SADDNS reimplementation

Shenghan Zheng

June 2022

1 Question from last week

Do we need to know the port number of recursive resolvers?
The answer is no. We need to have UDP 53 allowed for responses to DNS queries that your server sends, as UDP is a stateless protocol.

1.1 another question

Based on the aims of 'destination port randomization', is there any other ways make off-path attackers guess destination harder?

2 Background Info

2.1 ICMP message type

1.Fragment Needed
2.Redirect
3.Host/Port Unreachable

2.2 UDP

User datagram protocol
1. Provide a datagram abstraction
A message sent in single layer 3 packet
Max size limited by max size of packets
Data is divided to datagrams by application. It's sent and received by single unit
2. No reliability or ordering guarantee. But adds ports
3. Much faster than TCP since there is no 3-way handshake
UDP data structure
source port(16 bits)
destination port(16 bits)

length(16 bits)
checksum(16 bits)
data(variable length)

2.3 Defense

Usually port number and TxID are unknown to off-path attacker. With small attack window(<1s), It's impossible to bruteforce 2^{32} possibilities

2.4 Probe

A probe executes something, usually against a set of targets, to verify that the systems are working as expected from consumers' point of view. UDP probe sends a UDP packet to the configured targets. UDP probe (and all other probes that use ports) provides more coverage for the network elements on the data path as most packet forwarding elements use 5-tuple hashing and using a new source port for each probe ensures that we hit different network element each time.
Actually, probes May be very simple. In naive version, simply check the time gap between request and response will reveal some of the information.

2.5 Exploit of SAD DNS

Infer the port number before bruteforce TxID by checking the ICMP reply. To infer the port opened by recursive resolver, the UDP probe must appear to be sent from the name server. attacker can take advantage on the side-channel raised from ICMP global rate limit.
ICMP limit can be realized by using token bucket. For example, there are 50 token in a bucket and can be filled in 50ms. After 50ms, It can be refill again. Each solicited ICMP message will remove one token from the bucket to be sent on the link, and an empty bucket prevents any ICMP reply.
When attacker try with UDP probes, attacker can get ICMP(the port isn't open), normal response with error messages(the port is open but it's not intended), and normal response(intended port)
Since the bucket is shared among all IPs, including both the attacker's spoofed IP and their real IP, they can then infer if their spoofed UDP probe solicited an ICMP by checking whether their verification probes (UDP probes sent from the real host) can get an ICMP reply or not.
The number of remaining tokens in bucket is the number of open ports we hit in a round.

2.6 Threat Model

off-path attacker with IP spoofing

2.7 Steps

1. Start from triggering either forwarder or resolver to send a DNS query
2. Inferring source port by side-channel(port scan)
3. Extending attack window

2.8 Method to bypass traditional defend

1. attack window
In forwarder, attacker set his own domain name server to be unresponsive and query his domain
In resolver, attacker takes advantage of RRL by spoofed DNS queries.
2. ICMP rate limit
multiple IP addresses with only 1 ipv6 address
multiple IP addresses with only 1 ipv4 address with DHCP
IP spoofing

2.9 2 circumstances

1. Attack forwarder
a wifi router, a DNS server, a DHCP server
ICMP rate limit: use DHCP to bypass per-IP limit
Attack window: use a malicious name server to extend window
Phase I:
Acquire 240 IP addresses using DHCP
Phase II:
Issue a query to forwarder for a subdomain. If receive service fail or wait longer than 1 minutes, repeat a query. If get NOERROR, the response is injected.
2. Attack resolver
an attack machine in adjacent network, an authoritative server
Issue a query for a domain name and launch port scan. Mute authoritative server with loss rate 80

2.10 Mitigation

"Our solution is to add some noise, so that the attackers no longer can get help from the predictable token bucket limiter."
1. Destroy the side channel
Disable outgoing ICMP
Randomize ICMP global rate limit

3 Implementation

3.1 What we have

attackforwr (Forwarder attacking tool)
attackrecur (Resolver attacking tool)
attackrecurns (Resolver attacking tool, NS record polluting version)
nsmuter (Attack tool for flooding the victim NS)
udpscan (Private UDP port scanner)
delaynsThe program that sets up attacker-controlled NS.

3.2 What we don't have

a client(I can use my computer as client)
a forwarder()
a recursive resolver()
Use VM and build forwarder and recursive resolver through BIND

3.3 files

named.conf - contains main configuration file for the DNS file.

named.conf.local - contains the local DNS server configuration, and this is where you declare the zones associated with the domain.

3.4 important ip address

forwarder 8.8.8.8(google's name server) 8.8.4.4(its backup server)
listen-on any ubuntu.local 10.0.2.15 client ip

3.5 Command

ifconfig -a (find Ethernet name) ip -r (check Ubuntu server IP address) sudo tcpdump -vv -n -i -enp0s3 port 53 (listen on traffic) DNS name server 127.0.0.1

4 Terms

4.1 Socket

Socket is used on transport layer. DNS is an application protocol.
A network socket is a software structure within a network node of a computer network that serves as an endpoint for sending and receiving data across the network.

4.2 host

a computer or other device that communicates with other hosts on a network. Hosts on a network include clients and servers – that send or receive data, services or applications.
For individuals, a host is a web server that stores and transmits the data for one or more websites