DNS CACHE POISONING & DEFENCE VIA DNSSEC VALIDATION

Gautam Kumar Karamthot (G01481178), Bineeth Kumar Kollipalli (G01487072), Ankit Raut (G01476996),

Shalini Maknoor (G01503797) GROUP 12

CS 692, Network Security

George Mason University

Department of Computer Science

I. INTRODUCTION

1.1 Overview of the Domain Name System (DNS)

A DNS could be viewed as an internet phonebook because it provides a primary translation service. The service translates domain names to internet addresses. As an internet service, it plays an important role. It functions as an internet naming system with a hierarchical structure. It creates an opportunity for people to gain access to websites without requiring someone to remember internet addresses.

For example, if a user wants to visit a site and they enter www.example.com on their browser, maybe DNS will resolve www.example.com to a specific IP address like 10.0.1.20 so that it connects to the correct server. Otherwise, without DNS, people would need to enter all the IP addresses they want to visit, and that would be very impractical.

DNS Resolution Process

Four fundamental steps are involved in DNS resolution:

Query: It begins with a user device making a request for a domain name.

Resolution: To resolve names, the DNS resolver uses a hierarchical lookup process against authoritative servers, traversing from root servers to top-level domain (TLD) servers, and finally to authoritative nameservers.

Response: The correct IP address is returned to the user's device, often cached at multiple levels-that is, local resolver or ISP resolver-to improve performance.

Connection: The browser sends a connect request to the IP and fetches the content needed.

1.2. Security Vulnerabilities

Despite its crucial importance, the original DNS protocol was designed in the 1980s to be functional rather than secure. The protocol lacks native authentication mechanisms. Therefore, it is susceptible to a range of attack vectors that can degrade the integrity and confidence of DNS responses. This fundamental design weakness has resulted in several security incidents with high real-world impact.

Common DNS Attack Vectors

Below is a list summarizing core DNS security threats that organizations and end-users are exposed to:

Attack Type	Description
DNS Cache Poisoning	The attackers introduce false records into the resolver caches. As a result, user traffic is redirected to phishing sites.
Man-in-the-Middle	Attackers are able to intercept and change DNS queries between the user and server, enabling them to redirect traffic or steal sensitive information
DNS Spoofing	The forged DNS responses impersonate both real and

	phantom servers to make systems believe in fake IP addresses that reroute them to sham websites
DDoS Attacks	DNS amplification attacks overwhelm targets with massive query responses that can cause service disruptions and outages

1.3. Motivation for this System Implementation

Among these attack methods, one of the most serious and damaging threats would be the disruption caused by DNS cache poisoning. A DNS cache poisoning attack can be very damaging because it can affect thousands of users at once by poisoning a single DNS resolver cache. When a resolver cache is poisoned, it continues to supply malicious IP addresses to its clients until it times out.

The effects of successful attacks on DNS cache poisoning are more than just website redirects. A poisoned cache can be used for credential theft via phishing attacks, malware distribution, interception of SSL-secured communications, and even attacks against critical infrastructure.

To achieve these objectives, the project will propose a holistic system that illustrates attacks associated with DNS cache poisoning and develop a working defense mechanism based on DNSSEC verification. The system can be viewed as a learning platform for understanding attacks and an implementation platform for cryptographic security on the DNS.

2. BACKGROUND AND RELATED WORK

2.1. Historical Context of DNS Security

The original specifications of the DNS were created in RFC 1034 and RFC 1035 in 1987. During that era, it should be noted that the intervening Internet was largely being used for academic as well as research purposes. At that time, a certain level of trust existed within the Internet community, and security on this global network was not considered as a focus.

The most important pivot in DNS security awareness was in the year 2008 when security researcher Dan Kaminsky publicly revealed one critical vulnerability in DNS implementations. The Kaminsky Attack showed that DNS cache poisoning could be conducted much more efficiently than previously thought, virtually affecting all recursive DNS resolvers. This encouraged various emergency patches across the industry and accelerated DNSSEC development and deployment.

1.2. Real-World DNS Cache Poisoning Incidents

There have been a number of highly visible attacks that have shown the actual harm that can be caused as a result of DNS cache poisoning attacks. It is these attacks that have shaped our system designs:

2014 Turkey DNS Poisoning

A large DNS poisoning attack struck Turkey in 2014. It targeted country-level ISP resolvers at a time when there were political instabilities within the country. The attackers introduced fraudulent DNS answers into these resolvers, which caused all traffic meant for social networking sites to be redirected to government-managed sites instead. This event stood out as an important case because it highlighted the dangers posed by DNS poisoning attacks on a national level. It also highlighted its credibility as a tool for influencing millions of people on a national scale. The target platforms included Twitter and YouTube.

2010 Brazil ISP Poisoning

In 2010, there was a coordinated DNS cache poisoning attack on some Brazilian Internet Service Providers that targeted banking domains. The cyber-attack involved undermining the DNS cache and causing victims who were targeted and wanted to access legitimate banking websites to be redirected to very authentic phishing pages that were mimicking the original site. As a result, there were thousands of victims who suffered credential theft and financial fraud. The attack highlighted the financial motivation that drives attacks and clearly showed that scale can be achieved with DNS poisoning attacks targeted at

ISP-level resolvers, which could potentially affect all customers who were linked to the involved ISP.

2019 MyEtherWallet (MEW) Attack

MyEtherWallet 2019 Attacks Starting with 2019, attackers targeted cryptocurrency holders with a sophisticated MyEtherWallet, or MEW, attack. MEW is an Ethereum-based wallet service commonly among cryptocurrency holders. The attackers took advantage of BGP hijacking and exploited Route 53, a domain name service offered by cloud service provider Amazon. By so doing, they achieved DNS response poisoning and redirected MEW traffic to an attacker-controlled server. As a result, holders who tried accessing their cryptocurrency wallets were instead redirected to a phishing website that stole their private keys. Consequently, more than 150,000 dollars worth of cryptocurrency was stolen within a short period.

3. PROBLEM DEFINITION

3.1. Scope of the system

This project aims at developing a system that would demonstrate DNS cache poisoning attacks and then implements DNSSEC validation as a defensive countermeasure. The designed system should operate within a controlled lab environment, providing hands-on experience with attack and defence mechanisms.

B. Primary Objectives

The following are the main objectives of implementing this System:

- **Attack Simulation Environment:** To create a controlled lab environment, simulate DNS cache poisoning attacks with a vulnerable DNS resolver, an attacker system, and a victim client.
- **Cache Poisoning Implementation:** DNS cache poisoning is an attack mechanism where attackers forge DNS responses to inject malicious records in resolver caches.
- **DNSSEC Validation System:** Implement a DNSSEC-enabled DNS resolver to verify DNS record cryptographic signatures and avoid cache poisoning attacks.
- **Defence Demonstration:** To show that DNSSEC validation actually prevents cache poisoning attempts by rejecting unsigned or invalidly signed DNS responses.

4. SYSTEM REQUIREMENTS

The system should support and meet these functional and non-functional requirements:

Functional Requirements: Performed successful DNS cache poisoning attacks on the testing environment.

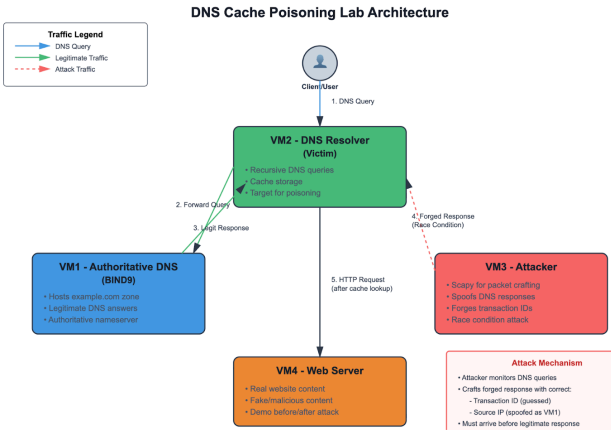
- Implement generation, signing, and verification of DNSSEC keys
- Prove successful defence against cache poisoning attacks with DNSSEC enabled
- Enable logging and monitoring of attack attempts and validation status

Non-Functional Requirements: Successfully execute DNS cache poisoning attacks in the test environment.

- Implement DNSSEC key generation, signing, and validation
- Demonstrate successful defence against cache poisoning with DNSSEC enabled
- Provide logging and monitoring of attack attempts and validation results

5. SYSTEM IMPLEMENTATION

5.1 Architecture



The implementation of the system brings together two distinct methods, namely virtual-machine (VM) implementation that enables an authentic network-level demonstration and user-interface (UI) implementation that enables visualization and conceptual knowledge. Specifically, VM implementation shows authentic methods for carrying out DNS cache poisoning attacks against a recursive resolver using forged responses, while UI implementation shows methods for defending against DNSSEC attacks at the application layer.

A system architecture would include the following necessary components:

Authoritative DNS Server: It contains the legitimate DNS zone and provides authentic responses to queries.

Recursive Resolver (Victim): It caches the DNS responses and acts as the main target for the poisoning.

Attacker System: It generates and sends fraudulent DNS responses with the intention of poisoning the resolver cache.

Monitoring System: It records and analyzes DNS traffic and uses it for verification and evidentiary purposes.

5.2. Environment Set Up

5.2.1. Attack Environment Set up

The attack environment uses four virtual machines deployed on the same subnet (192.168.230.0/24). IP addresses were verified using the ip a command.

VM	Role	IP Address
VM1	Authoritative DNS Server (BIND9)	192.168.230.130
VM2	Recursive Resolver / Victim	192.168.230.131
VM3	Attacker (Scapy + dig)	192.168.230.132
VM4	Packet Capture / Monitor	192.168.230.134

Operating System: Ubuntu Server (all VMs)

Network Configuration: Static IP addresses configured via Netplan, DHCP disabled

5.2.2 Defence Environment Setup

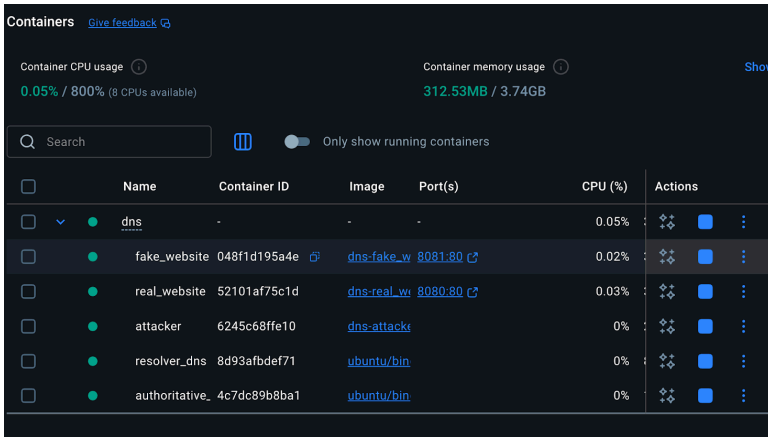
The UI-based defence implementation will be at an application level with Docker-based containers. It will serve as an environment that shows DNS cache poisoning and Defence for DNSSEC concepts.

Platform: Docker containerized application

DNS Simulation: Application-level cache simulating DNS resolution

DNSSEC Logic: Cryptographic validation using ZSK and KSK key hierarchy

Visualisation: Clear illustration of attack and defense mechanisms



5.3 DNS SERVER CONFIGURATION

5.3.1 Authoritative DNS Server Configuration (VM1)

The authoritative DNS server runs BIND9 and hosts the zone file located at /etc/bind/zones/example.com.zone:

```
$TTL 300
@   IN SOA ns1.example.com.
admin.example.com. (
        2024121102 ; Serial
        3600       ; Refresh
        1800       ; Retry
        604800     ; Expire
        300 )      ; Negative Cache
TTL
    IN NS ns1.example.com.
ns1 IN A 192.168.230.130
www IN A 10.0.1.20
```

The legitimate DNS answer for www.example.com is 10.0.1.20. A low TTL value (300 seconds) enables repeated poisoning attempts.

5.3.2 Recursive Resolver Configuration (VM2)

VM2 functions as the recursive caching resolver and is the primary attack target. Configuration (**/etc/netplan/00-installer-config.yaml**):

IP address: 192.168.230.131/24

Gateway: 192.168.230.2

DHCP: Disabled (static configuration)

DNS resolution is verified using: `dig @192.168.230.131 www.example.com +short`

5.4 Attack Implementation (DNS Cache Poisoning)

5.4.1 Attacker Configuration (VM3)

VM3 is configured as the attacker system with the following network parameters:

IP address: 192.168.230.132

Gateway: 192.168.230.2

DNS server: 192.168.230.130 (VM1)

5.4.2 Attack Tools

The attack uses the following tools:

Scapy: It is a Python-based library useful for working with network packets, used for creating forged DNS responses.

dig: A domain name lookup tool for initiating queries and examining results.

5.4.3 Forged Packet Construction

The forged DNS packets are created with the following properties:

- Source IP Spoofing: Packets will be sent with an indication that they originate from the authoritative server (192.168.230.130)
- TXID Guessing: Several transaction IDs are tried based on the resolver's query
- Port Guessing: Source ports are randomized

Malicious IP Injection: Forged responses contain fake IP address (10.0.100.100)

5.4.4 Attack Execution Steps

Step 1: Verify Legitimate Resolution

```
dig @192.168.230.130 www.example.com
+short
```

Expected output: 10.0.1.20

Step 2: Flush Resolver Cache (VM2)

```
sudo rndc flush
```

Step 3: Trigger Race Window (VM2)

```
dig @192.168.230.130 www.example.com
```

Step 4: Send Forged DNS Responses (VM4)

```
sudo python3 poison.py
```

The attacker floods forged DNS responses spoofing the authoritative server, injecting a fake IP address (10.0.100.100). The resolver accepts the first valid-looking response; if the forged response arrives first, the cache is poisoned.

Step 5: Verify Poisoning Result

```
dig @192.168.230.130 www.example.com
+short
```

If the result is 10.0.100.100, the cache poisoning attack is successful.

5.4.5 Packet Capture and Monitoring (VM4)

VM4 (IP: 192.168.230.134) is dedicated to packet capture using **tshark**:

sudo tshark -i ens33 -f "udp port 53"

It provides the evidence for DNS queries, legitimate responses, forged response floods, and the race-condition window..

5.5 Defence Implementation (DNSSEC Validation)

5.5.1 Key Generation (KSK and ZSK)

DNSSEC uses a two-key hierarchy for signing DNS zones:

Zone Signing Key (ZSK): The purpose of this key is to sign DNS resource records, including A, AAAA, MX, and so forth. This is a shorter key rotated more frequently.

Key Signing Key (KSK): A key used to sign the ZSK and establish the chain of trust. It should be longer than a ZSK and rotated less frequently.

For further security, keys can be generated on the authoritative server -that is, VM1 -securely for the zone signing operations.

5.5.2 Zone Signing Process

The process of zone signing introduces cryptographic signatures into all DNS records:

- Generate ZSK and KSK key pairs
- Sign all resource record sets with ZSK (creates RRSIG records)
- Sign the DNSKEY record set with KSK
- Publish DS record to parent zone (establishes chain of trust)
- Reload BIND9 to serve signed zone

5.5.3 Chain of Trust Configuration

The chain of trust makes it possible for cryptographic verification from the root zone down to specific domain records. At every hierarchical level, cryptographic verification is extended to the next level using delegation signer records. RRSIG records with cryptographic signatures are checked before a record can be cached.

5.5.4 Enabling Validation on Resolver

The activation of DNSSEC validation on the recursive resolver (VM2) can be done with these steps:

- Create trust anchors for the root zone
- Add support for enabling DNSSEC validation via BIND9
- Set up your validation policy so it will not accept unsigned/invalid responses

To implement these changes, it is necessary to start/stop the resolver service again

5.5.5 UI-Based DNSSEC Defense Logic

The user-interface driven implementation mimics the DNSSEC validation on the application layer. That is:

- Responses from DNS include RRSIG records mimicked
- The signatures are checked before records are included within the application cache.
- Unauthentic documents without genuine signatures are not acceptable.
- The validation process mirrors what occurs in actual DNSSEC operation at conceptual level.

5.6 Testing and Demonstration

5.6.1.VM Attack Result

Verification Using dig and Wireshark

The success or failure of attack is measured based on:

dig: sending a query to the resolver to check the cached IP address

tshark/Wireshark: Analysis of captured packets for traces of fraudulent response traffic

Before Poisoning

```
ankit@vm2:~$ dig @192.168.230.130 www.example.com

; <<>> DiG 9.18.39-0ubuntu0.22.04.2-Ubuntu <<>> @192.168.230.130 www
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 39018
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 955d1cc9ea4f30f101000000693b8c5d58430ea210402a0a (good)
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        300     IN      A       10.0.1.20

;; Query time: 203 msec
;; SERVER: 192.168.230.130#53(192.168.230.130) (UDP)
;; WHEN: Fri Dec 12 03:30:37 UTC 2025
;; MSG SIZE  rcvd: 88
```

Figure 1:Shows dns cache before attack in VM

After Poisoning

```
ankit@vm2:~$ dig @192.168.230.130 www.example.com

; <<>> DiG 9.18.39-0ubuntu0.22.04.2-Ubuntu <<>> @192.168.230.130
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26332
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL
;; WARNING: recursion requested but not available

;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        300     IN      A       10.0.100.100

;; Query time: 31 msec
;; SERVER: 192.168.230.130#53(192.168.230.130) (UDP)
;; WHEN: Fri Dec 12 03:34:04 UTC 2025
;; MSG SIZE  rcvd: 64
```

Figure 2:Shows dns cache after attack in VM

5.6.2. UI Implementation result

DNSSEC DISABLED

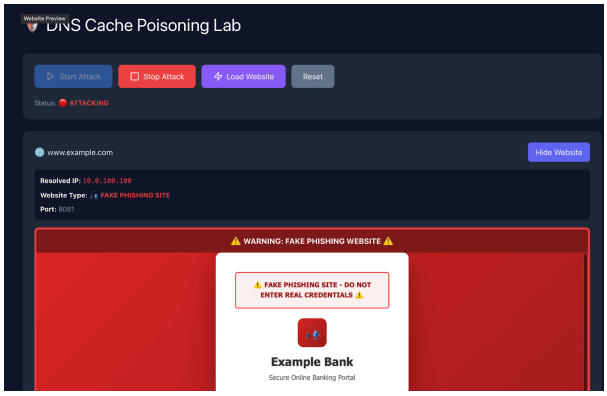
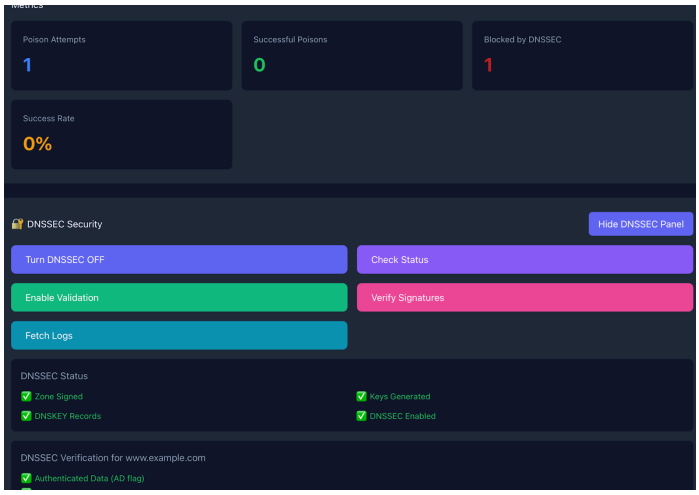


Figure 3: This shows poisoning successful in UI

DNSSEC ENABLED



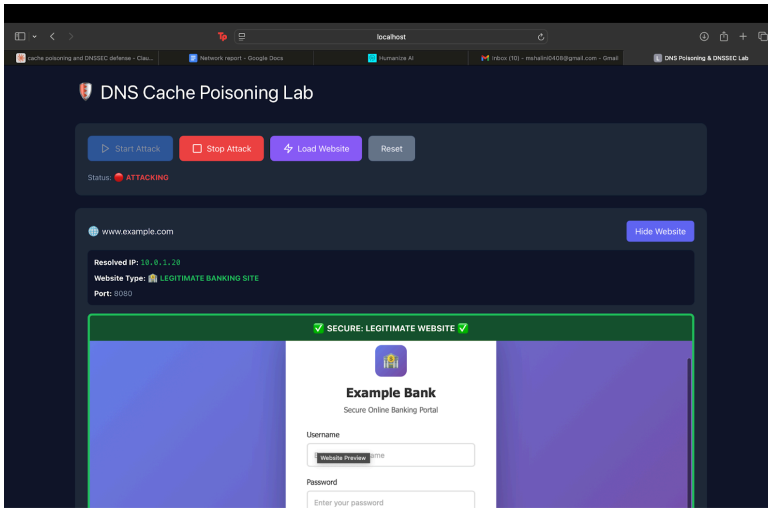


Figure 4: This shows attack is successfully prevented

6.LIMITATIONS

Controlled Environment Constraints: It is a controlled lab test. The timing availability for race condition attacks will affect success, and there are also additional ISP resolvers with source port randomization made tougher

Docker Networking Limitations: Because Docker networking does not work as it should within the UI environment, Doombricks doesn't support full IP address spoofing and some attack variables, like transaction numbers, are hard-coded. Nonetheless, the DNSSEC validation algorithm still maintains its cryptographic integrity and represents a correct implementation.

7. CONCLUSION

Our project aims to show DNS cache poisoning attacks and methods for defense using DNSSEC on a virtual and UI-based platforms.platform.The virtual implementation describes a closely realistic representation on a network level for a DNS cache poisoning attack with actual DNS packets and an Atlas response, while UI-based implementation helps improve understanding and represents attack and DNSSEC methods.

As evidenced by experimental data, it can be seen that DNSSEC works effectively against accepting forged information about the DNS. Without the verification process involved in DNSSEC, it can be realized that

cache poisoning attacks allow attackers to successfully insert malicious IP addresses into the resolver cache. As a result, all queries will be directed to attacker-controlled destinations. On the contrary, attacker-provided responses with no valid signatures are not accepted.

By addressing it from multiple facets, it presents a realistic and educational perspective on the seriousness of cache-poisoning attacks on the DNS and the efficacy of DNSSEC as an efficient safety measure. It can thus be said that it functions as an educational platform with respect to DNS security threats and the importance of cryptographic verification.

8. REFERENCES

[1] O. Alsad and Q. A. Al-Haija, "DNS cache poisoning attack detection: a systematic review," *7th IET Smart Cities Symposium (SCS 2023)*, Hybrid Conference, Bahrain, 2023, pp. 426-432, doi: 10.1049/icp.2024.0962.

[2] S. Abirami and R. Naresh, "DNS Enhancement with DNSSEC and DoT for Enhanced Online Security," *2024 2nd International Conference on Networking and Communications (ICNWC)*, Chennai, India, 2024, pp. 1-11, doi: 10.1109/ICNWC60771.2024.10537516.

[3] N. Alexiou, S. Basagiannis, P. Katsaros, T. Dashpande and S. A. Smolka, "Formal Analysis of the Kaminsky DNS Cache-Poisoning Attack Using Probabilistic Model Checking," *2010 IEEE 12th International Symposium on High Assurance Systems Engineering*, San Jose, CA, USA, 2010, pp. 94-103, doi: 10.1109/HASE.2010.25.

Project Note: *This project qualifies for the* ***System Building, Implementation, and Attack category****. The implementation demonstrates:*

1. ***How the exploit happens*** *– DNS cache poisoning is executed by flooding forged DNS responses to a recursive resolver, exploiting the race condition between legitimate and malicious replies*
2. ***How the exploit can be defeated*** *– DNSSEC validation is implemented to cryptographically sign DNS records, ensuring authenticity and integrity, thereby rejecting forged responses that lack valid signatures*