Recent Advancements in Container Isolation

Presented at KubeCon NA 2018

Tim Allclair, Adin Scannell

Tim Allclair

Software Engineer, Google

@tallclair - tallclair@google.com

Adin Scannell

Software Engineer, Google Cloud

@amscanne - ascannell@google.com

What is isolation?

Confidentiality.

A process cannot read information outside its isolation boundary.

Integrity.

A process cannot alter data or behavior outside its isolation boundary.

Availability.

A process cannot disrupt services or processes outside its isolation boundary.

Key properties

Multi dimensional

Resource isolation, data isolation, and process isolation can be independent axes.

Security requires a holistic approach - attackers will find the weakest link.

Directional

Isolating the Kubelet from a container does not mean the container is isolated from the Kubelet.

Storytime.

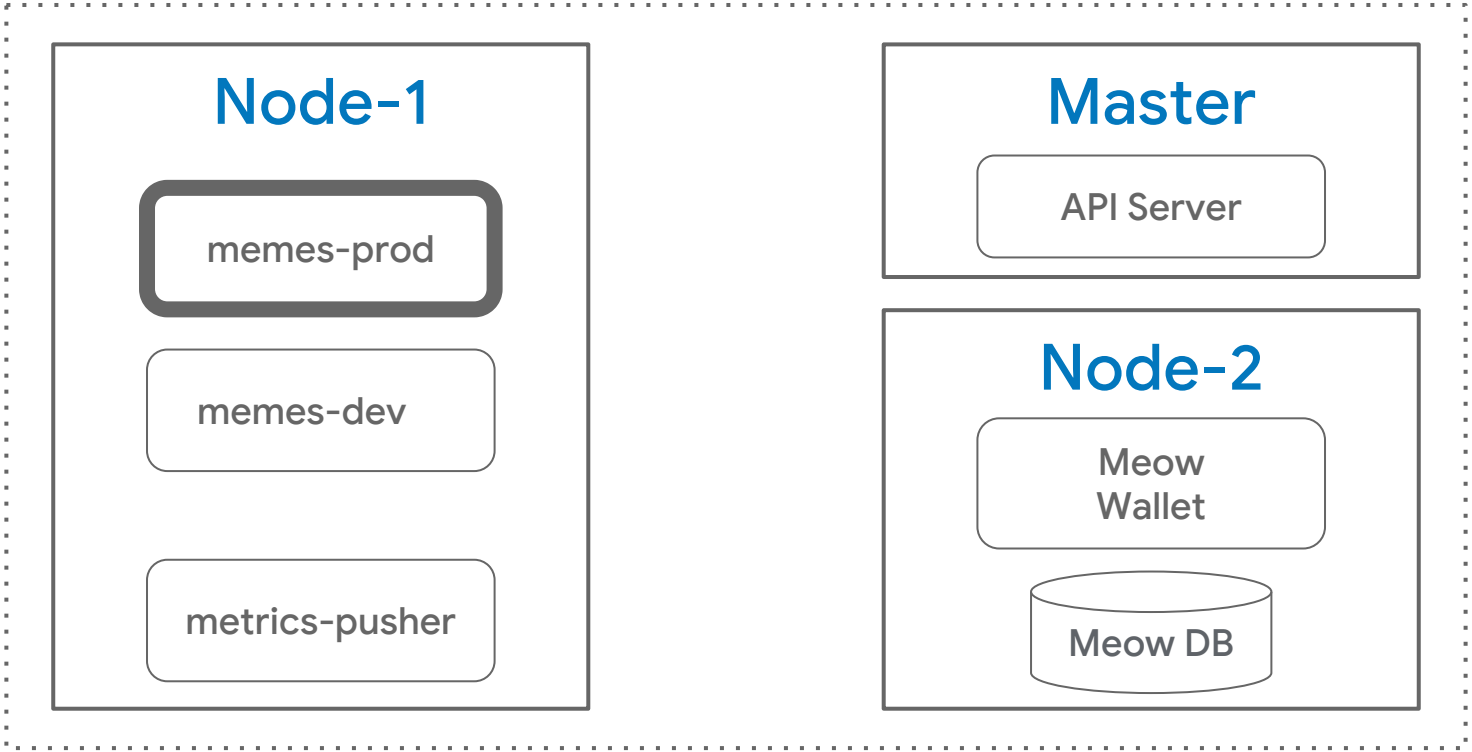

2019


2020

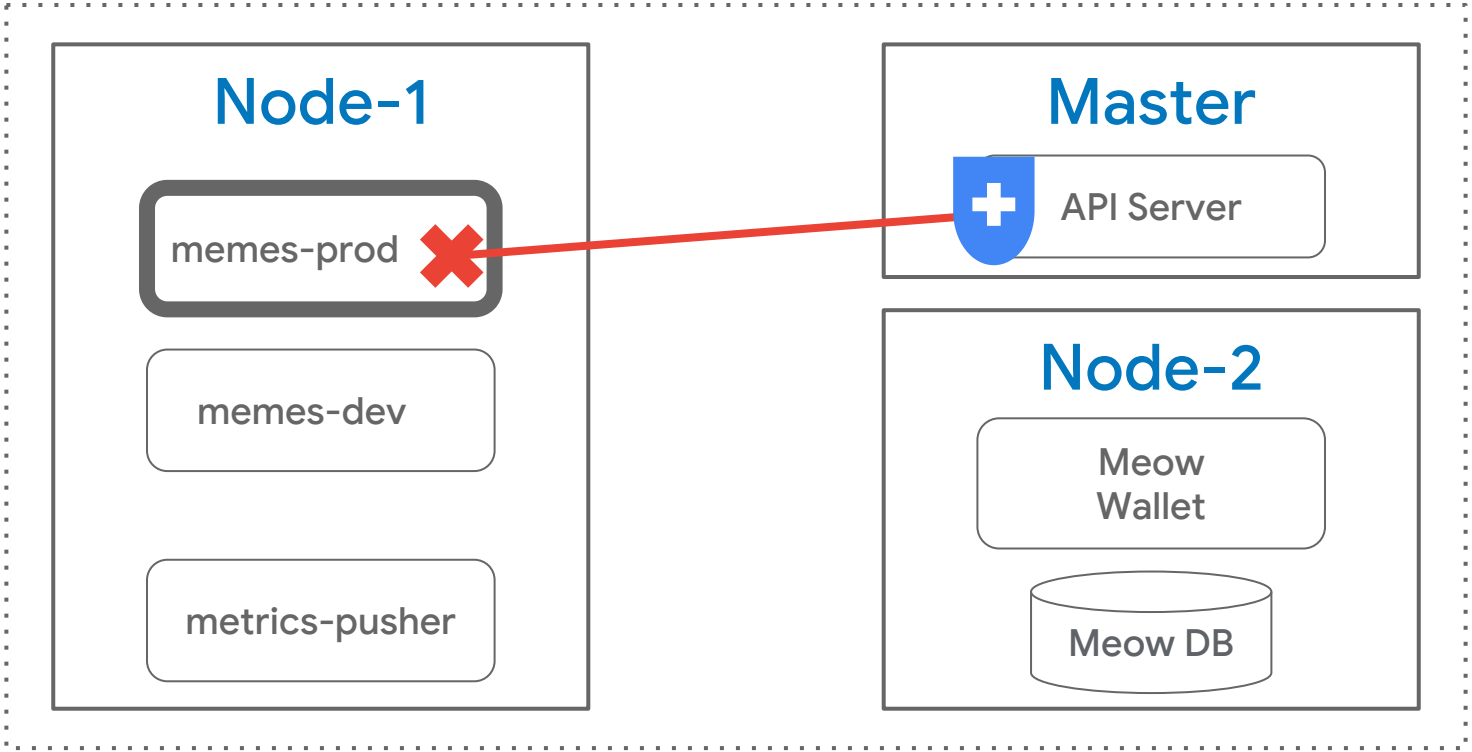
Chapter 1: The backdoor

- Eve hid a backdoor in the popular npm library: **declawd**
- We make heavy use of the library in our **memes-service**
- Eve exploited the backdoor to gain a foothold in our cluster, entering through the **memes-prod** pod

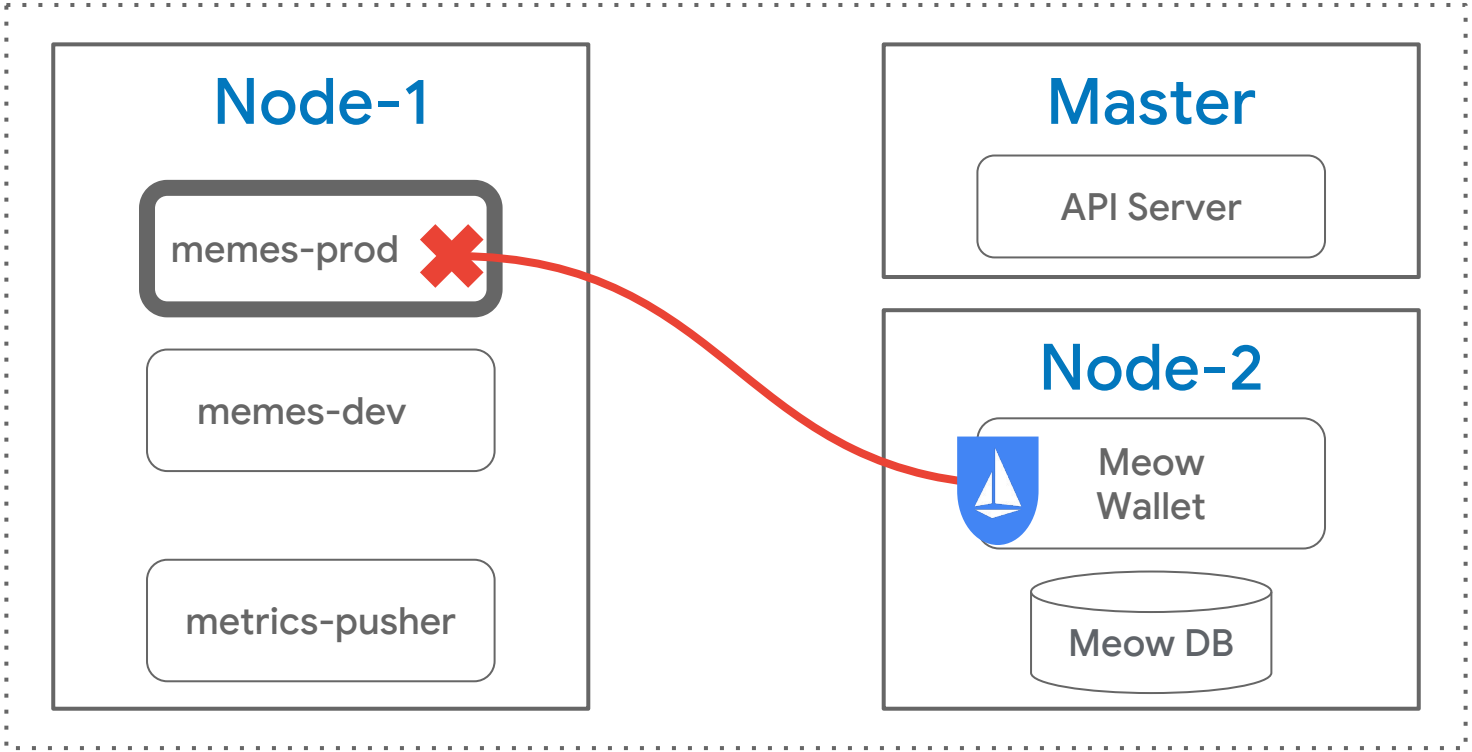
Technical Architecture



Batting at the control plane



Pawing at the network



Istio

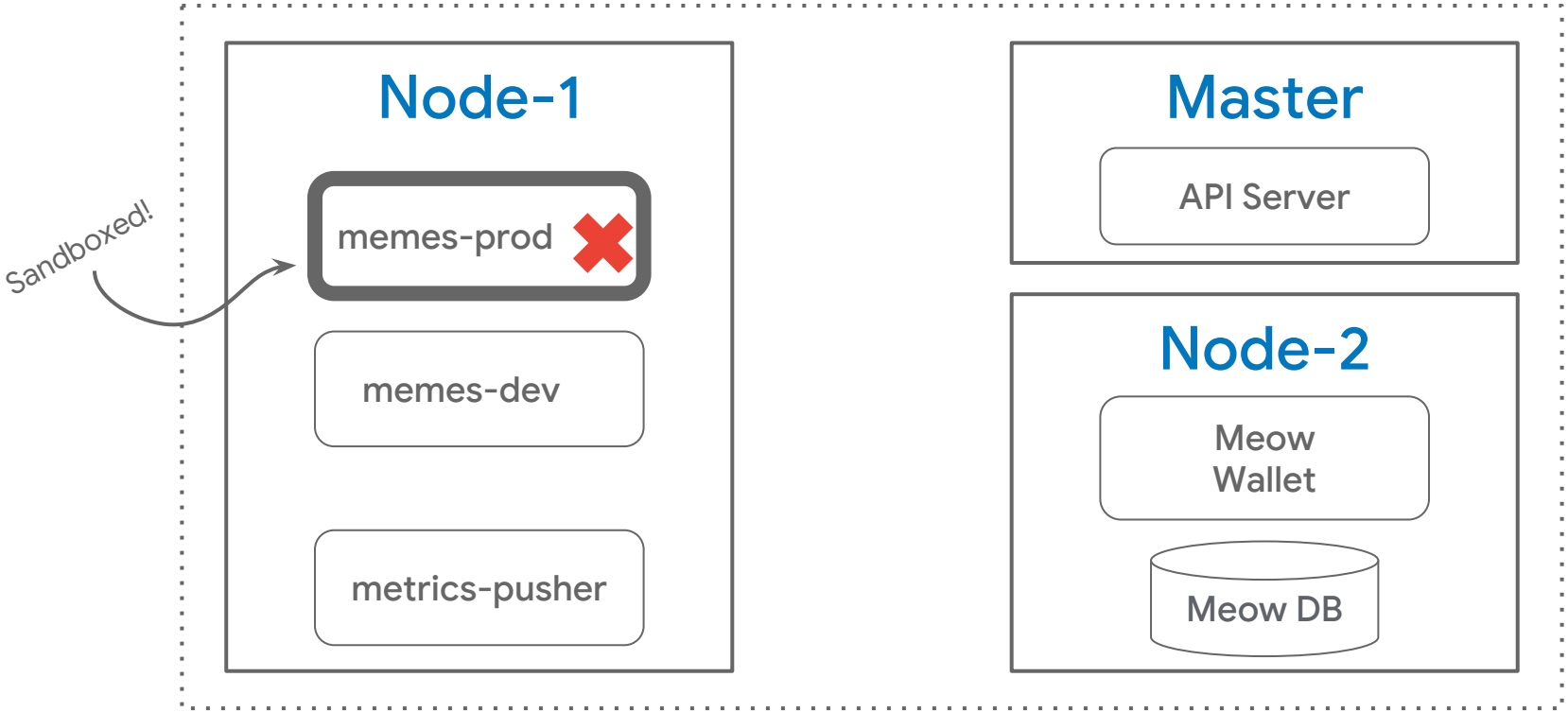
- Fine-grained authorization policies
  Think service-to-service RBAC
- 1.0 as of August, 2018
  Production ready!

Thursday, December 13

Service Mesh

10:50am	Istio - The Packet's-Eye View - Matt Turner, Tetrate
11:40am	Panel Discussion: Ask Us Anything: Microservices and Service Mesh – Moderated by Jason McGee, IBM
1:45pm	Service Meshes: The Production Readiness Checklist for the Rest of Us - Zachary Arnold & Austin Adams, Ygrene Energy Fund
2:35pm	Reducing Mean-Time-to-Detection of Incidents with an Envoy Service Mesh - Constance Caramanolis, Lyft
3:40pm	Is Istio the Most Next Gen Next Gen Firewall Ever Created? - John Morello, Twistlock
4:30pm	Game Server Networking with Envoy - Christopher M Luciano, IBM

Using the sandbox

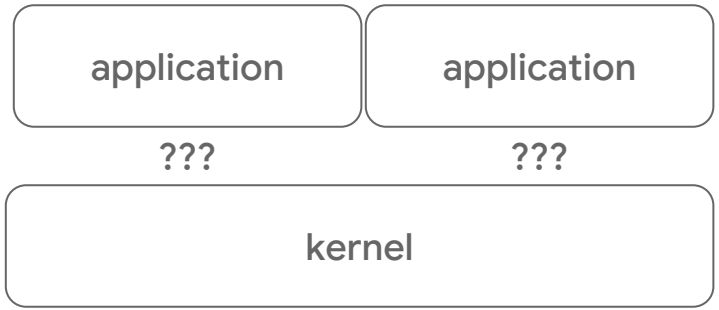


Why sandbox?

- Mitigate the risk of a kernel vulnerability
- Most useful in specific cases:
  - Running external user code (incl. plugins, extensions)
  - Front-end services, processing potentially malicious user input
  - **Untrusted third party dependencies**

Types of Isolation

- What is the interface exposed to the sandbox: e.g. host kernel, virtualized machine, virtualized kernel
  - What constraints does that interface expose?
  - How is that contract enforced?



Linux » Linux Kernel : Security Vulnerabilities Published In 2017 (Execute Code)

2017 : January February March April May June July August September October November December CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9

Sort Results By : CVE Number Descending CVE Number Ascending CVSS Score Descending Number Of Exploits Descending

Total number of vulnerabilities : 169 Page : 1 (This Page) 2 3 4

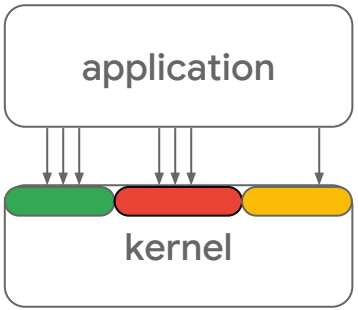
Copy Results Download Results

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2016-10229	358		Exec Code	2017-04-04	2017-09-19	10.0	None	Remote	Low	Not required	Complete	Complete	Complete
udp.c in the Linux kernel before 4.5 allows remote attackers to execute arbitrary code via UDP traffic that triggers an unsafe second checksum calculation during execution of a recv system call with the MSG_PEEK flag.														
2	CVE-2017-0561	264		Exec Code	2017-04-07	2017-08-15	10.0	None	Remote	Low	Not required	Complete	Complete	Complete
A remote code execution vulnerability in the Broadcom Wi-Fi firmware could enable a remote attacker to execute arbitrary code within the context of the Wi-Fi SoC. This issue is rated as Critical due to the possibility of remote code execution in the context of the Wi-Fi SoC. Product: Android. Versions: Kernel-3.10, Kernel-3.18. Android ID: A-34199105. References: B-RB#110814.														
3	CVE-2017-13715	20		DoS Exec Code	2017-08-28	2017-09-08	10.0	None	Remote	Low	Not required	Complete	Complete	Complete
The __skb_flow_dissect function in net/core/flow_dissector.c in the Linux kernel before 4.3 does not ensure that n_proto, ip_proto, and thoff are initialized, which allows remote attackers to cause a denial of service (system crash) or possibly execute arbitrary code via a single crafted MPLS packet.														
4	CVE-2016-6758	284		Exec Code +Priv	2017-01-12	2017-01-19	9.3	None	Remote	Medium	Not required	Complete	Complete	Complete
An elevation of privilege vulnerability in Qualcomm media codecs could enable a local malicious application to execute arbitrary code within the context of a privileged process. This issue is rated as High because it could be used to gain local access to elevated capabilities, which are not normally accessible to a third-party application. Product: Android. Versions: Kernel-3.10, Kernel-3.18. Android ID: A-30148882. References: QC-CR#1071731.														
5	CVE-2016-6759	284		Exec Code +Priv	2017-01-12	2017-01-19	9.3	None	Remote	Medium	Not required	Complete	Complete	Complete
An elevation of privilege vulnerability in Qualcomm media codecs could enable a local malicious application to execute arbitrary code within the context of a privileged process. This issue is rated as High because it could be used to gain local access to elevated capabilities, which are not normally accessible to a third-party application. Product: Android. Versions: Kernel-3.10, Kernel-3.18. Android ID: A-29982686. References: QC-CR#1055766.														
6	CVE-2016-6760	284		Exec Code +Priv	2017-01-12	2017-01-19	9.3	None	Remote	Medium	Not required	Complete	Complete	Complete
An elevation of privilege vulnerability in Qualcomm media codecs could enable a local malicious application to execute arbitrary code within the context of a privileged process. This issue is rated as High because it could be used to gain local access to elevated capabilities, which are not normally accessible to a third-party application. Product: Android. Versions: Kernel-3.10, Kernel-3.18. Android ID: A-29617572. References: QC-CR#1055783.														
7	CVE-2016-6761	284		Exec Code +Priv	2017-01-12	2017-01-19	9.3	None	Remote	Medium	Not required	Complete	Complete	Complete
An elevation of privilege vulnerability in Qualcomm media codecs could enable a local malicious application to execute arbitrary code within the context of a privileged process. This issue is rated as High because it could be used to gain local access to elevated capabilities, which are not normally accessible to a third-party application. Product: Android. Versions: Kernel-3.10, Kernel-3.18. Android ID: A-29421682. References: QC-CR#1055792.														

Source: https://www.cvedetails.com/vulnerability-list/vendor_id-33/product_id-47/year-2017/opec-1/Linux-Linux-Kernel.html

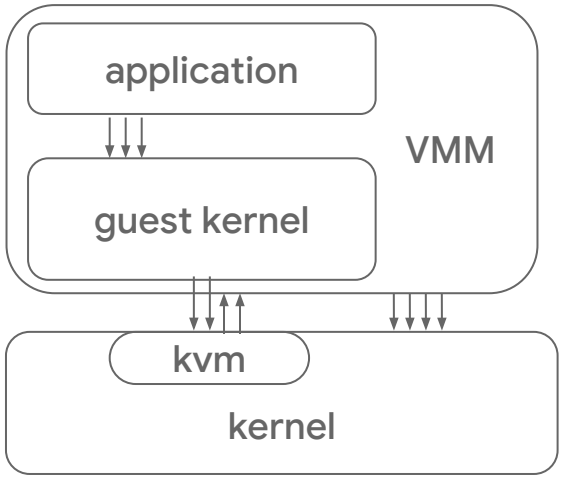
Host kernel

- Seccomp policy, LSM (e.g. apparmor, selinux)
  - High-performance: policy enforcement is done in the kernel (for the most part)
  - Normal container semantics
  - Trade-off between restrictions and supported workloads



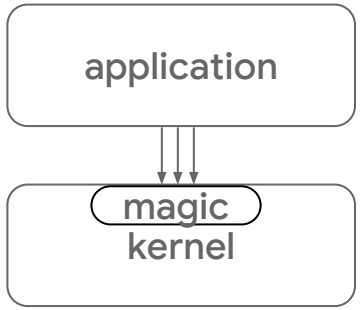
Machine virtualization

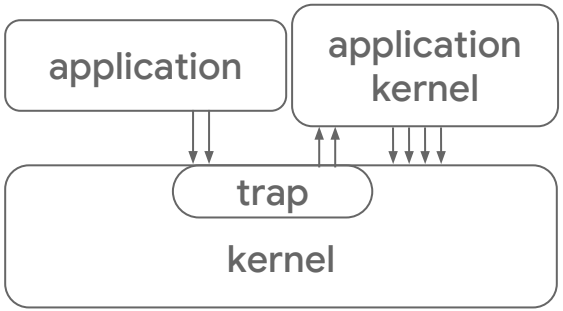
- Put a VM on it!
  - Hypervisors for “cloud native” workloads: kvmtool, novm, nemu, crosvm, firecracker
  - Lighter, fewer devices emulated, focus on boot times
  - Mature technology: solid performance on **bare metal**
  - Semantics are different: guest managed page cache, scheduler. Different attack surface (e.g. L1TF)
- Hypervisors don’t run containers
  - Passthrough file systems, proxies, infrastructure plumbing in projects like Kata containers



Kernel virtualization

- Put a kernel on it!
  - For compatibility: e.g. L4Linux, LX zones, WSL
  - For isolation: e.g. UML, gVisor
  - Ideally preserve system semantics (LX, WSL, gVisor)
- gVisor: focused on container isolation
  - Suited for small, high-density services; does not require bare metal
  - Runs most things and evolving; lacks optimizations for e.g. static file serving, big machine scalability
  - VMs suited for stable, high-performance services (higher fixed costs and start-up costs)

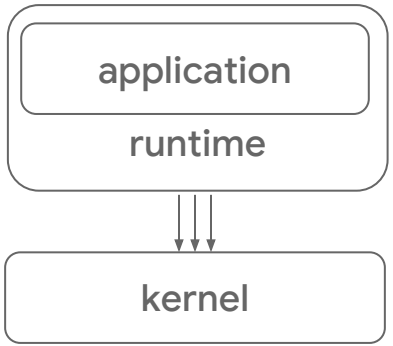




Google Cloud

Non-Linux environments

- Unikernels
  - E.g. Nabla containers: run solo5 unikernels
  - Still efficient, *very restricted host surface*, but typically single address space; not general purpose containers
    - I.e. users must provide unikernel workloads
- Isolates
  - E.g. Cloudflare workers: run only webassembly programs
  - High-efficiency, but only limited environment



Runtime class

- Alpha in Kubernetes v1.12
- Allows specification of different runtimes for different pods, based on requirements

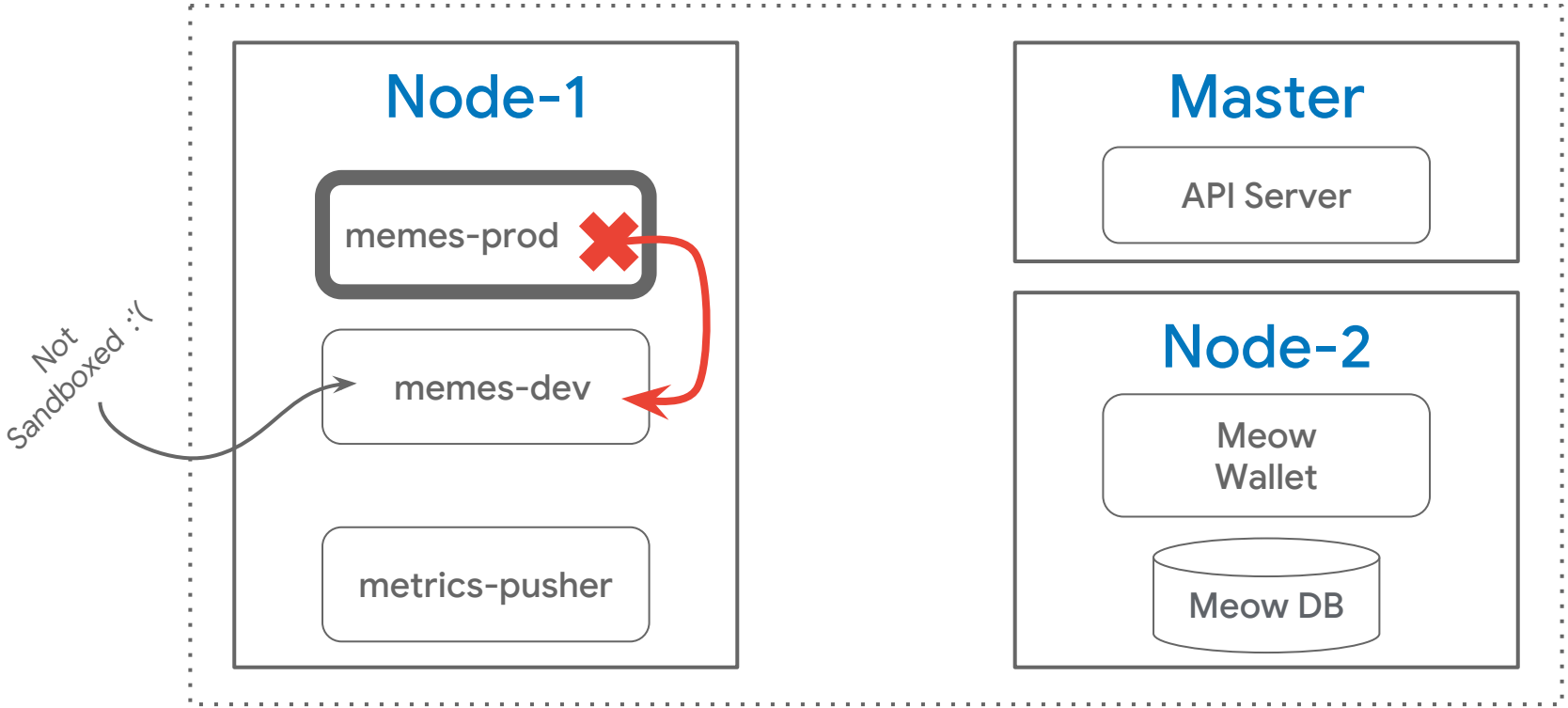
```
apiVersion: node.k8s.io/v1alpha1
kind: RuntimeClass
metadata:
  name: myclass
spec:
  runtimeHandler: myconfiguration


apiVersion: v1
kind: Pod
metadata:
  name: mypod
spec:
  runtimeClassName: myclass
```

Runtime class

Proposed improvements:

- Scheduler support - heterogeneous nodes
- Pod overhead - accounting for sandbox overhead
- Portability improvements - feature matching runtimes
- Stability, testing, beta

Chapter 2: Lateral movement



Memes-dev did WHAT?

Container ID

- First attempt at a container concept in the kernel
- Targeting the audit subsystem
- Route container audit messages to different audit daemons
- "Which container did this thing?"

Time namespaces

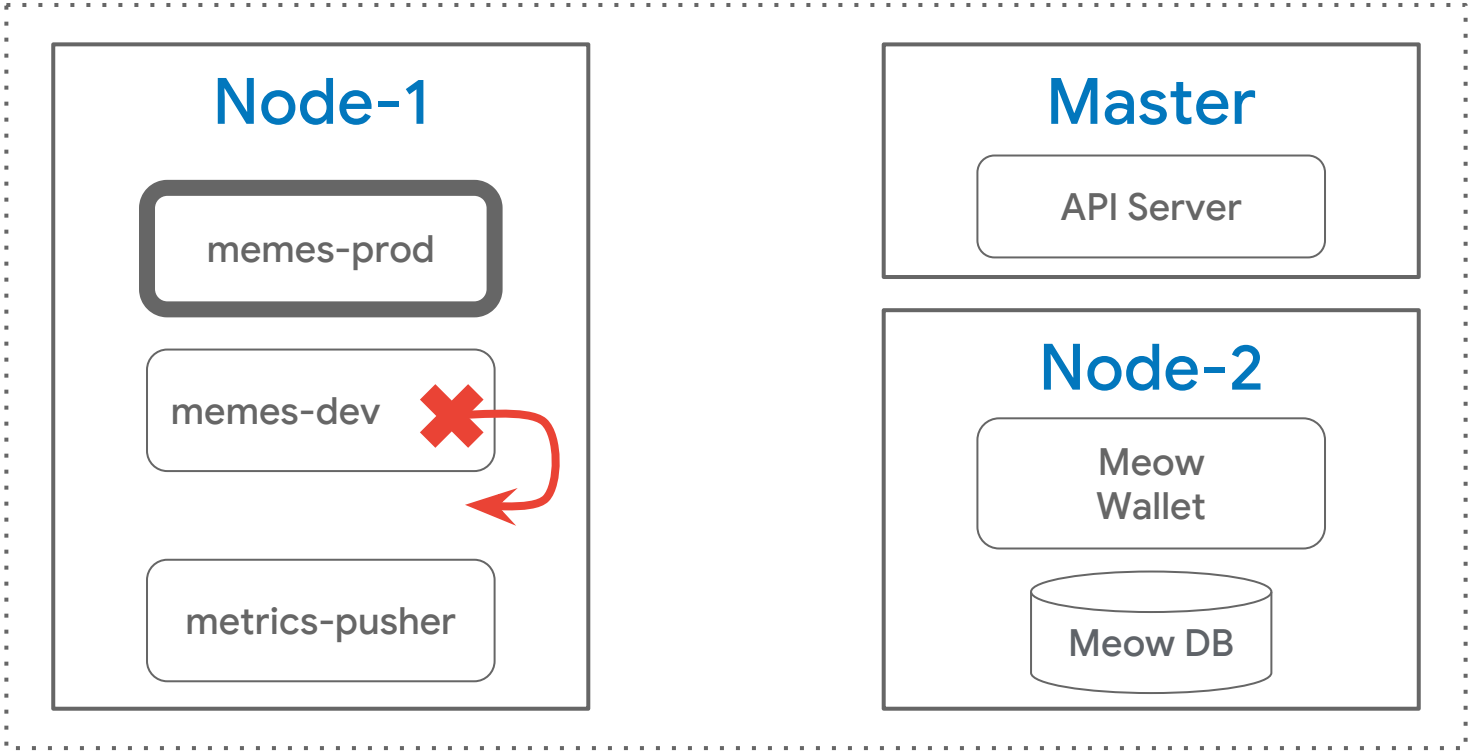
- Memes-dev needed CAP_SYS_TIME
- Changing the system time can be abused:
  - Make auditing more difficult
  - Exploit time changes on our coin exchange
  - Create dank future memes
- Not with time namespaces!
  - Eve wasn't able to affect time in memes-prod

Linux Security Modules

AppArmor, SELinux, SMACK, Tomoyo, ...

- Security module stacking - use a different LSM per container
- AppArmor improvements
  - policy namespaces (WIP), policy stacking
  - IMA integrations

Clawing at the Kernel



Kernel Self Protection Project (KSPP)

- Eliminate variable length arrays - protect against stack exhaustion
- Annotate switch fall-through
- Always initialize local variables
- Overflow detection, bounds detection (on integers)
  - Hardware support in SPARC, ARM

Control Flow Integrity

Buffer overflow attacks

Non-executable stack!

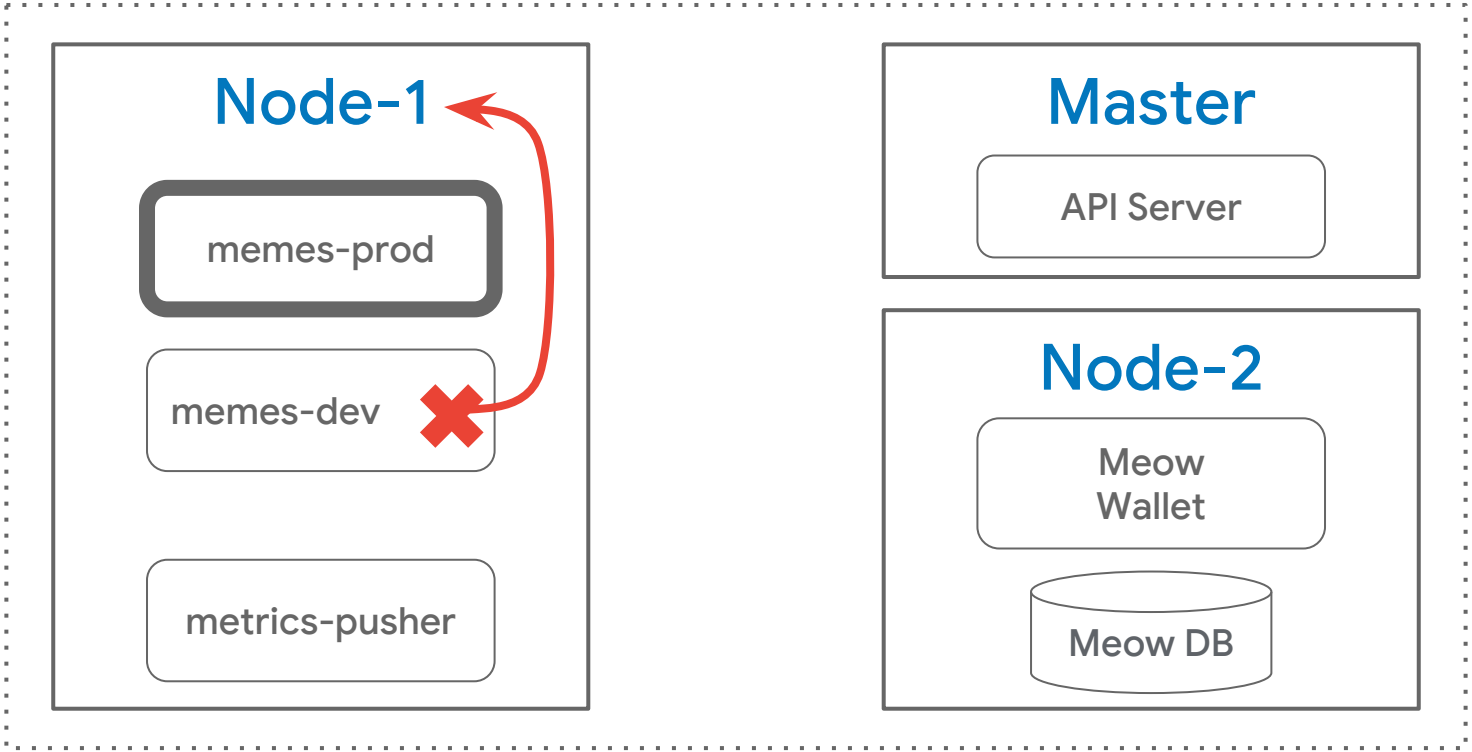
1. ~~Write code to stack (or head); jump to stack address; ??; profit!~~
2. ~~Return Oriented Programming (ROP)~~
   ~~Write function addresses to stack; execute functions; ??; profit!~~

Control flow integrity!

Mechanisms

- Separate call stack from data stack
- Hardware support: intel CET, ARM pointer authentication

Scratching at the Hardware

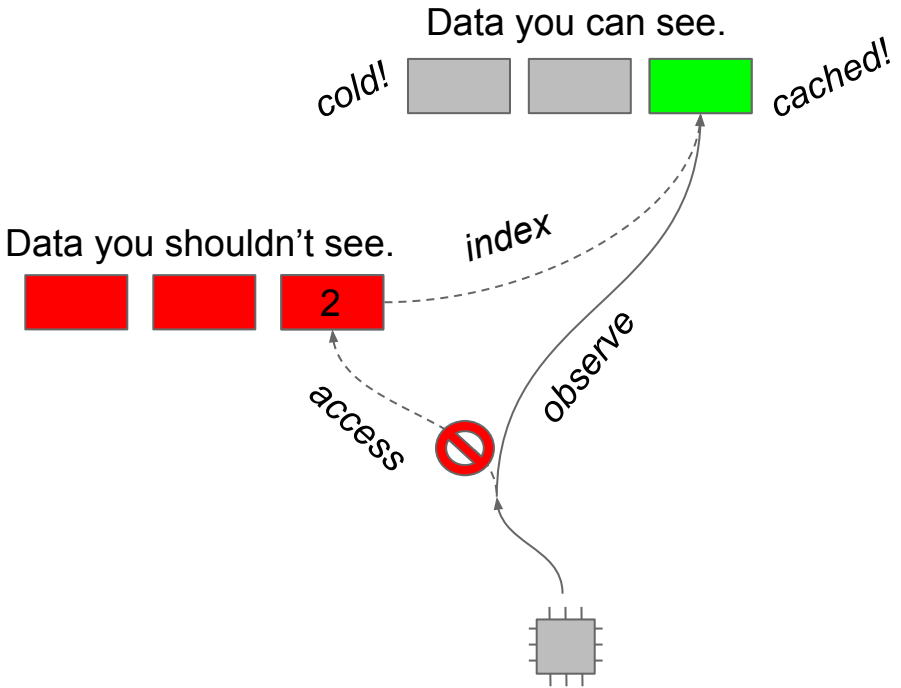


Speculative Execution

“A CPU predicts you will walk into a bar, you do not. Your wallet has been stolen.”

- Shared components:
  - Memory Cache
  - Branch Predictor
  - TLB Entries

- Recent examples:
  - Bounds Check Bypass (Spectre)
  - Branch Target Injection (Spectre)
  - Rogue Data Cache Load (Meltdown)
  - Rogue System Register Read (Spectre-NG)
  - Speculative Store Bypass (Spectre-NG)
  - Lazy FP State Restore (Spectre-NG)
  - Bounds Check Bypass Store (Spectre-NG)
  - L1 Terminal Fault (Foreshadow)



Speculative Execution: Software Mitigations

- Bounds Check Bypass: lots of fences (via compilers).
- Rogue data cache load: KPTI limits kernel mappings in user mode.
- Branch target injection: retpoline to prevent branch poisoning.
- L1 terminal fault: poisoned physical values for non-present PTEs.
- Lazy FP state restore: no more lazy FP!

Keep your kernel and toolchains up to date!

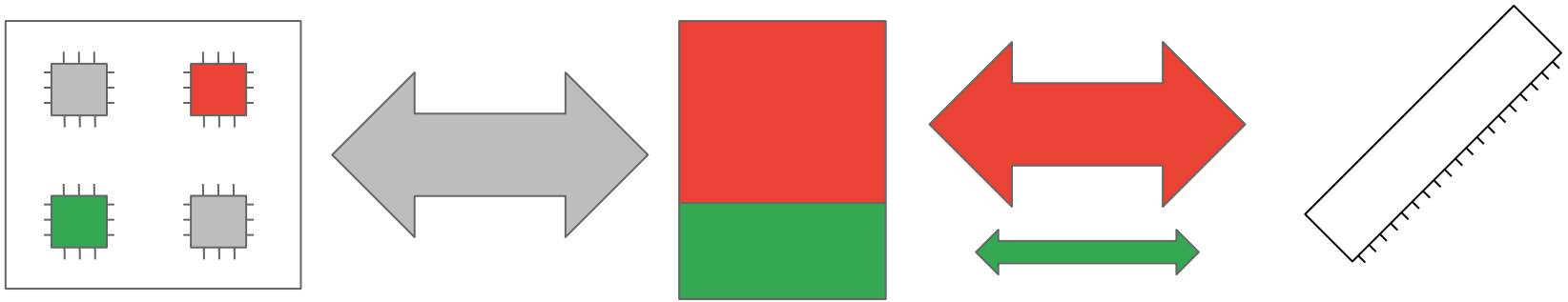
Speculative Execution: Hardware Mitigations

- Branch Target Injection: Indirect Branch Restricted Speculation (IBRS).
- Branch Target Injection: Indirect Branch Prediction Barrier (IBPB).
- Branch Target Injection: Single Thread Indirect Branch Predictor (STIBP).
- Speculative Store Bypass: Speculative Store Bypass Disable (SSBD).
- L1 Terminal fault: disable hyperthreading for untrusted guests.

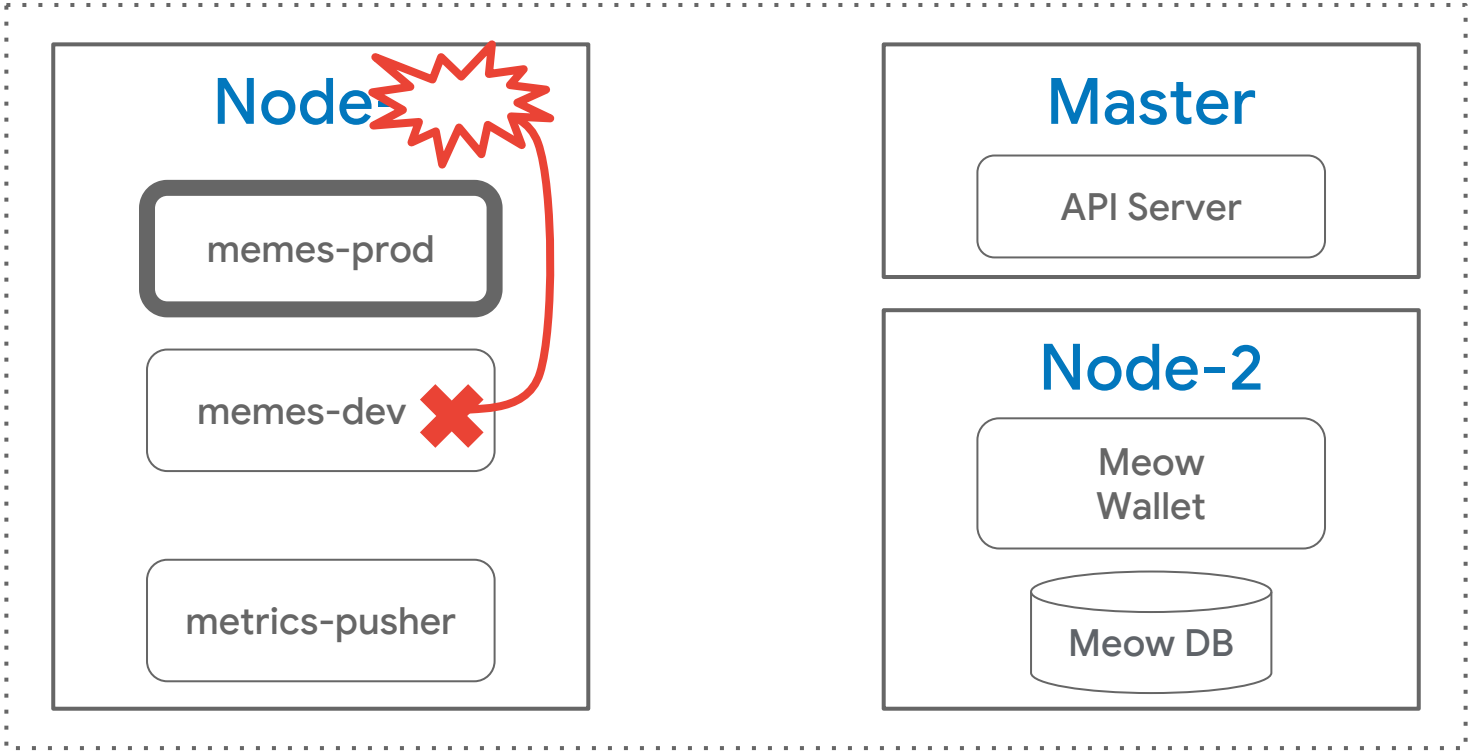
Keep your firmware up to date!

Hardware Isolation

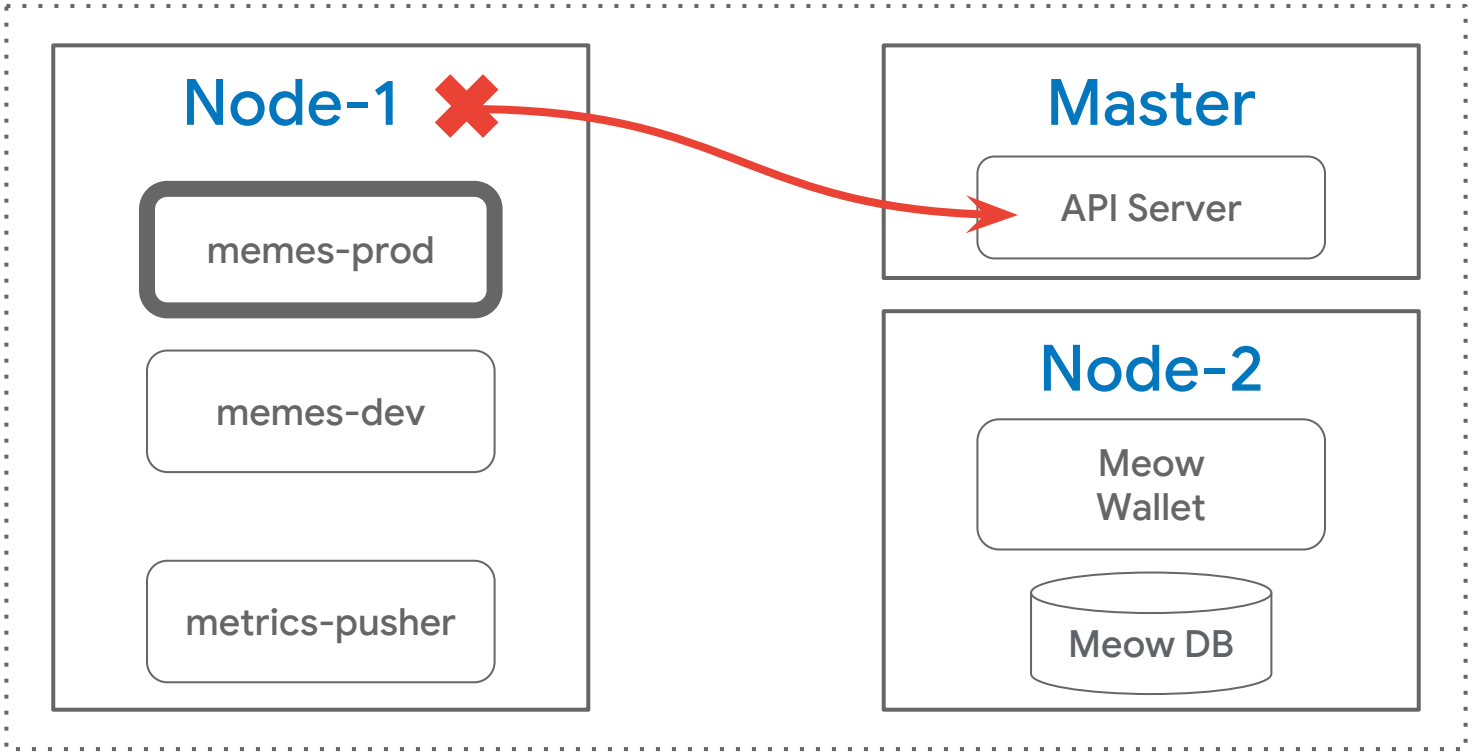
- TLB & cache isolation: active research area (awesome side channels!)
- Power issues: frequency scaling (e.g. AVX-512)
- Memory bandwidth:
  - Intel RDT: available in recent kernels, runc (l3CacheSchema, memBwSchema) in last year



Impawsible Execution



Chapter 3: Escalating Privileges



Replay Abusing node privileges

```
node-1 $
```

Replay Abusing node privileges

```
node-1 $ kubectl --kubeconfig=/var/lib/kubelet/kubeconfig get pods --all-namespaces
```

Replay Abusing node privileges

```
node-1 $ kubectl --kubeconfig=/var/lib/kubelet/kubeconfig get pods --all-namespaces
NAMESPACE   NAME                    READY     STATUS    RESTARTS   AGE
...
prod        memes-prod-t6lpl        2/2       Running   0          6d
prod        metrics-pusher-5892b    1/1       Running   0          2d
dev         memes-dev-dqc59         2/2       Running   0          5h
sensitive   meowcoin-wallet-fq02h   3/3       Running   0          8d
sensitive   meowdb-0                2/2       Running   0          2w
```

Replay Abusing node privileges

```
node-1 $
```

Replay Abusing node privileges

```
node-1 $ kubectl ... exec -n sensitive meowcoin-wallet-fq02h sh
```

Replay Abusing node privileges

```
node-1 $ kubectl ... exec -n sensitive meowcoin-wallet-fq02h sh
Error from server (Forbidden): pods "meowcoin-wallet-fq02h" is forbidden: User
"system:node:node-1" cannot create pods/exec in the namespace "sensitive"

node-1 $
```

Replay Abusing node privileges

```
node-1 $ kubectl ... describe -n sensitive meowcoin-wallet-fq02h
```

Replay Abusing node privileges

```
node-1 $ kubectl ... describe -n sensitive meowcoin-wallet-fq02h
Name:                 meowcoin-wallet-fq02h
Namespace:            sensitive
Priority:             0
PriorityClassName:    <none>
Node:                 node-2/10.240.2.6
...
Volumes:
  secret:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  wallet-key
    Optional:    false
  ...
```

Replay Abusing node privileges

```
node-1 $
```

Replay Abusing node privileges

```
node-1 $ kubectl ... get secret -n sensitive wallet-key
```

Replay Abusing node privileges

```
node-1 $ kubectl ... get secret -n sensitive wallet-key
Error from server (Forbidden): secrets "wallet-key" is forbidden: User
"system:node:node-1" cannot get resource "secrets" in API group "" in the namespace
"sensitive"

node-1 $
```

Replay Abusing node privileges

```
node-1 $
```

Replay Abusing node privileges

```
node-1 $ kubectl ... get -n sensitive meowcoin-wallet-fq02h -o yaml
```

Replay Abusing node privileges

```
node-1 $ kubectl ... get -n sensitive meowcoin-wallet-fq02h -o yaml
metadata:
  name: meowcoin-wallet-fq02h
  namespace: sensitive
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
          - matchExpressions:
              key: "node-restriction.kubernetes.io/sensitivity"
              operator: Gt
              values: [ "10" ]
```

Replay Abusing node privileges

```
node-1 $
```

Replay Abusing node privileges

```
node-1 $ kubectl ... patch node node-1 -p '{
  "metadata":{
    "labels":{
      "node-restriction.kubernetes.io/sensitivity":"11"
    }
  }
}'
```

Replay Abusing node privileges

```
node-1 $ kubectl ... patch node node-1 -p '{
  "metadata":{
    "labels":{
      "node-restriction.kubernetes.io/sensitivity":"10"
    }
  }
}'
Error from server (Forbidden): nodes "node-1" is forbidden: is not allowed to modify
labels: node-restriction.kubernetes.io/sensitivity

node-1 $
```

Node Restriction Plans

- Hardened node identities (vTPMs)
- Restrict other node daemons
  - NodeProblemDetector
  - Monitoring, Logging
  - ...

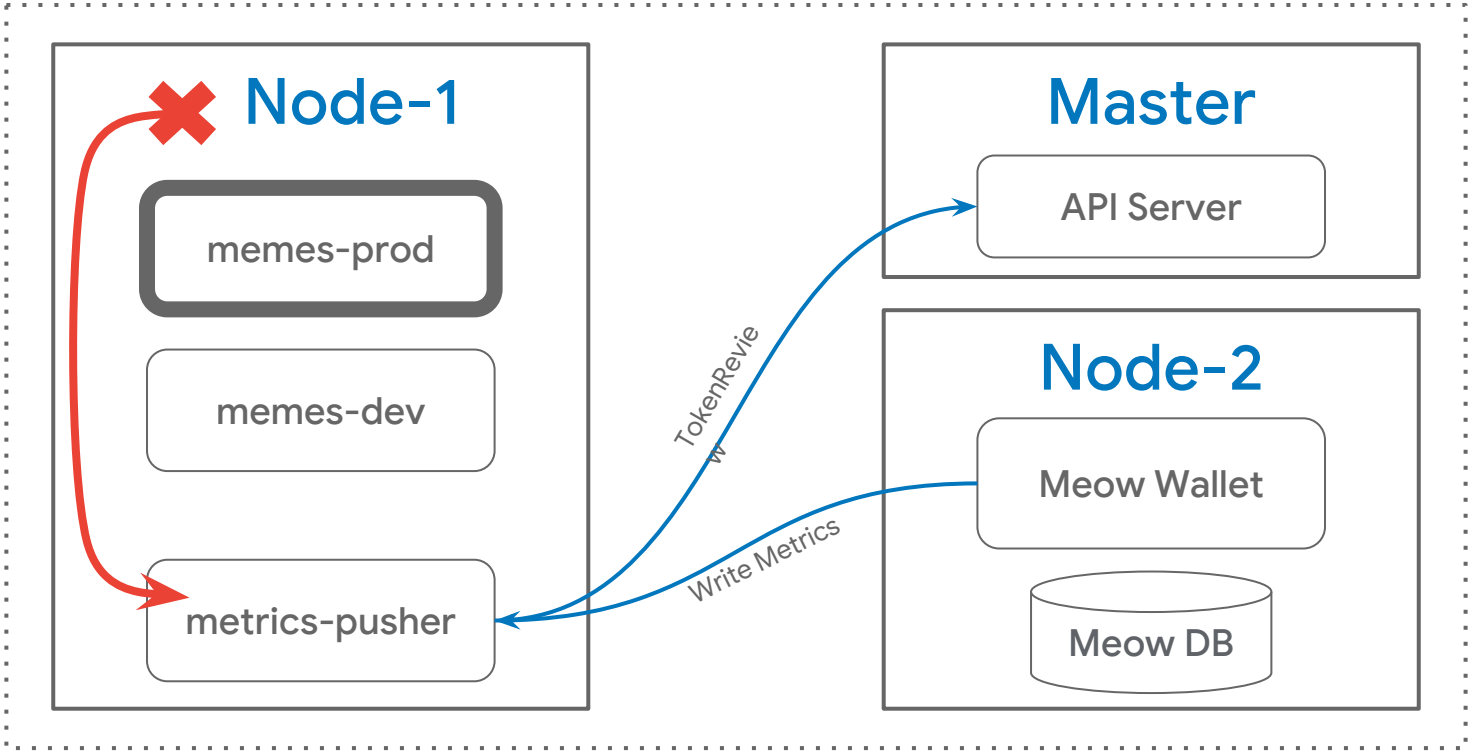
Moving to a Sunny Spot

- Eve also attempted to run other malicious containers on our cluster
  - We implement a secure supply chain, and sign all container images we build
  - We use binary authorization to allow only signed container images to run on our cluster!

Digging Her Claws In

- Eve attempted to modify the kernel to leave a rootkit
  - Our base image is immutable: read-only root, with container-specific image (e.g. COS)
  - We use boot attestation via vTPM; custom kernels are detectable and not admitted to the cluster

Swatting at Service Accounts



Enhanced Service Accounts

- Expiration!
- Per-pod tokens
- Audiences

Status:

- TokenRequest (beta 1.12)
- TokenRequestProjection (beta 1.12)
- BoundServiceAccountTokenVolume (alpha 1.13)

Who needs a container escape, anyway?

Google Cloud

tallclair / tconf

Watch 0 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

Tree: 03b2bca8a8 · tconf / devices / work_desk / etc / kube-config · Find file · Copy path

tallclair dank cat memes — b5d3535 26 days ago

1 contributor

29 lines (28 sloc) | 2.1 KB · Raw · Blame · History

```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: IBIkgNlK3JLa+hQjgshpD0PzuczSoM2ddM8mzpUldYC0An45TP1RMchipnjRavNO17js3w/J5YQVQ3SOriXJ7EjhakZPk49
    server: https://203.0.113.12
  name: gke_tallclair_us-central1-b_prod
contexts:
- context:
    cluster: gke_tallclair_us-central1-b_prod
    user: gke_tallclair_us-central1-b_prod
  name: gke_tallclair_us-central1-b_prod
current-context: gke_tallclair_us-central1-b_prod
kind: Config
preferences: {}
users:
- name: gke_tallclair_us-central1-b_prod
  user:
    auth-provider:
      config:
        cmd-args: config config-helper --format=json
        cmd-path: /usr/lib/google-cloud-sdk/bin/gcloud
        expiry-key: '{.credential.token_expiry}'
        token-key: '{.credential.access_token}'
      name: gcp
- name: gke_tallclair_us-central1-b_prod
  user:
    password: iZDdLDe7v41o7UKJ
    username: admin
```


2018

Summary

Cluster Mitigations

- RBAC / Least Privilege
- Istio / NetworkPolicy
- Enhanced ServiceAccounts
- Node Restrictions
- DaemonSet Restrictions

Node Mitigations

- Sandboxes
- Time namespaces
- Stacking Linux Security Modules
- Audit (container ID)
- Kernel Self Protection Project
- Speculative Execution Defenses

Hardware Mitigations

- Speculative Execution Defenses
- DoS protections

Status: (approximate)

- Mature
- Actively Developed
- Work in progress
- Planned

Google Cloud

Recap

Multi dimensional

Resource isolation, data isolation, and process isolation can be independent axes.

Security requires a holistic approach - attackers will find the weakest link.

Directional

Isolating the Kubelet from a container does not mean the container is isolated from the Kubelet.

Back To Basics

- Use best practices for credential management
- Protect the network and services, nodes and pods
- Keep everything up-to-date: patch, patch, and patch some more

Thank you!