JFTD-102 JFrog Platform Essentials

Hands-on Labs

Lab 1: Artifactory

1. Create three repositories, one each of Local, Remote, and Virtual on the JFrog UI. The virtual repository will enclose local and remote repositories. All are Docker repositories.
    a. Hint: Participants JPDs are preconfigured with credentials
    b. Observation: Participant is able to login & perform operation
    c. Conclusion: Local users are provisioned
2. Create an Access Token on the JFrog UI
    a. Hint: Participants JPDs are preconfigured with credentials
    b. Observation: Participant performs operation
    c. Conclusion: Access Tokens are created and participant understands RBAC
3. Download JFrog CLI, configure to connect to the JPD, and do a “ping”
    a. Hint: Download from web directly, also do this ahead of class
    b. Observation: Participant performs the operation
    c. Conclusion: Participant understands CLI setup and use of Access Tokens in authentication
4. OPTIONAL: Download a docker image from Docker Hub using JFrog CLI (jf command)
    a. Hint: A docker image (such as nginx) is identified for participants to download
    b. Observation: Participant performs operation
    c. Conclusion: Participant understands CLI operations to use remote repositories’ operation to download
5. OPTIONAL: Upload a npm binary to a Local repository using JFrog CLI Comments are:

    In package.json, add this line
    "publishConfig":{"registry":"https://soleng.jfrog.io/artifactory/api/npm/swampup-npm-virtual/"}

    npm publish

    a. Hint: A local npm repository is created for each participant and NPM package is available for participants
    b. Observation: Participant performs operation using SetMeUp and CLI
    c. Conclusion: Participant understands CLI operations to use local repositories’ operation to upload
6. BONUS: Configure VSCode with JFrog IDE Plugin

Lab 2: Xray

1. Enable Xray Indexing on Local & Remote repositories
    a. Hint: Docker repositories with existing scanned docker images (in another docker repositories) are preferred
    b. Observation: Participants perform the operation
    c. Conclusion: Participants understand CVEs
2. Upload a Docker image, using docker native commands, that was downloaded in Lab1
    a. Hint: Docker images with 1st party code preferred
    b. Observation: Participants perform the operation
    c. Conclusion: Participants understand CVEs, CWEs, Secrets, etc.
3. BONUS: Scan the same docker image locally in their workstation using JFrog CLI
    a. Hint: Use the docker image from Docker Hub (interested only in CVEs)
    b. Observation: Participants perform the operation
    c. Conclusion: Participants understand CLI on-demand scanning

Lab 3: Distribution

1. Create a Release Bundle, and Distribute to an Edge node
    a. Hint: Use previously uploaded Docker images in a Local repository, along with a MP4 file (assumption is that this is the Installation video)
    b. Observation: Participants perform the operation
    c. Conclusion: Participants understand Distribution process
2. BONUS: Create a Dynamic Release Bundle using JFrog REST API

    https://jfrog.com/help/r/jfrog-rest-apis/dynamic-release-bundle

    a. Hint: Please this request and curl for easy download to workstations
    b. Observation: Participants perform the operation
    c. Conclusion: Participants understand REST API authentication & REST API Distribution

JFROG SETUP: JPD w/ Edge. Has Docker & NPM repos configured. Indexing enabled. Docker images are scanned for Basic and Advanced Security. Keep JPDs alive for a week after the session (Sep 20th). GitHub account

WORKSTATION SETUP: Git CLI, Docker Runtime, JFrog CLI, NPM Native Client, IDE (VSCode, IntelliJ), Web Browser (Chrome)

Lab1
Adding and Editing Configured Servers
jf config add / edit

Ping server
jf rt ping
To specify server:
jf rt ping --server-id=SERVER_ID

docker login
docker pull sup111epsumXX.jfrog.io/docker-hub-remote-repo/nginx:mainline-alpine3.18-slim
docker push