Git

```
# Git Documentation
# Git global setup
git config --global user.name "John Doe"
git config --global user.email
"johndoe@example.com"

# Clone and Edit a repository
git clone
git@git.example.com:repository/project.git
cd project
touch README.md
git add README.md
git commit -m "docs: add README"
git push -u origin main

# Convert existing folder to repo and push
cd existing_folder
git init
git remote add origin
git@git.example.com:repository/project.git
git add -A
git commit -m "Initial commit"
git push -u origin main
```

Pre-Commit

```
# Install Pre-Commit to manage git hooks that
can run before or after git commands
pip3 install pre-commit

# Prevent storing AWS credentials in git
cd repository

cat >> .pre-commit-config.yaml << EOF
---
repos:
  - repo: https://github.com/pre-commit/pre-
commit-hooks
    rev: v4.5.0
    hooks:
      - id: detect-aws-credentials
EOF
pre-commit install

# Update hook to the latest release
pre-commit autoupdate
```

Cloud Security Scanning

```
# Prowler is a multi-cloud audit tool
# Install and run the tool, which will use the
cloud APIs to gather information
# Ensure you're logged into the correct cloud
provider prior to running prowler
pip3 install prowler
prowler --help

# Review options for the aws, azure, and gcp
providers
prowler aws --help
prowler azure --help
prowler gcp --help
```

Docker

```
# Docker Documentation
docker pull <image>:<tag>
docker image ls
docker image rm <imageid>
docker container ls -a

# Run Containers in Detached Mode
docker run -d -p 443:443 nginx

# List all nginx containers
docker ps --filter ancestor=nginx

# Stop and delete the running container
latest_container_id=$(docker ps -n 1 --format
"{{.ID}}")
docker kill "${latest_container_id}"
docker rm "${latest_container_id}"

# Mount the current directory on your host into
the /host directory inside a container
docker run -v "$(pwd):/host" -it ubuntu:22.04

# Setup the local system for multi-platform
image builds
docker buildx inspect multiplatform || docker
buildx create --name multiplatform --driver
docker-container --use
```

SANS CLOUD SECURITY

Cloud Native Security Tool

Cloud Security and DevOps

By Jon Zeolla

Cheat Sheet v1.0.0

sans.org/cloud-security

Docker (continued)

```
# Create a minimal Dockerfile into an OCI-
compliant artifact
cat >> Dockerfile << EOF
FROM nginx
EOF
docker buildx build -o type=oci,dest=nginx.tar .

# Build and push a multiplatform image
docker login --username ace135 # Example User
docker buildx build                              \
  --platform=linux/amd64,linux/arm64 --push \
  --build-arg KEY=VALUE --tag ace135/demo .

# Build and push an image with an accompanying
SLSA attestation
docker buildx build --push --attest
type=provenance,mode=max -t ace135/demo:slsa .

# Extract the Dockerfile used to create the
specified image
docker buildx imagetools inspect jonzeolla/docker-
provenance:latest --format '{{ range (index
.Provenance.SLSA.metadata
"https://mobyproject.org/buildkit@v1#metadata").so
urce.infos }}{{ if eq .filename "Dockerfile"
}}{{.data }}{{ end }}{{ end }}' | base64 -d
```

Infrastructure as Code Scans

```
# Checkov is a misconfiguration scanner. It can scan Terraform, Kubernetes, Dockerfiles, and other file types.

pip3 install checkov
checkov -f example/file.tf

# Recursively scan a directory
checkov --directory .

# Find security misconfigurations in Helm Charts
checkov --framework helm --directory .

# Use easy infra to run IaC tools in a secure-by-default docker image
# Turn off security scans to ensure it functionally works in your environment
docker run -e DISABLE_SECURITY=true -v .:/iac seiso/easy_infra:latest-terraform terraform validate

# Run security scans but suppress failures
docker run -e LEARNING_MODE=true -v .:/iac seiso/easy_infra:latest-terraform terraform validate

# Fail on detected security issues
docker run -v .:/iac seiso/easy_infra:latest-terraform terraform validate

# Detect directories with terraform files, and run security scans and then terraform validate in each directory
docker run -e AUTODETECT=true -v .:/iac seiso/easy_infra:latest-terraform terraform validate

# See the logs from latest docker run
docker cp $(docker ps -n 1 --format "{{.ID}}"):/var/log/easy_infra.log .
cat easy_infra.log
```

Policy as Code

```
# Conftest uses a language called Rego to scan configuration files such as Terraform, Dockerfiles, Kubernetes manifests, and any other structured data
# See numerous examples here

# Write a policy that disallows the use of nginx
mkdir policy && cat > policy/policy.rego <<
EOF
package main

denylist := ["nginx"]

deny[msg] {
  some i
  input[i].Cmd == "from"
  val := input[i].Value
  contains(val[i], denylist[_])
  msg = sprintf("unallowed image found %s", [val])
}
EOF

# Create and scan a disallowed Dockerfile
cat > Dockerfile << EOF
FROM nginx:latest
EOF

docker run --rm -v $(pwd):/project openpolicyagent/conftest test Dockerfile

# Create and scan an allowed Dockerfile
cat > Dockerfile << EOF
FROM httpd
EOF

docker run --rm -v $(pwd):/project openpolicyagent/conftest test Dockerfile
```

AWS Systems Manager Parameter Store

```
aws ssm put-parameter --name MyParameter --value "secret value" --type SecureString

aws ssm get-parameter --name MyParameter --with-decryption
```

Azure Key Vault

```
# Create a Resource Group
az group create --name MyResourceGroup --location EastUS

# Create a new key in the keyvault
az keyvault create --name MyKeyVault --resource-group MyResourceGroup --location EastUS

# Show details of a key vault
az keyvault show --name MyKeyVault

# List Azure Key Vaults
az keyvault list --resource-group MyResourceGroup

# Delete a Key Vault
az keyvault delete --name MyKeyVault --resource-group MyResourceGroup
```

Join Us on Social

Twitter: @sanscloudsec

LinkedIn: SANS Cloud Security

YouTube: youtube/sanscloudsecurity

Discord: sansurl.com/cloud-discord