



RCE-as-a-Service:

Lessons Learned from 5 Years of Real-World CI/CD Pipeline Compromise

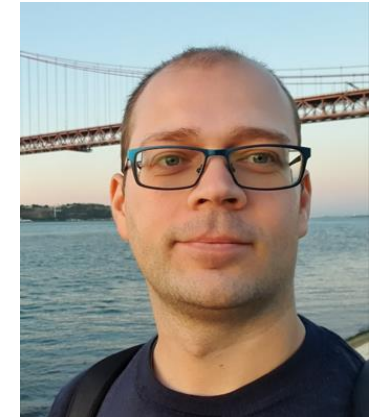
Iain Smart & Viktor Gazdag

WhoAreWe

Iain Smart (@smarticu5)

NCC Group Containerisation Practice Lead

6 years Container & Cloud Pentesting



Viktor Gazdag (@wucpi)

Jenkins Security MVP, Cloud Research Group Lead

7 years Pentest - Internal & External Team

Terminology

CI – Continuous Integration

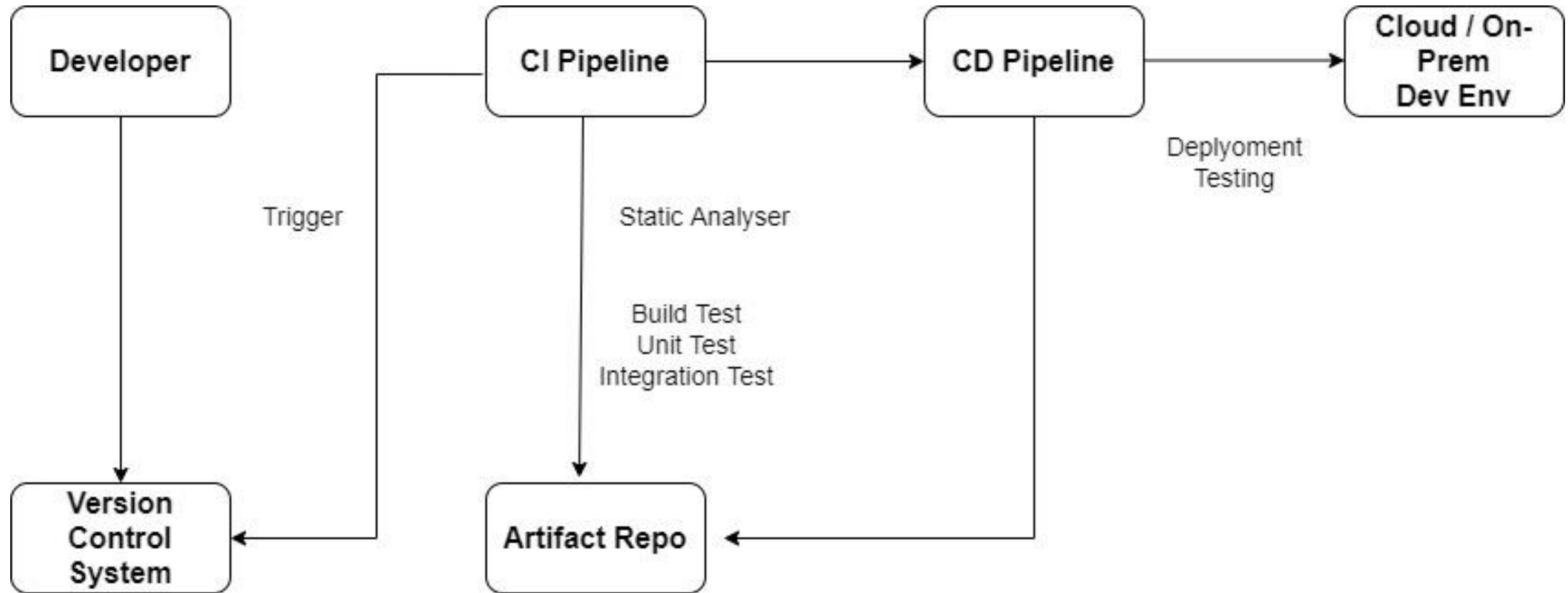
CD – Continuous Development/Deployment

Pipeline – A task or series of tasks performed automatically

Secrets – Something sensitive you wouldn't want to be made public (e.g. a password)

RCE – Remote Code Execution

CI/CD Overview



CI/CD Overview

PERIODIC TABLE OF DEVOPS TOOLS (V2) [EMBED](#) [DOWNLOAD](#) [ADD](#)

1 Fm Gh GitHub																	2 Fm Aws Amazon Web						
3 Os Gt Git	4 En Dt Datocal																	5 En Ch Chef	6 En Pu Puppet	7 Os An Ansible	8 Os Sl Salt	9 Os Dk Docker	10 Pd Az Azure
11 Fm Bb Bitbucket	12 Os Lb Liquibase																	13 Os Ot Otto	14 En Bl BladeLogic	15 Os Va Vagrant	16 Fr Tf Terraform	17 Os Rk rkt	18 En Gc Google Cloud
19 Os Gl GitLab	20 En Rg Redgate	21 Os Mv Maven	22 Os Gr Gradle	23 Os At ANT	24 Os Fn FitNesse	25 Fr Se Selenium	26 Os Ga Gatling	27 Fr Dh Docker Hub	28 Os Jn Jenkins	29 Pd Ba Bamboo	30 Os Tr Travis CI	31 Pd Gd Deployment Manager	32 Os Sf SmartFrog	33 Os Cn Consul	34 Os Bc Bcfg2	35 Os Mo Mesos	36 En Rs Rackspace						
37 Os Sv Subversion	38 En Dm DBmaestro	39 Os Gn Grunt	40 Os Gp Gulp	41 Os Br Broccoli	42 Fr Cu Cucumber	43 Os Cj Cucumber.js	44 Fr Qu Qunit	45 En Npm npm	46 Fm Cs Codeship	47 Pd Vs Visual Studio	48 Fm Cr CircleCI	49 Fr Cp Capistrano	50 Fr Ju JuJu	51 Os Rd Rundeck	52 Os Cf CFEngine	53 Fr Ds Swarm	54 Os Op OpenStack						
55 Os Hg Mercurial	56 En Dp Delphix	57 Fr Sb sbt	58 Os Mk Make	59 Os Ck CMake	60 Fr Jt JUnit	61 Fr Jm JMeter	62 Fr Tn TestNG	63 Os Ay Artifactory	64 Fm Tc TeamCity	65 Fm Sh Shippable	66 Os Cc CruiseControl	67 En Ry RapidDeploy	68 Fm Cy AWS CodeDeploy	69 En Oc Octopus Deploy	70 En No CA Nolio	71 Os Kb Kubernetes	72 Fm Hr Heroku						
73 En Cw ISPW	74 En Id Idera	75 Os Msb MSBuild	76 Os Rk Rake	77 Os Pk Packer	78 Os Mc Mocha	79 Fr Km Karma	80 Os Jm Jasmine	81 Os Nx Nexus	82 Os Co Continuum	83 Fm Ct Continua CI	84 Pd So Solano CI	85 En Xld XL Deploy	86 En EB ElasticBox	87 Fm Dp Deploybot	88 En Ud UrbanCode Deploy	89 Os Nm Nomad	90 En Os OpenShift						

PERIODIC TABLE OF DEVOPS TOOLS (V2) [EMBED](#) [DOWNLOAD](#) [ADD](#)

Os	Open Source	SCM	Database Mgmt	Build
Fr	Free	CI	Repo Mgmt	Testing
Fm	Freemium	Deployment	Config / Provisioning	Containerization
Pd	Paid	Cloud / IaaS / PaaS	Release Mgmt	Collaboration
En	Enterprise	BI / Monitoring	Logging	Security

Xebialabs
Enterprise DevOps

Follow @xebialabs
Publication Guidelines

91 En Xlr XL Release	92 En Ur UrbanCode Release	93 En Bm BMC Release	94 En Ca CA Release Automation	95 En Au CA Automic	96 En Pl Plutora Release	97 En Sr Micro Focus Release	98 Pd Tfs Team Foundation	99 Fm Tl Trello	100 Pd Jr Jira	101 Fm Rf Stride	102 Fm Sl Slack	103 Fm Fd Flowdock	104 Pd Pv Pivotal Tracker	105 En Sn ServiceNow
108 Os Ki Kibana	107 Fm Nr New Relic	108 En Dt Dynatrace	109 Os Ni Nagios	110 Os Zb Zabbix	111 En Dd Datadog	112 Os El Elasticsearch	113 Fm Ad AppDynamics	114 En Sp Splunk	115 Fm Le Logentries	116 Fm Sl Sumo Logic	117 Os Ls Logstash	118 Os Sn Snort	119 Os Tw Tripwire	120 En Ff Fortify WebInspect

Attacker's View

Code goes in, apps come out

Pipelines have network access to a range of environments

Pipelines have credentials to pull dependencies and push artifacts

Free compute resources



Scenarios

Developers Access

Git Access

Granted developer access to an internal Bitbucket repo

Git Access > Load Dependencies

Apache Maven Project

Could modify project dependencies

Hosted dependency on an attacker-controlled URL

Git Access > Load Dependencies > Jenkins Runner Shell

Meterpreter shell from Jenkins runner

Limited environment means easy recon

Found keys

Git Access > Load Dependencies > Jenkins Runner Shell > **Pivot**

Network Recon – SSH

File system access – SSH key

Put them together? It just makes sense.

Git Access > Load Dependencies > Jenkins Runner Shell > Pivot > **Root**

SSH access to Jenkins control nodes

Secrets for all projects

Kubeconfig file

Lessons Learned

Build/Credential hygiene

Network filtering

Dependency validation and sources

Confusing Wording

GitHub Authorization

GitHub Authorization Plugin

"Grant READ Permissions to All Authenticated Users"

- Not "All Authenticated Users in this Organisation"

GitHub Authorization > Jenkins Access with Gmail Account

Register an arbitrary account in GitHub to log in

Log in to Jenkins

Access to build jobs and history

No monitoring and alerting

Lessons Learned

Read the description and test

Least privilege principal

Separation of duties

Monitoring and alerting

Build Output

Jenkins Login

Authenticated Users had Overall/Administrator role in Jenkins

Infra as Code pipeline with terraform cli within the steps

No monitoring and alerting

Jenkins Login > Job Logs

Terraform cli output in build output

Output did not mask any secrets

Showed all users AWS API keys

Lessons Learned

Least privilege principal

Separation of duties

Dedicated plugin with output masking

Monitoring and alerting

FSAS Engagement

Entry

Red team compromised a developer account

Git access to a range of repositories

Entry > Pipeline Access

Pipeline modifications

Pipeline prevented displaying of env variables

```
printenv | base64
```

Entry > Pipeline Access > Credentials

Service account with Domain Admin access

Lessons Learned

Restrict secrets to specific branches

Don't run anything as Domain Admin

Web Application LFI

Web App

Internal infrastructure assessment

Identified web application SSRF/LFI vulnerabilities

Application was deployed for testing mid-pipeline

Web App > Kubernetes

Read service account token through LFI

Use to communicate with Kubernetes API Server

Permissions to edit Configmaps (AWS EKS authentication)

Gained cluster admin on the build cluster

Web App > Kubernetes > Container Registry

Stole AWS IAM credentials to deploy to ECR

Web App > Kubernetes > Container Registry > **Kubernetes**

Production workloads used pull-based CI

Overwrote images in ECR, new images were automatically pulled

Lessons Learned

Network isolation

RBAC – least privilege

From Cloud Web App to On-Prem Server

Web App

Started of WordPress with some custom pages

Application had directory listings enabled S3 bucket linked as sitemap

File responsible to push code to GitHub

Web App > Git

File had hardcoded credentials in S3 allowed access to VCS

Developer access to multiple repositories with read and write

No alert on signing in with the account, but had logs

Web App > Git > **Jenkins**

Same credentials allowed access to Jenkins, as an administrator

14 cluster with 200 build servers or agents

Web App > Git > Jenkins > **Lateral Movement**

Cluster admin on 14 Jenkins Master and 200 Jenkins build servers

Dump credentials from Jenkins: 200+ API Tokens, SSH Keys

Lessons Learned

Credential hygiene

Secret management

Least privilege principal

Lack of monitoring and alerting

Branched-Based Secrets

Developer Access

Developers had no direct access to production cloud environments

Only main branch deployed to prod

Main branch protected, and required merge approval

Developer Access > Circle CI

Pipelines configured using CircleCI YAML files

Configuration files in the same repo as application code

Secrets specified per branch

Developer Access > Circle CI

```
- name: Do Dev things
image: registry.customer.com/terraform:v0.12
environment:
  DEV_AWS_ACCESS_KEY_ID:
    from_secret: DEV_AWS_ACCESS_KEY_ID
  DEV_AWS_SECRET_ACCESS_KEY:
    from_secret: DEV_AWS_SECRET_ACCESS_KEY
commands:
- terraform apply
when:
  branch:
  - feature/dev*
```

Developer Access > Circle CI

```
- name: Do Prod things
image: registry.customer.com/terraform:v0.12
environment:
  PROD_AWS_ACCESS_KEY_ID:
    from_secret: PROD_AWS_ACCESS_KEY_ID
  PROD_AWS_SECRET_ACCESS_KEY:
    from_secret: PROD_AWS_SECRET_ACCESS_KEY
commands:
- terraform apply
when:
  branch:
  - main
```

Developer Access > Circle CI > Privilege Escalation

Developers could modify pipeline configuration file in non-Main branches

```
- name: Do Hacky things
  image: registry.customer.com/terraform:v0.12
  environment:
    PROD_AWS_ACCESS_KEY_ID:
      from_secret: PROD_AWS_ACCESS_KEY_ID
    PROD_AWS_SECRET_ACCESS_KEY:
      from_secret: PROD_AWS_SECRET_ACCESS_KEY
  commands:
  - printenv
  when:
    branch:
      - *
```

Lessons Learned

Validate secret management/protection

Assume every developer is malicious (or trust them)

Log and audit credential use

Least privilege RBAC

Privilege Escalation

SSO Credentials

Login with SSO Credentials

Testing 3 separated roles: read, build, admin

Locked down and documented roles

SSO Credentials > Privilege Escalation

Build user with Build/Replay permission

Code execution with Groovy

SSO Credentials > Privilege Escalation > Credentials Dump

Credentials dump

Lessons Learned

Least privilege principal

Common Themes

Service hardening

Network segmentation

Monitoring & alerting

Patch management

RBAC (is still difficult)

Preventions

Threat modelling

Network segmentation

Patch management

Monitoring & alerting

Secrets management

RBAC – Least privilege

Conclusion

CI/CD is beneficial, but also complex

None of these problems are new

Environments are rarely threat modelled

Little focus on the pipeline and infra

Questions?

@smarticu5 – iain.smart@nccgroup.com

@wucpi – viktor.gazdag@nccgroup.com