Intro

In the world of networking, computers don't go by names like 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 numbers such as IP

addresses. Humans on the other hand are accustomed to using names instead of

numbers, whether is talking directly to another person or identifying a country,

place, or thing, humans identify with names instead of numbers. So in order to

bridge the communication gap between computers and humans and make the

communication of a lot easier,

networking engineers developed DNS, and DNS stands for a domain name system. And DNS

What is DNS

The DNS resolves names to numbers, to be more specific, it resolves domain names to IP addresses. So if you type in a web address in your web browser, DNS will resolve the name to a number because the only thing computers know are numbers. So for example if you wanted to go to a certain website you would open up your web browser and type in the domain name of that website, so for example let's use belair.com. Now technically you really don't have to type in belair.com to retrieve the Belair web page, you can just type in the IP address instead if you already knew what the IP address was, but since we are not accustomed to memorizing and dealing with numbers, especially when there are millions of websites on the internet, we can just type in the domain name instead and let DNS convert it to an IP address for us. So back to our example, when you type in yahoo.com your web browser the DNS server with search through its database to find a matching IP address for that domain name, and when it finds it it will resolve that domain name to the IP address of the Yahoo web site, and once that is done then your computer is able to communicate with a Yahoo web server and retrieve the webpage. So DNS

basically 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.

How DNS works

So to break this down into further detail let's examine the steps that DNS

takes. So when you type in belair.com in your web browser and if your web browser

or operating system can't find the IP address in its own cache memory, it will

send the query to the next level to what is called the resolver server. The resolver

server is basically your ISP or Internet service provider, so when the resolver

receives the query, it will check its own 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 their own unique IP address. So 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. So the root server will direct the resolver to the TLD or top level domain server for the dot com domain. So the resolver will now ask the TLD server for the IP address for belair.com. The top level domain server stores the address information for a top level domains, such as .com, .net, .org and so on. This particular 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. So the TLD will direct the resolver to the next and final level, which are the authoritative name servers. So 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. So, when the authoritative name server 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 your 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.

DNS SPOOFING

DNS spoofing is an attack in which the attackers spoof the DNS result without changing the DNS server settings on the end user 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 resolve 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.

Why Is the DNS vulnerable

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 case, 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.

DNS Cache Poisoning

To see how a DNS cache poisoning attack works consider a network where a stub resolver issues a query to its recursive resolver and in turn it sends that record query to the start of authority for that domain. In an ideal world, the authoritative nameserver for that domain would reply with the correct IP address. If an attacker guesses that a recursive resolver might eventually need to issue a query for say google.com, the attacker can simply reply with multiple specially crafted replies each with different IDs. Although this query has same query ID, the attacker doesn't need to see that query because the attacker can simply flood the recursive resolver with a bunch of bogus replies and one of them.

In this case, the response with ID 3 will match if this bogus response reaches the recursive resolver before the legitimate response does. The recursive resolver will accept this bogus message. At worse, it caches the bogus message, and the DNS unfortunately has no way to expunge a message once it has been cached. This recursive resolver will continue to send bogus a record response for any query for this particular domain name until that entry expires from the cache.

Defenses of DNS Spoofing Attacks

There are several defenses against DNS cache poisoning and one, which is the query ID. However, the query ID can be guessed. The next defense is to randomize the ID. 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 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 be the ID for the real query. This is due to the birthday paradox where the success probability for achieving a collision between the query ID of the query and of the response only requires sending hundreds of replies not a complete 32,000.

Moreover, the probability that such an attack will succeed using only a few hundreds of replies is relatively close to one as the attacker does not need to send replies with all two to the 16th possible IDs. The success of a DNS cache poisoning attack not only depends on the ability to reply to a query with the correct matching ID, but it also depends on winning this race. The attacker must reply to that query before the legitimate authoritative name server replies. If the attacker loses the race then the attacker has to wait for that correct cached entry to expire before trying again.

However, the attacker can generate his own DNS query and for example query one google.com to google.com and so forth each one of these bogus queries will generate a new race and eventually the attacker will win one of these races for a record query. But no attacker necessarily cares to own one google.com or google.com. The attacker really wants to own the entire zone by responding with NS records for the entire zone of google.com. By creating one of these races using a record query and then responding not only with the record response, but also with the authoritative of the NS record for the entire zone, the attacker can in fact own the entire zone. This idea of generating a stream of record queries to generate a bunch of races and then stuffing the record responses for each of these with a bogus authoritative DNS record for the entire zone is what's called the Kaminsky attack after Dan Kaminsky who discovered the attack the defenses of picking a query ID and randomizing the ID help but remember the randomization is only 16 bits so let's think about other possible defenses.

Consequences of DNS Spoofing Attacks

Spoofing, hijacking, and cache poisoning attacks can also hurt website conversions by directing prospective customers away from the legitimate website. In addition, having one’s site be seen as poorly secured can hurt an organization’s brand reputation in the long run. #CrazyFaith#CrazyMe@270202!

Types of DNS spoofing

The following three attack types refer to the diagram above (A-C).

Type (A): attack on the client or local router

This type of DNS spoofing attack involves malicious tampering on the local device or home router. To the victim, everything seems fine at first. The device connects to the DNS server as usual. However, malicious IP addresses may be returned for the requested host names.

With this kind of attack, the threat will remain until the tampering has been corrected. Nevertheless, the attacker needs an attack vector in order to tamper with anything. This can be a technical factor, such as open admin access, a weak password, or something in a similar vein. An attacker can also use [social engineering](https://www.ionos.com/digitalguide/server/security/social-engineering-the-security-gap-at-layer-8/) to trick the victim into making the change themselves in good faith.

Changing the DNS server on the local system

The DNS spoofing attack, known as a “local hijack”, sets the IP address of the DNS server to a malicious address in the network settings of the local device.

This change can be detected by the victim and easily reversed. However, this form of tampering is often accompanied by malware which can restore the malicious entry if the victim changes it.

 Tip

Use the online tool [WhoismyDNS](http://whoismydns.com/" \o "Who is My DNS? Free DNS Hijack Detector" \t "_blank) to check whether you have fallen victim to this type of DNS spoofing.

Tampering with the hosts file on the local system

Most operating systems use a “hosts” file to enable name resolution of certain domains to be performed on the local system. If a malicious entry is placed in this file, data traffic will be redirected to a server being controlled by the attacker.

This type of tampering is permanent. However, it can easily be detected by an experienced victim. To fix this problem, all you need to do is change the [hosts file](https://www.ionos.com/digitalguide/server/configuration/hosts-file/).

Hijacking the local router

The IP address of the internet service provider’s DNS server is set on the local router by default. In a “router hijack”, this is replaced by a malicious address. This attack poses a threat to all data traffic passing through the router. Since there are usually multiple devices in a household that will use the router to establish a connection, several parties can fall victim to the attack.

Many users are unaware that they can configure their router themselves. So, this attack often remains undetected for a long time. If any problems occur later, the victims are more likely to suspect that the source is their own device rather than the router. Therefore, it’s well worth considering that the router might be the source of the error in the event of any weird problems.

 Tip

Use the [F-Secure Router Checker](https://www.f-secure.com/us-en/home/free-tools/router-checker) to check whether you have fallen victim to this type of DNS spoofing.

Type (B): attack on the DNS server’s response

This type of DNS spoofing is a man-in-the-middle attack. The attacker pretends to be the victim’s DNS server and sends them a malicious response. This type of attack works because DNS traffic uses the unencrypted [User Datagram Protocol (UDP)](https://www.ionos.com/digitalguide/server/know-how/udp-user-datagram-protocol/). There is no way for the victim to verify the authenticity of the DNS response.

Other kinds of attacks such as [ARP spoofing](https://www.ionos.com/digitalguide/server/security/arp-spoofing-attacks-from-the-internal-network/) and [MAC spoofing](https://www.ionos.com/digitalguide/server/know-how/what-is-mac-spoofing/) can be used to gain access to the local network. The use of encryption technologies protects against many man-in-the-middle attacks.

Type (C): attack on the DNS server

This type of DNS spoofing attack targets a legitimate DNS server and can affect a large number of users. It’s a high-level type of attack, as multiple security mechanisms usually have to be overcome to hack the server.

Poisoning the DNS cache on the server

DNS servers are arranged in hierarchies and communicate with one another. An attacker can use [IP spoofing](https://www.ionos.com/digitalguide/server/security/ip-spoofing-fundamentals-and-counter-measures/) to pretend to be one of these servers and trick a server into accepting a false IP address for a domain. The server places the malicious entry in its cache, and begins “poisoning” it.

Any request to the server after the cache is poisoned will result in the malicious entry being returned to the victim. The threat will remain until the entry is removed from the cache. The [DNSSEC extension](https://www.ionos.com/digitalguide/server/know-how/dnssec-internet-standards-for-authenticated-name-resolution/) serves as a server-side security mechanism. It can be used to secure server communication within the DNS.

 Tip

Use the [Domain Guard from IONOS](https://www.ionos.com/domains/domain-guard?ac=OM.US.USo50K361685T7073a&itc=5L4C4XEV-5ICVBT-38NS5G8) to protect your domain from any tampering.

Hijacking a DNS server

This type of attack, also known as a “rogue hijack”, is probably the most complex kind of DNS attack. This involves an attacker taking control of a legitimate DNS server. Once compromised, even the most current DNS encryption will provide no protection. However, the content encryption should at least alert the victim to the attack.

How to protect yourself from DNS spoofing

As you can see, DNS spoofing is a serious threat. Fortunately, there are a number of simple measures you can take that provide effective protection against DNS spoofing.