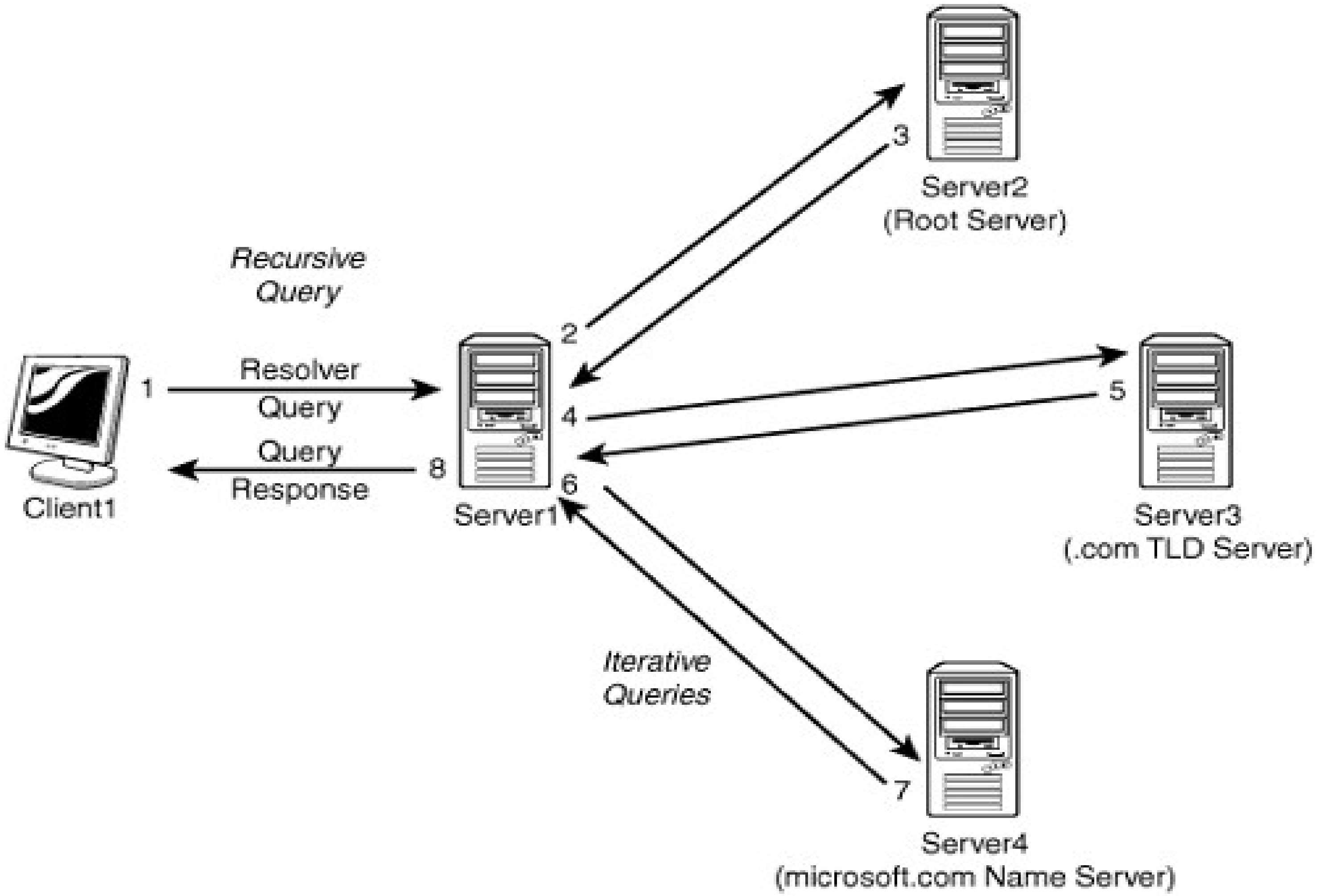
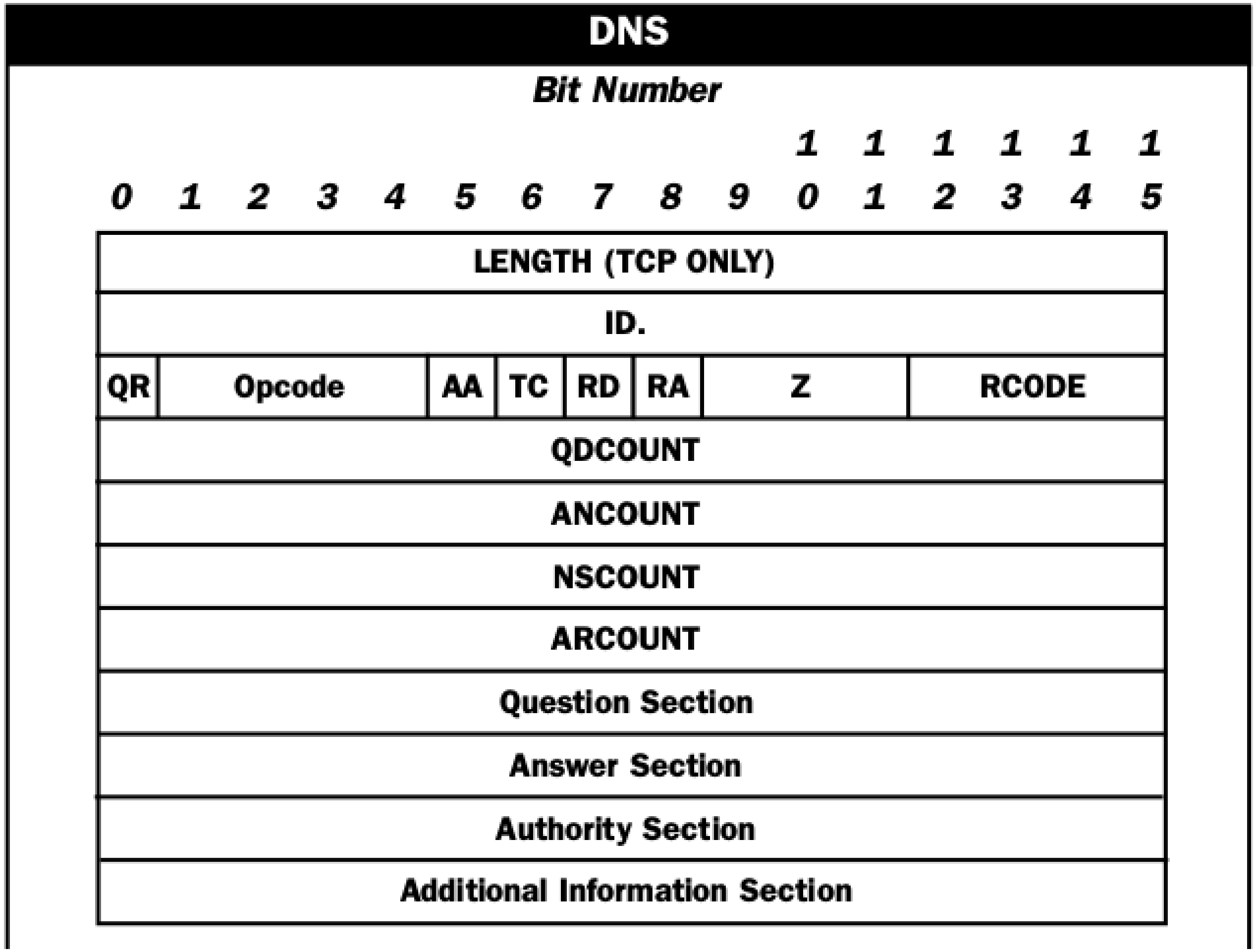
DNS
Cache
Poisoning

DNS Lookup



Recursive Query
Resolver Query
Query Response
1
8
Client1
Server1
2
3
4
5
6
7
Server2
(Root Server)
Server3
(.com TLD Server)
Iterative Queries
Server4
(microsoft.com Name Server)

DNS Header



DNS

Bit Number

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

LENGTH (TCP ONLY)

ID.

QR	Opcode	AA	TC	RD	RA	Z	RCODE

QDCOUNT

ANCOUNT

NSCOUNT

ARCOUNT

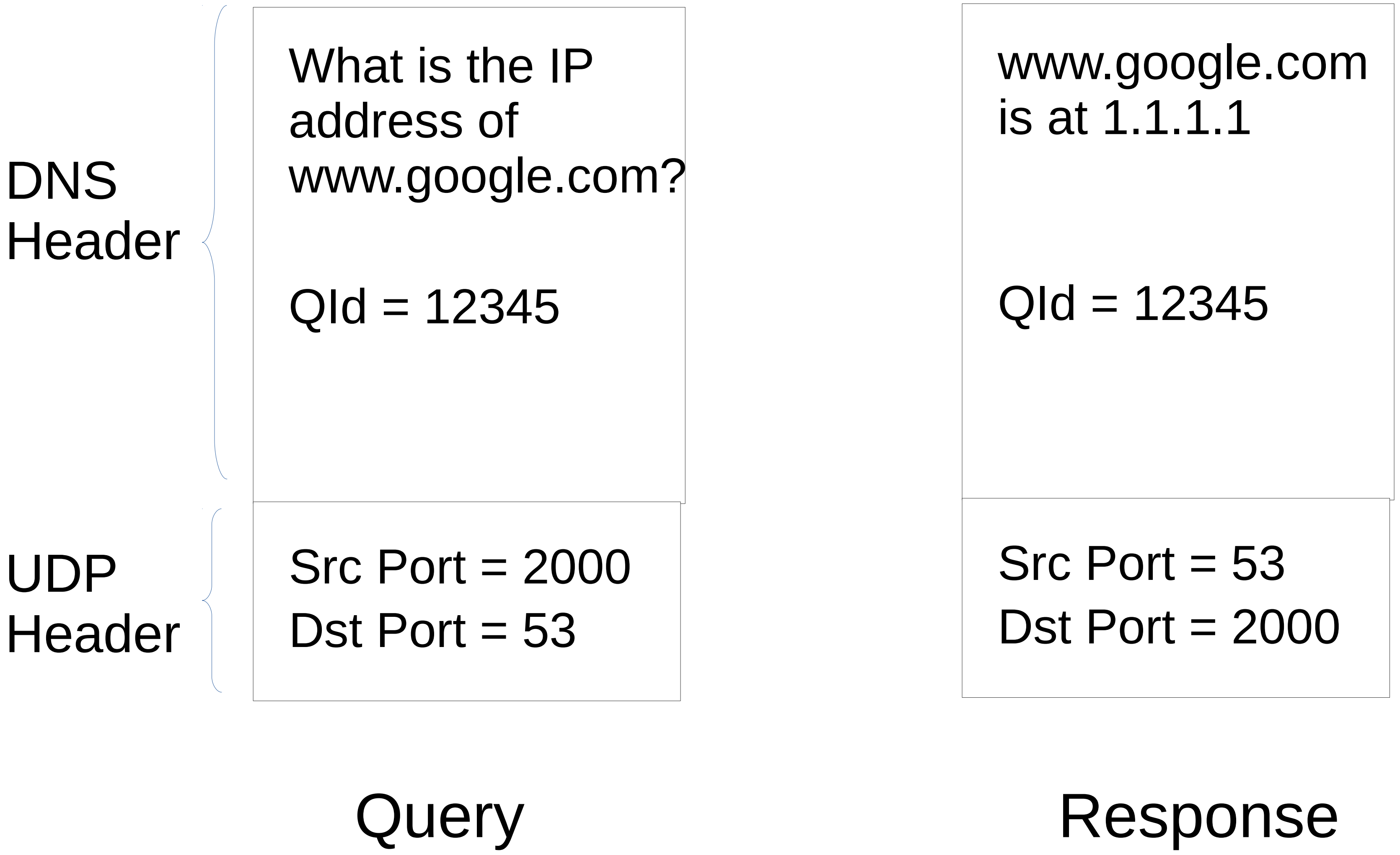
Question Section

Answer Section

Authority Section

Additional Information Section

DNS Query and Response



DNS Header

UDP Header

What is the IP address of www.google.com?

QId = 12345

Src Port = 2000

Dst Port = 53

Query

www.google.com is at 1.1.1.1

QId = 12345

Src Port = 53

Dst Port = 2000

Response

No Authentication

- Responses are not authenticated to queries.
- The only checks are:

  (1) the source IP address and destination port of the response must match, respectively, the destination IP address and source port of the query.

  (2) the 16-bit Query ID (QId) of the response must match that of the query

No Authentication

(3) The Question section (which is duplicated in the reply) matches the Question in the pending query

(4) The Authority and Additional sections represent names that are within the same domain as the question: this is known as "bailiwick checking".

> This prevents ns.google.com from replying with not only the IP address of www.google.com, but also fraudulent information about (say) aib.ie.

- First arriving UDP packet which satisfies these conditions is accepted
- On some servers, if another arrives within 1 second, it is accepted

DNS Cache

- When the DNS Response is accepted, it is recorded in the DNS cache for a time specified by the TTL
  - can be a short as a few minutes, or as long as a week or more

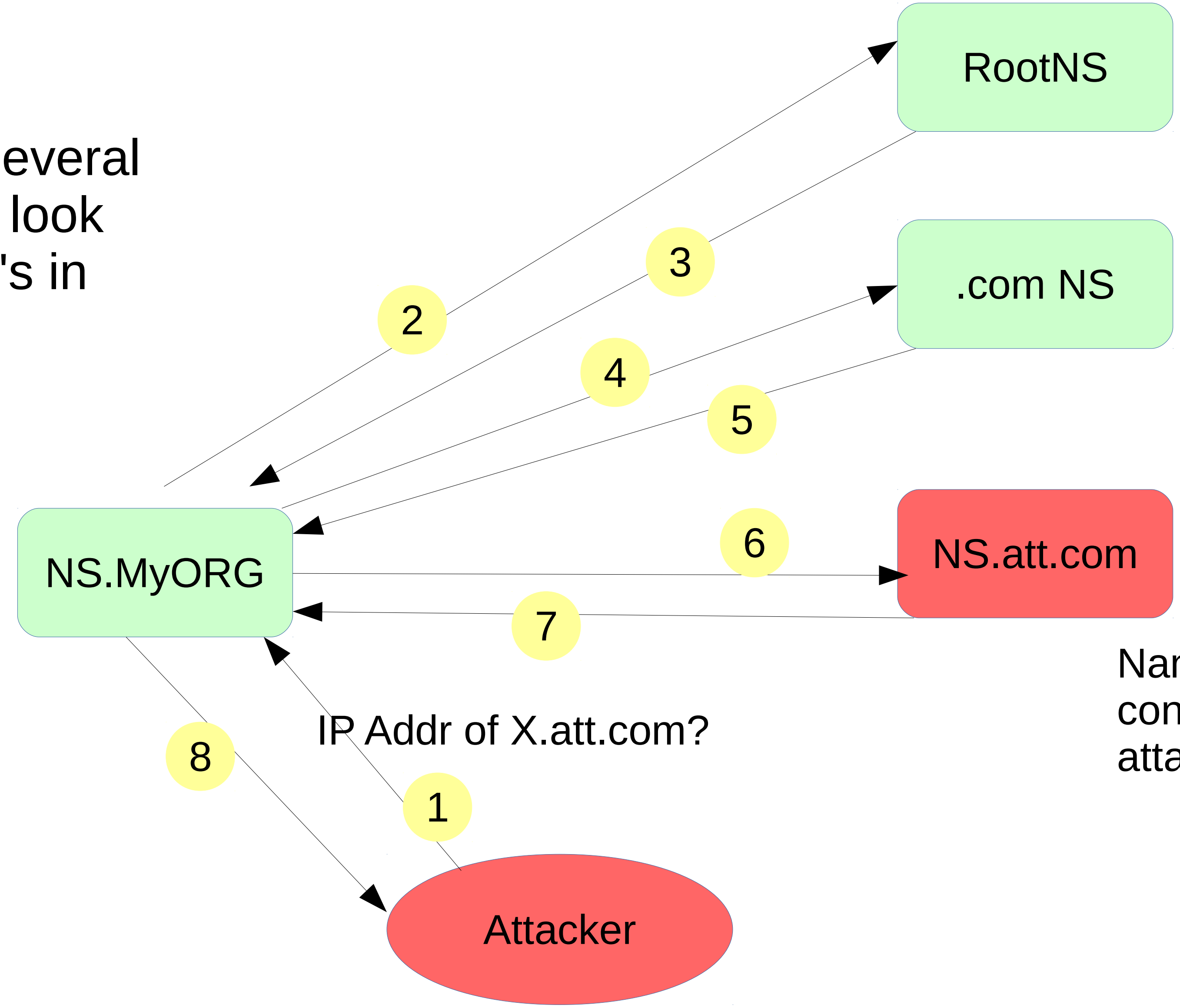
Fixed Port

- Prior to patches applied around 2008, most DNS resolvers used a fixed port to send queries.

Attack 1

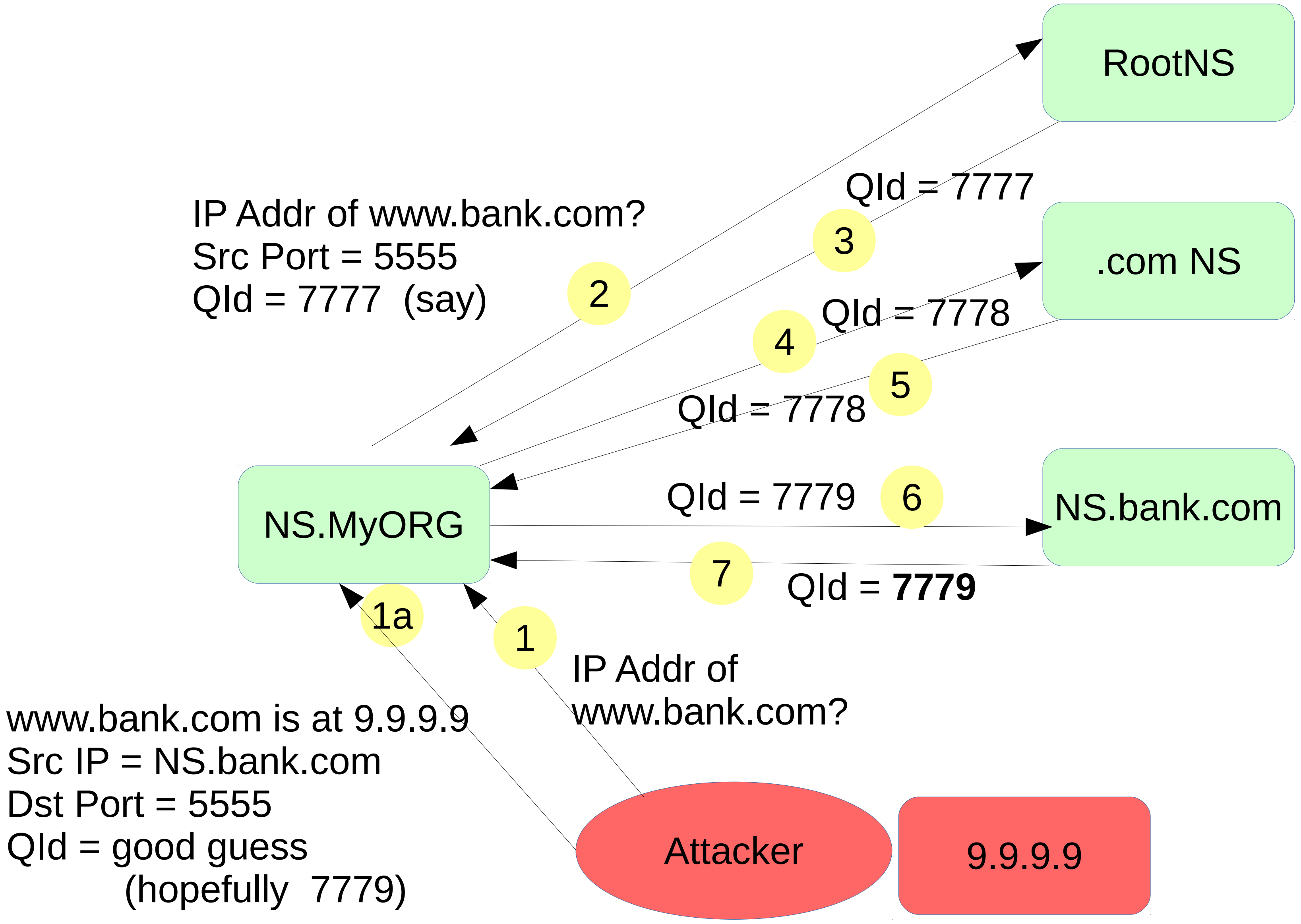
Sample the Query ID

Run this several times and look at the QId's in step 6



Nameserver controlled by attacker

Perform the Attack



IP Addr of www.bank.com?
Src Port = 5555
QId = 7777 (say)

QId = 7777

QId = 7778

QId = 7778

QId = 7779

QId = **7779**

IP Addr of
www.bank.com?

www.bank.com is at 9.9.9.9
Src IP = NS.bank.com
Dst Port = 5555
QId = good guess
(hopefully 7779)

Caveats

- The name can't already be in the cache
  - If so, there is no way to poison it in this manner.
  - The attacker has to wait for it to expire from cache (as determined by the TTL).
- The attacker has to guess the query ID
  - This was made easy because (now-obsolete) nameservers used to increment the Query ID by one each time
- The attacker has to be faster than the real nameserver
  - If the real nameserver wins, the correct DNS mapping will be recorded for the TTL

Source:http://unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html

Solution (around 2004-5)

Randomize the Query ID

Proper randomization

Attack 2

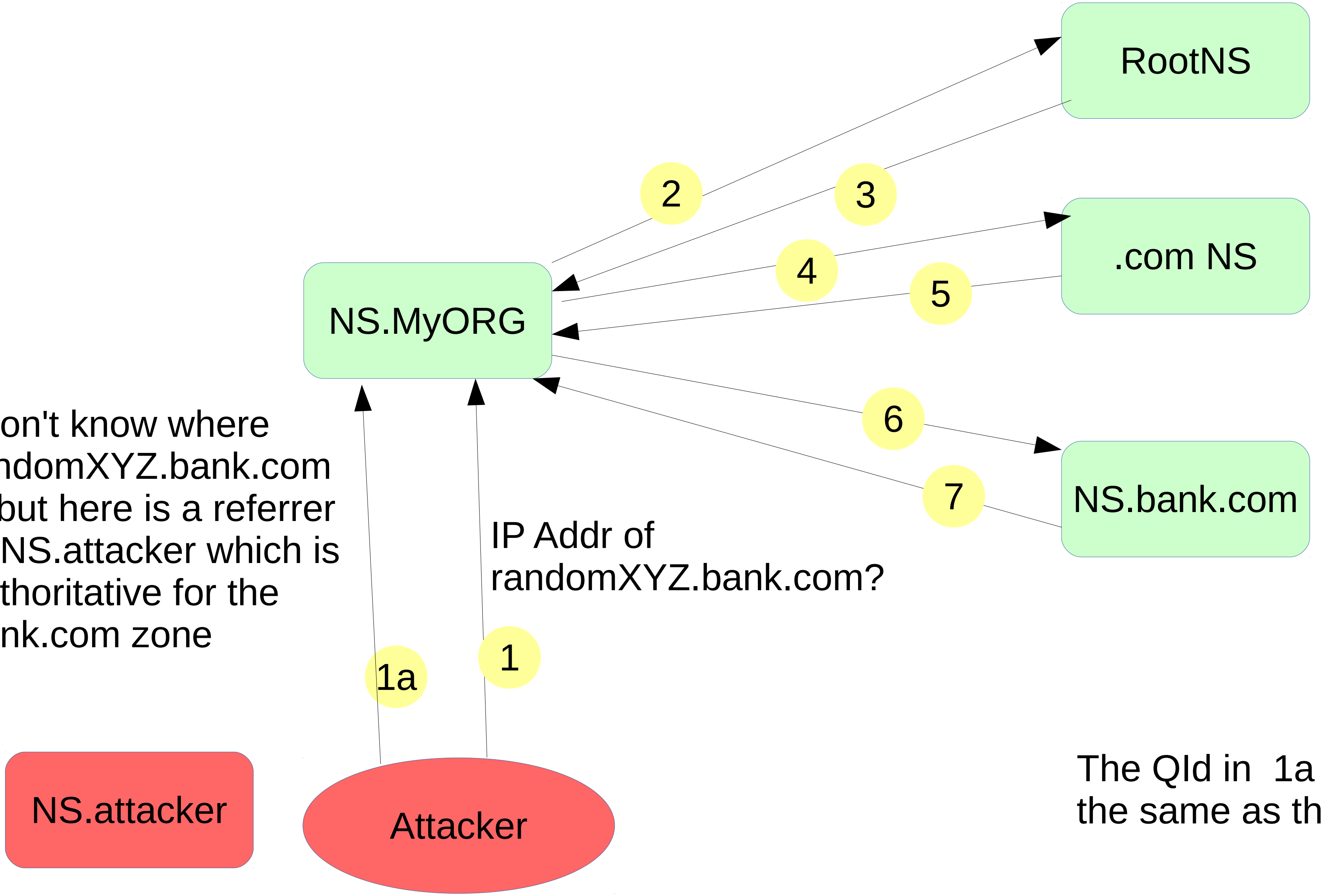
Kaminsky's Attack

- Attacker configures a nameserver that is authoritative for the bank.com zone, including whatever resource records he likes: A records, MX records, etc.
  - There's nothing stopping anybody from configuring his own nameserver to be authoritative for any domain, but it's pointless because the root servers won't point to it
  - it's got answers, but nobody ever asks it a question.

Kaminsky's Attack

- Attacker sends a request for randomXYX.bank.com
- Nameserver starts recursive search
- Eventually NS.bank.com replies with “I don't know”
- But the attacker spoofs that reply with a referrer: “I don’t know where randomXYX.bank.com is, but here is a referral to a nameserver that is authoritative for bank.com”

Perform the Attack

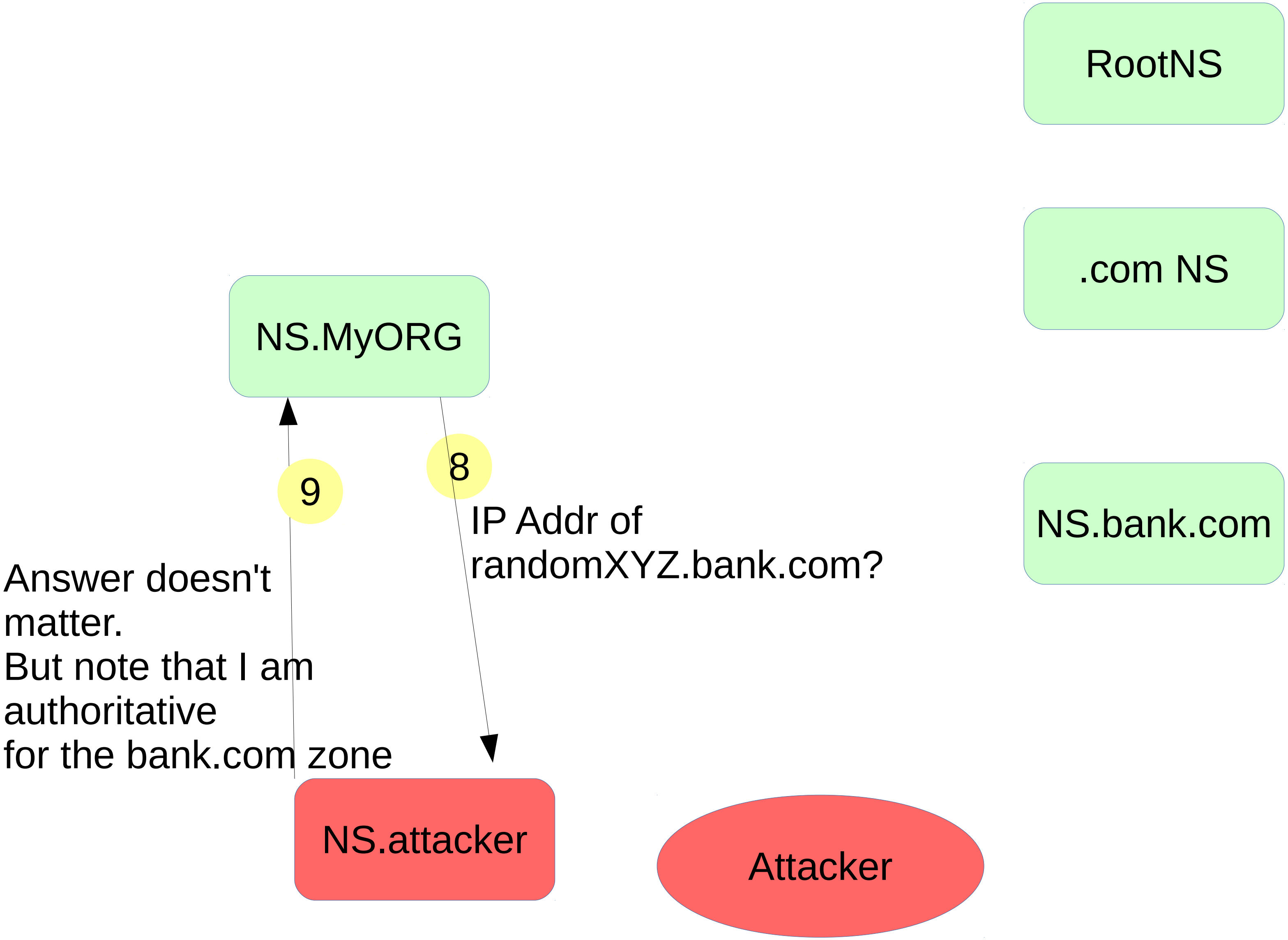


I don't know where randomXYZ.bank.com is but here is a referrer to NS.attacker which is authoritative for the bank.com zone

IP Addr of randomXYZ.bank.com?

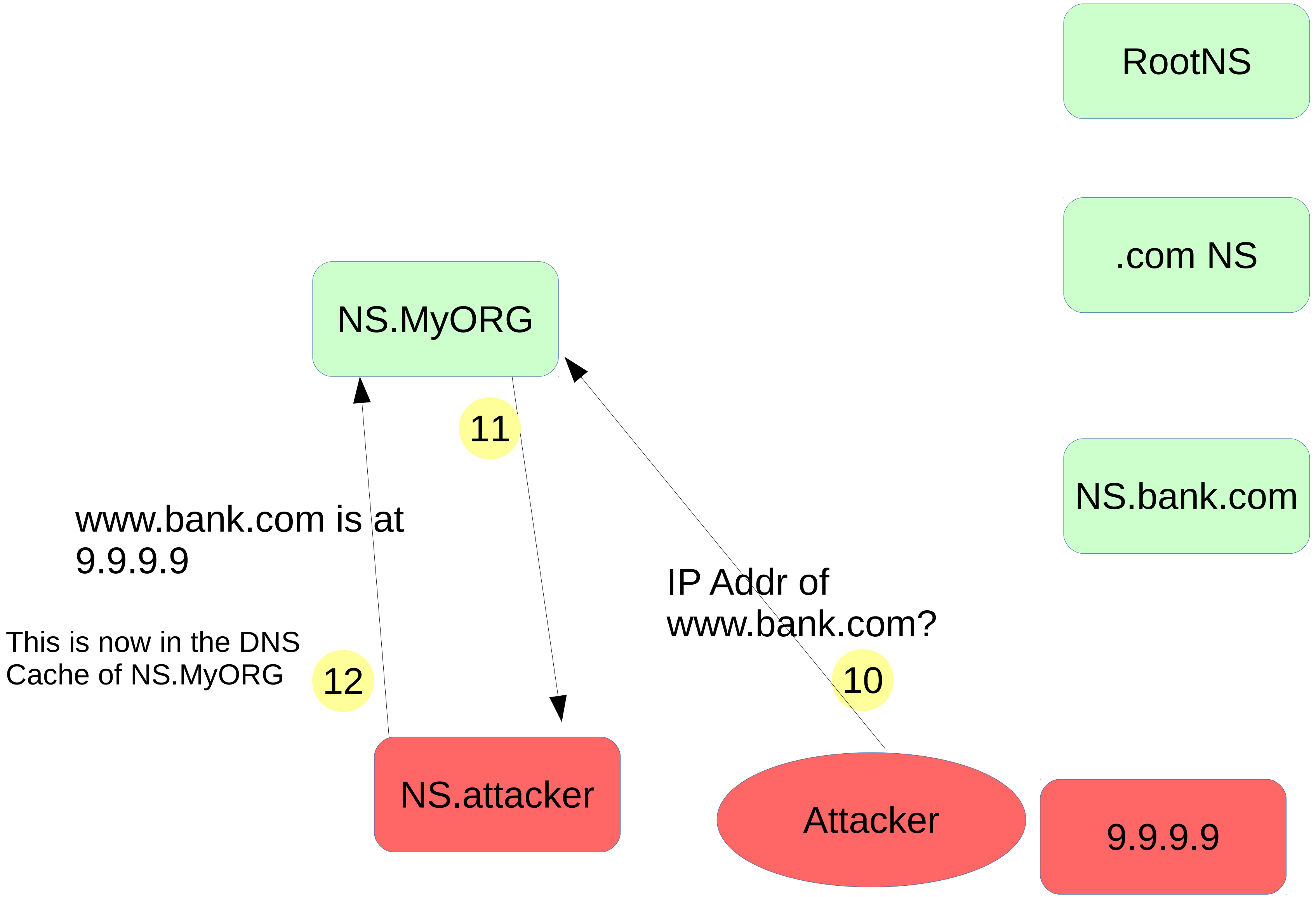
The QId in 1a must be the same as that in 6

Game Over



bank.com NS IN NS.attacker

Final Step


RootNS
.com NS
NS.MyORG
NS.bank.com
11
www.bank.com is at
9.9.9.9
IP Addr of
www.bank.com?
This is now in the DNS
Cache of NS.MyORG
12
10
NS.attacker
Attacker
9.9.9.9

Solution (around 2008)

Randomize the Port Number also

Entropy

$$2^{16} \quad * \quad 2^{11}$$

Source:http://unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html

Even More Entropy

Www.baNK.cOm

Some Recent Approaches

Reply Rate Limiting

- Modern DNS servers (e.g. BIND 9) rate limit how many responses a DNS server will send. If the limit is reached, the DNS server may either not respond at all, or reply with an empty truncated reply.
- If the attacker floods an authoritative DNS server to prevent it from sending responses it will provide more time to send spoofed responses back.
- Researchers have shown that this can lead to DNS Cache Poisoning.
- requires a lot of packets (100 MBit for 8 hours) to be successful, as the Query ID and the source port needs to be brute forced.

Source:https://isc.sans.edu/diary/New+tricks+that+may+bring+DNS+spoofing+back+or%3A+%22Why+you+should+enable+DNSSEC+even+if+it+is+a+pain+to+do%22/16859

EDNS0

- Originally, DNS replies were limited to 512 bytes to avoid fragmentation.
- But, modern DNS tends to use larger replies with IPv6 and DNSSEC records, as well as the use of DNS for load balancing.
- In response, EDNS0 was introduced.
- If enabled, the DNS server may signal a maximum response size that is typically 4096 bytes.
  - As a result, these responses are frequently fragmented.

Source:https://isc.sans.edu/diary/New+tricks+that+may+bring+DNS+spoofing+back+or%3A+%22Why+you+should+enable+DNSSEC+even+if+it+is+a+pain+to+do%22/16859

DNS Cache Poisoning based on Fragmentation

- The server that issued the query uses EDNS0
- The response is fragmented
- Only the first fragment includes the items needed to authenticate the response:
  - the UDP port, the answer and the DNS Query-ID
- The attacker injects a spoofed 2nd fragment
- (s)he needs to get the fragment offset and fragment ID correct

Source:https://isc.sans.edu/diary/New+tricks+that+may+bring+DNS+spoofing+back+or%3A+%22Why+you+should+enable+DNSSEC+even+if+it+is+a+pain+to+do%22/16859

DNS Cache Poisoning based on Fragmentation

- The fragment offset can be guessed assuming that the MTU is 1500 bytes .
- The fragment ID (or IP ID) is frequently incremented from packet to packet, so it can be easily guessed.
  - Even if it is random, it is still only 16 bit long.

Source:https://isc.sans.edu/diary/New+tricks+that+may+bring+DNS+spoofing+back+or
%3A+%22Why+you+should+enable+DNSSEC+even+if+it+is+a+pain+to+do%22/16859

Solution

- DNSSEC
- All DNS packets signed.
- Needs trusted authorities

References

http://dankaminsky.com/2008/07/24/details/

https://www.ietf.org/mail-archive/web/dnsop/current/pdf2jgx6rzxN4.pdf

http://unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html

Who am I?

Vincent.ryan@cit.ie

@vincentrya