The Future of AppSec is Cloud-Native

Jimmy Mesta
Founder at KSOC
CTO & Trainer at Manicode Security
@jimmesta

Jim Manico
Founder at Manicode Security
@manicode
AppSec and Cloud-Native
A Match Made in the Sky
AppSec and Cloud-Native
A Match Made in the Sky
Whether we like it or not
AppSec and Cloud-Native
A Match Made in the Sky
Whether we like it or not
Seriously...just get used to it
Hello! I’m Jimmy. 👋
- AppSec
- DevOps
- Cloud-Native
- CloudDevAppSecOpsNative
- Kubernetes: A Case Study
- Take Home Assignment
AppSec
Traditional AppSec relied on a monolith, guarded by a perimeter of firewalls and blinking boxes.
We scanned, we tested pens, we understood the OWASP Top Ten, maybe even began threat modeling.
We kicked off bug bounties, our pipelines started evolving and we even hired an “AppSec” engineer.
DevSecOps
The DevOps movement was a pivotal time in history for AppSec. We had to make a choice.
Do we embrace the new culture of collaboration and tooling that spans the entire SDLC or stick with what we know?
“If we have data, let’s look at data. If all we have are opinions, let’s go with mine.”

- Jim Barksdale
Accelerate
Building and Scaling High Performing Technology Organizations

Nicole Forsgren, PhD
Jez Humble, and Gene Kim
### Table 2.2 Software. Delivery Performance for 2016

<table>
<thead>
<tr>
<th></th>
<th>High Performers</th>
<th>Medium Performers</th>
<th>Low Performers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>2016</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deployment Frequency</td>
<td>On demand (multiple deploys per day)</td>
<td>Between once per week and once per month</td>
<td>Between once per month and once every six months</td>
</tr>
<tr>
<td>Lead Time for Changes</td>
<td>Less than one hour</td>
<td>Between one week and one month</td>
<td>Between one month and six months</td>
</tr>
<tr>
<td>MTTR</td>
<td>Less than one hour</td>
<td>Less than one day</td>
<td>Less than one day*</td>
</tr>
<tr>
<td>Change Failure Rate</td>
<td>0-15%</td>
<td>31-45%</td>
<td>16-30%</td>
</tr>
</tbody>
</table>
### Table 2.2 Software. Delivery Performance for 2016

<table>
<thead>
<tr>
<th></th>
<th>High Performers</th>
<th>Medium Performers</th>
<th>Low Performers</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Deployment Frequency</strong></td>
<td>On demand (multiple deploys per day)</td>
<td>Between once per week and once per month</td>
<td>Between once per month and once every six months</td>
</tr>
<tr>
<td><strong>Lead Time for Changes</strong></td>
<td>Less than one hour</td>
<td>Between one week and one month</td>
<td>Between one month and six months</td>
</tr>
<tr>
<td><strong>MTTR</strong></td>
<td>Less than one hour</td>
<td>Less than one day</td>
<td>Less than one day*</td>
</tr>
<tr>
<td><strong>Change Failure Rate</strong></td>
<td>0-15%</td>
<td>31-45%</td>
<td>16-30%</td>
</tr>
</tbody>
</table>
DevSecOps is the process of incorporating and enforcing meaningful security controls without slowing down deployment velocity.
DevSecOps means that we are all responsible for security.
Cloud-Native
Modern software teams must focus on speed, agility, security, and stability.
Cloud-Native is a set of design patterns, not just a collection of tools.
Cloud-Native patterns enable organizations to deliver software at a rapid velocity with more confidence.
- Observability
- Application Portability
- Loosely Coupled Services
- Policy Driven Infrastructure
- Automated Pipelines
Observability encompasses the log aggregation, monitoring, metric collection, to gain insight into a given system.
Application portability gives developers the ability to write code once and run [almost] anywhere.
Loosely coupled services are often API-driven and allow applications to run, scale, and interact as an independent entity.
Policy driven infrastructure gives operators the ability to define the desired state of applications and underlying infrastructure using definition files.
Automated pipelines remove manual steps involved in shipping code from commit to production.
The combination of cloud native tools and patterns help organizations achieve product goals.
The Future of AppSec is Cloud-Native
The Future of AppSec is CloudDevAppSecOpsNative
Our illusion of control is gone.

Public APIs, multi-cloud environments, third-party components, containers, and cloud-native tools changed the game.
### SSRF in Exchange leads to ROOT access in all instances

<table>
<thead>
<tr>
<th>State</th>
<th>Resolved (Closed)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Disclosed</td>
<td>May 23, 2018 2:09pm -0700</td>
</tr>
<tr>
<td>Reported To</td>
<td>Shopify</td>
</tr>
<tr>
<td>Asset</td>
<td><a href="https://exchangemarkplace.com/">https://exchangemarkplace.com/</a> (Domain)</td>
</tr>
<tr>
<td>Weakness</td>
<td>Server-Side Request Forgery (SSRF)</td>
</tr>
<tr>
<td>Bounty</td>
<td>$25,000</td>
</tr>
<tr>
<td>Severity</td>
<td>Medium (6.9)</td>
</tr>
<tr>
<td>Participants</td>
<td></td>
</tr>
<tr>
<td>Visibility</td>
<td>Disclosed (Full)</td>
</tr>
</tbody>
</table>

https://hackerone.com/reports/341876
“Through 2020, 80% of cloud breaches will be due to customer misconfiguration, mismanaged credentials or insider theft, not cloud provider vulnerabilities.”

Gartner
"A historic 424% jump in breaches related to misconfigured cloud infrastructure, largely due to human error."

2017 IBM X-Force Report
"Cloud configuration overtakes ‘phishing’ as top source of breached data."

Yep. This is happening as we speak.
CVE-2019-9901 - Istio/Envoy Path traversal

TLDR; I found a path traversal bug in Istio's authorization policy enforcement.

https://github.com/eoftedal
Envoy Authorization Policy

```yaml
rules:
  - services: ["backend.fishy.svc.cluster.local"]
    methods: ["GET"]
    paths: ["/public/*"]
```

Path Traversal Payload

curl -vvvv --path-as-is "http://backend.fishy.svc.cluster.local:8081/public/../secret/"

???

???
Contractor's AWS S3 server leaks data from Fortune 100 companies: Ford, Netflix, TD Bank

Exposed data includes passwords and private keys for production systems, employee details, sales information.

If blindly adopted, Cloud-Native patterns and tools can introduce a slew of new vulnerabilities into your infrastructure.
But, when properly prepared, Cloud-Native patterns can help AppSec move faster, and more safely than ever before.
Visibility
Discovery

<table>
<thead>
<tr>
<th>Service</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>consul</td>
<td>1 passing</td>
</tr>
<tr>
<td>product-service</td>
<td>2 passing</td>
</tr>
<tr>
<td>user-service</td>
<td>2 passing</td>
</tr>
</tbody>
</table>

product-service

TAGS
No tags

NODES

922c9dbc0c1f 127.0.0.1

Serf Health Status serfHealth

Service 'product-service' check
AuthN / AuthZ

"The SPIFFE Runtime Environment forms a powerful solution for connecting, authenticating, and securing workloads in distributed environments."

https://www.cncf.io/blog/2018/03/29/cncf-to-host-the-spiffe-project/
Community

CNCF Special Interest Group on Security -- secure access, policy control, privacy, auditing, explainability and more!

https://cncf.io/projects

https://github.com/cnf/sig-security
## Automation and Recovery

### Stage View

<table>
<thead>
<tr>
<th>Declarative: Check Differences</th>
<th>Declarative: Apply Changes</th>
<th>License Check</th>
<th>Prepare Environment</th>
<th>Build</th>
<th>Run Automation</th>
<th>Evaluate Changes</th>
<th>Publish</th>
<th>Collect Distribution Files</th>
<th>Finish Release</th>
<th>Declarative: Post Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>6s</td>
<td>121ms</td>
<td>9s</td>
<td>9s</td>
<td>4min 32s</td>
<td>28min 43s</td>
<td>5min 37s</td>
<td>6min 50s</td>
<td>22s</td>
<td>10s</td>
<td>11s</td>
</tr>
</tbody>
</table>

**Average stage times:**
(Average full run time: ~45min 52s)

<table>
<thead>
<tr>
<th>#3 Version: 1.49.0-01</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jul 11</td>
</tr>
<tr>
<td>10:38</td>
</tr>
<tr>
<td>30 commits</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#2 Version: 1.48.0-02</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jun 27</td>
</tr>
<tr>
<td>13:31</td>
</tr>
<tr>
<td>54 commits</td>
</tr>
</tbody>
</table>

**RSAC 2019 Conference - Asia Pacific & Japan**
## Whitepapers

<table>
<thead>
<tr>
<th>Title</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>AWS Best Practices for DDoS Resiliency</td>
<td>June 2018</td>
</tr>
<tr>
<td>AWS Cloud Adoption Framework: Security Perspective</td>
<td>June 2016</td>
</tr>
<tr>
<td>AWS Security Checklist</td>
<td></td>
</tr>
<tr>
<td>AWS Well-Architected Framework: Security Pillar</td>
<td>July 2018</td>
</tr>
<tr>
<td>Introduction to AWS Security</td>
<td>July 2015</td>
</tr>
<tr>
<td>Introduction to AWS Security Processes</td>
<td>June 2016</td>
</tr>
<tr>
<td>Overview of AWS Security - Analytics, Identity and Application Services</td>
<td>June 2016</td>
</tr>
<tr>
<td>Overview of AWS Security - Application Services</td>
<td>June 2016</td>
</tr>
<tr>
<td>Overview of AWS Security - Compute Services</td>
<td>June 2016</td>
</tr>
<tr>
<td>Overview of AWS Security - Database Services</td>
<td>June 2016</td>
</tr>
<tr>
<td>Overview of AWS Security - Network Services</td>
<td>August 2016</td>
</tr>
<tr>
<td>Overview of AWS Security - Storage Services</td>
<td>June 2016</td>
</tr>
<tr>
<td>Secure Content Delivery with CloudFront</td>
<td>November 2016</td>
</tr>
<tr>
<td>Securing Data at Rest with Encryption</td>
<td>November 2014</td>
</tr>
</tbody>
</table>

## Platform Security
Chenxi Wang
@chenxiwang

I am seeing a quiet revolt against many security tools, b/c they do not play nicely with modern dev practices.

1:51 PM - 6 Jun 2019

117 Retweets  309 Likes
Kubernetes: A Case Study
Kubernetes is an open-source platform built to automate deployment, scaling, and orchestration of containers.
”Kubernetes is a pile of Linux goop.”
https://thenewstack.io/introducing-microservices-hierarchy-needs/
I thought Kubernetes was secure by default?
Details

Name: aws-s3-credentials

Namespace: default

Creation time: 2017-10-12T22:29

Type: Opaque

Data

aws-s3-access-key-id:

aws-s3-secret-access-key:
SSRF in Exchange leads to ROOT access in all instances

3 - Using Kubelet to execute arbitrary commands

It's possible to list all pods (F289460):

```bash
$ kubectl --client-certificate client.crt --client-key client.pem --certificate-authority ca.crt --server

NAMESPACE  NAME
[REDACTED]  1/1
```

And create new pods as well:

```bash
$ kubectl --client-certificate client.crt --client-key client.pem --certificate-authority ca.crt --server

pod "shell-demo" created
```

```bash
$ kubectl --client-certificate client.crt --client-key client.pem --certificate-authority ca.crt --server

pod "shell-demo" deleted
```
How A Cryptocurrency Miner Made Its Way onto Our Internal Kubernetes Clusters

Brian Choy in JW Player Engineering  Follow
Mar 19 · 10 min read

Security bulletins

- May 31, 2019
- May 14, 2019
- April 5, 2019
- March 1, 2019

All security bulletins for Google Kubernetes Engine are described in this topic.
Source: Kubernetes Security - Operating Kubernetes Clusters and Applications Safely
RBAC
Container and Pod Permissions
Pod Security Policies
Dynamic Admission Control
Sandboxing
Node Protection
Role-Based Access Control (RBAC) is how we regulate access to Kubernetes resources.
Users
you@email.com
Service account

API Resources
Namespaces
Pod
Service
Secrets
...

Operations
Get
List
Delete
Patch
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  namespace: development
  name: pod-reader
rules:
- apiGroups: [""
  resources: ["pods"]
  verbs: ["get", "list"]
Containers may request elevated privileges such as running as root, mounting sensitive volumes, or requesting access to specific ports.
Pod specifications may declare to access devices on the host using privileged mode.
Pod Security Policies give administrators the ability to validate requests to the cluster based on security requirements.
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: my-psp
spec:
  privileged: false
  seLinux:
    rule: RunAsAny
  supplementalGroups:
    rule: RunAsAny
  runAsUser:
    rule: MustRunAsNonRoot
  volumes:
  - 'configMap'
  - 'emptyDir'
  - 'secret'
  - 'persistentVolumeClaim'
Dynamic Admission Control allows teams to build custom security checks by intercepting requests to the Kubernetes API server prior to scheduling the object.
An Kubernetes validating admission webhook that rejects pods that use environment variables.

- kubernetes
- serverless
- faas
- gcp-cloud-functions

https://github.com/kelseyhightower/denyenv-validating-admission-webhook
Tools such as gVisor and Kata Containers can help further isolate and sandbox containers that are running untrusted workloads inside of Kubernetes.
Remember, Kubernetes is just running servers under the hood. Our regular old OS hardening and network protections apply.
Kubernetes can be secure, but it is far from default.
In Summary
The future of AppSec is a deeper understanding and goes far beyond the source code.
The challenge going forward will be balancing security with deployment efficiency.
A scanner or firewall won’t save you.
Take Home Assignment
Make friends
Beware of blind spots
Embrace a beginners mindset mindset
Adapt and evolve
Practice

kube-goat

https://github.com/ksoclabs/kube-goat

https://github.com/RhinoSecurityLabs/cloudgoat

https://www.owasp.org/index.php/OWASP_Serverless_Goat