Milestone Presentation: DNS Rebinding attack for IoT devices

Jiangnan Yan, Sean Mawhinney, Phu-Khang Bui

Motivation

DNS Rebinding...

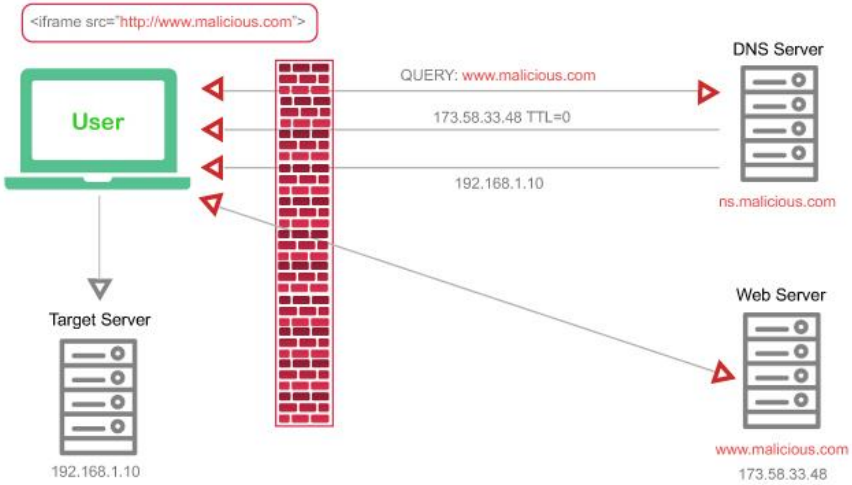
- ...can get past NAT (Network Address Translation) and Firewalls
- ...can allow attackers to remotely manipulate devices on a private network
- ...is particularly effective for attacking IoT (Internet of Things) devices

Relevant Studies:

- Tatang, Suurland, Holz (2019)
- Kalof, Shankar, Tygar, Wagner (2007)
- Kokkinopoulos (2009)
- ...and many more

Problem

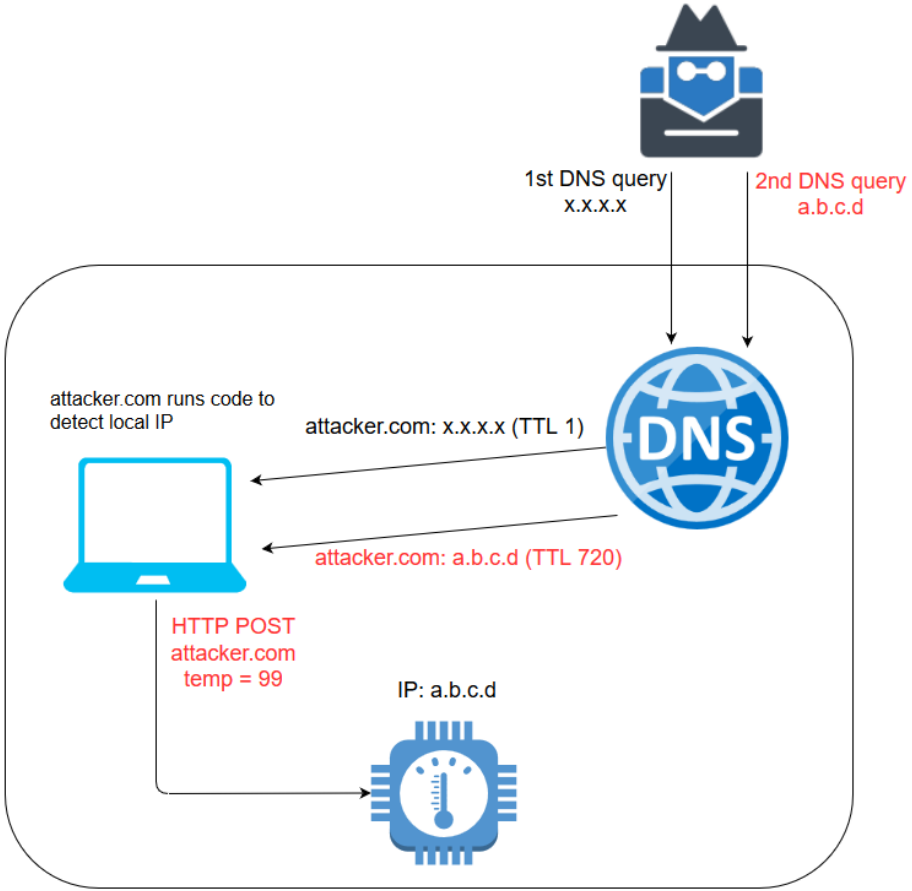
- Need to execute client-side code on victim that makes a request to malicious webserver
- Need to find victim's local IP address
- Need to update DNS records for malicious domain with local IP address



https://www.cyberpunk.rs/dns-rebinding-behind-the-enemy-lines

Challenges

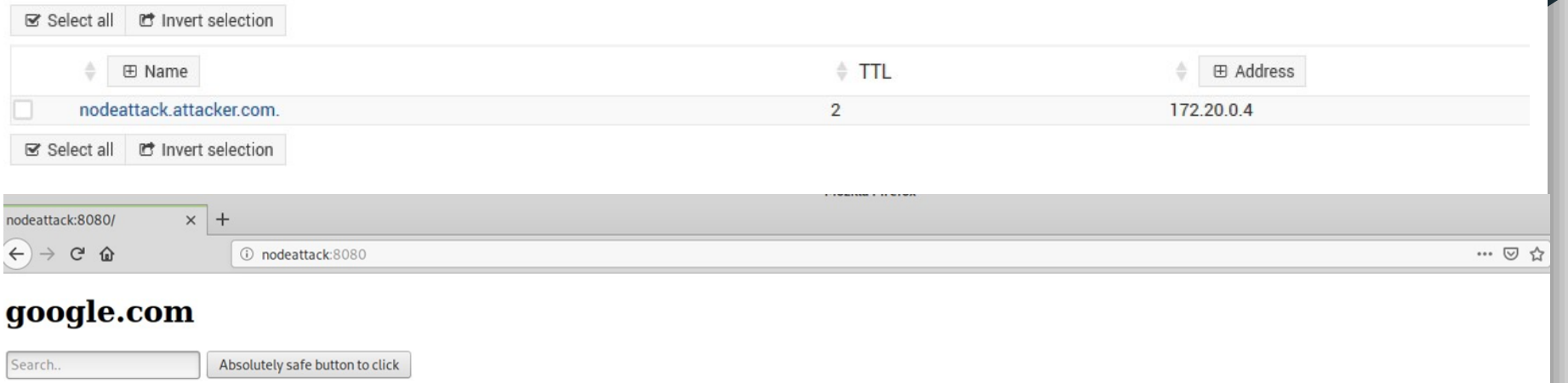
- Set up environment
  - Too many rules in the attack. (Attacker's web server, DNS-server, Target's web server, Victim)
  - If we use VM, we need at least 4 VMs, with VLAN set up.
- Getting LAN IP of target device.
  - The traditional approach is using attacker's advantage.
  - We guess.
- Update DNS record
  - Since Same-origin policy(SOP), we can't update DNS record at victim side.


1st DNS query
x.x.x.x
2nd DNS query
a.b.c.d
attacker.com runs code to
detect local IP
attacker.com: x.x.x.x (TTL 1)
DNS
attacker.com: a.b.c.d (TTL 720)
HTTP POST
attacker.com
temp = 99
IP: a.b.c.d

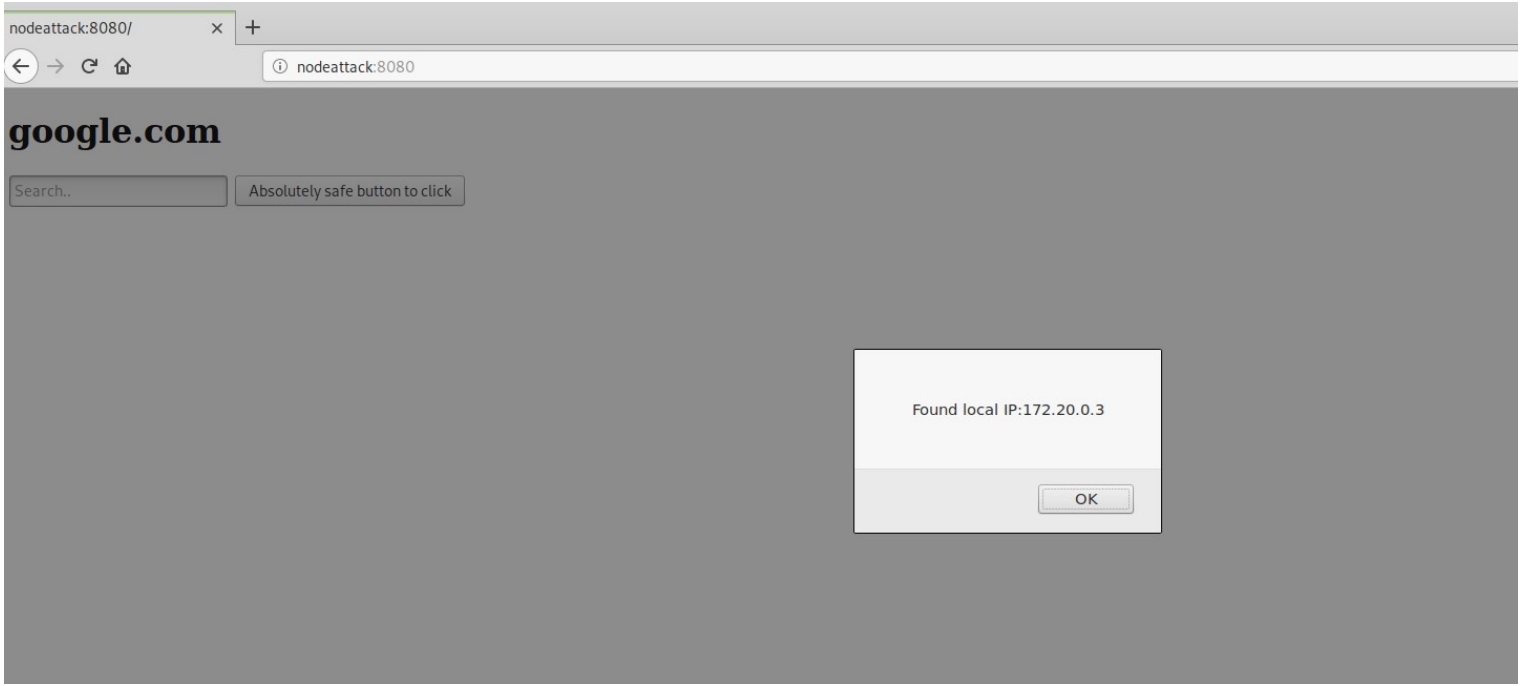
Solutions

- Environment
  - We use Docker for setting up each role in attack
    - Node.js for both web-server
    - BIND for DNS server
    - Linux with xRDP for victim machine.
- Getting LAN IP
  - WebRTC port scanning for target.
    - Private IPv4 addresses: 10.0.0.0/8 , 172.16.0.0/12, 192.168.0.0/16
    - Find subnet first
    - Then scanning thought subnet.
    - Only a need 300 scans.
- Update DNS record
  - Update DSN at backend of Node.js

Results


Select all
Invert selection
Name
TTL
Address
nodeattack.attacker.com.
2
172.20.0.4
Select all
Invert selection
nodeattack:8080/
nodeattack:8080
google.com
Search..
Absolutely safe button to click

Results


nodeattack:8080/
nodeattack:8080
google.com
Search..
Absolutely safe button to click
Found local IP:172.20.0.3
OK

Results

```
 WARNING: This is a development server. Do not use it in a production deployment.
 Use a production WSGI server instead.
* Debug mode: on
* Running on http://0.0.0.0:18180/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 112-187-555
{'host': 'www', 'ttl': '-1', 'ip': '1.1.1.1'}, {'host': 'nodeattack', 'ttl': '720', 'ip': '172.20.0.5'}]
72.20.0.4 - - [20/Mar/2020 07:09:43] "POST /host/nodeattack HTTP/1.1" 201 -
```

Results

- Success: Targeting Firefox browser
  - Acquired victim machine’s internal IP address
  - Located target IoT device IP address
  - Update DNS server to return target IP for attacker’s domain
- To be implemented:
  - Demonstrate exploit on IoT devices.
  - Defense technique (DNS Server and/or firewall filter, DNS pinning)
  - Explore authentication on IoT device

Learned Lesson

- Modern browser are quite secure
- Straightforward in theory; Challenging in configuration and actual implementation
  - Provide a realistic experience of performing attack
- Need better understanding of web development and networking
  - For either security knowledge, or just to become a more competent software developer