To: William H. Flathead III

From: Matteo Franzil

Subject: DNS and MiTM explaination

Date: October 31, 2021

I am writing this memo in order to explain the DNS and MiTM attacks. This is the latest related to the series of memos on vulnerability testing performed by my group.

This time, we decided to experiment on Frobozz's DNS server and see what would happen if an attacker were to perpetrate what's so called a MiTM (*Man-in-The-Middle*) attack on our DNS infrastructure. As it stands, our internal DNS system has not been updated for the last 7 years, and is in dire need of maintainance. Still, for legitimate users it works flawlessly: the DNS requests are first sent to the cache server, then forwarded to the authoritative servers, and then the response is forwarded back to the client, recursively.

However, assume that an intruder was able to intercept the DNS requests. The malicious server would then send back a forged response, which will be forwarded back to the client. The client would then receive the forged response, which would be then interpreted as a legitimate DNS response, unaware of the fact that it was forged. By doing this, any attacker would be able to redirect clients, read the contents of their messages, and much more.

The first and foremost method for subverting a DNS system is by ARP spoofing. ARP spoofing refers to a technique in which attackers repeatedly send forged ARP packets to a victim - the cache server, pretending to be someone else - for example, the authoritative server. The victim' will believe to the ARP content and their cache will be poinsoned. The attacker will then be able to pretend to be the authoritative server, and thus, intercept any DNS requests directed from the cache to the server.

Once the cache has been fully poisoned and the system has been set up in order to spoof DNS requests - which was done with a tool called `ettercap` and a plugin called `dns_spoof` - any request made from the client will now travel through the cache, the attacker, and then to the authoritative server, but no trace will be left of such a change in DNS responses, unless the client does a `traceroute` and notices the extra hop.

Mitigating this involves using what is called DNSSEC (*DNS Security Extensions*). DNSSEC is a standard for adding authentication, but not confidentiality, to DNS. DNSSEC allows DNS resolvers to verify the integrity of the response with public key cryptography, ensuring it is not forged. Nonetheless, DNSSEC does not prevent DoS attacks, ARP spoofing or MiTM itself - attackers will still be able to see traffic in clear, as DNSSEC is still not encrypted. The main point is that DNS clients will be able to verify the chain of trust by checking the digital signature embedded in the response. protect against DoS attacks directly.

In our testbed, we successfully run a test with the aformentioned setup (client, cache, attacker, auth-server), and we came to the conclusion that a migration to DNSSEC (or DNS-over-TLS, a different standard that does provide encryption) would be recommended to our system, in the wake of the recent attacks to webservers that may have let attackers into our network.