~/dc34/title

Disclaimer: This research was conducted independently. The views and opinions expressed in this work are solely those of the authors and do not reflect the views or positions of our employers.

DEF CON 34

No Socket, No Privs, No Problem.

Weaponizing OCI registries for SSRF, Credential Theft & Container Escapes

David Rochester (@davidrxchester) · Nicholas Gould (@gouldnicholas)

What even is an OCI Registry?

- HTTP API that stores and serves artifacts, most commonly container images and model files

- Like a web server hosting any other content

- Available Artifacts described by a manifest

The manifest is the menu

TAG

registry.io/model:latest

→

MANIFEST.JSON

```
{
  "config": { "digest": "sha256:9f2a…" },
  "layers": [
    { "digest": "sha256:a3ed…" },
    { "digest": "sha256:71bc…" }
  ]
}
```

→

BLOBS

- config JSON
- layer 128 MB
- layer 42 MB

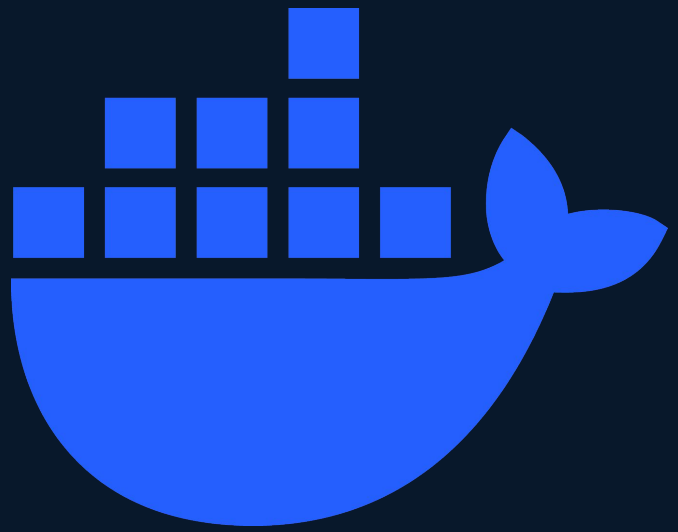
The manifest just lists **digests available on the server**.

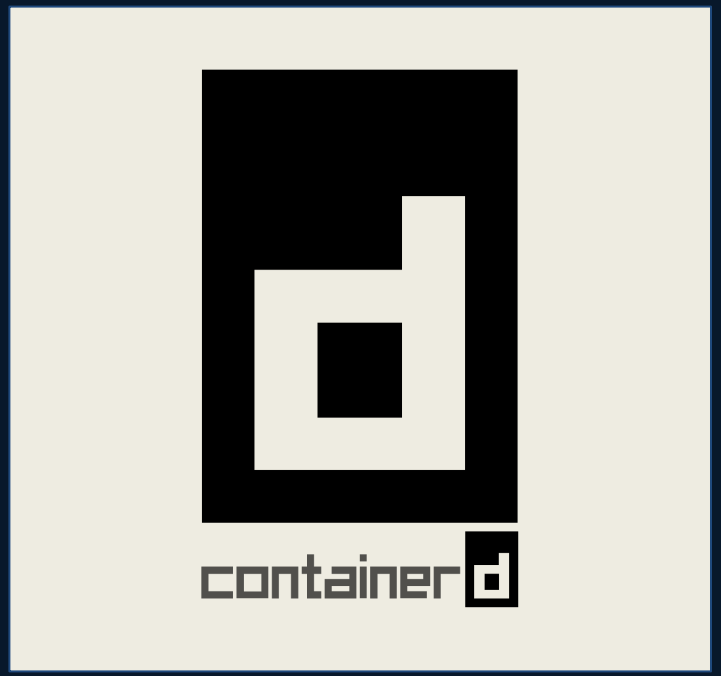
Communicate over the OCI protocol

```
GET    /v2/                                  # is this a registry?
GET    /v2/<name>/manifests/<ref>            # fetch a manifest
PUT    /v2/<name>/manifests/<ref>            # publish a manifest
GET    /v2/<name>/blobs/<digest>             # download a blob
POST   /v2/<name>/blobs/uploads/             # upload a blob
```

Example to download model blob:
curl -sSL https://registry.io/v2/model/blobs/sha256:9f2a3c8d…e41 -o model.gguf

Most software implements their own OCI client





Ollama

- Exposes APIs with zero authentication
- Commonly bound to 0.0.0.0 to facilitate access
- Users can
  - pull models
  - push models
  - request inference

```
POST   /api/generate    # run inference
POST   /api/chat        # chat completions
GET    /api/tags        # list local models
POST   /api/pull        # pull a model from any registry we name
POST   /api/push        # push a model to a registry
DELETE /api/delete      # delete a model
```

NO SOCKET, NO PRIVS, NO PROBLEM // OCI

Docker Model Runner

- Introduced to Docker Desktop in March 2025
- Run models locally and interact with service from containers
- Reachable from any container via model-runner.docker.internal
- Reachable locally on port 12434

```
POST   /engines/v1/chat/completions  # inference
POST   /engines/v1/embeddings  # embeddings
GET    /models  # list models
POST   /models/create  # pull from any registry
GET    /models/{ns}/{name}  # inspect
DELETE /models/{ns}/{name}  # delete
```

DMR - Pulling a Model

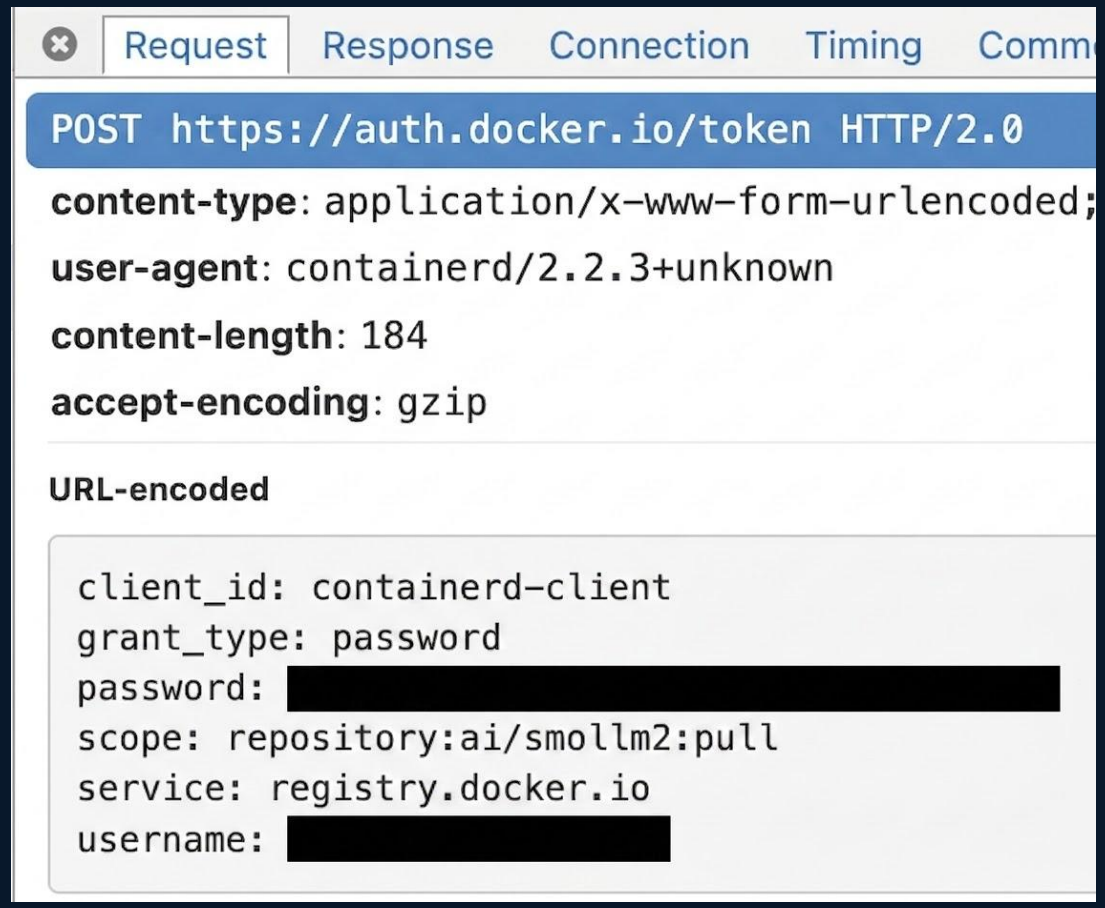
```
~ curl -s -X POST http://localhost:12434/models/create \
-H "Content-Type: application/json" -d '{"from":"ai/smollm2"}'
```

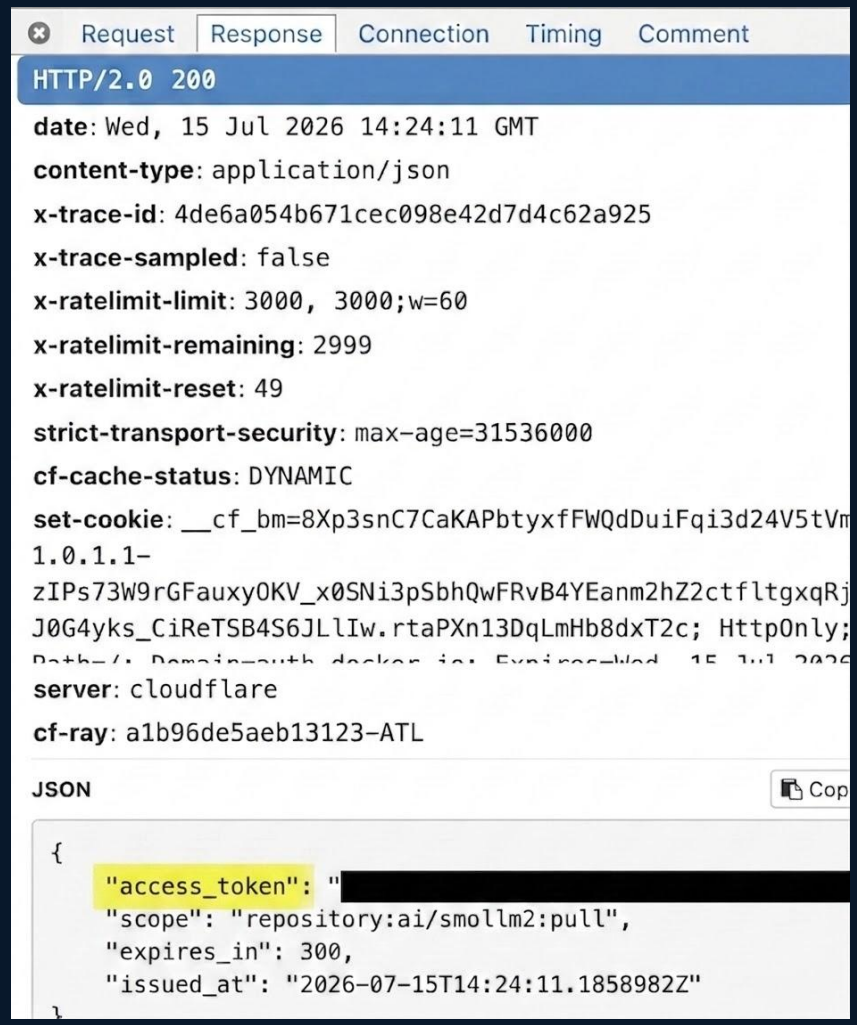
Request | Response | Connection | Timing | Comment

```
HEAD https://registry-1.docker.io/v2/ai/smollm2/manifests/latest HTTP/2.0
accept: application/vnd.docker.distribution.manifest.v2+json,
application/vnd.docker.distribution.manifest.list.v2+json,
application/vnd.oci.image.manifest.v1+json, application/vnd.oci.image.index.v1+json,
*/*
```

```
HTTP/2.0 401
date: Wed, 15 Jul 2026 14:24:10 GMT
content-type: application/json
content-length: 153
docker-distribution-api-version: registry/2.0
www-authenticate: Bearer
realm="https://auth.docker.io/token",service="registry.docker.io",scope="repository:
ai/smollm2:pull"
docker-ratelimit-source: 2600:6c5e:1340:1::
strict-transport-security: max-age=31536000
```

DMR - Pulling a Model cont.





DMR- Pulling a Model cont.

Request | Response | Connection | Timing | Comment

```
GET https://registry-1.docker.io/v2/ai/smollm2/manifests/sha256:354bf30d0aa3af413d2aa5ae4f23c66d78980072d1e07a5b0d776e9606a2f0b9 HTTP/2.0
```

user-agent: Docker-Desktop/4.74.0 (Mac; arm64)

accept: application/vnd.oci.image.manifest.v1+json, */*

accept-encoding: zstd;q=1.0, gzip;q=0.8, deflate;q=0.5

authorization: Bearer
eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCIsIng1YyI6WyJNSUlFRmpDQ0F2NmdBd0lCQWdJVU9yTFd5UVpx
MmFuZXd6WnhYN1JLbHQ3bDVUTXdEUVlKS29aSWh2Y05BUUVMQlFBd2dZWXhDekFKQmdOVkJBWVRBbFZUTVJN
d0VRWURWUVFJRXdwRFlXeHBabTl5Ym1saE1SSXdFQVlEVlFRSEV3bFFZV3h2SUVGc2RHOHhGVEFUQmdOVkJB

No content | Edit | Replace | View: auto

Request | Response | Connection | Timing | Comment

```
HTTP/2.0 200
```

date: Wed, 15 Jul 2026 14:24:11 GMT

content-type: application/vnd.oci.image.manifest.v1+json

content-length: 551

docker-content-digest: sha256:354bf30d0aa3af413d2aa5ae4f23c66d78980072d1e07a5b0d776e9606a2f0b9

docker-distribution-api-version: registry/2.0

etag: "sha256:354bf30d0aa3af413d2aa5ae4f23c66d78980072d1e07a5b0d776e9606a2f0b9"

docker-ratelimit-source: davidrochester1

x-ratelimit-limit: 200;w=3600

ratelimit-limit: 200;w=3600

x-ratelimit-remaining: 198;w=3600

ratelimit-remaining: 198;w=3600

strict-transport-security: max-age=31536000

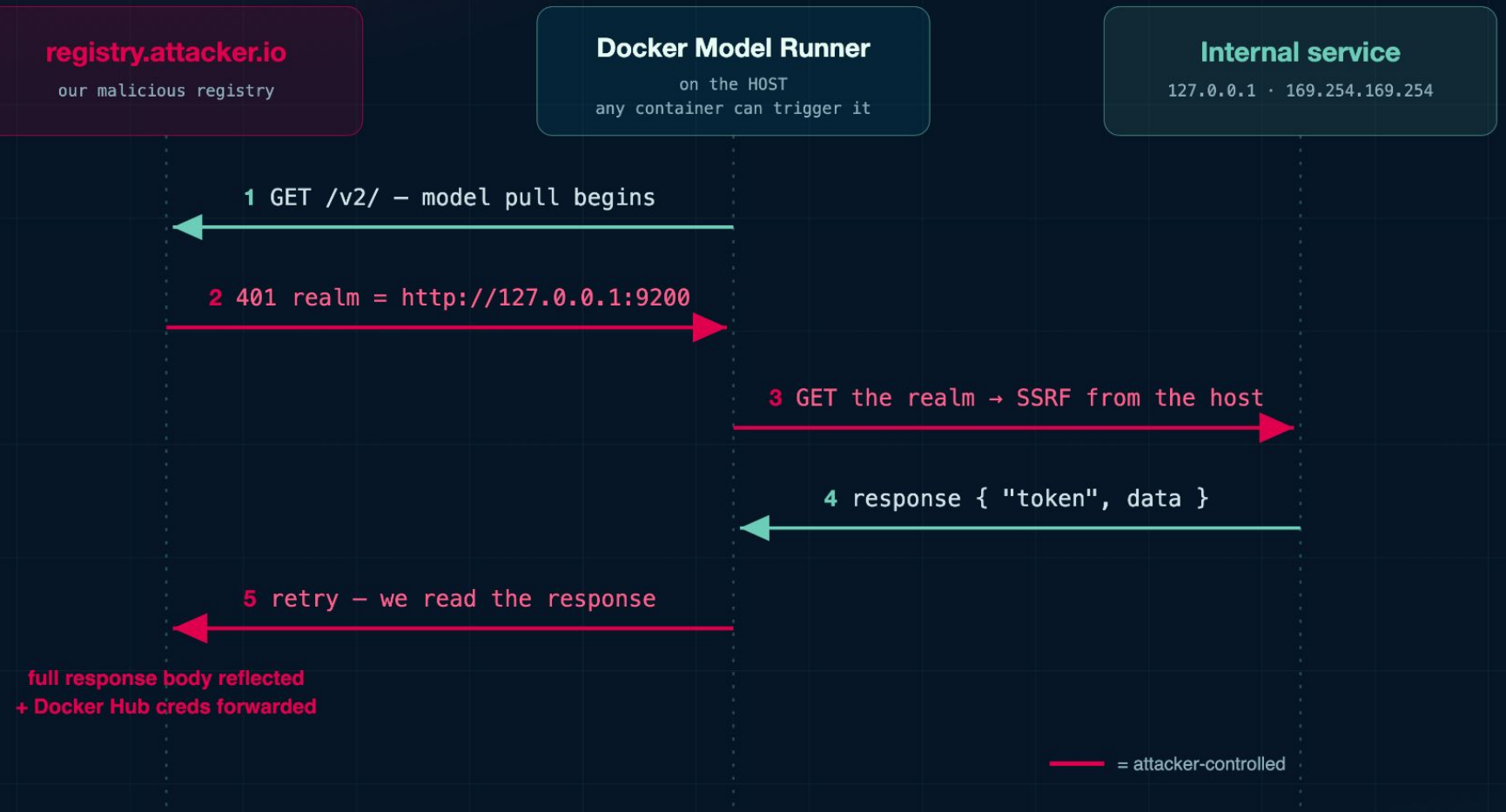
JSON | Copy | Edit | Replace | View: auto


```
{
    "schemaVersion": 2,
    "mediaType": "application/vnd.oci.image.manifest.v1+json",
    "config": {
        "mediaType": "application/vnd.docker.ai.model.config.v0.1+json",
        "size": 375,
        "digest": "sha256:32aaa72e00e9a79d38786abd1e37388e7f01181a0f1bdf23efff6433c12f88
    }
```

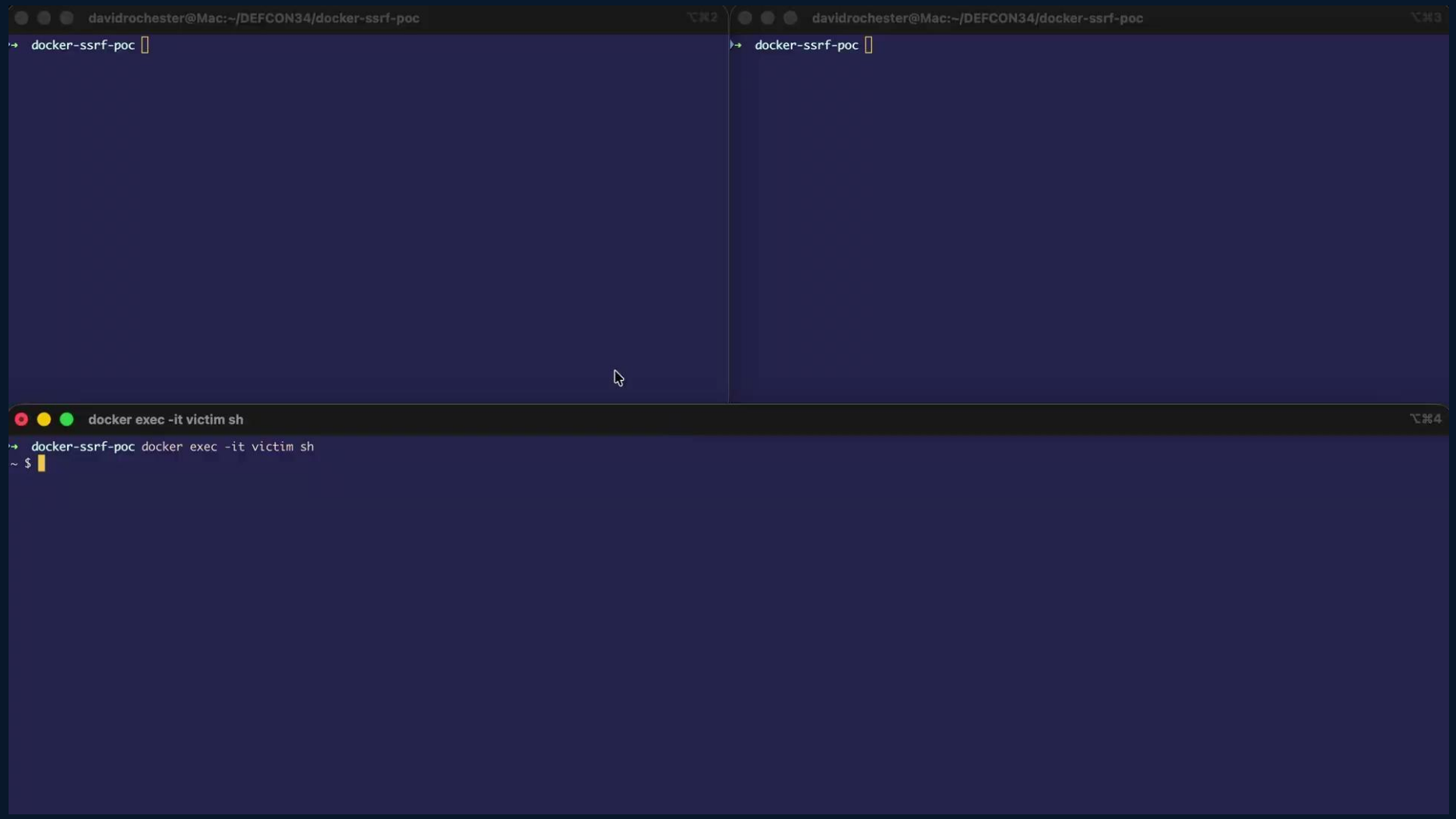
DMR - Pulling a Model cont.

```
HTTP/2.0 401
date: Wed, 15 Jul 2026 14:24:10 GMT
content-type: application/json
content-length: 153
docker-distribution-api-version: registry/2.0
www-authenticate: Bearer
realm="https://auth.docker.io/token",service="registry.docker.io",scope="repository:
ai/smollm2:pull"
docker-ratelimit-source: 2600:6c5e:1340:1::
strict-transport-security: max-age=31536000
```

Abusing the Auth Flow



DMR - SSRF Demo



Hunting SSRF in Ollama

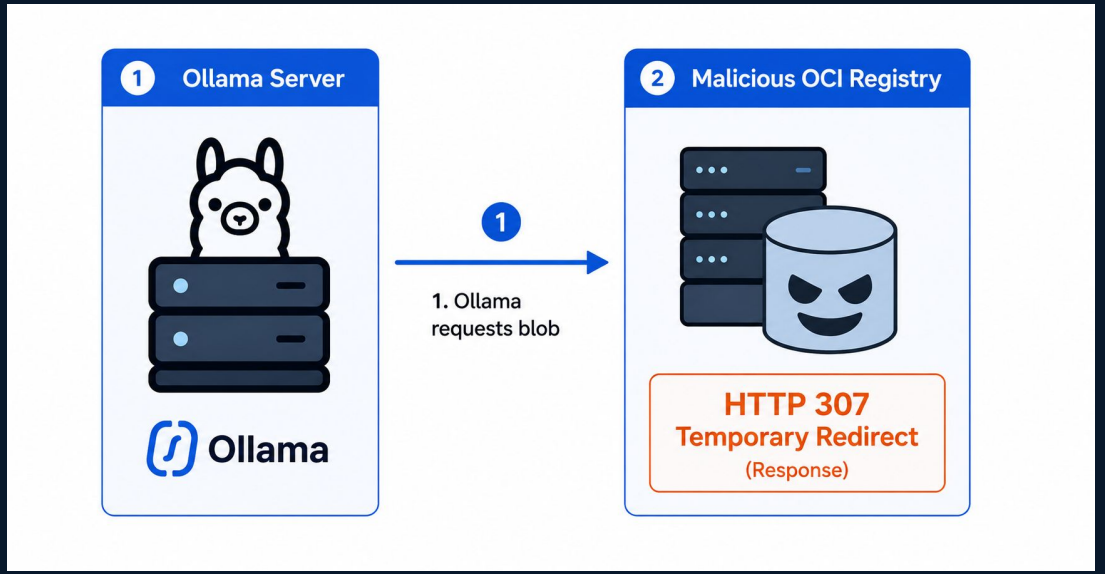
- Ollama validates realm before sending tokens cross-origin

```
// Validate that the realm host matches the original request host to prevent sending tokens cross-origin.
if redirectURL.Host != originalHost {
    return "", fmt.Errorf("realm host %q does not match original host %q", redirectURL.Host, originalHost)
}
```

`server/auth.go`

Hunting SSRF in Ollama

- 307 Redirects




```
if resp.StatusCode != http.StatusTemporaryRedirect && resp.StatusCode != http.StatusOK {
    return nil, fmt.Errorf("unexpected status code %d", resp.StatusCode)
}
return resp.Location()
```

`server/download.go`

HASH VERIFY · THE BYPASS

One digest, twice. Verification skipped.

01

Malicious Manifest

Two layers · same digest

sha256:a1b2...

02

downloadBlob()

307 → SSRF body written

03

digest	skipVerify
a1b2...	~~false~~ → **true**
c9d0...	false

skipVerify Map

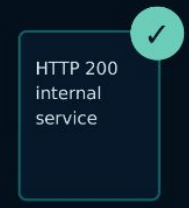
Cache hit overwrites the flag

04

verifyBlob()

Skipped — flag is true

05



Blob on Disk

Internal response persists

```
skipVerify[layer.Digest] = cacheHit
```

LOGIC FLAW A cache hit overwrites verification state — the unverified blob stays on disk.

02

ATTACK CLASS

ATTACK CLASS

Credential Theft

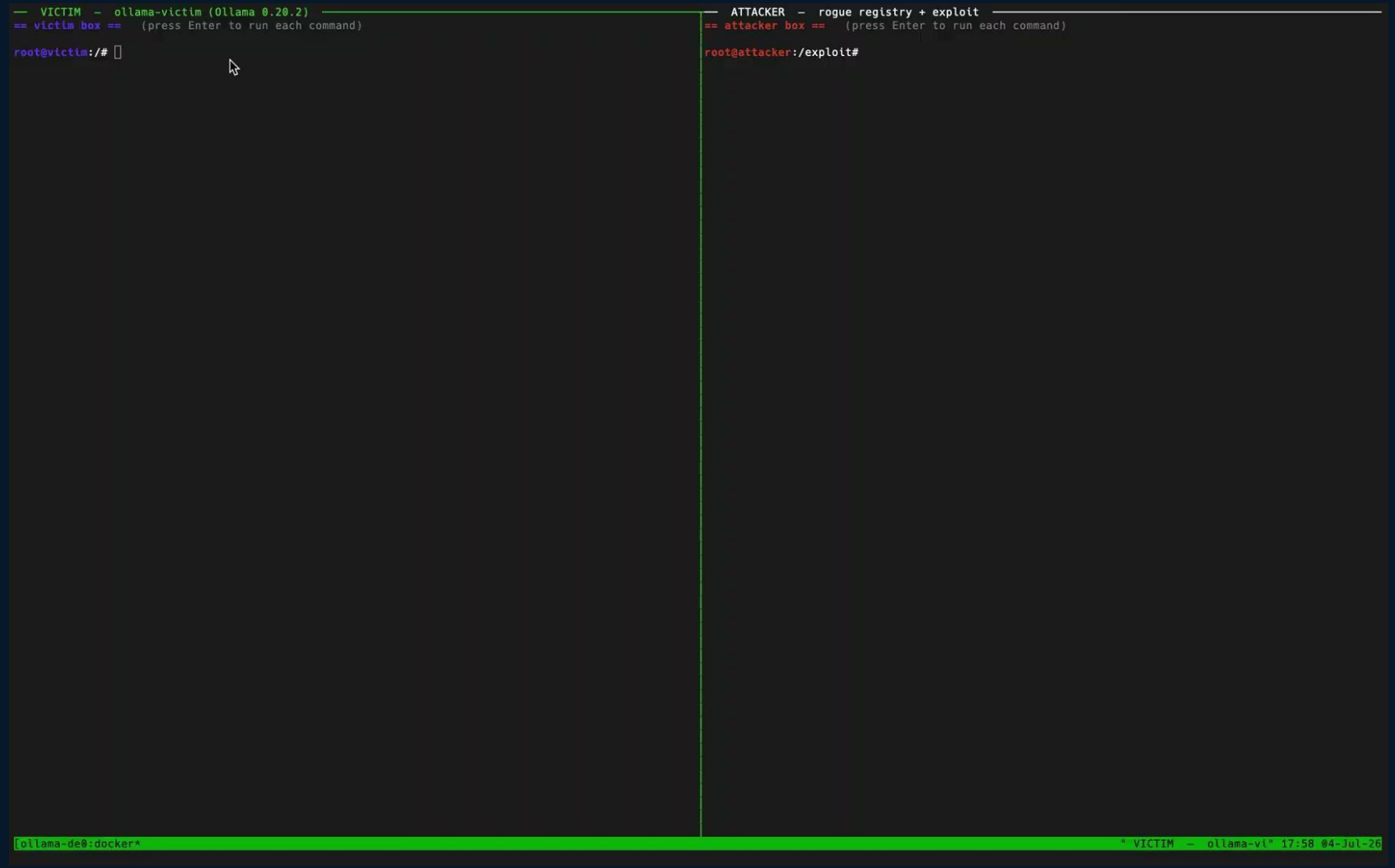
Access Galore

Ex: (O)llama Whisper

Ex: (O)llama Whisper

- Attacker reads any file on the server (e.g. /etc/shadow, SSH keys)
- No login required: Ollama has no auth by default
- Ollama commonly listens on 0.0.0.0
- Bug: a file "digest" isn't checked for ../, so the path escapes its folder
- Trigger: three normal API calls with a booby-trapped digest
- In Docker = runs as root ⟶ reads the whole filesystem
- Unpatched : works on latest release (0.31.1) and main, disclosed to Ollama multiple times

Ex: (O)llama Whisper PoC



Ex: (O)llama Whisper Current State

- Disclosed to Ollama several times (starting April 5 2026)
- received a CVE
- Publicly Exposed Ollama servers:
    - Jan 2026: 175,000 over 130 countries

03

ATTACK CLASS

ATTACK CLASS

Container Escape

Abusing Docker Model Runner’s inference backends

Inference backends

llama.cpp

GGUF · Metal

vLLM

Metal backend

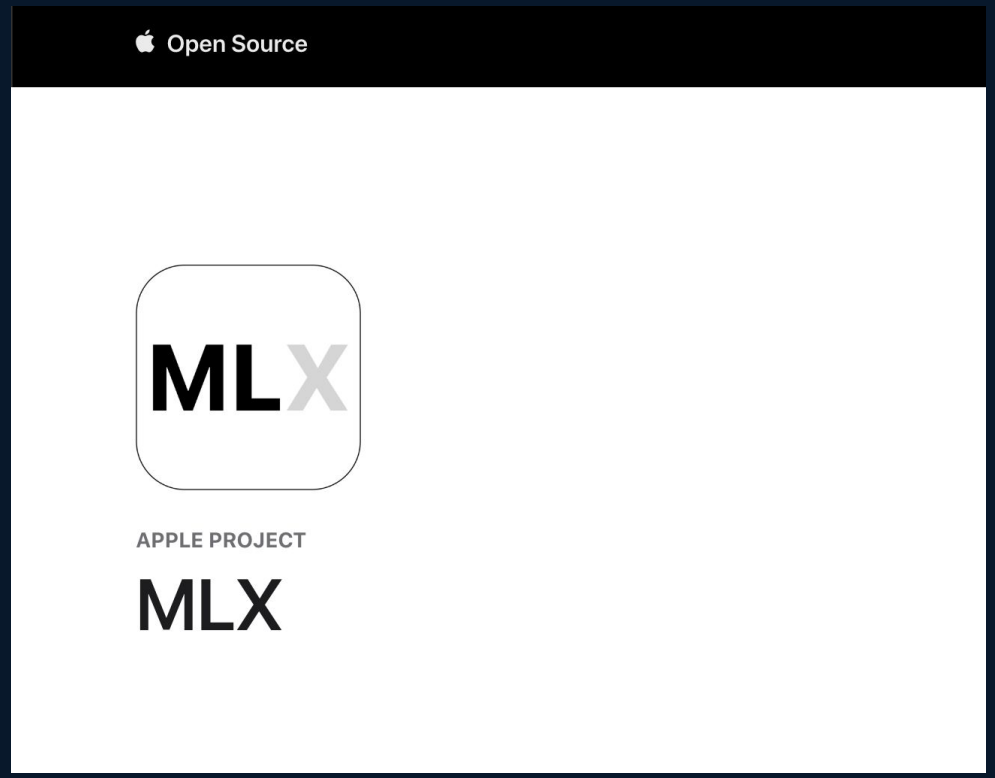
MLX

mlx-lm · Apple Silicon

native, **Metal-accelerated** inference — runs as a host process

MLX-LM

Apple’s open source library for running LLMs on Apple Silicon



`config.json` gets one extra field:

```json
{
  "model_file": "model.py",
  "architectures": ["LlamaForCausalLM"],
  "model_type": "llama",
  ...
}
```

Docker Model Runner Sandbox

```
return backends.RunBackend(ctx, backends.RunnerConfig{
    BackendName:     "MLX",
    Socket:          socket,
    BinaryPath:      m.pythonPath,
    SandboxPath:     "",
    SandboxConfig:   "",
    Args:            args,
    Logger:          m.log,
    ServerLogWriter: logging.NewWriter(m.serverLog),
})
```

`pkg/inference/backends/mlx/mlx.go`

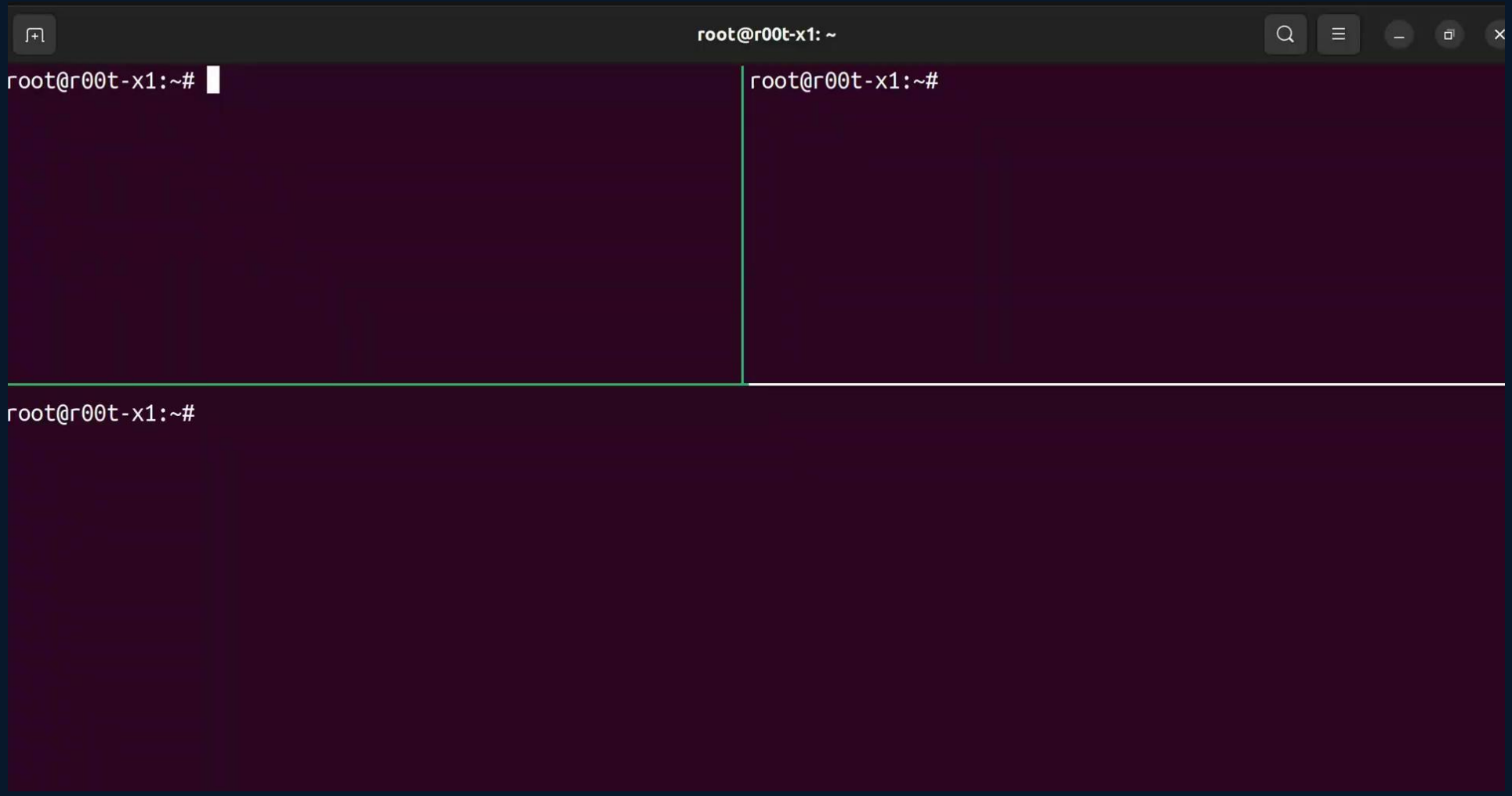
Docker Model Runner Sandbox

```
return backends.RunBackend(ctx, backends.RunnerConfig{
    BackendName:     "MLX",
    Socket:          socket,
    BinaryPath:      m.pythonPath,
    SandboxPath:     "",
    SandboxConfig:   "",
    Args:            args,
    Logger:          m.log,
    ServerLogWriter: logging.NewWriter(m.serverLog),
})
```

Its empty

`pkg/inference/backends/mlx/mlx.go`

Ex: Free Willy...errrr Docker PoC

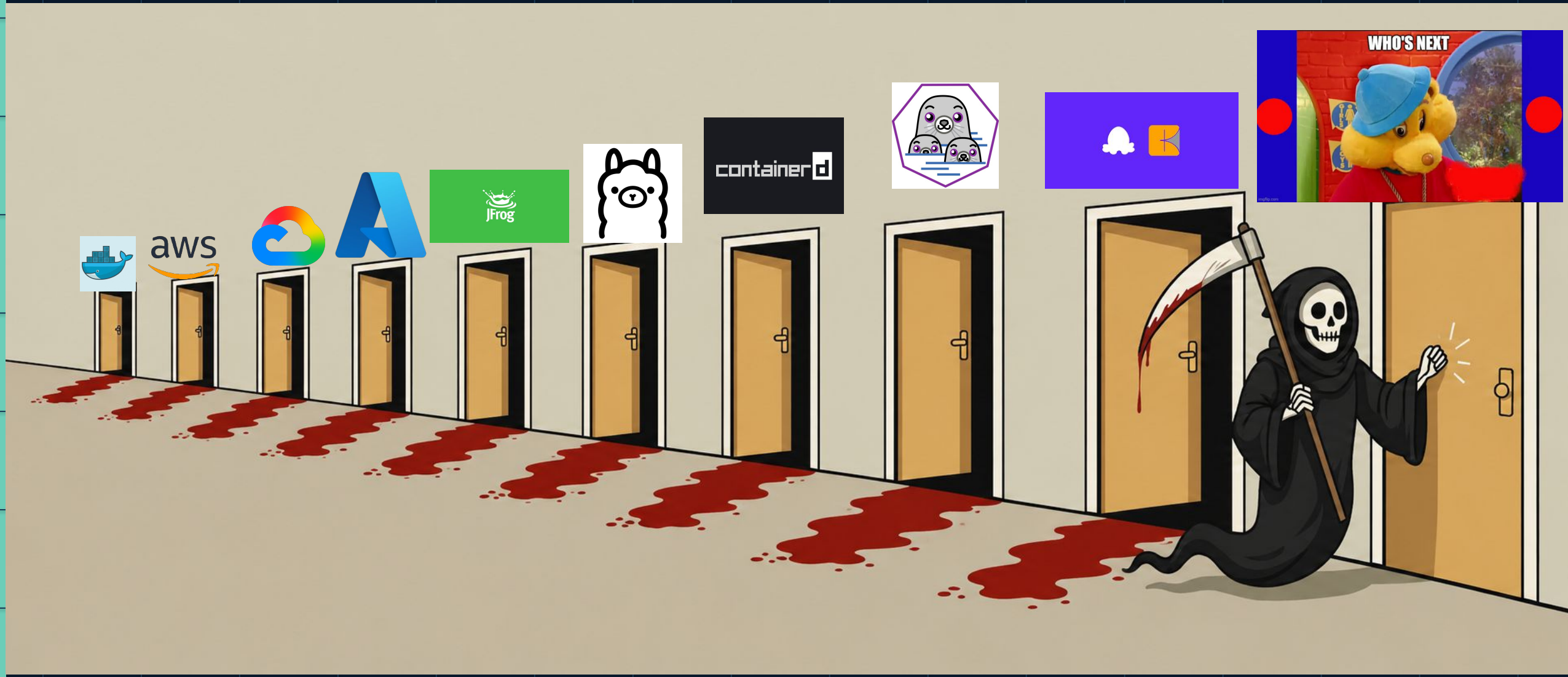


But Wait there's more…

Some of the other OCI primitives we've discovered

- SSRF Family: 3XX redirects, 401 auths, etc
- Producer-controlled metadata XSS Family: stored XSS
- Referrers API / attestation / signature Family: DOS, Injection, bypasses
- Tar layer extraction Family: LFI , RFI, RCE
- Image config / build directive Family: RCE, DOS, RFI, LFI
- Parser / ACL / mediaType / digest differentials Family: injection, data leak, bypasses
- Cross-class compositions: above chained

Turns out, OCI can be used to perpetuate almost any traditional attack type, not just SSRF / RFI. You just need to find a way to embed it into your OCI artifact


WHO'S NEXT
aws
JFrog
containerd

~/dc34/closing

DEF CON 34

No Socket. No Privs.
No Problem.

$ questions?

David Rochester · @davidrxchester · **Nicholas Gould** · @gouldnicholas

References:

- https://www.docker.com/products/docker-desktop/
- https://github.com/docker/model-runner/security
- https://github.com/ml-explore/mlx-lm
- https://github.com/ggml-org/llama.cpp
- https://github.com/vllm-project/vllm-metal
- https://thehackernews.com/2026/01/researchers-find-175000-publicly.html
- https://github.com/ollama
- https://github.com/containerd/containerd
- https://opencontainers.org/