# Chapter 2: Literature Review

## 2.1 Introduction.

In this chapter, related works pertinent to the proposal at hand focus on its nature, security issues, and existing gaps that contribute to its significance. A conceptual framework will map how the system works and how the proposed solution will fit within it.

## 2.2 DNS and DNS Spoofing Attacks.

### 2.2.1 Background of the DNS.

In the world of networking, computers don't go by names as humans do, they go by numbers, because that's how computers and other similar devices talk and identify with each other over a network, which is by using IP addresses. Humans on the other hand are accustomed to using names instead of numbers, whether talking directly to another person or identifying a country, place, or thing, humans identify with names instead of numbers. To bridge the communication gap between computers and humans and make communication a lot easier, Paul Mockapetris invented the DNS.

The DNS resolves domain names to IP addresses. If a web address is typed in your web browser, the DNS will resolve the name to a number because computers only know numbers. It works like a phone book, when you want to find a number, you don't look up the number first, you look up the name first, then it will give you the number. A web user doesn’t have to type in a web domain name to retrieve its web page, one could just type in the IP address instead, with the knowledge of what the IP address was. However, it is tiresome to memorize and deal with numbers, especially when there are millions of websites on the internet. It would be easier to type in the domain name instead and let DNS convert it to an IP address.

For example, to go to a website, belair.com, the DNS server will search through its database to find a matching IP address for that domain name, and when found, it will resolve that domain name to the IP address of the Belair website, and once that is done, then your computer can communicate with a Belair web server and retrieve the webpage.

If the web browser or operating system can't find the IP address in its cache memory, it will send the query to the next level to what is called the **resolver server**. The resolver server is your Internet service provider (ISP), so when the resolver receives the query, it will check its cache memory to find an IP address for belair.com, and if it can't find it, it will send the query to the next level which is the root server. The root servers are the top or the root of a DNS hierarchy. There are 13 sets of these root servers, and they are strategically placed around the world, and they are operated by 12 different organizations and each set of these root servers has its unique IP address.

When the root server receives the query for the IP address for belair.com, the root server is not going to know what the IP address is, but the root server does know where to send the resolver to help it find the IP address. Thus, the root server will direct the resolver to the TLD or top-level domain server for the dot com domain to now ask the TLD server for the IP address for belair.com. The top-level domain server stores the address information for top-level domains, such as .com, .net, .org, and so on. This TLD server manages the dot-com domain which belair.com is a part of. So, when a TLD server receives the query for the IP address for belair.com, the TLD server is not going to know what the IP addresses for belair.com are. The TLD will direct the resolver to the next and final level, which is the authoritative name servers.

Once again, the resolver will now ask the authoritative name server for the IP address for belair.com. The authoritative name server or servers are responsible for knowing everything about the domain which includes the IP address. They are the final authority. When it receives the query from the resolver, the authoritative name server will respond with the IP address for belair.com. And finally, the resolver will tell the computer the IP address for belair.com, and then your computer can now retrieve the Belair web page. It's important to note that once the resolver receives the IP address, it will store it in its cache memory in case it receives another query for belair.com so it doesn't have to go through all those steps again.

### 2.2.2 Analysis of Security Concerns and Spoofing Attacks on the DNS

Along with the benefits of the DNS, came a security concern stemming from the fact that security was not a consideration in its design, making the DNS susceptible to attacks. The fundamental reason is that the resolvers that issue the DNS query trust the responses that are received after they send out a query regardless of where that response comes from; In some cases, these responses can be forged. When a resolver sends a query, it typically generates what's called a **race condition** and if the Man in the Middle attacker replies before the legitimate responder, then the resolver is likely to believe the attacker. DNS responses can also contain additional DNS information that's unrelated to the query. The fundamental problem is that the basic DNS protocols have no means for authenticating responses which allows an attacker to forge responses after a resolver sends a query. A secondary reason that these types of spoofed replies are possible is that DNS queries are typically connectionless, unlike BPG where routing messages are transmitted over a reliable TCP connection. User Datagram Protocol (UDP) queries are sent over a connectionless UDP connection. Therefore, a resolver does not have a way of mapping the response that it receives for a query other than the query ID which can be forged by the attacker. let's look at how the combination of the lack of authentication and the connectionless nature of a DNS query allows the possibility for cache poisoning.

An attacker can easily gain access to and poison the DNS server such that it causes it not to point to the actual IP address of the queried domain name and gives a resolution of an IP address that points a computer user to a hacker-controlled website that could be a replica of the original website resulting to the user becoming a victim of a DNS Spoofing Attack.

It is done without changing the DNS server settings on the end user’s computer. A typical DNS spoofing act is cache poisoning or DNS poisoning. In DNS cache poisoning an attacker spoofs the DNS cache redirecting the user to the malicious site. The DNS cache is used to speed up the DNS resolution process of a website. Its IP address is resolved from the DNS server then it is stored as a DNS cache to avoid the need to contact the DNS server again next time when a query of the same website is made. The DNS cache stored on your computer quickly resolves the IP address speeding up the whole process. This cache is poisoned by replacing the IP address resolving a redirection to an attacker’s malicious site instead of the actual site. Attackers can also be a man in the middle and intercept the communication between the client and DNS server and spoof the DNS records. The DNS server could also be hijacked, and the DNS records be spoofed directly there. With the access, the attacker could gain access to the computer user’s information, breaking the confidentiality and privacy of the user or downloading ransomware or malware on the victim’s device.

## 2.3 Impact and Consequences of DNS spoofing attacks.

Due to the lack of proper authentication of the DNS records, can lead to security vulnerabilities, and with that follows a negative impact on users, institutions, and organizations. Attackers use DNS spoofing intending to intercept sensitive data through phishing and pharming attacks. It makes victims believe that the domain accessed is legitimate and in return, the attackers use the victims’ trust to infect them with malware and infect their systems.

As a result, confidential data can be breached or stolen such as passwords, credit card information, and other important data that can also be devastating for companies and institutions both in terms of financial cost and reputational damage. Attackers can use this stolen data as leverage to commit more crimes such as selling data, extracting money, perpetrating fraud, identity theft, and more that affect companies, individuals, institutions, and good governments. It affects everyone and anyone.

DNS spoofing could also lead to redirection to sites controlled by attackers that can unknowingly trick and cause a victim to install malware on a device. Automatically, this opens doors to further attacks and extensive espionage.

When a stub resolver sends a request and receives a response, it is stored in the cache such that any request of the same domain can be resolved faster by any client who makes the request. In the case where an attacker spoofs and sends a false response and the false response is stored in the cache, it will result in cache poisoning. This may pose a persistent threat as communication will be compromised from this point and cause damage over a longer period.

## 2.4 Detection and Prevention of DNS Spoofing Attacks and Existing Gaps.

There are several defenses against DNS cache poisoning and one is the query ID. However, the query ID can be guessed. Therefore, instead of having a resolver send queries where the IDs increment in sequence, the resolver can pick a random ID which makes the ID tougher for an attacker to guess. But the query ID is only 16 bits which still makes it possible for an attacker to flood the recursive resolver with many possible responses. It is also likely that with relatively few responses, one of these bogus responses will match the ID for the real query.

You can learn more about what's going on in your network by analyzing DNS logs. One can identify who is there, what they are attempting to access, and any deviations from the usual pattern of activity. However, admin users were also capable of more than just reviewing query logs and determining query intent. Administrators can spot malicious behavior patterns and infected devices by having complete information about every network query. However, merely examining these DNS requests does not provide a security guarantee. Suppose a hacker infiltrates your network and redirects everybody who goes to jumia.com to their phishing site. By only logging users’ queries, everything would appear to be perfectly fine. There would be no way to identify that the answer is pointing users to a malicious site.

A DNS firewall can also be an ideal tool to use to counter DNS from spoofing attacks among other attacks. It can help to detect and block malicious traffic before it reaches a network. It is located between the client’s stub resolver and the authoritative nameserver of the service to be reached. The firewall filters the traffic based on preconfigured rules and policies, allowing, or denying requests based on its origin, the data it contains, and whether it looks suspicious based on its nature. It can also be configured to block unwanted content or sites that can be of risk. However, DNS-based firewalls can be difficult to manage and maintain. Any changes to the DNS records must be manually updated in the firewall. The policies need constant monitoring and maintenance to ensure proper enforcement as well as updates to the policies. This can be a time-consuming process and can lead to errors if the firewall is not configured correctly.

VPNs could also be used to prevent DNS spoofing attacks as with a private network, there is a lower risk of an attacker interfering with the traffic compared to a public network. A problem can occur, for example, a DNS server of an organization or service is attacked which can result in that private network being insecure. Thus, any request from a client can be spoofed and additionally can lead to negative consequences on both ends.

## 2.5 Conceptual Framework.

A picture containing text, screenshot, diagram, design

Description automatically generated

A diagram of a computer server

Description automatically generated with low confidence

