# Cyber Security TA Assignment

**Name :** Kiran K Patil 211070904

Mayuresh Murudkar 211070903

**Batch :** D

**Layer :** Application layer

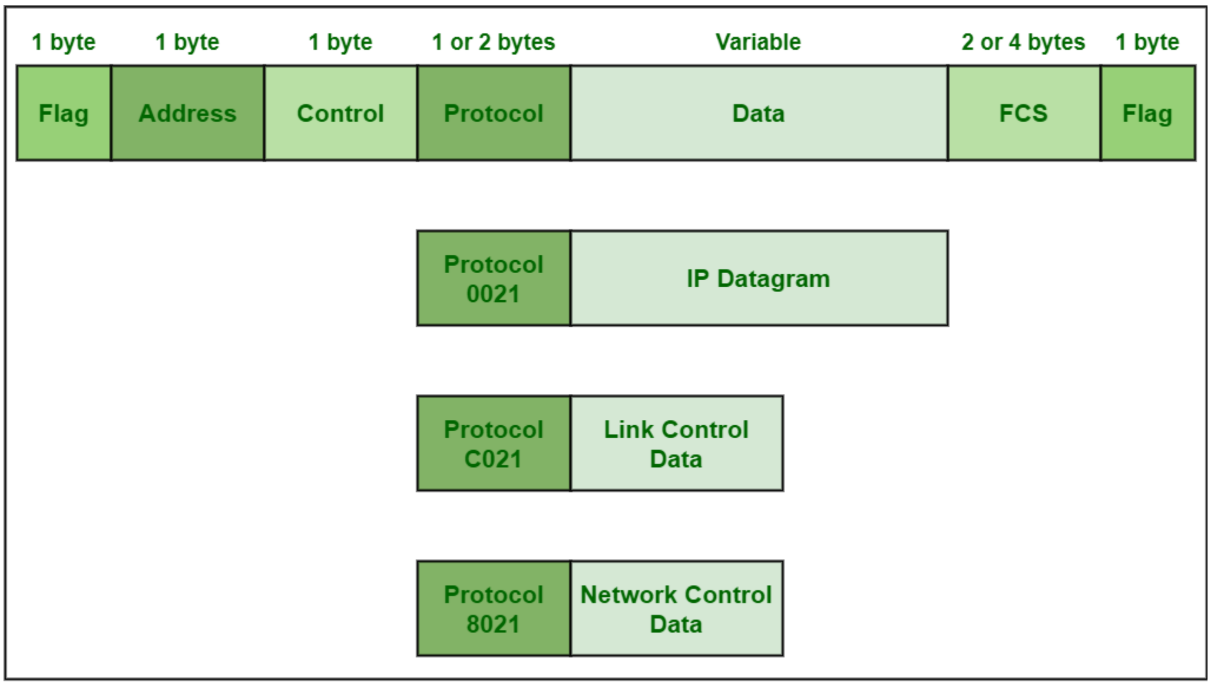
**Protocol :** DNS

## 1. Theory:

### PPP Protocol:

Point-to-Point Protocol (PPP) is a data link layer (layer 2) communication protocol between two routers directly without any host or any other networking in between. It can provide loop detection authentication, transmission encryption, and data compression. PPP is used over many types of physical networks, including serial cable, phone line, trunk line, cellular telephone, specialized radio links, ISDN, and fiber optic links such as SONET.

PPP Frame Format



PPP Frame Format

1. Flag field – PPP frame similar to HDLC frame, always begins and ends with standard HDLC flag. It always has a value of 1 byte i.e., 01111110 binary value.

2. Address field – Address field is basically broadcast address. In this, all 1's simply indicates that all of the stations are ready to accept frame. It has the value of 1 byte i.e., 11111111 binary values. PPP on the other hand, does not provide or assign individual station addresses.

3. Control field – This field basically uses format of U-frame i.e., Unnumbered frame in HDLC. In HDLC, control field is required for various purposes but in PPP, this field is set to 1 byte i.e., 00000011 binary value. This 1 byte is used for a connection-less data link.

4. Protocol field – This field basically identifies network protocol of the datagram. It usually identifies the kind of packet in the data field i.e., what exactly is being carried in data field. This field is of 1 or 2 bytes and helps in identifies the PDU (Protocol Data Unit) that is being encapsulated by PPP frame.

5. Data field – It usually contains the upper layer datagram. Network layer datagram is particularly encapsulated in this field for regular PPP data frames. Length of this field is not constant rather it varies.

6. FCS field – This field usually contains checksum simply for identification of errors. It can be either 16 bits or 32 bits in size. It is also calculated over address, control, protocol, and even information fields. Characters are added to frame for control and handling of errors.

**PPP architecture consists of three components:**

- **Encapsulation**: Encapsulates network layer protocol packets within PPP frames
- **Link Control Protocol (LCP):** Establishes, configures, and tests the data link connection
- **Network Control Protocols (NCPs):** Negotiate and manage options for the multiple network layer protocols used over the PPP connection

**PPP Operation**

PPP operates by encapsulating network layer protocol packets within PPP frames, transmitting them over the physical layer protocol. The PPP frames consist of a header and a trailer, providing information about the encapsulated packet and ensuring its reliable transmission. Additionally, the PPP header contains fields for the protocol type, address, control, and protocol identifier.

**PPP Features**

PPP offers several features, including:

- Authentication: Supports different authentication protocols, such as PAP and CHAP, to verify the identity of the peer
- Compression: Supports data compression protocols, such as Van Jacobson TCP/IP Compression (VJTCP) and Stac Lempel Ziv (LZS), to reduce bandwidth usage
- Error detection: Employs error detection mechanisms, such as Cyclic Redundancy Check (CRC), to identify and correct transmission errors
- Link quality monitoring: Monitors the quality of the physical link and can initiate negotiations to improve performance

**PPP Applications**

PPP is widely used in various applications, including:

- Dial-up connections: PPP is commonly used for dial-up connections to the Internet, providing a reliable and secure connection over phone lines
- Virtual private networks (VPNs): PPP is often used to establish VPNs, creating a secure tunnel over a public network
- Leased line connections: PPP can be used over leased lines to provide high-speed connections for businesses and organizations
- Remote access: PPP can be used to provide remote access to corporate networks for employees working from home or traveling

**PPP Advantages**

PPP offers several advantages, including:

- Simplicity: PPP is relatively simple to configure and implement
- Flexibility: PPP can be used over a variety of physical networks
- Security: PPP supports authentication and encryption mechanisms to protect data confidentiality and integrity
- Reliability: PPP provides reliable data transmission and error detection mechanisms

**PPP Limitations**

PPP also has some limitations, including:

- Limited bandwidth: PPP connections are typically limited in bandwidth compared to other technologies, such as Ethernet
- Complexity: PPP can become complex when used with multiple network layer protocols
- Troubleshooting: Troubleshooting PPP issues can be challenging due to its layered architecture

Overall, PPP is a versatile and widely used data link layer protocol that provides a reliable and secure connection for various applications. Its simplicity, flexibility, and security features make it a valuable tool for network administrators and users alike.

## DNS Protocol

The Domain Network System (DNS) protocol helps Internet users and network devices discover websites using human-readable hostnames, instead of numeric IP addresses. Domain Name System (DNS) is a hostname for IP address translation service. DNS is a distributed database implemented in a hierarchy of name servers. It is an application layer protocol for message exchange between clients and servers. It is required for the functioning of the Internet.

### Need of DNS

Every host is identified by the IP address but remembering numbers is very difficult for people also the IP addresses are not static therefore a mapping is required to change the domain name to the IP address. So DNS is used to convert the domain name of the websites to their numerical IP address.

### The DNS process, simplified, works as follows:

1. A browser, application or device called the DNS client, issues a DNS request or DNS address lookup, providing a hostname such as “example.com”.

2. The request is received by a DNS resolver, which is responsible for finding the correct IP address for that hostname. The DNS resolver looks for a DNS name server that holds the IP address for the hostname in the DNS request.

3. The resolver starts from the Internet’s root DNS server, moving down the hierarchy to Top Level Domain (TLD) DNS servers (“.com” in this case), down to the name server responsible for the specific domain “example.com”.

4. When the resolver reaches the authoritative DNS name server for “example.com”, it receives the IP address and other relevant details, and returns it to the DNS client. The DNS request is now resolved.

5. The DNS client device can connect to the server directly using the correct IP address.

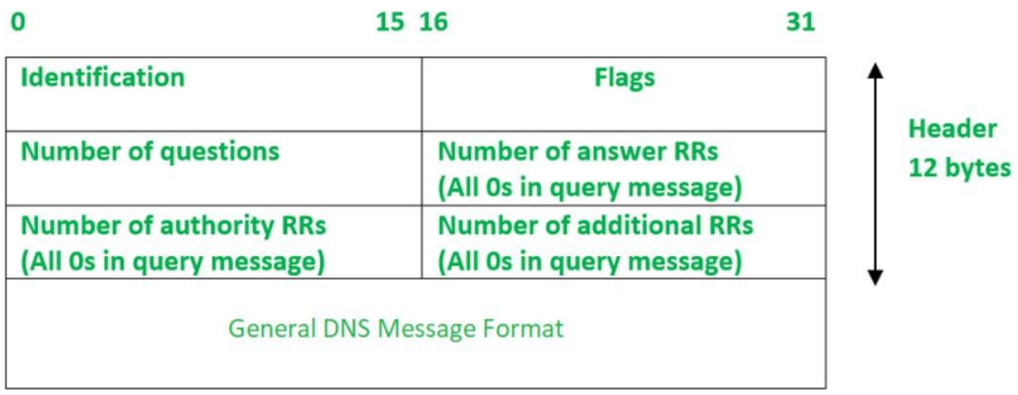
### Types of Domain

1. **Generic domains:** .com(commercial), .edu(educational), .mil(military), .org(nonprofit organization), .net(similar to commercial) all these are generic domains.

2. **Country domain:** .in (India) .us .uk

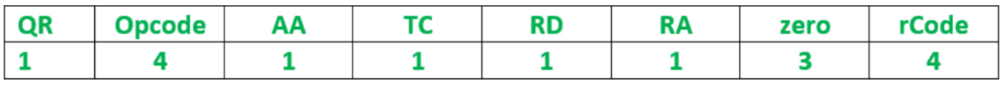
3. **Inverse domain:** if we want to know what is the domain name of the website. Ip to domain name mapping. So DNS can provide both the mapping for example to find the IP addresses of geeksforgeeks.org then we have to type

### DNS Packet format:


0
15 16
31
Identification
Flags
Number of questions
Number of answer RRs (All 0s in query message)
Number of authority RRs (All 0s in query message)
Number of additional RRs (All 0s in query message)
General DNS Message Format
Header 12 bytes

- **Identification**: The identification field is made up of 16 bits which are used to match the response with the request sent from the client-side. The matching is carried out by this field as the server copies the 16-bit value of identification in the response message so the client device can match the queries with the corresponding response received from the server-side.

- **Flags**: It is 16 bits and is divided into the following Fields : -



| QR | Opcode | AA | TC | RD | RA | zero | rCode |
|---|---|---|---|---|---|---|---|
| 1 | 4 | 1 | 1 | 1 | 1 | 3 | 4 |

Here is the description of each subfield of the Flags field:

- **QR (query/response):** It is a 1-bit subfield. If its value is 0, the message is of request type and if its value is 1, the message is of response type.

- **opcode**: It is a 4-bit subfield that defines the type of query carried by a message. This field value is repeated in the response. Following is the list of opcode values with a brief description:

  - If the value of the opcode subfield is 0 then it is a standard query.
  - The value 1 corresponds to an inverse of the query that implies finding the domain name from the IP Address.
  - The value 2 refers to the server status request. The value 3 specifies the status reserved and therefore not used.

- **AA**: It is an Authoritative Answer. It is a 1-bit subfield that specifies the server is authoritative if the value is 1 otherwise it is non-authoritative for a 0 value.

- **TC**: It is Truncation. This is a 1-bit subfield that specifies if the length of the message exceeds the allowed length of 512 bytes, the message is truncated when using UDP services.

- **RD**: It is Recursion Desired. It is a 1-bit subfield that specifies if the value is set to 1 in the query message then the server needs to answer the query recursively. Its value is copied to the response message.

- **RA**: It is Recursion Available. It is a 1-bit subfield that specifies the availability of recursive response if the value is set to 1 in the response message.

- **Zero**: It is a 3-bit reserved subfield set to 0.

- **rCode**: It stands for Response Code. It is a 4-bit subfield used to denote whether the query was answered successfully or not. If not answered successfully then the status of error is provided in the response. Following is the list of values with their error status –

  - The value 0 of rcode indicates no error.
  - A value of 1 indicates that there is a problem with the format specification.
  - Value 2 indicates server failure.
  - Value 3 refers to the Name Error that implies the name given by the query does not exist in the domain.
  - Value of 4 indicates that the request type is not supported by the server.
  - The value 5 refers to the nonexecution of queries by the server due to policy reasons.

- **Number of Questions-** It is a 16-bit field to specify the count of questions in the Question Section of the message. It is present in both query and response messages.

- **A number of answer RRs-** It is a 16-bit field that specifies the count of answer records in the Answer section of the message. This section has a value of 0 in query messages. The server answers the query received from the client. It is available only in response messages.

- **A number of authority RRs-** It is a 16-bit field that gives the count of the resource records in the Authoritative section of the message. This section has a value of 0 in query messages. It is available only in response messages. It gives information that comprises domain names about one or more authoritative servers.

**1. The DNS protocol is as follows: -**

63f1 8180 0001 0001 0000 0000 0377 7777 0667 6f6f 676c 6503 636f 6d00 0001 0001 c00c 0001 0001 0000 0027 0004 8efa c024

**Fields in DNS Datagram: -**

1. Transaction ID: 63f1 (16)
2. Flags: 8180 (16)
3. Questions: 0001 (16)
4. Answer RRs: 0001 (16)
5. Authority RRs: 0000 (16)
6. Additional RRs: 0000 (16)
7. Queries: 0377 7777 0667 6f6f 676c 6503 636f 6d00 0001 0001 (160) [Q Name + 32 bits]
8. Answers: c00c 0001 0001 0000 0027 0004 8efa c024 (128) [This length can vary]

**2. Questionnaire: -**

**Answer the following questions regarding the given DNS packet header: -**

**63f1 8180 0001 0001 0000 0000 0377 7777 0667 6f6f 676c 6503 636f 6d00 0001 0001 c00c 0001 0001 0000 0027 0004 8efa c024**

**1) What is the length of the DNS header?**

**Ans**: 12 Bytes (96 Bits)

**2) What is the Transaction ID of this DNS datagram?**

**Ans**: 63f1 (1st and 2nd Byte)

**3) Identify the type of given data packet as Request or Response.**

**Ans**: The Flag field of given datagram is (8180)16. In binary = $(1000\ 0001\ 1000\ 0000)_2$
The first bit represents type of message (0 -> Response; 1-> Request)
Hence, the given packet is Response packet.

**4) Is the server authoritative?**

**Ans**: No.

**Reason**: Flag = 1000 0001 1000 0000; 6th bit of the flag is 0 means server is not an authority for domain.

**5) Is the DNS Query corresponding to this response recursive?**

**Ans:** Yes.

**Reason**: The 8th bit of the Flag is 1. Hence, the Response is recursive in nature.

**6) Are there any errors?**

**Ans**: No. The last nibble (4 bits) of flags are 00002. Hence there is no error.

**7) What is the type of query and what it signifies?**

**Ans:** Last 4 bytes of Query field: (0001 0001)16. (0001 0001)16 => [Q Type(16 bit) + Q Class(16 bit)] 000116 = 110; Hence the query type is A. ‘A’ signifies Host address ipv4.

**8) What is TTL?**

**Ans:** 41st & 42nd by represents TTL. TTL = 002716 = 9910 (161 x 2 + 160 x 7)

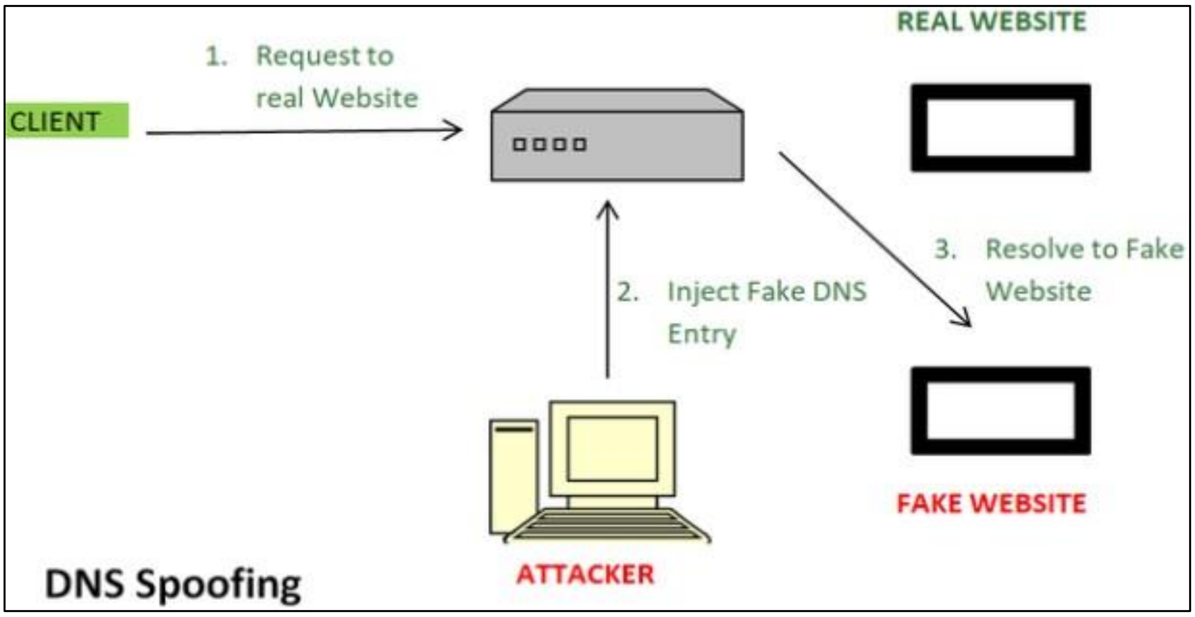
Hence, TTL is 39 seconds.

## 2. Attack

### 1) DNS Spoofing

**Attack Explanation with figure**

DNS spoofing, also known as DNS cache poisoning, is a form of cyber-attack that manipulates the Domain Name System (DNS) resolution process. DNS is a crucial component of the internet infrastructure responsible for translating human-readable domain names into IP addresses that computers use to identify each other on the network.



In a DNS spoofing attack, an attacker exploits vulnerabilities in the DNS system to provide false or malicious IP address information to a DNS resolver.

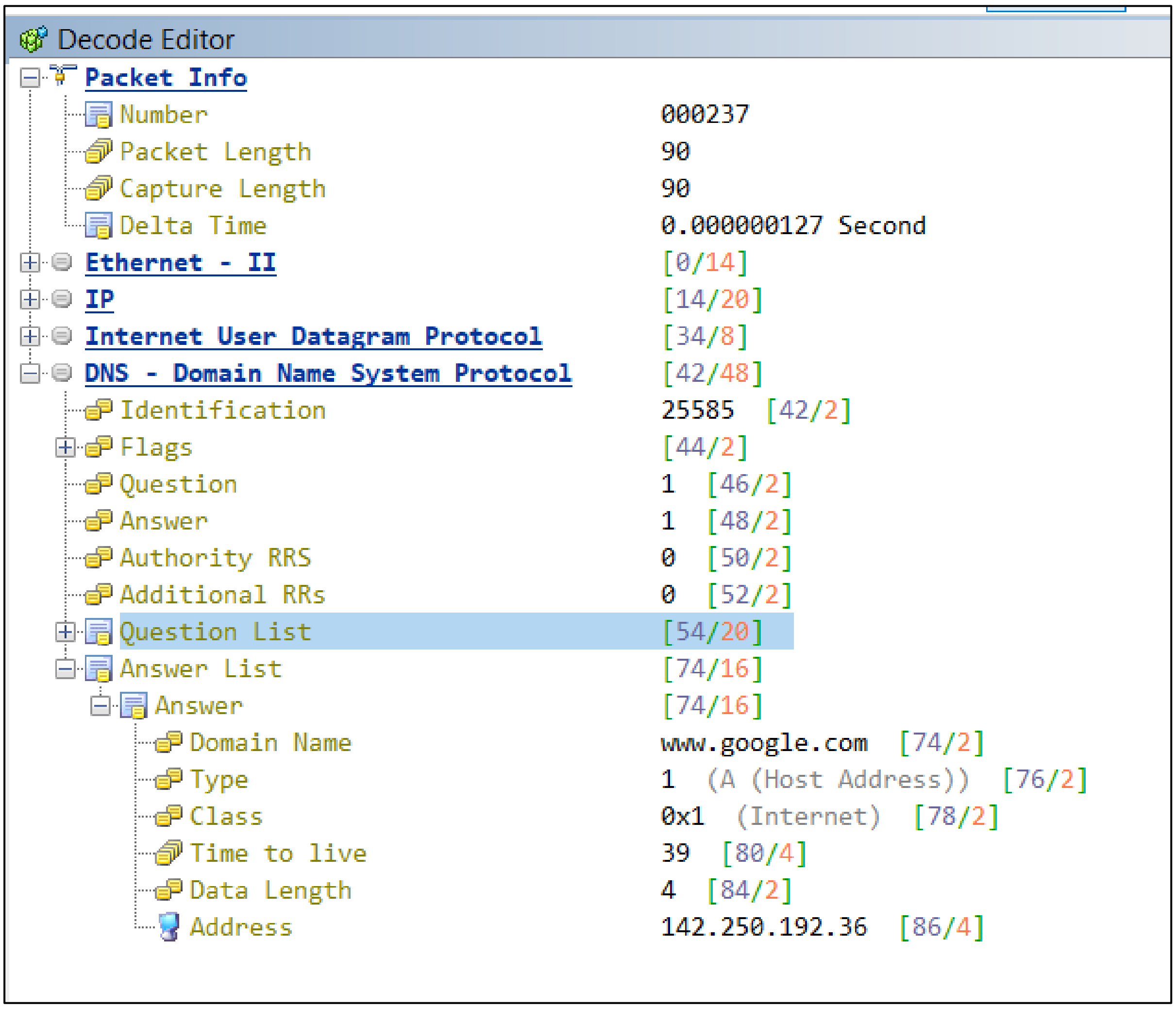
1. **DNS Resolution Process:** When a user enters a domain name in a web browser, the computer queries a DNS resolver to obtain the corresponding IP address.
2. **Request Intercept:** In a DNS spoofing attack, the attacker intercepts the DNS request and responds with a false IP address.
3. **Cache Poisoning:** The false information is then stored in the DNS resolver's cache. Subsequent requests for the same domain may be directed to the malicious IP address until the cache is refreshed.
4. **Redirected Traffic:** Users attempting to access the legitimate website are unknowingly redirected to a malicious site controlled by the attacker.

**Identify fields involved**

In DNS spoofing, the attacker typically sends a forged DNS response to a DNS query. The attacker may modify the IP address in the response to redirect the user to a malicious site. The modification usually occurs in the "Answer" section of the DNS response

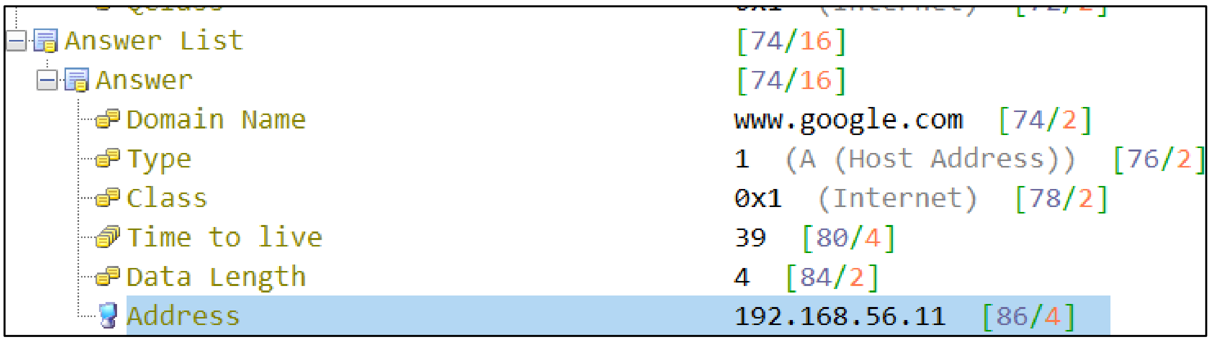
**Engage packet builder / Colasoft packet Builder**

1) Original DNS Packet

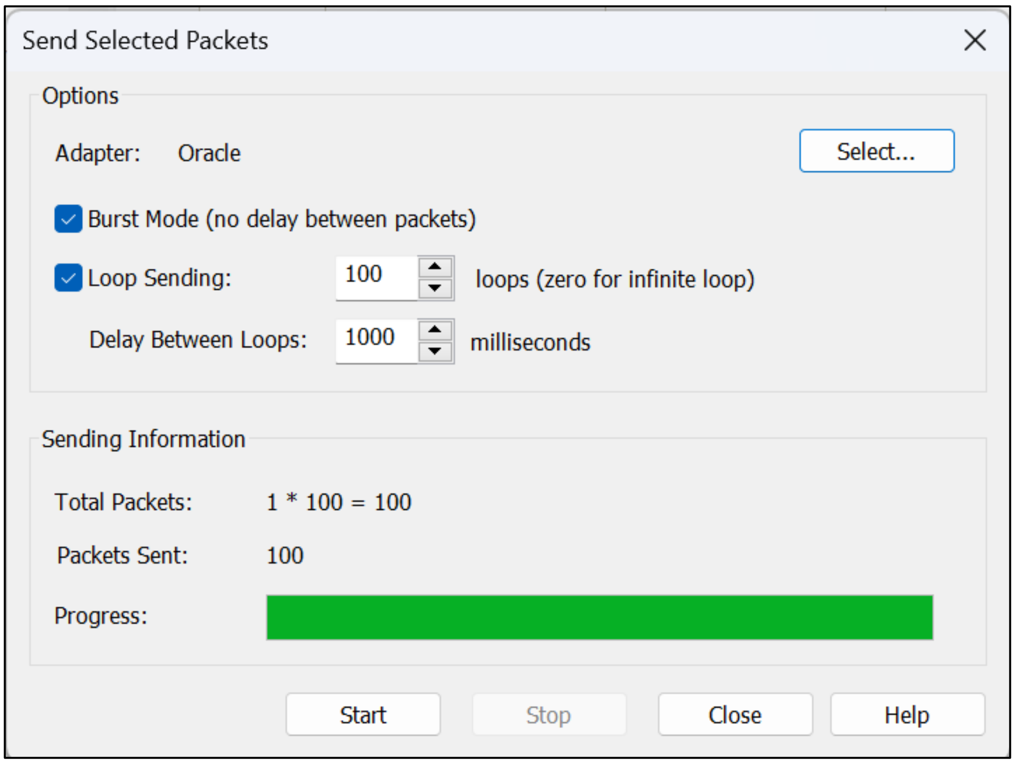


```
Decode Editor
Packet Info
    Number                                  000237
    Packet Length                           90
    Capture Length                          90
    Delta Time                              0.000000127 Second
Ethernet - II                               [0/14]
IP                                          [14/20]
Internet User Datagram Protocol             [34/8]
DNS - Domain Name System Protocol           [42/48]
    Identification                          25585  [42/2]
    Flags                                   [44/2]
    Question                                1  [46/2]
    Answer                                  1  [48/2]
    Authority RRS                           0  [50/2]
    Additional RRs                          0  [52/2]
    Question List                           [54/20]
    Answer List                             [74/16]
        Answer                              [74/16]
            Domain Name                     www.google.com  [74/2]
            Type                            1  (A (Host Address))  [76/2]
            Class                           0x1  (Internet)  [78/2]
            Time to live                    39  [80/4]
            Data Length                     4  [84/2]
            Address                         142.250.192.36  [86/4]
```

2) Modified Address Field in DNS Packet:



```
Answer List                                 [74/16]
    Answer                                  [74/16]
        Domain Name                         www.google.com  [74/2]
        Type                                1  (A (Host Address))  [76/2]
        Class                               0x1  (Internet)  [78/2]
        Time to live                        39  [80/4]
        Data Length                         4  [84/2]
        Address                             192.168.56.11  [86/4]
```



Send Selected Packets

Options

Adapter: Oracle

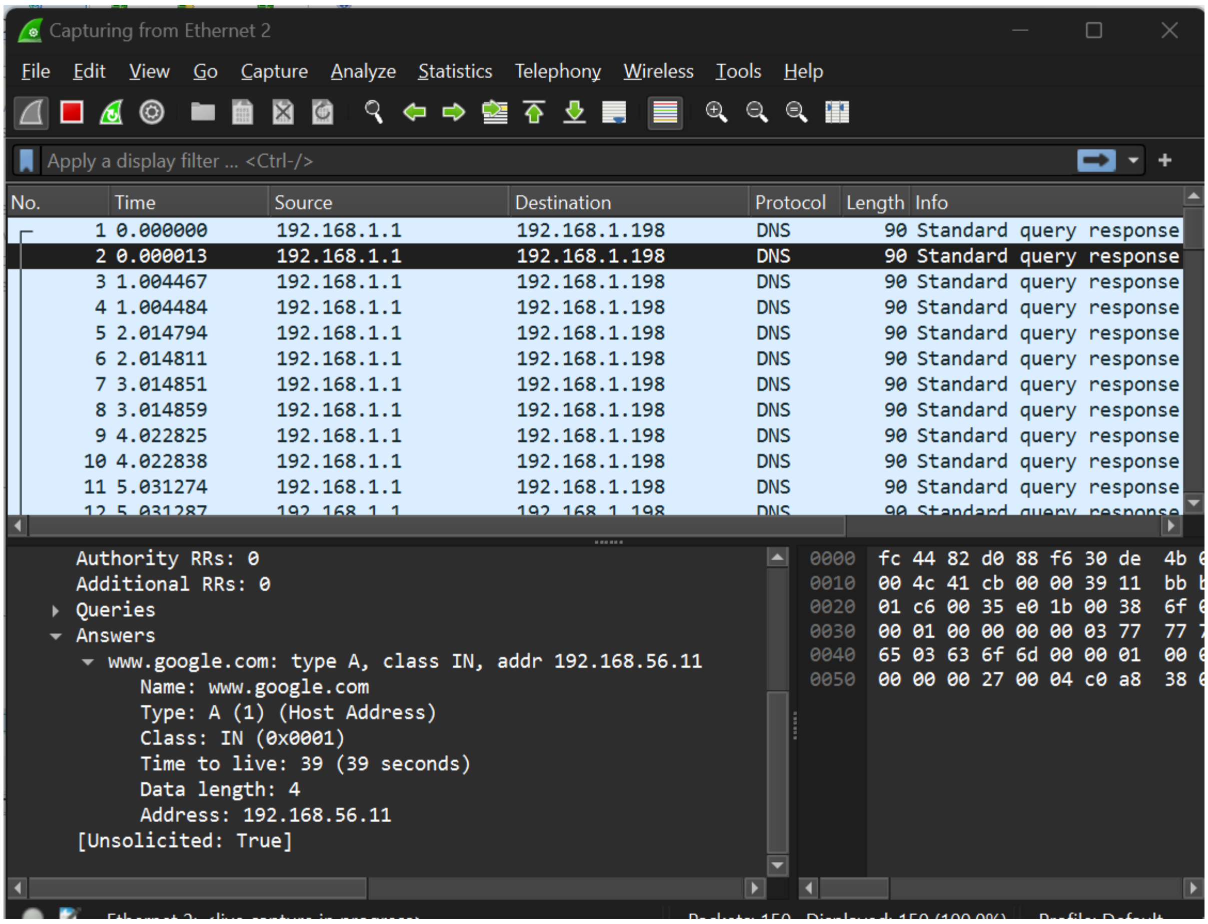
- [✓] Burst Mode (no delay between packets)
- [✓] Loop Sending: 100 loops (zero for infinite loop)

Delay Between Loops: 1000 milliseconds

Sending Information

| | |
|---|---|
| Total Packets: | 1 * 100 = 100 |
| Packets Sent: | 100 |
| Progress: | 🟩 |

**Tracing DNS packets in Wireshark :**



Capturing from Ethernet 2

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 1 | 0.000000 | 192.168.1.1 | 192.168.1.198 | DNS | 90 | Standard query response |
| 2 | 0.000013 | 192.168.1.1 | 192.168.1.198 | DNS | 90 | Standard query response |
| 3 | 1.004467 | 192.168.1.1 | 192.168.1.198 | DNS | 90 | Standard query response |
| 4 | 1.004484 | 192.168.1.1 | 192.168.1.198 | DNS | 90 | Standard query response |
| 5 | 2.014794 | 192.168.1.1 | 192.168.1.198 | DNS | 90 | Standard query response |
| 6 | 2.014811 | 192.168.1.1 | 192.168.1.198 | DNS | 90 | Standard query response |
| 7 | 3.014851 | 192.168.1.1 | 192.168.1.198 | DNS | 90 | Standard query response |
| 8 | 3.014859 | 192.168.1.1 | 192.168.1.198 | DNS | 90 | Standard query response |
| 9 | 4.022825 | 192.168.1.1 | 192.168.1.198 | DNS | 90 | Standard query response |
| 10 | 4.022838 | 192.168.1.1 | 192.168.1.198 | DNS | 90 | Standard query response |
| 11 | 5.031274 | 192.168.1.1 | 192.168.1.198 | DNS | 90 | Standard query response |
| 12 | 5.031287 | 192.168.1.1 | 192.168.1.198 | DNS | 90 | Standard query response |

```
    Authority RRs: 0
    Additional RRs: 0
  > Queries
  v Answers
    v www.google.com: type A, class IN, addr 192.168.56.11
          Name: www.google.com
          Type: A (1) (Host Address)
          Class: IN (0x0001)
          Time to live: 39 (39 seconds)
          Data length: 4
          Address: 192.168.56.11
    [Unsolicited: True]
```

```
0000  fc 44 82 d0 88 f6 30 de  4b
0010  00 4c 41 cb 00 00 39 11  bb
0020  01 c6 00 35 e0 1b 00 38  6f
0030  00 01 00 00 00 00 03 77  77
0040  65 03 63 6f 6d 00 00 01  00
0050  00 00 00 27 00 04 c0 a8  38
```

Ethernet 2: <live capture in progress>    Packets: 150  Displayed: 150 (100.0%)    Profile: Default

**Algorithm**

```
# Initialize data  structures

local_dns_cache = {}
threshold = 5

# Adjust as needed expected_transaction_ids = set() response_anomalies = 0

# Main loop for continuous monitoring

while True:

        # Capture DNS traffic
        dns_packet = capture_dns_packet()

        # Extract relevant information from DNS response
        transaction_id, queried_domain, response_ip = extract_dns_info(dns_packet)

        # Check if the response is authoritative
        is_authoritative = is_authoritative_response(dns_packet)

        # Maintain local
        DNS cacheif
        is_authoritative:
            local_dns_cache[queried_domain] = response_ip

        # Monitor response anomalies
        if transaction_id in expected_transaction_ids:
                    if queried_domain in local_dns_cache and response_ip !=
        local_dns_cache[queried_domain]:
                            response_anomalies += 1

        # Threshold check
        if response_anomalies >= threshold:
                    trigger_alert("Possible DNS spoofing attack detected")

        # Continue capturing and analyzing DNS traffic
```

**Defense Mechanisms:**

Defending against DNS spoofing involves implementing various security mechanisms and best practices to detect and prevent malicious manipulation of the Domain Name System. Here are several defence mechanisms:

1. DNS Security Extensions (DNSSEC): DNSSEC is a set of extensions to DNS that adds an additional layer of security by digitally signing DNS data. This helps ensure the integrity and authenticity of DNS responses, making it more difficult for attackers to manipulate or spoof DNS records.

2. Use of Validated DNS Resolvers: Organizations and individuals should use DNS resolvers that validate DNSSEC signatures. Validated DNS resolvers verify the authenticity of DNS data and can help protect against spoofed DNS responses.

3. Cache Poisoning Protection: Implement measures to protect against DNS cache poisoning, which is a common technique used in DNS spoofing attacks. Techniques may include implementing source port randomization, using random transaction IDs, and employing DNS response rate limiting.

4. DNS Filtering and Monitoring: Use DNS filtering solutions to block access to known malicious domains and monitor DNS traffic for unusual patterns. Anomalies, such as a high volume of requests for a specific domain or unexpected changes in DNS records, may indicate a DNS spoofing attempt.

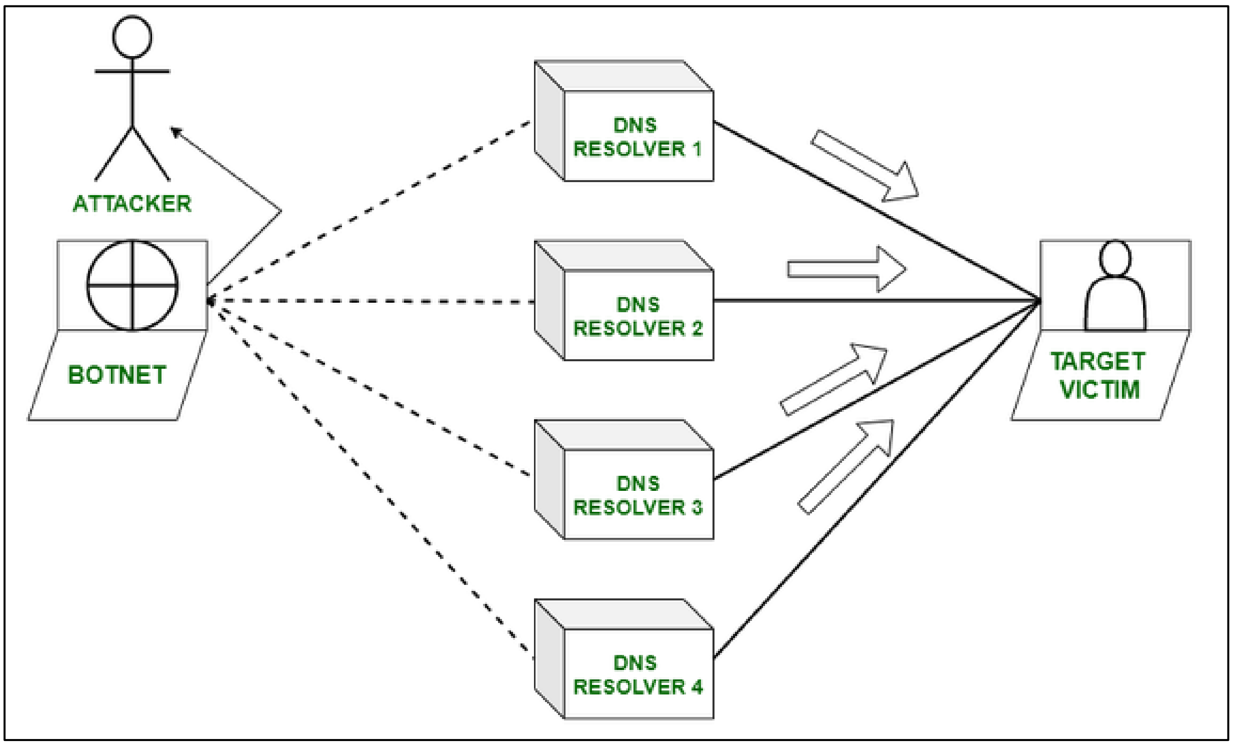
5. Firewall Rules and Access Controls: Configure firewalls to filter DNS traffic and block requests from untrusted or suspicious sources. Implement access controls to restrict access to DNS servers.

6. Use Encrypted DNS Protocols: Consider using encrypted DNS protocols such as DNS over HTTPS (DoH) or DNS over TLS (DoT) to protect the confidentiality of DNS queries and responses. This prevents eavesdropping and man-in-the-middle attacks.

By combining these defense mechanisms, organizations can significantly enhance their ability to detect and prevent DNS spoofing attacks, safeguarding the integrity of their DNS infrastructure and the security of their network communications.

**2) DNS Amplification**

In this attack, the attacker replicates the domains and sends a large number of DNS queries to the server, this results in server sending all the records of the responses of the queries to the attacker which then gains the access over the network. For example, if the attacker generates 10 MB of DNS queries, then the server sends back about 1 TB of responses to those queries. After that, the servers become so busy in handling the queries and traffic that they cannot request any other service from the legitimate users.



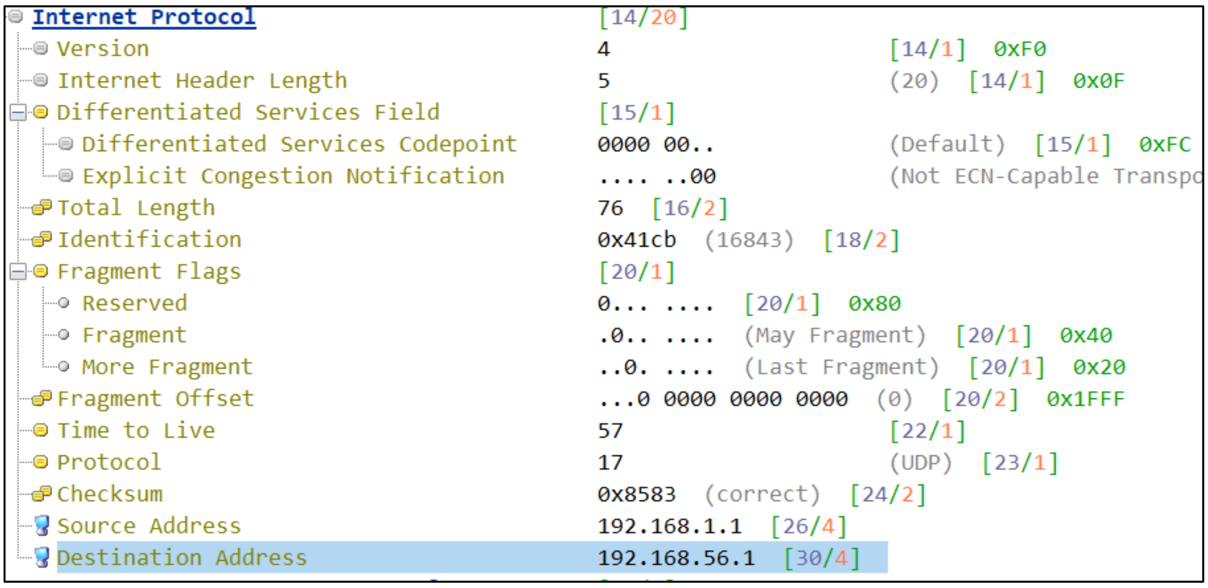
simplified explanation of how DNS amplification attacks work:

1. **Spoofed Requests:** The attacker sends DNS lookup requests to open DNS resolvers, but they spoof the source IP address to make it appear as if the requests are coming from the target system.
2. **Amplification:** The DNS resolvers respond to the requests with much larger responses than the original queries. This amplification effect is possible because DNS responses are typically much smaller than the requests. For example, a small DNS query can result in a much larger response, amplifying the data sent to the target.
3. **Traffic Flood:** The attacker floods the target server with the amplified DNS responses, overwhelming its capacity to handle incoming requests.

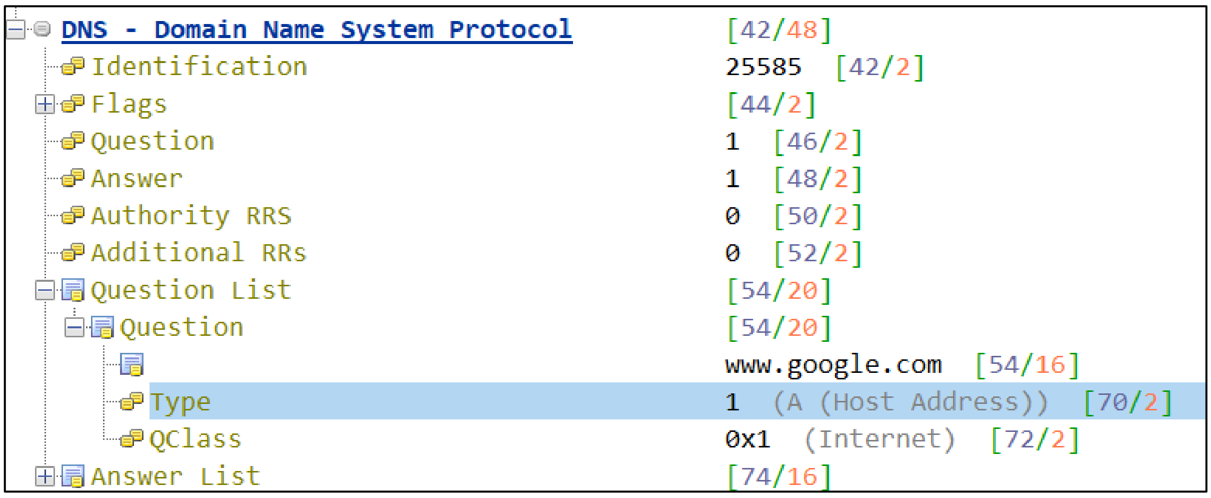
**Identify fields involved**

The modification involves forging the source IP address to make it appear as if the query is coming from the target (victim).

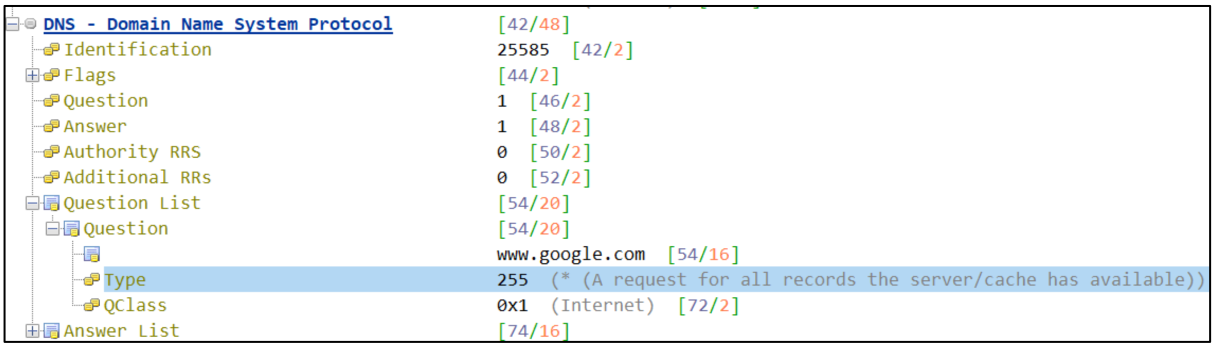
**Engage packet builder / Colasoft packet Builder**



```
Internet Protocol                              [14/20]
  Version                                      4                     [14/1]  0xF0
  Internet Header Length                       5                     (20)  [14/1]  0x0F
  Differentiated Services Field                [15/1]
    Differentiated Services Codepoint          0000 00..             (Default)  [15/1]  0xFC
    Explicit Congestion Notification           .... ..00             (Not ECN-Capable Transpo
  Total Length                                 76  [16/2]
  Identification                               0x41cb  (16843)  [18/2]
  Fragment Flags                               [20/1]
    Reserved                                   0... ....  [20/1]  0x80
    Fragment                                   .0.. ....  (May Fragment)  [20/1]  0x40
    More Fragment                              ..0. ....  (Last Fragment)  [20/1]  0x20
  Fragment Offset                              ...0 0000 0000 0000  (0)  [20/2]  0x1FFF
  Time to Live                                 57                    [22/1]
  Protocol                                     17                    (UDP)  [23/1]
  Checksum                                     0x8583  (correct)  [24/2]
  Source Address                               192.168.1.1  [26/4]
  Destination Address                          192.168.56.1  [30/4]
```

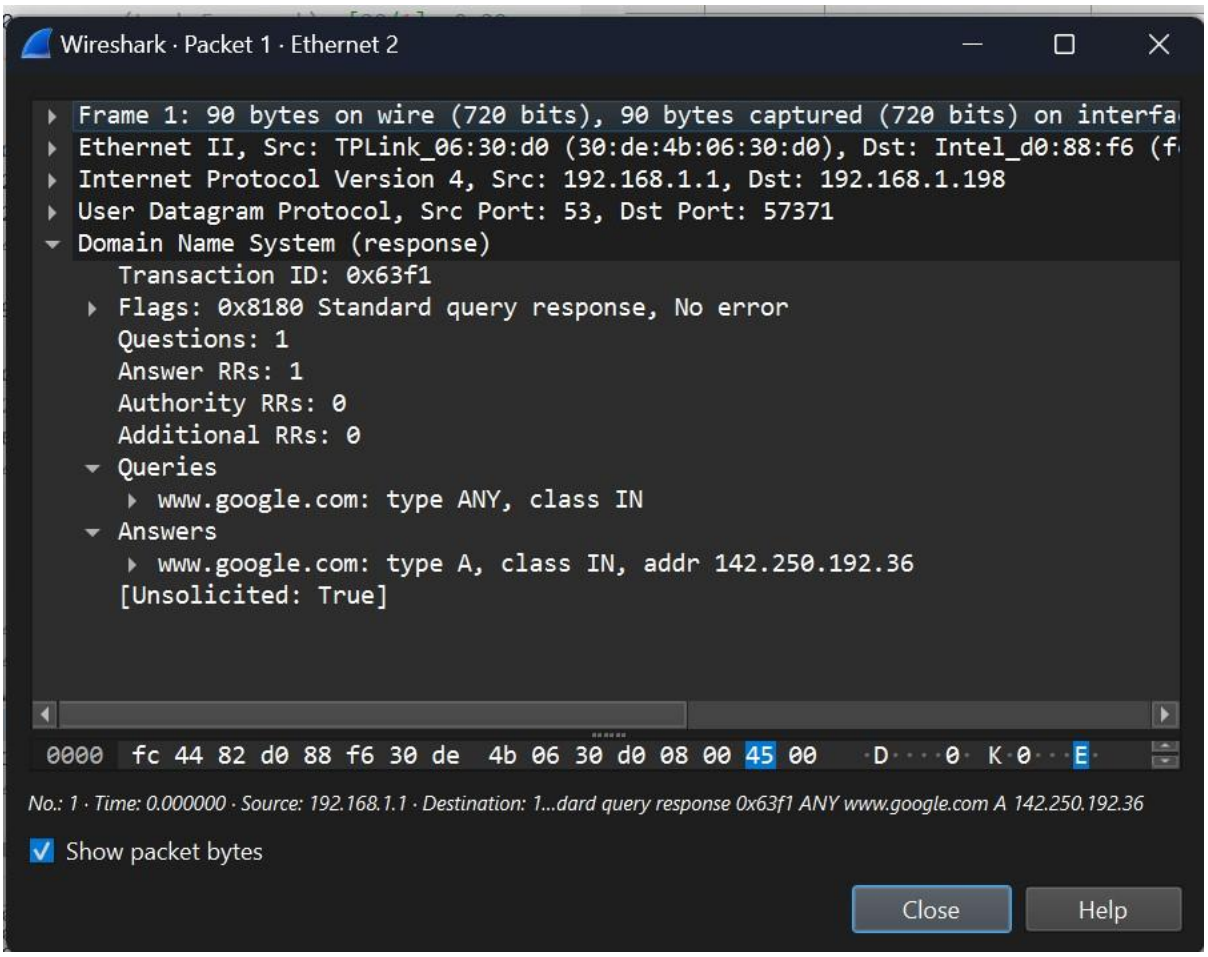


```
DNS - Domain Name System Protocol              [42/48]
  Identification                               25585  [42/2]
  Flags                                        [44/2]
  Question                                     1  [46/2]
  Answer                                       1  [48/2]
  Authority RRS                                0  [50/2]
  Additional RRs                               0  [52/2]
  Question List                                [54/20]
    Question                                   [54/20]
                                               www.google.com  [54/16]
      Type                                     1  (A (Host Address))  [70/2]
      QClass                                   0x1  (Internet)  [72/2]
  Answer List                                  [74/16]
```



```
DNS - Domain Name System Protocol              [42/48]
  Identification                               25585  [42/2]
  Flags                                        [44/2]
  Question                                     1  [46/2]
  Answer                                       1  [48/2]
  Authority RRS                                0  [50/2]
  Additional RRs                               0  [52/2]
  Question List                                [54/20]
    Question                                   [54/20]
                                               www.google.com  [54/16]
      Type                                     255  (* (A request for all records the server/cache has available))
      QClass                                   0x1  (Internet)  [72/2]
  Answer List                                  [74/16]
```

**Packet tracing in Wireshark**



```
▸ Frame 1: 90 bytes on wire (720 bits), 90 bytes captured (720 bits) on interfa
▸ Ethernet II, Src: TPLink_06:30:d0 (30:de:4b:06:30:d0), Dst: Intel_d0:88:f6 (f
▸ Internet Protocol Version 4, Src: 192.168.1.1, Dst: 192.168.1.198
▸ User Datagram Protocol, Src Port: 53, Dst Port: 57371
▾ Domain Name System (response)
    Transaction ID: 0x63f1
  ▸ Flags: 0x8180 Standard query response, No error
    Questions: 1
    Answer RRs: 1
    Authority RRs: 0
    Additional RRs: 0
  ▾ Queries
    ▸ www.google.com: type ANY, class IN
  ▾ Answers
    ▸ www.google.com: type A, class IN, addr 142.250.192.36
    [Unsolicited: True]

0000  fc 44 82 d0 88 f6 30 de  4b 06 30 d0 08 00 45 00   ·D····0· K·0···E·

No.: 1 · Time: 0.000000 · Source: 192.168.1.1 · Destination: 1...dard query response 0x63f1 ANY www.google.com A 142.250.192.36
```

- **Algorithm**

```
# Initialize data structures
dns_amplification_threshold = 10 #
Adjust as needed
dns_amplification_counter = 0

while True:
    # Capture DNS traffic
    dns_packet = capture_dns_packet()

    # Extract relevant information from DNS response
    transaction_id, queried_domain, response_ip = extract_dns_info(dns_packet)
```

```
# Check for DNS
amplificationif
len(queried_dom
ain) > 1:
    dns_amplification_counter += 1
# DNS amplification threshold check
if dns_amplification_counter >=
    dns_amplification_threshold:
    trigger_alert("Possible DNS amplification
    attack detected")
# Continue capturing and analyzing DNS traffic
```

**Defense mechanism:**

To mitigate the risk of DNS amplification attacks, it's important to:

1. Configure DNS Servers Properly: Administrators should configure DNS servers to prevent them from being used as open resolvers and implement measures to restrict recursive queries.

2. Rate Limiting: Implement rate limiting on DNS servers to restrict the number of responses sent to a particular source within a specified timeframe.

3. Monitoring and Filtering: Use traffic monitoring and filtering solutions to identify and filter out malicious DNS traffic.

4. Firewall Rules: Employ firewall rules to block traffic from known malicious IP addresses or limit access to DNS servers.

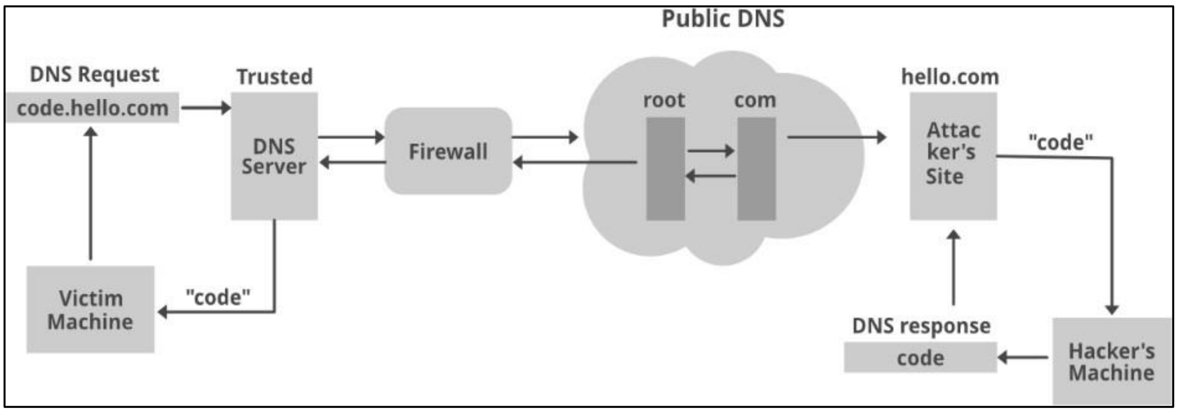
5. DNS Response Validation: Implement DNS Security Extensions (DNSSEC) to add an additional layer of security by digitally signing DNS data, helping to prevent tampering and data manipulation.

These measures can help organizations defend against DNS amplification attacks and ensure the integrity and availability of their DNS infrastructure.

## 3) DNS Tunneling

### Attack Explanation with figure

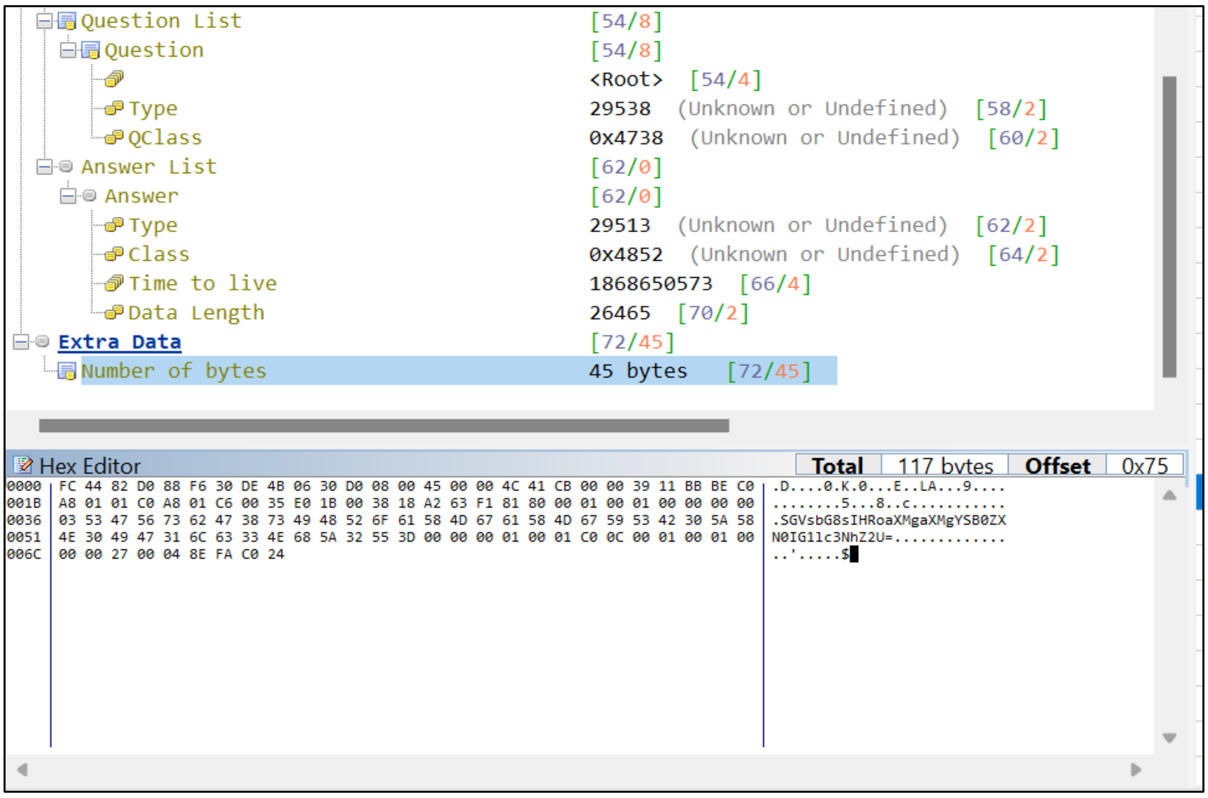
DNS Tunneling is a strategy for a digital exploit that encodes the information of different programs or protocols in DNS inquiries and responses. DNS tunnelling frequently incorporates information payloads that can be added to an exploited domain name server and used to control a distant system and applications.



### Identify fields involved

DNS tunneling involves encoding non-DNS data within DNS queries and responses. Attackers manipulate various fields, including the "Query" and "Answer" sections, to embed data covertly. This can include payload data and control information, allowing unauthorized communication through DNS.

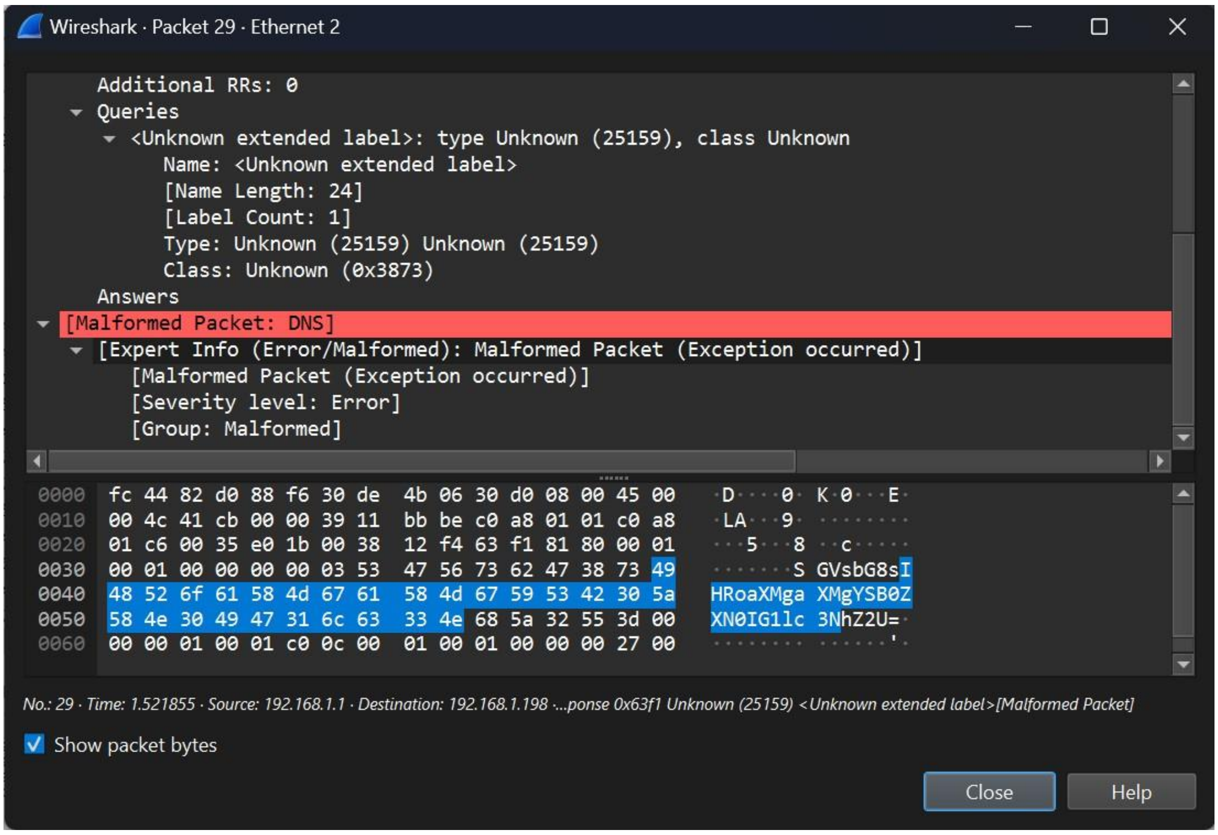
### Engage packet builder / Colasoft packet Builder



```
Question List                 [54/8]
  Question                    [54/8]
                              <Root>  [54/4]
    Type                      29538  (Unknown or Undefined)  [58/2]
    QClass                    0x4738  (Unknown or Undefined)  [60/2]
Answer List                   [62/0]
  Answer                      [62/0]
    Type                      29513  (Unknown or Undefined)  [62/2]
    Class                     0x4852  (Unknown or Undefined)  [64/2]
    Time to live              1868650573  [66/4]
    Data Length               26465  [70/2]
Extra Data                    [72/45]
  Number of bytes             45 bytes  [72/45]
```

```
Hex Editor                                      Total  117 bytes  Offset  0x75
0000 | FC 44 82 D0 88 F6 30 DE 4B 06 30 D0 08 00 45 00 00 4C 41 CB 00 00 39 11 BB BE C0 | .D....0.K.0...E..LA...9....
001B | A8 01 01 C0 A8 01 C6 00 35 E0 1B 00 38 18 A2 63 F1 81 80 00 01 00 01 00 00 00 00 | ........5...8..c...........
0036 | 03 53 47 56 73 62 47 38 73 49 48 52 6F 61 58 4D 67 61 58 4D 67 59 53 42 30 5A 58 | .SGVsbG8sIHRoaXMgaXMgYSB0ZX
0051 | 4E 30 49 47 31 6C 63 33 4E 68 5A 32 55 3D 00 00 00 01 00 01 C0 0C 00 01 00 01 00 | N0IG1lc3NhZ2U=.............
006C | 00 00 27 00 04 8E FA C0 24                                                       | ..'.....$
```

Malformed packet found in wireshark

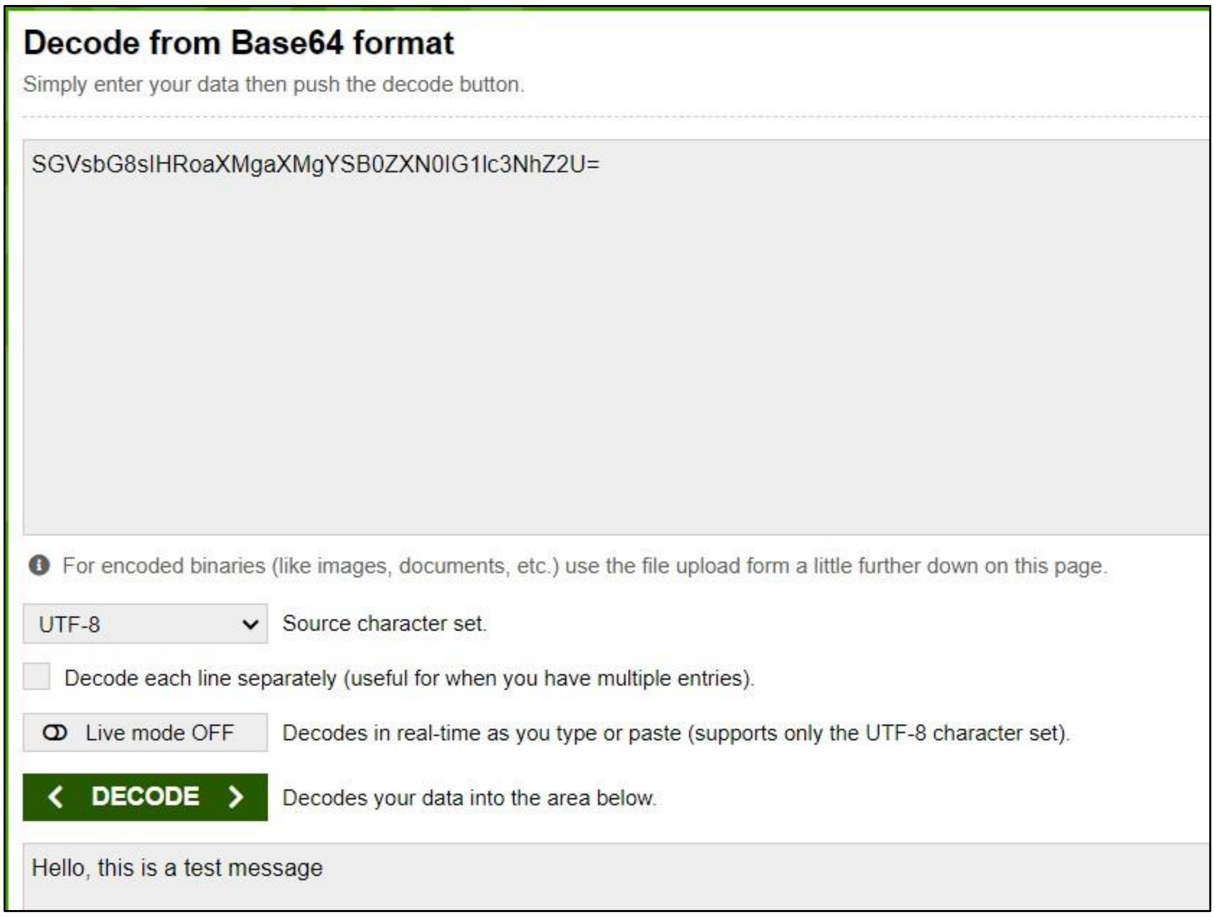

Wireshark · Packet 29 · Ethernet 2

```
    Additional RRs: 0
  ▾ Queries
    ▾ <Unknown extended label>: type Unknown (25159), class Unknown
        Name: <Unknown extended label>
        [Name Length: 24]
        [Label Count: 1]
        Type: Unknown (25159) Unknown (25159)
        Class: Unknown (0x3873)
    Answers
▾ [Malformed Packet: DNS]
  ▾ [Expert Info (Error/Malformed): Malformed Packet (Exception occurred)]
      [Malformed Packet (Exception occurred)]
      [Severity level: Error]
      [Group: Malformed]
```

```
0000  fc 44 82 d0 88 f6 30 de  4b 06 30 d0 08 00 45 00   ·D····0· K·0···E·
0010  00 4c 41 cb 00 00 39 11  bb be c0 a8 01 01 c0 a8   ·LA···9· ········
0020  01 c6 00 35 e0 1b 00 38  12 f4 63 f1 81 80 00 01   ···5···8 ··c·····
0030  00 01 00 00 00 00 03 53  47 56 73 62 47 38 73 49   ·······S GVsbG8sI
0040  48 52 6f 61 58 4d 67 61  58 4d 67 59 53 42 30 5a   HRoaXMga XMgYSB0Z
0050  58 4e 30 49 47 31 6c 63  33 4e 68 5a 32 55 3d 00   XN0IG1lc 3NhZ2U=·
0060  00 00 01 00 01 c0 0c 00  01 00 01 00 00 00 27 00   ········ ······'·
```

*No.: 29 · Time: 1.521855 · Source: 192.168.1.1 · Destination: 192.168.1.198 ·...ponse 0x63f1 Unknown (25159) <Unknown extended label>[Malformed Packet]*

☑ Show packet bytes


**Decode from Base64 format**

Simply enter your data then push the decode button.

SGVsbG8sIHRoaXMgaXMgYSB0ZXN0IG1lc3NhZ2U=

For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8 — Source character set.

☐ Decode each line separately (useful for when you have multiple entries).

Live mode OFF — Decodes in real-time as you type or paste (supports only the UTF-8 character set).

DECODE — Decodes your data into the area below.

Hello, this is a test message

- **Algorithm**

```
# Initialize data structures
dns_tunneling_threshold = 5 #
Adjust as needed
dns_tunneling_counter = 0

while True:
   # Capture DNS traffic
   dns_packet = capture_dns_packet()

   # Extract relevant information from DNS response
   transaction_id, queried_domain, response_ip = extract_dns_info(dns_packet)

   # Check for DNS tunneling
   if
      is_unusual_query_pattern(que
      ried_domain):
      dns_tunneling_counter += 1
   # DNS tunneling threshold check
   if dns_tunneling_counter >=
      dns_tunneling_threshold:
      trigger_alert("Possible DNS tunneling
      detected")

   # Continue capturing and analyzing DNS traffic
```

**Defense mechanism:**

To detect and prevent DNS tunneling, organizations can implement the following measures:

1. Deep Packet Inspection: Analyze network traffic for anomalies and inspect DNS packets for unusual characteristics that may indicate tunneling.

2. Signature-Based Detection: Use signatures to identify known DNS tunneling tools and patterns.

3. DNS Logging and Monitoring: Log and monitor DNS requests for unusual patterns, high-frequency queries, or unexpected data in DNS payloads.

4. Rate Limiting: Implement rate limiting on DNS requests to prevent the excessive use of DNS for data transmission.

5. DNS Sinkholing: Redirect suspicious DNS traffic to a sinkhole, preventing communication with malicious domains.

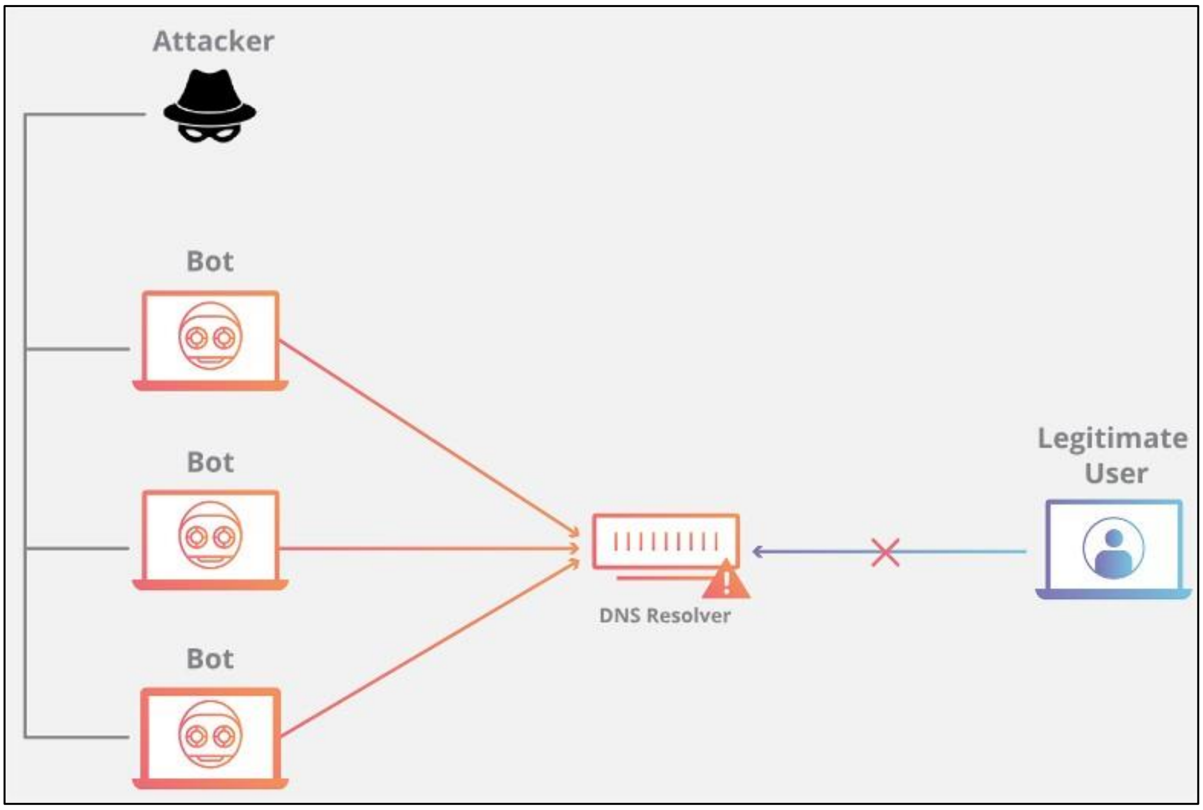
6. Security Policies: Enforce strict security policies regarding DNS traffic and consider blocking or closely monitoring DNS traffic that deviates from normal patterns.

It's important for organizations to be aware of the potential risks associated with DNS tunneling and to implement appropriate security measures to detect and mitigate this type of covert communication.

**4) DDoS Attack**

**Attack Explanation with figure**

A DNS flood is a type of distributed denial-of-service attack (DDoS) where an attacker floods a particular domain’s DNS servers in an attempt to disrupt DNS resolution for that domain. If a user is unable to find the phonebook, it cannot lookup the address in order to make the call for a particular resource. By disrupting DNS resolution, a DNS flood attack will compromise a website, API, or web application's ability respond to legitimate traffic.

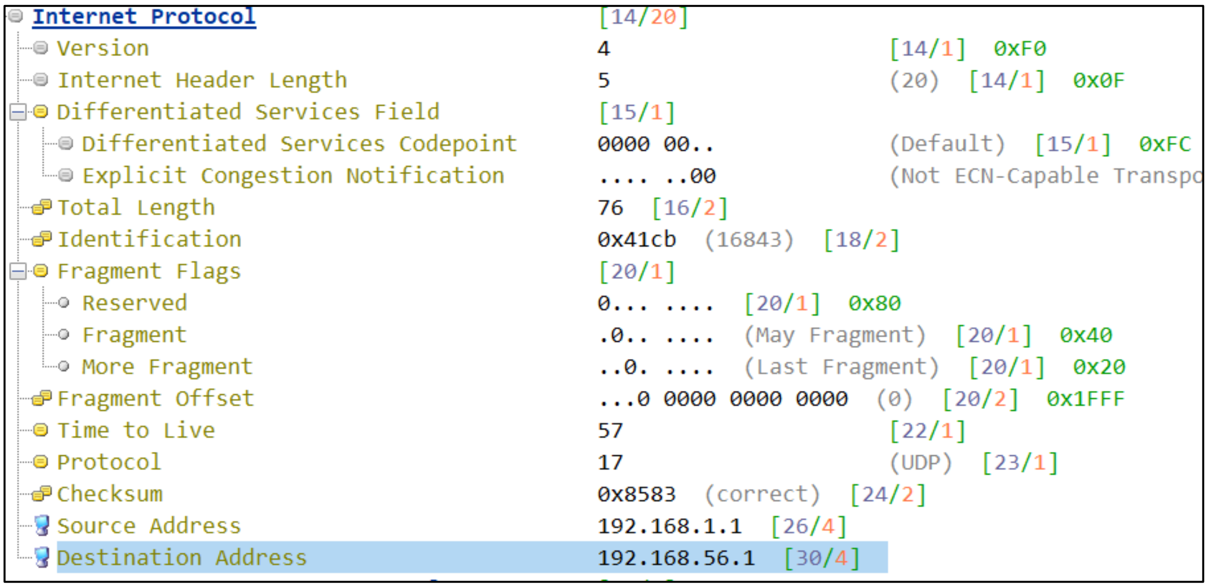


A DNS Distributed Denial of Service (DDoS) attack is a type of cyber attack that aims to disrupt the normal functioning of a targeted domain's DNS infrastructure. In a DNS DDoS attack, multiple compromised computers (often part of a botnet) are used to flood the target's DNS servers with a large volume of traffic, overwhelming their capacity to respond to legitimate DNS queries.

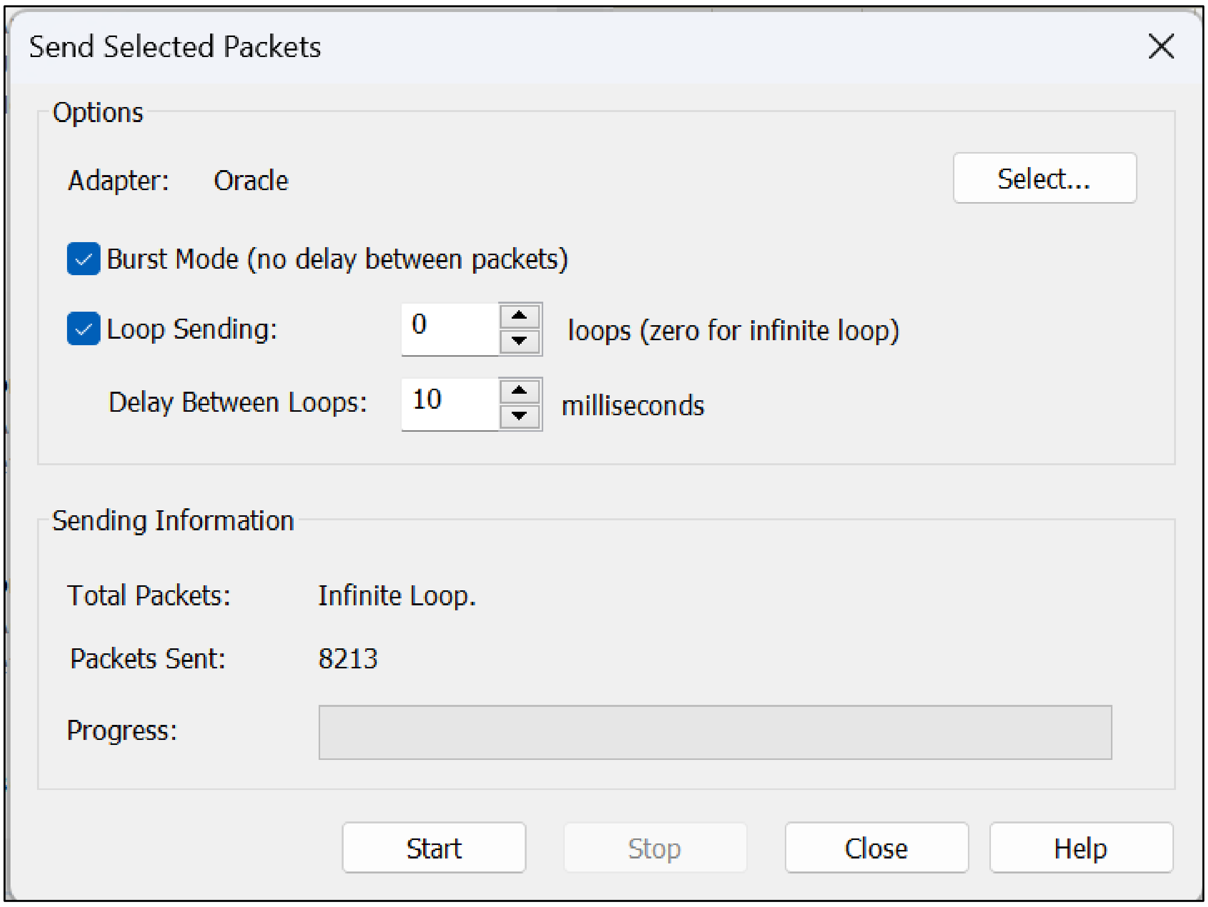
**Identify fields involved**

The modification may involve forging the source IP address in these requests to make it difficult to trace the origin of the attack. Additionally, attackers may manipulate various fields in DNS packets to increase the load on DNS servers and exhaust their resources.

**Engage packet builder / colasoft packet builder**



```
Internet Protocol                                   [14/20]
  Version                                           4                      [14/1]  0xF0
  Internet Header Length                            5                      (20)  [14/1]  0x0F
  Differentiated Services Field                     [15/1]
    Differentiated Services Codepoint               0000 00..              (Default)  [15/1]  0xFC
    Explicit Congestion Notification                .... ..00              (Not ECN-Capable Transpo
  Total Length                                      76  [16/2]
  Identification                                    0x41cb  (16843)  [18/2]
  Fragment Flags                                    [20/1]
    Reserved                                        0... ....  [20/1]  0x80
    Fragment                                        .0.. ....  (May Fragment)  [20/1]  0x40
    More Fragment                                   ..0. ....  (Last Fragment)  [20/1]  0x20
  Fragment Offset                                   ...0 0000 0000 0000  (0)  [20/2]  0x1FFF
  Time to Live                                      57                     [22/1]
  Protocol                                          17                     (UDP)  [23/1]
  Checksum                                          0x8583  (correct)  [24/2]
  Source Address                                    192.168.1.1  [26/4]
  Destination Address                               192.168.56.1  [30/4]
```



Send Selected Packets

Options

Adapter: Oracle

Burst Mode (no delay between packets) ✓

Loop Sending: ✓ 0 loops (zero for infinite loop)

Delay Between Loops: 10 milliseconds

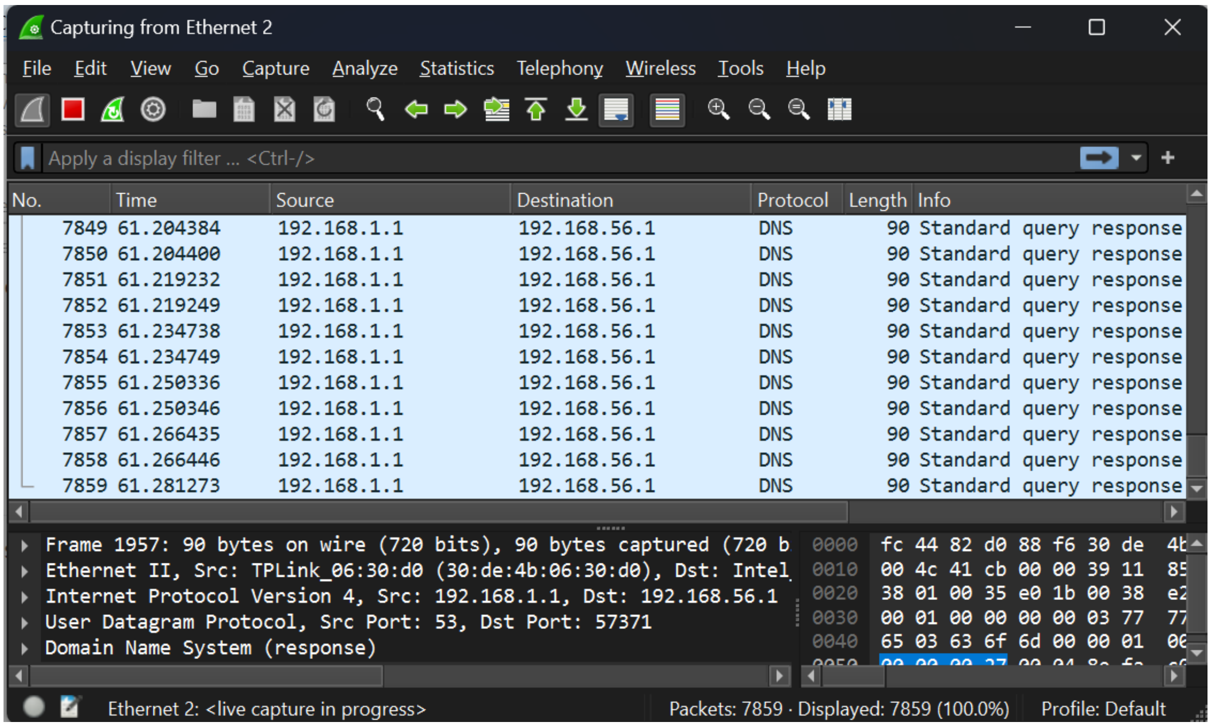
Sending Information

Total Packets: Infinite Loop.

Packets Sent: 8213

Progress:

**Tracing DNS packets in Wireshark :**


Capturing from Ethernet 2
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.   Time       Source        Destination    Protocol Length Info
 7849 61.204384  192.168.1.1   192.168.56.1   DNS       90 Standard query response
 7850 61.204400  192.168.1.1   192.168.56.1   DNS       90 Standard query response
 7851 61.219232  192.168.1.1   192.168.56.1   DNS       90 Standard query response
 7852 61.219249  192.168.1.1   192.168.56.1   DNS       90 Standard query response
 7853 61.234738  192.168.1.1   192.168.56.1   DNS       90 Standard query response
 7854 61.234749  192.168.1.1   192.168.56.1   DNS       90 Standard query response
 7855 61.250336  192.168.1.1   192.168.56.1   DNS       90 Standard query response
 7856 61.250346  192.168.1.1   192.168.56.1   DNS       90 Standard query response
 7857 61.266435  192.168.1.1   192.168.56.1   DNS       90 Standard query response
 7858 61.266446  192.168.1.1   192.168.56.1   DNS       90 Standard query response
 7859 61.281273  192.168.1.1   192.168.56.1   DNS       90 Standard query response

Frame 1957: 90 bytes on wire (720 bits), 90 bytes captured (720 b
Ethernet II, Src: TPLink_06:30:d0 (30:de:4b:06:30:d0), Dst: Intel
Internet Protocol Version 4, Src: 192.168.1.1, Dst: 192.168.56.1
User Datagram Protocol, Src Port: 53, Dst Port: 57371
Domain Name System (response)

0000  fc 44 82 d0 88 f6 30 de  4b
0010  00 4c 41 cb 00 00 39 11  85
0020  38 01 00 35 e0 1b 00 38  e2
0030  00 01 00 00 00 00 03 77  77
0040  65 03 63 6f 6d 00 00 01  06

Ethernet 2: <live capture in progress>   Packets: 7859 · Displayed: 7859 (100.0%)   Profile: Default
```

- **Algorithm**

```
# Initialize data structures
ddos_threshold = 100 #
Adjust as needed
ddos_counter = 0

while True:
    # Capture DNS traffic
    dns_packet = capture_dns_packet()

    # Extract relevant information from DNS response
    transaction_id, queried_domain, response_ip = extract_dns_info(dns_packet)

    # Check for DDoS
    if
        is_multiple_queries_same_domain(tr
        ansaction_id):ddos_counter += 1

    # DDoS threshold check
    if ddos_counter >= ddos_threshold:
        trigger_alert("Possible DDoS attack
        detected")
    # Continue capturing and analyzing DNS traffic
```

**Defense mechanism:**

To mitigate the risk of DNS DDoS attacks, organizations can implement several strategies:

1. DDoS Mitigation Services: Employ DDoS mitigation services that can detect and filter out malicious traffic, helping to ensure the availability of DNS services.

2. Anycast DNS Configuration: Distribute DNS servers across multiple geographically dispersed locations using Anycast technology. This can help distribute the load and provide redundancy, making it more challenging for attackers to overwhelm a single point of failure.

3. Traffic Monitoring and Analysis: Regularly monitor network traffic patterns and behavior to detect unusual or malicious activity.

4. Rate Limiting: Implement rate limiting on DNS servers to restrict the number of queries from a single source within a specified timeframe.

5. Firewall Configuration: Configure firewalls to filter out traffic from known malicious IP addresses.

By implementing these measures, organizations can enhance the resilience of their DNS infrastructure against DDoS attacks and ensure the continued availability of their online services.

**Conclusion**:

| **Attack** | **Fields Used in Attack** | **Comments** | **Defense Mechanism** |
|---|---|---|---|
| **DNS Spoofing** | DNS IP Address | Attacker modifies the DNS response to a DNS Query to redirect the target to a malicious URL. | Implement DNSSEC, use validated DNS resolvers, and employ cache poisoning protection measures. |
| **DNS Amplification** | Source IP Address | Attackers send small queries with spoofed source IP addresses to open DNS resolvers, triggering responses that are disproportionately larger, causing amplification. | Configure DNS servers properly, implement rate limiting, and use DDoS mitigation services. |

|  |  |  |  |
|---|---|---|---|
| **DNS Tunneling** | Query and Answer Sections | Attackers manipulate various fields, including the "Query" and "Answer" sections, to embed non-DNS data covertly. | Employ DDoS mitigation services, distribute DNS servers using Anycast, monitor network traffic, and implement rate limiting. |
| **DDoS Attack** | Source IP Address | The attacker floods the DNS server with a massive volume of requests. | Use deep packet inspection, signature-based detection, DNS logging, rate limiting, and DNS sinkholing. |

In conclusion, safeguarding the DNS protocol involves implementing DNSSEC for integrity, secure server configurations, monitoring for anomalies, defending against DDoS attacks, educating users, and considering encrypted DNS options. A comprehensive approach is crucial to ensuring the reliability and security of network services.