Cache Poisoning Attacks

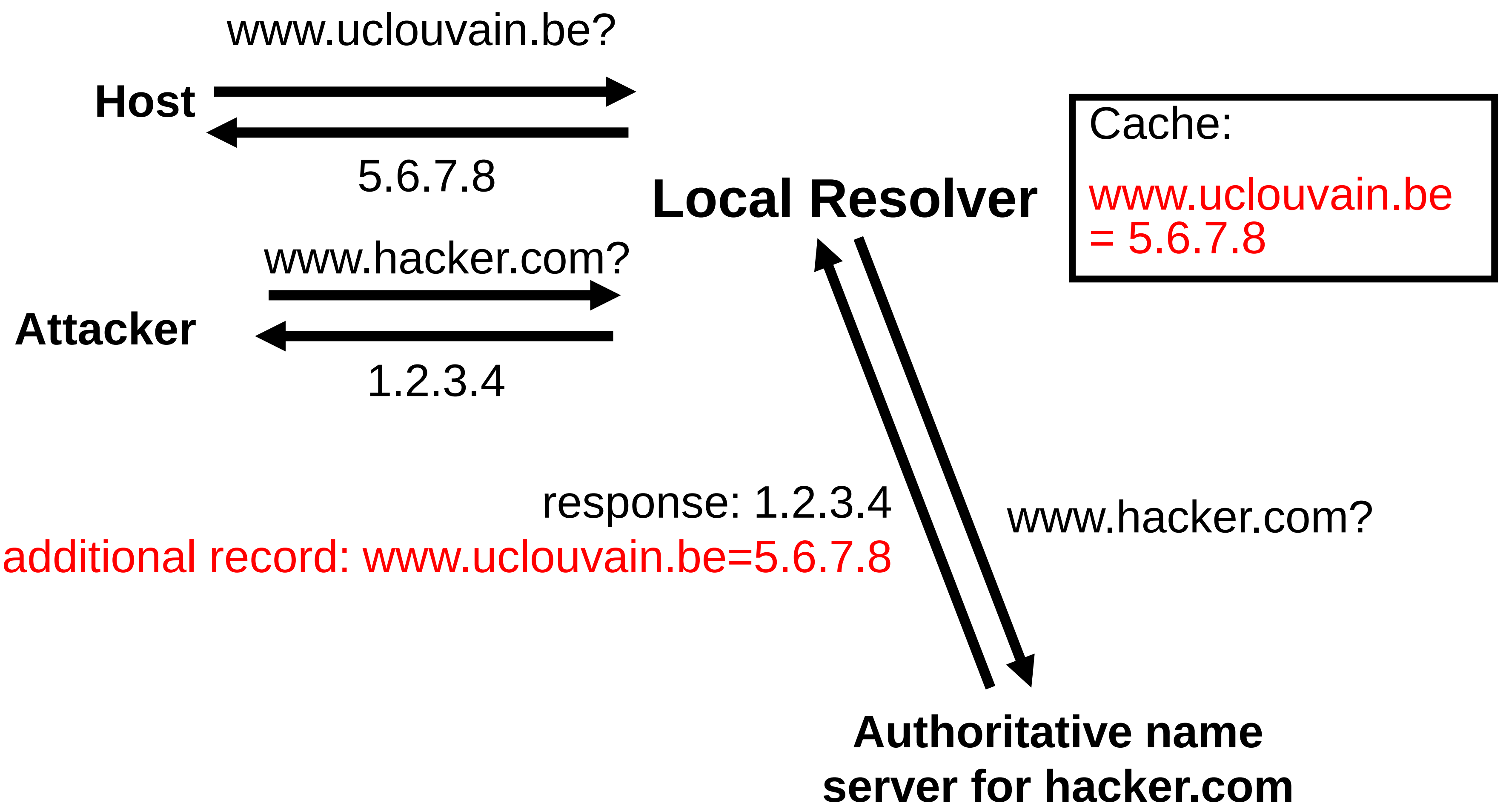
Ramin Sadre

Cache Poisoning: Variant 1

- Appeared first around 2008
- In the past, a DNS server could not only send the requested information for the specified domain but also “additional records”

  = information about domains not requested
- Can be misused by attackers by setting up a malicious authoritative name server for a foreign domain

Cache Poisoning: Variant 1 (2)

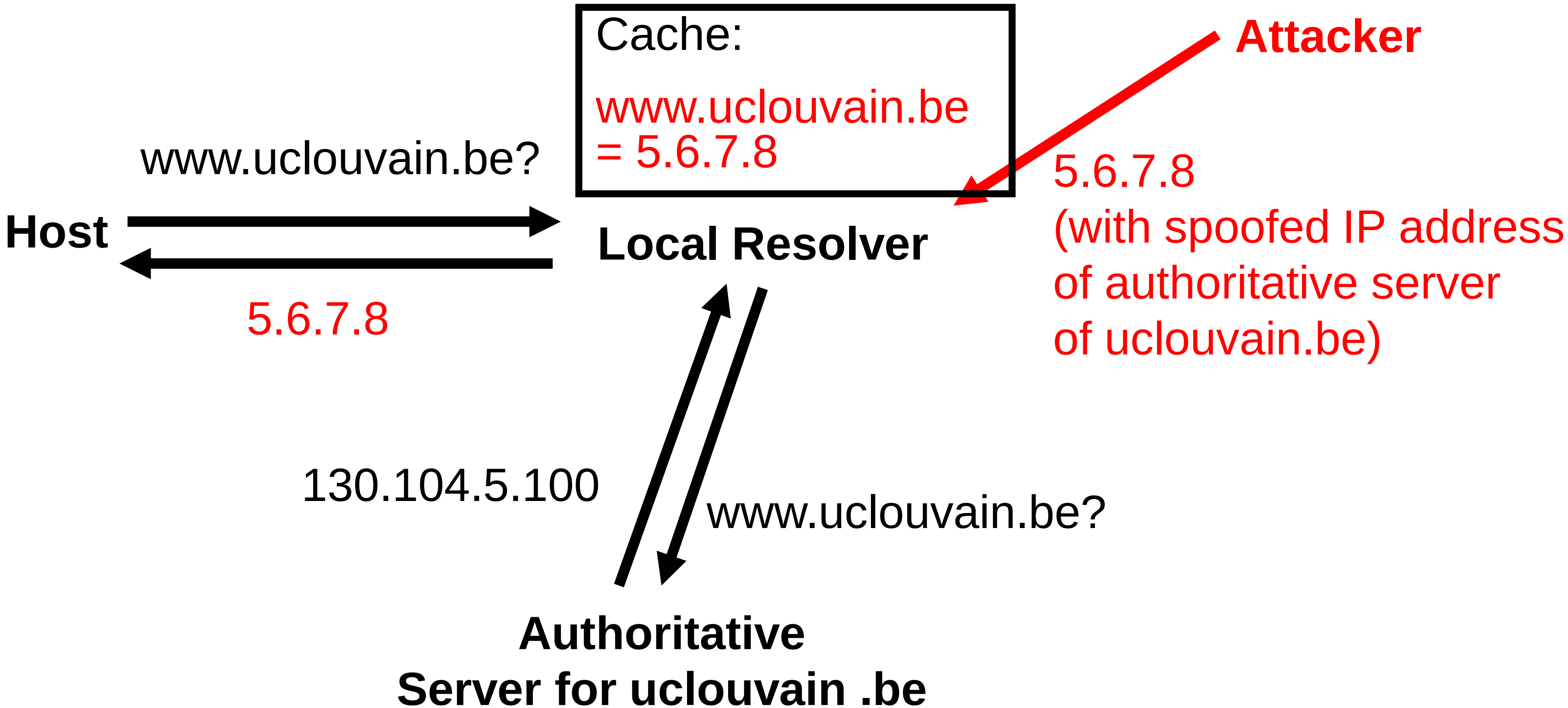


Cache Poisoning: Variant 1 (3)

- This attack does not work anymore (hopefully)
- Modern local resolvers do a *Bailiwick Check*:
  - Local resolvers accept additional records only if they contain information about the same domain as the request

Cache Poising: Variant 2

- In this variant, the attacker sends a fake response with the spoofed IP address of the authoritative server



Cache Poising: Variant 2 (2)

- IP spoofing is possible because DNS uses UDP: no connection (no sequence numbers etc.)
- Timing important:
  - Fake answer has to arrive at the Local Server before the real answer. Can be achieved by sending many fake answers, hoping that one will succeed.
  - In that case, the Local Server will ignore the second (real) answer, assuming that it is a mistake

- This sounds too good to be true. Is DNS so easy to break?

Cache Poising: Variant 2 (3)

- Query IDs in DNS
  - Each query to a DNS server contains a 16-bit query ID
  - The response from the DNS server uses the same ID
  - A DNS client will only accept a response if its ID matches the query ID
- In addition:
  - The 16-bit port number in the UDP packet has to match

- For a successful attack, the attacker has to guess correctly the port number and the query ID in the fake response
- Sounds impossible. Around 2^{32} possible combinations!

Cache Poising: Variant 2 (4)

- This attack worked in the past because
  - DNS resolvers used the same source port for all queries
  - Query IDs were predictable (1,2,3,4,5,…) or were using bad random-number generators
- Mitigation: New DNS software uses
  - Random port numbers
  - (Better) Random query IDs

Why Cache Poisoning?

- Victim's traffic is directed to a different host controlled by the attacker
- If the attacker forwards the traffic to the original destination, victim will not notice
  - Attacker can inspect victim's traffic: spy messages,...
  - Attacker can present a fake website (phishing): steal passwords, credit card numbers,...
  - Attacker can modify the traffic before forwarding it
- Can be also used for DoS attack
  - If traffic is not forwarded, the victim cannot use the network anymore

Summary

- Cache poisoning attacks work (or worked in the past) because
  1. Resolvers accepted additional records without checking
  2. Spoofing possible
  3. Bad randomization of transaction parameters

- Underlying problem: Resolvers cannot verify the identity of the authoritative servers
  - DNSSEC tries to solve this problem by introducing digital signatures and certificates (similar to HTTPS)
  - Although introduced many years ago, DNSSec is not yet widely used. Good news: Today, all original TLDs and the TLDs of most large countries support DNSSEC

ARP Cache Poisoning (or ARP Spoofing)

- Cache poisoning also works for other types of caches where the cache does not verify the identity and authority of the source of the data
- Example: ARP Cache poisoning
  - ARP = “DNS for Ethernet addresses”
    1. Client asks:
       “Ethernet address of 1.2.3.4?”
    2. Anybody who knows the answer can reply:
       “1.2.3.4 has Ethernet address 01:02:03:04:05:06”
    3. Answers are cached locally in the client
  - ARP clients also accept unsolicited responses!