Market: Implementing DevOps pipeline security can help organizations achieve faster time to market for their products and services. By automating various processes, such as testing and deployment, organizations can speed up their software delivery process, resulting in faster product releases.
d. Improved Collaboration: Implementing DevOps pipeline security can also help improve collaboration between different teams, such as developers, testers, and operations personnel. By breaking down silos and promoting cross-functional collaboration, organizations

can improve communication, reduce errors, and achieve better results.

e. Compliance and Regulatory Requirements: Many organizations need to comply with various regulatory requirements, such as GDPR, HIPAA, and PCI-DSS. Implementing secure DevOps pipelines can help organizations meet these requirements by ensuring that security is integrated into the development process.

VII. CONCLUSION

A. *Summary of the paper's key points*

This research paper has explored the topic of securing DevOps pipelines. The paper began by providing an overview of DevOps and its importance in modern software development. The need for securing DevOps pipelines was also discussed, highlighting the various risks and threats that exist in this area. The paper then presented best practices for securing DevOps pipelines, including secure software development lifecycle (SDLC) practices, secure configuration management practices, secure infrastructure management practices, secure deployment and release management practices, and secure testing and quality assurance practices.

The available tools and technologies for securing DevOps pipelines were then discussed, with a particular focus on continuous security testing tools, infrastructure security tools, compliance and governance tools, and security information and event management (SIEM) tools. The challenges and limitations of securing DevOps pipelines were also explored, including cultural and organizational barriers to security, lack of skilled security personnel, and integration with existing security systems.

A. *Discussion of the importance of securing DevOps pipelines*

The importance of securing DevOps pipelines cannot be overstated, as organizations increasingly rely on DevOps to accelerate their software development and deployment processes. However, the speed and agility of DevOps can create security risks, and failing to address these risks can result in serious consequences such as data breaches, financial losses, and reputational damage.

B. *Recommendations for organizations looking ot improve the security of their DevOps pipelines*

Organizations looking to improve the security of their DevOps pipelines should start by implementing the best practices discussed in this paper, including secure software development lifecycle (SDLC) practices, secure configuration management practices, secure infrastructure management practices, secure deployment and release management practices, and secure testing and quality assurance practices. They should also invest in the available tools and technologies for securing DevOps pipelines, including continuous security testing tools, infrastructure security tools, compliance and governance tools, and security information and event management (SIEM) tools.

To overcome the challenges and limitations of securing DevOps pipelines, organizations should focus on building a strong security culture, providing training and resources to their employees, and collaborating with their security and development teams to integrate security into the DevOps pipeline.

C. *Areas for future research*

Future research in this area could focus on developing new tools and technologies specifically designed for securing DevOps pipelines, exploring the impact of emerging technologies such as artificial intelligence and machine learning on DevOps pipeline security, and investigating the effectiveness of different approaches to integrating security into DevOps pipelines. Additionally, further research could be conducted to explore the impact of DevOps pipeline security on organizational performance and competitive advantage.

REFERENCES

1. A. Bassam, "DevOps Security: Protecting Continuous Delivery," IEEE Security & Privacy, vol. 15, no. 6, pp. 44-52, Nov.-Dec. 2017. doi: 10.1109/MSP.2017.3641439
2. A. Khan, M. Ahmed, and J. Zhang, "Towards Securing DevOps Pipeline: A Survey," in 2019 IEEE International Conference on Big Data (Big Data), 2019, pp. 3740-3749. doi: 10.1109/BigData47090.2019.9006464
3. D. Leach and D. Drescher, "DevOpsSec: A Complete Guide to Gaining Visibility and Security in DevOps," Syngress, 2018.
4. P. Mehta, "Securing DevOps," O'Reilly Media, Inc., 2018.
5. S. Kim, "The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations," IT Revolution Press, 2016.
6. S. Parthasarathy and R. Anuradha, "Security issues in DevOps and their remedies: A review," in 2016 International Conference on Circuit, Power and Computing Technologies (ICCPCT), 2016, pp. 1-5. doi: 10.1109/ICCPCT.2016.7530366