DoS Attack to the DNS Server (Using Spoofed IP Address)

Design Report

❖ *What is DoS (Denial-of-Service) attack?*

A denial-of-service (DoS) attack is a cyber-attack in which the attacker seeks to make a machine or network resource unavailable to its intended users by temporarily or indefinitely disrupting services of a host connected to the Internet. DoS is typically accomplished by flooding the targeted machine or resource with superfluous requests to overload systems and prevent some or all legitimate requests from being fulfilled.

❖ *What is a DNS server?*

The Domain Name System (DNS) is the phonebook of the Internet. When users type domain names such as ‘google.com’ or ‘nytimes.com’ into web browsers, DNS is responsible for finding the correct IP address for those sites. Browsers then use those addresses to communicate with origin servers to access website information.

❖ *What is IP address spoofing?*

IP address spoofing or IP spoofing is the creation of Internet Protocol (IP) packets with a false source IP address, for the purpose of impersonating another computing system.

IP address spoofing is most frequently used in denial-of-service attacks, where the objective is to flood the target with an overwhelming volume of traffic, and the attacker does not care about receiving responses to the attack packets. Packets with spoofed IP addresses are more difficult to filter since each spoofed packet appears to come from a different address, and they hide the true source of the attack. Denial of service attacks that use spoofing typically randomly choose addresses from the entire IP address space, though more sophisticated spoofing mechanisms might avoid non routable addresses or unused portions of the IP address space.

❖ *DoS Attack to the DNS Server (Using Spoofed IP Address)*

This attack is like DNS flood attack. This attack attempts to exhaust server-side assets (e.g., memory or CPU) with a flood of UDP requests, generated by scripts running on several compromised botnet machines.

A DNS flood attack is considered a variant of the UDP flood attack since DNS servers rely on the UDP protocol for name resolution and is a Layer 7 attack. With UDP-based queries (unlike TCP queries), a full circuit is never established, and thus spoofing is more easily accomplished.

To attack a DNS server with a DNS flood, the attacker runs a script, generally from multiple servers. These scripts send malformed packets from spoofed IP addresses. Since Layer 7 attacks like DNS flood require no response to be effective, the attacker can send packets that are neither accurate nor even correctly formatted.

The attacker can spoof all packet information, including source IP and make it appear that the attack is coming from multiple sources. Randomized packet data also helps offenders to avoid common DoS and DDoS protection mechanisms, while also making IP filtering completely useless.

Another common type of DNS flood attack is DNS NXDOMAIN flood attack, in which the attacker floods the DNS server with requests for records that are nonexistent or invalid. The

DNS server expends all its resources looking for these records, its cache fills with bad requests, and it eventually has no resources to serve legitimate requests.

❖ *Topology Diagram*

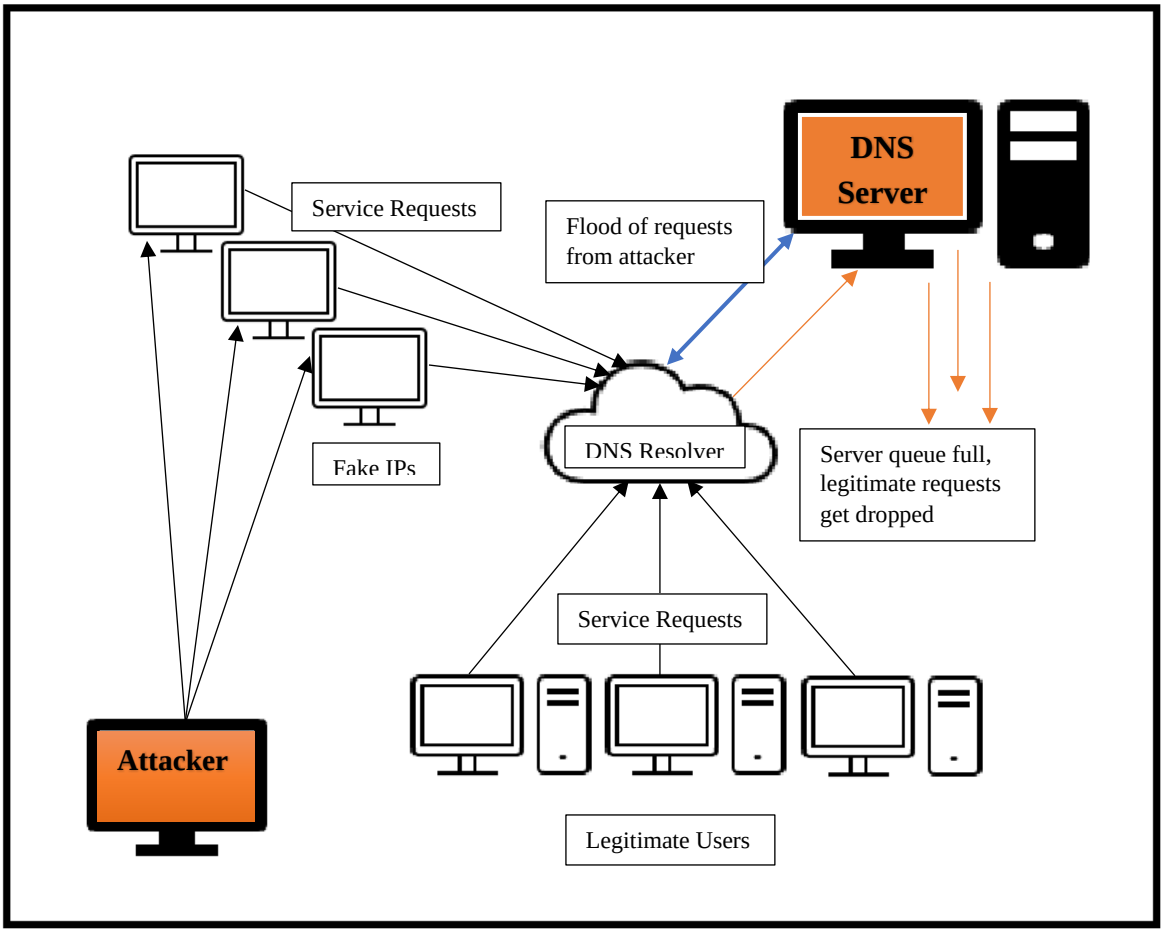


Figure: DoS Attack to the DNS Server (Using Spoofed IP Address)

❖ ***Timing diagram of User Datagram Protocol (UDP) in DNS server***

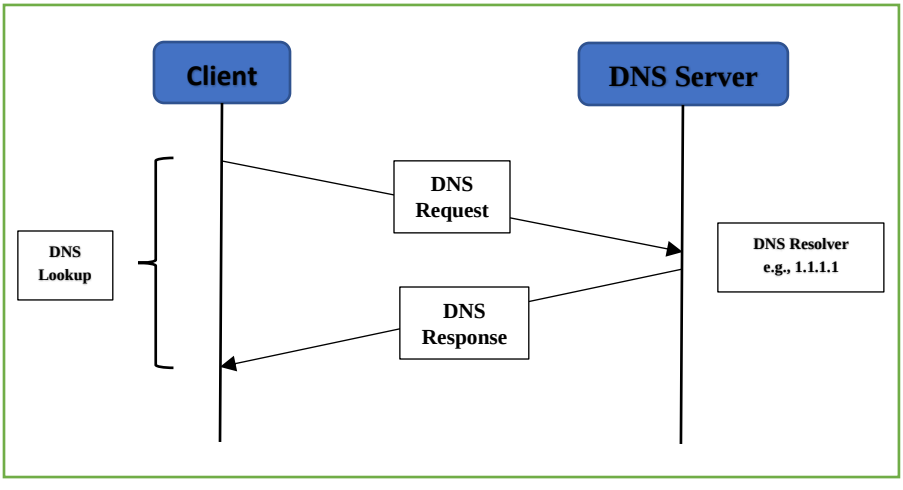


Figure: Timing Diagram for Original UDP

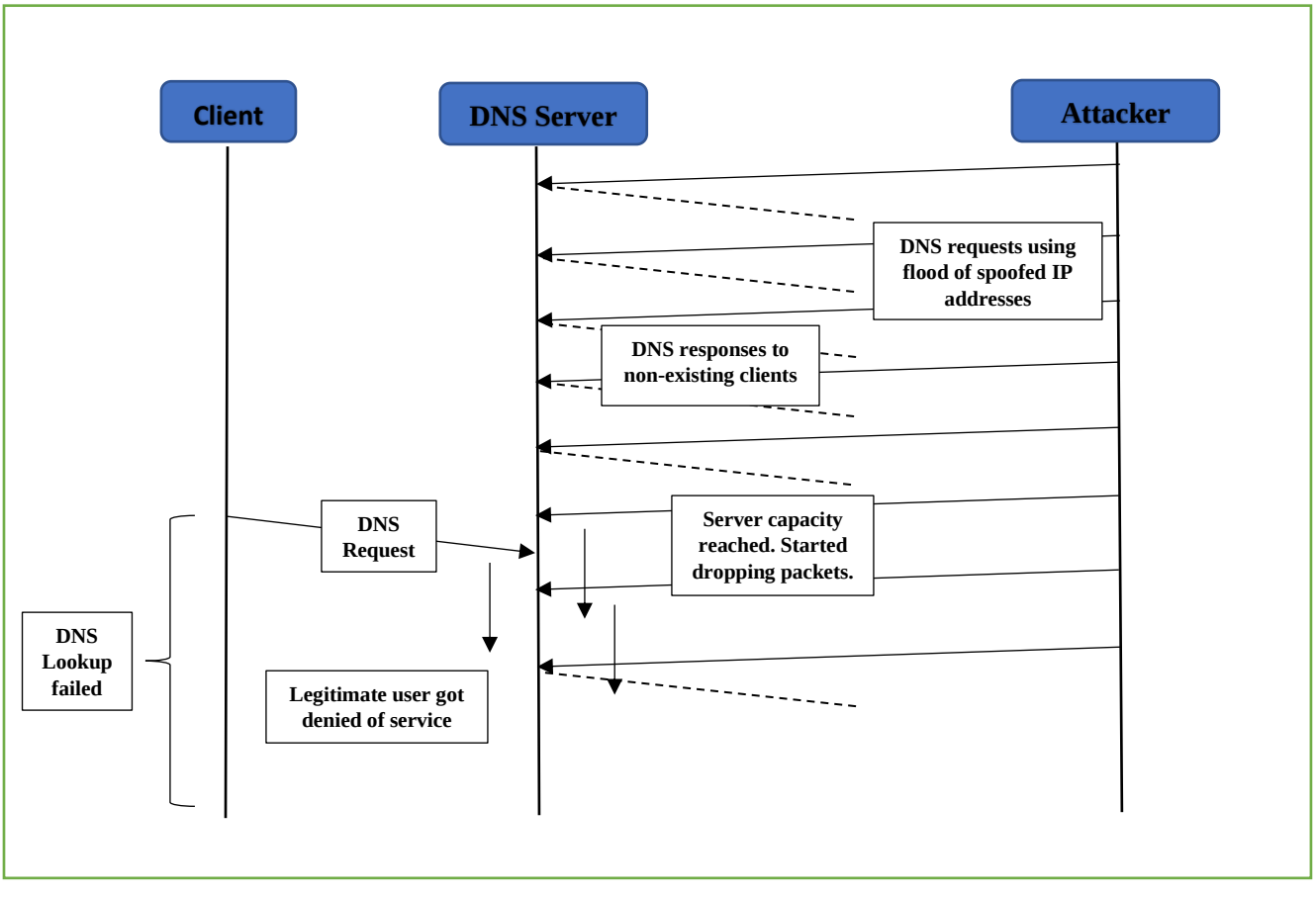


Figure: Timing Diagram at the time of DoS Attack

❖ *Packet/Frame details*

DNS uses the User Datagram Protocol (UDP) on port 53 to serve DNS queries. UDP is preferred because it is fast and has low overhead. A DNS query is a single UDP request from the DNS client followed by a single UDP reply from the server.

Messages carried by UDP are restricted to 512 bytes (not counting the IP or UDP headers). Longer messages are truncated, and the TC bit is set in the header.

The top-level format of message is divided into 5 sections (some of which are empty in certain cases) shown below:

Message Format:

Header	
Question	The question for the name server
Answer	RRs answering the question
Authority	RRs pointing toward an authority
Additional	RRs holding additional information

The header section is always present. The header includes fields that specify which of the remaining sections are present, and specify whether the message is a query or a response, a standard query or another opcode, etc.

The names of the sections after the header are derived from their use in standard queries.

The question section contains fields that describe a question to a name server. These fields are a query type (QTYPE), a query class (QCLASS), and a query domain name (QNAME).

The last three sections have the same format: a possibly empty list of concatenated resource records (RRs). The answer section contains RRs that answer the question; the authority section contains RRs that point toward an authoritative name server; the additional records section contains RRs which relate to the query but are not strictly answers for the question.

Header Section Format:

The header contains the following fields:

<table>
<tr><td colspan="10"></td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr>
<tr><td colspan="16">ID</td></tr>
<tr><td>QR</td><td colspan="4">Opcode</td><td>AA</td><td>TC</td><td>RD</td><td>RA</td><td colspan="3">Z</td><td colspan="5">RCODE</td></tr>
<tr><td colspan="16">QDCOUNT</td></tr>
<tr><td colspan="16">ANCOUNT</td></tr>
<tr><td colspan="16">NSCOUNT</td></tr>
<tr><td colspan="16">ARCOUNT</td></tr>
</table>

where:

ID A 16-bit identifier assigned by the program that generates any kind of query. This identifier is copied the corresponding reply and can be used by the requester to match up replies to outstanding queries.

QR A one bit field that specifies whether this message is a query (0), or a response (1).

OPCODE A 4-bit field that specifies kind of query in this message. This value is set by the originator of a query and copied into the response. The values are:

0	A standard query (QUERY)
1	An inverse query (IQUERY)
2	A server status request (STATUS)
3-15	Reserved for future use

AA Authoritative Answer - this bit is valid in responses, and specifies that the responding name server is an authority for the domain name in question section.

The contents of the answer section may have multiple owner names because of aliases. The AA bit corresponds to the name which matches the query name, or the first owner's name in the answer section.

TC TrunCation - specifies that this message was truncated due to length greater than that permitted on the transmission channel.

RD Recursion Desired - this bit may be set in a query and is copied into the response. If RD is set, it directs the name server to pursue the query recursively. Recursive query support is optional.

RA Recursion Available - this be is set or cleared in a response, and denotes whether recursive query support is available in the name server.

Z Reserved for future use. Must be zero in all queries and responses.

RCODE Response code - this 4-bit field is set as part of responses. The values have the Following interpretation:

0	No error condition
1	Format error - The name server was unable to interpret the query.
2	Server failure - The name server was unable to process this query due to a problem with the name server.
3	Name Error - Meaningful only for responses from an authoritative name server, this code signifies that the domain name referenced in the query does not exist.
4	Not Implemented - The name server does not support the requested kind of query.
5	Refused - The name server refuses to perform the specified operation for policy reasons. For example, a name server may not wish to provide the information to the requester, or a name server may not wish to perform a particular operation (e.g., zone transfer) for particular data.
6-15	Reserved for future use.

QDCOUNT An unsigned 16-bit integer specifying the number of entries in the question section.

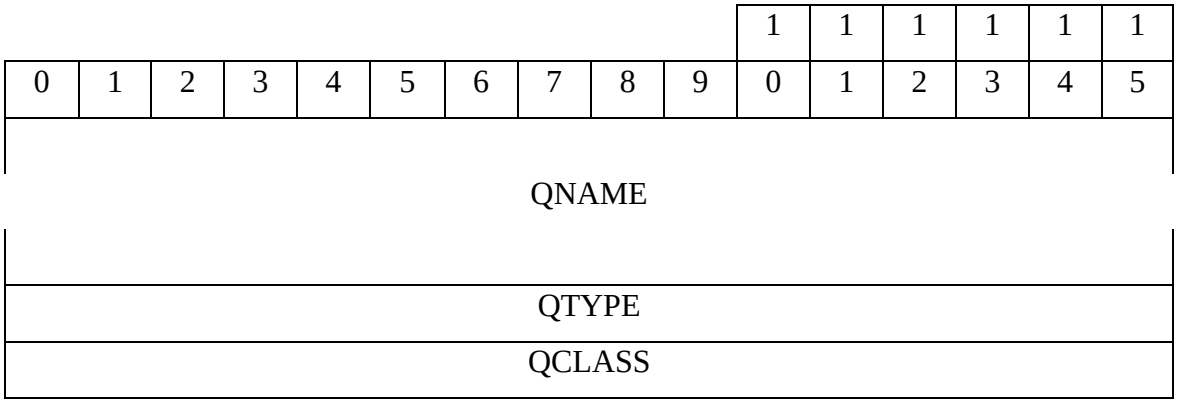
ANCOUNT An unsigned 16-bit integer specifying the number of resource records in the answer section.

NSCOUNT An unsigned 16-bit integer specifying the number of name server resource records in the authority records section.

ARCOUNT An unsigned 16-bit integer specifying the number of resource records in the additional records section.

Question Section Format:

The question section is used to carry the "Question" in most queries, i.e., the parameters that define what is being asked. The section contains QDCOUNT (usually 1) entries, each of the following format:



where:

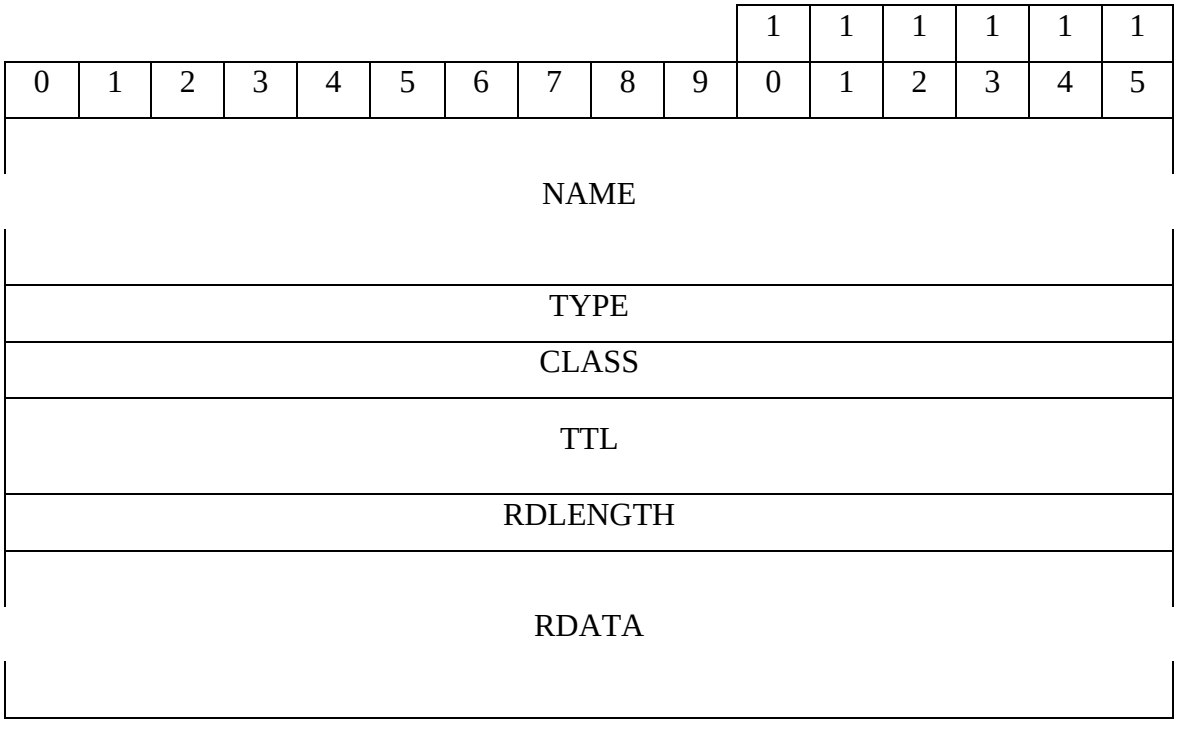
QNAME A domain name represented as a sequence of labels, where each label consists of a length octet followed by that number of octets. The domain name terminates with the zero-length octet for the null label of the root. This field may be an odd number of octets; no padding is used.

QTYPE A 2-octet code which specifies the type of the query. The values for this field include all codes valid for a TYPE field, together with some more general codes which can match more than one type of RR.

QCLASS A 2-octet code that specifies the class of the query. For example, the QCLASS field is IN for the Internet.

Resource Record (RR) Format:

The answer, authority, and additional sections all share the same format: a variable number of resource records, where the number of records is specified in the corresponding count field in the header. Each resource record has the following format:



where:

NAME A domain name to which this resource record pertains.

TYPE 2 octets containing one of the RR type codes. This field specifies the meaning of the data in the RDATA field.

CLASS	2 octets which specify the class of the data in the RDATA field.
TTL	A 32-bit unsigned integer that specifies the time interval (in seconds) that the resource record may be cached before it should be discarded. Zero values are interpreted to mean that the RR can only be used for the transaction in progress and should not be cached.
RDLENGTH	An unsigned 16-bit integer that specifies the length in octets of the RDATA field.
RDATA	A variable length string of octets that describes the resource. The format of this information varies according to the TYPE and CLASS of the resource record. For example, the if the TYPE is A and the CLASS is IN, the RDATA field is a 4-octet ARPA Internet address.

❖ *Modification in Packet Structure to execute DoS attack to the DNS Server*

The attacker can set the RD bit to enable recursive query for the DNS resolver. They can try building malformed headers and randomize packet data to bypass DoS or DDoS detection of the server. They can set the TTL value to maximum so that the attacking RR may be cached in the server for maximum amount of time. They can randomize the TTL value also to avoid detection.

❖ ***Why should this design work?***

A DNS flood attack is considered a variant of the UDP flood attack since DNS servers rely on the UDP protocol for name resolution and is a Layer 7 attack. With UDP-based queries (unlike TCP queries), a full circuit is never established, and thus spoofing is more easily accomplished.

To attack a DNS server with a DNS flood, the attacker runs a script, generally from multiple servers. These scripts send malformed packets from spoofed IP addresses. Since Layer 7 attacks like DNS flood require no response to be effective, the attacker can send packets that are neither accurate nor even correctly formatted.

The attacker can spoof all packet information, including source IP and make it appear that the attack is coming from multiple sources. Randomized packet data also helps offenders to avoid common DoS and DDoS protection mechanisms, while also making IP filtering completely useless.

The attacker can also flood the DNS server with requests for records that are nonexistent or invalid. The DNS server expends all its resources looking for these records, its cache fills with bad requests, and it eventually has no resources to serve legitimate requests.

The above explained scenarios make it quite tricky to handle or prevent this kind of attack on DNS server. IP spoofing makes it more difficult to handle. So, in theory this attack should work with this approach.
