CWL
CyberWarFare Labs

Breaking Jenkins to

BUILD IT BETTER

A Beginner's Guide to Jenkins Security Fundamentals

About CyberWarFare Labs :

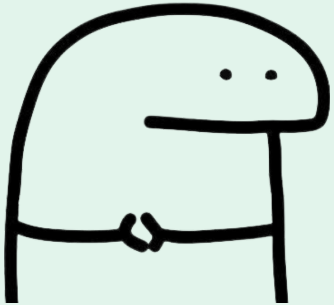
CW Labs is a renowned Infosec company specializing in cybersecurity practical learning. They provide on-demand educational services. The company has 3 primary divisions :

1. **Learning Management System (LMS) Platform**
2. **CWL CyberSecurity Playground (CCSP) Platform**
3. **Infinity Learning Platform**

About Me

> Security Intern at CW Labs

> Part time hacker, full time disaster!



What we will cover in this session

1. Introduction to Jenkins
2. Common Jenkins Vulns & Flaws
3. Demos
4. Jenkins Security Best Practices
5. Checklist for common Jenkins security flaws
6. Conclusion and QNA

1

Introduction to Jenkins

What is Jenkins?

- Java based program
- Used in software development to automate task
- Automates building, testing & deploying software
- Also helps automate CI/CD pipelines

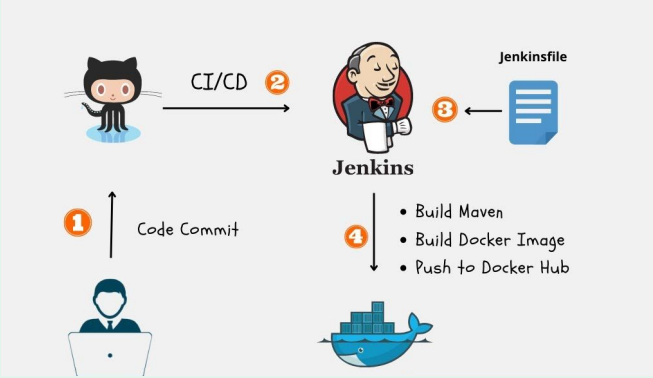


Image by Java Techie

Why Jenkins?

- Jenkins is not as popular as before, but it’s still alive.
- Legacy systems continue to use Jenkins.
- Jenkins is everywhere, and "everywhere" means "everywhere attackers look!"
- Easy to learn and gets the job done. Also, who has time for security anyway? (It didn’t end well.)

Why Should You Care About Jenkins Security?

- Your codebase
- Deployment credentials for AWS, Kubernetes etc
- Secrets like API keys and passwords
- The power to deploy (or destroy) production environments
- Many possible misconfigurations leave Jenkins servers vulnerable.

2

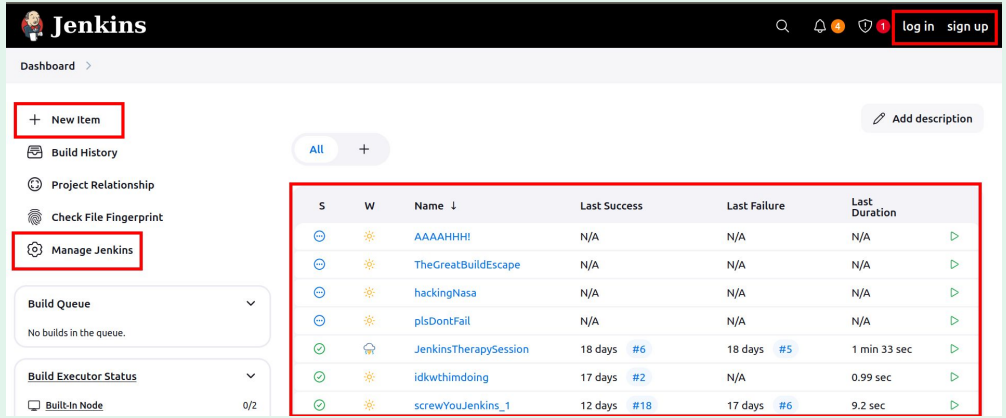
Common Jenkins Vulnerabilities and Security Flaws

1. The “Open For All” Jenkins Dashboard

- Sometimes Jenkins instances are left exposed with no authentication.

- Attackers could skip the login screen completely and walk right in and cause chaos

- Developers might think that the Jenkins server is safe behind a private network.
- An attacker could modify and mess around with the builds if left open without any authentication



2. Plugins: The Wolf in Sheep's Clothing

- Outdated or vulnerable plugins leave your system open to attacks.
- Installing plugins from untrusted sources jeopardizes your system's security.
- Avoid or use plugins like "**Script Security**" with extreme caution.

3. Secrets in Plain Sight

- Credentials in plaintext in a pipeline script is a security flaw.
- Credentials should not be exposed in the build logs.
- Usernames, passwords, and API tokens, should be managed using Jenkins' credential storage feature.

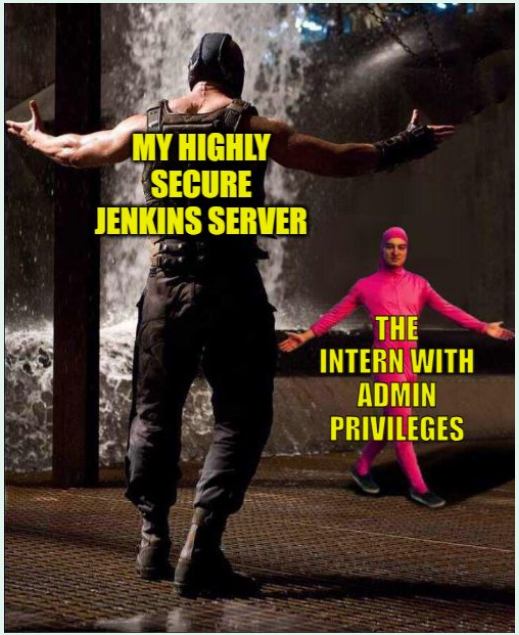
- Take a look at this demo pipeline script. It's a great example of how not to pass credentials into a job!

```
pipeline {
    agent any
    stages {
        stage('Login to Jenkins UI') {
            steps {
                script {
                    sh '''
                    DEMO_URL="http://localhost:8080"
                    USER="enoch"
                    PASSWORD="testing@enoch"

                    RESPONSE=$(curl -s -u "$USER:$PASSWOR
                    echo "Login Response: $RESPONSE"
                    '''
                }
```

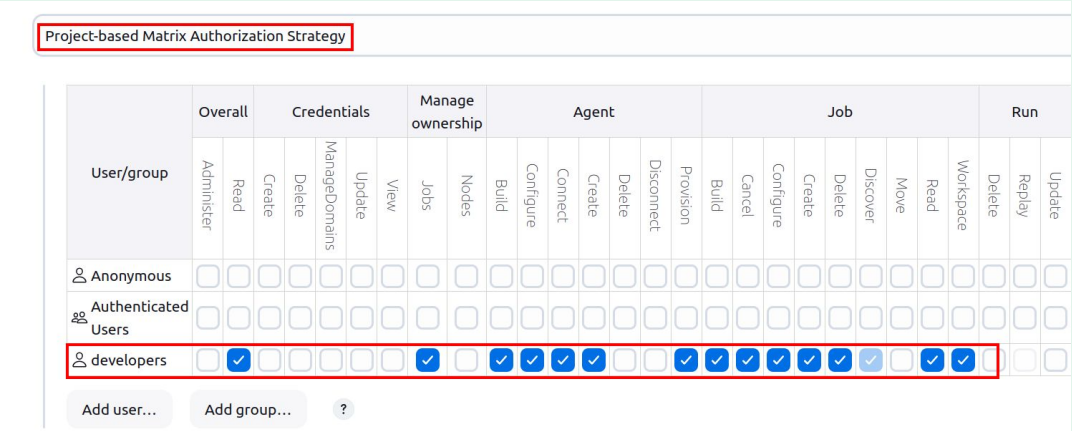
4. The Over Privileged “Admin”

- Assign appropriate permissions to users in Jenkins.
- Follow the principle of least privilege
- Give users only necessary permissions.



4. The Over Privileged “Admin”

- Use Matrix-based Security for flexible user access control.
- Implement Project-based Matrix Authorization Strategy for project-specific permissions.

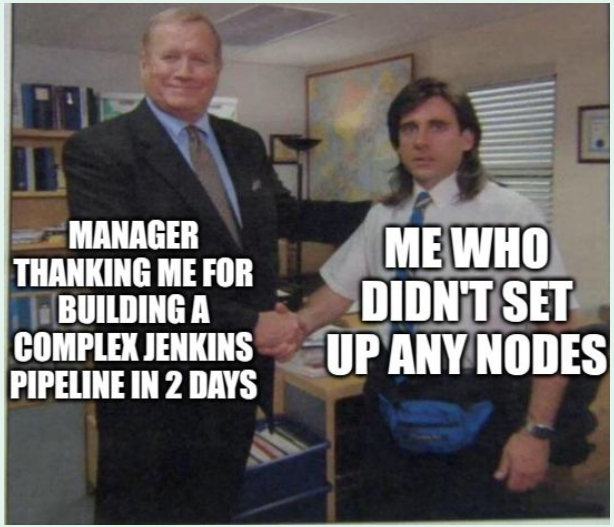


5. Allowing builds to run on the Built-In node

- Running builds on the "master" node (a.k.a. the Built-In Node) puts your systems at risks.
- Opens up more doors for threat actors
- Malicious actions can be performed on the system running jenkins.

5. Allowing builds to run on the Built-In node

- Attackers could execute commands, steal credentials, or access sensitive files.
- Could allow attackers to leave a backdoor into the system.



3

Demos

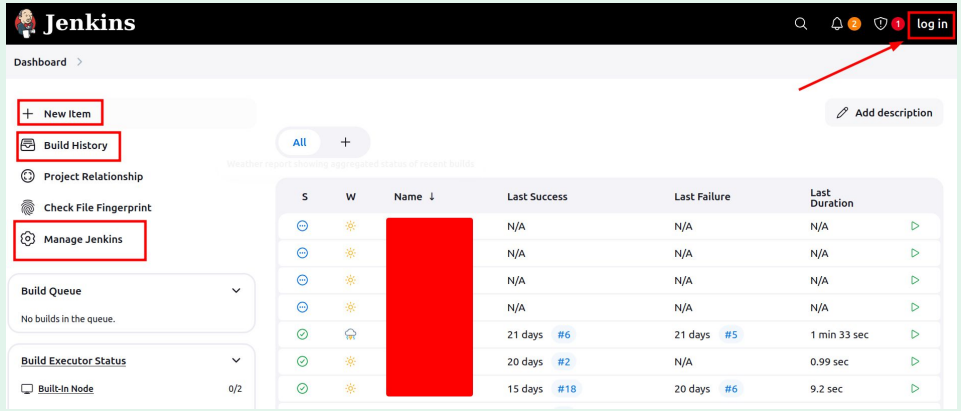
Demo 1: The Unauthenticated Dashboard

Demo 2: Groovy Script Injection

Demo 3: Exploiting Open Sign-Up and Unlimited Access

Demo 1: The Unauthenticated Dashboard

- Anyone on the network can control Jenkins without restrictions.
- Attackers can find Jenkins using network scanning tools like Nmap.
- Unauthorized users can execute arbitrary scripts and commands.



Demo 2: Groovy Script Injection

- The Script Console allows admins to run Groovy code, typically in the **`/script`** directory.
- Security mechanisms are usually in place to protect it from unauthorized or unauthenticated users.

Demo 2: Groovy Script Injection

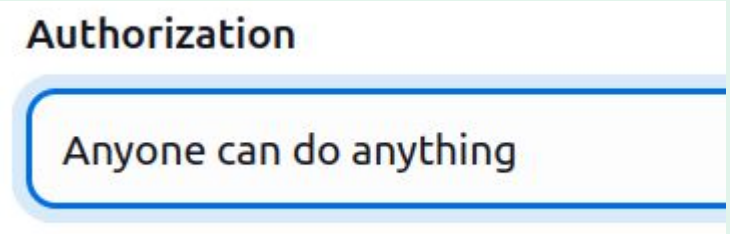
- Without security, unauthorized users could access the Script Console.
- Malicious users could exploit the console to run harmful Groovy code.

Result


```
<?xml version='1.1' encoding='UTF-8'?>
<com.cloudbees.plugins.credentials.SystemCredentialsProvider plugin="credentials@1408.va_622a_b_f5b_1b_1">
  <domainCredentialsMap class="hudson.util.CopyOnWriteMap$Hash">
    <entry>
      <com.cloudbees.plugins.credentials.domains.Domain>
        <specifications/>
      </com.cloudbees.plugins.credentials.domains.Domain>
      <java.util.concurrent.CopyOnWriteArrayList>
        <org.jenkinsci.plugins.plaincredentials.impl.StringCredentialsImpl plugin="plain-credentials@183.va_de8f1dd5a_2b_">
          <scope>GLOBAL</scope>
          <id>dockerhubpwdcwl</id>
          <description></description>
          <secret>[redacted]}</secret>
        </org.jenkinsci.plugins.plaincredentials.impl.StringCredentialsImpl>
        <com.cloudbees.plugins.credentials.impl.UsernamePasswordCredentialsImpl>
          <scope>GLOBAL</scope>
          <id>gitPAT</id>
          <description></description>
          <username>enoch-cwl</username>
          <password>[redacted]</password>
          <usernameSecret>false</usernameSecret>
        </com.cloudbees.plugins.credentials.impl.UsernamePasswordCredentialsImpl>
        <org.jenkinsci.plugins.plaincredentials.impl.StringCredentialsImpl plugin="plain-credentials@183.va_de8f1dd5a_2b_">
          <scope>GLOBAL</scope>
```

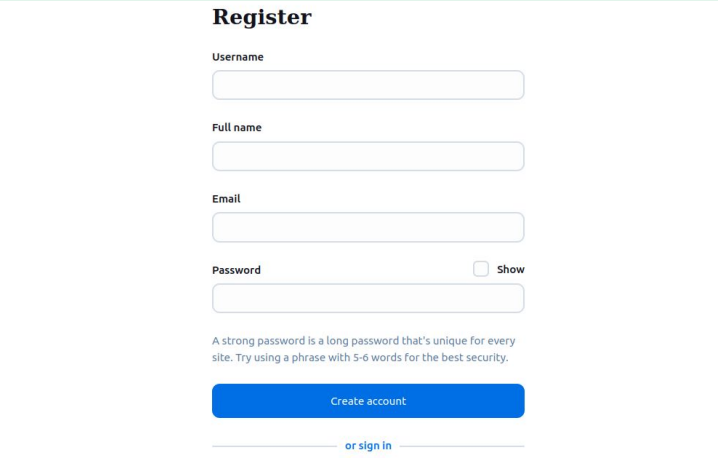
Demo 3: Exploiting Open Sign-Up and Unlimited Access

- The user sign-up option should never be enabled.
- Its disadvantages are more the advantages.



Demo 3: Exploiting Open Sign-Up and Unlimited Access

- Users are prompted to sign up once enabled.
- The system allows authenticated users full access.
- This can let attackers sign up and act maliciously.



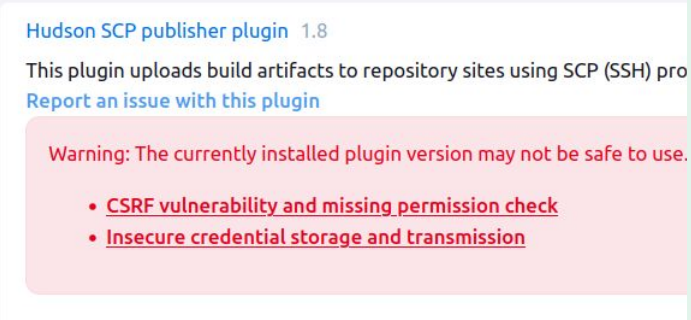
Jenkins Security Best Practices

Secure the Dashboard

- Never expose Jenkins over HTTP. Implement a reverse proxy (eg: Nginx, Traefik).
- Keep Jenkins in a private subnet, not letting the public access it.
- Use a strong password and implement additional verification strategies like MFA

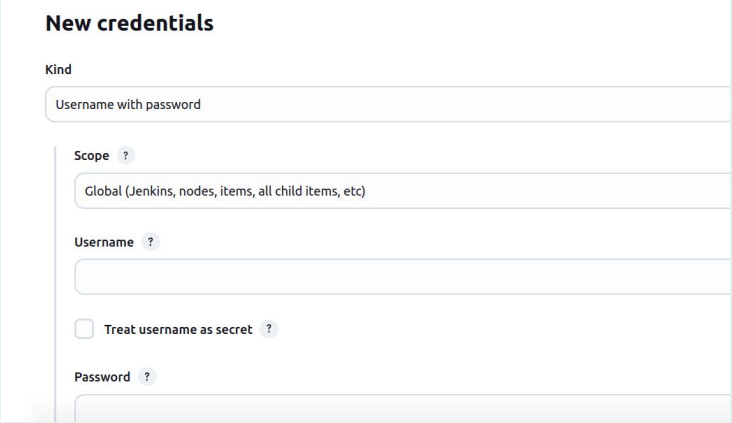
Plugins: Less Is More

- Avoid experimenting with plugins in production and delete any unused plugins.
- Update plugins religiously. Jenkins plugin manager is your friend.



Utilize Secret Management

- Use Jenkins' Credentials Binding Plugin to store secrets.
- Never print secrets in logs
- Make sure that sensitive information is not displayed in the output console.
- Consider adding set +x in scripts to disable command echoing.




```
stage('Docker Push') {
      agent any
      steps {
        withCredentials([usernamePassword(credentialsId: 'dockerHub', passwordV
          sh "docker login -u ${env.dockerHubUser} -p ${env.dockerHubPassword}"
          sh 'docker push shanem/spring-petclinic:latest'
        }
      }
    }
  }
}
```

Role-Based Access Control

- Create roles (e.g., “developer”, “tester”) with the Role-Based Strategy Plugin.
- Follow the principle of least privilege.

User/group	Credentials					Manage ownership	Job								Run			SCM
	Create	Delete	ManageDomains	Update	View	Jobs	Build	Cancel	Configure	Delete	Discover	Move	Read	Workspace	Delete	Replay	Update	Tag
Anonymous	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Authenticated Users	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
developer	☐	☐	☐	☐	☐	☐	☑	☑	☑	☑	☐	☐	☑	☐	☐	☐	☐	☐

Add user... Add group... ?

Audit 👏 Audit 👏 Audit 👏 Everything

- Check Manage `Jenkins > log` for suspicious activities.
- Plugins like "**Probely Security Scanner**" can help you perform vulnerability scans.
- Always keep your Jenkins server updated

5

Checklist for Common Jenkins Security Flaws

- ❑ Is Jenkins exposed to the internet? Use Shodan to check if your Jenkins server is exposed.
- ❑ Is the Jenkins UI left without any authentication
- ❑ Are the plugins up to date? Manage Jenkins > Plugin Manager tells you all.
- ❑ Can non admin users create jobs or run scripts?

- ❑ Are secrets (e.g., AWS keys) exposed in logs/Jenkinsfiles?
- ❑ Are sensitive data like usernames and passwords in plaintext in pipeline scripts?
- ❑ Are you logging too much?
- ❑ Is the **/script** directory accessible without authentication in Jenkins?
- ❑ Are you still using 'admin/admin' for authentication?

Here are some resources for future learning:

- Official documentation on Securing Jenkins
- Hacking Jenkins! ~ By Orange Tsai
- Continuous Intrusion: Why CI Tools Are An Attackers Best Friend

5

Conclusion and QNA

Thank you for making it to the end! 👏👏

If you have any questions, now's your chance to ask

Feel free to connect with me on Linkedin

https://www.linkedin.com/in/enoch-benjamin-4835191a9/