How We Used Kubernetes to Host a CTF Competition

Liron Levin
Ariel Zelivansky

Who we are

- Ariel Zelivansky / Security Research Lead
  - Vulnerability research on open source projects, CVEs & blog
  - Best security practices for Twistlock platform

- Liron Levin / Chief Architect
  - Ph.D. on distributed network algorithms BGU
  - Designs and builds Twistlock platform

Agenda

1. What is a CTF
2. Why K8S
3. Engineering
4. Securing the infrastructure
5. Results
6. Key takeouts

What's a CTF?

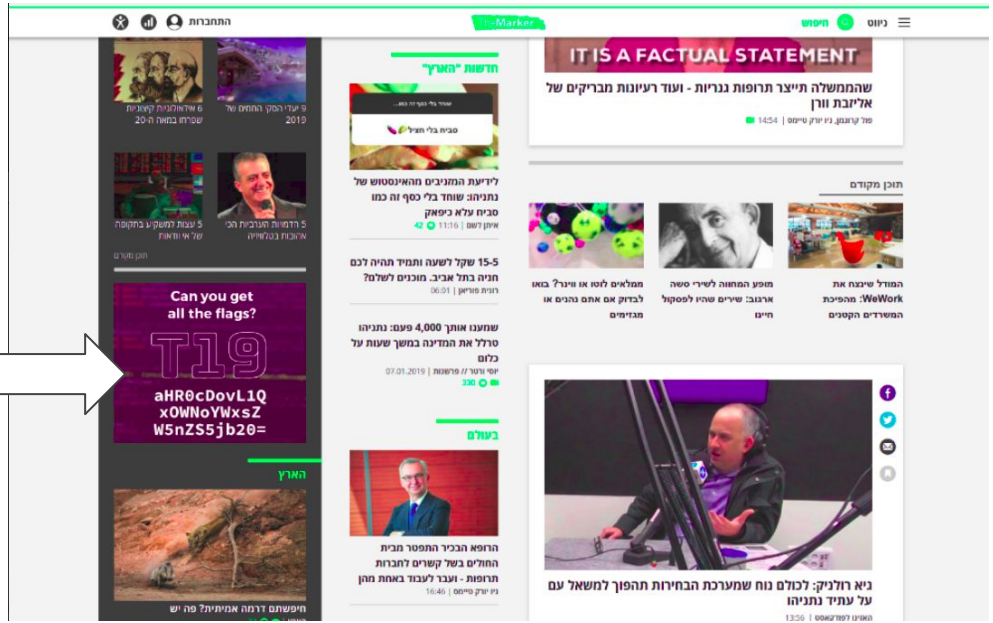
- "Capture the flag" challenge
  - Jeopardy style/Attack defense/Wargames (OTW)
- Good for education, conventions

Twistlock CTF - Why?

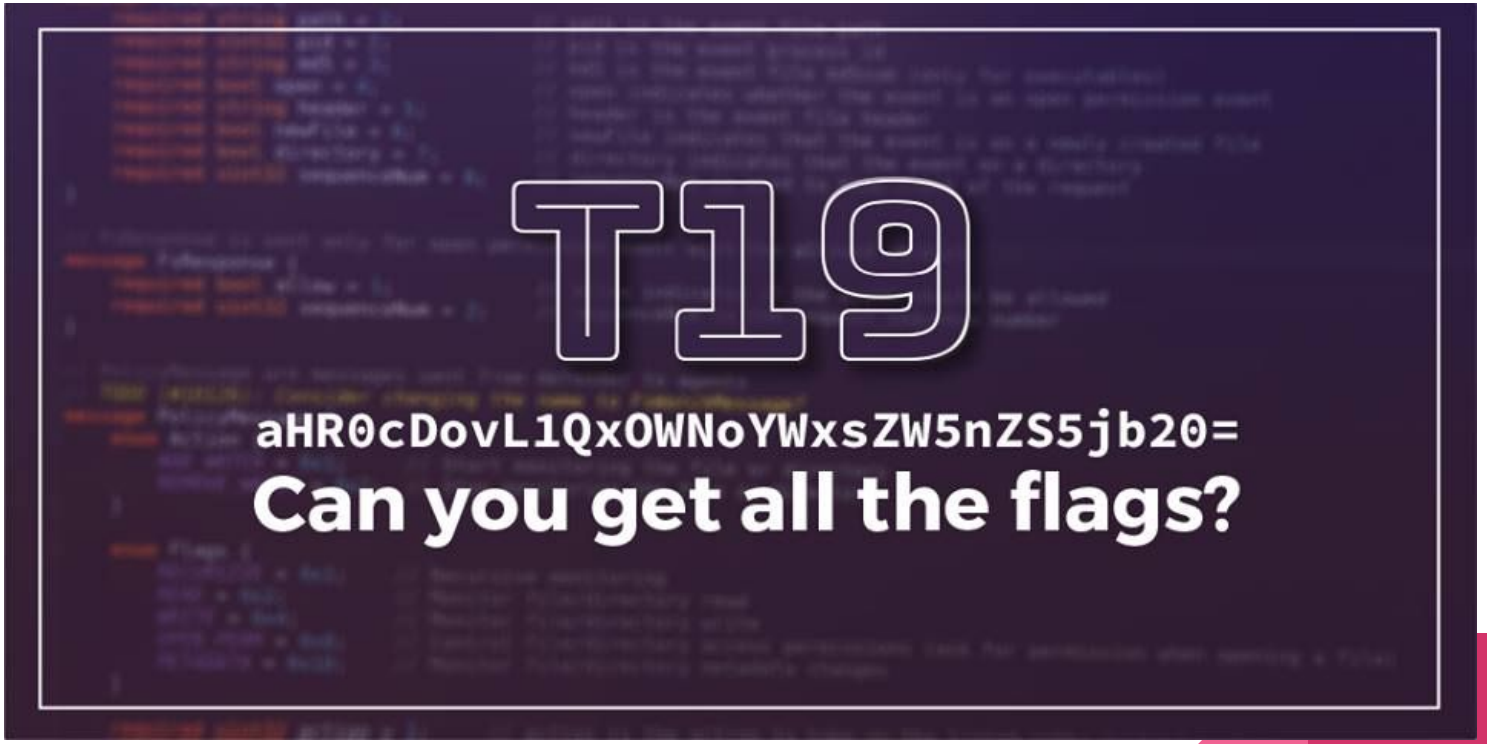
- Find good security researchers
- Creating challenges forces us to learn a lot
- Fun!

Advertised!

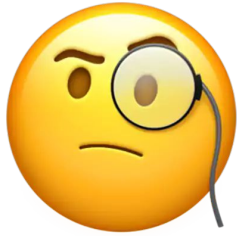
- Reddit for CTFs (securityCTF)
- Local news sites
- Facebook/Whatsapp groups



Advertised!

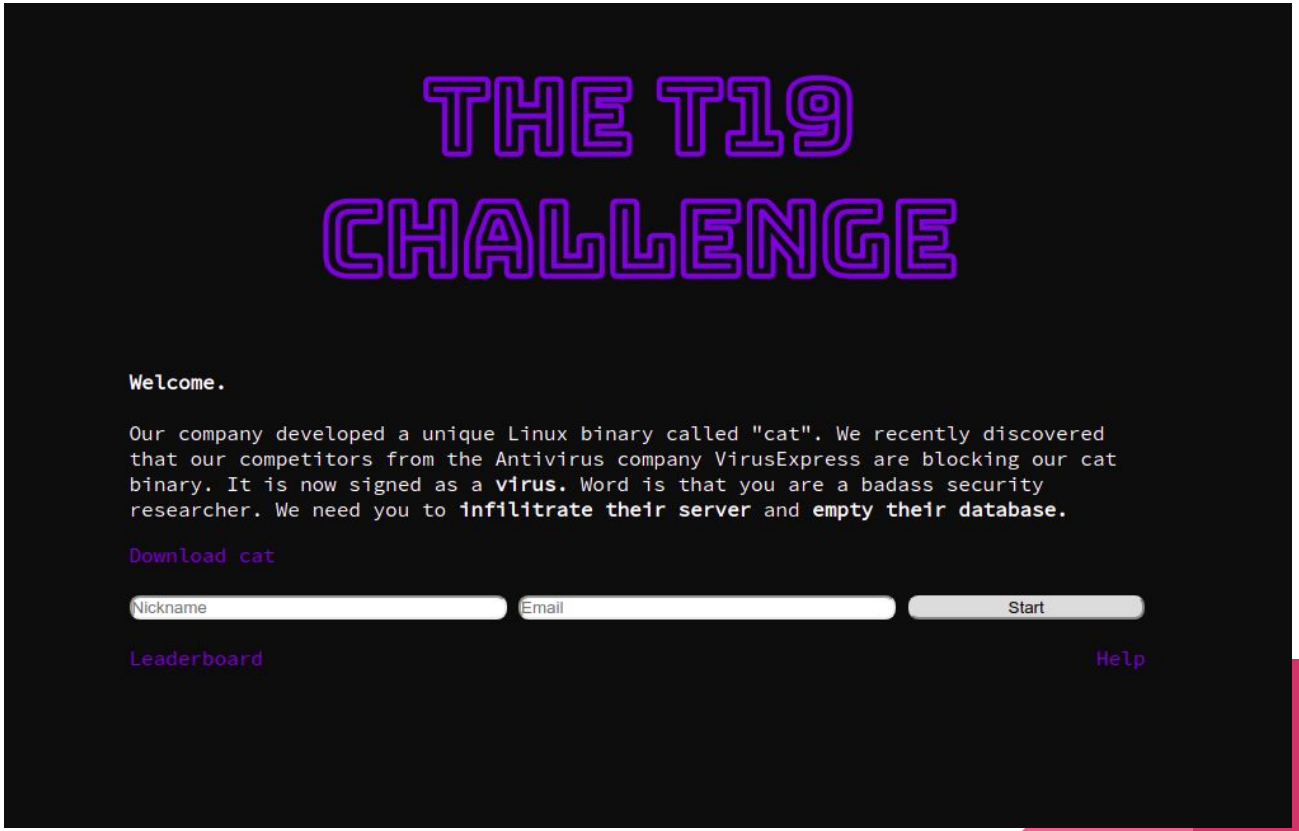

T19
aHR0cDovL1QxOWNoYWxsZW5nZS5jb20=
Can you get all the flags?

Making it interesting



- Wargame style
- Same machine - multiple challenges!
  - Different users, need to **escalate permissions**
  - Flags hidden as files
- Different challenge subjects - web/scripting, reverse-engineering, Linux internals, modern exploitation…

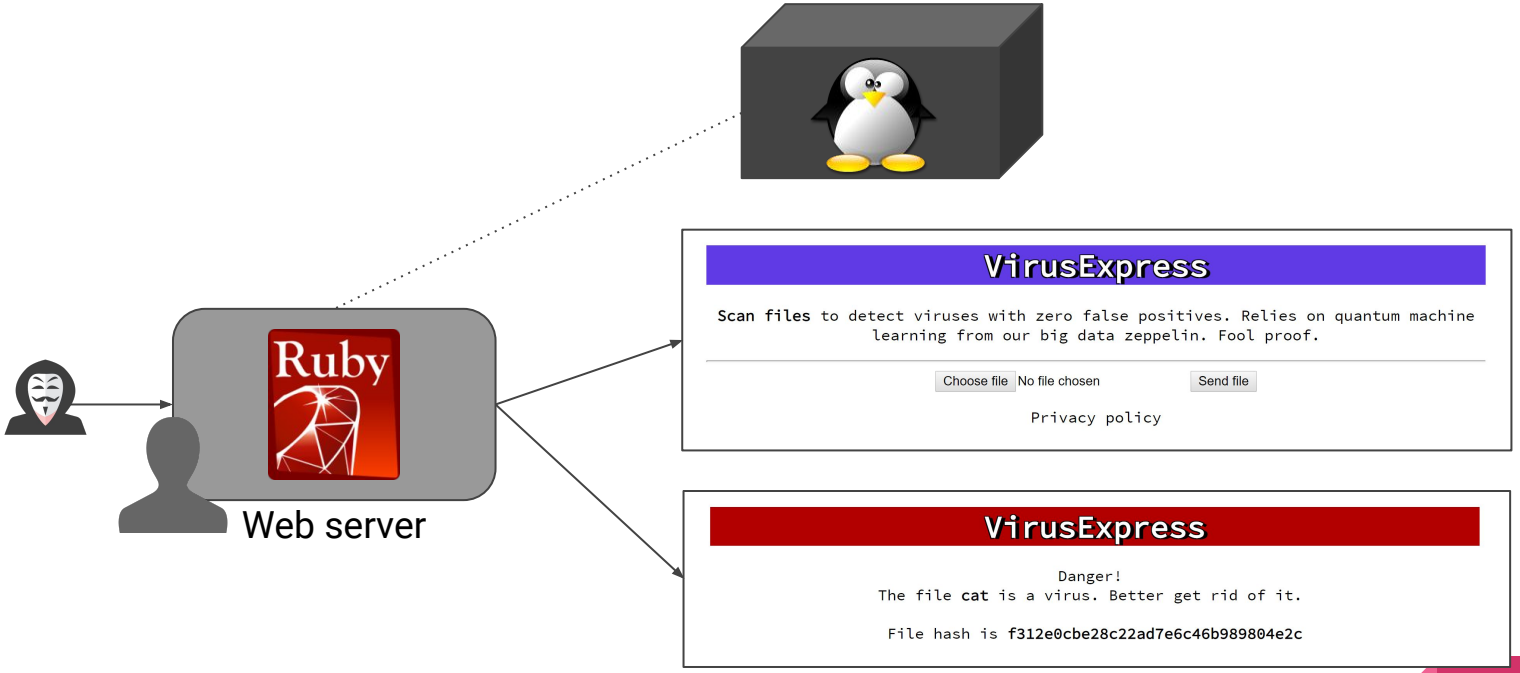
The challenge



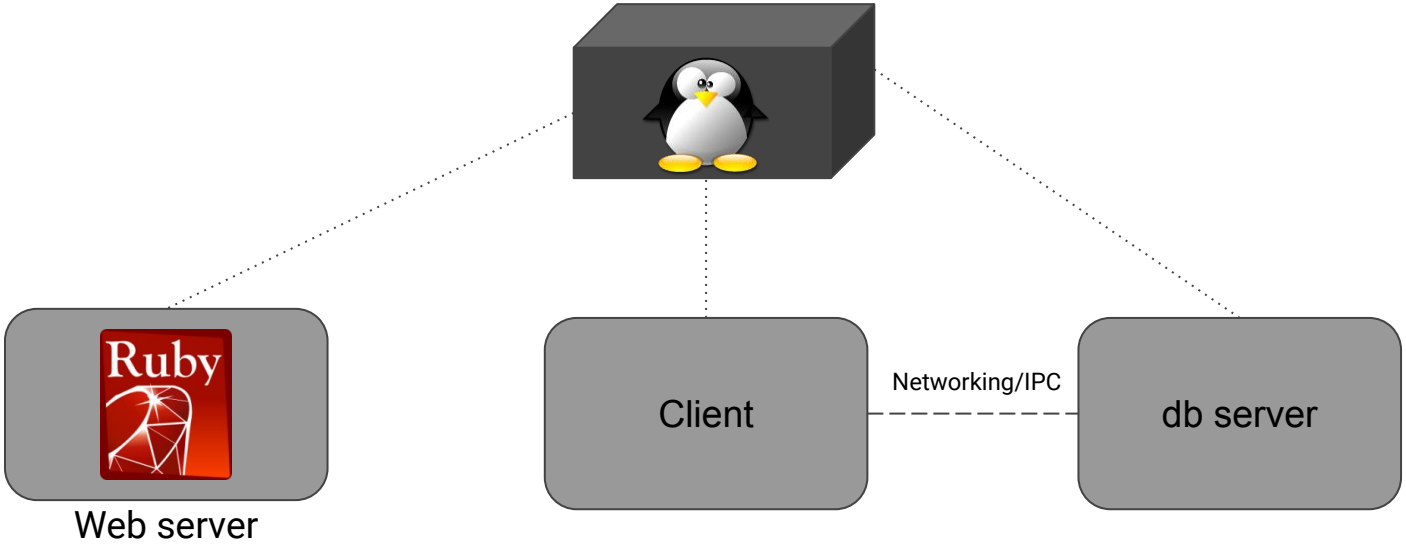
The challenge

```
ariel@ariel-ThinkPad-T470p:/tmp/app_cat$ ./cat
  /\___/\
 ( ~   ~ ) zzZz
 (  =^=  )
 (       )
 (        )
 (          ))))))))))
ariel@ariel-ThinkPad-T470p:/tmp/app_cat$ ./cat hello
  /\___/\
 ( o   o )
 (  =^=  )
 (       )
 (        )
 (          ))))))))))) meow
ariel@ariel-ThinkPad-T470p:/tmp/app_cat$ ./cat hello world
  /\___/\
 ( o   o )
 (  =^=  )
 (       )
 (        )
 (          ))))))))))) meow meow
ariel@ariel-ThinkPad-T470p:/tmp/app_cat$
```

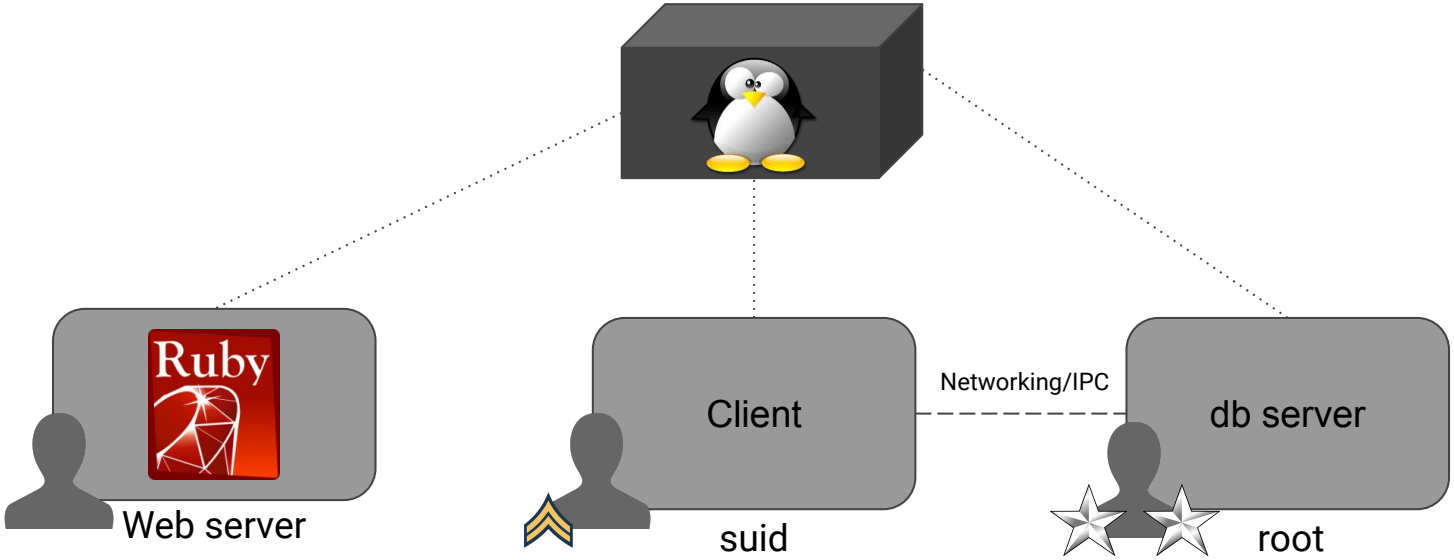
The challenge



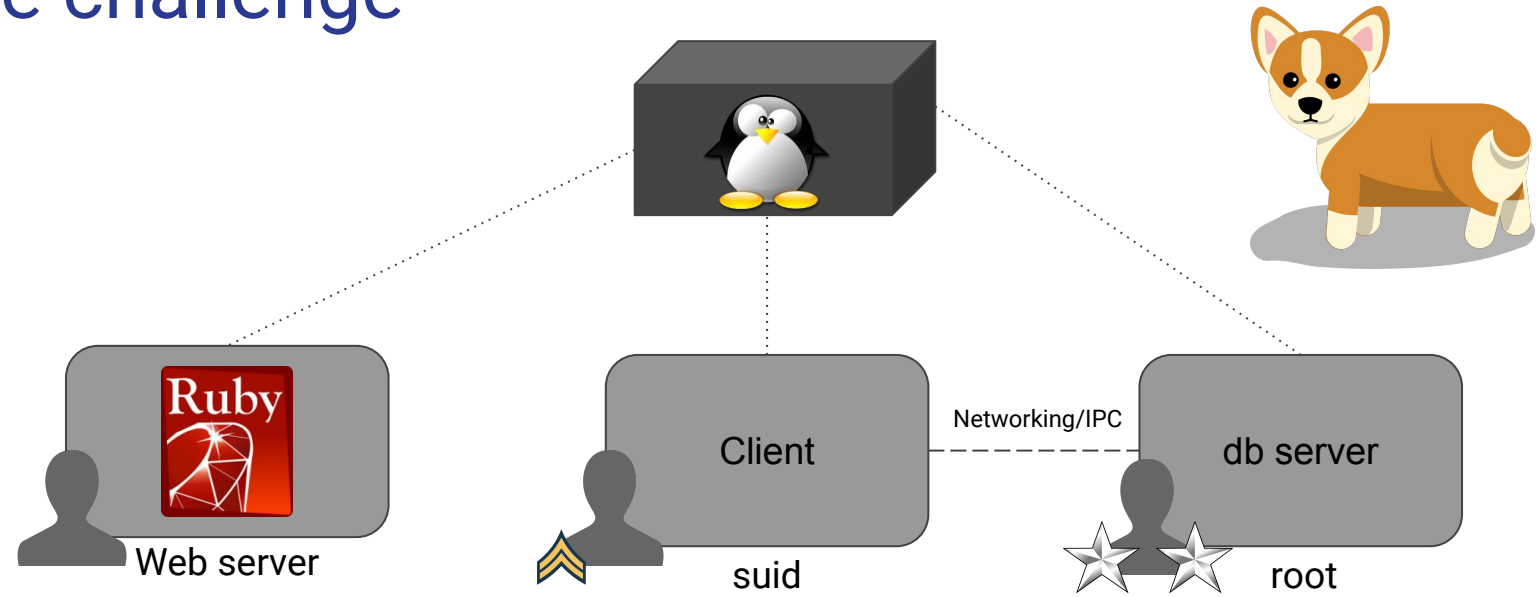
The challenge



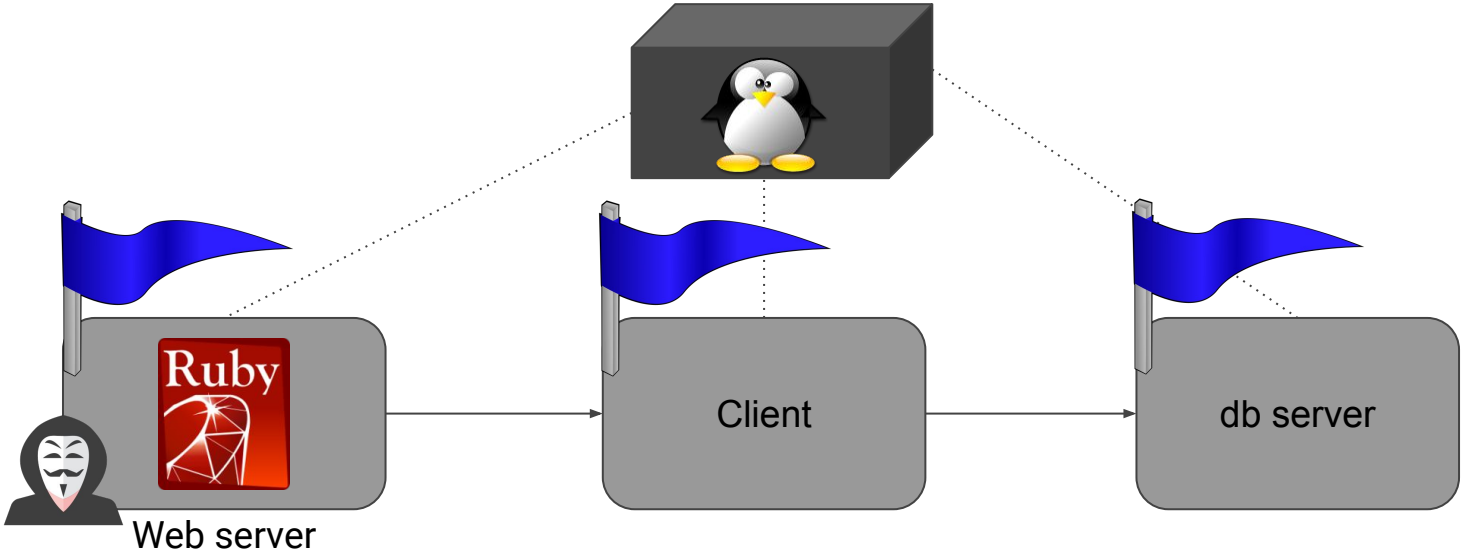
The challenge



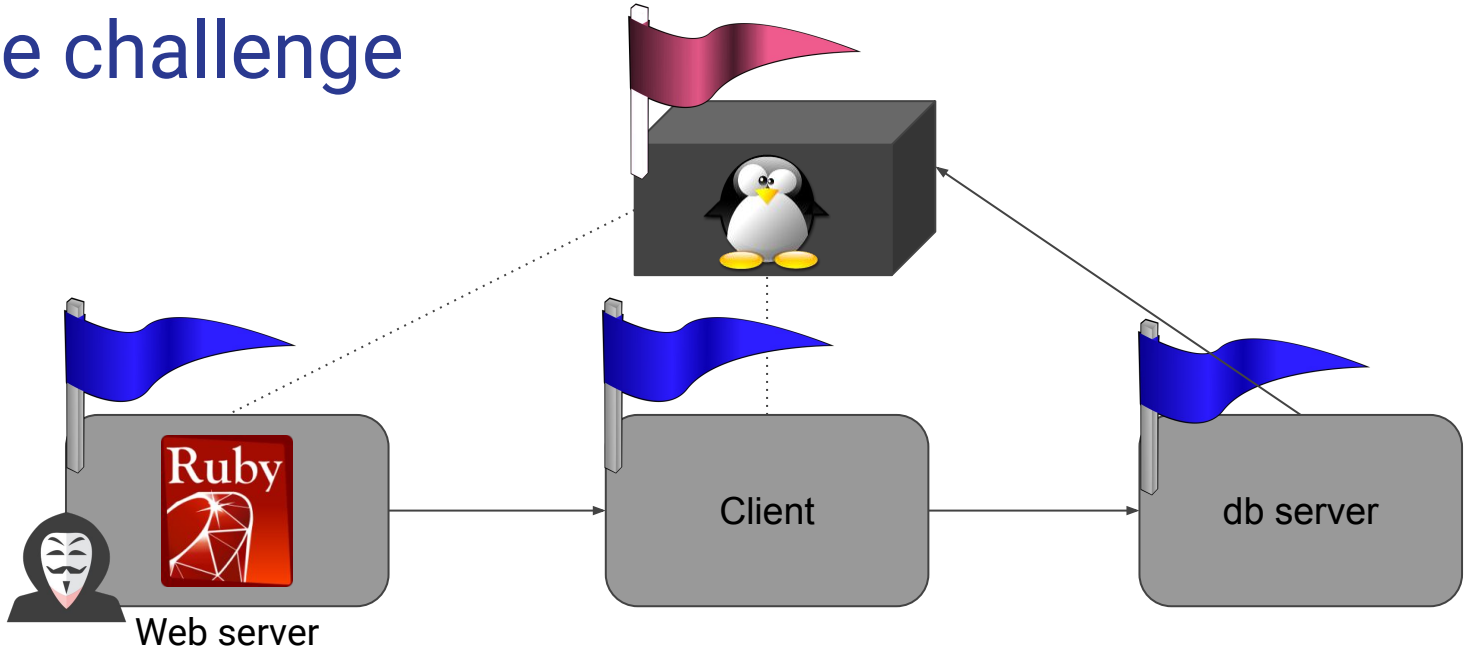
The challenge



The challenge



The challenge

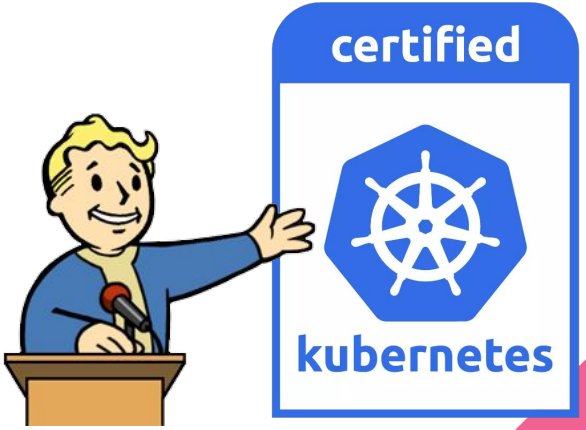


Why cloud?

- Machines hosted on our side
  - Impossible to cheat (by reading memory/docker exec)
  - Control and monitor all instances
- Researching cloud attack patterns

Why Kubernetes?

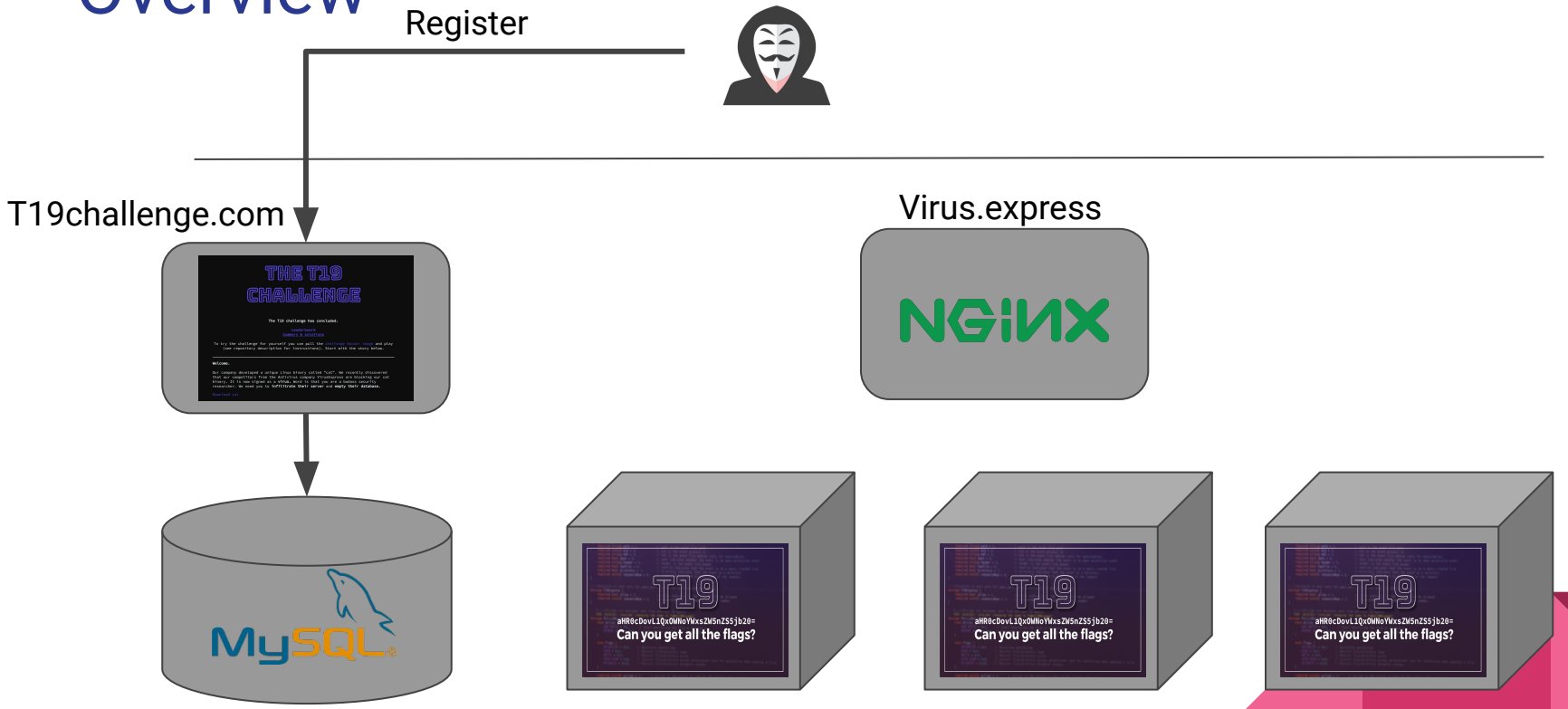
- Easy to scale
- Easy to update (hotfix)
- Easy configuration management
- Good baseline security



Engineering requirements

1. Simple (but not simplistic)
2. Cheap / Cost effective (time + resources)
3. Reproducible and partially automated*
4. Secure* by default

Overview


Register
T19challenge.com
Virus.express
THE T19 CHALLENGE
NGINX
MySQL
T19
aHR0cDovL1QxOWNoYWxsZW5nZS5jb20=
Can you get all the flags?
T19
aHR0cDovL1QxOWNoYWxsZW5nZS5jb20=
Can you get all the flags?
T19
aHR0cDovL1QxOWNoYWxsZW5nZS5jb20=
Can you get all the flags?

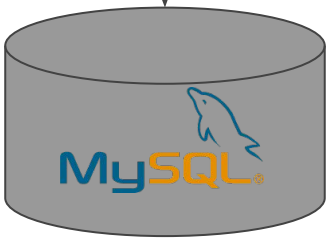
Overview

Register

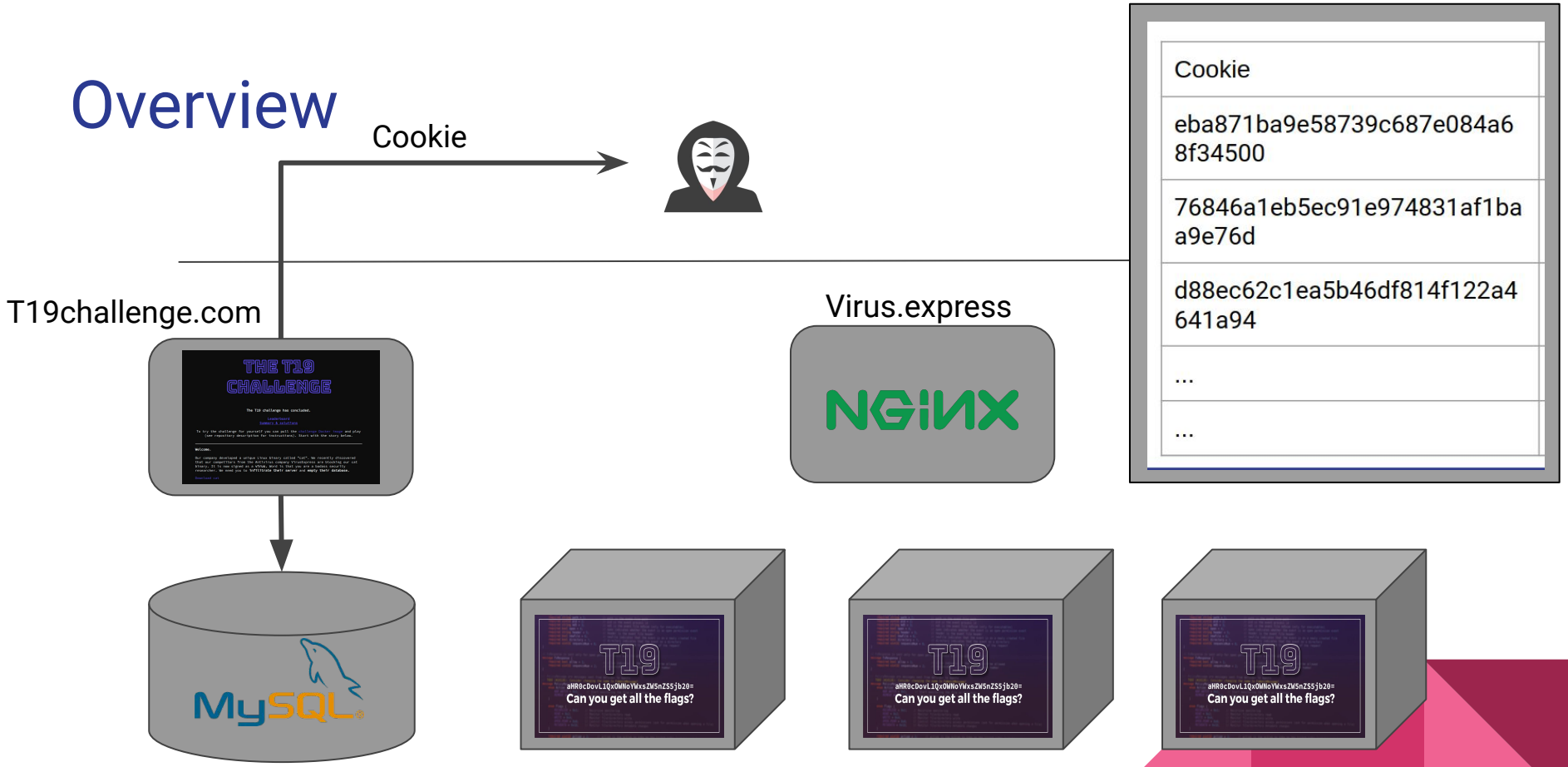
Cookie
eba871ba9e58739c687e084a68f34500
76846a1eb5ec91e974831af1baa9e76d
d88ec62c1ea5b46df814f122a4641a94
...
...

T19challenge.com

Virus.express



Overview

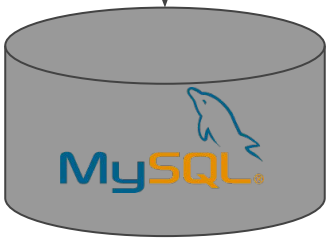

Cookie
T19challenge.com
Virus.express
NGINX
MySQL
Cookie
eba871ba9e58739c687e084a68f34500
76846a1eb5ec91e974831af1baa9e76d
d88ec62c1ea5b46df814f122a4641a94
...
...
T19
aHR0cDovL1QxOWNoYWxsZW5nZS5jb20=
Can you get all the flags?

Overview

Cookie

Cookie
eba871ba9e58739c687e084a68f34500
76846a1eb5ec91e974831af1baa9e76d
d88ec62c1ea5b46df814f122a4641a94
...
...

T19challenge.com



Virus.express

Overview

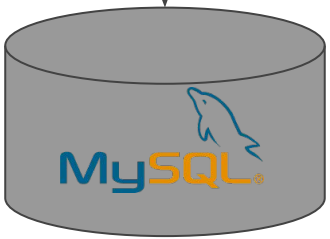
Cookie

```
apiVersion: v1
kind: ConfigMap
metadata:
 name: nginx-config
data:
 nginx.conf: |
  http {
  limit_req_zone $binary_remote_addr zone=one:10m rate=1r/s;
  map $cookie_t19userid $backend {
  default ";

  eba871ba9e58739c687e084a68f34500 http://10.245.0.3:13337;
  76846a1eb5ec91e974831af1baa9e76d http://10.245.0.4:13337;
  d88ec62c1ea5b46df814f122a4641a94 http://10.245.0.5:13337;
```

T19challenge.com

Virus.expres



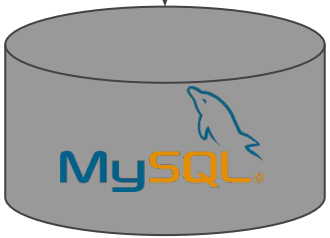
Overview

Cookie

```
apiVersion: v1
kind: ConfigMap
metadata:
 name: nginx-config
data:
 nginx.conf: |
  http {
  limit_req_zone $binary_remote_addr zone=one:10m rate=1r/s;
  map $cookie_t19userid $backend {
  default ";

  eba871ba9e58739c687e084a68f34500 http://10.245.0.3:13337;
  76846a1eb5ec91e974831af1baa9e76d http://10.245.0.4:13337;
  d88ec62c1ea5b46df814f122a4641a94 http://10.245.0.5:13337;
```

T19challenge.com



Virus.expres

10.245.0.3

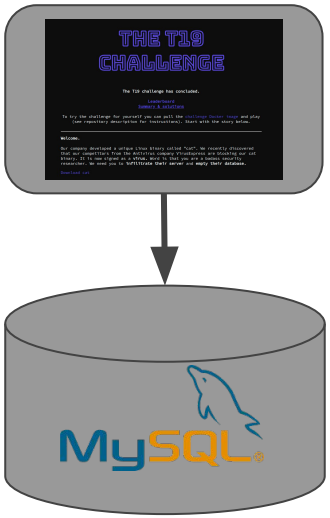
10.245.0.4

10.245.0.5

Overview

Cookie

T19challenge.com



Virus.express

10.245.0.4


```
kind: Service
apiVersion: v1
metadata:
  name: ctf-1
spec:
  selector:
    app: ctf-1
  ports:
  - protocol: TCP
    port: 13337
    targetPort: 13337
```

Infrastructure setup

1. Statically allocate all resources -
   Expensive, non-deterministic
2. On demand allocate pods + services -
   Complex, require nginx change + k8s access
3. Hybrid - statically allocate services + dynamically allocate pods

Pre-allocated service IPs

Predefined service subnet (--service-cidr=10.245.0.0/16)

Create all services (>k before) before creating pods

```
kind: Service
apiVersion: v1
metadata:
  name: ctf-1
spec:
  clusterIP: 10.245.0.3
  selector:
    app: ctf-1
  ports:
  - protocol: TCP
    port: 13337
    targetPort: 13337
```

Pre-allocated service IPs

Predefined service subnet (--service-cidr=10.245.0.0/16)

Create all services (>k before) before creating pods

```
kind: Service
apiVersion: v1
metadata:
  name: ctf-1
spec:
  clusterIP: 10.245.0.3
  selector:
    app: ctf-1
  ports:
  - protocol: TCP
    port: 13337
    targetPort: 13337
```

Pre-allocated service IPs

Predefined service subnet (--service-cidr=10.245.0.0/16)

Create all services (>k before) before creating pods

```
kind: Service
apiVersion: v1
metadata:
  name: ctf-1
spec:
  clusterIP: 10.245.0.3
  selector:
    app: ctf-1
  ports:
  - protocol: TCP
    port: 13337
    targetPort: 13337
```

Static storage and load balancer

Cookie	Cluster-ip
eba871….	10.245.0.3
76846a...	10.245.0.4
d88ec6...	10.245.0.5
...	10.245.0.5
...	...


```
apiVersion: v1
kind: ConfigMap
metadata:
  name: nginx-config
data:
  nginx.conf: |
    http {
    limit_req_zone $binary_remote_addr zone=one:10m rate=1r/s;
    map $cookie_t19userid $backend {
    default ";

    eba871ba9e58739c687e084a68f34500 http://10.245.0.3:13337;
    76846a1eb5ec91e974831af1baa9e76d http://10.245.0.4:13337;
    d88ec62c1ea5b46df814f122a4641a94 http://10.245.0.5:13337;
```

Static storage and load balancer

Cookie	Cluster-ip
eba871....	10.245.0.3
76846a...	10.245.0.4
d88ec6...	10.245.0.5
...	10.245.0.5
...	...


```
apiVersion: v1
kind: ConfigMap
metadata:
  name: nginx-config
data:
  nginx.conf: |
    http {
    limit_req_zone $binary_remote_addr zone=one:10m rate=1r/s;
    map $cookie_t19userid $backend {
    default '';

    eba871ba9e58739c687e084a68f34500 http://10.245.0.3:13337;
    76846a1eb5ec91e974831af1baa9e76d http://10.245.0.4:13337;
    d88ec62c1ea5b46df814f122a4641a94 http://10.245.0.5:13337;
```

On demand* pod allocation

Create pods on demand (or in batches)

```
kind: Deployment
metadata:
  name: ctf-1
  labels:
    app: ctf-1
spec:
    spec:
      containers:
      - name: ctf-1
        image: twistlock/t19
        ports:
        - containerPort: 13337
```

On demand* pod allocation

Create pods on demand (or in batches)

```
kind: Deployment
metadata:
  name: ctf-1
  labels:
    app: ctf-1
spec:
    spec:
      containers:
      - name: ctf-1
        image: twistlock/t19
        ports:
        - containerPort: 13337
```

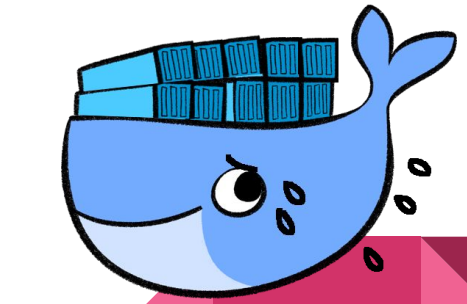
Security challenges

Risk	Causes
Local resource exhaustion	Crypto miners CPU/memory exhaustion (accident)
Attacker breaks out of the container	Misconfiguration (host mount/secrets) Vulnerabilities
Cluster compromised - Steal sensitive data (images)	API compromised (k8s/cloud) Example 1 (Bsides CTF) Example 2 (SSRF to takeover)

Local resource exhaustion - mitigations

- Block outgoing ports used for crypto miners (30303,8545,18080,18081…)
- Pod security policy (cgroups)

```
apiVersion: v1
kind: Pod
metadata:
  name: ctf
spec:
  containers:
  - name: ctf-app
    image: twistlock/t19
    resources:
      requests:
        memory: "30Mi"
        cpu: "50m"
      limits:
        memory: "50Mi"
        cpu: "50m"
```



Container breakout - mitigations

- Classic container - No mounts/secrets - simple app
- Default container profile (no additional LINUX capabilities + seccomp)
- Container optimized OS - read only root partition (CVE-2019-5736 mitigation)
- User namespaces*

Cluster takeover - mitigations

- Completely isolated environment
- automountServiceAccountToken: false
- Metadata concealment / Network policies

Network policy

```
kind: NetworkPolicy
spec:
  podSelector:
    matchLabels:
      app: t19
  policyTypes:
  - Ingress
  - Egress
  egress:
  - to:
    - ipBlock:
        cidr: 0.0.0.0/0
        except:
        - 169.254.169.254/32
  ingress:
  - from:
    - podSelector:
        matchLabels:
          app: t19-nginx
```

Challenge conclusion

- 8 participants solved
  - 6 found 4th flag
- Excellent write-ups with solutions
- Links and finalists
- Challenge coins molded

Key takeouts

- Good engineering == cost saving
- Good security
- Kubernetes is a great platform to host a live CTF

Try to solve?

- http://t19challenge.com/
- Follow the instructions to run
- Don’t cheat and good luck!
- See you in T20?

KubeCon | CloudNativeCon

Europe 2019

Thank you!

Twistlock.com/labs

@TwistlockLabs