Payatu

xyz

1st xxxx 202x

Devsecops Assessment

Client Details

Company Name: xyz

Contact Person: n/a

Address: N/A

Email:

Telephone:

Document History

Version	Date	Author	Remark
1.0		Payatu	Sample report

Table of Contents

1. About Payatu

Payatu is a research-focused security testing service organization specialized in IoT and embedded products, web, mobile, cloud and infrastructure security assessments and in-depth technical security training. Our state-of-the-art research, methodologies, and tools ensure the safety of our client's assets.

At Payatu, we believe in following one's passion, and with that thought, we have created a world-class team of researchers and executors who are bending the rules to provide the best security services. We are a passionate bunch of folks working on the latest and leading-edge security technology.

We are proud to be part of a vibrant security community and don't miss any opportunity to give back. Some of the contributions in the following fields reflect our dedication and passion

nullcon - nullcon security conference is an annual security event held in Goa, India. After years of efforts put in the event, it has now become a world-renowned platform to showcase the latest and undisclosed research.
hardwear.io - Hardware security conference is an annual hardware security event held in The Hague, Netherlands. It is being organized to answer emerging threats and attacks on hardware. We aim to make it the largest platform where hardware security innovation happens.
Dedicated fuzzing infrastructure - We are proud to be one of the few security research companies to own an in-house infrastructure and hardware for distributed fuzzing of software such as browsers, client and server applications.
null - It all started with null - The open security community. It's a registered non-profit society and one of the most active security community. null is driven totally by passionate volunteers.
Open source - Our team regularly authors open source tools to aid in security learning and research.
Talks and Training: Our team delivers talks/highly technical training in various international security and hacking conference, i.e., DEFCON Las Vegas, BlackHat Las Vegas, HITB Amsterdam, Consecwest Vancouver, nullcon Goa, HackinParis Paris, Brucon Belgium, zer0con Seoul, PoC Seoul to name few.

We are catering to a diverse portfolio of clients across the world, who are leaders in banking, finance, technology, healthcare, manufacturing, media houses, information security, and education, including government agencies. Having various empanelment and accreditations, along with a strong word of mouth has helped us win new customers, and our thorough professionalism and quality of work, have brought repeat business from our existing clients.

We thank you for considering our security services and requesting a proposal. We look forward to extending the expertise of our passionate, world-class professionals to achieve your security objectives.

2. Project Details

2.1 Executive Summary

The Payatu security team performs assessments on current state devsecops model . These assessments aim is to uncover any security issues in the assessed process and CI/CD pipeline / Infrastructure, explain the impact and risks associated with the found issues, and provide guidance in the prioritization and remediation steps. Devsecops Assessment of XYZ has been performed, considering devsecops best practices

2.2 Scope and Objective

The scope of this assessment was limited to Devsecops process and CI/CD Pipeline and infrastructure created by XYZ.

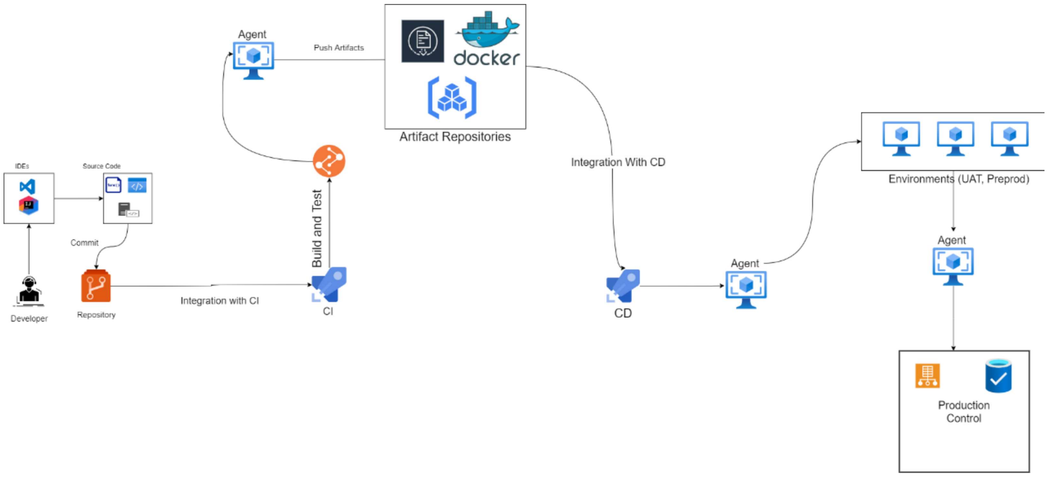
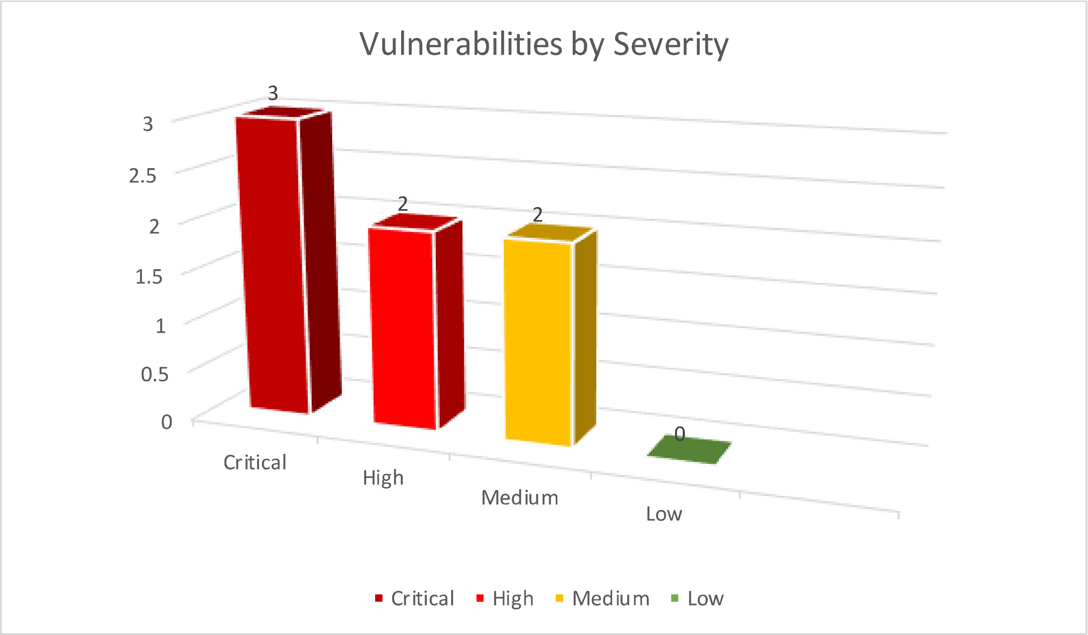


Figure 1: xyz Devops Pipeline

2.3 Vulnerability Chart

The discovered vulnerabilities table and chart illustrated below, provides a snapshot view of the number and severity of issues discovered during this devsecops assessment.

CRITICAL	This issue can impact the application severely and should be addressed immediately. Attackers can gain root or super user access or severely impact system operation.
HIGH	This issue can cause a problem like unprivileged access and should be addressed as soon as possible.
MEDIUM	This issue may pose a significant threat over a longer period of time.
LOW	This issue is more likely an information disclosure and may be an acceptable threat.



2.4 Table of Findings

S. No.	Policy/Parameter	Current	Status
1	Third-party components pulled from public repositories	Critical	
2	No mechanism to detect leaked secrets on GitHub	Critical	
3	No integrated scanning for Docker images	High	
4	Self-managed Jenkins server is exposed to the internet	High	
5	Improper controls on deployment pipelines	Medium	
6	GitHub signed commits are not enabled /enforced	Medium	
7	No GitHub Personal access token rotation policy	Medium	

2.5 Technical Findings

2.5.1 PY-XYZ-001: Third-party components pulled from public repositories

Potential Impact: **CRITICAL**

Description:

During the assessment of Jenkins Pipeline of XYZ Company, we found that third party components are being pulled from public repositories and no SCA scanners are deployed to scan third party components

Business Risk: Due to this, malicious third-party components could get placed into the codebase

Technical Risk: Upon successful execution of putting malicious component, an attacker can:

- Get control of the logic flow of the application and leak user credentials and other sensitive information
- Can get command execution on the server-side of the application.

Remediation:

- Use third party components from valid public repositories
- Deploy SCA scanners on pipeline
- Introduce integrity checks while using public libraries

Steps to Reproduce: NA

2.5.2 PY-XYZ-002: No mechanism to detect leaked secrets in GitHub

Potential Impact: CRITICAL

Description:

During the assessment of the CI/CD pipeline of XYZ Company, we did not find any scanners or monitoring systems to check leaked secrets on the company's GitHub repository.

Business Risk: Due to this misconfiguration, a developer can commit sensitive API keys or credentials into the GitHub repository, which will be publicly visible.

Technical Risk: Attacker can make use of leaked credentials to take control of company assets such as Jenkins server, Database Servers, etc.

Remediation:

•Deploy Scanners such as Trufflehog to scan git repositories**.**

- Do not use hardcoded sensitive information in the source code.

Steps to Reproduce: NA

2.5.3 PY-XYZ-003: No integrated scanning for Docker images

Potential Impact: **High**

Description:

During the assessment of the CI pipeline, we found that security checks are not implemented for Docker scanning

Business Risk:

Without Docker image security scanning in place, insecure Docker images may get used in infrastructure. Sensitive information or vulnerable third-party components can reside inside the container.

Technical Risk: Attackers can exploit vulnerable Docker images to steal sensitive information present inside the container or take control of services that Docker images are running and eventually take control of software.

Remediation:

- Deploy container security checks before pushing images into the registry

Steps to Reproduce: NA

2.5.4 PY-XYZ-004: Self-managed Jenkins server is exposed to the internet

Potential Impact: **High**

Description:

During Assessment, we found the Jenkins server publicly available with no authentication mechanism.

Business Risk:

A compromised Jenkins system can threaten the entire production environment and be an entry point for inserting malicious software and exfiltration of sensitive data.

Technical Risk: With access to the Jenkins server, an attacker can

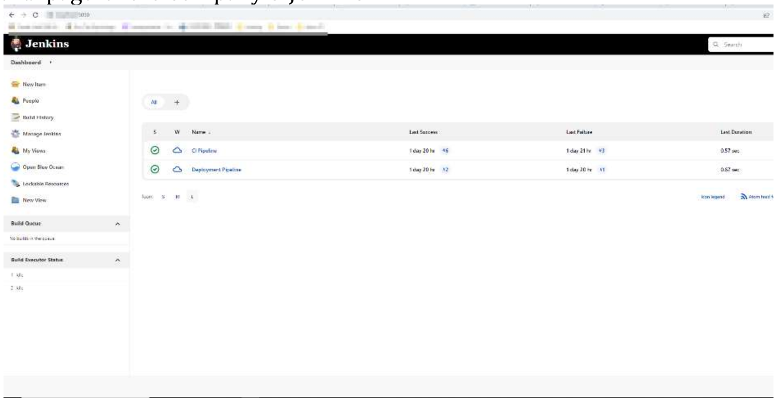
- Get control of CI/CD pipelines, Steal user information present on that server
- Get command execution on the company environment

Remediation:

- If possible, restrict public access to the Jenkins server
- Use an Authentication mechanism and strong password policies to secure the Jenkins server

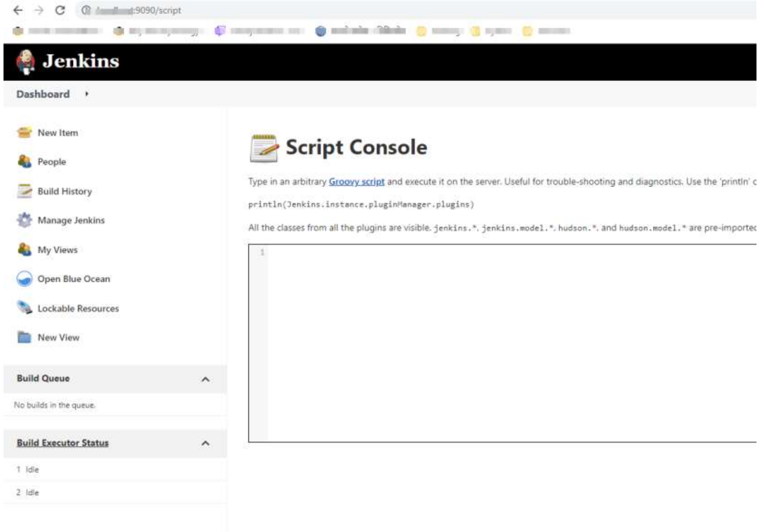
Steps to Reproduce:

- Navigate to the company's website http://jenkins.xyz.com:9090. It will display the dashboard page of the company's Jenkins.



S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☁	CI Pipeline	1 day 20 hr #6	1 day 21 hr #3	0.57 sec
✓	☁	Deployment Pipeline	1 day 20 hr #2	1 day 20 hr #1	0.67 sec

- Append the "/script" endpoint after the URL and check if a console is accessible.



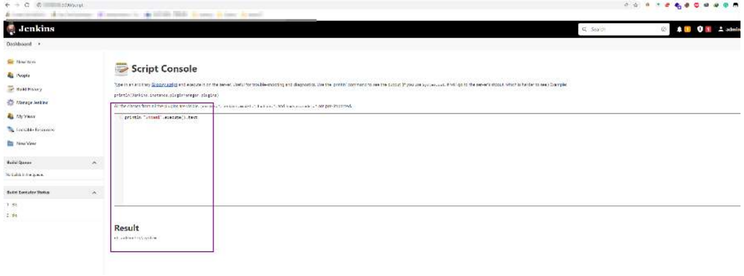
Script Console

Type in an arbitrary Groovy script and execute it on the server. Useful for trouble-shooting and diagnostics. Use the 'println' c

```
println(Jenkins.instance.pluginManager.plugins)
```

All the classes from all the plugins are visible. jenkins.*, jenkins.model.*, hudson.*, and hudson.model.* are pre-importec

- In the text box, type groovy command ***<u>println "whoami".execute().text</u>*** to check if code execution is possible or not.



Script Console

```
println "whoami".execute().text
```

Result

- As we can see above, code execution is successful.

2.5.5 PY-XYZ-005: Improper controls on deployment pipelines

Potential Impact: **Medium**

Description: During the assessment, we found that there is an auto-merge rule enabled on the ABC repository.

Business Risk: Due to the auto-merge rule, malicious code can get committed into the production systems.

Technical Risk: Upon successful exploitation, an attacker can:

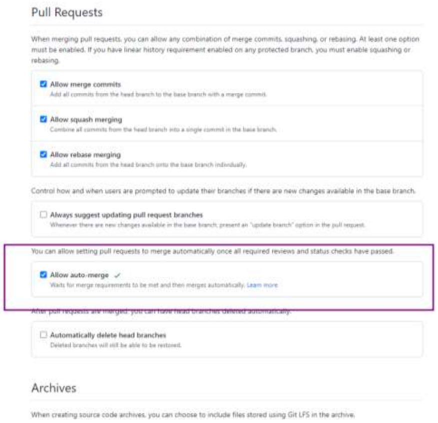
- Deploy malware/backdoor in the production environment and change the logical flow of the application
- Can get Command execution on production environment.

Remediation:

- Limit the usage of auto-merge rules and ensure that wherever they are in use – they are applicable to a minimal amount of contexts.
- Review the code of all auto-merge rules thoroughly to ensure they cannot be bypassed and avoid importing third-party code in the auto-merge process.

Steps to Reproduce:

- Check the company's ABC project repository. Click on the settings section of the particular project repository.



Pull Requests

When merging pull requests, you can allow any combination of merge commits, squashing, or rebasing. At least one option must be enabled. If you have linear history requirement enabled on any protected branch, you must enable squashing or rebasing.

☑ Allow merge commits
Add all commits from the head branch to the base branch with a merge commit.

☑ Allow squash merging
Combine all commits from the head branch into a single commit in the base branch.

☑ Allow rebase merging
Add all commits from the head branch onto the base branch individually.

Control how and when users are prompted to update their branches if there are new changes available in the base branch.

☐ Always suggest updating pull request branches
Whenever there are new changes available in the base branch, present an "update branch" option in the pull request.

You can allow setting pull requests to merge automatically once all required reviews and status checks have passed.

☑ Allow auto-merge ✓
Waits for merge requirements to be met and then merges automatically. Learn more

After pull requests are merged, you can have head branches deleted automatically.

☐ Automatically delete head branches
Deleted branches will still be able to be restored.

Archives

When creating source code archives, you can choose to include files stored using Git LFS in the archive.

- As we can see above image, the auto-merge rule is enabled, which is one of the improper deployment controls.

2.5.6 PY-XYZ-006: GitHub signed commits are not enabled /enforced

Potential Impact: **Medium**

Description:

During the XYZ GitHub repository assessment, we noticed that the commits of the ABC repository are not signed.

Business Risk: Due to this misconfiguration, any developer can commit using the name of another developer.

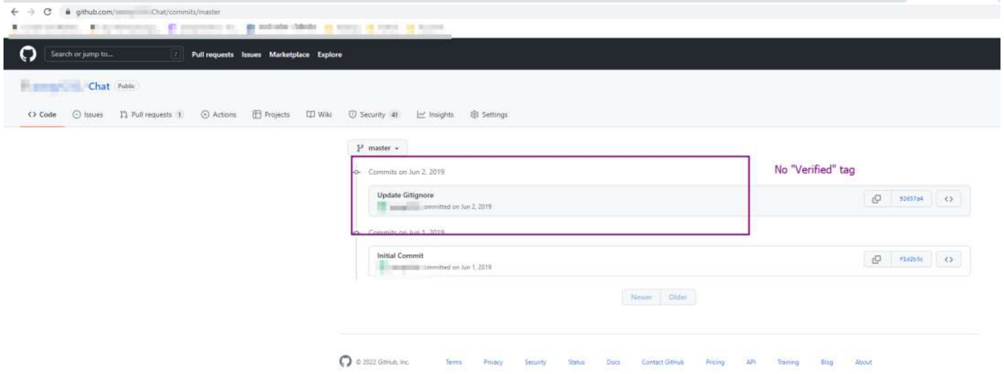
Technical Risk: Due to this misconfiguration, an attacker can commit malicious code on behalf of another developer

Remediation:

- Enable GitHub signed commits

Steps to Reproduce:

- Navigate to Company's GitHub Repository and click on any project commits. You will see no commits are being verified.



github.com/[redacted]Chat/commits/master

[redacted] Chat Public

Commits on Jun 2, 2019

Update Gitignore
[redacted] committed on Jun 2, 2019 — 92657a4

Commits on Jun 1, 2019

Initial Commit
[redacted] committed on Jun 1, 2019 — f1d2b5c

No "Verified" tag

2.5.7 PY-XYZ-007: Third-party components pulled from public repositories

Potential Impact: **Medium**

Description: During the assessment, we found that personal tokens are getting used in CI/CD pipelines and there is no Personal access token rotation policy introduced.

Business Risk: If personal GitHub tokens are leaked outside, Application source code belonging to the company can be leaked

Technical Risk: An attacker can get access to the company's GitHub repository and he/she can commit, download or delete any source code belonging to the company.

Remediation:

- Create multiple tokens with expiration time and limited access to usage of the repository.

Steps to Reproduce: NA

- Get control of the logic flow of the application and leak user credentials and other sensitive information
- Can get command execution on the server-side of the application.

Remediation:

- Use third party components from valid public repositories
- Deploy SCA scanners on pipeline
- Introduce integrity checks while using public libraries

Steps to Reproduce: NA

3. We Prescribe

The below table provides an at-a-glance view of the remediation solution and the best practices that should be taken into consideration to improve the security posture of the scoped-in application:

Vuln ID	Recommendation & Best Practices
1	• Apply third-party governance controls to repository management platforms (GitHub, GitLab, Bitbucket, etc.) and use binary vulnerability scanning for all third-party components before building them into the software product. • Enable workload to run time protection initially in an alert mode, and eventually in a prevent mode.
2	• Enable secret scanning on all your repositories and implement pre-commit hooks for secrets.
3	• DevSecOps processes involving containers must include image scanning at every stage of the CI/CD pipeline.

4	• If possible restrict public access to the Jenkins server • Use an Authentication mechanism and strong password policies to secure the Jenkins server
5	• . Limit the usage of auto-merge rules and ensure that wherever they are in use – they are applicable to a minimal amount of contexts. • Review the code of all auto-merge rules thoroughly to ensure they cannot be bypassed and avoid importing third-party code in the auto-merge process.
6	• Enable commit signing on GitHub. Repository administrators can enforce required commit signing on a branch to block all commits that are not signed and verified.
7	• Implement Access key rotation policy and automation