



Cloud Native Days Austria
2025

Catch Hackers with KONEY

Automated Honeytokens for Cloud-Native Apps



Cloud Native Days Austria
2025



Catch Hackers with KONEY

Automated Honeytokens for Cloud-Native Apps



CO-DEVELOPER

Matteo Golinelli

University of Trento



PRESENTER

Mario Kahlhofer

Dynatrace Research

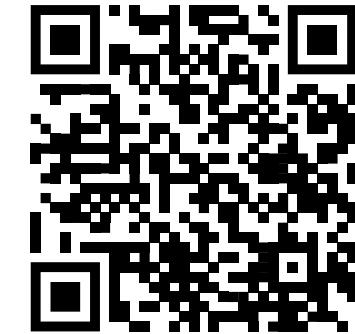
whoami



MARIO KAHLHOFER

Senior Research Scientist · Dynatrace Research
PhD Student · Johannes Kepler University Linz

mario.kahlhofer@dynatrace.com
github.com/blu3r4y

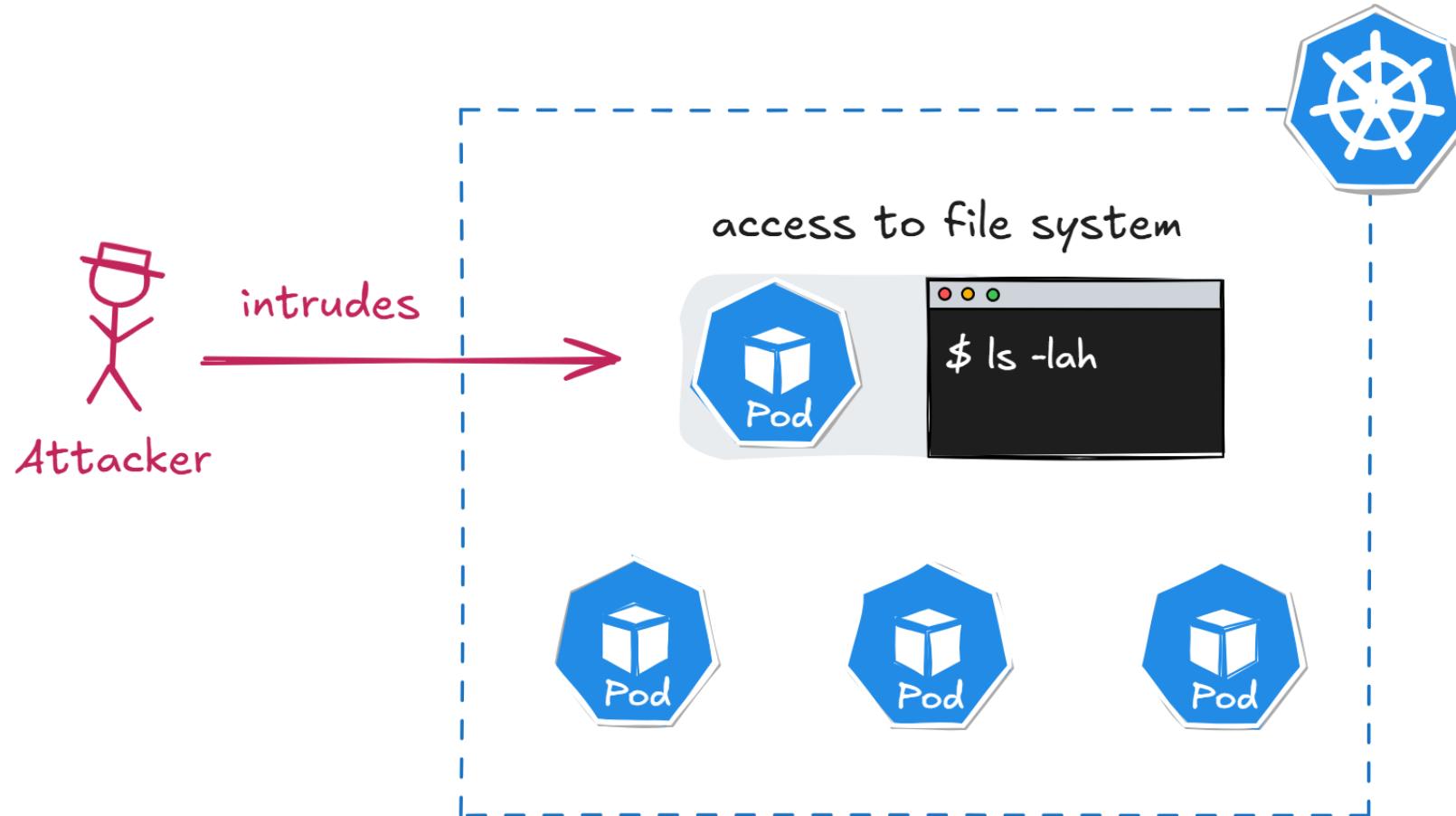


LinkedIn

Cloud Security · **Kubernetes** · Containers · Linux · eBPF
Cyber Deception · **Honeytokens** · Honeypots
Game Theory · Machine Learning

INTRODUCTION

Imagine an attacker infiltrated your cluster ...



INTRODUCTION

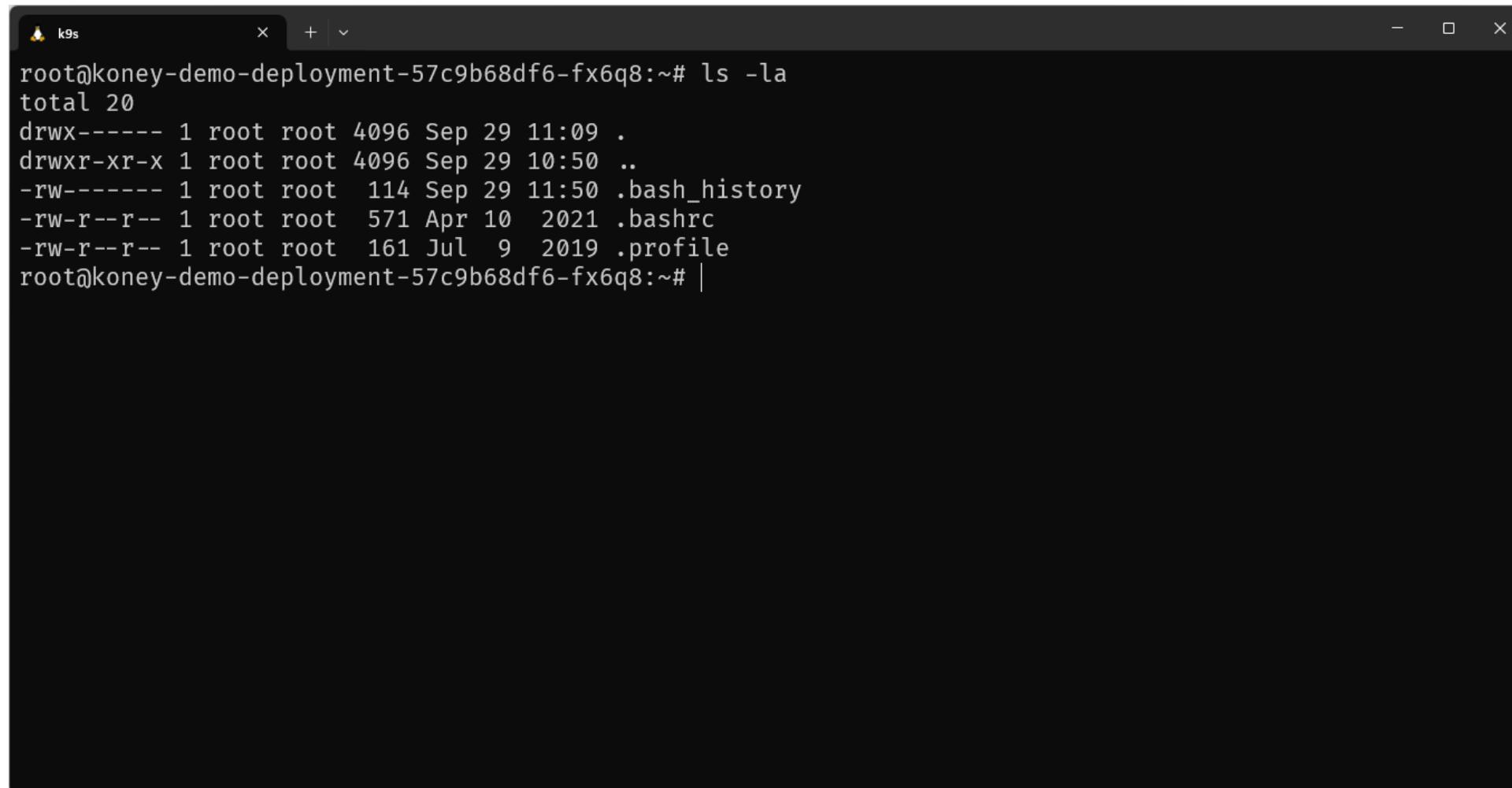
You're the attacker. Where should we go first?



```
k9s           x + v - □ ×
root@koney-demo-deployment-57c9b68df6-fx6q8:/# ls -l
total 64
lrwxrwxrwx  1 root root   7 Sep  8 00:00 bin → usr/bin
drwxr-xr-x  2 root root 4096 Aug 24 16:05 boot
drwxr-xr-x  5 root root  360 Sep 29 10:50 dev
drwxr-xr-x  1 root root 4096 Sep  8 21:14 docker-entrypoint.d
-rw xr-xr-x  1 root root 1620 Sep  8 21:13 docker-entrypoint.sh
drwxr-xr-x  1 root root 4096 Sep 29 10:50 etc
drwxr-xr-x  2 root root 4096 Aug 24 16:05 home
lrwxrwxrwx  1 root root   7 Sep  8 00:00 lib → usr/lib
lrwxrwxrwx  1 root root   9 Sep  8 00:00 lib64 → usr/lib64
drwxr-xr-x  2 root root 4096 Sep  8 00:00 media
drwxr-xr-x  2 root root 4096 Sep  8 00:00 mnt
drwxr-xr-x  2 root root 4096 Sep  8 00:00 opt
dr-xr-xr-x 237 root root    0 Sep 29 10:50 proc
drwx-----  1 root root 4096 Sep 29 11:09 root
drwxr-xr-x  1 root root 4096 Sep 29 10:50 run
lrwxrwxrwx  1 root root   8 Sep  8 00:00 sbin → usr/sbin
drwxr-xr-x  2 root root 4096 Sep  8 00:00 srv
dr-xr-xr-x 12 root root    0 Sep 29 10:50 sys
drwxrwxrwt  2 root root 4096 Sep  8 00:00 tmp
drwxr-xr-x  1 root root 4096 Sep  8 00:00 usr
drwxr-xr-x  1 root root 4096 Sep  8 00:00 var
root@koney-demo-deployment-57c9b68df6-fx6q8:/# |
```



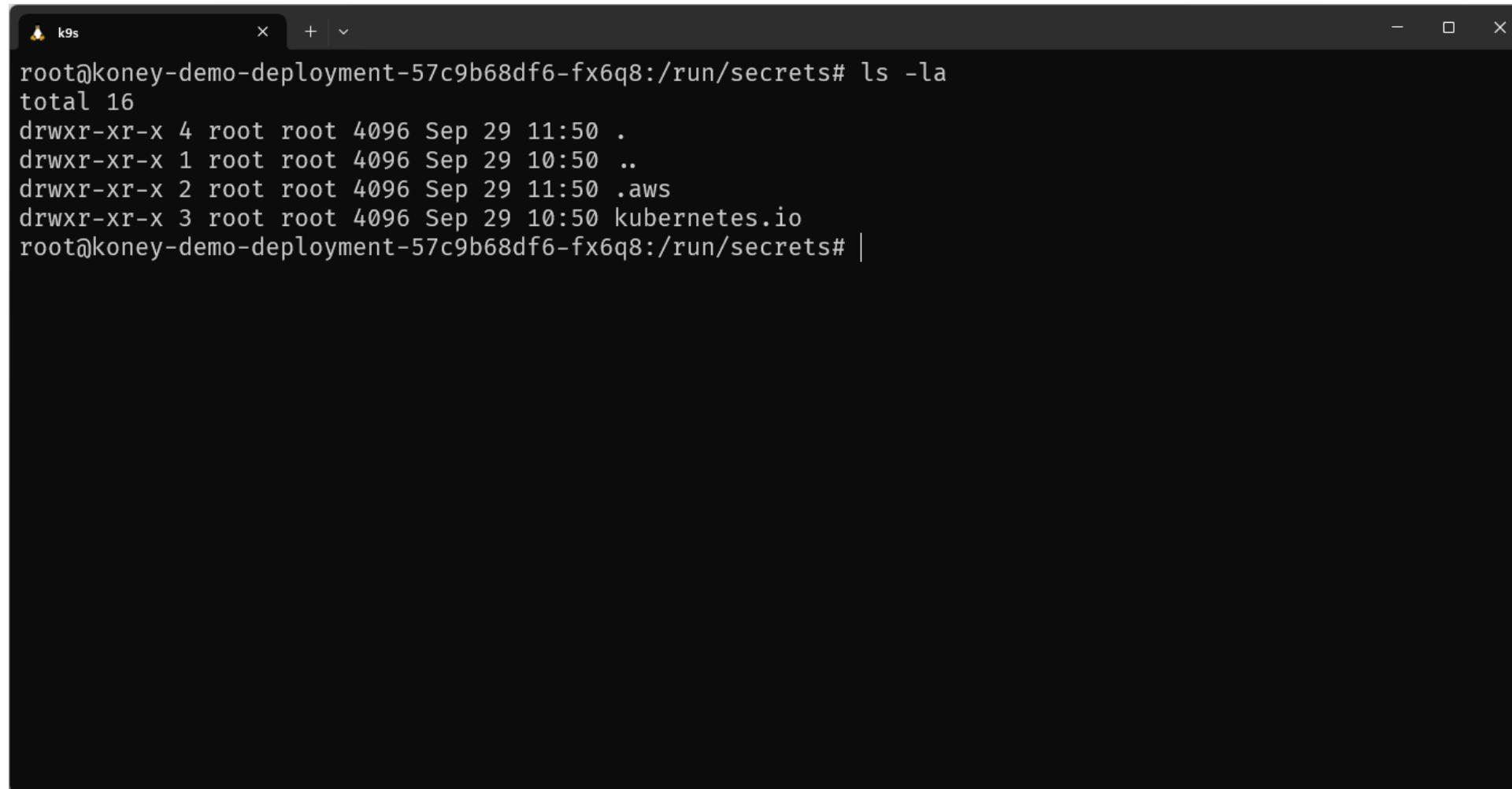
~ /

A screenshot of a terminal window titled "k9s". The window shows a command-line interface with the following output:

```
root@koney-demo-deployment-57c9b68df6-fx6q8:~# ls -la
total 20
drwx----- 1 root root 4096 Sep 29 11:09 .
drwxr-xr-x 1 root root 4096 Sep 29 10:50 ..
-rw----- 1 root root 114 Sep 29 11:50 .bash_history
-rw-r--r-- 1 root root 571 Apr 10 2021 .bashrc
-rw-r--r-- 1 root root 161 Jul 9 2019 .profile
root@koney-demo-deployment-57c9b68df6-fx6q8:~# |
```

The terminal has a dark theme with light-colored text. The title bar includes standard window controls (minimize, maximize, close).

/run/secrets/

A terminal window titled 'k9s' showing the command 'ls -la' being run in the directory '/run/secrets'. The output lists several files and directories, including '.', '..', '.aws', and 'kubernetes.io'.

```
root@koney-demo-deployment-57c9b68df6-fx6q8:/run/secrets# ls -la
total 16
drwxr-xr-x 4 root root 4096 Sep 29 11:50 .
drwxr-xr-x 1 root root 4096 Sep 29 10:50 ..
drwxr-xr-x 2 root root 4096 Sep 29 11:50 .aws
drwxr-xr-x 3 root root 4096 Sep 29 10:50 kubernetes.io
root@koney-demo-deployment-57c9b68df6-fx6q8:/run/secrets# |
```



If an attacker breaks into your container, they may search for sensitive information.



/run/secrets/kubernetes.io/serviceaccount/

```
k9s
root@koney-demo-deployment-57c9b68df6-fx6q8:/run/secrets# ls -la
total 16
drwxr-xr-x 4 root root 4096 Sep 29 11:50 .
drwxr-xr-x 1 root root 4096 Sep 29 10:50 ..
drwxr-xr-x 2 root root 4096 Sep 29 11:50 .aws
drwxr-xr-x 3 root root 4096 Sep 29 10:50 kubernetes.io
root@koney-demo-deployment-57c9b68df6-fx6q8:/run/secrets# ls -l kubernetes.io/serviceaccount/
total 0
lrwxrwxrwx 1 root root 13 Sep 29 10:50 ca.crt → ..data/ca.crt
lrwxrwxrwx 1 root root 16 Sep 29 10:50 namespace → ..data/namespace
lrwxrwxrwx 1 root root 12 Sep 29 10:50 token → ..data/token
root@koney-demo-deployment-57c9b68df6-fx6q8:/run/secrets# cat kubernetes.io/serviceaccount/token
eyJhbGciOiJSUzI1NiIsImtpZCI6IkdlFybGN3cFhHUEcybUFFQ0JPbXBjX1ZySTFwN1hsdTdWYkLCQjlWRHcifQ.eyJhdWQiols
iaHR0cHM6Ly9rdWJlc5ldGVzLmRlZmF1bHQuc3ZjLmNsdxN0ZXIubG9jYWwiXSwizXhwIjoxNzkwNjgxOTE5LCJpYXQioje3NTkxN
DU5MTksImlzcyI6Imh0dHBzOi8va3ViZXJuZXRLcy5kZWZhdWx0LnN2Yy5jbHVzdGVyLmxvY2FsIiwianRpIjoiNDY0ODY10TgtOTU
2Yi00M2FhLTg4MDItOWI4NzhhZTEzZTg2Iiwick3ViZXJuZXRLcy5pbvI6eyJuYW1lc3BhY2UiOjrb25leS1kZW1vIiwbm9kZSI6e
yJuYW1lIjoibWluawt1YmUiLCJ1aWQiOjJkNmZkY2MzZC0wOTE0LTQyZmUtODQ3Ny1iYmNhYzg4ZDU4ZGUifSwicG9kIjp7Im5hbWU
iOjrb25leS1kZW1vLWRlcGxveW1bnQtNTdjOWI20GRmNi1meDzxOCIsInVpZCI6IjQ5NTY30TExLWZiNjMtNDEyYy04ZmU1LTM2M
TNJODVjNDYzNiJ9LCJzZXJ2aWNlyWNjb3VudCI6eyJuYW1lIjoizGVmYXVsdCIIsInVpZCI6IjBmNTZhYjA1LWE3MDUtNDhmNC04ZDF
hLWM3ZWMwMThlNTA1MyJ9LCJ3YXJuYWZ0ZXi0je3NTkxNDk1MjZ9LCJuYmYiOjE3NTkxNDU5MTksInN1YiI6InN5c3RlbTpzZXJ2a
wnlYWNjb3VudDprb25leS1kZW1vOmRlZmF1bHQifQ.Eo5HIeHpLz2c7al_oadgiFJHJNm1r1WJdAJ1q57lPUTlgfmi-XpcDQPUqVMc
tbKssfXXEykIAi8uVW-7Fui_x1AIcFY0WORlhP81asmYW1zT85NdgJNw11zUEsa0ZGr7H0CrtzNGFUNUkCpvkFUAcinZY_dc3W9V3
CLCA3fQSfZIOT8hHtsvr36p4uiSRNd9NF6rBVbh4L5D5vHKubB8zrrsy4Ml1dm_NfhSNRdCaUESTdolg-xCJnTgKxqixLtgkLkhL6cA
BA_M5e6pE8CzJhVTHgVUvpls_posPoWexjCTbFFG_Bfr0C64vzgHBzfWK34ZWk6M0ncg-rMNuFMSkAroot@koney-demo-deployment-57c9b68df6-fx6q8:/run/secrets# |
```



If an attacker breaks into your container, they may search for sensitive information.

/run/secrets/.aws/

```
k9s x + v - □ ×  
root@koney-demo-deployment-57c9b68df6-fx6q8:/run/secrets# ls -la  
total 16  
drwxr-xr-x 4 root root 4096 Sep 29 11:50 .  
drwxr-xr-x 1 root root 4096 Sep 29 10:50 ..  
drwxr-xr-x 2 root root 4096 Sep 29 11:50 .aws  
drwxr-xr-x 3 root root 4096 Sep 29 10:50 kubernetes.io  
root@koney-demo-deployment-57c9b68df6-fx6q8:/run/secrets# ls -l .aws  
total 4  
-r--r--r-- 1 root root 371 Sep 29 11:50 credentials  
root@koney-demo-deployment-57c9b68df6-fx6q8:/run/secrets# cat .aws/credentials  
[default]  
aws_access_key_id = FZWGQYZRVMVQKCWXWJ  
aws_secret_access_key = ToDNS2V5wkEuPmfocmNDHiVBspaywo  
region = us-east-1  
  
[staging]  
aws_access_key_id = BPDHLBUJTYMKRVBMNB  
aws_secret_access_key = Jf4mTvTG5zbLksf464nYwEVyDkbjdp  
region = eu-central-1  
  
[prod]  
aws_access_key_id = GQRMYWAXADTVUPFMZJ  
aws_secret_access_key = gUFQzyJWJSkMkWpay4Umxzg4oFkkju  
region = us-east-1  
root@koney-demo-deployment-57c9b68df6-fx6q8:/run/secrets# |
```



If an attacker breaks into your container, they may search for sensitive information.

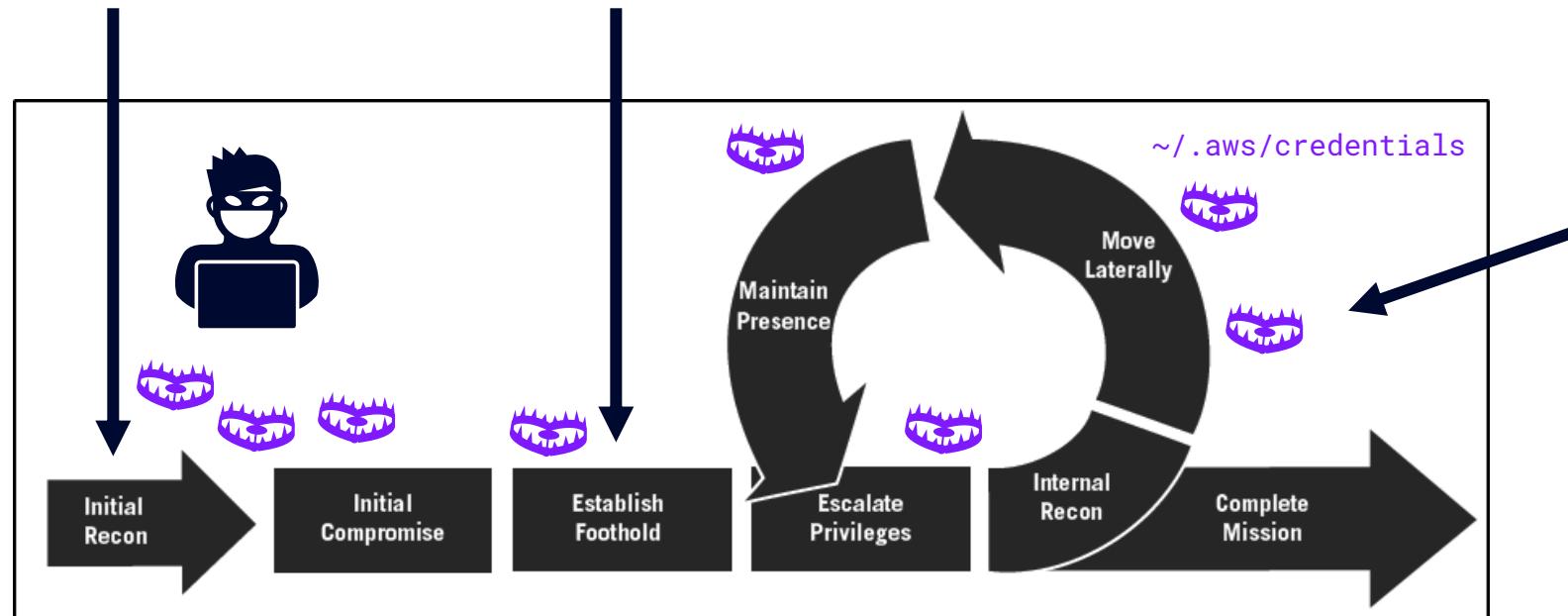
INTRODUCTION

What is Cyber Deception?



Attacks start with reconnaissance

Sophisticated adversaries will eventually find a way to infiltrate your network



We can place **honeytokens**, install **fake endpoints**, and inject **deceptive content** to ...

- slow-down attackers by expending their “energy”
- assist defenders with strong indicators of compromise

[1]



[1] Mandiant, "APT1: Exposing One of China's Cyber Espionage Units," 2013. [Online].

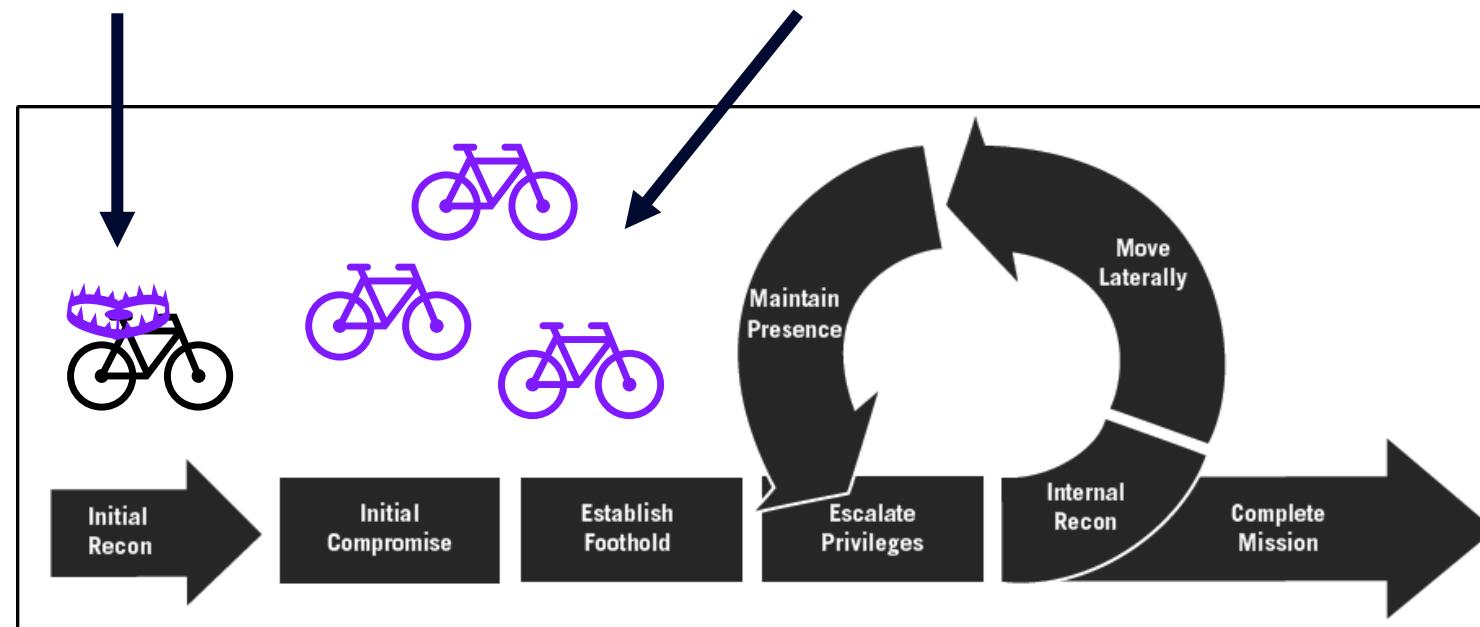
INTRODUCTION

Honeypots and Honeytokens



Thieves shall not
steal your bike

Let's place some
fake bikes then



[1]



'Classic' honeypots are isolated fake applications, reachable over the network (e.g., to collect threat intel)



Talk Focus

Honeytokens & application layer deception techniques are traps embedded into applications & systems

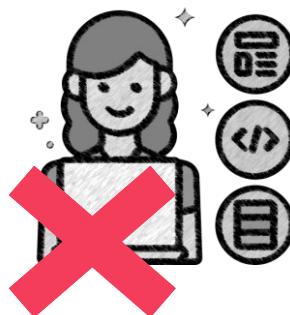


[1] Mandiant, "APT1: Exposing One of China's Cyber Espionage Units," 2013. [Online].

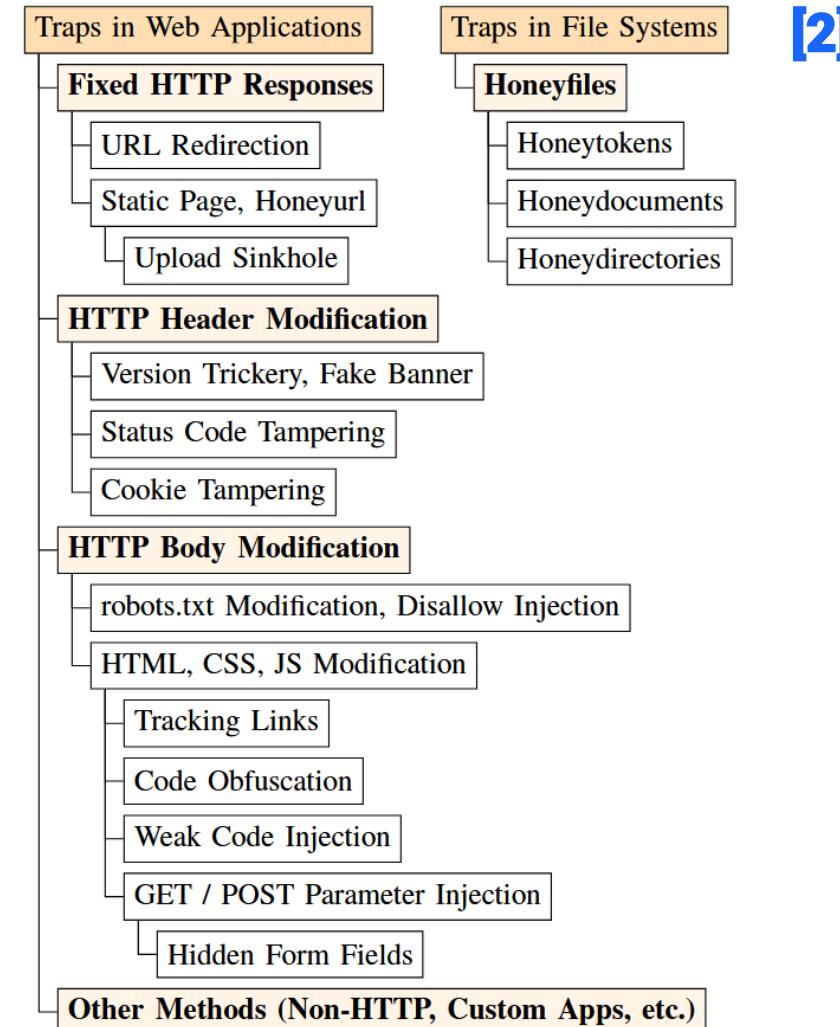
APPLICATION LAYER CYBER DECEPTION

Let's Embed Traps Directly Into Applications!

- Place **Honeytokens** in (container) filesystems
- Add new **HTTP endpoints** to mislead attackers
- Modify **HTTP headers**, e.g., version numbers
- Modify **HTTP bodies**, e.g., hidden form fields
- Other (non-HTTP) methods



But software applications are rarely deployed by the team that wrote the code, and often the responsibility for security measures lies entirely elsewhere.



THE IDEA

“Deception-As-Code” with Koney [2]

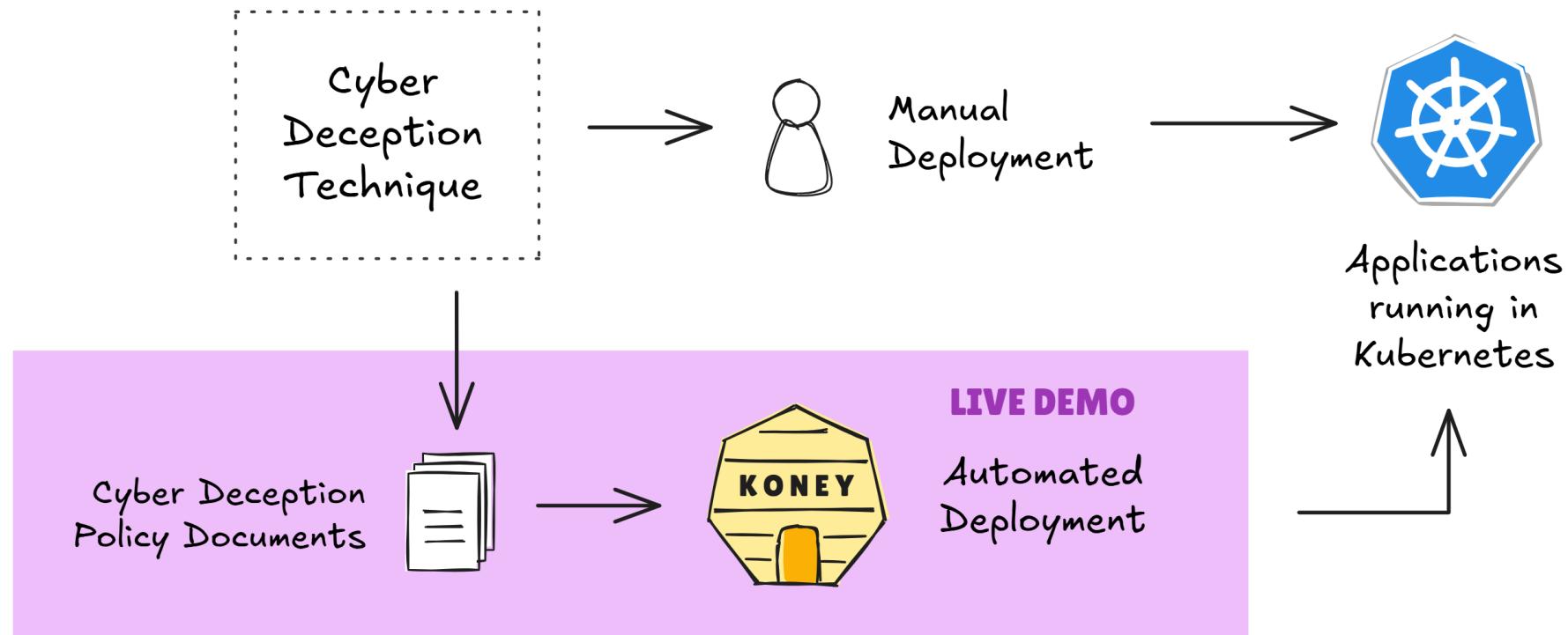


github.com/dynatrace-oss/koney



GitHub

... instead of manually deciphering techniques from academic papers.



[2] M. Kahlhofer, M. Golinelli, and S. Rass, “Koney: A Cyber Deception Orchestration Framework for Kubernetes,” in EuroS&PW ’25. Venice, Italy: IEEE, 2025, pp. 690–702. doi: [10.1109/EuroSPW67616.2025.00084](https://doi.org/10.1109/EuroSPW67616.2025.00084).



13

THE IDEA

Cyber Deception Policy Documents

```
honeytoken.yaml          YAML

apiVersion: koney.io/v1alpha1
kind: DeceptionPolicy
spec:
  traps:
    - filesystemHoneytoken:
        filePath: /run/secrets/token
        fileContent: "secret"
    match:
      any:
        - resources:
            selector:
              matchLabels:
                op/honeytoken: true
  decoyDeployment:
    strategy: containerExec
  captorDeployment:
    strategy: tetragnon
```



github.com/dynatrace-oss/koney



GitHub

- ← Trap-specific parameterization
- ← Criteria for selecting the workloads
(e.g., containers) in which to deploy the traps
- ← Decoy. [3] Strategy to deploy the trap itself
- ← Captor. [3] Strategy for monitoring the trap



[3] W. Fan, Z. Du, D. Fernández, and V. A. Villagrá, "Enabling an Anatomic View to Investigate Honeypot Systems: A Survey," IEEE Sys. J., vol. 12, no. 4, pp. 3906–3919, 2018, doi: [10.1109/jsyst.2017.2762161](https://doi.org/10.1109/jsyst.2017.2762161).



LIVE DEMO
DeceptionPolicy

INIT



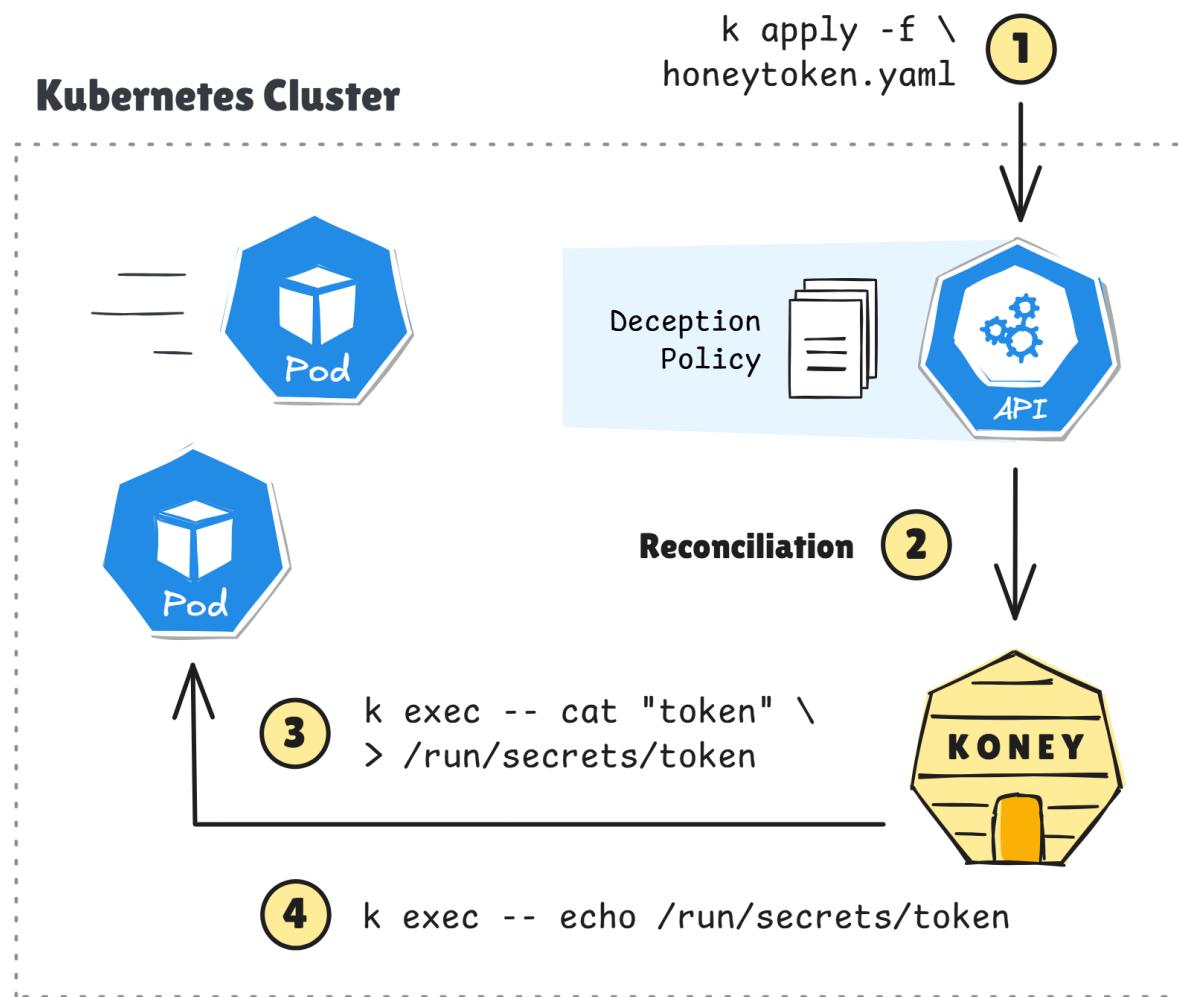
LIVE DEMO
More Traps!





DECOY STRATEGY

Placing Honeytokens by Executing Shell Commands



```
honeytoken.yaml          YAML

apiVersion: koney.io/v1alpha1
kind: DeceptionPolicy
spec:
  traps:
    - filesystemHoneypot:
        filePath: /run/secrets/token
        fileContent: "secret"
  match:
    any:
      - resources:
          selector:
            matchLabels:
              op/honeypot: true
  decoyDeployment:
    strategy: containerExec
  captorDeployment:
    strategy: tetragon
```



Placing Honeytokens by Executing Shell Commands (cont.)

Deployment

```
cat "secret" > /run/secrets/token
```

Verification

```
echo /run/secrets/token
```

Clean-Up

```
rm /run/secrets/token
```

Monitoring

of access attempts

?

Transparency

for system operators

✓ **DeceptionPolicy**

Zero Downtime

of application services

✓ yes

Non-interference

with genuine operation

✓ just a few process executions

LIVE DEMO

File Access Monitoring



LIVE DEMO

Koney Alert Example

alert.json

JSON

```
{  
  "timestamp": "2025-06-02T11:17:02Z",  
  "deception_policy_name": "deceptionpolicy-aws-credentials",  
  "trap_type": "filesystem_honeytoken",  
  "metadata": { "file_path": "/run/secrets/.aws/credentials" },  
  "pod": {  
    "name": "koney-demo-deployment-5bcbd78875-45qpn",  
    "namespace": "koney-demo",  
    "container": {  
      "id": "docker://e19c1827e255ce7a5c5fd74eb4ee861388f83a16410effd65e30d3b051cd815f",  
      "name": "nginx"  
    }  
  },  
  "process": {  
    "pid": 148373, "uid": 0, "cwd": "/", "binary": "/usr/bin/cat",  
    "arguments": "/run/secrets/.aws/credentials"  
  }  
}
```



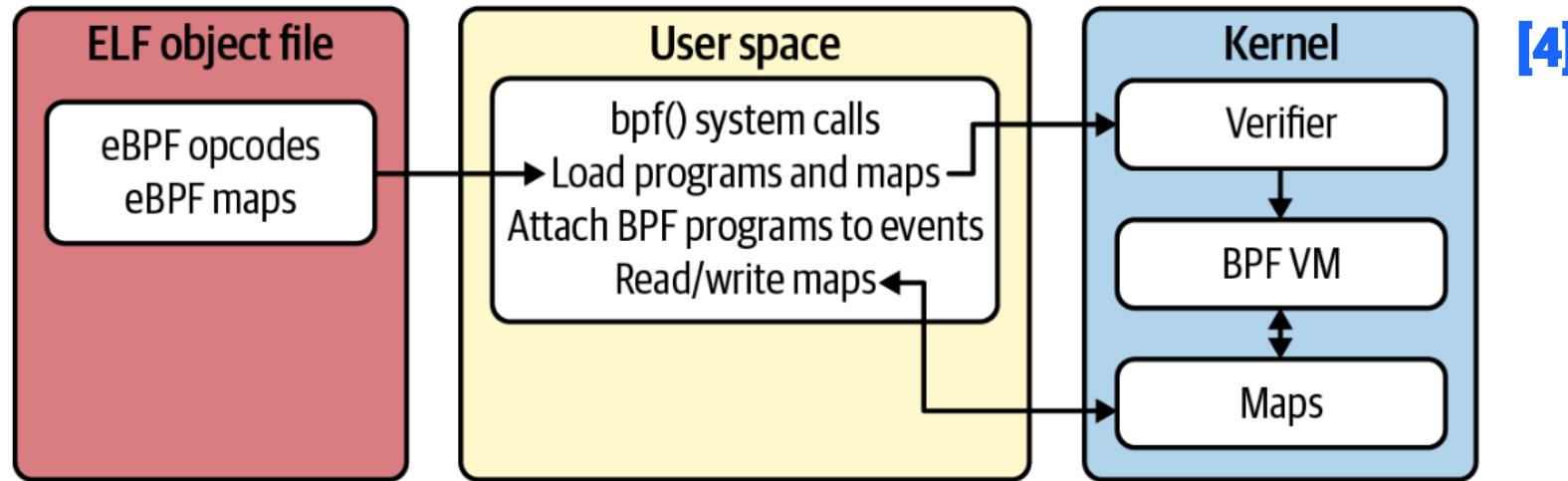


CAPTOR STRATEGY

File Access Monitoring with eBPF



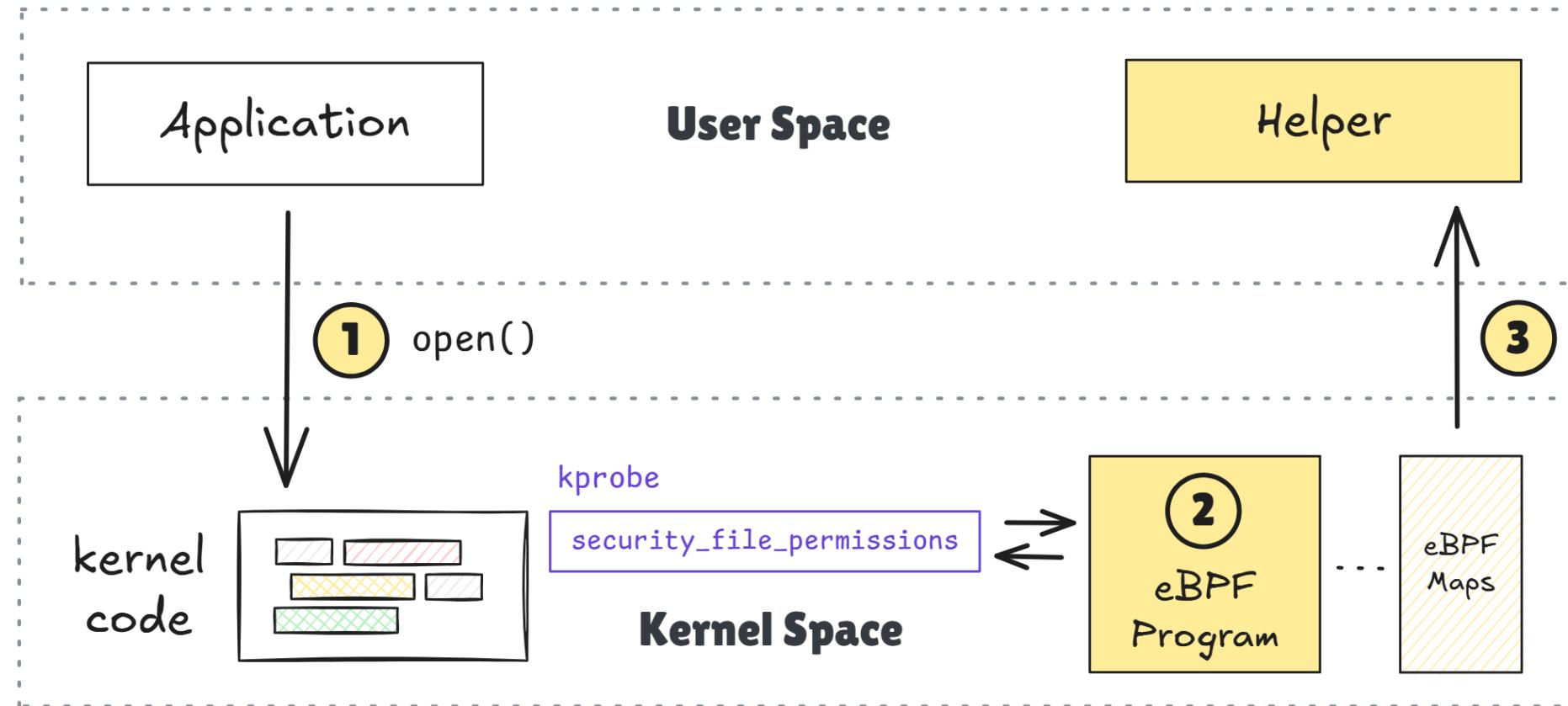
eBPF makes the kernel programmable. eBPF programs are typically written in a subset of C or Rust and compiled to an object file.





File Access Monitoring with eBPF (cont.)

We hook the **security_file_permissions** kprobe in kernel space.





CAPTOR STRATEGY

File Access Monitoring with eBPF (cont.)



```
policy.yaml          YAML  
  
apiVersion: cilium.io/v1alpha1  
kind: TracingPolicy  
metadata:  
  name: monitor-honeytoken  
spec:  
  kprobes:  
    - call: security_file_permission  
      syscall: false  
      return: true  
      args:  
        - index: 0  
          type: file  
        - index: 1  
          type: int  
      returnArg:  
        index: 0  
        type: int  
      returnArgAction: Post  
  selectors:  
    - matchArgs:  
      - index: 0  
        operator: Prefix  
      values:  
        - /run/secrets/token
```

[5]



Tetragon is a Kubernetes Operator
that simplifies the creation of
“tracing policies” in K8s clusters.

Falco and **Tracee** are popular alternatives.



[5] K. Kourtis and A. Papagiannis, “File Monitoring with eBPF and Tetragon (Part 1),” Isovalent Blog. Accessed: Feb. 2025. Available: <https://isovalent.com/blog/post/file-monitoring-with-ebpf-and-tetragon-part-1/>



LIVE DEMO
Distroless Images



SECURE

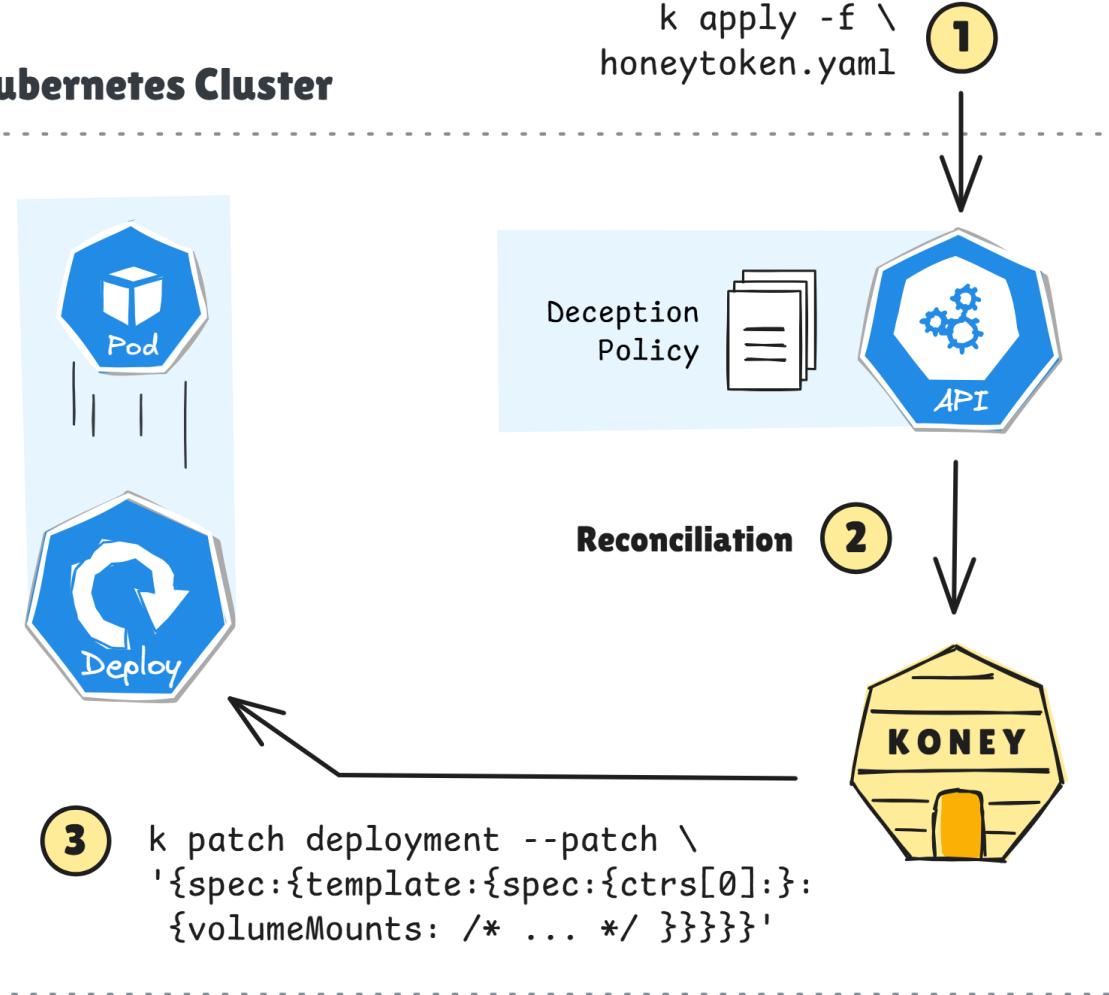


DECOY STRATEGY

Placing Honeytokens by Mounting Volumes



Kubernetes Cluster



deployment.yaml

```
YAML  
  
apiVersion: apps/v1  
kind: Deployment  
spec:  
  replicas: 2  
  selector:  
    matchLabels:  
      app: nginx  
  template:  
    metadata:  
      labels:  
        app: nginx  
  spec:  
    containers:  
      - image: nginx:latest  
        name: web  
        volumeMounts:  
          - mountPath: /run/secrets/token  
            name: koney-volume  
            readOnly: true  
    volumes:  
      - name: koney-volume  
        secret:  
          secretName: koney-secret
```



Placing Honeytokens With Koney

Strategy 1: Shell Commands

Deployment

```
cat "secret" > /run/secrets/token
```

Verification

```
echo /run/secrets/token
```

Clean-Up

```
rm /run/secrets/token
```

Monitoring

of access attempts

✓ eBPF (via Tetragon)**Transparency**

for system operators

✓ DeceptionPolicy**Zero Downtime**

of application services

✓ yes**Non-interference**

with genuine operation

✓ just a few process executions

Strategy 2: Volume Mounts

+spec.containers.volumeMounts

```
echo /run/secrets/token
```

-spec.containers.volumeMounts**✓ eBPF (via Tetragon)****✓ even better, visible manifest change****✗ needs container restart in Kubernetes****✓ even better, no process executions**

OUTLOOK

Deceive. Test. Repeat.

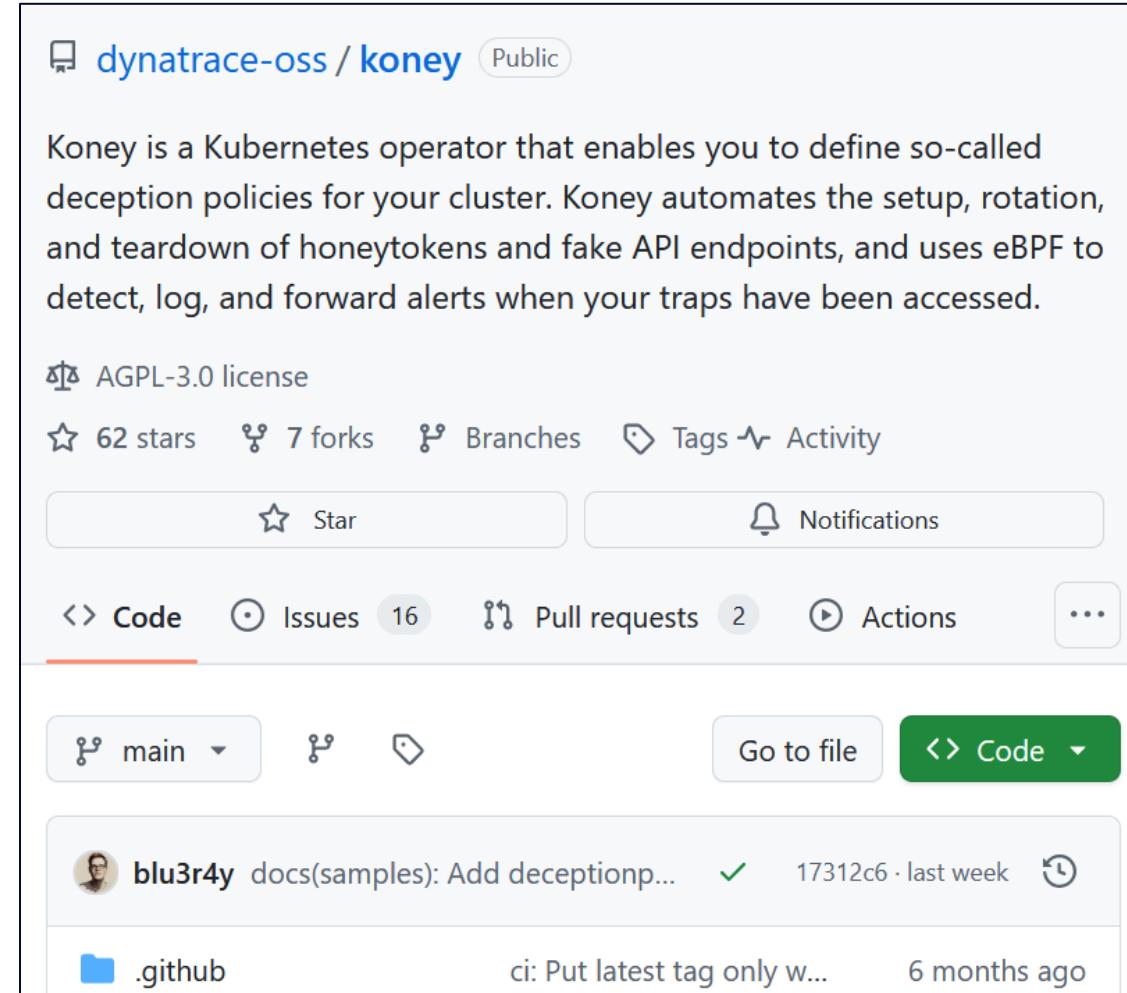
Let's provide **app developers** and **system operators** with a refreshingly different concept on detecting attackers!

Development Outlook

- Traps for HTTP-based apps
- eBPF Monitoring w/o Tetragon



GitHub

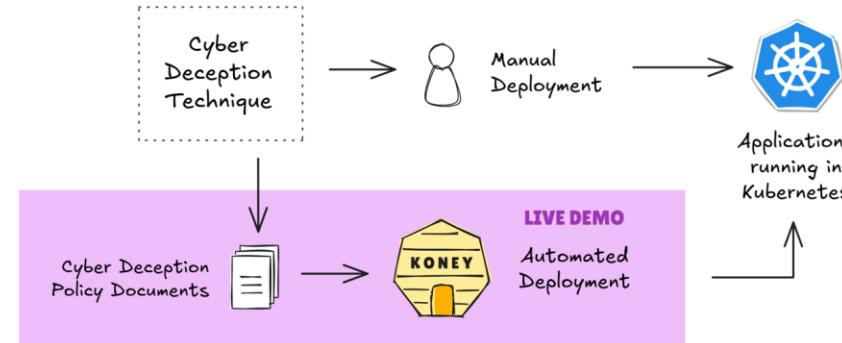


A screenshot of the GitHub repository page for 'koney' under the 'dynatrace-oss' organization. The repository is described as a Kubernetes operator for deception policies. It includes sections for license (AGPL-3.0), statistics (62 stars, 7 forks, 16 issues, 2 pull requests, 2 actions), and a code editor with a main branch dropdown and a green 'Code' button. Recent commits are listed at the bottom, including one from 'blu3r4y' and another from 'ci'.

THE IDEA

"Deception-As-Code" with Koney [2]

... instead of manually deciphering techniques from academic papers.



[2] M. Kahlhofer, M. Golinelli, and S. Rass, "Koney: A Cyber Deception Orchestration Framework for Kubernetes," in EuroS&PW '25, Venice, Italy: IEEE, 2025, pp. 690–702. doi: 10.1109/EuroSPW67616.2025.00084.

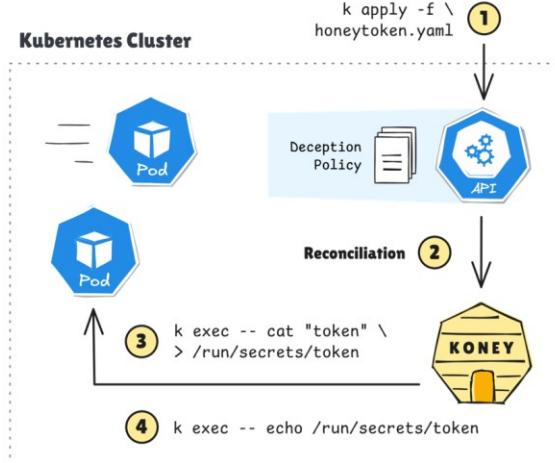


GitHub



DECOY STRATEGY

Placing Honeytokens by Executing Shell Commands



honeytoken.yaml

```
apiVersion: koney.io/v1alpha1
kind: DeceptionPolicy
spec:
  traps:
    - filesystemHoneytoken:
        filePath: /run/secrets/token
        fileContent: "secret"
  match:
    any:
      - resources:
          selector:
            matchLabels:
              op/honeytoken: true
  decoyDeployment:
    strategy: containerExec
  captorDeployment:
    strategy: tetragon
```

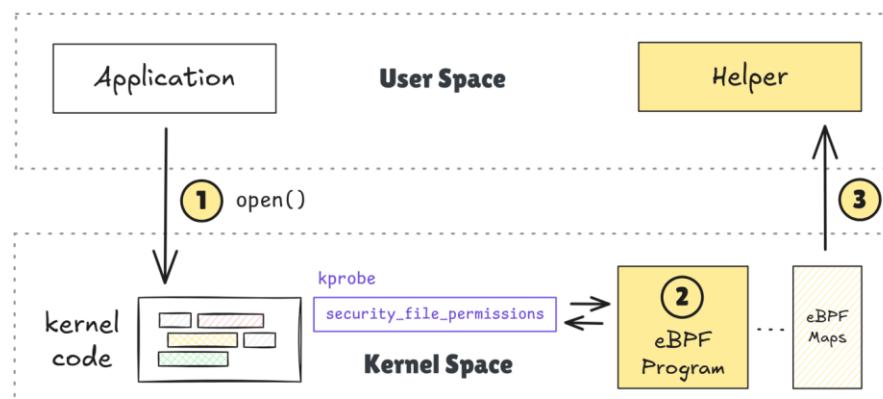


18

CAPTOR STRATEGY

File Access Monitoring with eBPF (cont.)

We hook the `security_file_permissions` kprobe in kernel space.



OUTLOOK

Deceive. Test. Repeat.

Let's provide **app developers** and **system operators** with a refreshingly different concept on detecting attackers!



github.com/dynatrace-oss/koney



GitHub

dynatrace-oss / koney (Public)

Koney is a Kubernetes operator that enables you to define so-called deception policies for your cluster. Koney automates the setup, rotation, and teardown of honeypoints and fake API endpoints, and uses eBPF to detect, log, and forward alerts when your traps have been accessed.

AGPL-3.0 license

62 stars 7 forks 1 Branches Tags Activity

Star Notifications

Code Issues 16 Pull requests 2 Actions ...

main Go to file Code

blu3r4y docs(samples): Add deception... 1731c6 · last week

.github ci: Put latest tag only w... 6 months ago

24

30