Securing Microservice Interactions in Openstack and Kubernetes

Yoshio Turner & Jayanth Gummaraju
Co-Founders @ Banyan
https://www.banyanops.com
Banyan

• Founded in the middle of 2015
  – In San Francisco, CA

• Veterans from VMware, HP Labs, and Moovweb
  – 50+ patents and publications in Virtualization, Network Security, and Big Data

• Incubated at Stanford’s StartX accelerator
  – Currently our product is in private beta
Outline

1. Introduction to Microservices
2. Dynamic and sprawling attack surface
3. Frankensteining existing solutions
4. A New approach to Application Security
5. Demo
6. Conclusions & discussion
Outline

1. Introduction to Microservices
2. Dynamic and sprawling attack surface
3. Frankensteining existing solutions
4. A New approach to Application Security
5. Demo
6. Conclusions & discussion
Transitioning from Monoliths to Microservices
Benefits of Microservice architecture

• People
  – Allows different teams to build single-purpose application components independently

• Process
  – Enables fast deployment (months -> days/hours) from development to production

• Technology
  – Each team can independently evolve their own development and deployment stacks

* Caveat: Not all applications fit this architecture *
Microservice architectures are getting complex
Outline

1. Introduction to Microservices

2. Dynamic and sprawling attack surface

3. Frankensteining existing solutions

4. A New approach to Application Security

5. Demo

6. Conclusions & discussion
Securing monoliths

AppSec
- API Gateways
- TLS Proxies
- App Firewalls/RASP

NetSec
- Segmentation
- Network Firewalls/Gateways
- IPS/IDS systems
New Attack Surface is Dynamic and Sprawling

Rapid deployments
More exposed APIs

New communication channels
Untrusted apps & infrastructure
Major IT Trends Put Data at Risk

1. Cloud-native and microservice architectures
   - Function calls change to network traffic: intra-host to cross-cloud

2. Agile deployments using containers
   - New privileged components (e.g., orchestration engines)

3. Public, multi-, and hybrid clouds
   - Lack of global identities, visibility and control
Security Evolution

- **AppSec**
  - 1990: Physical Appliances
  - 2000: Virtual Appliances
  - 2010: SMDs, RASP

- **NetSec**
  - 1990: Network Appliances
  - 2000: Overlays, VPCs, SGs
  - 2010: SDN, Micro-segments

- **Applications**
  - 1990: Monoliths
  - 2000: Three-tier
  - 2010: Microservices
    - Functions

- **Cloud**
  - 1990: Physical
  - 2000: Private
  - 2010: Public
    - Multi / Hybrid

- **Compute**
  - 1990: Physical
  - 2000: Virtual
  - 2010: Containers
    - Server-less

- **BANYAN**
  - Proprietary & Confidential
Outline

1. Introduction to Microservices
2. Dynamic and sprawling attack surface
3. Frankensteining existing solutions
4. A New approach to Application Security
5. Demo
6. Conclusions & discussion
Requirements for a Microservices Security Solution

• Addresses new threats
  – Dynamically adapt to the ever-changing workloads
  – Minimal trust in application and infrastructure
  – Distributed and pervasive across clouds

• Works across traditional and modern environments
  – Microservices and traditional applications need to interact
  – Independent of underlying infrastructure
  – Works in polyglot environments

• Ease of use
  – Application-level abstraction, rather than low-level constructs
  – Single pane visibility and control
  – “Set it and forget it” security
Openstack tools

• NetSec
  – Neutron network topologies: provider and tenant networks
  – Virtual routing & firewall-as-a-service: reachability between networks
  – Security Groups: per VIF ingress/egress rulesets based on ip_proto/CIDR:portrange or remote_group_id
  – Role based access control (RBAC): enable different projects to share access to Neutron resources

• AppSec
  – Barbican: secrets store, certificate authority, HSM integration
  – LBaaS plugins: L7 policies using load balancer virtual appliances
Kubernetes tools

- **NetSec**
  - Network policy: reachability based on pod labels and values
  - Segmentation: provided by some CNI plugins like OpenContrail, Calico

- **AppSec**
  - Kubernetes secrets management
  - Discussions on service policies
Kubernetes and Openstack

• Deployed Side-by-side
  – Separate OpenStack cluster hosting Nova instances
  – Separate Kubernetes cluster, possibly bare metal

• K8s deployed on OpenStack nova instances
  – Run kube-up.sh for OpenStack – based on OpenStack Heat
  – OpenStack Magnum
Kubernetes & OpenStack networking options (over-simplified)

- Independent network frameworks for each cluster
  - OpenStack services exposed through floating IPs, etc.
  - K8s services exposed through external load balancer, or nodeport (static port on all k8s nodes)

- Project Kuryr: Neutron networks spanning OpenStack & Kubernetes
  - Kuryr controller detects K8s events & allocates Neutron resources
  - Kuryr CNI plugin on each K8s node attaches Neutron resources to K8s pods, integrates with Neutron driver, e.g., openvswitch
  - Result
    - Neutron networks spanning OS & K8s
    - OpenStack security groups on K8s pods
    - OpenStack LBaaS, FWaaS, etc. for k8s services
Kubernetes and Openstack AppSec

• Secrets
  – Separate frameworks for now
  – Even if federated, secrets are problematic
    • Burdens developers and operators
    • May rule out some programming languages or frameworks

• L7 authorization
  – Possible in load balancing solutions
  – Security concerns
    • Lack of application context
    • Requires strict networking controls to avoid evasion
Solving for the General Case
Outline

1. Introduction to Microservices
2. Dynamic and sprawling attack surface
3. Frankensteining existing solutions
4. A New approach to Application Security
5. Demo
6. Conclusions & discussion
Security Micro-engines

- AppSec
  - API Gateways
  - TLS Proxies
  - App Firewalls/RASP

- NetSec
  - Segmentation
  - Network Firewalls/Gateways
  - IPS/IDS systems

- On-Prem
  - sql

- Internet

- AWS
- Azure
- kafka

BANYAN
Security Evolution: Convergence of AppSec & NetSec

- **AppSec**
  - 1990: Physical Appliances
  - 2000: Virtual Appliances
  - 2010: SDKs, RASP
  - 2020: Security Micro-engines

- **NetSec**
  - 1990: Network Appliances
  - 2000: Overlays, VPCs, SGs
  - 2010: SDN, Micro-segments
  - 2020: Security Micro-engines

- **Applications**
  - 1990: Monoliths
  - 2000: Three-tier
  - 2010: Microservices
  - 2020: Functions

- **Cloud**
  - 1990: Physical
  - 2000: Private
  - 2010: Public
  - 2020: Multi / Hybrid

- **Compute**
  - 1990: Physical
  - 2000: Virtual
  - 2010: Containers
  - 2020: Serverless
Introducing Banyan Security Micro-engines Platform

Securely bridges traditional and modern IT

BANYAN PLATFORM
Centralized Policy Mgmt
Distributed Control Plane
Transparent Enforcement
Cryptovisor: Security Virtualization Layer

App-transparent and Network-independent
Example application using microservices

- Web browser
- Mobile app
- TLS Proxy
- API Gateway
- Web portal
  - Payments service (mysql)
  - User service (kafka)
  - Notifications service (redis)
- Catalog service
- Perimeter Security
Securing inter-service communications

Least privilege access & encryption in-transit

- web browser
- mobile app
- web portal
- payments service
- user service
- notifications service
- mysql
- kafka
- redis

**Perimeter Security**

BANYAN
Key attributes

1. **Transparent security layer**
   - No changes to application or infrastructure
   - Works across traditional and modern IT

2. **Global service identity**
   - Cryptography-based (not IP/Port or username/passwd)
   - Cross-cluster, Cross-cloud interoperability

3. **High-performance at the desired level of security**
   - Invoke security functions tailored to each service
   - Set policies using ABAC and RBAC
Security Virtualization Technology

- **vHost S1**
  - Cryptovisor
  - Network Stack

- **vHost S2**
  - Cryptovisor
  - Network Stack

**Dashboard**
- Transparent TLS Encryption
- Identity & L7 Access Control
- Real-time Flow Visibility

**Connection**
- **CI**
- End-To-End mTLS

**Unique Cryptographic Identity**

**Middle boxes**
Outline

1. Introduction to Microservices
2. Dynamic and sprawling attack surface
3. Frankensteining existing solutions
4. A New approach to Application Security
5. Demo
6. Conclusions & discussion
Banyan demo video
Outline

1. Introduction to Microservices
2. Dynamic and sprawling attack surface
3. Frankensteining existing solutions
4. A New approach to Application Security
5. Demo
6. Conclusions & discussion
Conclusions and Discussion

• Microservices, Containers, and Hybrid Clouds open up a new attack surface

• Existing security solutions are not designed for such dynamic environments

• New solutions are needed that are:
  – Tailored to microservice architectures
  – Operate at the application layer for identity and access control
  – Works across traditional and modern environments

• Security virtualization technology brings together AppSec and NetSec to protect both modern and traditional applications
Thank You

Yoshio Turner: yoshio@banyanops.com
Jayanth Gummaraju: jayanth@banyanops.com

https://www.banyanops.com