DNS Spoofing techniques

By Spacefox, spacefox@securesphere.net
Secure Sphere Crew - January 23rd, 2002

"What you must learn is that these rules are no different than the rules
of a computer system. Some of them can be bent, others can be broken." - Matrix

Overview : What is DNS Spoofing ?

DNS Spoofing is the art of making a DNS entry to point to an another IP than it would be supposed to point to. To understand better, let's see an example.
You're on your web browser and wish to see the news on www.cnn.com, without to think of it, you just enter this URL in your address bar and press enter.

Now, what's happening behind the scenes ? Well... basically, your browser is going to send a request to a DNS Server to get the matching IP address for www.cnn.com, then the DNS server tells your browser the IP address of CNN, so your browser to connect to CNN's IP address and display the content of the main page.

Hold on a minute... You get a message saying that CNN's web site has closed because they don't have anymore money to pay for their web site. You're so amazed, you call and tell that to your best friend on the phone, of course he's laughing at you, but to be sure, he goes to CNN web site to check by himself.

You are surprised when he tells you he can see the news of the day as usual and you start to wonder what's going on. Are you sure you are talking to the good IP address ? Let's check. You ask your friend to fire up his favorite DNS resolving tool (or simply ping) and to give you the IP address he's getting for www.cnn.com.
Once you got it, you put it in your browser URL bar : http://212.153.32.65
You feel ridiculous and frustrated when you see CNN's web page with its daily news.

Well you've just been the witness of a DNS hijacking scenario. You're wondering what happened, did the DNS Server told you the wrong IP address ? Maybe... At least this is the most obvious answer coming to our mind.

In fact there are two techniques for accomplishing this DNS hijacking. Let's see the first one, the "DNS ID Spoofing" technique.

A) DNS Cache Poisoning

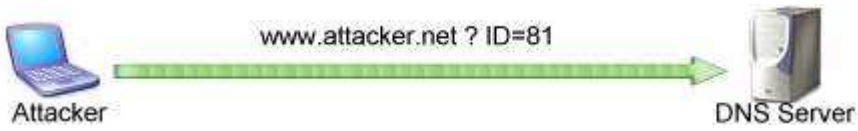
As you can imagine, a DNS server can't store information about all existing names/IP on the net in its own memory space.
That's why DNS server have a cache, it enables them to keep a DNS record for a while.

In fact, A DNS Server has the records only for the machines of the domain it has the authority, if it needs to know about machines out of his domain, it has to send a request to the DNS Server which handles these machines and since it doesn't want to ask all the time about records, it can store in its cache the replies returned by other DNS servers.

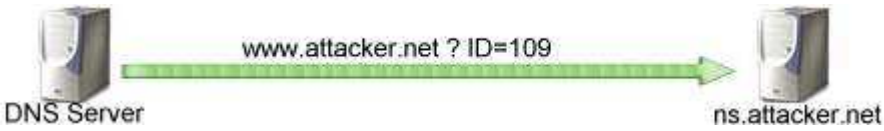
Now let's see how someone could poison the cache of our DNS Server.

An attacker his running is own domain (attacker.net) with his own hacked DNS Server (ns.attacker.net)
Note that I said hacked DNS Server because the attacker customized the records in his own DNS server, for instance one record could be www.cnn.com=81.81.81.81

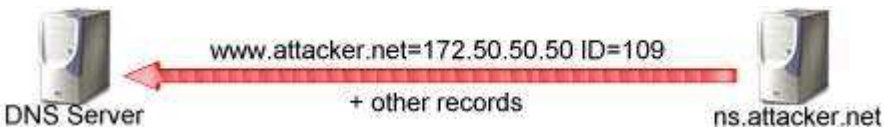
1) The attacker sends a request to your DNS Server asking it to resolve www.attacker.net



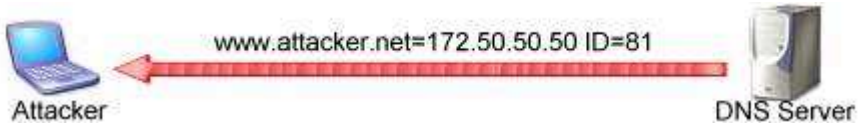
2) Your DNS Server is not aware of this machine IP address, it doesn't belongs to his domain, so it needs to asks to the responsible name server.



3) The hacked DNS Server is replying to your DNS server, and at the same time, giving all his records (including his record concerning www.cnn.com)
Note : this process is called a zone transfer.



4) The DNS server is not "poisoned".
The attacker got his IP, but who cares, his goal was not to get the IP address of his web server but to force a zone transfer and make your DNS server poisoned as long as the cache will not be cleared or updated.



5) Now if you ask your DNS server, about www.cnn.com IP address it will give you 172.50.50.50, where the attacker run his own web server. Or even simple, the attacker could just run a bouncer forwarding all packets to the real web site and vice versa, so you would see the real web site, but all your traffic would be passing through the attacker's web site.

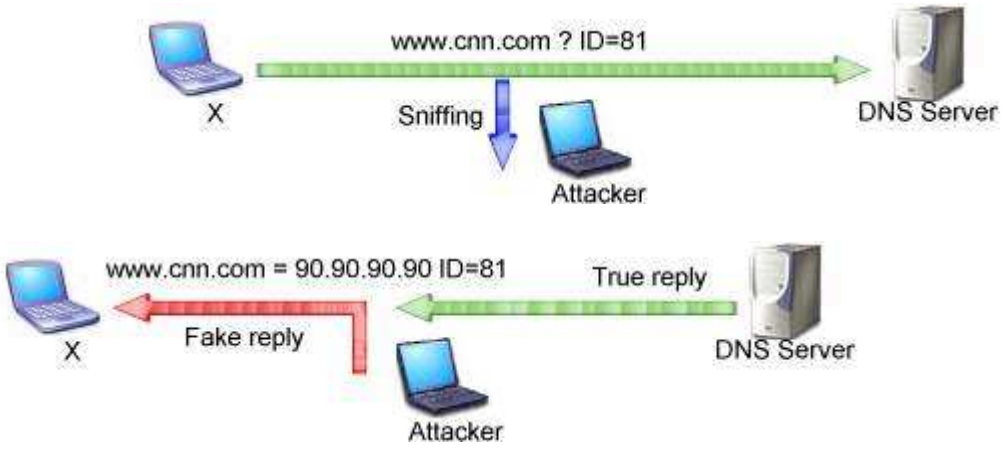
B) DNS ID Spoofing

We saw that when a machine X wants to communicate with a machine Y, the former always
needs the latter IP address. However in most of cases, X only has the name of Y, in that case, the DNS protocol is used to resolve the name of Y into its IP address.

Therefore, a DNS request is sent to a DNS Server declared at X, asking for the IP address of the machine Y. Meanwhile, the machine X assigned a pseudo random identification number to its request which should be present in the answer from the DNS server.
Then when the answer from the DNS server will be received by X, it will just have to compare both numbers if they're the same, in this case, the answer is taken as valid, otherwise it will be simply ignored by X.

Does this concept is safe ? Not completely. Anyone could lead an attack getting this ID number. If you're for example on LAN, someone who runs a sniffer could intercept DNS requests on the fly, see the request ID number and send you a fake reply with the correct ID number... but with the IP address of his choice.
Then, without to realize it, the machine X will be talking to the IP of attacker's choice thinking it's Y.

By the way, the DNS protocol relies on UDP for requests (TCP is used only for zone transferts), which means that it is easy to send a packet coming from a fake IP since there are no SYN/ACK numbers (Unlike TCP, UDP doesn't provide a minimum of protection
against IP spoofing).



Nevertheless, there are some limitations to accomplish this attack.

In my example above, the attacker runs a sniffer, intercept the ID number and replies to his victim with the same ID number and with a reply of his choice.

In the other hand, even if the attacker intercepted your request, it will be transmitted to the DNS Server anyway which will also reply to the request (unless the attacker is blocking the request at the gateway or carry out ARP cache poisoning which would make the attack possible on a switched network by the way).

That means that the attacker has to reply BEFORE the real DNS server, which means that to succeed this attack, the attacker MUST be on the same LAN so to have a very quick ping to your machine, and also to be able to capture your packets.

Practical example (to be done a network for testing purposes ONLY)

To see yourself how to hijack a connection from a machine on your local area network, we can do the followings :

First step : Poison the ARP cache of the victim's machine (tools and explanations for realizing this task can be found at http://www.arp-sk.org)

Second step : Now, outgoing packets of the target will be redirected to your host, but you have to forward the traffic to the real gateway, this can be achieved with a tool like Winroute Pro.

Third step : We then use WinDNSSpoof, developed by valgasu (www.securiteinfo.org) which is
a tool that greatly help to carry out DNS ID Spoofing. (Before to use this tool be sure you have the Winpcap library installed on your machine, see http://winpcap.polito.it).
We run it in the cmd like :

wds -n www.cnn.com -i 123.123.123.123 -g 00-C0-26-DD-59-CF -v

This will make www.cnn.com to point to 123.123.123.123 on the victim's machine.
00-C0-26-DD-59-C being the MAC Address of the gateway or DNS server.

WARNING : Please keep in mind that the use of these tools on a network without explicit authorization of the administrator is strictly forbidden.

C) Making the attack more accurate with the Birthday Paradox

What is that Birthday Paradox ?

The birthday paradox gets its name from the "strange" fact that in a gathering of 23 persons, it's likely that 2 of these persons will have the same birthday date. To understand it, no need to be an ace in mathematics.

You're in a party and you go to ask someone his birthday date, the chances that you not sharing the same birthday date with this person are 364/365 or 0.997, therefore the probability that you do share the same birthday date is 1 - 0.997 = 0.003

Now if you ask somebody else, the chances that you don't share the same birthday date than him AND the guy before are (364/365) x (363/365) = 0.992 and so we can deduce that the probability that at least, 2 of all of you share the same birthday date is 1 - 0.992 = 0.008

If we carry on these computations for some time, we find out that in a group of 23 persons, the chances are 50% that you someone finds someone else who has the same birthday date than him. You can use the following C code snippet to see how chances are evoluating in function of the number of persons.

```
#define POSSIBILITIES 365.0

void main (void)
{
 float chances;
 int i, j;

 for (i = 1; i < 100; i++)
 {
  for (j = 1, chances = 1; j < i; j++)
    chances *= (float)((POSSIBILITIES - j) / POSSIBILITIES);

  printf("For %d people, chances are %f\n", i, 1-chances);
 }
}
```

For people unable to compile this code, here's an interesting array of outputed values :

People	2	9	16	23	30	37	44	65	79
Chances	0.0027	0.0946	0.2836	0.5073	0.7063	0.8487	0.9329	0.9977	0.9999

Let's get back to our attack, in a conventionnal DNS spoofing attack as explained previously, our attacker was wiretapping on the network so as to get the ID number of the request from X and to send a reply with the same ID but containing an IP of the attacker's choice.

As I said before, this attack requires the attacker to be able to wiretap the DNS network traffic generated by X. Does this means that the attacker can't carry out this attack without running a sniffer ?

What about trying to "guess" this ID instead ?

Why not, but the ID number is coded on 2 bytes, in other words... 65535 possible values ! That means that if he wishes to succeed this attack, he would need to fire 65535 fake replies with different ID numbers so at least one packet would contain the correct ID number.

This would require a good bandwidth/ping and the main problem is to know WHEN to send the
replies ? He would have to know when the request has been made, and fire up his packets RIGHT after that, and wishes that his fake replies reach X at the right moment (that is to say, before the real reply from the DNS server gets to X).

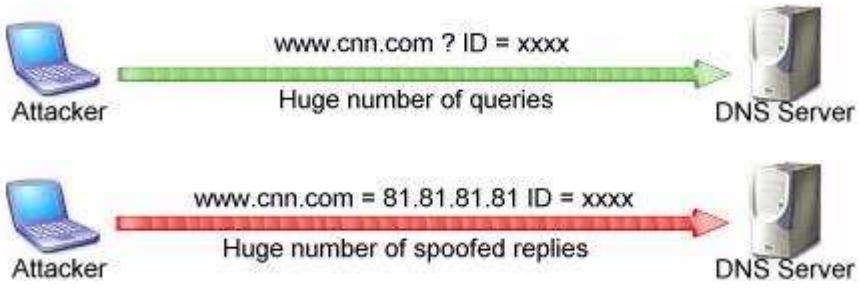
Let's take an another point of view, we saw before that it is possible to "poison" directly the DNS server.
To remind you, the attacker was asking a DNS server to resolve www.attacker.net, and then thanks to the zone transfert with malicious records from the ns.attacker.net, the attacker was able to poison the cache of our DNS server. Once again, the limitation of this attack is that the attacker must run its own DNS server with malicious records in it.

What if this attacker does not have ability to sniff your traffic or run its own name server, would that mean that you are safe from any DNS hijacking technique ?
The answer is, not at all.

As I mentionned earlier, the DNS protocol is relying on UDP, I also remind you that UDP is not a "connected" protocol, in the way that we just send and receive data without creating a session like TCP. It is then, pretty easy to send an UDP packet with an IP of your choice.
So why the attacker woud give himself hard time by setting up a malicious DNS server, when he could send spoofed packet from any DNS server ? He could just ask the victim's DNS server to resolve www.cnn.com, and then immediately send a spoofed reply from the nameserver of www.cnn.com with a fake IP address.

Yes, in timing matter, it's feasible, the problem is that the victim's DNS server is going to send a query to ns.cnn.com to get www.cnn.com IP address with an ID number, once again the attacker would have to send 65535 spoofed packets with ns.cnn.com as source address to the victim's name server. At least one answer would be correct.
This "could" be successfull.

Now here's the interesting part... what if the attacker would send let's say 100 requests to the victim's DNS server for resolving www.cnn.com ? Well ns.victim.com would send also 100 requests to ns.cnn.com, then what if we would send 100 spoofed replies from ns.cnn.com to ns.victim.com ? Well if you understood the Birthday Paradox mentionned earlier, you should understand that probabilities of collision are seriously increased compared to the other method.



But there's still a small detail which needs to be taken in account, the source port !

Yes, think of it, ns.victim.com is going to send a request to ns.cnn.com, the UDP header would be like this :

Source address : ns.victim.com
Destination address : ns.cnn.com
Source port : 1256 (choosed randomly and > 1024)
Destination port : 53 (DNS port)
Data : What is the IP of www.cnn.com ?

It is obvious that the attacker must send the spoofed DNS reply on the source port of ns.victim.com as a destination port, which would be like this :

Source address : ns.cnn.com
Destination address : ns.victim.com
Source port : 53
Destination port : 1256
Data : The IP of www.cnn.com is 81.81.81.81

So how to guess this source port if we are not sniffing ?
Well unfortunately on most of the DNS server, the source port will not change for each client, so an attacker could know the source port currently used by ns.victim.com pretty easily.
For instance, if he would be running an authoritative nameserver, he could just send a request for a DNS lookup of a hostname of his domain, the received recursive query packet
would contain the source port currently used by ns.victim.com for sending DNS queries.

Ok, now that we know how we can find the source port, you may wonder what are the probabilities of this attack to be successful, and this is exactly what we are going to study now, and Why not to carry out some changes to our C program to calculate all this ?

```
#define POSSIBILITIES 65535.0

void main (void)
{
 float chances;
 int i, j;

 for (i = 0; i < 800; i+=50)
 {
  for (j = 1, chances = 1; j < i; j++)
    chances *= (float)((POSSIBILITIES - j) / POSSIBILITIES);

  printf("For %d fake replies, chances are %f\n", i, 1-chances);
 }
}
```

And here's the reported array :

Queries	50	100	150	200	250	300	350	400	500	550	650	750
Chances	0.0185	0.0728	0.1569	0.2621	0.3785	0.4961	0.6069	0.7048	0.8517	0.9008	0.9604	0.9865

We can see that, after that for 650 queries/fake replies, chances are about 0.960411, that is almost 100% !

For more detailed informations, I advice you to read the following articles :

http://www.kb.cert.org/vuls/id/457875
http://www.securityfocus.com/guest/17905

Conclusion :

In this article, I took the example of www.cnn.com, which is not really interesting for an attacker to redirect. But the problem is much more important when accessing to your bank, to your online book store or even to your web mail cause all given information could be easily read by the attacker.

The counter-measures which can be taken by administrators to reduce the risks are mainly :

- To limit the cache and check that it's not keeping additional records.
- Not to make security systems to use/rely on DNS.
- Use cryptography like SSL, even if the problem remains the same, it increase difficulty level
for the attacker (See article on Man in the Middle)