



CYBERLOOP

MICROSERVICES 2020

TOWARDS MICROSERVICES
SECURITY CONTROL LAYER

CYBERLOOP

SPEAKERS



Alessandro Molari
Cyber Security Senior Advisor
alessandro.molari@cyberloop.it



Eugenio Cavina
Cyber Security Consultant
eugenio.cavina@cyberloop.it



Eugenio Pierfederici
Cyber Security Consultant
eugenio.pierfederici@cyberloop.it



Cyber Security Consultancy company

- modern approach to cyber security:
[operational](#) «real» [security](#) (compliance is not enough)
- follows [secure-by-design](#) approach: it proposes the same security measures followed by Cyberloop itself
- follows [security-in-depth](#) approach: layered security considering all factors cyber, physical and human (incl. psychological)
- promotes [security-as-a-process](#) based on methodological approach and international standards

INFO@CYBERLOOP.IT

LINKEDIN.COM/COMPANY/CYBERLOOP



INDEX

1. CONTEXT: CONCERNS, ISSUES, GOALS
2. SECURITY CONTROL PLANE
3. USE CASE: INCIDENT MITIGATION

CONTEXT

CONCERNS, ISSUES, GOALS

CONCERNS

(SOME) SECURITY CONCERNS IN MICROSERVICES ECOSYSTEMS

From what we've seen, main security concerns about cybersecurity in microservices are:

- **Heterogeneity**: different languages, different toolchains
- **Observability**: effective incident mitigation needs rapid detection
- **Governance**: difficult to manage specific security needs in all microservices without logically centralizing it (even if physically distributed)
- **Skillgap**: proper cybersecurity needs specialists

ISSUES

(SOME) ISSUES WITH "LOCAL" APPROACH

- Microservices do one thing and do it well
 - Security inside a microservice is limited to local microservice scope
 - This way, it's difficult to define global policies or relationships policies
- Many microservices may be difficult to govern
 - many different implementations
 - complex set of relationships difficult to handle
 - different people handling same kind of security aspects
 - fragmentated registries
 - expensive and difficult patching and updates
- By default, security is hard to manage and easy to lose track

GOALS

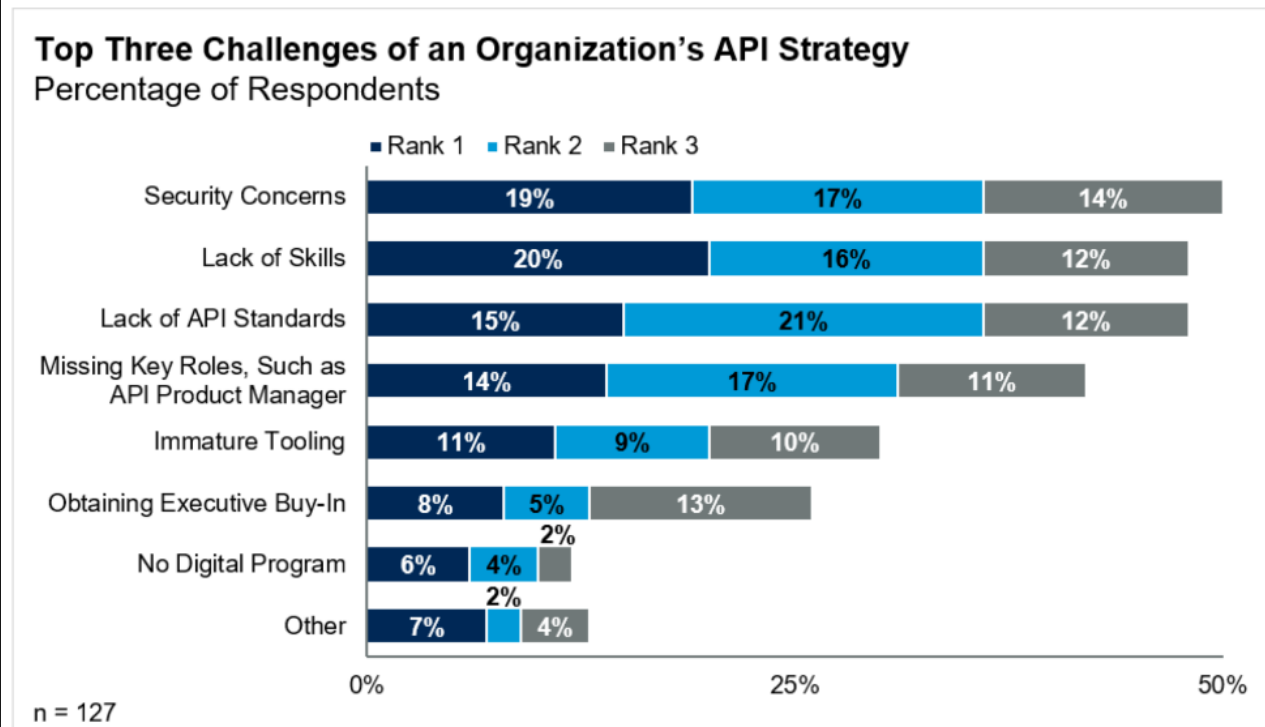
(SOME) NEEDS TO EFFECTIVELY INTRODUCE CYBERSECURITY IN MICROSERVICES ECOSYSTEMS

- Need to treat cybersecurity orthogonally to business logic (at least, in analysis and design phases)
- Need of a common security strategy for heterogeneous microservices, independently from technologies/vendors
- Need of a common framework to define security policies decoupled from microservices logic
- Need to manage/govern cybersecurity from a single logical point, without touching implementations
- Need to allocate cyber professionals for specific tasks, which may know not much about microservices (and vice versa)

ENTERPRISES PERSPECTIVE

API SECURITY ISSUES, SOMEHOW RELATED TO MICROSERVICES

Figure 1. Top Three Challenges of an Organization's API Strategy



"[...] However, the benefits which APIs bring in opening access to data and application functionality naturally also bring security concerns. Already, many API security incidents have occurred, particularly in the form of data leaks [...]"



Reflecting this, Gartner has noted a 30% year-on-year increase in client inquiries related to API security. Furthermore, Gartner's survey [...] found that API security ranked in the top three challenges to API strategy for 50% of respondents, followed by lack of skills and lack of API standards "

Source: Gartner, 2018

ENTERPRISES PERSPECTIVE

API SECURITY ISSUES, SOMEHOW RELATED TO MICROSERVICES

API Security Consists of API Protection and API Access Control

	 API Threat Protection	 API Access Control
Key functionality	Content validation, threat detection, traffic throttling	Authentication, authorization, identity propagation
Key technologies used	Attack signature, reputation-based control, anomaly detection, OAS message validation	OAuth 2.0, OpenID Connect, JSON Web Tokens
Product categories	Web application firewalls, API management, application delivery controllers	API management, access management software, IDaaS.

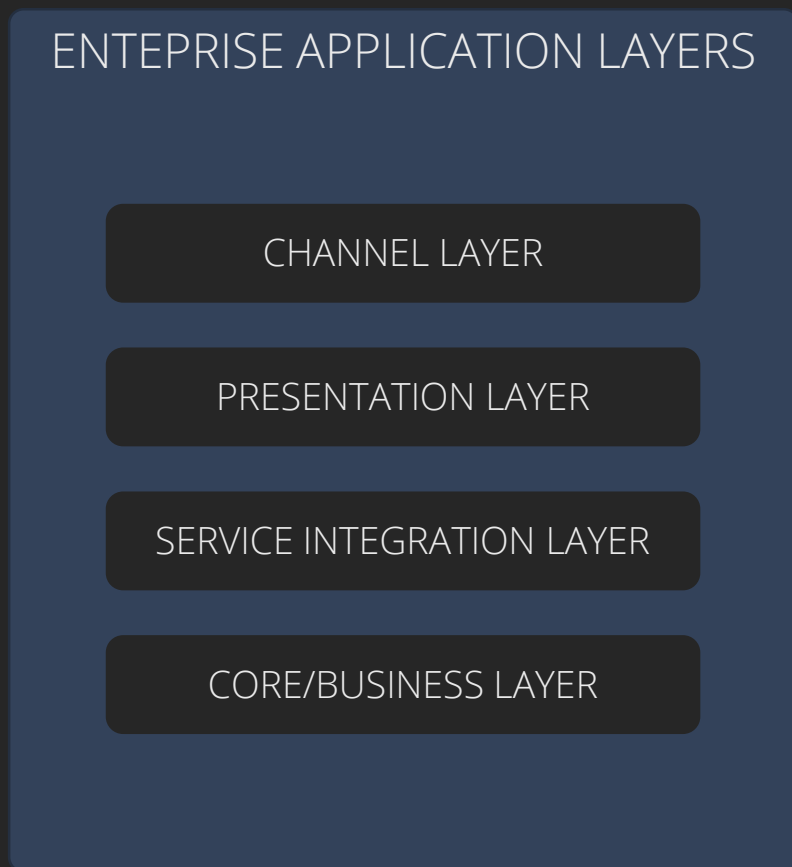
“[...] API security can be divided into two broad aspects: API threat protection and API access control. API threat protection means detecting and blocking attacks on APIs, while API access control means controlling which applications and users can access APIs. Organizations need both. [...]”

Source: Gartner, 2018

SECURITY CONTROL PLANE

SECURITY CONTROL PLANE

LAYERED AND CENTRALIZED APPROACH



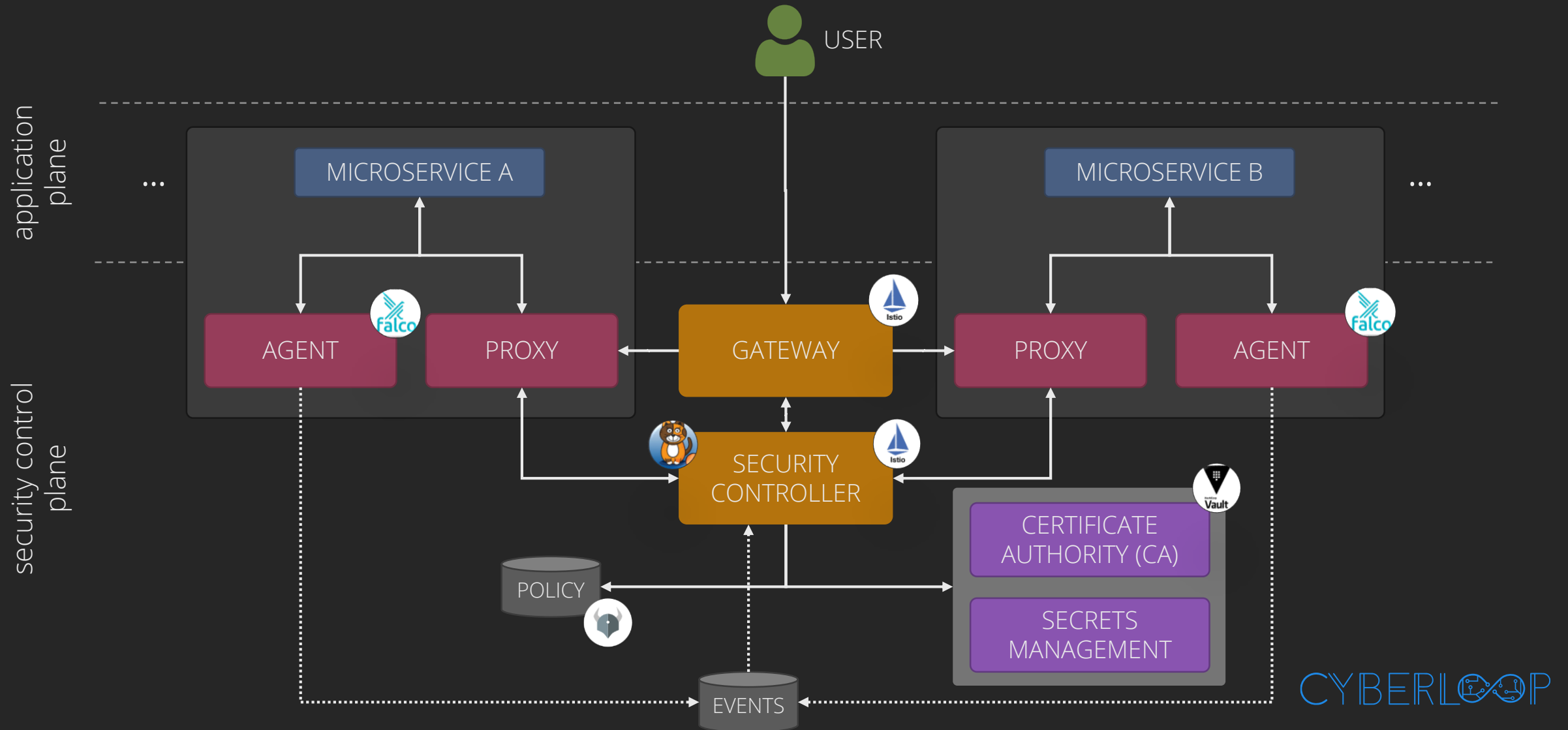
SECURITY CONTROL LAYER

Features:

- Service-to-service security and authentic. / authoriz.
- Microservice runtime protection
- Malicious behavior detection & mitigation
- Global security policies and distributed evaluation
 - Traffic routing security policies
 - Endpoint security policies
- Container isolation
- Secrets/certificates management with ACLs
- Continuous pro-active (agent) monitoring
- Anomaly detection
- Mitigation policies

SECURITY CONTROL PLANE

ARCHITECTURE



ADVANTAGES

- Cross-layer security, orthogonal
- Secure-by-design approach
 - by default, least privilege to microservices
- Logically centralized, single place to govern and version
 - can apply mitigation actions in a single place (e.g., policy blocking traffic), without touching the microservice
- Rapid response, incident mitigation as first-class citizen

USE CASE

INCIDENT MITIGATION

INCIDENT MITIGATION

USE CASE

Let's consider a security incident happening to a particular microservice: it has been compromised.

We can apply two types of mitigation strategies:

- Endpoint Mitigation
- Network Mitigation

INCIDENT MITIGATION

We need to apply the **incident triage**:

1. Understand that an incident is happening
2. Find out:
 - which microservice is compromised
 - any communications / lateral movements
3. Understand the level of compromise
4. Apply mitigations

Some mitigation actions could involve:

1. Prevent further compromises by blocking network traffic with all other microservices
2. (if possible) Block other attacks on the target with endpoint rules

NETWORK MITIGATION

Container with a microservice is compromised:
we want to prevent infection to extend to neighbors

There is no need to find the container IP address and
apply any firewall rule:
traffic is dropped by the security control plane.

This also allows to apply mitigations:

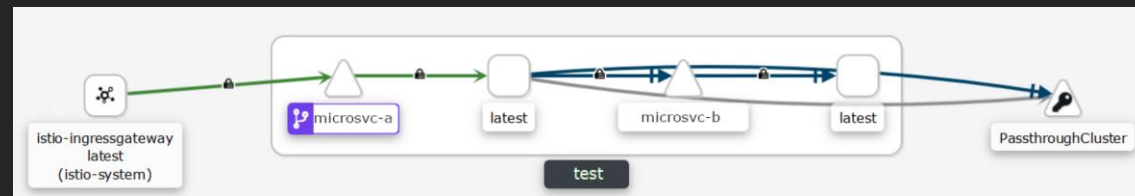
- Declaratively
- Vendor independent
- In a single place

```
...  
kind: AuthorizationPolicy  
...  
spec:  
  selector:  
    matchLabels:  
      app: MICROSERVICE-A  
  action: DENY  
  rules:  
  - {}
```

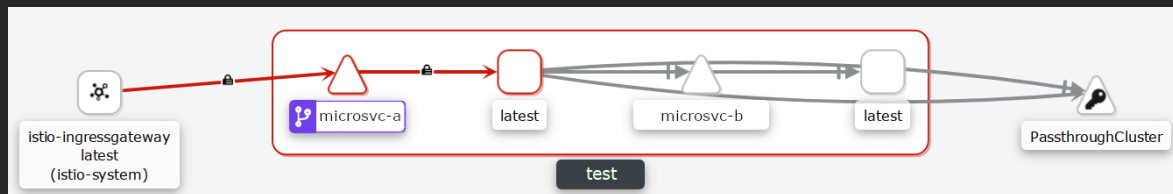
NETWORK MITIGATION

EXAMPLE

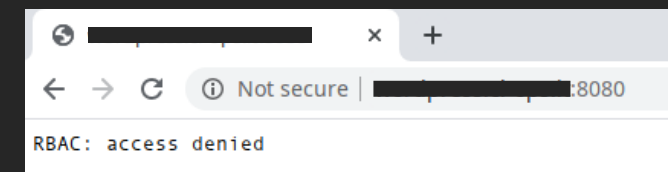
before mitigation actions:
network traffic is allowed



after mitigation actions:
network traffic is blocked



```
...  
kind: AuthorizationPolicy  
...  
spec:  
  selector:  
    matchLabels:  
      app: MICROSVCA  
  action: DENY  
  rules:  
  - {}
```



ENDPOINT MITIGATION

Microservice endpoint has well-known behavior:
expected I/O operations are defined and can be described.

Applying endpoint mitigation is easier in a microservices environment.

ENDPOINT MITIGATION

SOME EXAMPLES

A shell is run (could mean backdoor)

```
11:09:22.526653952: Debug Shell spawned by untrusted binary (user=www-data shell=sh parent=apache2 cmdline=sh -c ls pcm  
dline=apache2 -DFOREGROUND gparent=apache2 ggparent=<NA> aname[4]=<NA> aname[5]=<NA> aname[6]=<NA> aname[7]=<NA> contai  
ner_id=a063cc49571b image=wordpress) k8s.ns=wordpress k8s.pod=wordpress-65f86bbf6f-9s726 container=a063cc49571b k8s.ns=  
wordpress k8s.pod=wordpress-65f86bbf6f-9s726 container=a063cc49571b
```

Unexpected outbound connection (could mean lateral movement to another microservice)

```
12:19:48.158207641: Error File below known binary directory renamed/removed (user=root command=rm /bin/rootkit.sh /dev/  
rootkit pcmdline=bash operation=unlinkat file=<NA> res=0 dirfd=-100(AT_FDCWD) name=/bin/rootkit.sh flags=0 container_i  
d=a063cc49571b image=wordpress) k8s.ns=wordpress k8s.pod=wordpress-65f86bbf6f-9s726 container=a063cc49571b k8s.ns=wordp  
ress k8s.pod=wordpress-65f86bbf6f-9s726 container=a063cc49571b
```

Write operation to system directory (could mean later stage of infection, e.g. ransomware)

```
12:18:51.817177123: Error File created below /dev by untrusted program (user=root command=cp /bin/rootkit.sh /dev/rootk  
it file=/dev/rootkit container_id=a063cc49571b image=wordpress) k8s.ns=wordpress k8s.pod=wordpress-65f86bbf6f-9s726 con  
tainer=a063cc49571b k8s.ns=wordpress k8s.pod=wordpress-65f86bbf6f-9s726 container=a063cc49571b
```

Unexpected process is spawned (could mean malware persistence)

```
12:16:08.612510861: Error File below a known binary directory opened for writing (user=root command=touch /bin/rootkit.  
sh file=/bin/rootkit.sh parent=bash pcmdline=bash gparent=sh container_id=a063cc49571b image=wordpress) k8s.ns=wordpres  
s k8s.pod=wordpress-65f86bbf6f-9s726 container=a063cc49571b k8s.ns=wordpress k8s.pod=wordpress-65f86bbf6f-9s726 contain  
er=a063cc49571b
```



CYBERLOOP

MICROSERVICES 2020

TOWARDS MICROSERVICES
SECURITY CONTROL LAYER

CYBERLOOP

EMAIL: info@cyberloop.it
HQ: Cesena, Italy

SPEAKERS

alessandro.molari@cyberloop.it
eugenio.cavina@cyberloop.it
eugenio.pierfederici@cyberloop.it