Exploiting DNS from Multiple Attack Vectors

Kareem Elkhayat
Department of Electrical Engineering and Computer
Science, University of Central Florida,
Orlando, Florida
ake126@knights.ucf.edu

Allison Young
Department of Electrical Engineering and Computer
Science, University of Central Florida,
Orlando, Florida
acyoung@knights.ucf.edu

Abstract - *In this paper, we demonstrate the multiple levels at which an adversary can exploit the domain name system (DNS) services. We discuss the importance of DNS to availability and security of internet resources. We show that gaining remote or local administrator access to a workstation can direct a user to a malicious site, regardless of using a trusted DNS server. We demonstrate a simulated denial of service attack against a DNS. We also show an exploit scenario in which an unpatched Window DNS server is configured to send dependent devices to malicious sites. Our findings will include suggestions on preventing and identifying DNS-based attacks.*

I. INTRODUCTION

A. Domain Name System

DNS services are a crucial backbone to the internet, connecting users to sites with human-readable domain names and updating records when resources depended upon by applications change location. DNS protocol translates human-readable names into IP-addressed locations on a network. Servers store and update tables containing the results, or resource records (RRs), to return for IP address forward and reverse lookups [1]. The stored tables of RRs are servers' zone files or local cache. Not all DNS servers contain all DNS records; the system is distributed, with servers requesting locations of resources from one another to deliver to clients. The highest level of DNS is the root zone, consisting of 13 primary servers that store records of the top-level domain (TLD) servers for the internet. TLD servers contain the locations of domains, such as ".com", ".edu", and ".io" [2]. The TLD servers then store references to lower levels of the DNS system.

When a client computer requests resources using a domain name, it first checks its local *hosts file*, a text file that contains domain name to IP address resolutions for any hard-coded domains added by an administrator. If there are no entries for the requested domain, the client sends the query to a nameserver or DNS server. There are two types of DNS servers: resolvers and authoritative servers. Resolvers query other nameservers recursively until they find the answer to the DNS query and then respond to the client. Authoritative DNS servers will not query other servers, responding instead with the IP address of the name in the query or with the IP address of another server to query.

DNS is distributed, but different name servers use different operating systems and protocols to communicate. DNS queries are generally done using UDP protocol, which is stateless; because clients expect quick responses from DNS servers, it is unnecessary to establish a three-way handshake for message integrity before responding with the answer to the request [3]. Many servers use BIND, which is an

open source DNS software for Unix and Linux machines, but Windows DNS servers use Microsoft DNS. The IETF (Internet Engineering Task Force) introduced DNSSEC in RFC 4033, which is a set of security extensions that require key signing and authentication to improve the integrity of name resolution responses. However, DNSSEC packets are larger because they are signed, so they sometimes exceed the 512 byte message limit for UDP and must use TCP network protocol [4].

With connected web applications, microservice development, and the Internet of Things (IoT), DNS has increased in importance beyond looking up sites in a browser. Web applications request resources from other services they are integrated with, and they often depend on DNS to access resource endpoints. While they could hard-code IP addresses into their requests, this presents an issue with large applications that may migrate endpoints to support outages or infrastructure changes. An inability for devices or servers to receive answers from a DNS server can cause a full loss of business or infrastructure operations [5].

II. BACKGROUND

In 2014, research conducted by Cloudmark, Inc and Vanson Bourne found that 76% of organizations' IT professionals reported experiencing a DNS attack, often resulting in data or asset loss [6]. The DNS system is highly available due to its distributed nature; DNS servers and resolvers duplicate lookup tables stored on other servers. However, these are often underprotected due to ossification, in which infrastructure is too mission critical to take offline for a hardware or software upgrade [7]. In other organizations, they are not given enough security resources because they do not directly contribute to revenue sources. Overlooked DNS security in small businesses offers a large scope of potential victims to attackers.

DNS attacks are not always easy to mitigate and can lead to organizational losses. In October 2016, a Brazilian bank's internal DNS was compromised and customers were directed to phishing sites instead of any of the bank's domains for over five hours [8]. The attackers took control of the domain registrars to change the addresses associated with bank sites to the attackers' copy sites, harvesting user logins and data. This type of attack is more effective than phishing through spam emails, which may be blocked by a user's mail client. Users affected by the DNS attack would have no way of knowing that their connection was insecure, as the domains still displayed in HTTPS with a valid SSL certificate.

In another case, a University's vending machines were used in a DNS-based DDoS (distributed denial of service) attack against the network. Unsecured IoT vending machines on campus prevented other devices from accessing the internet by spamming DNS servers with requests to resolve the IP addresses of seafood domains [9]. This attack did not originate with a DNS vulnerability, but DNS servers that were not hardened against the DDoS attack allowed for the University network outage.

Because domain resolution is necessary to access resources across the internet, DNS is a high-value target for attackers. The variety of methods of exploitation make it difficult for administrators to maintain DNS performance, let alone protect against attacks. The distributed nature of DNS allows for a breach in the upper levels of domain name resolution to impact dependent clients, including other resolvers

or authoritative servers.

III. DNS EXPLOITS

DNS exploits can be variable and difficult to diagnose; users who rely on sites to serve reliable information may never notice that they are connecting to the wrong resource. Exploits can affect availability, such as a DDoS attack that consumes resources on the DNS server and prevents it from responding to legitimate requests [7]. Man in the middle attacks, rogue DHCP servers, cache poisoning, and malware are other attack vectors hackers can use to compromise the DNS service.

A. Rogue DHCP

If a hacker deploys a rogue DHCP server on a network, they can force users to use their IP address range and DNS servers. The threat with having a rogue DHCP server stems from the fact that DHCP uses a broadcast frame consisting of a MAC address with all fs – such as ff:ff:ff:ff:ff:ff – to request an IP address. Once received, this tells the network to send the frame out all active ports, excluding the port it received the frame on. When a rogue DHCP server is closer to the client than the enterprise DHCP server, the rogue DHCP server will be the first to respond to the frame and issue an IP address, gateway address, and DNS server address to the client [10]. The DNS server does not have to be one located in the enterprise – it can be located anywhere on the internet, making this attack much more dangerous as the hacker can have the client point to multiple malicious DNS servers. Once the client is on the hacker's DNS server, the hacker can redirect any domain to their desired IP address. This can result in attacks that include phishing, malware propagation, and man in the middle attack interception. A rogue DHCP attack could be used in combination with a DDoS attack to send corrupt resolution answers.

B. Cache Poisoning

Cache poisoning, or DNS spoofing, can corrupt an organization or Internet Service Provider's DNS server to point DNS requests for a certain domain to a different IP address. This can be done by creating or overwriting a local record on the server itself without having the DNS server reach out to the authoritative server to resolve the domain name. In this way, resolvers are given a "poisoned" record from a spoofed response [11], [12]. This false record can exist for as long as the TTL (time to live) is set in the spoofed response – hypothetically, this could be indefinite. The victim server will not request that resource record until a TTL is expired, so valid responses that reach the server after poisoning will be ignored.

Similar to a rogue DHCP attack, an attacker using cache poisoning attempts to respond to a recursive query before a valid name server does. In open networks, an attacker might first listen for DHCP and DNS responses to determine their pattern. It would then inject them into the stream of valid responses and reach the victim first. To increase the chance of responding before valid resolvers, attackers can use BGP (Border Gateway Protocol) to conduct *route poisoning* alongside cache poisoning. BGP protocol seeks to improve performance by broadcasting the fastest routes to a client; if an attacker advertises a bad path, they can more easily conduct DNS attacks [12]. Unlike rogue DHCP, cache poisoning is used to spoof a single domain record, as opposed to changing the DNS settings on a machine.

C. DNS Rebinding, Hijacking, and Malware

Malware can depend on DNS, using URLs to reach out to Command and Control hosts. Malware changes local settings on an infected machine, manipulating host files to point specific domains to controlled IP addresses or change DNS server settings on the computer. This is referred to as *DNS hijacking*, where attackers direct all traffic to their owned DNS servers to harvest sensitive data or send users to sites that will download more malicious files on the infected machine [7]. *DNS rebinding* is another type of attack that starts from a user's local computer, executing upon a web page loading a script. This script can expand to change DNS settings for the user's entire network by infecting the router [13]. These exploits occur at a lower level of name resolution with effective results on user machines.

There have been emerging companies that have dedicated themselves to analyzing patterns with malware DNS requests and blocking the resolution of these domains. This has helped the internet contain infectious malware from spreading on the internet, as these companies can find patterns of zero-day exploits and attacks quickly by analyzing and aggregating the DNS traffic. However, removing a malicious resource record from one server cannot remove them from all dependent servers and resolvers once they are affected. Upper level DNS servers cannot revoke domains from downstream caches, leading to a phenomenon, discussed by Duan et al., in which malicious authoritative servers continue to update records in resolvers after a domain is revoked upstream. Recursive requests can perpetually spread bad RRs (*ghost domains*) across the internet, directing users to malware delivering sites [14].

D. Privacy

DNS requests contain sensitive browsing data. A user might be cautious about what sites they visit if they are aware of an ISP viewing and responding to their DNS requests, but as Krishnan and Monrose discuss in their research, DNS prefetching can reveal browsing interests without a user's intending to. Modern browsers like Chrome conduct *DNS prefetching* to resolve domains for links on a page. Without clicking on it, a DNS query is sent on behalf of the user to improve browsing experience [15]. Pre-resolved domains can be stored on the local cache, giving an attacker insight into a user's search behavior.

IV. DEMONSTRATED EXPLOITS

In this next section, we demonstrate simple DNS attacks on local machines and networks within a virtualized lab environment.

A. Compromise local hosts file on target workstation

The simplest method to compromise DNS is through a user's local host file on their workstation. In Unix-based operating systems, this is located at /etc/hosts; in Windows operating systems, this is located at C:\Windows\System32\drivers\etc. The local host file contains mappings from domain name to IP address. Because DNS seeks to resolve a name from the closest source first, modifying the host file can stop the browser from seeking any further resolution of a malicious name. The host file is a text file, making editing easy for any user with access.

Our lab demonstrates a remote exploit of DNS by gaining administrator access to a target workstation using another machine. The target was a virtual machine running an

unpatched install of the Windows 7 Service Pack 1 operating system. We used a virtual machine running Kali Linux with Metasploit Framework version 4.12.22-dev. The target machine's host file was set to the Windows default, where it only contains example mappings commented out with the pound or hash symbol (#). Before performing the exploit, the machine was using Google DNS to resolve "www.google.com" to the correct IP. In the attacker machine, we generated a payload for the target machine to execute. This payload was a binary script named 'kidpix.exe' (with the intention to pose as the popular Broderbund software). Double clicking on the file from the target machine would open reverse shell access for the attacker machine.

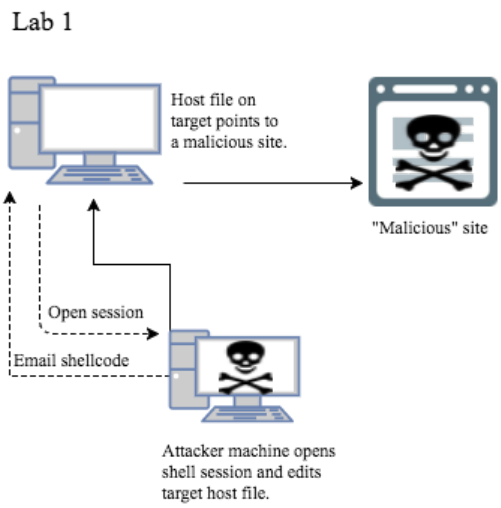


Figure 1. The diagram above shows the attack by Kali Linux to direct the victim to a malicious address.

To gain remote access to the victim machine, we first generated and encoded the 'kidpix.exe' shellcode using msfvenom. The code generated uses a reverse TCP exploit for Windows, where code will execute a command to open a TCP connection to the attacker machine on port 4444. From the attacker machine, we started the payload handler to listen to and respond to the connection. The 'kidpix.exe' payload was then emailed to the 'victim'. From the vulnerable Windows machine, we acted as the victim to open the file, therefore launching the shellcode and creating a connection to the attacker machine.

Once remote access was initiated, we used Meterpreter to act as an administrator on the victim machine. Meterpreter is a tool included in Kali Linux and the Metasploit framework to inject commands into a process. From this tool within the attacker machine, we navigated through the victim directory to find the Windows host file. We modified the contents to include a line resolving 'google.com' to 208.194.116.21. This IP address was not a Google endpoint, but instead pointed the victim to a phishing site, 'ianfette.org', used in the other lab exercises in this paper. This IP no longer resolves to that domain, as it resides on a virtual host.

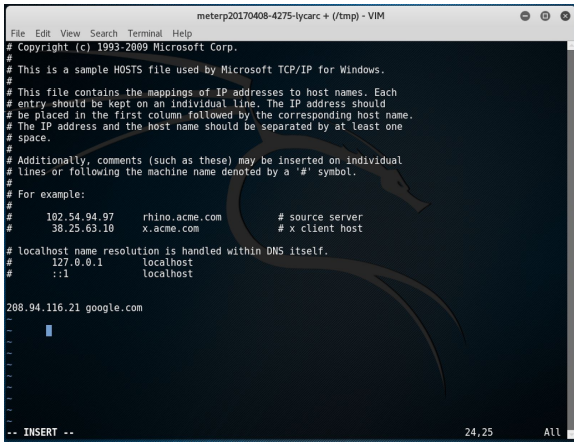


Figure 2. From a Meterpreter session on the attacker machine, we are able edit the victim's host file to point to 'google.com' to another IP address.

From our victim machine, we entered the domain 'google.com' into the browser bar. As expected, the exploit was successful – the browser directed us to the resource indicated in the host file instead of the true Google site. This attack was simple to execute and, with a well-crafted phishing

site, virtually imperceptible to the average user.

B. Execute simulated denial of service against DNS server

We used an open source tool, hping3, to simulate a denial of service attack on an in-lab DNS server [16]. This attack could apply to any type of server, but a flood of packets on a DNS server could result in halting the service to client machines.

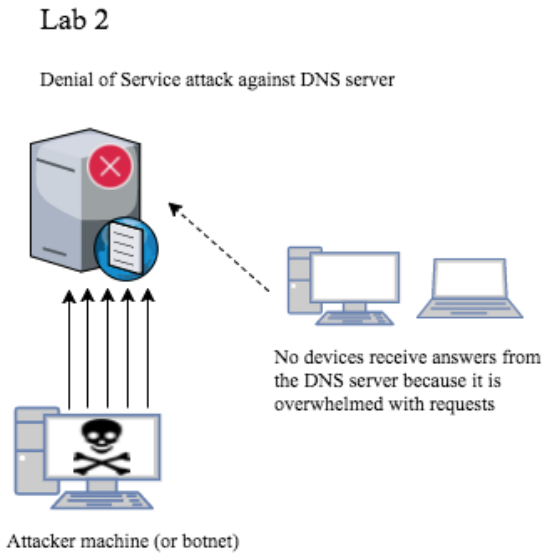


Figure 3. The second attack scenario carried out in our lab sent a flood of packets from multiple VMs to overwhelm a DNS server in a DDoS attack.

For our lab experiment, we utilized a VMware ESXi environment with 8 CPUs x Intel(R) Xeon(R) CPU E5-2630 @ 2.30GHz and 64GB of DDR3 registered RAM. This provided us with enough resources to run our test environment in parallel with other Virtual Machines that we would use to perform the attack. All Virtual Machines shared a 1Gbps connection to an Internet Service Provider business class router. We could not perform a 100% saturation on the link due to shared bandwidth and the router's low processing power. In order to rule out the ISP's router as a potential bottleneck, we created a vSwitch designed to keep all Layer 2 and East-West traffic from being sent outside the ESXi host. However, North-South traffic and ARP resolutions would need to be resolved by the ISP router. Our initial testing found that we were able to lock up the machine by using approximately 5 hping3 streams. We define an hping3 stream as a separate Kali Linux attacker host running a console session of hping3 against the target machine. You can perform multiple streams on a single Kali Linux host; however, you will run into CPU and saturation issues. We performed separate instances of the attack to prevent overburdening a single attacker host.

Our attacking computer was a Kali Linux VM residing in the testing environment and 4 other Kali Linux machines that were residing outside of our testing environment. These VMs were run on the same computer, however, by having them operate outside of our testing environment, we were able to avoid contention of resources and provide them with their own dedicated resources. This allowed us to go from sending around 100 packets per second to over 1000 packets per second.

In order to setup Kali Linux to use hping3, we needed to acquire the hping software package. This was accomplished by using the command below [16]:

```
root@kali:~#sudo apt-get install hping3
```

Once we had the package installed, we then passed the following variables into the command line and started the attack:

```
root@kali:~# sudo hping3 -i u1 -S -p 80 172.16.0.38
```

The following parameters have hping3 perform our desired action:

- '-i' changes the interval from the default to the user specified variable of 1 second or 'u1'
- '-S' indicates the use of the Syn flag
- '-p' specifies the port we would like to flood, which in this instance was '80'[16]
- The last parameter is the IP address of the host we would like to attack.

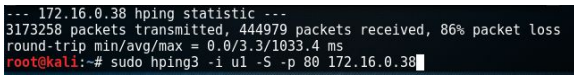


Figure 4. The packet loss indicates that sending multiple packets to a DNS server will prevent it from responding to valid requests.

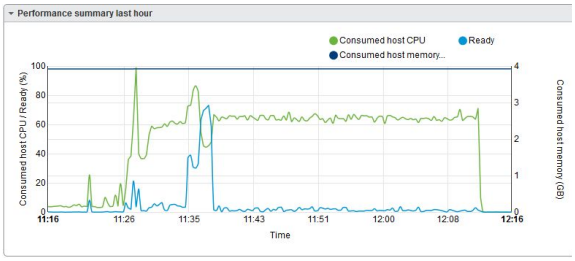


Figure 5. The VM that was being attacked was contesting for resources from the ESXi server.

We can see in Figure 6 and Figure 7 that we were able to hit approximately 96% CPU utilization with about 2100 packets per second and 4 hping3 streams. Increasing both the number of packets and hping3 stream, we were able to completely lock up the system to the point of the GUI not responding and attacking systems reporting dropped packets.

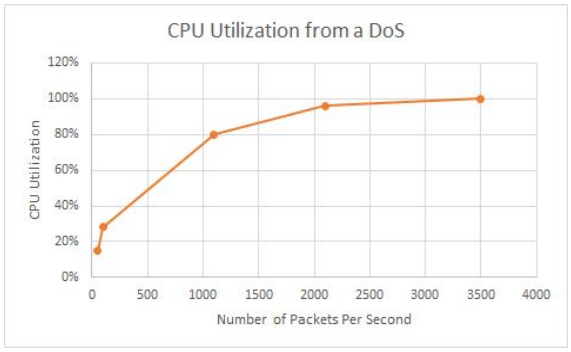


Figure 6. This graph shows the CPU Utilization increasing exponentially with the number of packets sent.

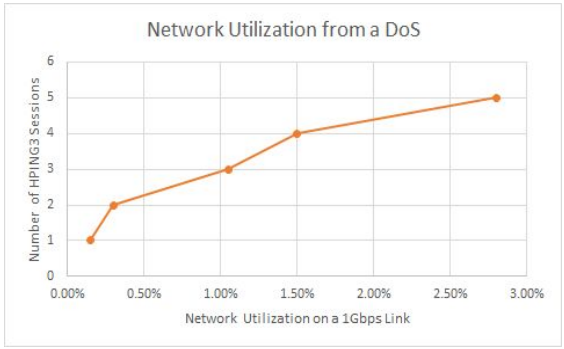


Figure 7. The link does not need 100% saturation to overload the CPU.

C. Gain shell access to exploit DNS server

The next attack scenario we demonstrated used Kali Linux to compromise a Windows 2008 DNS server. We performed a Nessus scan to find CVE MS09_050, a vulnerability in SMBv2 which can enable remote code execution if the attacker sends a specific packet to the affected machine [17]. We used this exploit against our Windows 2008 DNS server using Kali Linux and Metasploit framework to alter DNS settings on the target server.

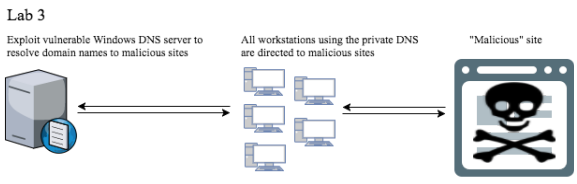
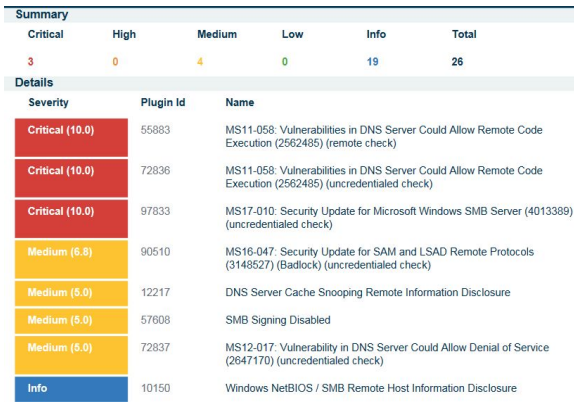


Figure 8. An exploited server can misdirect all machines to a malicious site.



Summary

Critical	High	Medium	Low	Info	Total
3	0	4	0	19	26

Details

Severity	Plugin Id	Name
Critical (10.0)	55883	MS11-058: Vulnerabilities in DNS Server Could Allow Remote Code Execution (2562485) (remote check)
Critical (10.0)	72836	MS11-058: Vulnerabilities in DNS Server Could Allow Remote Code Execution (2562485) (uncredentialed check)
Critical (10.0)	97833	MS17-010: Security Update for Microsoft Windows SMB Server (4013389) (uncredentialed check)
Medium (6.8)	90510	MS16-047: Security Update for SAM and LSAD Remote Protocols (3148527) (Badlock) (uncredentialed check)
Medium (5.0)	12217	DNS Server Cache Snooping Remote Information Disclosure
Medium (5.0)	57608	SMB Signing Disabled
Medium (5.0)	72837	MS12-017: Vulnerability in DNS Server Could Allow Denial of Service (2647170) (uncredentialed check)
Info	10150	Windows NetBIOS / SMB Remote Host Information Disclosure

Figure 9. An example of the Nessus scan outputted results.

Within Metasploit's msfconsole, we used the command 'use exploit/windows/smb/ms09_050_smb2_negotiate_func_index' to load the exploit. From here we set the remote host with the vulnerability, or 'RHOST', to the IP address of 172.16.0.25, our target DNS server. We set the 'LHOST' IP address to 172.16.0.20, our Kali Linux attacker machine. The last step before issuing the exploit command is to set the payload, done with command 'set payload windows/meterpreter/reverse_tcp'. The output below was shown when we were able to successfully gain access into the DNS server.

[*] Started reverse TCP handler on 172.16.0.20:4444
[*] 172.16.0.25:445 - Connecting to the target (172.16.0.25:445)...
[*] 172.16.0.25:445 - Sending the exploit packet (930 bytes)...
[*] 172.16.0.25:445 - Waiting up to 180 seconds for exploit to trigger...
[*] Sending stage (957999 bytes) to 172.16.0.25
[*] Meterpreter session 1 opened (172.16.0.20:4444 -> 172.16.0.25:49159) at 2017-04-08 22:21:25 +0000

Once we gained command line access to the DNS server, we navigated to 'C:\Windows\System32\drivers\etc\' and modified the hosts file to redirect DNS requests for 'google.com' to IP address 208.94.116.21. If this machine were relied on by other clients, we could alter zone files on the compromised DNS server to point to malicious sites for multiple victims or take the server offline completely. Controlling a DNS server through a simple persistent vulnerability allows an attacker to intercept all incoming requests as well.

D. Controlled DNS insights

For the last attack scenario, we configured a Windows Server 2012 DNS server to point to our malicious site, 'ianfette.org', instead of 'google.com', and collected packets from the network by pointing the Internet Service Provider router to the controlled server. The ISP router model is an Arris DG1670a which has a 600Mhz dual core CPU. In a DNS rebinding attack, we could infect the router to use the DNS we indicated, but in this lab we used physical access to the router to alter network DNS configuration. Because routers come with default passwords, it is also sometimes possible to conduct an authority impersonation attack and gain access to admin privileges with stolen credentials [7].

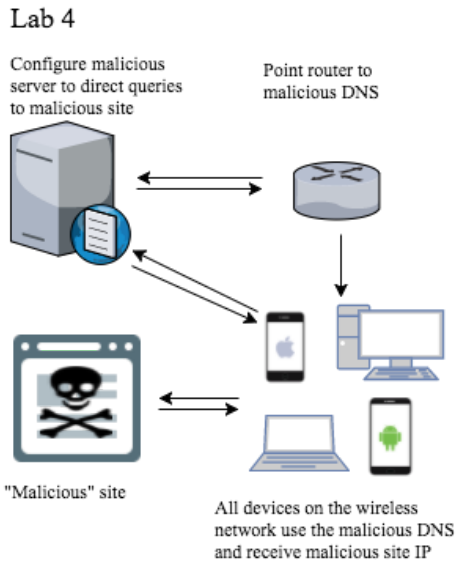


Figure 10. Directing a router to a malicious DNS or a DNS server with poisoned records will send all clients on the network to a malicious site.

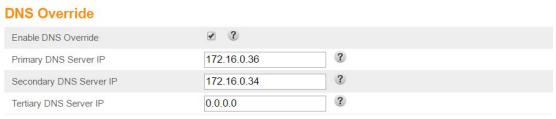


Figure 11. The GUI of the ISP router overriding the default configuration to use our compromised DNS server

After pointing the ISP router to the controlled DNS server, we demonstrated how domain spoofing can be invisible to users by visiting 'google.com' to instead be directed to 'ianfette.org'.

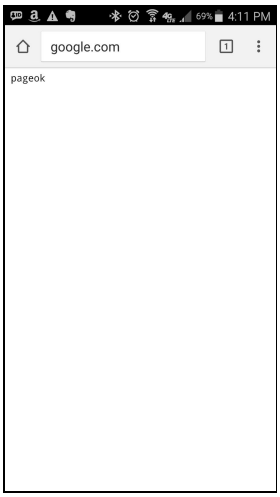


Figure 12 (left). There is no warning on a mobile phone or desktop browser when accessing a spoofed domain on a compromised network.

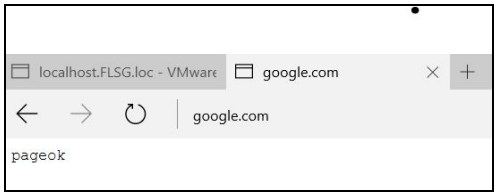


Figure 13. There is no warning in the browser that this is a spoofed DNS record.

Access to intercepted packets can give an attacker insight into client vulnerabilities. The contents of packets show protocol, which can indicate the types of devices in a network, and browsing behavior.

Capturing the contents of DNS queries to harvest destination URLs can give attackers ideas of which sites to spoof. This information is valuable in setting up phishing attacks against users. For instance, if an organization uses a particular cloud service to store employee clock-in and clock-out times that integrates with a cloud payroll service, attackers could mirror this site to collect employee login information which can then be used to change the direct-deposit bank account information into a bank account held by the hacker. The payroll application may also contain personal and financial information in regards to the employee such as name, address, social security number, email address, etc. The mirrored website would need to parse the same information as the real website, which would require time for the hacker to emulate perfectly. Once the website is built, the hacker would then need to redirect the URL to his IP address which would compromise the website.

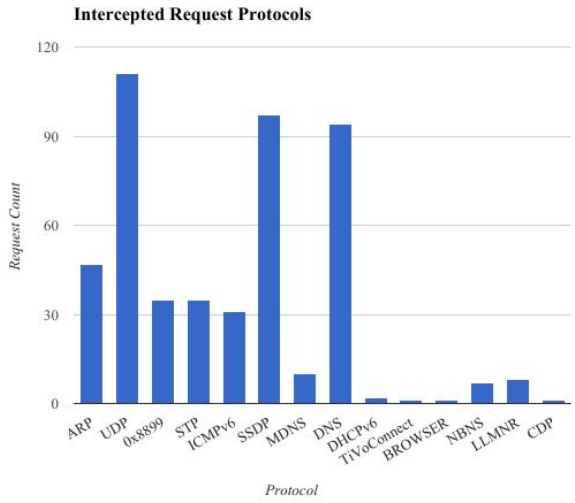


Figure 14. Routing all network requests to a malicious DNS can collect data from a large spectrum of devices, as shown by the protocols above.

In addition to showing the reconnaissance that an attacker can do through controlled DNS, the variety of protocols captured by Wireshark for DNS queries shows how many devices require name resolution. ARP, UDP, ICMPv6, MDNS, DNS, and DHCPv6 are all used in name resolution.

IV. DEFENCE AGAINST DNS ATTACKS

There are many recommendations to help defend against DNS attacks. We will go over some of the most common ways to mitigate the attacks on the DNS service.

Device 'hardening' is when protective measures are implemented around your network or hardware devices to make them less susceptible to known attacks. Hardening includes securely configuring operating systems and applications according to the individual company's or the industries best practice. Filtering against suspicious DNS traffic and deploying real time flow monitoring can create a near immediate response to threats. Hardening can also include 'over-provisioning' against resource depletion. *Over-provisionin*g is creating resources that can handle a load of 10 to 100 times the current load to protect against DDoS attacks that will deplete the available resources. It can also entail providing multiple links and paths to your DNS servers to prevent a single point of failure such as a fiber cut. Administrators can harden networks by adding redundancy to their infrastructure, with load balancing and fallback machines in place in case of server failure.

We advise that administrators monitor 'patch Tuesday' when tech companies, such as Microsoft, release security patches to close vulnerabilities in their system [18]. While this does not always relate directly to DNS, securing the software that is running on the server can prevent remote code execution and known vulnerabilities that hackers can exploit. Devices other than DNS should also be secured to protect against infecting a network through DNS rebinding or being directed to a phishing site through malware that overwrites local DNS settings [13] [18].

DNS data can be corrupted in many different ways. Some of which include the propagation of corrupted DNS data from an authoritative server, within the DNS cache itself, or data that is in transit to its source or destination. Focusing on data corruption during transit, this can be due to poor infrastructure, or when using unencrypted DNS traffic, a man in the middle attack can be performed. An attacker with control over the intermediate network can implement a variety of attacks, such as cache poisoning, through manipulation of the DNS traffic [7]. It is recommend to secure the transit path of DNS messages as much as possible and to restrict domain name zone transfers.

DNS amplification attacks use the DNS service to amplify the power of a DoS attack. *Authoritative amplification* is when an attack reflects DNS traffic against a

resolver and returns packets that are larger in size than need to be, such as if an attacker were to send a 28-byte query to a DNS server for the NS record associated with a root server, the response received would be over 800-bytes. When used in conjunction with spoofing the source address of the query, the amplified response can perform a DoS attack on the target. Since DNS traffic appears to be the same with a spoofed source address, the use of 'Network Ingress Filtering' is recommended as well as rate limiting multiple requests from the same source address [7].

The threats of the system becoming flexible and alternate roots are lower on the spectrum than some of the other attacks, there are mitigation techniques that exist to reduce the risk of exposure. When speaking of DNS flexibility, we refer to the implementation of new standards, as many organizations have grown lax in conforming to recommended standards. For example, the deployment of DNSSEC is dependent on the support of new protocol extensions, but it was found that the infrastructure equipment did not support the new protocol extension, making the use of DNSSEC impossible in environments using such equipment. The threat of alternative roots is lessened with ICANN's deployment of International Domain Names (IDNs), which was implemented in 2007. New generic TLDs have also helped to mitigate this threat [7].

The best defense available to DNS actors currently is the widespread implementation of DNSSEC. A single corrupted DNS can broadcast corrupt paths to downstream servers regardless of their security. For this reason, most technologists advocate for the use of these extensions to increase the authenticity of query responses. However, not all levels maintain patches or perform key-signing. The 13 root servers did not implement DNSSEC until 2010, when ICANN and VeriSign established a public key signing ceremony [19].

V. CONCLUSION

In this paper, we have shown that DNS is a high-value target for attackers because of the variety of attack vectors. Our labs demonstrate the ease of conducting simple attacks, which can be combined and scaled to target larger organizations for malicious purposes. Because domain name resolution stops searching for RRs after the first response from a server, DNS cache poisoning, hijacking, and spoofing can be effective and hard to detect attacks.

As DNS is a critical component of the Internet and will continue to be targeted for malicious intent and Denial of Service, there has been a larger focus on the evolution of the DNS protocol and best practices used to ensure the smooth operation of DNS. The need for security, stability, and redundancy will continue to grow to meet the matching demand of the growth of the Internet.

Administrators should not neglect DNS security when reviewing risks to infrastructure. As discussed, DNS is layered and interdependent with other servers and clients in resolving domains. One weak point can create an opening for an attacker to exploit more secure servers. Utilizing DNS traffic filtering, DNSSEC, and auditing general server and client software security will strengthen the overall system against attacks.

VI. REFERENCES

[1] D. Karrenberg, “The Internet Domain Name System Explained for Non-Experts”, no. 16, Internet Society. [Online]. Available: https://www.internetsociety.org/sites/default/files/The%20Internet%20Domain%20Name%20System%20Explained%20for%20Non-Experts%20%28ENGLISH%29.pdf. [Accessed April 29, 2017].

[2] S. Baranowski, “How Secure are the Root DNS Servers?”, March 2013, SANS Institute. [Online]. Available: https://www.sans.org/reading-room/whitepapers/dns/secure-root-dns-servers-991. [Accessed April 30, 2017].

[3] P. Mockapetris, "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987. [Online]. Available: https://www.ietf.org/rfc/rfc1034.txt. [Accessed April 30, 2017].

[4] D. Massey, M. Larson, R. Arends, R. Austein, and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005. [Online]. Available: https://www.ietf.org/rfc/rfc4033.txt. [Accessed April 30, 2017].

[5] “The Importance of a Resilient DNS and DHCP Infrastructure”, 2014. [Online]. Available: Global Technology Resources, Inc., http://www.gtri.com/wp-content/uploads/2014/08/GTRI-Resilient-DNS-and-DHCP-Infrastructure-White-Paper_140402.pdf. [Accessed April 30, 2017].

[6] “New Research Reveals More than Three Quarters of Organizations Have Suffered a DNS Attack”, Cloudmark, December 16, 2014. [Online]. Available: https://www.cloudmark.com/en/press/new-research-reveals-more-than-three-quarters-of-organizations-have-suffered-a-dns-attack. [Accessed April 29, 2017].

[7] D. Conrad, “Towards Improving DNS Security, Stability, and Resiliency”, 2012. [Online]. Available: Internet Society, https://www.internetsociety.org/sites/default/files/bp-dnsresiliency-201201-en_0.pdf. [Accessed April 30, 2017].

[8] F. Rashid, “Clean up your DNS act or get pwned like this bank”, April 11, 2017. [Online]. Available: InfoWorld, http://www.infoworld.com/article/3188884/security/clean-up-your-dns-act-or-get-pwned-like-this-bank.html. [Accessed April 29, 2017].

[9] M. Smith, “University attacked by its own vending machines, smart light bulbs & 5,000 IoT devices”, February 12, 2017 [Online]. Available: http://www.networkworld.com/article/3168763/security/university-attacked-by-its-own-vending-machines-smart-light-bulbs-and-5-000-iot-devices.html. [Accessed April 29, 2017].

[10] Y. Bhaiji, “Understanding, Preventing, and Defending Against
Layer 2 Attacks”, 2009. [Online]. Available: Cisco, http://www.cisco.com/c/dam/global/en_ae/assets/exposaudi2009/assets/docs/layer2-attacks-and-mitigation-t.pdf. [Accessed April 30, 2017].

[11] S. Son & V. Shmatikov, “The Hitchhiker’s Guide to DNS Cache Poisoning”. [Online]. Available: The University of Texas at Austin, https://www.cs.cornell.edu/~shmat/shmat_securecomm10.pdf. [Accessed April 30, 2017].

[12] H. Shulman & M. Waidner, “Towards Forensic Analysis of Attacks with DNSSEC”. [Online]. Available: IEEE Computer Society, http://www.ieee-security.org/TC/SPW2014/papers/5103a069.PDF. [Accessed April 30, 2017].

[13] A. Barth, D. Boneh, A. Bortz, C. Jackson, & W. Shao, “Protecting Browsers from DNS Rebinding Attacks”. [Online]. Available: https://crypto.stanford.edu/dns/dns-rebinding.pdf. [Accessed April 29, 2017].

[14] H. Duan, K. Li, J. Jiang, J. Li, J. Liang, & J. Wu, “Ghost Domain Names: Revoked Yet Still Resolvable”. [Online]. Available: Internet Society, https://www.internetsociety.org/sites/default/files/12_1.pdf. [Accessed April 30, 2017]

[15] S. Krishnan & F. Monrose, “DNS Prefetching and Its Privacy Implications:
When Good Things Go Bad”. [Online]. Available: .https://www.usenix.org/legacy/event/leet10/tech/full_papers/Krishnan.pdf. [Accessed April 30, 2017].

[16] S. Moon, “TCP SYN flood DOS attack with hping”, November 2, 2011. [Online]. Available: http://www.binarytides.com/tcp-syn-flood-dos-attack-with-hping/. [Accessed April 8, 2017].

[17] “Microsoft Security Bulletin MS09-050 - Critical”, October 13, 2009. [Online]. Available: https://technet.microsoft.com/en-us/library/security/ms09-050.aspx. [Accessed April 29, 2017].

[18] S. Nichols, “It's nearly 2016, and Windows DNS servers can be pwned remotely.”, December 18, 2015. [Online]. Available: *The Register,* https://www.theregister.co.uk/2015/12/08/patch_tuesday_december2015/. [Accessed April 29, 2017].

[19] “Root DNSSEC”. [Online]. Available: http://www.root-dnssec.org/. [Accessed April 30, 2017].