DNS Spoofing

DNS spoofing essentially means that the Domain Name Server uses incorrect information and returns that to the user accessing a website. What this allows attackers to do is reroute legitimate domain names to unauthorized servers or redirect emails to unauthorized mail servers (SANS 16). Within the SANS DNS spoofing paper there are three ways that the attack can be most likely carried out. The first one described involves exploiting the way that DNS requests work. If Trudy knows that there is a 16-bit ID associated with a DNS request, and has a way of guessing what that request might be then there is a chance to redirect the user looking for the original web server to a false server. This is extremely dangerous because the typical user that is trying to access a website will not know that they are being redirected to the incorrect web server (SANS 16). The URL will be identical, but the IP address will be for a malicious website that Trudy has established. This ties into another topic of discussion within security: phishing. If Trudy is able to set up a false site that is convincing enough, and is able to spoof the DNS to redirect to this convincing server she may be able to get information from the oblivious user. The way to avoid this type of attack (reliably) from a user level is to use the IP addresses of the websites that you are trying to access instead of the URL. This is not a viable option for typical users, and therefore this type of attack can be very destructive to those not expecting DNS spoofing.

The second way to carry out a DNS spoofing attack described by SANS is DNS cache poisoning (SANS 16). What this entails is that the malicious user, Trudy, finds a way to insert fake records into the DNS cache, which can be information that redirects to the website that Trudy has established. (Wikipedia 2017) The “poison” in this scenario is the false record, and this would allow a similar exploit to the one about to be executed. Another example of something that can be targeted by a DNS spoofing described in the paper is falsifying auto-updates. If Trudy can make the user download fake updates that run her malicious code instead of an actual update while the user is oblivious then that can have devastating effects on the user.

The final attack described is more involved, and utilizes other common attacks to try to gain access to the system. The specific attack given as an example is using a buffer overflow to break into the system that is running the DNS server (SANS 16). If this malicious user is actually able to gain root access of the system then they may have complete control over cache, records, IP addresses, etc. This would be a potential worst case scenario for the system administrators and user, but best case for Trudy. If she is using the DNS on the system to redirect to her false servers then this is still technically DNS spoofing, but this is a much more direct attack, and one that allows Trudy much more devastating access to the system.

What quickly becomes clear with DNS spoofing is that it may be a very efficient way for Trudy to target users without them suspecting anything is wrong. There are many plausible scenarios where an attack of this nature may be useful to Trudy, the most obvious being phishing. After DNS spoofing (either by spoofing response IDs, cache poisoning, or direct attack), Trudy is able to redirect the users who are looking for the legitimate site to her own malicious site. In an ideal world (for Trudy) the website that Trudy has established will be virtually indistinguishable from the actual site to the user. If this is a banking website, for example, what Trudy would want is to setup a login field that resembles the one found on the actual site and have the unsuspecting users enter their identifying information (which is being logged). In order to cover the fact that this is not the real site Trudy would then want to send the users to the real bank’s website if possible. Even without this final step, Trudy would still have gotten the information she was looking for from the users, but it would be less obvious that something is wrong if the users ended up on the correct website (actually looking at their account information). This is a fairly sophisticated phishing attack that is facilitated by the use of DNS spoofing to get the user onto the malicious server.

Another potential attack (and a very interesting concept) is using auto-updates in windows to install malicious software by way of DNS spoofing (SANS 17). This is a concept that would allow Trudy to hide her activity behind the guise of updating software. The way this attack would basically function is Trudy would serve the “update” from her own server by spoofing the DNS for the address of the updates website. This would allow Trudy to feed the user any code she wanted, and allow malicious software on the user’s computer. If Trudy were trying to cover her footprints then she would want to use a similar tactic to the one above, in that she would want to serve an actual update along with the malicious software. This way if the user actually receives the update they were expecting it becomes less suspicious.

Conclusion

Spoofing is clearly a key tool in the arsenal of Trudy, and understanding how IP, DNS, and ARP spoofing work is extremely important. This topic appealed as an area of research for us because of how powerful of a tool it can be in the hands of a malicious user.