*UConn CSE 5095: Network Security, Fall 2024*

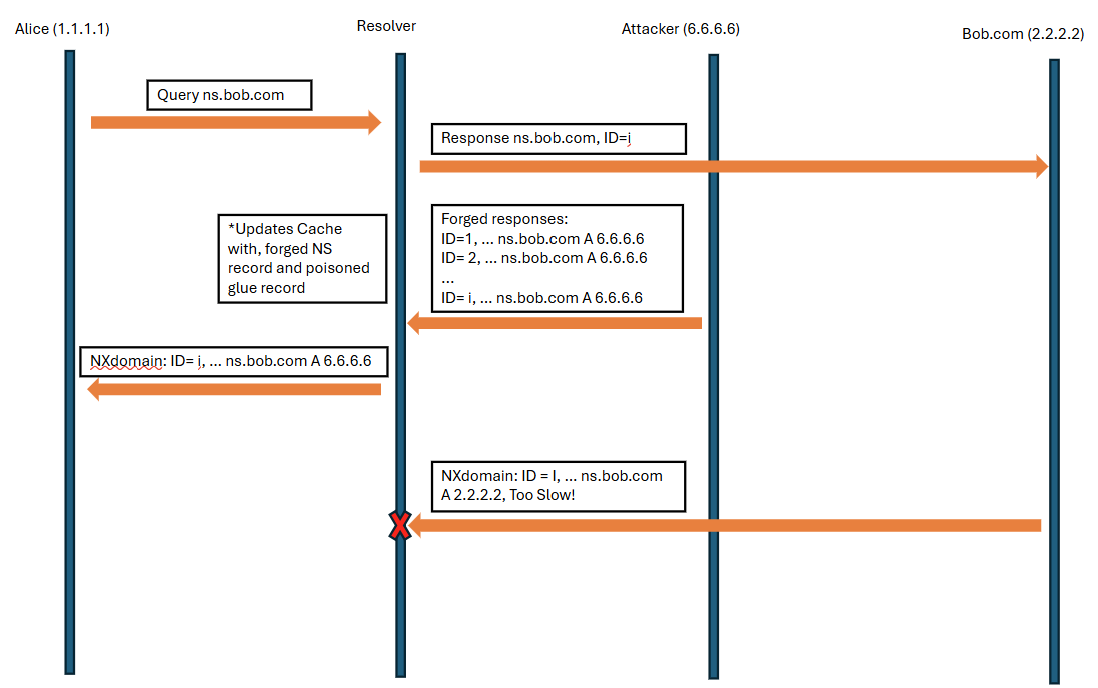
*HW 3: DNS scans, submit in Husky by Nov. 7th, 11:59pm*

*You may work in pairs. Printed solutions receive 10 points bonus. Prof. Amir Herzberg*

*Name: Luke Pepin*

*Exercise 1. We discussed that when a name server receives a query for a subdomain, they often respond with the NS record specifying the name server of the subdomain, and the ‘glue’ records, giving the IP address of that name server.*

1. *We discussed that this (and few other responses, e.g., CNAME) may be abused by (variants of) the Kaminski poisoning attack, and that, therefore, some resolvers will not trust these records bundled in the DNS response. Show, with a sequence diagram, how the attacker can poison a resolver that trusts an NS record and/or a glue record received in a response. You may assume that the source port used by the resolver is predictable.*



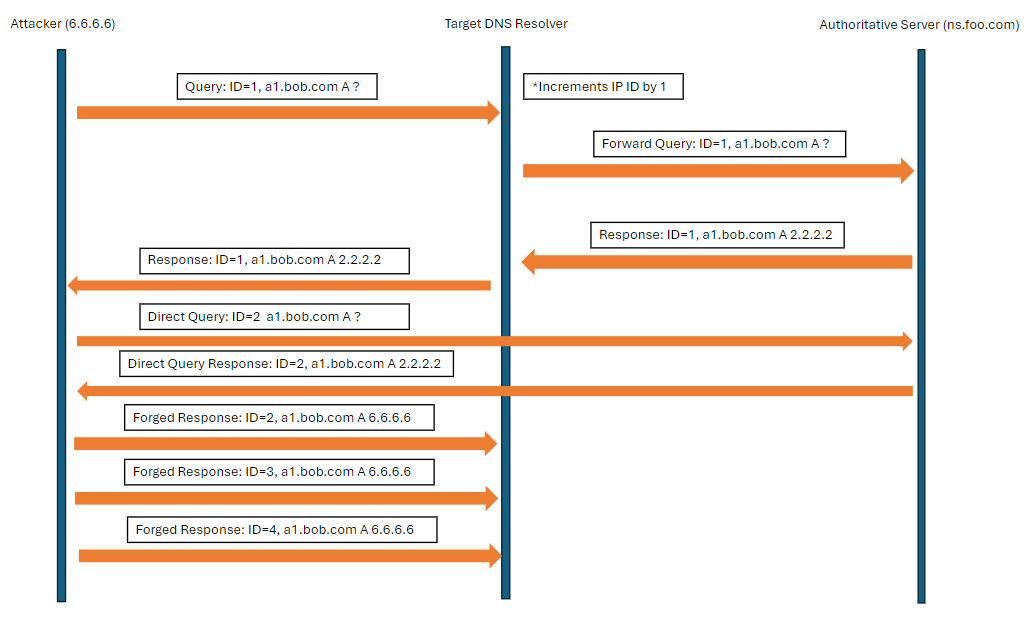
*Explanation:*

*The Kaminski poisoning attack exploits predictable source ports and transaction IDs to inject malicious DNS records into a resolver’s cache.*

* *The attack begins when the client (Alice) submits a DNS query for a1.bob.com to the resolver, which assigns a unique transaction ID (i) to track the query.*
* *The resolver then queries ns.bob.com for the authoritative answer.*
* *The attacker, predicting or brute-forcing the transaction ID, sends a forged response containing a spoofed NS record and a poisoned glue record with a malicious IP address (e.g., 6.6.6.6). The resolver fails to detect the spoofed data and caches the malicious information, which is then returned to the client as the address for a1.bob.com.*
* *The attacker continues by sending high volumes of queries for different subdomains (e.g., a2.bob.com, etc.), each time sending forged responses to poison the resolver’s cache further.*
* *Finally, when the resolver queries bob.com again, the attacker intercepts and sends a malicious IP (e.g., 6.6.6.5). The resolver caches this poisoned data, and any future queries for bob.com are directed to the attacker’s servers.*

*This attack allows the attacker to manipulate DNS responses and redirect traffic, exploiting the resolver’s trust in NS and glue records received in responses.*

1. *(Same as class quiz 3.) Present a sequence diagram for a scan which identifies open resolvers which are vulnerable (i.e., accept NS and/or glue, and use predictable source port). Explain the scan, and, in particular, how you checked if a resolver is vulnerable to accepting NS and/or glue and how the attacker can predict the source port (for resolvers that the scan found to be vulnerable).*



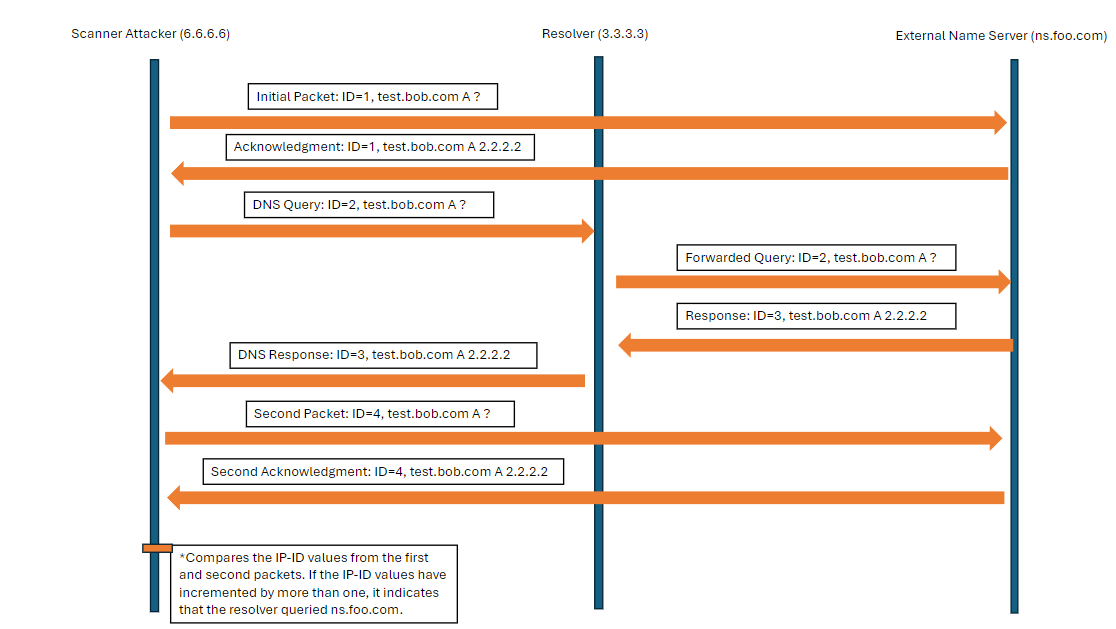
*Explanation:*

The Kaminski poisoning attack exploits predictable source ports and transaction IDs to inject malicious DNS records into a resolver’s cache.

* The attack begins when the attacker sends a DNS query for `a1.bob.com` to the target resolver.
* If the resolver is open, it forwards the query to the authoritative server, `ns.foo.com`.
* The resolver receives a response from `ns.foo.com` and increments its IP ID.
* To check the resolver's predictability, the attacker sends a direct query to `ns.foo.com` and observes the resolver's IP ID increment. If the IP ID is predictable, the attacker confirms the ability to spoof responses.
* Next, the attacker floods the resolver with spoofed responses, attempting to poison the cache by injecting malicious NS and glue records. These forged records associate `test.bob.com` with the attacker's malicious IP address (e.g., 6.6.6.6).
* The resolver fails to detect the spoofed data and caches the malicious information. Once the cache is poisoned, any future queries for `test.bob.com` (or other subdomains of `bob.com`) will resolve to the attacker’s malicious IP, redirecting traffic away from legitimate servers.

The attacker checks if the resolver accepts NS and glue records and uses predictable source ports. Owning the authoritative name server for bob.com, the attacker compares legitimate responses with spoofed ones to confirm the attack’s success. This lets the attacker manipulate DNS responses and redirect traffic by exploiting the resolver’s trust in these records.

*Exercise 2. In this question we investigate a stealthy scan for open DNS resolvers. This scan should not expose to the resolver either the IP address of the scanner, or any domain name controlled by the scanner. The scan can use the assumption that a given name server, e.g., ns.foo.com (for the foo.com domain), uses globally-incrementing IP-IDs; and it is Ok to send to ns.foo.com packets from the scanner’s source IP address. Present the scan as a sequence diagram.*



*Explanation:*

*The stealthy scan for open DNS resolvers involves sending packets to an external name server, such as ns.foo.com, to observe its globally-incrementing IP-ID values.*

* *The scanner first sends a packet to ns.foo.com to note the initial IP-ID.*
* *Then, the scanner sends a DNS query to the resolver for a domain not controlled by the scanner.*
* *The resolver forwards this query to ns.foo.com, which increments its IP-ID and sends a response back to the resolver.*
* *The resolver then sends the DNS response back to the scanner.*
* *The scanner sends another packet to ns.foo.com to observe the new IP-ID value.*
* *By comparing the IP-ID values from the first and second packets, the scanner can determine if the resolver queried ns.foo.com.*

*This method ensures that the scanner’s IP address and any domain controlled by the scanner are not exposed to the resolver, meeting the exercise’s requirements. The use of globally-incrementing IP-IDs from ns.foo.com allows the scanner to infer the resolver’s behavior without direct interaction, maintaining stealth.*