Exercise 7 – DNS lab report

Stewart Johnston
CIS 127 – Intro to Information Security
NCMC
johnstons1@student.ncmich.edu

November 24, 2018

1 Domain Name System

DNS is a complicated procedure, much of which is outside the scope of this report. The basic principle is as such:

1. Numbers are hard for people to remember, by comparison with distinct names. DNS servers store the relationship between IP addresses and registered domain names.
2. Every machine needs to retrieve the IP address from a DNS server for a domain name when the relationship is not already known.
3. Many home users rely on their gateway (modem/router) to point them to a DNS server, which is often provided by the Internet Service Provider. Many organizations host their own DNS servers which perform the decentralized workings themselves, so as not to rely on outside resources which can't be controlled. They may also make this server the authoritative DNS server for their domain, and make it internet facing.
4. DNS servers don't retrieve the information fresh from other servers every time they need to serve requests; that would be monumentally noisy and slow. Instead, they cache results for some amount of time.
5. Malicious attackers can exploit the DNS protocol to point users towards a phony website which is designed with hostile intent. One such example is:
    - Lookalike sites built to scrape user credentials before logging them in. (If I were a threat, I would not want my threat discovered quickly. As such, I would do everything I could to hide the existence of my threat.)

2 Attacks

There are multiple kinds of attacks which can be leveraged on the DNS protocol. Generally, there are two broad methods, with varying degrees of longevity and effectiveness.

2.1 Man in the Middle

If the attacker and the victim are on the same LAN, some tools can be used to intercept and spoof a legitimate DNS response. The idea is to act quickly enough that the first DNS response the victim receives is from the attacker. However, for the attack to work consistently, the tool needs to keep listening for requests that the end-user machine makes. If the latency between the attacker and the victim is too high, and the real DNS responds first, the attack fails. If the attacker never hears the request to begin with, such as if the victim is using an encrypted VPN tunnel: no dice. This kind of attack could work well in situations where there is no locally hosted DNS server, or where latency to the server would be wide enough for the attacker to slip in. A cafe's wifi network, for instance, would probably be prime territory because:

- There may be many wireless devices (phones, especially) seeking the same websites, and making requests.
- It is unlikely that a cafe would be high-tech enough to have a locally hosted DNS server. The latency between phone `->` router `->` DNS server and back almost certainly leaves a wide enough opening for an attack.

The MitM technique serves as an enormous advertisement for VPN services and a reminder of the importance of checking certificates for important sites.

2.2 Poisoning the Cache

Dumping poison in the watering hole, here meaning the DNS server's cache, is very similar to the MitM attack in several ways, but with the following requirements/distinctions:

- The DNS server must be someplace such the attacker can insert their response between the local server and other name servers on the internet. It follows that this attack may work quite well against an organization with the technical chutzpah to host their own DNS server, and it would be a rather pointless endeavor in nearly any other situation.
- The response only need be spoofed *once* for any given domain, as long as the cache keeps the spoofed response. This may be minutes, hours, or days. This has an obvious benefit over needing to keep constant watch for requests after the initial attack.

  However, unless an attacker has their fingers in the DNS server, they must keep watch until the cache is flushed and a new request is made, which is not necessarily easy to know. Because the response can set the time that a particular domain is kept in the cache, targeted domains which have permanent IP addresses may not be as vulnerable.
- Authority and other domains can also be targeted. An attacker can point the DNS server to their own nameserver elsewhere for the duration that the cache keeps that record.

Like the MitM attack, this primarily is effective through use of a tool to listen for and respond to a request quickly enough to be taken as legitimate. However, if the DNS server is not secured properly, manual poisoning can be achieved to great effect, without having to wait for the cache to flush for any given target domain.

3 Conclusion

A more up-close-and-personal kind of attack than typosquatting, and almost certainly more dangerous, and difficult to detect, DNS hijacking is likely to be a vulnerability for any individual or organization, with only a handful of security countermeasures available. Being a more intimate kind of attack, the threat is liable to be significantly lower the smaller the gains. Grabbing someone's reddit credentials, for example, may not help an attacker much, but grabbing their social media, email, or online-banking information is incredibly valuable.

For individuals, some countermeasures include:

- Using encrypted tunnels to serve DNS requests, which may be as simple as a VPN.
- *Always* checking the browser to see if the legitimate certificate is being served. Certificate Authorities and HTTPS are what make such things as online banking a possibility in the first place; ignoring their importance is not a good idea.

For organizations, countermeasures for end-users are as those for individuals. End-users are generally not reliably security-conscious creatures, however, so the procedure the organization can follow includes (off the top of my head):

- Increasing the Time To Live of known-good records, and watching for inconsistencies in responses. If a response carries a TTL which is not consistent with previous responses for the same domain, flushing sooner may help reduce the attack window.
- Corroborating responses from other, decentralized DNS servers, and randomizing the ports in doing so.
- Following industry-standard security policy, most all of which is unknown to the author.