



KubeCon



CloudNativeCon

Europe 2026

#KubeCon #CloudNativeCon

From “It Works!” to “It’s Secure!” Hardening Your First Kubernetes Cluster

Paul Zerdilas Herrera | Leon Schulze





KubeCon



CloudNativeCon

Europe 2026

ACT I: It works!

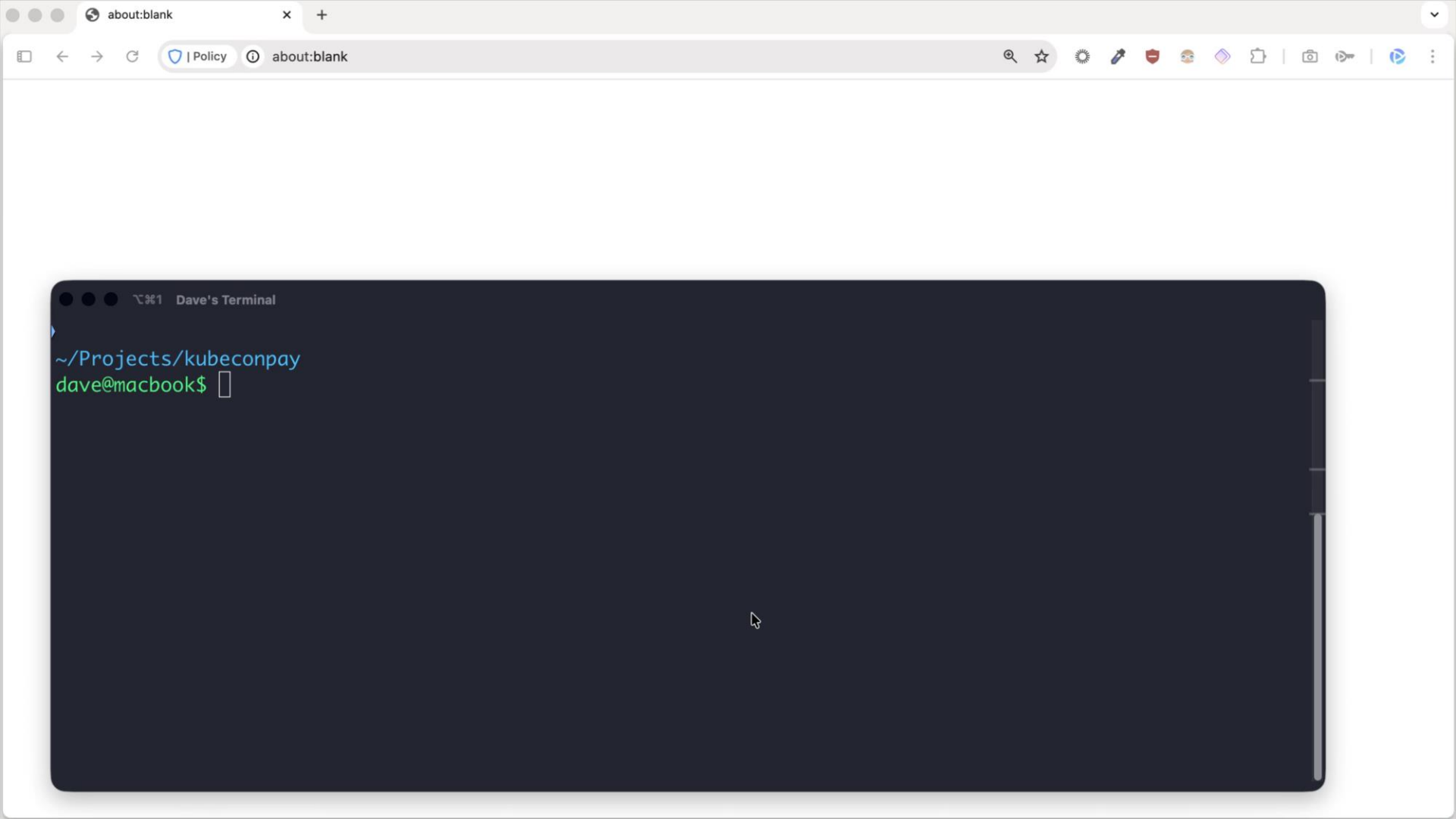


Meet Dave, our Junior Developer



But then the first task is flying in...





about:blank

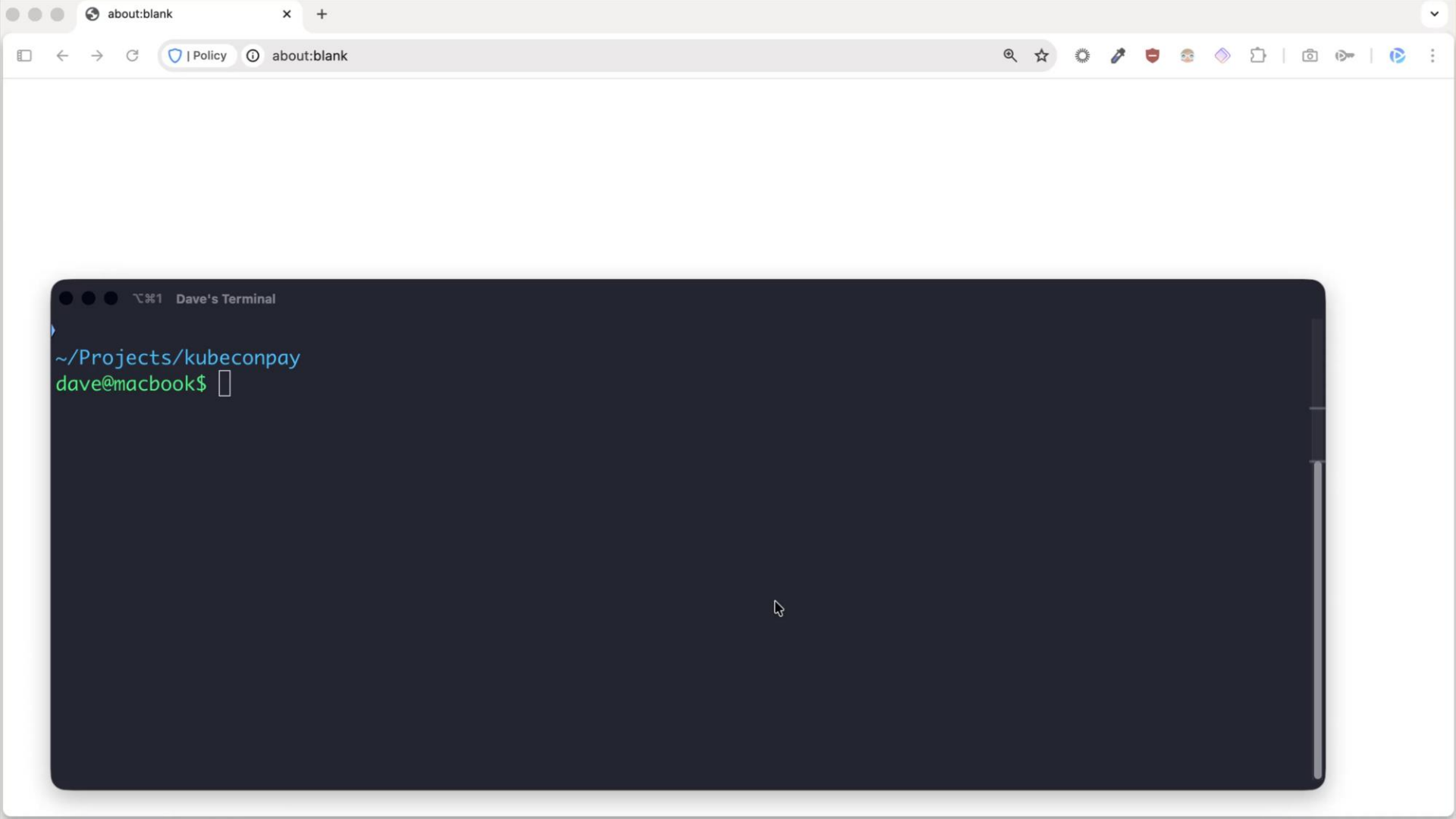
about:blank

Policy

Dave's Terminal

~/Projects/kubeconpay

dave@macbook\$



about:blank

x

+

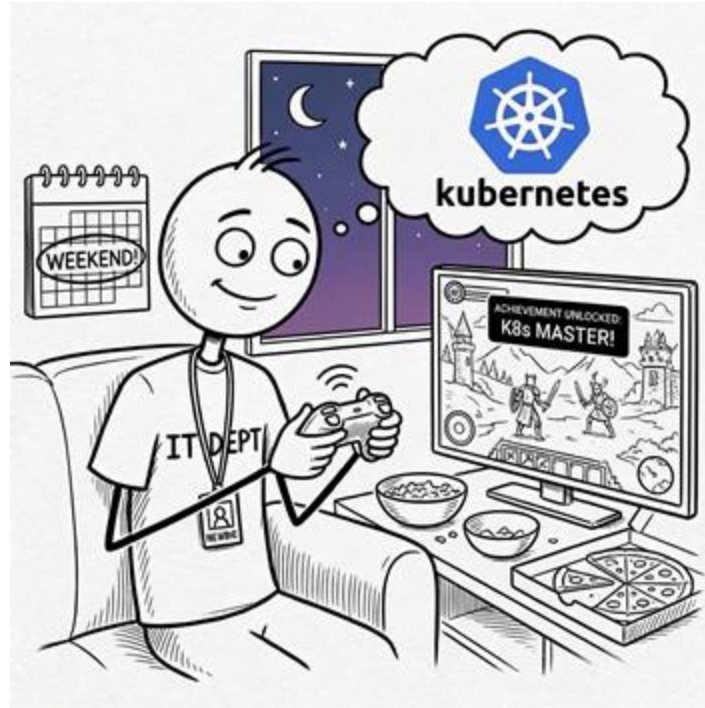
Policy about:blank

Search Star Settings Extensions

Dave's Terminal

~/Projects/kubeconpay
dave@macbook\$

... Dave finally deserves his gaming weekend!





KubeCon



CloudNativeCon

Europe 2026

ACT II: The 🍌 hits the fan



So, on Monday...



The open door (Initial Access)

Attack Phase 1/3



```
apiVersion: v1
kind: Service
spec:
  type: LoadBalancer # ← public IP, anyone
  can reach it
  ports:
  - port: 80
```

+ vulnerable web app
= bad



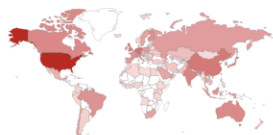
It's the #1 way hackers get in. **33%** of all intrusions start with a door left open just like this one!

Finding K8s related services in real life is easy



TOTAL RESULTS
1,385,004

TOP COUNTRIES



United States	839,410
Belgium	80,976
Germany	50,434
China	43,427
Netherlands	39,063

[More...](#)

TOP PORTS

443	1,225,952
6443	63,151
9443	1,488

View Report

Browse Images

View on Map

Advanced Search

Product Spotlight: We've Launched a new API for Fast Vulnerability Lookups. Check out [CVEDB](#)

35.187.254.123

123.254.187.35.bc.googleusercontent.com

Google LLC

Singapore, Singapore

[cloud](#) [devops](#)

SSL Certificate

Issued By:

|- Common Name:

75aa926a-6dd2-4750-b4b6-a5902ef6018e

Issued To:

|- Common Name:

34.87.127.24

Supported SSL Versions:

TLSv1.2, TLSv1.3

HTTP/1.1 403 Forbidden

Audit-Id: 22fd5456-7dfb-45e8-8bfa-9e23c92dffc0

Cache-Control: no-cache, private

Content-Type: application/json

X-Content-Type-Options: nosniff

X-Kubernetes-Pf-Flowschema-Uid: b8292035-5793-41b0-b845-85afa36fc2d7

X-Kubernetes-Pf-Prioritylevel-Uid: d2b149f7-0897-49c7-a8...

2026-03-23T21:27:35.562404

34.45.149.85

85.149.45.34.bc.googleusercontent.com

Google LLC

United States, Council Bluffs

[cloud](#) [devops](#) [eol-product](#)

SSL Certificate

Issued By:

|- Common Name:

1c3f80b7-4c1d-45ca-bf20-f474f6298c89

Issued To:

|- Common Name:

35.222.94.149

Supported SSL Versions:

TLSv1.2, TLSv1.3

HTTP/1.1 403 Forbidden

Audit-Id: 0aa211c6-07cc-41b0-af1f-3cb51c24b0f7

Cache-Control: no-cache, private

Content-Type: application/json

X-Content-Type-Options: nosniff

X-Kubernetes-Pf-Flowschema-Uid: 5dc08a4c-e5f4-4fce-aac4-5817864ee16d

X-Kubernetes-Pf-Prioritylevel-Uid: d739a83f-b62e-4fb6-b4...

2026-03-23T21:27:34.162723



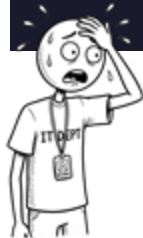
```
~  
> nmap -sV -p 80,443,8080,6443,10250 34.53.208.125
```

The keys under the doormat (Credential Access)

Attack Phase 2/3

INSIDE EVERY POD, BY DEFAULT

```
/var/  
├── run/  
├── secrets/  
├── kubernetes.io/  
├── serviceaccount/ AUTO-MOUNTED  
├── ca.crt cluster certificate  
├── namespace current namespace  
└── token API access credential
```



Over 70% of us leave this 'key' under the mat without any protection!

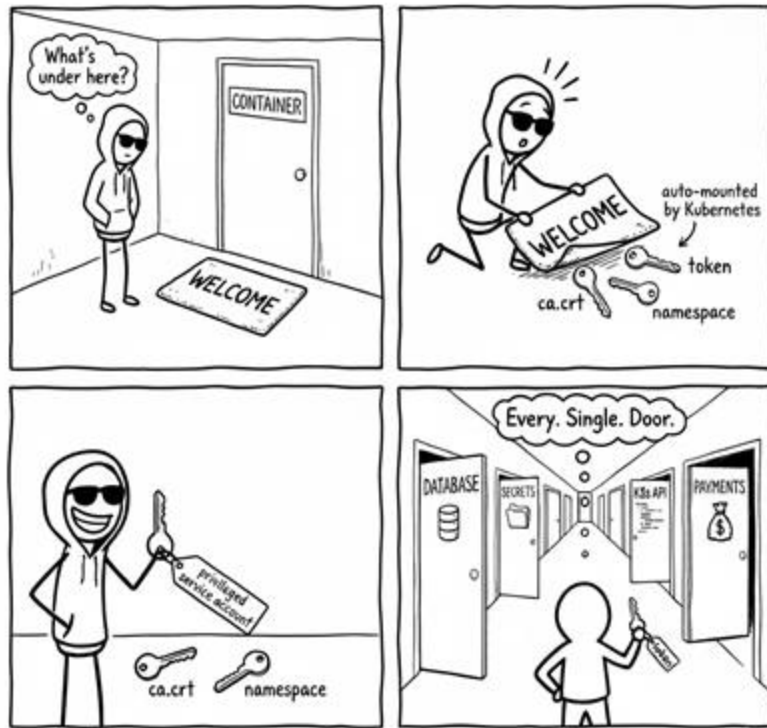


KubeCon



CloudNativeCon

Europe 2026



⚠ PCI-DSS Level 1 — All customer data access requires Payment Gateway authentication. Do not share this URL.

Customer Lookup

Enter the customer's name, email address, or account number to pull up their profile.

Search for a customer

Look up

Search matches customer names, email addresses, and account numbers in the Payment Gateway.

⚠ Unexpected server response

kubeconpay-d57457cff-twgrp

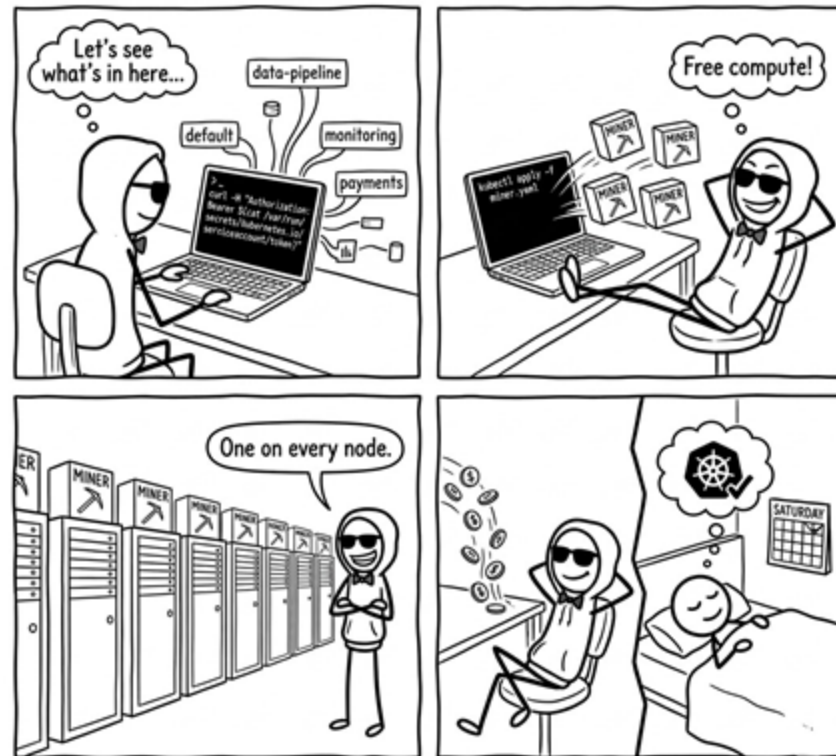
```
KUBECONPAY_PORT_80_TCP_ADDR=34.118.239.170
KUBERNETES_PORT=tcp://34.118.224.1:443
KUBERNETES_SERVICE_PORT=443
KUBECONPAY_PORT_80_TCP_PORT=80
HOSTNAME=kubeconpay-d57457cff-twgrp
KUBECONPAY_PORT_80_TCP_PROTO=tcp
HOME=/root
GPG_KEY=A035C8C19219BA821ECA86B64E628F8D684696D
PYTHON_SHA256=272179ddd9a2e41a0fc8e42e33dfbdca0b3711aa5abf372d3f2d51543
d09b625
WERKZEUG_SERVER_FD=3
KUBECONPAY_PORT_80_TCP=tcp://34.118.239.170:80
KUBERNETES_PORT_443_TCP_ADDR=34.118.224.1
PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/
sbin:/bin
KUBERNETES_PORT_443_TCP_PORT=443
```

Under new management (Execution & Impact)

Attack Phase 3/3

```
# Attacker uses the stolen token from the search box
curl -sk -H "Authorization: Bearer $(cat
/var/run/secrets/.../token)" \
https://kubernetes.default.svc/api/v1/namespaces
```

```
# Attacker deploys cryptominer on all nodes and
hides it as "system-monitor"
curl -sk -X POST -H "Authorization: Bearer ..." \
.../apis/apps/v1/.../daemonsets \
-d '{"kind": "DaemonSet",
  "metadata": {"name": "system-monitor"},
  ...
  "image": "alpine:3.19"}'
```



Dave's Terminal

~ 8s

> kubectl get pods

NAME	READY	STATUS	RESTARTS	AGE
kubeconpay-d57457cff-twgrp	1/1	Running	0	15h

~

> kubectl get pods

NAME	READY	STATUS	RESTARTS	AGE
kubeconpay-d57457cff-twgrp	1/1	Running	0	24h
system-monitor-c7k6h	1/1	Running	0	42s
system-monitor-gfnxq	1/1	Running	0	42s
system-monitor-xfm7g	1/1	Running	0	42s

~

>

Dave is not alone



The Deployment Fatigue

In the struggle to master new concepts, **"making it work"** often wins the race against **"making it secure."**



I barely hit 'deploy'!
89% of organizations have been hit this year! Automated scanning find exposed cluster within minutes!



KubeCon



CloudNativeCon

Europe 2026

ACT III: It is secure!

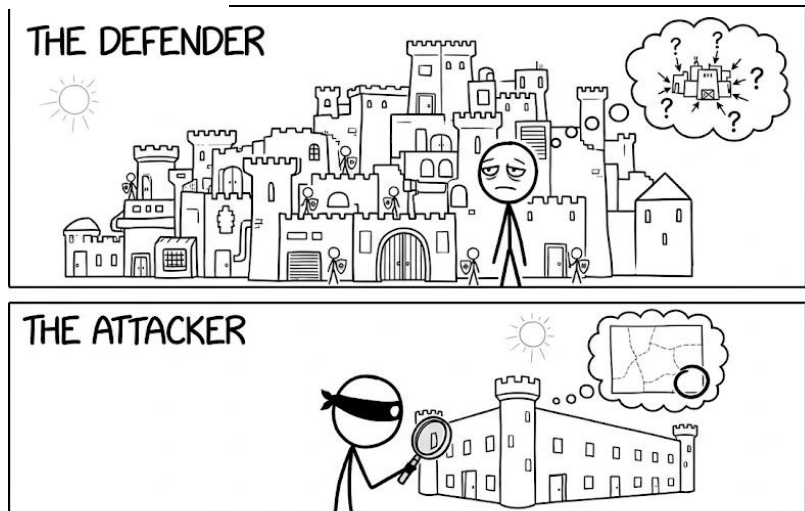


Don't forget the Fundamentals

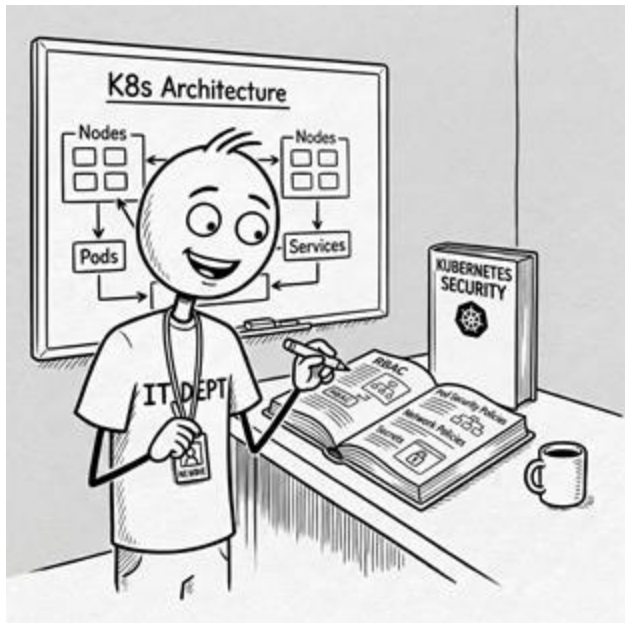
Security is a Spectrum

The Attacker's Advantage

Layered Security/ Defense in Depth



How Dave could have avoided this



-  Limit the attack surface
-  Least privilege access & RBAC
-  Contain workloads and limit impact

Lock the cluster door!

Check IDs



```
# kube-apiserver.yaml
spec:
  containers:
    - command:
      - kube-apiserver
      - --anonymous-auth=false
```

Build a Fence



Private Clusters

(Hide the door)

IP Allowlists

(Choose who enters)

VPN / Bastion Hosts

(The secure tunnel)

Patrol the Perimeter



Scan for Vulnerabilities



Don't Leave the Keys Under the Doormat!

```
# Disable automatic service account mounting
apiVersion: v1
kind: Pod
metadata:
  name: novapay
spec:
  automountServiceAccountToken: false
  containers:
  - name: novapay
    image: nginx
```

Hardening [↗](#)

Kubernetes defaults to providing access which may not be required in every cluster. Reviewing the RBAC rights provided by default can provide opportunities for security hardening. In general, changes should not be made to rights provided to `system:` accounts some options to harden cluster rights exist:

- Review bindings for the `system:unauthenticated` group and remove them where possible, as this gives access to anyone who can contact the API server at a network level.
- Avoid the default auto-mounting of service account tokens by setting `automountServiceAccountToken: false`. For more details, see [using default service account token](#). Setting this value for a Pod will overwrite the service account setting, workloads which require service account tokens can still mount them.

Source: <https://kubernetes.io/docs/concepts/security/rbac-good-practices/>

The "ID Badge" (Least Privilege RBAC)

The Identity (Who)



```
# Dave just created an "Identity"
apiVersion: v1
kind: ServiceAccount
metadata:
  name: dave-admin
  namespace: default
```

The Privileges (What)

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: dave-admin-binding
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: dave-admin
  namespace: default
```

The "ID Badge" (Least Privilege RBAC)

The Identity (Who)



```
# 1. Create the "Identity"
apiVersion: v1
kind: ServiceAccount
metadata:
  name: novapay-svc
  namespace: default
```

The Bridge

```
# 3. Link the Identity to the NEW Role
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: novapay-read-binding
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: novapay-reader
subjects:
- kind: ServiceAccount
  name: novapay-svc
  namespace: default
```

The Privileges (What)

```
# 2. CREATE THE "ROLE" (Dave skipped this!)
# This defines the "Limited Badge"
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: novapay-reader
  namespace: default
rules:
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["get", "list"] # <-- ONLY "LOOK", NO "TOUCH"
```

Contain workloads, limit impact!

```
# Pod Security – one block to rule them all
apiVersion: v1
kind: Pod
metadata:
  name: novapay
spec:
  automountServiceAccountToken: false
  containers:
  - name: novapay
    image: novapay:v3
    securityContext:
      runAsNonRoot: true
      runAsUser: 1000
```

No root access

App runs as a regular user, not an admin.

`runAsNonRoot: true`

Contain workloads, limit impact!

```
# Pod Security – one block to rule them all
apiVersion: v1
kind: Pod
metadata:
  name: novapay
spec:
  automountServiceAccountToken: false
  containers:
  - name: novapay
    image: novapay:v3
    securityContext:
      runAsNonRoot: true
      runAsUser: 1000
      readOnlyRootFilesystem: true
```

No root access

App runs as a regular user, not an admin.

`runAsNonRoot: true`

Read-only filesystem

Attackers can't write malware to disk.

`readOnlyRootFilesystem: true`

Contain workloads, limit impact!

```
# Pod Security – one block to rule them all
apiVersion: v1
kind: Pod
metadata:
  name: novapay
spec:
  automountServiceAccountToken: false
  containers:
  - name: novapay
    image: novapay:v3
    securityContext:
      runAsNonRoot: true
      runAsUser: 1000
      readOnlyRootFilesystem: true
      allowPrivilegeEscalation: false
```

No root access

App runs as a regular user, not an admin.

`runAsNonRoot: true`

Read-only filesystem

Attackers can't write malware to disk.

`readOnlyRootFilesystem: true`

No privilege escalation

Even if they're inside, they can't promote themselves to admin.

`allowPrivilegeEscalation: false`

Contain workloads, limit impact!

```
# Pod Security – one block to rule them all
apiVersion: v1
kind: Pod
metadata:
  name: novapay
spec:
  automountServiceAccountToken: false
  containers:
  - name: novapay
    image: novapay:v3
    securityContext:
      runAsNonRoot: true
      runAsUser: 1000
      readOnlyRootFilesystem: true
      allowPrivilegeEscalation: false
      capabilities:
        drop: ["ALL"]
```

No root access

App runs as a regular user, not an admin.

```
runAsNonRoot: true
```

Read-only filesystem

Attackers can't write malware to disk.

```
readOnlyRootFilesystem: true
```

No privilege escalation

Even if they're inside, they can't promote themselves to admin.

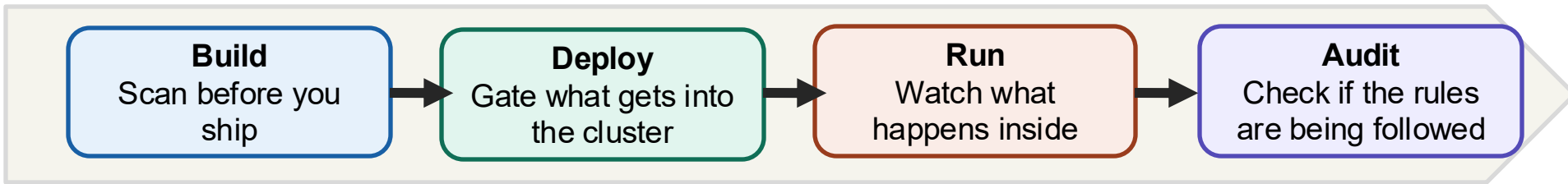
```
allowPrivilegeEscalation: false
```

Drop all capabilities

Strip away every special power the container could have.

```
capabilities: drop: ["ALL"]
```

Dave's Security Toolkit



Open Source



Image Scanning



Policy Engine



Runtime detection



CIS benchmarks



Audit Logging

kubectl-who-can
RBAC Query tool

Enterprise Platforms

Nutanix NKP
Production-ready

Cortex Cloud
Full CNAPP

Dave's story was fictitious but...

Something similar happened 3 months ago!

DAY 0: VULNERABILITY DISCLOSED

**Automated scanning
begins instantly**

Same as Dave's
exposed app

DAY 2: SOME ATTACKS GOT THROUGH

**Some attackers achieved
code execution**

Similar chain we
just demoed

LAYER 3 - NETWORK

**Segmentation blocked
outbound connection**

Contain the
workloads; limit
impact!

WALL 1: PERIMETER

**Firewalls +
WAF blocked
most attempts**

Lock the door

WALL 2: ENDPOINT

**Runtime Security
caught suspicious
commands**

Detect the intruder



KubeCon

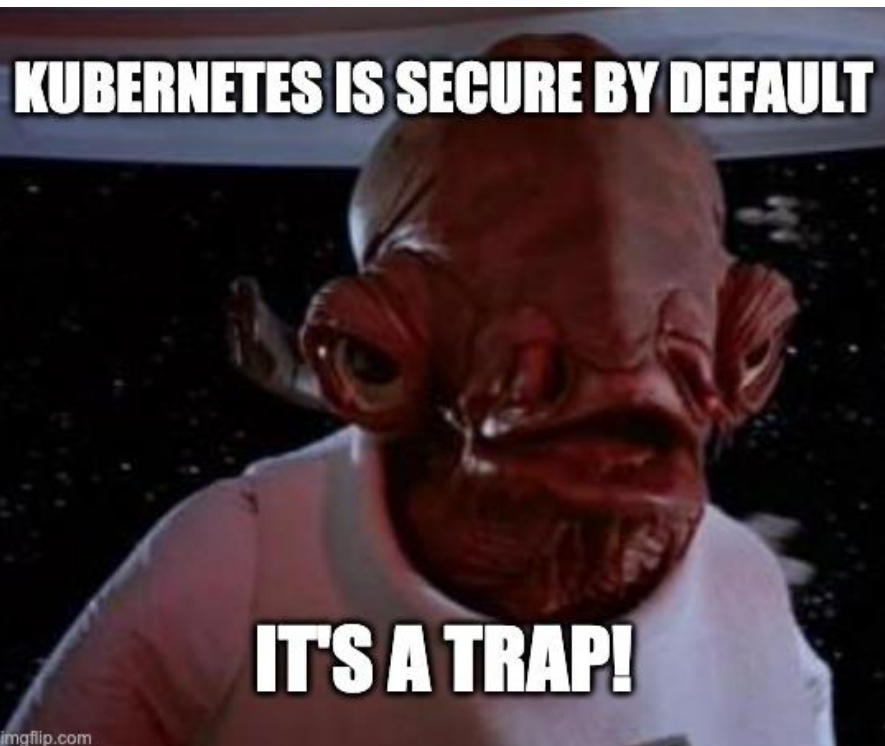


CloudNativeCon

Europe 2026

**“In this ecosystem, there are two types of people:
Those who learn from Dave’s mistakes, and those who
haven't been hacked yet.”**

Be the first kind.





KubeCon



CloudNativeCon

Europe 2026

Q&A

Thank you very much!

