CENTQUATRE-PARIS
5 rue Curial, 75019

Mardi 3 Février
2026



CLOUD NATIVE DAYS
FRANCE 2026

Tokyo Drift Containers : Immuabilité au service de la sécurité Cloud Native

Speaker :
Daniel Maher // Datadog // @phrawzty



Maîtriser la dérive des conteneurs

- Les mythes et la réalité
- Techniques de détection
- Stratégies de remédiation

Qui suis-je ?

- Daniel "*phrawzty*" Maher
- Senior Security Advocate chez Datadog
- Trajet : Admin Sys → DevOps Engineer → Analyste en Sécurité

Le Contrat Cloud Native

- Immuabilité : L'artefact est scellé au build
- Reproductibilité : De la CI au Cluster, rien ne bouge
- Prédictibilité : On sait exactement ce qui tourne

La Réalité : *Drift Happens*

- `kubectl exec` : Le pêché originel du debug
- Les sidecars qui modifient le réseau ou le FS
- Les scripts d'entrypoint qui font des `pip install` ou `apt upgrade`
- Résultat : Votre conteneur en prod est un *snowflake*

Anatomie d'une dérive : la *Kill Chain*

- Recon : Scan de vulnérabilité (e.g. Log4Shell)
- Exploit : Injection de code en mémoire
- Mutation : Téléchargement d'un binaire dans `/tmp`
- Persistance : Modification d'un script pour rester là

1. Shift Left : Forger l'immuabilité

- Réduire la surface d'attaque
- Supprimer les outils de mutation
- Rendre le conteneur "stupide" et focalisé

L'approche "Distroless"

- Pas de gestionnaire de paquets (apt, apk, etc.)
- Pas de Shell (bash, zsh, etc.)
- Uniquement le runtime (Go, Node, JRE, etc. selon le besoin) et l'application elle-même

Visibilité & SBOM


```
"components": [
  {
    "type": "library",
    "bom-ref": "pkg:npm/%40babel/polyfill@7.10.4",
    "group": "@babel",
    "name": "polyfill",
    "version": "7.10.4",
    "description": "Provides polyfills necessary for a full ES2015+ environment",
    "hashes": [
      {
        "alg": "SHA-512",
        "content": "f0161c9d5a90e64303d875e81c89c11fb1f56ffb8fdca767c026173aa1675ea82e3b2baee38dd65eeb1b2146d611f6376744f1934335b86ffc62e00c84d97ace"
      }
    ],
    "licenses": [
      {
        "license": {
          "id": "MIT"
        }
      }
    ],
    "purl": "pkg:npm/%40babel/polyfill@7.10.4",
    "externalReferences": [
      {
        "type": "website"
```

2. Le Runtime : Le moment de vérité

- Kubernetes est-il une passoire ?
- Les limites des outils traditionnels
- Pourquoi l'observabilité doit être *kernel-level*

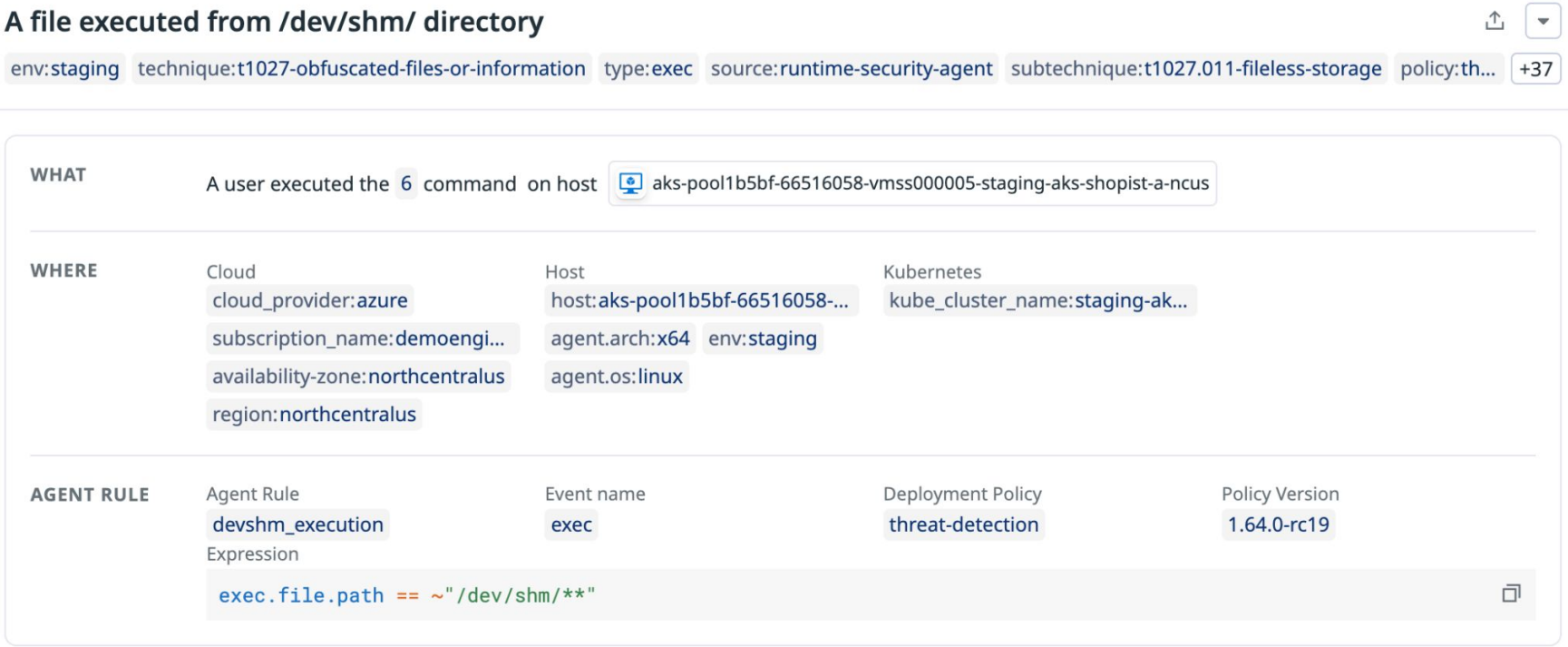
Protection Native Kubernetes

```
securityContext:
  readOnlyRootFilesystem: true
  runAsNonRoot: true
  allowPrivilegeEscalation: false
```

Pourquoi l'eBPF change la donne

- Logs ? Faciles à effacer ou à bypasser, souvent cryptique
- Sidecars ? Lourds, ne voient pas tout
- eBPF ! Directement dans le noyau, faible latence, visibilité absolue sur les syscalls 👍

Détecter la dérive en temps réel



Processus inattendu

```
  "/usr/bin/yash",
  "/bin/zsh5",
  "/bin/zsh5-static" ] ||
 exec.comm in ["wget", "curl", "lwp-download"] ||
 exec.file.path in
["/bin/cat","/bin/chgrp","/bin/chmod","/bin/chown","/bin/cp","/bin/date","/bin/dd","/bin/df","/bin/d
in/echo","/bin/ln","/bin/ls","/bin/mkdir","/bin/mknod","/bin/mktemp","/bin/mv","/bin/pwd","/bin/read
/bin/rm","/bin/rmdir","/bin/sleep","/bin/stty","/bin/sync","/bin/touch","/bin/uname","/bin/vdir","/u
```

Agent Rule	Event name	Deployment Policies	Policy Version
systemd_modification_chmod	chmod	threat-detection compliance	1.64.0-rc13

Expression

```
(
    (chmod.file.path in [ ~"/lib/systemd/system/**", ~"/usr/lib/systemd/system/**",
~"/etc/systemd/system/**", ~"/usr/local/lib/systemd/system/**", ~"/run/systemd/system/**"])
    && process.file.path not in [~"/usr/bin/apt*", "/usr/bin/dpkg", "/usr/bin/rpm", "/usr/bin/unattended-
upgrade", "/usr/bin/npm", ~"/usr/bin/pip*", ~"/usr/local/bin/pip*", "/usr/bin/yum", "/sbin/apk",
"/usr/lib/snapd/snapd"]
) && chmod.file.destination.mode != chmod.file.mode
```

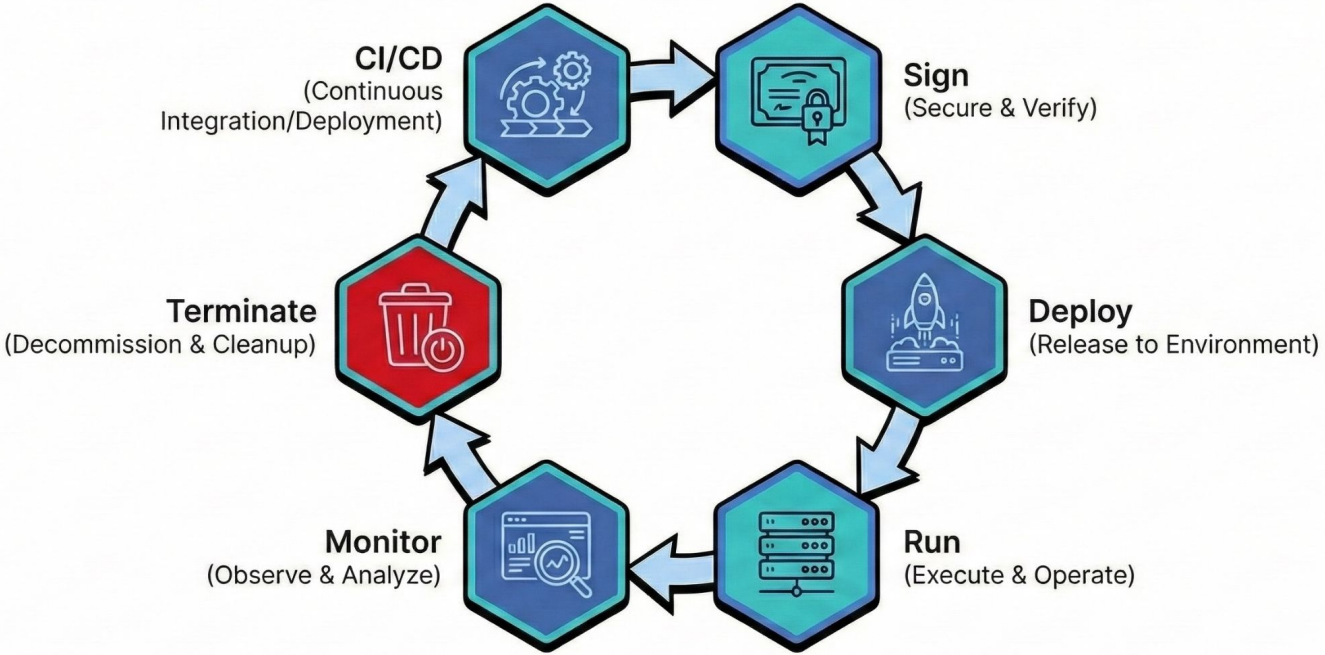
3. Réponse sur Incident : "*Cattle, not Pets*"

- On ne patche pas un conteneur en prod
- On ne "nettoie" pas une dérive
- On termine et on remplace 👍

Rollback immuable

- Isolation : Couper le réseau du pod déviant
- Snapshot : Garder une trace pour l'analyse
- Termination : `kubectl delete pod`
- Fix : Mettre à jour l'image à la source (Dockerfile)

Le cycle de vie complet



Conclusion

- La dérive est inévitable sans mécanismes de contrôle
- Le Shift Left simplifie le runtime.
- Le Runtime Observability (eBPF) est votre source de vérité.
- Ne réparez jamais : remplacez !

Avis :

