**Lab 6: Use version control systems command to clone, commit, push, fetch, pull, checkout, reset, and delete repositories**

**Prerequisite:**

**AWS EC2 Account**

**GitHub Account**

**Procedure:**

**Step 1: Install Git on EC2**

**sudo yum update -y**

**sudo yum install git -y**

**git –version**

**Step 2: Configure Git**

**Set your name and email (for commit tracking):**

**git config --global user.name "Your Name"**

**git config --global user.email** [**your-email@example.com**](mailto:your-email@example.com)

**git config –list**

**Step 3: Create or Clone a Repository**

**-Can either create a new repository or clone an existing one**

**Option 1: Create a New Repository in AWS (Yellow color can be replaced as per user)**

**mkdir your\_repository\_name**

**cd your\_repository\_name**

**Initialize Git :**

**git init**

**Create a test file:**

**echo "Hello, Git!" > README.md**

**Add and commit:**

**git add .**

**git commit -m "Initial commit"**

**Option 2: Clone an Existing Repository**

**git clone** [**https://github.com/vsdevipriya/Datasets.git**](https://github.com/vsdevipriya/Datasets.git)

**(Datasets is an existing repository in your git)**

**git pull origin main**

**Troubleshooting**

**Authentication Failed? Use a GitHub Personal Access Token (PAT) instead of a password.**

**Log in to your GitHub account.**

**Click on your profile icon (top-right corner) → Select Settings.**

**Scroll down and click on "Developer settings" (left sidebar).**

**Click "Personal access tokens" → Then select "Tokens (classic)".**

**Click "Generate new token" → "Generate new token (classic)".**

**Select Scopes (Permissions): Check the boxes based on what you need:**

* **✅ repo → Full control of private repositories (for pushing code).**
* **✅ workflow → If working with GitHub Actions.**
* **✅ read:packages, write:packages → If using GitHub Packages.**
* **✅ admin:repo\_hook → If working with webhooks.**

**Click "Generate Token".**

**Copy & Save the Token**

**Step4:** **Push Your Local Code**

Create a **repository of your own in Github (here aws-git) and** update your Git remote URL in AWS:

git remote set-url origin <https://github.com/vsdevipriya/aws-git.git>

git remote -v

git add .

git commit -m "Initial commit"

git push -u origin main

**TroubleShoot:**

**If no origin :then add origin**

**git remote add origin https://github.com/vsdevipriya/aws-git.git**

**git status**

**git branch …**

**rename branch using… git branch -M main**

**Questions:**

**How Version Control Can Provide Information About Attacks**

Version control systems (VCS) like Git can help detect, analyze, and respond to security attacks in cloud-based environments. Here’s how they contribute to cybersecurity:

Detect Unauthorized Changes (Insider Threats & Supply Chain Attacks)-Git logs provide a detailed history of all changes made to the repository.If an attacker modifies critical files, the changes are recorded with who made the changes and when.Code reviews and commit signatures help verify if the changes were made by trusted contributors.

**Sample commands: git log –patch**

**git diff HEAD~1 //Check the latest changes**

**git blame filename // which user last modified each line**

**git log -p | grep -i "password\|api\_key" //** **Find exposed secrets in commit history**

**git checkout – filename //** **If files are encrypted or deleted, restore them using this**

**or**

**git checkout HEAD -- .**

**Further Questions:**

**How can Git help detect unauthorized changes in a cloud-based repository?**

**What Git commands can be used to track and analyze security breaches in a repository?**

**How does version control contribute to preventing credential leaks in cloud environments?**

**What is the role of Git rollback features in mitigating ransomware or malicious attacks?**

**How can CI/CD integration with version control enhance security in cloud deployments?**