

Red Teaming Devops

Jose Hernandez & Rod Soto

\$whoami

Jose Hernandez

José is a Principal Security Researcher at Splunk. He started his professional career at Prolexic Technologies (now Akamai), fighting DDOS attacks from “*anonymous*” and “*lulzsec*” against Fortune 100 companies. As a engineering co-founder of Zenedge Inc. (acquired by Oracle Inc.), José helped build technologies to fight bots and web-application attacks. While working at Splunk as a Security Architect, he built and released an auto-mitigation framework that has been used to automatically fight attacks in large organizations. He has also built security operation centers and run a public threat-intelligence service. Although security information has been the focus of his career, José has found that his true passion is in solving problems and creating solutions. As an example, he built an underwater remote-control vehicle called the SensorSub, which was used to test and measure toxicity in Miami's waterways.

Rod Soto

Principal Security Research Engineer at Splunk. Worked at Prolexic, Akamai, Caspida. Won BlackHat CTF in 2012. Co-founded Hackmiami, Pacific Hackers meetup and conferences.

What is Devops

DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes. [* Source AWS](#)

Why is Devops so popular ?

- Faster deployment of applications
- Shared responsibility and collaboration among teams
- Apply security to software development practices and platform application deployment
- Scale and manage infrastructure and development of software
- Make applications and deployed infrastructure reliable by being able to apply updates & upgrades
- The Devops practices have become a key factor in the use and expansion of cloud platforms.

Devops practices are tied to a software development approach

Agile software development comprises various approaches to **software development** under which requirements and solutions evolve through the collaborative effort of **self-organizing** and **cross-functional** teams and their **customer(s)/end user(s)**.^[1] It advocates adaptive planning, evolutionary development, early delivery, and **continual improvement**, and it encourages rapid and flexible response to change. [Wikipedia](#)

Devops toolchains

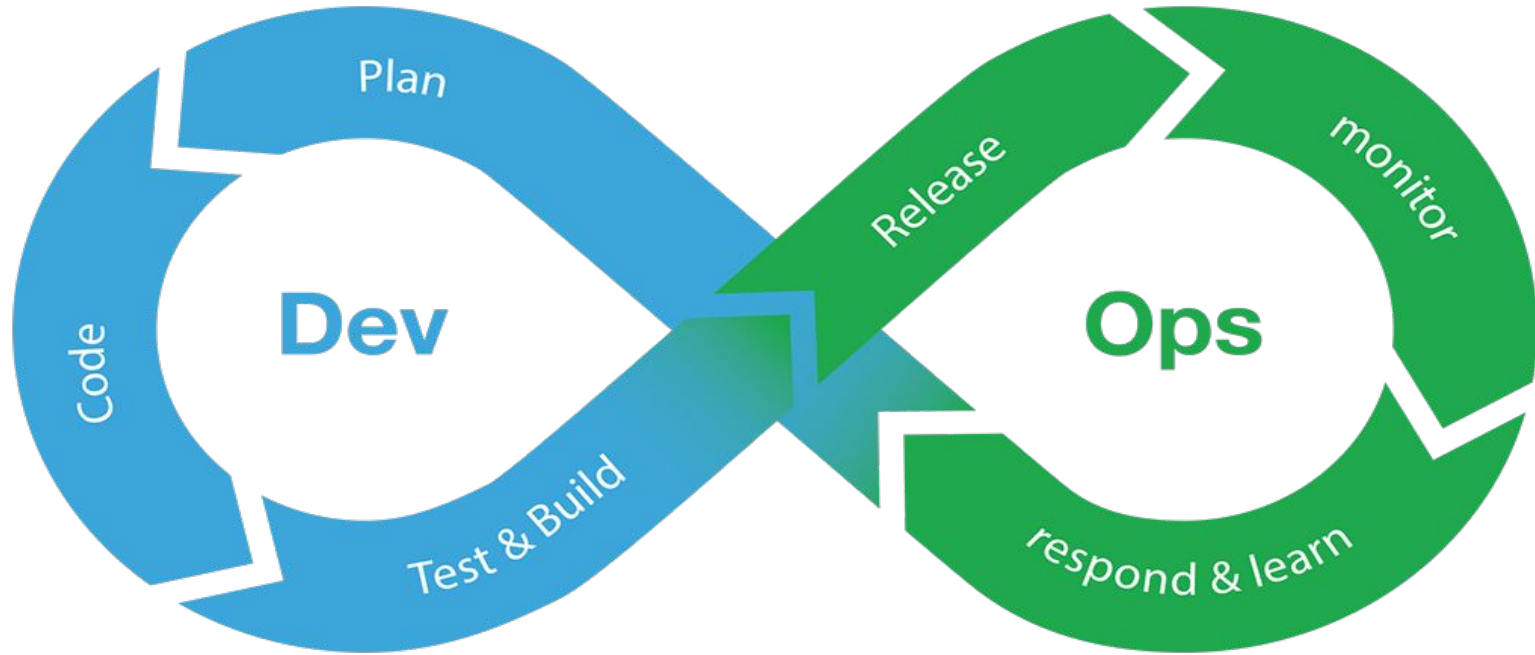
A **DevOps toolchain** is a set or combination of tools that aid in the delivery, development, and management of software applications throughout the systems development life cycle, as coordinated by an organisation that uses **DevOps** practices.[*Wikipedia](#)

Devops toolchains



DevOps Pipeline Tool Overview (selection of tools, note: image/logo rights are with the respective copyright owners)

Continuous Integration/Delivery/Deployment



Devops Attack Surface (CI / CD Pipeline)

Source Code Repository (Code)

Bitbucket, Beanstalk, Github, Gitlab, SVN, S3 buckets

CI / CD Platform (Build)

TravisCI, Jenkins, CircleCI, Gitlab

Container Repository (Code, Build, Test)

Docker, Vagrant

Planning Ticket (Plan)

Github, JIRA, Trello

IaaS Provider (Release, Deploy, Operate)

Kubernetes flavor, OpenStack (this may also be local in some private, hybrid environments)

IaC (Release, Deploy)

Ansible, Terraform, Chef, Cloudformation, PowerShell DSC

Monitor tool (Monitor)

ELK, Splunk

Testing toolset (Test)

Selenium, Chai

Red teaming Devops

due to less polution, the files stored on cloud are now clearly visible



Red teaming Devops - hypothesis / adv sim

- Is the target organization aware of Devops CI/CD attack surface?
- Is this being addressed or just dismissed as another 3rd party (pre cloud adoption mentality)
- Are there any mechanisms to assess risk and deploy operational controls based on threat likelihood?
- Is there any type of monitoring or detection against related threats?
- What is the organization's defense posture and possible stop-gap measures against these attack vectors (resilience)?

Red Teaming Devops

OPERATOR

Internet Cloak & Dagger

TARGET

Client Based - Inside Perimeter

- API Keys/Certificates at Desktop/Servers
- Testing toolset (Selenium, Chai)
- B. Platform (Jenkins)
- IaC language (Ansible, Powershell DSC)
- Phish admin/dev creds for Cloud platform/tools
- Exploit SDK/CLI at client Cloudformation

API, Web service, Storage object - Outside Perimeter

- API Keys at repository (github)
- App vulnerabilities (Ngnix RCE)
- Exposed Writable Buckets (S3)
- Container Repos (Dockerhub)
- Monitor ELK/Splunk
- Misconfigured Provider Mgmt platform (No MFA) AWS, GCP, AZ

Red teaming Devops

- Use 3rd party SaaS PaaS IaaS C2s (Gmail, Github, Twitter, Box, AWS, GCP, Azure, etc)
- Research 3rd parties (DB, Analytic providers, SaaS)
- Usually common misconfigurations and or NO asset management at all
- Open source projects are usually based on SHARING info and code...
- Many organizations do not have ANY control on Container repo version uploading rights or even sharing

Red teaming Devops

- Target release/deploy languages (Powershell DSC, Ansible, Terraform)
- Dependence on the cloud means many times there is no communication between cloud provider and actual security/support team
- If access is achieved then emulate target behavior, many cloud operating environments have very poor vision on cross-account activity unless it is noisy or obvious
- Cloud facing applications in many environments have partial or direct connection to internal environments. Pivoting north south is possible now as lateral movement.

Pick your toolchain target...

CODE	BUILD	TEST	RELEASE	DEPLOY
Hardcoded APIs	Docker (CVE-2019-16884) Vagrant (CVE-2017-1617)	Docker (CVE-2019-16884) Vagrant (CVE-2017-1617)	Kubernetes (CVE-2018-1002105)	Ansible (CVE-2019-14905) CloudFormation (CVE-2017-9450)
Application, dev language Vulnerabilities	Container Implantation Mitre T1125	CircleCI (Breach)	Gitlab (CVE-2019-15737)	Kubernetes (CVE-2019-1002100)
Weak Authentication	Jenkins CVE-2019-1003000 CircleCI (Breach)	Selenium (CVE-2016-10589) Chai -> based on nodejs	Ansible (CVE-2019-3828)	Terraform (CVE-2018-9057) Powershell DSC

Pick your toolchain target...

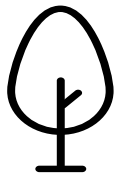
OPERATE	PLAN	MONITOR
Kubernetes (CVE-2019-11246)	Github (CVE-2019-1349)	ELK (CVE-2019-7609)
Terraform (CVE-2018-9057)	JIRA (CVE-2019-11581) Trello (CVE-2017-9244)	Splunk (CVE-2017-17067)

Practical Examples



Exploiting Github for secrets

Low Hanging Fruit



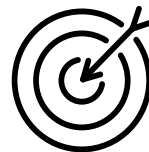
Easy to commit mistakes,
usually high priv accounts

Industry Standard



Very commonly
used by devops

Many Targets



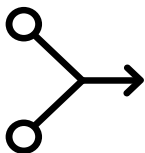
Gitlabs, BitBucket, and
others

Exposition #1 🤫 shhgit 📺 demo

<https://www.youtube.com/watch?v=JYl8zt0Czkg>

Challenges with Hunting for Secrets in Public Repos

API Limits



APIs usually limit you to query by user or repo

Scale Verification



Verifying secrets by hand does not scale

No Automation



Workflow of Parsing, Storing, Reporting is ripe for automation

git-wild-hunt 📺 demo

<https://www.youtube.com/watch?v=dtAcPuy8TR8>

Good old Phishing

<https://github.com/htr-tech/zphisher>

<https://medium.com/@rvrsh3ll/azure-app-services-for-offensive-operations-f1de99a83fa0>

The image is a screenshot of a web browser displaying a phishing attack on a Microsoft Azure App Service. The browser's address bar shows the URL `portal.azure.com/#@rodhackmiami.onmicrosoft.com/resource/subscriptions/1c551808-ebc3-4bc7-8001-63c65248d506/res...`. The browser's tab bar shows several tabs, including "Not Secure | percent27.azurewebsites.net/login.html" and "portal.azure.com/#@rodhackmiami.onmicrosoft.com/resource/subscriptions/1c551808-ebc3-4bc7-8001-63c65248d506/res...".

The main content area is split into two panes. The left pane displays a Microsoft login page with the Microsoft logo, the text "Sign in", and input fields for "Email, phone, or Skype" and "Password". There is a "Keep me signed in" checkbox and a "Forgot my password" link. A blue "Sign in" button is at the bottom. The right pane displays the Azure portal console for the "percent27" App Service. The console shows a file explorer with a search bar and a list of files and folders, including "MySQL In App", "Properties", "Locks", "Export template", "App Service plan", "App Service plan", "Quotas", "Change App Service plan", "Development Tools", "Clone App", "Console", "Advanced Tools", "App Service Editor (Preview)", "Resource explorer", and "Extensions". The "Console" tab is selected, showing a terminal window with the following output:

```
Converged_v21033.css
Converged_v21033.css2
boot.css
boot.js
boot_002.js
boot_003.js
boot_004.js
ellipsis_grey.svg
ellipsis_white.svg
index.php
ip.php
ip.txt
login.html
login.php
login_info.txt
microsoft_logo.svg
prefetch.html
sprite1.css
sprite1.png
usernames.txt
victim_ip.txt

D:\home\site\wwwroot>cat usernames.txt
Username: rodvictim Pass: victimpassword

D:\home\site\wwwroot>
```

The bottom of the image shows a footer with the text "©2020 Microsoft Terms of use Privacy & cookies".

Now you have creds now what?

How you heard of the MITRE ATT&CK Matrix .. for cloud 😂

Implant Container Image |

Amazon Web Service (AWS) Amazon Machine Images (AMI), Google Cloud Platform (GCP) Images, and Azure Images as well as popular container runtimes such as Docker can be implanted or backdoored to include malicious code. Depending on how the infrastructure is provisioned, this could provide persistent access if the infrastructure provisioning tool is instructed to always use the latest image.^[1]

A tool has been developed to facilitate planting backdoors in cloud container images.^[2] If an attacker has access to a compromised AWS instance, and permissions to list the available container images, they may implant a backdoor such as a web shell.^[1] Adversaries may also implant Docker images that may be inadvertently used in cloud deployments, which has been reported in some instances of cryptomining botnets.^[3]

ID: T1525

Tactic: Persistence

Platform: GCP, Azure, AWS

Permissions Required: User

Contributors: Praetorian

Version: 1.0

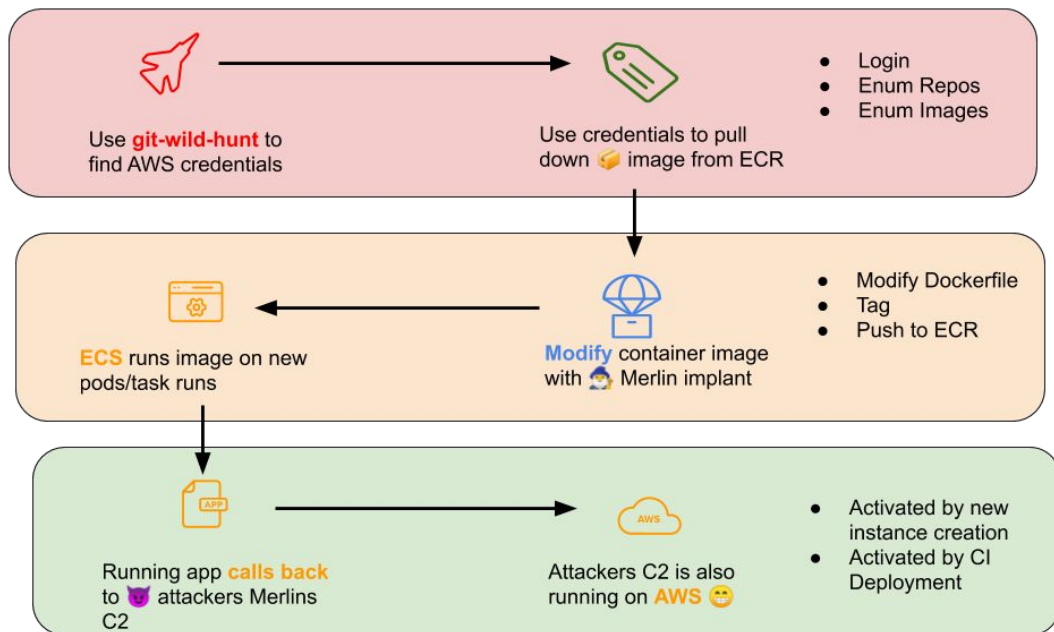
Created: 04 September 2019

Last Modified: 22 October 2019

Mitigations

Mitigation	Description
Audit	Periodically check the integrity of images and containers used in cloud deployments to ensure they have not been modified to include malicious software.
Code Signing	Several cloud service providers support content trust models that require container images be signed by trusted sources. ^{[4][5]}
Privileged Account Management	Limit permissions associated with creating and modifying platform images or containers based on the principle of least privilege.

Attack Flow



Implantation with  Merlin C2  demo

<https://youtu.be/Nv9bMnCrKD8>

Other post exploitation opportunities to consider

Master_Of_Servers ([MOSE](#)) - Ansible/Chef/Puppet post exploitation tool

Splunk [backdoors](#) - backdoored apps

Enumerate further cloud resources for further compromise

Find hybrid cloud/perimeter assets North->South - East<=>West

Find pre-stage potentially production bound code, creds and data

Apply what you learned today



Tools of the Trade:

- [Shhgit](#) to see real time secrets leaked
- [Git-wild-hunt](#) search github for secrets
- [Merlin](#) C2 (Go) Precompiled HTTPS C2
- Azure Phishing [zphisher](#)
- [MOSE](#) Configuration Management Systems post-exploitation
- Videos of demos here <https://www.youtube.com/playlist?list=PLzU1TULfG3ozAIWZ1RUTAcekJX7uJCEh>

Chat with us in Discord:

<https://redteamvillage.io/discord/>

Or  Twitter

Rod Soto [@rodsoto](#)

Jose Hernandez [@d1vious](#)