Off-Path Attacks Against Public Key Infrastructures

Arizona State University CSE 548 Advanced Network Security

Derbique, Austin
aderbiqu@asu.edu

April 17, 2021

Abstract

Public Key Infrastructure (PKI) is an ubiquitous method for securing applications and systems that deal with protecting data. While the cryptography and computer security research efforts spent on securing PKI for general use across all of computing, a system is only as strong as its weakest link. In this paper, we examine a 2018 Black-Hat conference presentation on *Off-Path Attacks Against Public Key Infrastructures* in which the presenter successfully uses DNS cache poisoning as a means to circumvent domain name validation to obtain a valid, certificate authority signed SSL certificate [6]. To better asses this attack and how this is possible, we implement a similar test environment set of algorithms to focus on the attack portion of this presentation and attempt to recreate it. In order to achieve this, we create several virtual machines, stand up a domain name server along with a suite of attack tools on the attacker node, run several scripts to monitor outcomes. Later, we perform analysis to observe the viability of an attack in the real world. Finally, we check to see if a similar attack is still possible today, and if it is, the size of the attack surface.

Keywords— Public Key Infrastructure, DNS Cache Poisoning, ICMP Fragmentation, Network Security, Network Attacks

I. Introduction

A. Black Hat Project

Black Hat is a world renowned information security event where new, cutting-edge security research is discussed [1]. A range of topics are discussed from cryptography, to mobile computing, to network security. For this take home exam, we were tasked with selecting a session topic to research further. For my topic, I chose the Network Defense and Applied Security category. This seemed like a good path to take as applied security is actionable, often based on applying theory or exploiting bugs in a given protocol.

The Black hat project I chose has to do with off-path attacks exploiting DNS caching mechanisms in order to circumvent verification processes for public key infrastructure. Off-path attacks are attacks carried out where the malicious agent is not directly intercepting and modifying the packets of information, otherwise known as man-in-the-middle (MitM) [11]. The blackhat project selected highlights that even with strong cryptographic foundations, PKI and challenge-response validation is built on top of weak building blocks, subject to exploitation that compromise security of the system in its entirety. To better understand this, we briefly discuss what was accomplished in this blackhat project.

The first step in the attack consists of an attacker carrying out defragmentation cache poisoning. While other DNS cache poisoning methods have proven to be successful like the famous attack performed in 2008 [13], many have since been patched. [RFC5452] [14]. This presentation uses a method of lowering the Maximum Transmission Unit (MTU) from the server to the certificate authority such that packets become fragmented, or split up into multiple pieces. These fragmented packets are cached on the resolver until all pieces are present, and then the packet is recompiled and sent off. This particular attack utilizes ICMP fragmentation in order to reduce the MTU of the name server. This allows for the next step of the attack to take place.

Next, fake fragments from the attacker are sent to DNS resolver with the goal that the resolver will assemble the legitimate first fragment with the forged second fragment. The resulting packet will contain information that points the DNS record to the attacker's IP address. By the time the

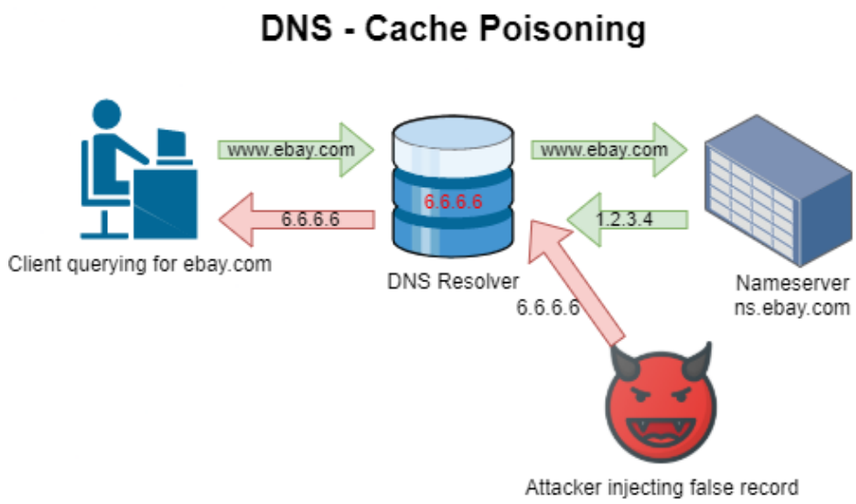

Figure 1: This figure depicts DNS Cache Poisoning in which a client performs a forward lookup for www.ebay.com to it's DNS resolver. The DNS resolver performs a recursive lookup and reaches ns.ebay.com which returns the address 1.2.3.4. At this point, the attacker injects a forged fragmented packet containing the attacker's IP address instead of the actual IP address. The packet is reconstructed and the false IP address is returned to the client.

legitimate second fragment arrives, the packet will already have been reconstructed with the forged second fragment.

with the DNS Cache attack now complete, the attacker demonstrated sends a new CSR with the victim's DNS name to the certificate authority with a mail based challenge response for proof of ownership of the domain. The challenge is now mailed to the attacker's IP address instead of the victims. Upon validating the challenge with the response code email, the attacke can now download the signed certificate from the CA with the victim's DNS name listed. This is a legitimately signed certificate by the CA.

B. Deliverables

In this project, we attempt to re-create the attack portion of this demonstration. Namely, the act of lowering the maximum transmission unit (MTU) by ICMP fragmentation. The second portion of the attack is sending the falsified fragments to the DNS resolver. In order to accomplish these two tasks, a test network must be built out consisting of an attacker node, a service node with a DNS resolver, and a victim node. Additionally, automation scripts will be needed for deploying and running our implemented solution. Finally, analysis will be performed if this attack which was discussed in 2018 is still exploitable today.

II. Description

A. System Model

The system model consists of several components. First, there is the virtual machine and networking layer of the model. Second, there is the applications layer of the model. Finally, there is the attack flow. The attack flow is described in security model below.

Virtual Machines: In order to simulate a test network for the purposes of carrying out a simulated attack, several virtual machines will be required. The concept is pretty straightforward. The virtual machines required are as follows:

1. An Attacker virtual machine which carries out the attack.

2. A Server virtual Machine which runs a DNS resolver and Web Server.

3. A Victim virtual machine which has a legitimate DNS name and IP associated with it.

4. A deployment host to orchestrate the initialization, configuration, and attack procedure.

Networking Networking for the test environment consists of bridged adapters allowing the host machine the ability to communicate with the guest machines. The guest virtual machines have the ability to communicate with each other. The deployment host can login to each of the attacker,

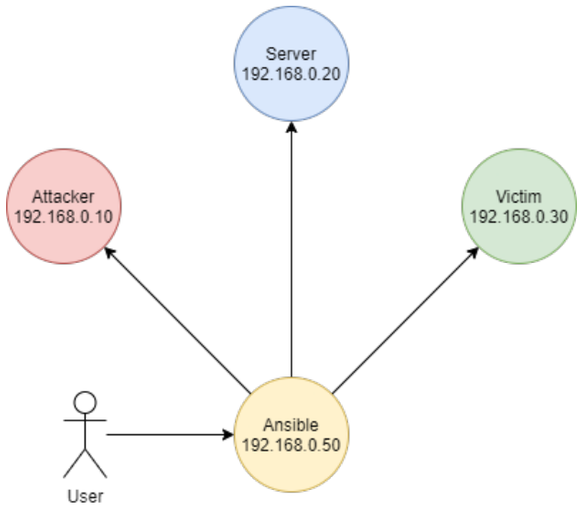

Figure 2: The network topology that we implemented to conduct tests on. Each node is a separate virtual machine connected together according to the edges shown.

server, and victim VMs to perform maintenance. Each of the machines are given a static IP address. The attacker is 192.168.0.10, server is 192.168.0.20, victim is 192.168.0.30, and the deployment VM is 192.168.0.50. Finally, because the server is the VM that contains a DNS resolver, the attacker and resolver both use server's domain name server as their own local DNS resolver. This means that if server goes down, DNS does not work for the attacker or the victim.

Deployment To provision the virtual machines, I used Ansible. Several purpose built "playbooks" were written to configure each of the virtual machines to become their specific "role". This automation allows for installing specific packages, copying over customized files, and executing arbitrary scripts. For the server, a Bind9 DNS Server is installed using Ansible's server role and custom configurations for forward and reverse zone files are copied to the server. Additionally, an Nginx web server is deployed and custom scripts are copied to the other hosts. Finally, this deployment mechanism gives full functionality for rebooting the test network, rolling everything back to a vanilla state, or carrying out the attack.

Pivots The original plan for implementation was to include all aspects of what was shown in the Blackhat project, including implementing a web-based certificate authority. Unfortunately, it appears that no off-the-shelf open source applications exist that provide web based PKI solutions. Due to this limitation, project scope was narrowed to focusing on the attack, rather than what comes after the attack. Additionally, Amazon EC2 was going to be used for virtual compute and networking, however it was deemed local VMs running in Virtualbox would meet all requirements for the project.

B. Software

Various tools we have used are as follows:

- **Oracle VirtualBox:** Virtualbox provided a virtualized computing platform for running the test network. Running on a Windows 10 host, Virtualbox hosted four different virtual machines. Each of the virtual machines ran Ubuntu Server 20.04 LTS [17].
- **Ansible:** Ansible is used as the configuration management orchestrator for carrying our deployments and running the attack [4].
- **Bind9 DNS Server:** The Bind9 DNS Server will provide a name server that will be victim to the attacker. This will run on the "server" virtual machine and its IP address will be used as the DNS resolver for other virtual machines [5].
- **Nginx Web Server:** A Nginx web server will be used to serve basic html content, allowing other machines to perform DNS lookups of the web server [2].
- **OpenSSL:** OpenSSL will be used as the public key infrastructure (PKI) mechanism used. While PKI itself will not be exploited, the building blocks beneath the PKI will be exploited, leading to compromised SSL certificates to be created [15].
- **Traffic Shark:** Will be used for obtaining general information about the network. This will be used to capture packets and monitor how the attack works [12].

- **hping3 Network Security Tool:** This opensource tool will be used to assemble and analyze UDP data packets. It will also be used for generating fragmented packets sent to the DNS resolver as part of the attack carried out by the attacker [3].
- **Python:** The attack script will be written using Python 3 and will wrap various commands such as hping and traffic shark to help carry out the attack [16].
- **Draw.io** This will be used to generate figures for the report [10].

C. Security Model

Domain Name System Cache Poisoning is nothing new. With the first notable attack presented in 2008 [13], ongoing improvements have been made to the algorithms and protocols responsible for securing such systems. Upon choosing this project and learning more about the topic, it is clear that some of the largest vulnerabilities had to do with poor security assumptions regarding what level of entropy is sufficient to ward off an attacker. Put another way, this security model exploits one of the key vulnerabilities in packet defragmentation. That is, with the correct Internet Protocol Identification (IPID) value, fragmented packets can be recompiled to form a complete packet. While simple, the attacker still needs to correctly guess the challenge-response authentication parameters in the DNS request. While this may sound hard to achieve, it is poor assumption of what level of entropy that is required that makes this fail. By generating tens of thousands of packet fragments all with incremental changes to the IPID values and other key attributes, at least one packet is bound to be accepted. Because the MTU is lowered, this allows for more time for the attacker, therefore increasing the possibility for a correct combination to be guessed.

III. Project Description

A. Project Overview

In general, this project can be broken down into several key components. First, a portion of the paper must be written to provide guidance on flow of the project. Next, a test network needs to be created and configured to allow for the attack to occur. After a functioning network with DNS server is in place, the attack can be written and tooling for carrying out the attack deployed. Finally, results and conclusions are drawn up and placed back in the paper as well as a demonstration video included with submission. Below is a task breakdown discussing in more detail what each task entails as well as dependencies for each task.

B. Task 1: Create Report Document

In order to better define the goals, deliverables, and outcomes of the project, a roadmap has to be established. This task's purpose is to define those key pieces to help give clarity to the project and ensure all components are completed.

C. Task 2: Provision Network Resources

Research on best platforms for mocking up the network must first be completed before acting on provisioning the network resources. Two likely possibilities include running the network on Amazon EC2. The other includes Oracle Virtualbox. Due to the cost associated with Amazon EC2, virtualbox will likely be selected, as well as its ease of use in provisioning said resources.

D. Task 3: Create Test Network

With a platform in place for simulating network resources, virtualized compute has to be created. This consists of creating four virtual machines, all within the 192.168.0.0/24 network. The deployment VM will be used to connecting to the other three virutal machines used to facilitate the test network environment. The test network is intended to be "real enough" such that is mimics a real life scenario of an off-path attack.

E. Task 4: Run Services on Test Network

In order to ensure that the attack can work successfully, a systems check must first be run on the test network. This includes testing for connectivity, proper dns name resolution, as well as ensuring all packages and libraries are up to date. This is a combination of manual checks as well as Ansible playbooks to accomplish this task.

F. Task 5: Create DNS Cache Attack Utility

This task involves implementing a portion of the attack from the Blackhat project in which the attacker spoofs the fragmented packet to the DNS resolver. This task will rely heavily on outside resources such as hping, tshark, and other libraries available for carrying out network related attacks. The logic of this will likely be written in Python as a wrapper and will be executed on the attacker host virtual machine.

G. Task 6: Test DNS Cache Attack Utility

A simple python script will be written to validate that the attack is successful by performing a forward lookup of the DNS result to show the attacker's IP address instead of the victim's. Results of this will be dumped to the terminal screen.

H. Task 7: Gather and Evaluate Result

Upon completing the previous tasks, results will be gathered and evaluated. This portion also includes reading up on current research to see if patches have been made to nullify this attack strategy or at least check if the attack vector has been mitigated. This results will be distilled into a figure or two to be shown in the final report or in the demonstration video.

I. Task 8: Finalize Report Document

Compile all results and efforts from previous tasks and place back into the report document. Perform a fluency check and also check for grammatical errors.

J. Task 9: Create Video Demonstration of Project

The final task includes creating a video up to 10 minutes in length to describe the Black Hat project and my contributions of implementing the attack. This will be a brief overview and also discuss outcomes.

IV. Discussion

In today's world, we often take security for granted. Nearly every website has TLS enabled with an SSL certificate, and we rely on our browser to perform verification. The research performed here explores an attack that is blind to CA signature checks as the signatures are completely valid. The damage from a large scale attack like this has the potential to be catastrophic, therefore the more we understand how the attack works, the better prepared we will be for coming up with a solution to defend against it.

A. Implementation of Proposed Methodology

Test Network For the test network, Oracle Virtualbox was ultimately chosen. I struggled for several hours getting the networking to behave how I wanted, but eventually discovered that a bridged networking adapter on each of the virtual machines would allow them to communicate with each other. Using Ubuntu 20.04, each of the machines were static IP'd. Route tables were untouched, however DNS resolvers pointed to the server virtual machine (192.168.0.20).

Deployment Mechanism Ansible was used to carry out deployment. Scripts for setting up the environment are available in the Github repository. The Ansible deployments responsible for initializing and configuring server infrastructure as well as carrying out the attack was deployed from the Ansible virtual machine (192.168.0.50). This served as a central place of operations relating to control of the project. The automated configuration management solution proved extremely useful in repeating testing of the attack over and over again.

DNS Cache Poisoning This was arguably the most difficult portion of the project. In the blackhat presentation, a lot of assumptions are made that are glossed over in the presentation slides. This is likely because of the short amount of time available for them to present. Further reading was required from several research papers on the topic of DNS Cache Poisoning and IP Fragmentation. It turns out that there are actually multiple different types of Off-Path attacks related to DNS. Most have been patched, however due to the nature of how this attack works, there is no pure fix. There are only mitigations. As mentioned in the paper by *Brandt et. al*, the best current way to mitigate off-path attacks like this is to use DNSSEC, however this very solution

relies on PKI, which has been shown to be subject to such attacks. Therefore, the root problem of off-path DNS cache poisoning attacks still exist today [8].

Attack Algorithm Overview The attacker script psuedocode to poison a DNS resolver's IP Fragmentation Cache is shown below.

1. Estimate IPID speed of Victim name Server
2. Query victim name server for fragmented response
3. Forge 2nd Fragment
4. Spoof 2$\hat{6}$4 forged 2nd fragments with different IPIDs and send to DNS resolver
5. Run a listener that checks to see when the DNS cache is updated with forged values

B. Challenges

There were several challeneges with this project. First and foremost, the time constrains for the project meant there was not a lot of time to backtrack or find alternate paths. As such, once the topic of Off-Path attacks against PKI was selected, I proceeded with the project as best as I could. Minor challeneges such as networking presented themselves, but these were worked out.

The second largest challenge of the project had to do with generating the attack utility for DNS poisoning. While academic papers exist, it appears most of them intentionally leave out technical details. I would imagine this is because of the large risk if an attacker carried out an attack for real on the internet.

C. Analysis

Overall, this project has shown that an attack today is still possible if all the right combinations are set. This includes a CA that allows for fragmentation, that the attacker can send out the fragmented packets quickly enough before the legitimate packet arrives, that the valid entry is not already cached in the resolver. While this attack is possible, it is unlikely. That said, the consequences of such an attack could be catastrophic. That is why some recommendations include disabling fragmentation, implementing a new type of domain validation algorithm [8], and doing a better job at rate limiting requests sent to the DNS resolver from neighboring hosts.

V. Conclusion

In this paper, I have described what off-path attacks are and how they can be used for DNS cache poisoning using ICMP defragmentation. I provided an overview of the Black hat project as well as mentioned related works. A test network was stood up in order to implement the attack algorithm used in the Black hat project. Finally, I discuss project challenges and provide an analysis of the project as well as future directions for the work.

Youtube Video: `https://youtu.be/76vzbf7bOdc` [7]

Github Repository: `https://github.com/aderbique/CSE548` [9]

Acknowledgement

The author thanks Dr. Dijiang Huang for his guidance on selecting a novel research topic.

References

[1] URL: `https://www.blackhat.com/about.html`.

[2] URL: `https://nginx.org/en/`.

[3] *Active Network Security Tool*. URL: `http://hping.org/`.

[4] Red Hat Ansible. URL: `https://www.ansible.com/`.

[5] *Bind9.net*. URL: `https://bind9.net/`.

[6] *Black Hat Europe 2018*. URL: https://www.blackhat.com/eu-18/briefings/schedule/#off-path-attacks-against-pki-12681.

[7] *CSE548 Final Project: Off-Path Attacks Against PKI*. Apr. 2021. URL: https://youtu.be/76vzbf7b0dc.

[8] Tianxiang Dai, Haya Shulman, and Michael Waidner. "Off-Path Attacks Against PKI". In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. CCS '18. Toronto, Canada: Association for Computing Machinery, 2018, pp. 2213–2215. ISBN: 9781450356930. DOI: 10.1145/3243734.3278516. URL: https://doi-org.ezproxy1.lib.asu.edu/10.1145/3243734.3278516.

[9] Austin Derbique. *aderbique/CSE548*. URL: https://github.com/aderbique/CSE548.

[10] *diagrams.net - free flowchart maker and diagrams online*. URL: https://app.diagrams.net/.

[11] Yossi Gilad and Amir Herzberg. "Off-Path Attacking the Web". In: *CoRR* abs/1204.6623 (2012). arXiv: 1204.6623. URL: http://arxiv.org/abs/1204.6623.

[12] Ross Jacobs. *Install*. URL: https://tshark.dev/setup/install/.

[13] Dan Kaminsky. *It's the end of the cache as we know it*. Apr. 2021.

[14] *Measures for Making DNS More Resilient against Forged Answers*. URL: https://tools.ietf.org/html/rfc5452.

[15] Inc. OpenSSL Foundation. *OpenSSL*. URL: https://www.openssl.org/.

[16] *Welcome to Python.org*. URL: https://www.python.org/.

[17] *Welcome to VirtualBox.org!* URL: https://www.virtualbox.org/.