
KubeCon
CloudNativeCon
North America 2019

Knative: The Security Platypus

Ariel Shuper,

VP Product, Portshift

Who Am I?

VP Product Management @**Portshift : Identity based runtime protection powered by Istio/service-mesh architecture**

Prior to **Portshift,** Sr. Director, Head of Serverless Security @**Aqua Security** (responsible for Aqua Security serverless offering, working on AWS Lambda, Azure Functions and Google Functions). Head of Public-Cloud security products **@Check Point**

Why we're here?

Platypus

Animal

The platypus, sometimes referred to as the duck-billed platypus, is a semiaquatic egg-laying mammal endemic to eastern Australia, including Tasmania. The platypus is the sole living representative of its family and genus, though a number of related species appear in the fossil record. Wikipedia

Class: Mammalia

Order: Monotremata

Family: Ornithorhynchidae

Scientific name: Ornithorhynchus anatinus

Lifespan: 17 years (In captivity)

Did you know: Experiments have shown that the Platypus will even react to an 'artificial shrimp' if a small electrical current is passed through it. animalcorner.co.uk

K-native: Quick Recap

Knative extends Kubernetes to provide a set of middleware components to build modern container-based applications…

Knative offers Kubernetes-native APIs for deploying serverless-style functions, applications, and containers to an auto-scaling runtime

~~Build~~	Serving	Eventing
~~Source to Container~~	Request-driven, scaling up/down, container-based compute	Attach work to event sources

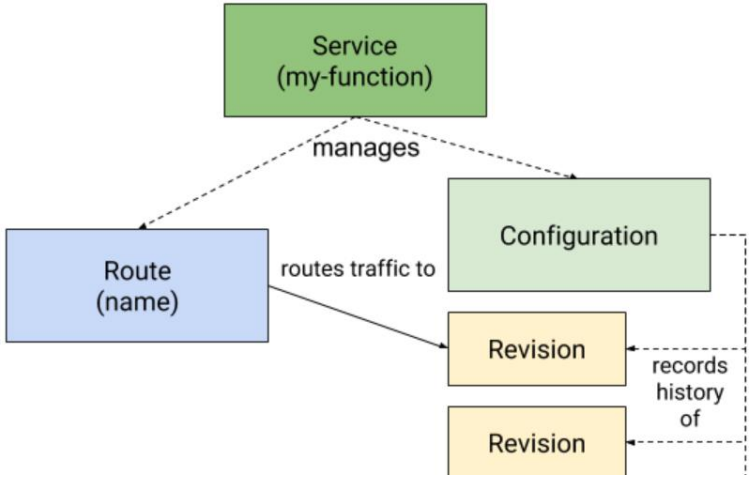
Knative Internals

Knative Serving

Stateless, HTTP request-driven,

container autoscaling platform

on top of **Kubernetes and Istio**

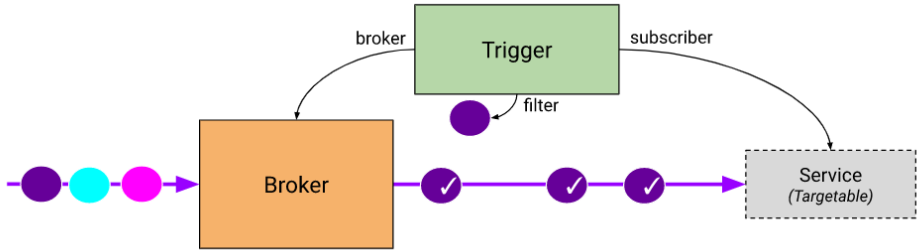


Knative Eventing

HTTP based events triggering system for

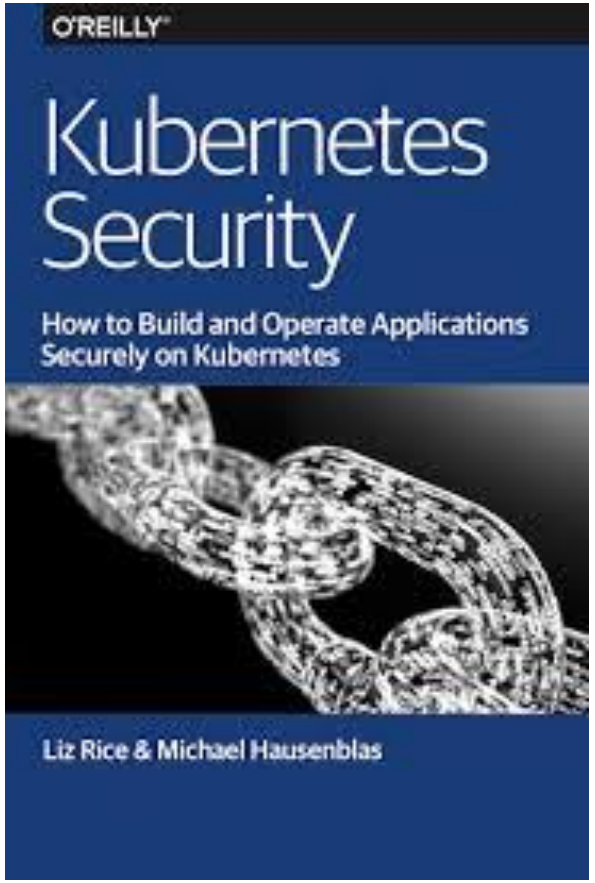
loosely coupled services on top of **Kubernetes**

and Istio



K-native: Security angle (a.k.a Why a Platypus?)

- Security aspects aren’t addresses independently in Knative
- But Knative uses Kubernetes and Istio:
  - Kubernetes security tools (RBAC, network policies, secrets etc,)
  - Istio adds on additional tools (traffic encryption, services authentication/authorization)



Serverless Security Landscape

- When Serverless (FaaS) was born it was perceived as the most secured cloud service:
  - **Ephemeral** (5-10min. max)
  - **No write access to the host filesystem**
  - **No fixed network address**
- but, serverless architecture has few security pitfalls
  - Application code
  - The Cloud infrastructure
- Security became a challenge: existing security tools were inadequate

Serverless Security Landscape

There are few serverless security frameworks

Notables ones:

- OWASP top-10 (Serverless interpretations)
- CSA top 12 critical risks

OWASP

OWASP Top 10 (2017)

Interpretation for Serverless

Serverless Security Landscape

CSA Top 12 Risks for Serverless Apps

SAS-1 : Function Event-Data Injection

SAS-2 : Broken Authentication

SAS-3 : Insecure Serverless Deployment Configuration

SAS-4 : Over-Privileged Function Permissions and Roles

SAS-5 : Inadequate Function Monitoring and Logging

SAS-6 : Insecure Third-Party Dependencies

SAS-7 : Insecure Application Secrets Storage

SAS-8 : Denial of Service and Financial Resource Exhaustion

SAS-9 : Serverless Business Logic Manipulation

SAS-10 : Improper Exception Handling and Verbose Error Messages

SAS-11: Legacy / Unused functions & cloud resources

SAS-12: Cross-Execution Data Persistency

Serverless Security Landscape (2)

A1	Code Injection
A2	Broken Authentication
A3	Sensitive Data Exposure
A4:	XML External Entities (XXE)
A5:	Broken Access Control
A6:	Security Misconfiguration
A7:	Cross-Site Scripting (XSS)
A8:	Insecure Deserialization
A9:	Using Components with Known Vulnerabilities
A10:	Insufficient Logging and Monitoring

OWASP Top 10 (2017)

Interpretation for Serverless

Serverless Security Landscape (3)

You can classify the “Serverless” risks arena into 2 classes

Your App Code

Code Injection

Vulnerable code

Sensitive Data exposure (secrets)

Cross-Site-Scripting (XSS)

Exception handling messages

Your Serverless Infra

Broken Authentication

Insecure Deployment configuration

Over privileged permissions

Inadequate Monitoring & Logging

Denial of Service/ Resources

Exhaustion

Demo: Serverless Attack

Demo is based on the OWASP Serverless_Goat vulnerable App

Code is available at: https://github.com/OWASP/Serverless-Goat

Knative: The Security Angle

Let’s examine Knative’s Security angle

Assumptions:

A. Kubernetes uses its security controls (Secrets, Network Policies, RBAC)

B. Istio is configured by Knative

Knative Security: Code Flaws

Code Injection -> remains a risk (but to a lower extent)

Vulnerable code ->Lower risk (vulnerable packages/dependencies inspection is in large usage defacto)

Sensitive Data exposure (secrets) -> Kubernetes secrets minimize its impact

Cross-Site-Scripting (XSS) -> can be more relevant and effective attack

Exception handling messages -> Less relevant

Knative Security: Serverless Infra

Broken Authentication -> can be solved by Istio mtls authentication

Insecure Deployment configuration -> Can be a real challenge

Over privileged permissions -> lower risk, can be easily solved

Inadequate Monitoring & Logging -> lots of monitoring & logging options

Denial of Service/ Resources Exhaustion

ISTIO-SECURITY-2019-006: DoS affecting Istio 1.3.x versions

Security

Francois 10d

The Istio Product Security Committee would like to inform you that a vulnerability affecting all Istio versions released after 1.3 (included) has been discovered. Note that the 1.4-alpha and 1.4-beta releases are also affected.

This vulnerability has been discussed publicly as a “high CPU” or “100% CPU” bug, and as such is considered a 0-day vulnerability.

As we are working on a code fix to address this issue, we would like to share an existing workaround. The exploitation of that vulnerability can be prevented by customizing your Istio install (as described in https://istio.io/docs/reference/config/installation-options/#pilot-options 10), using Helm to override the following options:

```
--set pilot.env.PILOT_INBOUND_PROTOCOL_DETECTION_TIMEOUT=0s --set global.proxy.prot
```

What About Knative Architecture?

- Is Knative architecture secured?
- Can Istio simplify the security challenges of Knative (hint: w/o cold start impact) ?

Knative Architecture: More worries

Knative Architecture adds few security challenges

- Eventing:
  - Knative's eventing does not perform or configure additional security controls beyond the underlying Kubernetes cluster (Unauthorized events subscription, false events injection)
  - Cloud events are loosely coupled using different platforms (VMs, Containers, SaaS, FaaS) making the authentication/authorization even more challenging

Knative Security: Istio to the rescure?

- Istio can mitigate lots of the Knative security challenges
  - Using granular Identities the authentication/authorization challenges in Eventing can be solved
  - Traffic controls options can mitigate the DoS/Service Exhaustion
  - Validation of deployments configuation

But: Istio has performance considerations: first invocations (cold-calls initializations) takes long times…

Summary

- Knative is the Kuberntes Serverless platform
- Knative Security is like a Platypus:
  - Lots of the classical serverless risks are solved
  - Some risks are relevant and some risks are unique
- Istio can mitigate most of the challenges
  - But it has performance impact (cold calls)

Thank You!

KubeCon | CloudNativeCon

North America 2019

ariel@Portshift.io

@ArielShuper

Visit us at booth CE30 Startup pavilion